OETCS/WP7/O7.1.3_O7.1.7 – 00/02

openETCS

ITEA2
INFORMATION TECHNOLOGY FOR EUROPEAN ADVANCEMENT

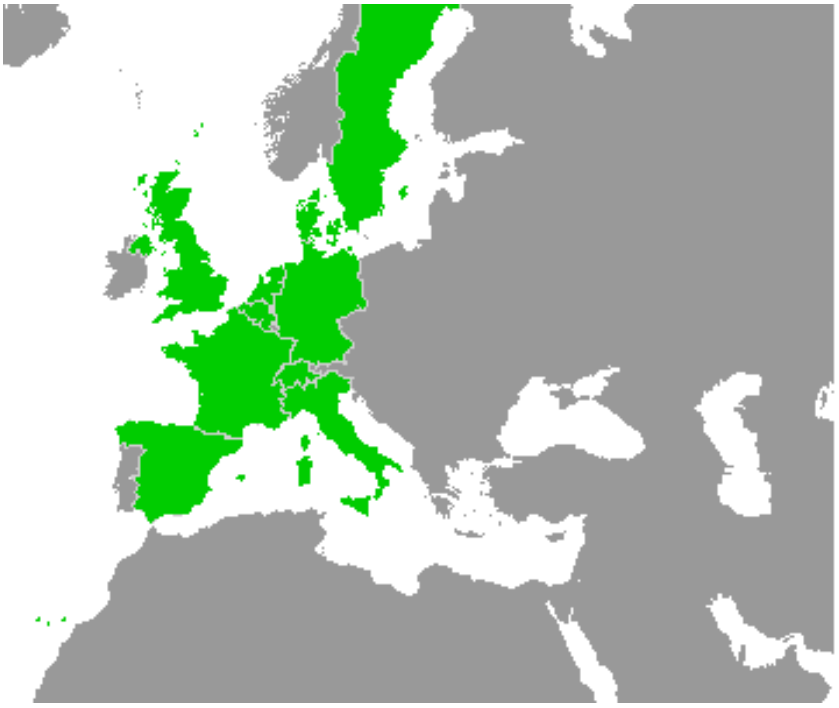
ITEA2 Project
Call 6 11025
2012 – 2015

Work-Package 7: “Primary tool chain”

Evaluation of the models and tools against the WP2 requirements

List of criteria on means, models and tools and results on the benchmark

Marielle Petit-Doche May 2013



This page is intentionally left blank

Work-Package 7: "Primary tool chain" **OETCS/WP7/O7.1.3_O7.1.7 – 00/02**
May 2013

Evaluation of the models and tools against the WP2 requirements

List of criteria on means, models and tools and results on the benchmark

Marielle Petit-Doche

Systerel

Definition

Prepared for ITEA2 openETCS consortium
Europa

Abstract: This document gives elements to evaluate the means of modeling and the associated tools according WP2 requirements. Evaluation on the models and tools of benchmark is also described.

Table of Contents

Figures and Tables

Figures

Tables

Document information	
Work Package	WP7
Deliverable ID or doc. ref.	O7.3.1_O7.1.7
Document title	Evaluation of the models and toolsagainst the WP2 requirements
Document version	00.02
Document authors (org.)	Marielle Petit-Doche (Systerel)

Review information	
Last version reviewed	00.01
Main reviewers	Uwe Steinke (Siemens) Armand Nachef (CEA) Cyril Cornu (All4Tech) Alexandre Ginisty (All4Tech) Mathieu Perrin (CEA)

Approbation			
	Name	Role	Date
Written by	Marielle Petit-Doche	WP7-T7.1 Sub-Task Leader	
Approved by	Michael Jastram	WP7 leader	

Document evolution			
Version	Date	Author(s)	Justification
00.01	19/04/2013	M. Petit-Doche	Document creation by merging O7.1.3 and O7.1.7
00.02	02/05/2013	M. Petit-Doche	Review remarks Tool evaluation matrix

1 Introduction

The aim of this document is to report the results of the evaluation of the means of description to model the requirements of SUBSET-026 concerning the on-board unit and their associated tools.

This evaluation task is part of work package WP7, task 1 "Primary tool Chain analyses and recommendations". According to the results of WP2, especially the OpenETCS process and the requirements on language and tools, the aim of this task is to determine the best candidates to produce models of the on-board units, following the OpenETCS process

This process is described in details in D2.3 " Description of the openETCS process" and is summed up in the figure 1. Requirements references quoted in the current document are defined in D2.6 "Requirements for openETCS".

Yellow elements are inputs, blue elements are part of the design process, red elements are verification and validation activities, green elements are safety activities. Each line (between dash or full blue lines) is a phase of the process, with a name on the right.

The chapter 2 of this document provides a template to describe the means and tools and a list of criteria according WP2 requirements on language, models and tools. The objectives of this description and criteria are to allow to determine the best means of description and associated tool for a given activities.

The chapter 3 resumes the results of the evaluation at the end of the benchmark activities.

In Appendix, a chapter is dedicated to each models produced during the benchmark activities :

- CORE
- GOPRR
- ERTMSFormalSpecs
- SysML with Papyrus
- SysML with Entreprise Architect
- SCADE
- EventB with Rodin
- Classical B with Atelier B
- Petri Nets
- System C
- GNATprove

For each approach and tool, the initial author of the evaluation is the partner in charge of the modelling. Two assessors, for each approaches, are in charge of the review of the evaluation and can correct it or add comments.

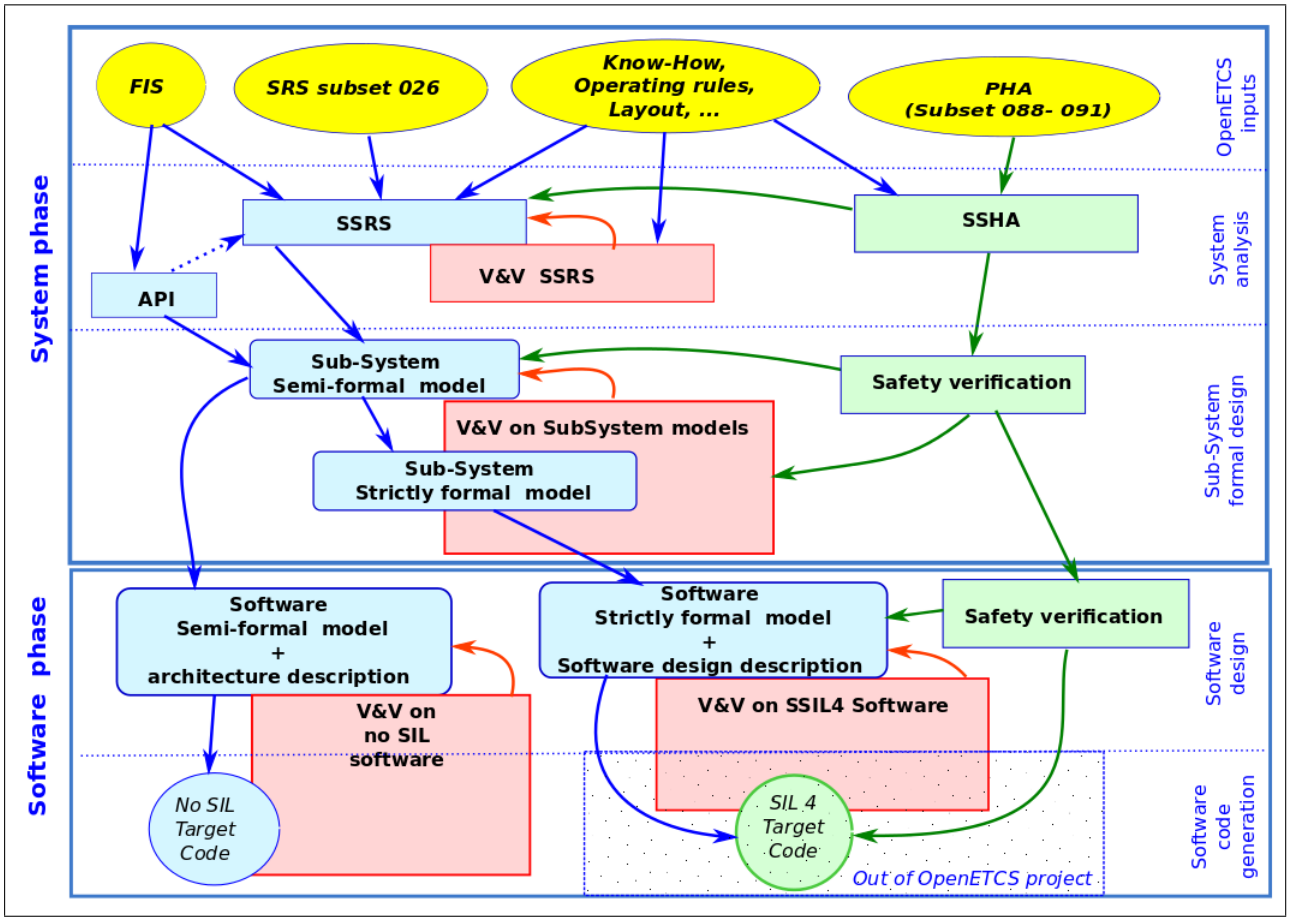


Figure 1. Main OpenETCS process

Tool platform are not covered by this document but in an other output of WP7 : O7.1.9 "Evaluation of each tool platform against WP2 requirements, independent of target tools". Besides, Task 7.1 is focussing on design activities : despite that some means can provide verification artefacts for example, tools and means for validation, verification, test generation,... are in the scope of task 2 and will be analysed later.

1.1 Reference Documents

- CENELEC EN 50126-1 — 01/2000 — *Railways applications — The specification and demonstration of Reliability, Availability, Maintainability 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*
- D2.1 – Report on existing methodologies
- D2.2 – Report on CENELEC standards
- D2.3 – Definition of the overall process for the formal description of ETCS and the rail system it works in
- D2.4 – Definition of the methods used to perform the formal description
- D2.6 – Requirements for OpenETCS

1.2 Glossary

API Application Programming Interface

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

FIS Functional Interface Specification

HW Hardware

I/O Input/Output

OBU On-Board Unit

PHA Preliminary Hazard Analysis

QA Quality Analysis

RBC Radio Block Center

RTM RunTime Model

SIL Safety Integrity Level

SRS System Requirement Specification

SSHA Sub-System Hazard Analysis

SSRS Sub-System Requirement Specification

SW Software

THR Tolerable Hazard Rate

V&V Verification & Validation

2 Templates

Author Author of the approaches description %%Name - Company%%

Assessor 1 First assessor of the approaches %%Name - Company%%

Assessor 2 Second assessor of the approaches %%Name - Company%%

In the sequel, main text is under the responsibilities of the author.

Author. Author can add comments using this format at any place.

Assessor 1. First assessor can add comments using this format at any place.

Assessor 2. Second assessor can add comments using this format at any place.

When a note is required, please follow this list :

0 not recommended, not adapted, rejected

1 weakly recommended, adapted after major improvements, weakly rejected

2 recommended, adapted (with light improvements if necessary) weakly accepted

3 highly recommended, well adapted,strongly accepted

***** difficult to evaluate with a note (please add a comment under the table)

All the notes can be commented under each table.

2.1 Presentation

This section gives a quick presentation of the approach and the tool.

Name %%Name of the approach and the tool%%

Web site %%if available, how to find information%%

Licence %%Kind of licence%%

Abstract

Short abstract on the approach and tool (10 lines max)

Publications

Short list of publications on the approach (5 max)

2.2 Main usage of the approach

This section discusses the main usage of the approach.

According to the figure 1, for which phases do you recommend the approach (give a note from 0 to 3) :

	Author	Assessor 1	Assessor 2	Total
System Analysis				
Sub-system formal design				
Software design				
Software code generation				

According to the figure 1, for which type of activities do you recommend the approach (give a note from 0 to 3) :

	Author	Assessor 1	Assessor 2	Total
Documentation				
Modeling				
Design				
Code generation				
Verification				
Validation				
Safety analyses				

Known usages

Have you some examples of usage of this approach to compare with the OpenETCS objectives ?

2.3 Language

This section discusses the main element of the language.

Which are the main characteristics of the language :

	Author	Assessor 1	Assessor 2	Total
Informal language				
Semi-formal language				
Formal language				
Structured language				
Modular language				
Textual language				
Mathematical symbols or code				
Graphical language				

According WP2 requirements, give a note for the capabilities of the language (from 0 to 3) :

	Author	Assessor 1	Assessor 2	Total
Declarative formalization of properties (D.2.6-X-28)				
Simple formalization of properties (D.2.6-X-28.1)				
Scalability : capability to design large model				
Easily translatable to other languages (D.2.6-X-30)				
Executable directly (D.2.6-X-33)				
Executable after translation to a code (D.2.6-X-33) (precise if the translation is automatic)				
Simulation, animation (D.2.6-X-33)				
Easily understandable (D.2.6-X-27)				
Expertise level needed (0 High level, 3 few level)				
Standardization (D.2.6-X-29)				
Documented (D.2.6-X-29)				
Extensible language (D.2.6-01-28)				

Documentation

Describe how the language is documented, the existing guidelines, coding rules, standardization...

Language usage

Describe the possible restriction on the language

2.4 System Analysis

This section discusses the usage of the approach for system analysis. It can be skipped depending the results of L.8.

Acoording WP2 requirements, how the approach can be involved for the sub-system requirement specification ?

	Author	Assessor 1	Assessor 2	Total
Independent System functions definition (D.2.6-X-10.2.1)				
System architecture design (D.2.6-X-10.2)				
System data flow identification (D.2.6-X-10.2.3)				
Sub-system focus (D.2.6-X-10.2.4)				
System interfaces definition (D.2.6-X-10.2.5)				
System requirement allocation (D.2.6-X-10.3)				
Traceability with SRS (D.2.6-X-10.5)				
Traceability with Safety activities (D.2.6-X-11)				

2.5 Sub-System formal design

This section discusses the usage of the approach for sub-system formal design. It can be skipped depending the results of L.8.

Two kinds of model can be planned during this phase: semi-formal models to cover the SSRS (D.2.6-X-12.1) and strictly formal models to focuss on some functional and safety aspects (D.2.6-X-14). Obviously some strictly formal means can be used to define the semi-formal model.

2.5.1 Semi-formal model

Concerning semi-formal model, how the WP2 requirements are covered ?

	Author	Assessor 1	Assessor 2	Total
Consistency to SSRS (D.2.6-X-12.2)				
Coverage of SSRS (D.2.6-X-12.2.1)				
Coverage of SSHA (D.2.6-X-12.2.2)				
Management of requirement justification (D.2.6-X-12.2.3)				
Traceability to SSRS (D.2.6-X-12.2.5)				
Traceability of exported requirements (D.2.6-X-12.2.6)				
Simulation or animation (D.2.6-X-13 partial)				
Execution (D.2.6-X-13 partial)				
Extensible to strictly formal model (D.2.6-X-14.3)				
Easy to refine towards strictly formal model (D.2.6-X-14.4)				
Extensible and modular design (D.2.6-X-15)				
Extensible to software architecture and design (D.2.6-X-30)				

Concerning safety properties management, how the WP2 requirements are covered ?

	Author	Assessor 1	Assessor 2	Total
Safety function isolation (D.2.6-X-17)				
Safety properties formalisation (D.2.6-X-22)				
Logical expression (D.2.6-X-28.2.2)				
Timing constraints (D.2.6-X-28.2.3)				
Safety properties validation (D.2.6-X-23.2)				
Logical properties assertion (D.2.6-X-34)				
Check of assertions (D.2.6-X-34.1)				

Does the language allow to formalize (D.2.6-X-31):

	Author	Assessor 1	Assessor 2	Total
State machines				
Time-outs				
Truth tables				
Arithmetic				
Braking curves				
Logical statements				
Message and fields				

Additional comments on semi-formal model

Do you think your semi-formal model is sufficient to cover a safe design of the on-board unit until code generation ? All comments on links to other models, validation and verification activities are welcomed.

2.5.2 Strictly formal model

Concerning strictly formal model, how the WP2 requirements are covered ?

	Author	Assessor 1	Assessor 2	Total
Consistency to SFM (D.2.6-X-14.2)				
Coverage of SSRS (D.2.6-X-14.2)				
Traceability to SSRS (D.2.6-X-14.3)				
Extensible to software design (D.2.6-X-16)				
Safety function isolation (D.2.6-X-17)				
Safety properties formalisation (D.2.6-X-22)				
Logical expression (D.2.6-X-28.2.2)				
Timing constraints (D.2.6-X-28.2.3)				
Safety properties validation (D.2.6-X-23.3)				
Logical properties assertion (D.2.6-X-34)				
Proof of assertions (D.2.6-X-34.2)				

Does the language allow to formalize (D.2.6-X-32):

	Author	Assessor 1	Assessor 2	Total
State machines				
Time-outs				
Truth tables				
Arithmetic				
Braking curves				
Logical statements				
Message and fields				

Additional comments on semi-formal model

Do you think your strictly formal model can be directly defined from the SSRS ? All comments on links to other models, validation and verification activities are welcomed.

2.6 Software design

This section discusses the usage of the approach for software design. It can be skipped depending the results of L.8.

2.6.1 Functional design

How the approach allows to produce a functional software model of the on-board unit ?

	Author	Assessor 1	Assessor 2	Total
Derivation from system semi-formal model				
Software architecture description				
Software constraints				
Traceability				
Executable				

2.6.2 SSIL4 design

How the approach allows to produce in safety a software model ?

	Author	Assessor 1	Assessor 2	Total
Derivation from system semi-formal or strictly formal model				
Software architecture description				
Software constraints				
Traceability				
Executable				
Conformance to EN50128 § 7.2				
Conformance to EN50128 § 7.3				
Conformance to EN50128 § 7.4				

Which criteria for software architecture are covered by the methodology (see EN50128 table A.3) :

	Author	Assessor 1	Assessor 2	Total
Defensive programming				
Fault detection & diagnostic				
Error detecting code				
Failure assertion programming				
Diverse programming				
Memorising executed cases				
Software error effect analysis				
Fully defined interface				
Modelling				
Structured methodology				

2.7 Software code generation

This section discusses the usage of the approach for software code generation. It can be skipped depending the results of L.8.

Which criteria for software design and implementation are covered by the methodology (see EN50128 table A.4) :

	Author	Assessor 1	Assessor 2	Total
Formal methods				
Modeling				
Modular approach (mandatory)				
Components				
Design and coding standards (mandatory)				
Strongly typed programming language				

2.8 Main usage of the tool

This section discusses the main usage of the tool.

Which task are covered by the tool ?

	Author	Assessor 1	Assessor 2	Total
Modelling support				
Automatic translation				
Code Generation				
Model verification				
Test generation				
Simulation, execution, debugging				
Formal proof				

Modelling support

Does the tool provide a textual or a graphical editor ?

Automatic translation and code generation

Which translation or code generation is supported by the tool ?

Model verification

Which verification on models are provided by the tool?

Test generation

Does the tool allow to generate tests ? For which purpose ?

Simulation, execution, debugging

Does the tool allow to simulate or to debbug step by step a model or a code ?

Formal proof

Does the tool allow formal proof ? How ?

2.9 Use of the tool

According WP2 requirements, give a note for characteristics of the use of the tool (from 0 to 3) :

	Author	Assessor 1	Assessor 2	Total
Open Source (D2.6-X-36)				
Portability to operating systems (D2.6-X-37)				
Cooperation of tools (D2.6-X-38)				
Robustness (D2.6-X-41)				
Modularity (D2.6-X-41.1)				
Documentation management (D.2.6-X-41.2)				
Distributed software development (D.2.6-X-41.3)				
Simultaneous multi-users (D.2.6-X-41.4)				
Issue tracking (D.2.6-X-41.5)				
Differences between models (D.2.6-X-41.6)				
Version management (D.2.6-X-41.7)				
Concurrent version development (D.2.6-X-41.8)				
Model-based version control (D.2.6-X-41.9)				
Role traceability (D.2.6-X-41.10)				
Safety version traceability (D.2.6-X-41.11)				
Model traceability (D.2.6-01-035)				
Tool chain integration				
Scalability				

2.10 Certifiability

This section discusses how the tool can be classified according EN50128 requirements (D.2.6-X-50).

	Author	Assessor 1	Assessor 2	Total
Tool manual (D.2.6-01-42.02)				
Proof of correctness (D.2.6-01-42.03)				
Existing industrial usage				
Model verification				
Test generation				
Simulation, execution, debugging				
Formal proof				

Other elements for tool certification

2.11 Other comments

Please to give free comments on the approach.

3 Conclusion

This conclusion give a sum up of the evaluation results for each approach. The detailed results of each approach are given in the appendix.

3.1 Main usage of the approach

This section discusses the main usage of the approach.

According to the figure 1, for which phases do you recommend the approach (give a note from 0 to 3) :

	CORE	GOPRR	ERTMSFormalSpecs	SysML with Papyrus	SysML with Entreprise Architect	SCADE	EventB	Classical B	Petri Nets	System C	GNATprove
System Analysis											
Sub-system formal design											
Software design											
Software code generation											

According to the figure 1, for which type of activities do you recommend the approach (give a note from 0 to 3) :

	CORE	GOPRR	ERTMSFormalSpecs	SysML with Papyrus	SysML with Entreprise Architect	SCADE	EventB	Classical B	Petri Nets	System C	GNATprove
Documentation											
Modeling											
Design											
Code generation											
Verification											
Validation											
Safety analyses											

3.2 Language

This section discusses the main element of the language.

Which are the main characteristics of the language :

	CORE	GOPRR	ERTMSFormalSpecs	SysML with Papyrus	SysML with Entreprise Architect	SCADE	EventB	Classical B	Petri Nets	System C	GNATprove
Informal language											
Semi-formal language											
Formal language											
Structured language											
Modular language											
Textual language											
Mathematical symbols or code											
Graphical language											

According WP2 requirements, give a note for the capabilities of the language (from 0 to 3) :

	CORE	GOPRR	ERTMSFormalSpecs	SysML with Papyrus	SysML with Entreprise Architect	SCADE	EventB	Classical B	Petri Nets	System C	GNATprove
Declarative formalization of properties (D.2.6-X-28)											
Simple formalization of properties (D.2.6-X-28.1)											
Scalability : capability to design large model											
Easily translatable to other languages (D.2.6-X-30)											
Executable directly (D.2.6-X-33)											
Executable after translation to a code (D.2.6-X-33) (precise if the translation is automatic)											
Simulation, animation (D.2.6-X-33)											
Easily understandable (D.2.6-X-27)											
Expertise level needed (0 High level, 3 few level)											
Standardization (D.2.6-X-29)											
Documented (D.2.6-X-29)											
Extensible language (D.2.6-01-28)											

3.3 System Analysis

This section discusses the usage of the approach for system analysis. It can be skipped depending the results of L.8.

Acoording WP2 requirements, how the approach can be involved for the sub-system requirement specification ?

	CORE	GOPRR	ERTMSFormalSpecs	SysML with Papyrus	SysML with Entreprise Architect	SCADE	EventB	Classical B	Petri Nets	System C	GNATprove
Independent System functions definition (D.2.6-X-10.2.1)											
System architecture design (D.2.6-X-10.2)											
System data flow identification (D.2.6-X-10.2.3)											
Sub-system focus (D.2.6-X-10.2.4)											
System interfaces definition (D.2.6-X-10.2.5)											
System requirement allocation (D.2.6-X-10.3)											
Traceability with SRS (D.2.6-X-10.5)											
Traceability with Safety activities (D.2.6-X-11)											

3.4 Sub-System formal design

This section discusses the usage of the approach for sub-system formal design. It can be skipped depending the results of L.8.

Two kinds of model can be planned during this phase: semi-formal models to cover the SSRS (D.2.6-X-12.1) and strictly formal models to focuss on some functional and safety aspects (D.2.6-X-14). Obviously some strictly formal means can be used to define the semi-formal model.

3.4.1 Semi-formal model

Concerning semi-formal model, how the WP2 requirements are covered ?

	CORE	GOPRR	ERTMSFormalSpecs	SysML with Papyrus	SysML with Entreprise Architect	SCADE	EventB	Classical B	Petri Nets	System C	GNATprove
Consistency to SSRS (D.2.6-X-12.2)											
Coverage of SSRS (D.2.6-X-12.2.1)											
Coverage of SSHA (D.2.6-X-12.2.2)											
Management of requirement justification (D.2.6-X-12.2.3)											
Traceability to SSRS (D.2.6-X-12.2.5)											
Traceability of exported requirements (D.2.6-X-12.2.6)											
Simulation or animation (D.2.6-X-13 partial)											
Execution (D.2.6-X-13 partial)											
Extensible to strictly formal model (D.2.6-X-14.3)											
Easy to refine towards strictly formal model (D.2.6-X-14.4)											
Extensible and modular design (D.2.6-X-15)											
Extensible to software architecture and design (D.2.6-X-15)											

Concerning safety properties management, how the WP2 requirements are covered ?

	CORE	GOPRR	ERTMSFormalSpecs	SysML with Papyrus	SysML with Entreprise Architect	SCADE	EventB	Classical B	Petri Nets	System C	GNATprove
Safety function isolation (D.2.6-X-17)											
Safety properties formalisation (D.2.6-X-22)											
Logical expression (D.2.6-X-28.2.2)											
Timing constraints (D.2.6-X-28.2.3)											
Safety properties validation (D.2.6-X-23.2)											
Logical properties assertion (D.2.6-X-34)											
Check of assertions (D.2.6-X-34.1)											

Does the language allow to formalize (D.2.6-X-31):

	CORE	GOPRR	ERTMSFormalSpecs	SysML with Papyrus	SysML with Entreprise Architect	SCADE	EventB	Classical B	Petri Nets	System C	GNATprove
State machines											
Time-outs											
Truth tables											
Arithmetic											
Braking curves											
Logical statements											
Message and fields											

3.4.2 Strictly formal model

Concerning strictly formal model, how the WP2 requirements are covered ?

	CORE	GOPRR	ERTMSFormalSpecs	SysML with Papyrus	SysML with Entreprise Architect	SCADE	EventB	Classical B	Petri Nets	System C	GNATprove
Consistency to SFM (D.2.6-X-14.2)											
Coverage of SSRS (D.2.6-X-14.2)											
Traceability to SSRS (D.2.6-X-14.3)											
Extensible to software design (D.2.6-X-16)											
Safety function isolation (D.2.6-X-17)											
Safety properties formalisation (D.2.6-X-22)											
Logical expression (D.2.6-X-28.2.2)											
Timing constraints (D.2.6-X-28.2.3)											
Safety properties validation (D.2.6-X-23.3)											
Logical properties assertion (D.2.6-X-34)											
Proof of assertions (D.2.6-X-34.2)											

Does the language allow to formalize (D.2.6-X-32):

	CORE	GOPRR	ERTMSFormalSpecs	SysML with Papyrus	SysML with Entreprise Architect	SCADE	EventB	Classical B	Petri Nets	System C	GNATprove
State machines											
Time-outs											
Truth tables											
Arithmetic											
Braking curves											
Logical statements											
Message and fields											

3.5 Software design

This section discusses the usage of the approach for software design. It can be skipped depending the results of L.8.

3.5.1 Functional design

How the approach allows to produce a functional software model of the on-board unit ?

	CORE	GOPRR	ERTMSFormalSpecs	SysML with Papyrus	SysML with Entreprise Architect	SCADE	EventB	Classical B	Petri Nets	System C	GNATprove
Derivation from system semi-formal model											
Software architecture description											
Software constraints											
Traceability											
Executable											

3.5.2 SSIL4 design

How the approach allows to produce in safety a software model ?

	CORE	GOPRR	ERTMSFormalSpecs	SysML with Papyrus	SysML with Entreprise Architect	SCADE	EventB	Classical B	Petri Nets	System C	GNATprove
Derivation from system semi-formal or strictly formal model											
Software architecture description											
Software constraints											
Traceability											
Executable											
Conformance to EN50128 § 7.2											
Conformance to EN50128 § 7.3											
Conformance to EN50128 § 7.4											

Which criteria for software architecture are covered by the methodology (see EN50128 table A.3) :

	CORE	GOPRR	ERTMSFormalSpecs	SysML with Papyrus	SysML with Entreprise Architect	SCADE	EventB	Classical B	Petri Nets	System C	GNATprove
Defensive programming											
Fault detection & diagnostic											
Error detecting code											
Failure assertion programming											
Diverse programming											
Memorising executed cases											
Software error effect analysis											
Fully defined interface											
Modelling											
Structured methodology											

3.6 Software code generation

This section discusses the usage of the approach for software code generation. It can be skipped depending the results of L.8.

Which criteria for software design and implementation are covered by the methodology (see EN50128 table A.4) :

	CORE	GOPRR	ERTMSFormalSpecs	SysML with Papyrus	SysML with Entreprise Architect	SCADE	EventB	Classical B	Petri Nets	System C	GNATprove
Formal methods											
Modeling											
Modular approach (mandatory)											
Components											
Design and coding standards (mandatory)											
Strongly typed programming language											

3.7 Main usage of the tool

This section discusses the main usage of the tool.

Which task are covered by the tool ?

	CORE	GOPRR	ERTMSFormalSpecs	SysML with Papyrus	SysML with Entreprise Architect	SCADE	EventB	Classical B	Petri Nets	System C	GNATprove
Modelling support											
Automatic translation											
Code Generation											
Model verification											
Test generation											
Simulation, execution, debugging											
Formal proof											

3.8 Use of the tool

	CORE	GOPRR	ERTMSFormalSpecs	SysML with Papyrus	SysML with Entreprise Architect	SCADE	EventB	Classical B	Petri Nets	System C	GNATprove
Open Source (D2.6-X-36)											
Portability to operating systems (D2.6-X-37)											
Cooperation of tools (D2.6-X-38)											
Robustness (D2.6-X-41)											
Modularity (D2.6-X-41.1)											
Documentation management (D.2.6-X-41.2)											
Distributed software development (D.2.6-X-41.3)											
Simultaneous multi-users (D.2.6-X-41.4)											
Issue tracking (D.2.6-X-41.5)											
Differences between models (D.2.6-X-41.6)											
Version management (D.2.6-X-41.7)											
Concurrent version development (D.2.6-X-41.8)											
Model-based version control (D.2.6-X-41.9)											
Role traceability (D.2.6-X-41.10)											
Safety version traceability (D.2.6-X-41.11)											
Model traceability (D.2.6-01-035)											
Tool chain integration											
Scalability											

3.9 Certifiability

This section discusses how the tool can be classified according EN50128 requirements (D.2.6-X-50).

	CORE	GOPRR	ERTMSFormalSpecs	SysML with Papyrus	SysML with Entreprise Architect	SCADE	EventB	Classical B	Petri Nets	System C	GNATprove
Tool manual (D.2.6-01-42.02)											
Proof of correctness (D.2.6-01-42.03)											
Existing industrial usage											
Model verification											
Test generation											
Simulation, execution, debugging											
Formal proof											

Appendix A: CORE Workstation 5.1

Author Author of the approaches description: Cyril Cornu (All4Tec)

Assessor 1 First assessor of the approaches Marielle Petit-Doche (Systerel)

Assessor 2 Second assessor of the approaches %%Name - Company%%

In the sequel, main text is under the responsibilities of the author.

Author. Author can add comments using this format at any place.

Assessor 1. First assessor can add comments using this format at any place.

Assessor 2. Second assessor can add comments using this format at any place.

When a note is required, please follow this list :

0 not recommended, not adapted, rejected

1 weakly recommended, adapted after major improvements, weakly rejected

2 recommended, adapted (with light improvements if necessary) weakly accepted

3 highly recommended, well adapted,strongly accepted

***** difficult to evaluate with a note (please add a comment under the table)

All the notes can be commented under each table.

A.1 Presentation

This section gives a quick presentation of the approach and the tool.

Name %%Name of the approach and the tool%%

Web site %%if available, how to find information%%

Licence %%Kind of licence%%

Abstract

Short abstract on the approach and tool (10 lines max)

Publications

Short list of publications on the approach (5 max)

A.2 Main usage of the approach

This section discusses the main usage of the approach.

According to the figure 1, for which phases do you recommend the approach (give a note from 0 to 3) :

	Author	Assessor 1	Assessor 2	Total
System Analysis				
Sub-system formal design				
Software design				
Software code generation				

According to the figure 1, for which type of activities do you recommend the approach (give a note from 0 to 3) :

	Author	Assessor 1	Assessor 2	Total
Documentation				
Modeling				
Design				
Code generation				
Verification				
Validation				
Safety analyses				

Known usages

Have you some examples of usage of this approach to compare with the OpenETCS objectives ?

A.3 Language

This section discusses the main element of the language.

Which are the main characteristics of the language :

	Author	Assessor 1	Assessor 2	Total
Informal language				
Semi-formal language				
Formal language				
Structured language				
Modular language				
Textual language				
Mathematical symbols or code				
Graphical language				

According WP2 requirements, give a note for the capabilities of the language (from 0 to 3) :

	Author	Assessor 1	Assessor 2	Total
Declarative formalization of properties (D.2.6-X-28)				
Simple formalization of properties (D.2.6-X-28.1)				
Scalability : capability to design large model				
Easily translatable to other languages (D.2.6-X-30)				
Executable directly (D.2.6-X-33)				
Executable after translation to a code (D.2.6-X-33) (precise if the translation is automatic)				
Simulation, animation (D.2.6-X-33)				
Easily understandable (D.2.6-X-27)				
Expertise level needed (0 High level, 3 few level)				
Standardization (D.2.6-X-29)				
Documented (D.2.6-X-29)				
Extensible language (D.2.6-01-28)				

Documentation

Describe how the language is documented, the existing guidelines, coding rules, standardization...

Language usage

Describe the possible restriction on the language

A.4 System Analysis

This section discusses the usage of the approach for system analysis. It can be skipped depending the results of L.8.

Acoording WP2 requirements, how the approach can be involved for the sub-system requirement specification ?

	Author	Assessor 1	Assessor 2	Total
Independent System functions definition (D.2.6-X-10.2.1)				
System architecture design (D.2.6-X-10.2)				
System data flow identification (D.2.6-X-10.2.3)				
Sub-system focus (D.2.6-X-10.2.4)				
System interfaces definition (D.2.6-X-10.2.5)				
System requirement allocation (D.2.6-X-10.3)				
Traceability with SRS (D.2.6-X-10.5)				
Traceability with Safety activities (D.2.6-X-11)				

A.5 Sub-System formal design

This section discusses the usage of the approach for sub-system formal design. It can be skipped depending the results of L.8.

Two kinds of model can be planned during this phase: semi-formal models to cover the SSRS (D.2.6-X-12.1) and strictly formal models to focuss on some functional and safety aspects (D.2.6-X-14). Obviously some strictly formal means can be used to define the semi-formal model.

A.5.1 Semi-formal model

Concerning semi-formal model, how the WP2 requirements are covered ?

	Author	Assessor 1	Assessor 2	Total
Consistency to SSRS (D.2.6-X-12.2)				
Coverage of SSRS (D.2.6-X-12.2.1)				
Coverage of SSHA (D.2.6-X-12.2.2)				
Management of requirement justification (D.2.6-X-12.2.3)				
Traceability to SSRS (D.2.6-X-12.2.5)				
Traceability of exported requirements (D.2.6-X-12.2.6)				
Simulation or animation (D.2.6-X-13 partial)				
Execution (D.2.6-X-13 partial)				
Extensible to strictly formal model (D.2.6-X-14.3)				
Easy to refine towards strictly formal model (D.2.6-X-14.4)				
Extensible and modular design (D.2.6-X-15)				
Extensible to software architecture and design (D.2.6-X-30)				

Concerning safety properties management, how the WP2 requirements are covered ?

	Author	Assessor 1	Assessor 2	Total
Safety function isolation (D.2.6-X-17)				
Safety properties formalisation (D.2.6-X-22)				
Logical expression (D.2.6-X-28.2.2)				
Timing constraints (D.2.6-X-28.2.3)				
Safety properties validation (D.2.6-X-23.2)				
Logical properties assertion (D.2.6-X-34)				
Check of assertions (D.2.6-X-34.1)				

Does the language allow to formalize (D.2.6-X-31):

	Author	Assessor 1	Assessor 2	Total
State machines				
Time-outs				
Truth tables				
Arithmetic				
Braking curves				
Logical statements				
Message and fields				

Additional comments on semi-formal model

Do you think your semi-formal model is sufficient to cover a safe design of the on-board unit until code generation ? All comments on links to other models, validation and verification activities are welcomed.

A.5.2 Strictly formal model

Concerning strictly formal model, how the WP2 requirements are covered ?

	Author	Assessor 1	Assessor 2	Total
Consistency to SFM (D.2.6-X-14.2)				
Coverage of SSRS (D.2.6-X-14.2)				
Traceability to SSRS (D.2.6-X-14.3)				
Extensible to software design (D.2.6-X-16)				
Safety function isolation (D.2.6-X-17)				
Safety properties formalisation (D.2.6-X-22)				
Logical expression (D.2.6-X-28.2.2)				
Timing constraints (D.2.6-X-28.2.3)				
Safety properties validation (D.2.6-X-23.3)				
Logical properties assertion (D.2.6-X-34)				
Proof of assertions (D.2.6-X-34.2)				

Does the language allow to formalize (D.2.6-X-32):

	Author	Assessor 1	Assessor 2	Total
State machines				
Time-outs				
Truth tables				
Arithmetic				
Braking curves				
Logical statements				
Message and fields				

Additional comments on semi-formal model

Do you think your strictly formal model can be directly defined from the SSRS ? All comments on links to other models, validation and verification activities are welcomed.

A.6 Software design

This section discusses the usage of the approach for software design. It can be skipped depending the results of L.8.

A.6.1 Functional design

How the approach allows to produce a functional software model of the on-board unit ?

	Author	Assessor 1	Assessor 2	Total
Derivation from system semi-formal model				
Software architecture description				
Software constraints				
Traceability				
Executable				

A.6.2 SSIL4 design

How the approach allows to produce in safety a software model ?

	Author	Assessor 1	Assessor 2	Total
Derivation from system semi-formal or strictly formal model				
Software architecture description				
Software constraints				
Traceability				
Executable				
Conformance to EN50128 § 7.2				
Conformance to EN50128 § 7.3				
Conformance to EN50128 § 7.4				

Which criteria for software architecture are covered by the methodology (see EN50128 table A.3) :

	Author	Assessor 1	Assessor 2	Total
Defensive programming				
Fault detection & diagnostic				
Error detecting code				
Failure assertion programming				
Diverse programming				
Memorising executed cases				
Software error effect analysis				
Fully defined interface				
Modelling				
Structured methodology				

A.7 Software code generation

This section discusses the usage of the approach for software code generation. It can be skipped depending the results of L.8.

Which criteria for software design and implementation are covered by the methodology (see EN50128 table A.4) :

	Author	Assessor 1	Assessor 2	Total
Formal methods				
Modeling				
Modular approach (mandatory)				
Components				
Design and coding standards (mandatory)				
Strongly typed programming language				

A.8 Main usage of the tool

This section discusses the main usage of the tool.

Which task are covered by the tool ?

	Author	Assessor 1	Assessor 2	Total
Modelling support				
Automatic translation				
Code Generation				
Model verification				
Test generation				
Simulation, execution, debugging				
Formal proof				

Modelling support

Does the tool provide a textual or a graphical editor ?

Automatic translation and code generation

Which translation or code generation is supported by the tool ?

Model verification

Which verification on models are provided by the tool?

Test generation

Does the tool allow to generate tests ? For which purpose ?

Simulation, execution, debugging

Does the tool allow to simulate or to debbug step by step a model or a code ?

Formal proof

Does the tool allow formal proof ? How ?

A.9 Use of the tool

According WP2 requirements, give a note for characteristics of the use of the tool (from 0 to 3) :

	Author	Assessor 1	Assessor 2	Total
Open Source (D2.6-X-36)				
Portability to operating systems (D2.6-X-37)				
Cooperation of tools (D2.6-X-38)				
Robustness (D2.6-X-41)				
Modularity (D2.6-X-41.1)				
Documentation management (D.2.6-X-41.2)				
Distributed software development (D.2.6-X-41.3)				
Simultaneous multi-users (D.2.6-X-41.4)				
Issue tracking (D.2.6-X-41.5)				
Differences between models (D.2.6-X-41.6)				
Version management (D.2.6-X-41.7)				
Concurrent version development (D.2.6-X-41.8)				
Model-based version control (D.2.6-X-41.9)				
Role traceability (D.2.6-X-41.10)				
Safety version traceability (D.2.6-X-41.11)				
Model traceability (D.2.6-01-035)				
Tool chain integration				
Scalability				

A.10 Certifiability

This section discusses how the tool can be classified according EN50128 requirements (D.2.6-X-50).

	Author	Assessor 1	Assessor 2	Total
Tool manual (D.2.6-01-42.02)				
Proof of correctness (D.2.6-01-42.03)				
Existing industrial usage				
Model verification				
Test generation				
Simulation, execution, debugging				
Formal proof				

Other elements for tool certification

A.11 Other comments

Please to give free comments on the approach.

Appendix B: GOPRR

Author Author of the approaches description Johannes Feuser (Uni. Bremen)

Assessor 1 First assessor of the approaches Alexandre Ginisty (All4Tec)

Assessor 2 Second assessor of the approaches Matthias Gudemann (Systerel)

In the sequel, main text is under the responsibilities of the author.

Author. Author can add comments using this format at any place.

Assessor 1. First assessor can add comments using this format at any place.

Assessor 2. Second assessor can add comments using this format at any place.

When a note is required, please follow this list :

0 not recommended, not adapted, rejected

1 weakly recommended, adapted after major improvements, weakly rejected

2 recommended, adapted (with light improvements if necessary) weakly accepted

3 highly recommended, well adapted,strongly accepted

***** difficult to evaluate with a note (please add a comment under the table)

All the notes can be commented under each table.

B.1 Presentation

This section gives a quick presentation of the approach and the tool.

Name %%Name of the approach and the tool%%

Web site %%if available, how to find information%%

Licence %%Kind of licence%%

Abstract

Short abstract on the approach and tool (10 lines max)

Publications

Short list of publications on the approach (5 max)

B.2 Main usage of the approach

This section discusses the main usage of the approach.

According to the figure 1, for which phases do you recommend the approach (give a note from 0 to 3) :

	Author	Assessor 1	Assessor 2	Total
System Analysis				
Sub-system formal design				
Software design				
Software code generation				

According to the figure 1, for which type of activities do you recommend the approach (give a note from 0 to 3) :

	Author	Assessor 1	Assessor 2	Total
Documentation				
Modeling				
Design				
Code generation				
Verification				
Validation				
Safety analyses				

Known usages

Have you some examples of usage of this approach to compare with the OpenETCS objectives ?

B.3 Language

This section discusses the main element of the language.

Which are the main characteristics of the language :

	Author	Assessor 1	Assessor 2	Total
Informal language				
Semi-formal language				
Formal language				
Structured language				
Modular language				
Textual language				
Mathematical symbols or code				
Graphical language				

According WP2 requirements, give a note for the capabilities of the language (from 0 to 3) :

	Author	Assessor 1	Assessor 2	Total
Declarative formalization of properties (D.2.6-X-28)				
Simple formalization of properties (D.2.6-X-28.1)				
Scalability : capability to design large model				
Easily translatable to other languages (D.2.6-X-30)				
Executable directly (D.2.6-X-33)				
Executable after translation to a code (D.2.6-X-33) (precise if the translation is automatic)				
Simulation, animation (D.2.6-X-33)				
Easily understandable (D.2.6-X-27)				
Expertise level needed (0 High level, 3 few level)				
Standardization (D.2.6-X-29)				
Documented (D.2.6-X-29)				
Extensible language (D.2.6-01-28)				

Documentation

Describe how the language is documented, the existing guidelines, coding rules, standardization...

Language usage

Describe the possible restriction on the language

B.4 System Analysis

This section discusses the usage of the approach for system analysis. It can be skipped depending the results of L.8.

Acoording WP2 requirements, how the approach can be involved for the sub-system requirement specification ?

	Author	Assessor 1	Assessor 2	Total
Independent System functions definition (D.2.6-X-10.2.1)				
System architecture design (D.2.6-X-10.2)				
System data flow identification (D.2.6-X-10.2.3)				
Sub-system focus (D.2.6-X-10.2.4)				
System interfaces definition (D.2.6-X-10.2.5)				
System requirement allocation (D.2.6-X-10.3)				
Traceability with SRS (D.2.6-X-10.5)				
Traceability with Safety activities (D.2.6-X-11)				

B.5 Sub-System formal design

This section discusses the usage of the approach for sub-system formal design. It can be skipped depending the results of L.8.

Two kinds of model can be planned during this phase: semi-formal models to cover the SSRS (D.2.6-X-12.1) and strictly formal models to focuss on some functional and safety aspects (D.2.6-X-14). Obviously some strictly formal means can be used to define the semi-formal model.

B.5.1 Semi-formal model

Concerning semi-formal model, how the WP2 requirements are covered ?

	Author	Assessor 1	Assessor 2	Total
Consistency to SSRS (D.2.6-X-12.2)				
Coverage of SSRS (D.2.6-X-12.2.1)				
Coverage of SSHA (D.2.6-X-12.2.2)				
Management of requirement justification (D.2.6-X-12.2.3)				
Traceability to SSRS (D.2.6-X-12.2.5)				
Traceability of exported requirements (D.2.6-X-12.2.6)				
Simulation or animation (D.2.6-X-13 partial)				
Execution (D.2.6-X-13 partial)				
Extensible to strictly formal model (D.2.6-X-14.3)				
Easy to refine towards strictly formal model (D.2.6-X-14.4)				
Extensible and modular design (D.2.6-X-15)				
Extensible to software architecture and design (D.2.6-X-30)				

Concerning safety properties management, how the WP2 requirements are covered ?

	Author	Assessor 1	Assessor 2	Total
Safety function isolation (D.2.6-X-17)				
Safety properties formalisation (D.2.6-X-22)				
Logical expression (D.2.6-X-28.2.2)				
Timing constraints (D.2.6-X-28.2.3)				
Safety properties validation (D.2.6-X-23.2)				
Logical properties assertion (D.2.6-X-34)				
Check of assertions (D.2.6-X-34.1)				

Does the language allow to formalize (D.2.6-X-31):

	Author	Assessor 1	Assessor 2	Total
State machines				
Time-outs				
Truth tables				
Arithmetic				
Braking curves				
Logical statements				
Message and fields				

Additional comments on semi-formal model

Do you think your semi-formal model is sufficient to cover a safe design of the on-board unit until code generation ? All comments on links to other models, validation and verification activities are welcomed.

B.5.2 Strictly formal model

Concerning strictly formal model, how the WP2 requirements are covered ?

	Author	Assessor 1	Assessor 2	Total
Consistency to SFM (D.2.6-X-14.2)				
Coverage of SSRS (D.2.6-X-14.2)				
Traceability to SSRS (D.2.6-X-14.3)				
Extensible to software design (D.2.6-X-16)				
Safety function isolation (D.2.6-X-17)				
Safety properties formalisation (D.2.6-X-22)				
Logical expression (D.2.6-X-28.2.2)				
Timing constraints (D.2.6-X-28.2.3)				
Safety properties validation (D.2.6-X-23.3)				
Logical properties assertion (D.2.6-X-34)				
Proof of assertions (D.2.6-X-34.2)				

Does the language allow to formalize (D.2.6-X-32):

	Author	Assessor 1	Assessor 2	Total
State machines				
Time-outs				
Truth tables				
Arithmetic				
Braking curves				
Logical statements				
Message and fields				

Additional comments on semi-formal model

Do you think your strictly formal model can be directly defined from the SSRS ? All comments on links to other models, validation and verification activities are welcomed.

B.6 Software design

This section discusses the usage of the approach for software design. It can be skipped depending the results of L.8.

B.6.1 Functional design

How the approach allows to produce a functional software model of the on-board unit ?

	Author	Assessor 1	Assessor 2	Total
Derivation from system semi-formal model				
Software architecture description				
Software constraints				
Traceability				
Executable				

B.6.2 SSIL4 design

How the approach allows to produce in safety a software model ?

	Author	Assessor 1	Assessor 2	Total
Derivation from system semi-formal or strictly formal model				
Software architecture description				
Software constraints				
Traceability				
Executable				
Conformance to EN50128 § 7.2				
Conformance to EN50128 § 7.3				
Conformance to EN50128 § 7.4				

Which criteria for software architecture are covered by the methodology (see EN50128 table A.3) :

	Author	Assessor 1	Assessor 2	Total
Defensive programming				
Fault detection & diagnostic				
Error detecting code				
Failure assertion programming				
Diverse programming				
Memorising executed cases				
Software error effect analysis				
Fully defined interface				
Modelling				
Structured methodology				

B.7 Software code generation

This section discusses the usage of the approach for software code generation. It can be skipped depending the results of L.8.

Which criteria for software design and implementation are covered by the methodology (see EN50128 table A.4) :

	Author	Assessor 1	Assessor 2	Total
Formal methods				
Modeling				
Modular approach (mandatory)				
Components				
Design and coding standards (mandatory)				
Strongly typed programming language				

B.8 Main usage of the tool

This section discusses the main usage of the tool.

Which task are covered by the tool ?

	Author	Assessor 1	Assessor 2	Total
Modelling support				
Automatic translation				
Code Generation				
Model verification				
Test generation				
Simulation, execution, debugging				
Formal proof				

Modelling support

Does the tool provide a textual or a graphical editor ?

Automatic translation and code generation

Which translation or code generation is supported by the tool ?

Model verification

Which verification on models are provided by the tool?

Test generation

Does the tool allow to generate tests ? For which purpose ?

Simulation, execution, debugging

Does the tool allow to simulate or to debbug step by step a model or a code ?

Formal proof

Does the tool allow formal proof ? How ?

B.9 Use of the tool

According WP2 requirements, give a note for characteristics of the use of the tool (from 0 to 3) :

	Author	Assessor 1	Assessor 2	Total
Open Source (D2.6-X-36)				
Portability to operating systems (D2.6-X-37)				
Cooperation of tools (D2.6-X-38)				
Robustness (D2.6-X-41)				
Modularity (D2.6-X-41.1)				
Documentation management (D.2.6-X-41.2)				
Distributed software development (D.2.6-X-41.3)				
Simultaneous multi-users (D.2.6-X-41.4)				
Issue tracking (D.2.6-X-41.5)				
Differences between models (D.2.6-X-41.6)				
Version management (D.2.6-X-41.7)				
Concurrent version development (D.2.6-X-41.8)				
Model-based version control (D.2.6-X-41.9)				
Role traceability (D.2.6-X-41.10)				
Safety version traceability (D.2.6-X-41.11)				
Model traceability (D.2.6-01-035)				
Tool chain integration				
Scalability				

B.10 Certifiability

This section discusses how the tool can be classified according EN50128 requirements (D.2.6-X-50).

	Author	Assessor 1	Assessor 2	Total
Tool manual (D.2.6-01-42.02)				
Proof of correctness (D.2.6-01-42.03)				
Existing industrial usage				
Model verification				
Test generation				
Simulation, execution, debugging				
Formal proof				

Other elements for tool certification

B.11 Other comments

Please to give free comments on the approach.

Appendix C: ERTMSFormalSpecs

Author Author of the approaches description Stanislas Pinte (ERTMS Solutions)

Assessor 1 First assessor of the approaches Renaud De Landtsheer (Alstom Be)

Assessor 2 Second assessor of the approaches Marielle Petit-Doche (Systerel)

In the sequel, main text is under the responsibilities of the author.

Author. Author can add comments using this format at any place.

Assessor 1. First assessor can add comments using this format at any place.

Assessor 2. Second assessor can add comments using this format at any place.

When a note is required, please follow this list :

0 not recommended, not adapted, rejected

1 weakly recommended, adapted after major improvements, weakly rejected

2 recommended, adapted (with light improvements if necessary) weakly accepted

3 highly recommended, well adapted,strongly accepted

***** difficult to evaluate with a note (please add a comment under the table)

All the notes can be commented under each table.

C.1 Presentation

This section gives a quick presentation of the approach and the tool.

Name %%Name of the approach and the tool%%

Web site %%if available, how to find information%%

Licence %%Kind of licence%%

Abstract

Short abstract on the approach and tool (10 lines max)

Publications

Short list of publications on the approach (5 max)

C.2 Main usage of the approach

This section discusses the main usage of the approach.

According to the figure 1, for which phases do you recommend the approach (give a note from 0 to 3) :

	Author	Assessor 1	Assessor 2	Total
System Analysis				
Sub-system formal design				
Software design				
Software code generation				

According to the figure 1, for which type of activities do you recommend the approach (give a note from 0 to 3) :

	Author	Assessor 1	Assessor 2	Total
Documentation				
Modeling				
Design				
Code generation				
Verification				
Validation				
Safety analyses				

Known usages

Have you some examples of usage of this approach to compare with the OpenETCS objectives ?

C.3 Language

This section discusses the main element of the language.

Which are the main characteristics of the language :

	Author	Assessor 1	Assessor 2	Total
Informal language				
Semi-formal language				
Formal language				
Structured language				
Modular language				
Textual language				
Mathematical symbols or code				
Graphical language				

According WP2 requirements, give a note for the capabilities of the language (from 0 to 3) :

	Author	Assessor 1	Assessor 2	Total
Declarative formalization of properties (D.2.6-X-28)				
Simple formalization of properties (D.2.6-X-28.1)				
Scalability : capability to design large model				
Easily translatable to other languages (D.2.6-X-30)				
Executable directly (D.2.6-X-33)				
Executable after translation to a code (D.2.6-X-33) (precise if the translation is automatic)				
Simulation, animation (D.2.6-X-33)				
Easily understandable (D.2.6-X-27)				
Expertise level needed (0 High level, 3 few level)				
Standardization (D.2.6-X-29)				
Documented (D.2.6-X-29)				
Extensible language (D.2.6-01-28)				

Documentation

Describe how the language is documented, the existing guidelines, coding rules, standardization...

Language usage

Describe the possible restriction on the language

C.4 System Analysis

This section discusses the usage of the approach for system analysis. It can be skipped depending the results of L.8.

Acoording WP2 requirements, how the approach can be involved for the sub-system requirement specification ?

	Author	Assessor 1	Assessor 2	Total
Independent System functions definition (D.2.6-X-10.2.1)				
System architecture design (D.2.6-X-10.2)				
System data flow identification (D.2.6-X-10.2.3)				
Sub-system focus (D.2.6-X-10.2.4)				
System interfaces definition (D.2.6-X-10.2.5)				
System requirement allocation (D.2.6-X-10.3)				
Traceability with SRS (D.2.6-X-10.5)				
Traceability with Safety activities (D.2.6-X-11)				

C.5 Sub-System formal design

This section discusses the usage of the approach for sub-system formal design. It can be skipped depending the results of L.8.

Two kinds of model can be planned during this phase: semi-formal models to cover the SSRS (D.2.6-X-12.1) and strictly formal models to focuss on some functional and safety aspects (D.2.6-X-14). Obviously some strictly formal means can be used to define the semi-formal model.

C.5.1 Semi-formal model

Concerning semi-formal model, how the WP2 requirements are covered ?

	Author	Assessor 1	Assessor 2	Total
Consistency to SSRS (D.2.6-X-12.2)				
Coverage of SSRS (D.2.6-X-12.2.1)				
Coverage of SSHA (D.2.6-X-12.2.2)				
Management of requirement justification (D.2.6-X-12.2.3)				
Traceability to SSRS (D.2.6-X-12.2.5)				
Traceability of exported requirements (D.2.6-X-12.2.6)				
Simulation or animation (D.2.6-X-13 partial)				
Execution (D.2.6-X-13 partial)				
Extensible to strictly formal model (D.2.6-X-14.3)				
Easy to refine towards strictly formal model (D.2.6-X-14.4)				
Extensible and modular design (D.2.6-X-15)				
Extensible to software architecture and design (D.2.6-X-30)				

Concerning safety properties management, how the WP2 requirements are covered ?

	Author	Assessor 1	Assessor 2	Total
Safety function isolation (D.2.6-X-17)				
Safety properties formalisation (D.2.6-X-22)				
Logical expression (D.2.6-X-28.2.2)				
Timing constraints (D.2.6-X-28.2.3)				
Safety properties validation (D.2.6-X-23.2)				
Logical properties assertion (D.2.6-X-34)				
Check of assertions (D.2.6-X-34.1)				

Does the language allow to formalize (D.2.6-X-31):

	Author	Assessor 1	Assessor 2	Total
State machines				
Time-outs				
Truth tables				
Arithmetic				
Braking curves				
Logical statements				
Message and fields				

Additional comments on semi-formal model

Do you think your semi-formal model is sufficient to cover a safe design of the on-board unit until code generation ? All comments on links to other models, validation and verification activities are welcomed.

C.5.2 Strictly formal model

Concerning strictly formal model, how the WP2 requirements are covered ?

	Author	Assessor 1	Assessor 2	Total
Consistency to SFM (D.2.6-X-14.2)				
Coverage of SSRS (D.2.6-X-14.2)				
Traceability to SSRS (D.2.6-X-14.3)				
Extensible to software design (D.2.6-X-16)				
Safety function isolation (D.2.6-X-17)				
Safety properties formalisation (D.2.6-X-22)				
Logical expression (D.2.6-X-28.2.2)				
Timing constraints (D.2.6-X-28.2.3)				
Safety properties validation (D.2.6-X-23.3)				
Logical properties assertion (D.2.6-X-34)				
Proof of assertions (D.2.6-X-34.2)				

Does the language allow to formalize (D.2.6-X-32):

	Author	Assessor 1	Assessor 2	Total
State machines				
Time-outs				
Truth tables				
Arithmetic				
Braking curves				
Logical statements				
Message and fields				

Additional comments on semi-formal model

Do you think your strictly formal model can be directly defined from the SSRS ? All comments on links to other models, validation and verification activities are welcomed.

C.6 Software design

This section discusses the usage of the approach for software design. It can be skipped depending the results of L.8.

C.6.1 Functional design

How the approach allows to produce a functional software model of the on-board unit ?

	Author	Assessor 1	Assessor 2	Total
Derivation from system semi-formal model				
Software architecture description				
Software constraints				
Traceability				
Executable				

C.6.2 SSIL4 design

How the approach allows to produce in safety a software model ?

	Author	Assessor 1	Assessor 2	Total
Derivation from system semi-formal or strictly formal model				
Software architecture description				
Software constraints				
Traceability				
Executable				
Conformance to EN50128 § 7.2				
Conformance to EN50128 § 7.3				
Conformance to EN50128 § 7.4				

Which criteria for software architecture are covered by the methodology (see EN50128 table A.3) :

	Author	Assessor 1	Assessor 2	Total
Defensive programming				
Fault detection & diagnostic				
Error detecting code				
Failure assertion programming				
Diverse programming				
Memorising executed cases				
Software error effect analysis				
Fully defined interface				
Modelling				
Structured methodology				

C.7 Software code generation

This section discusses the usage of the approach for software code generation. It can be skipped depending the results of L.8.

Which criteria for software design and implementation are covered by the methodology (see EN50128 table A.4) :

	Author	Assessor 1	Assessor 2	Total
Formal methods				
Modeling				
Modular approach (mandatory)				
Components				
Design and coding standards (mandatory)				
Strongly typed programming language				

C.8 Main usage of the tool

This section discusses the main usage of the tool.

Which task are covered by the tool ?

	Author	Assessor 1	Assessor 2	Total
Modelling support				
Automatic translation				
Code Generation				
Model verification				
Test generation				
Simulation, execution, debugging				
Formal proof				

Modelling support

Does the tool provide a textual or a graphical editor ?

Automatic translation and code generation

Which translation or code generation is supported by the tool ?

Model verification

Which verification on models are provided by the tool?

Test generation

Does the tool allow to generate tests ? For which purpose ?

Simulation, execution, debugging

Does the tool allow to simulate or to debbug step by step a model or a code ?

Formal proof

Does the tool allow formal proof ? How ?

C.9 Use of the tool

According WP2 requirements, give a note for characteristics of the use of the tool (from 0 to 3) :

	Author	Assessor 1	Assessor 2	Total
Open Source (D2.6-X-36)				
Portability to operating systems (D2.6-X-37)				
Cooperation of tools (D2.6-X-38)				
Robustness (D2.6-X-41)				
Modularity (D2.6-X-41.1)				
Documentation management (D.2.6-X-41.2)				
Distributed software development (D.2.6-X-41.3)				
Simultaneous multi-users (D.2.6-X-41.4)				
Issue tracking (D.2.6-X-41.5)				
Differences between models (D.2.6-X-41.6)				
Version management (D.2.6-X-41.7)				
Concurrent version development (D.2.6-X-41.8)				
Model-based version control (D.2.6-X-41.9)				
Role traceability (D.2.6-X-41.10)				
Safety version traceability (D.2.6-X-41.11)				
Model traceability (D.2.6-01-035)				
Tool chain integration				
Scalability				

C.10 Certifiability

This section discusses how the tool can be classified according EN50128 requirements (D.2.6-X-50).

	Author	Assessor 1	Assessor 2	Total
Tool manual (D.2.6-01-42.02)				
Proof of correctness (D.2.6-01-42.03)				
Existing industrial usage				
Model verification				
Test generation				
Simulation, execution, debugging				
Formal proof				

Other elements for tool certification

C.11 Other comments

Please to give free comments on the approach.

Appendix D: SysML with Papyrus

Author Author of the approaches description Alexander Stante (Fraunhofer)

Assessor 1 First assessor of the approaches Renaud De Landtsheer (Alstom BE)

Assessor 2 Second assessor of the approaches Cyril Cornu (All4Tec)

In the sequel, main text is under the responsibilities of the author.

Author. Author can add comments using this format at any place.

Assessor 1. First assessor can add comments using this format at any place.

Assessor 2. Second assessor can add comments using this format at any place.

When a note is required, please follow this list :

0 not recommended, not adapted, rejected

1 weakly recommended, adapted after major improvements, weakly rejected

2 recommended, adapted (with light improvements if necessary) weakly accepted

3 highly recommended, well adapted,strongly accepted

***** difficult to evaluate with a note (please add a comment under the table)

All the notes can be commented under each table.

D.1 Presentation

This section gives a quick presentation of the approach and the tool.

Name %%Name of the approach and the tool%%

Web site %%if available, how to find information%%

Licence %%Kind of licence%%

Abstract

Short abstract on the approach and tool (10 lines max)

Publications

Short list of publications on the approach (5 max)

D.2 Main usage of the approach

This section discusses the main usage of the approach.

According to the figure 1, for which phases do you recommend the approach (give a note from 0 to 3) :

	Author	Assessor 1	Assessor 2	Total
System Analysis				
Sub-system formal design				
Software design				
Software code generation				

According to the figure 1, for which type of activities do you recommend the approach (give a note from 0 to 3) :

	Author	Assessor 1	Assessor 2	Total
Documentation				
Modeling				
Design				
Code generation				
Verification				
Validation				
Safety analyses				

Known usages

Have you some examples of usage of this approach to compare with the OpenETCS objectives ?

D.3 Language

This section discusses the main element of the language.

Which are the main characteristics of the language :

	Author	Assessor 1	Assessor 2	Total
Informal language				
Semi-formal language				
Formal language				
Structured language				
Modular language				
Textual language				
Mathematical symbols or code				
Graphical language				

According WP2 requirements, give a note for the capabilities of the language (from 0 to 3) :

	Author	Assessor 1	Assessor 2	Total
Declarative formalization of properties (D.2.6-X-28)				
Simple formalization of properties (D.2.6-X-28.1)				
Scalability : capability to design large model				
Easily translatable to other languages (D.2.6-X-30)				
Executable directly (D.2.6-X-33)				
Executable after translation to a code (D.2.6-X-33) (precise if the translation is automatic)				
Simulation, animation (D.2.6-X-33)				
Easily understandable (D.2.6-X-27)				
Expertise level needed (0 High level, 3 few level)				
Standardization (D.2.6-X-29)				
Documented (D.2.6-X-29)				
Extensible language (D.2.6-01-28)				

Documentation

Describe how the language is documented, the existing guidelines, coding rules, standardization...

Language usage

Describe the possible restriction on the language

D.4 System Analysis

This section discusses the usage of the approach for system analysis. It can be skipped depending the results of L.8.

Acoording WP2 requirements, how the approach can be involved for the sub-system requirement specification ?

	Author	Assessor 1	Assessor 2	Total
Independent System functions definition (D.2.6-X-10.2.1)				
System architecture design (D.2.6-X-10.2)				
System data flow identification (D.2.6-X-10.2.3)				
Sub-system focus (D.2.6-X-10.2.4)				
System interfaces definition (D.2.6-X-10.2.5)				
System requirement allocation (D.2.6-X-10.3)				
Traceability with SRS (D.2.6-X-10.5)				
Traceability with Safety activities (D.2.6-X-11)				

D.5 Sub-System formal design

This section discusses the usage of the approach for sub-system formal design. It can be skipped depending the results of L.8.

Two kinds of model can be planned during this phase: semi-formal models to cover the SSRS (D.2.6-X-12.1) and strictly formal models to focuss on some functional and safety aspects (D.2.6-X-14). Obviously some strictly formal means can be used to define the semi-formal model.

D.5.1 Semi-formal model

Concerning semi-formal model, how the WP2 requirements are covered ?

	Author	Assessor 1	Assessor 2	Total
Consistency to SSRS (D.2.6-X-12.2)				
Coverage of SSRS (D.2.6-X-12.2.1)				
Coverage of SSHA (D.2.6-X-12.2.2)				
Management of requirement justification (D.2.6-X-12.2.3)				
Traceability to SSRS (D.2.6-X-12.2.5)				
Traceability of exported requirements (D.2.6-X-12.2.6)				
Simulation or animation (D.2.6-X-13 partial)				
Execution (D.2.6-X-13 partial)				
Extensible to strictly formal model (D.2.6-X-14.3)				
Easy to refine towards strictly formal model (D.2.6-X-14.4)				
Extensible and modular design (D.2.6-X-15)				
Extensible to software architecture and design (D.2.6-X-30)				

Concerning safety properties management, how the WP2 requirements are covered ?

	Author	Assessor 1	Assessor 2	Total
Safety function isolation (D.2.6-X-17)				
Safety properties formalisation (D.2.6-X-22)				
Logical expression (D.2.6-X-28.2.2)				
Timing constraints (D.2.6-X-28.2.3)				
Safety properties validation (D.2.6-X-23.2)				
Logical properties assertion (D.2.6-X-34)				
Check of assertions (D.2.6-X-34.1)				

Does the language allow to formalize (D.2.6-X-31):

	Author	Assessor 1	Assessor 2	Total
State machines				
Time-outs				
Truth tables				
Arithmetic				
Braking curves				
Logical statements				
Message and fields				

Additional comments on semi-formal model

Do you think your semi-formal model is sufficient to cover a safe design of the on-board unit until code generation ? All comments on links to other models, validation and verification activities are welcomed.

D.5.2 Strictly formal model

Concerning strictly formal model, how the WP2 requirements are covered ?

	Author	Assessor 1	Assessor 2	Total
Consistency to SFM (D.2.6-X-14.2)				
Coverage of SSRS (D.2.6-X-14.2)				
Traceability to SSRS (D.2.6-X-14.3)				
Extensible to software design (D.2.6-X-16)				
Safety function isolation (D.2.6-X-17)				
Safety properties formalisation (D.2.6-X-22)				
Logical expression (D.2.6-X-28.2.2)				
Timing constraints (D.2.6-X-28.2.3)				
Safety properties validation (D.2.6-X-23.3)				
Logical properties assertion (D.2.6-X-34)				
Proof of assertions (D.2.6-X-34.2)				

Does the language allow to formalize (D.2.6-X-32):

	Author	Assessor 1	Assessor 2	Total
State machines				
Time-outs				
Truth tables				
Arithmetic				
Braking curves				
Logical statements				
Message and fields				

Additional comments on semi-formal model

Do you think your strictly formal model can be directly defined from the SSRS ? All comments on links to other models, validation and verification activities are welcomed.

D.6 Software design

This section discusses the usage of the approach for software design. It can be skipped depending the results of L.8.

D.6.1 Functional design

How the approach allows to produce a functional software model of the on-board unit ?

	Author	Assessor 1	Assessor 2	Total
Derivation from system semi-formal model				
Software architecture description				
Software constraints				
Traceability				
Executable				

D.6.2 SSIL4 design

How the approach allows to produce in safety a software model ?

	Author	Assessor 1	Assessor 2	Total
Derivation from system semi-formal or strictly formal model				
Software architecture description				
Software constraints				
Traceability				
Executable				
Conformance to EN50128 § 7.2				
Conformance to EN50128 § 7.3				
Conformance to EN50128 § 7.4				

Which criteria for software architecture are covered by the methodology (see EN50128 table A.3) :

	Author	Assessor 1	Assessor 2	Total
Defensive programming				
Fault detection & diagnostic				
Error detecting code				
Failure assertion programming				
Diverse programming				
Memorising executed cases				
Software error effect analysis				
Fully defined interface				
Modelling				
Structured methodology				

D.7 Software code generation

This section discusses the usage of the approach for software code generation. It can be skipped depending the results of L.8.

Which criteria for software design and implementation are covered by the methodology (see EN50128 table A.4) :

	Author	Assessor 1	Assessor 2	Total
Formal methods				
Modeling				
Modular approach (mandatory)				
Components				
Design and coding standards (mandatory)				
Strongly typed programming language				

D.8 Main usage of the tool

This section discusses the main usage of the tool.

Which task are covered by the tool ?

	Author	Assessor 1	Assessor 2	Total
Modelling support				
Automatic translation				
Code Generation				
Model verification				
Test generation				
Simulation, execution, debugging				
Formal proof				

Modelling support

Does the tool provide a textual or a graphical editor ?

Automatic translation and code generation

Which translation or code generation is supported by the tool ?

Model verification

Which verification on models are provided by the tool?

Test generation

Does the tool allow to generate tests ? For which purpose ?

Simulation, execution, debugging

Does the tool allow to simulate or to debbug step by step a model or a code ?

Formal proof

Does the tool allow formal proof ? How ?

D.9 Use of the tool

According WP2 requirements, give a note for characteristics of the use of the tool (from 0 to 3) :

	Author	Assessor 1	Assessor 2	Total
Open Source (D2.6-X-36)				
Portability to operating systems (D2.6-X-37)				
Cooperation of tools (D2.6-X-38)				
Robustness (D2.6-X-41)				
Modularity (D2.6-X-41.1)				
Documentation management (D.2.6-X-41.2)				
Distributed software development (D.2.6-X-41.3)				
Simultaneous multi-users (D.2.6-X-41.4)				
Issue tracking (D.2.6-X-41.5)				
Differences between models (D.2.6-X-41.6)				
Version management (D.2.6-X-41.7)				
Concurrent version development (D.2.6-X-41.8)				
Model-based version control (D.2.6-X-41.9)				
Role traceability (D.2.6-X-41.10)				
Safety version traceability (D.2.6-X-41.11)				
Model traceability (D.2.6-01-035)				
Tool chain integration				
Scalability				

D.10 Certifiability

This section discusses how the tool can be classified according EN50128 requirements (D.2.6-X-50).

	Author	Assessor 1	Assessor 2	Total
Tool manual (D.2.6-01-42.02)				
Proof of correctness (D.2.6-01-42.03)				
Existing industrial usage				
Model verification				
Test generation				
Simulation, execution, debugging				
Formal proof				

Other elements for tool certification

D.11 Other comments

Please to give free comments on the approach.

Appendix E: SysML with Entreprise Architect

Author Author of the approaches description Cécile Braunstein (Uni. Bremen)

Assessor 1 First assessor of the approaches Uwe Steinke (Siemens)

Assessor 2 Second assessor of the approaches Roberto Kretschmer (TWT)

In the sequel, main text is under the responsibilities of the author.

Author. Author can add comments using this format at any place.

Assessor 1. First assessor can add comments using this format at any place.

Assessor 2. Second assessor can add comments using this format at any place.

When a note is required, please follow this list :

0 not recommended, not adapted, rejected

1 weakly recommended, adapted after major improvements, weakly rejected

2 recommended, adapted (with light improvements if necessary) weakly accepted

3 highly recommended, well adapted,strongly accepted

***** difficult to evaluate with a note (please add a comment under the table)

All the notes can be commented under each table.

E.1 Presentation

This section gives a quick presentation of the approach and the tool.

Name %%Name of the approach and the tool%%

Web site %%if available, how to find information%%

Licence %%Kind of licence%%

Abstract

Short abstract on the approach and tool (10 lines max)

Publications

Short list of publications on the approach (5 max)

E.2 Main usage of the approach

This section discusses the main usage of the approach.

According to the figure 1, for which phases do you recommend the approach (give a note from 0 to 3) :

	Author	Assessor 1	Assessor 2	Total
System Analysis				
Sub-system formal design				
Software design				
Software code generation				

According to the figure 1, for which type of activities do you recommend the approach (give a note from 0 to 3) :

	Author	Assessor 1	Assessor 2	Total
Documentation				
Modeling				
Design				
Code generation				
Verification				
Validation				
Safety analyses				

Known usages

Have you some examples of usage of this approach to compare with the OpenETCS objectives ?

E.3 Language

This section discusses the main element of the language.

Which are the main characteristics of the language :

	Author	Assessor 1	Assessor 2	Total
Informal language				
Semi-formal language				
Formal language				
Structured language				
Modular language				
Textual language				
Mathematical symbols or code				
Graphical language				

According WP2 requirements, give a note for the capabilities of the language (from 0 to 3) :

	Author	Assessor 1	Assessor 2	Total
Declarative formalization of properties (D.2.6-X-28)				
Simple formalization of properties (D.2.6-X-28.1)				
Scalability : capability to design large model				
Easily translatable to other languages (D.2.6-X-30)				
Executable directly (D.2.6-X-33)				
Executable after translation to a code (D.2.6-X-33) (precise if the translation is automatic)				
Simulation, animation (D.2.6-X-33)				
Easily understandable (D.2.6-X-27)				
Expertise level needed (0 High level, 3 few level)				
Standardization (D.2.6-X-29)				
Documented (D.2.6-X-29)				
Extensible language (D.2.6-01-28)				

Documentation

Describe how the language is documented, the existing guidelines, coding rules, standardization...

Language usage

Describe the possible restriction on the language

E.4 System Analysis

This section discusses the usage of the approach for system analysis. It can be skipped depending the results of L.8.

Acoording WP2 requirements, how the approach can be involved for the sub-system requirement specification ?

	Author	Assessor 1	Assessor 2	Total
Independent System functions definition (D.2.6-X-10.2.1)				
System architecture design (D.2.6-X-10.2)				
System data flow identification (D.2.6-X-10.2.3)				
Sub-system focus (D.2.6-X-10.2.4)				
System interfaces definition (D.2.6-X-10.2.5)				
System requirement allocation (D.2.6-X-10.3)				
Traceability with SRS (D.2.6-X-10.5)				
Traceability with Safety activities (D.2.6-X-11)				

E.5 Sub-System formal design

This section discusses the usage of the approach for sub-system formal design. It can be skipped depending the results of L.8.

Two kinds of model can be planned during this phase: semi-formal models to cover the SSRS (D.2.6-X-12.1) and strictly formal models to focuss on some functional and safety aspects (D.2.6-X-14). Obviously some strictly formal means can be used to define the semi-formal model.

E.5.1 Semi-formal model

Concerning semi-formal model, how the WP2 requirements are covered ?

	Author	Assessor 1	Assessor 2	Total
Consistency to SSRS (D.2.6-X-12.2)				
Coverage of SSRS (D.2.6-X-12.2.1)				
Coverage of SSHA (D.2.6-X-12.2.2)				
Management of requirement justification (D.2.6-X-12.2.3)				
Traceability to SSRS (D.2.6-X-12.2.5)				
Traceability of exported requirements (D.2.6-X-12.2.6)				
Simulation or animation (D.2.6-X-13 partial)				
Execution (D.2.6-X-13 partial)				
Extensible to strictly formal model (D.2.6-X-14.3)				
Easy to refine towards strictly formal model (D.2.6-X-14.4)				
Extensible and modular design (D.2.6-X-15)				
Extensible to software architecture and design (D.2.6-X-30)				

Concerning safety properties management, how the WP2 requirements are covered ?

	Author	Assessor 1	Assessor 2	Total
Safety function isolation (D.2.6-X-17)				
Safety properties formalisation (D.2.6-X-22)				
Logical expression (D.2.6-X-28.2.2)				
Timing constraints (D.2.6-X-28.2.3)				
Safety properties validation (D.2.6-X-23.2)				
Logical properties assertion (D.2.6-X-34)				
Check of assertions (D.2.6-X-34.1)				

Does the language allow to formalize (D.2.6-X-31):

	Author	Assessor 1	Assessor 2	Total
State machines				
Time-outs				
Truth tables				
Arithmetic				
Braking curves				
Logical statements				
Message and fields				

Additional comments on semi-formal model

Do you think your semi-formal model is sufficient to cover a safe design of the on-board unit until code generation ? All comments on links to other models, validation and verification activities are welcomed.

E.5.2 Strictly formal model

Concerning strictly formal model, how the WP2 requirements are covered ?

	Author	Assessor 1	Assessor 2	Total
Consistency to SFM (D.2.6-X-14.2)				
Coverage of SSRS (D.2.6-X-14.2)				
Traceability to SSRS (D.2.6-X-14.3)				
Extensible to software design (D.2.6-X-16)				
Safety function isolation (D.2.6-X-17)				
Safety properties formalisation (D.2.6-X-22)				
Logical expression (D.2.6-X-28.2.2)				
Timing constraints (D.2.6-X-28.2.3)				
Safety properties validation (D.2.6-X-23.3)				
Logical properties assertion (D.2.6-X-34)				
Proof of assertions (D.2.6-X-34.2)				

Does the language allow to formalize (D.2.6-X-32):

	Author	Assessor 1	Assessor 2	Total
State machines				
Time-outs				
Truth tables				
Arithmetic				
Braking curves				
Logical statements				
Message and fields				

Additional comments on semi-formal model

Do you think your strictly formal model can be directly defined from the SSRS ? All comments on links to other models, validation and verification activities are welcomed.

E.6 Software design

This section discusses the usage of the approach for software design. It can be skipped depending the results of L.8.

E.6.1 Functional design

How the approach allows to produce a functional software model of the on-board unit ?

	Author	Assessor 1	Assessor 2	Total
Derivation from system semi-formal model				
Software architecture description				
Software constraints				
Traceability				
Executable				

E.6.2 SSIL4 design

How the approach allows to produce in safety a software model ?

	Author	Assessor 1	Assessor 2	Total
Derivation from system semi-formal or strictly formal model				
Software architecture description				
Software constraints				
Traceability				
Executable				
Conformance to EN50128 § 7.2				
Conformance to EN50128 § 7.3				
Conformance to EN50128 § 7.4				

Which criteria for software architecture are covered by the methodology (see EN50128 table A.3) :

	Author	Assessor 1	Assessor 2	Total
Defensive programming				
Fault detection & diagnostic				
Error detecting code				
Failure assertion programming				
Diverse programming				
Memorising executed cases				
Software error effect analysis				
Fully defined interface				
Modelling				
Structured methodology				

E.7 Software code generation

This section discusses the usage of the approach for software code generation. It can be skipped depending the results of L.8.

Which criteria for software design and implementation are covered by the methodology (see EN50128 table A.4) :

	Author	Assessor 1	Assessor 2	Total
Formal methods				
Modeling				
Modular approach (mandatory)				
Components				
Design and coding standards (mandatory)				
Strongly typed programming language				

E.8 Main usage of the tool

This section discusses the main usage of the tool.

Which task are covered by the tool ?

	Author	Assessor 1	Assessor 2	Total
Modelling support				
Automatic translation				
Code Generation				
Model verification				
Test generation				
Simulation, execution, debugging				
Formal proof				

Modelling support

Does the tool provide a textual or a graphical editor ?

Automatic translation and code generation

Which translation or code generation is supported by the tool ?

Model verification

Which verification on models are provided by the tool?

Test generation

Does the tool allow to generate tests ? For which purpose ?

Simulation, execution, debugging

Does the tool allow to simulate or to debbug step by step a model or a code ?

Formal proof

Does the tool allow formal proof ? How ?

E.9 Use of the tool

According WP2 requirements, give a note for characteristics of the use of the tool (from 0 to 3) :

	Author	Assessor 1	Assessor 2	Total
Open Source (D2.6-X-36)				
Portability to operating systems (D2.6-X-37)				
Cooperation of tools (D2.6-X-38)				
Robustness (D2.6-X-41)				
Modularity (D2.6-X-41.1)				
Documentation management (D.2.6-X-41.2)				
Distributed software development (D.2.6-X-41.3)				
Simultaneous multi-users (D.2.6-X-41.4)				
Issue tracking (D.2.6-X-41.5)				
Differences between models (D.2.6-X-41.6)				
Version management (D.2.6-X-41.7)				
Concurrent version development (D.2.6-X-41.8)				
Model-based version control (D.2.6-X-41.9)				
Role traceability (D.2.6-X-41.10)				
Safety version traceability (D.2.6-X-41.11)				
Model traceability (D.2.6-01-035)				
Tool chain integration				
Scalability				

E.10 Certifiability

This section discusses how the tool can be classified according EN50128 requirements (D.2.6-X-50).

	Author	Assessor 1	Assessor 2	Total
Tool manual (D.2.6-01-42.02)				
Proof of correctness (D.2.6-01-42.03)				
Existing industrial usage				
Model verification				
Test generation				
Simulation, execution, debugging				
Formal proof				

Other elements for tool certification

E.11 Other comments

Please to give free comments on the approach.

Appendix F: SCADE

Author Author of the approaches description Uwe Steinke (Siemens)

Assessor 1 First assessor of the approaches David Mentré (MERCE)

Assessor 2 Second assessor of the approaches Cécile Braunstein (Uni. Bremen)

In the sequel, main text is under the responsibilities of the author.

Author. Author can add comments using this format at any place.

Assessor 1. First assessor can add comments using this format at any place.

Assessor 2. Second assessor can add comments using this format at any place.

When a note is required, please follow this list :

0 not recommended, not adapted, rejected

1 weakly recommended, adapted after major improvements, weakly rejected

2 recommended, adapted (with light improvements if necessary) weakly accepted

3 highly recommended, well adapted,strongly accepted

***** difficult to evaluate with a note (please add a comment under the table)

All the notes can be commented under each table.

F.1 Presentation

This section gives a quick presentation of the approach and the tool.

Name: SCADE Suite / SCADE System / SCADE LifeCycle

Web site: http://esterel-technologies.com

Licence: Commercial

Abstract

Short abstract on the approach and tool (10 lines max)

SCADE is a formal modelling language targeted for safety-critical embedded control applications in the avionics, rail, automotive and industrial automation domain. SCADE source code can be written as text (for anyone who likes writing plain text) or (more usual) as schematic diagrams.

SCADE models are synchronously clocked data flow and state machines, that can be nested and intermixed with each other without limitations. SCADE provides DO-178B- and EN50128-certified code generators producing C or ADA code as output. SCADE models are therefore concrete, deterministic, executable and verifiable; it allows the production of rapid prototype as well as of safety related target system software.

SCADE comes with an integrated development environment (SCADE Suite IDE) including code generator, graphical simulator, model checker/prover, model test coverage analyzer, report generators, version and requirements management gateway with interfaces to various other tools like static code and timing analyzers, System/SysML modelling tools etc.. The IDE provides automatization interfaces to be controlled from external tools, and all SCADE tools itself can also be used in batch mode. In addition, plugins for Eclipse integration are available.

The SCADE paradigm of synchronously clocked data flow and state machines works perfect for embedded control or industry automation software. It is less suitable for tasks like text processing or computer graphic applications. SCADE models do not only describe the structure of software; instead, they are the software implementation itself too. System architectures typically require a higher abstraction means of description at top level like SysML. While SysML modelling can be achieved with any SysML tools, SCADE System provides an automatic transformation from SysML to SCADE.

Publications

Short list of publications on the approach (5 max)

- http://www.interested-ip.eu/
- http://http://www.interested-ip.eu/final-report.html/

F.2 Main usage of the approach

This section discusses the main usage of the approach.

According to the figure 1, for which phases do you recommend the approach (give a note from 0 to 3) :

	Author	Assessor 1	Assessor 2	Total
System Analysis	1			
Sub-system formal design	3			
Software design	3			
Software code generation	3			

Author. *SCADE can be used for analyzing tasks on system level, especially to clarify complex system behaviour and functions by practical modelling, execution, simulation and test. For a higher abstraction level, this should be enhanced with system modelling languages as SysML.*

According to the figure 1, for which type of activities do you recommend the approach (give a note from 0 to 3) :

	Author	Assessor 1	Assessor 2	Total
Documentation	3			
Modeling	3			
Design	3			
Code generation	3			
Verification	3			
Validation	3			
Safety analyses	0			

Known usages

Have you some examples of usage of this approach to compare with the OpenETCS objectives ?

Field of usage: Safety critical systems like

- Rail interlocking systems
- Rail track vacancy detection systems
- Rail train control systems
- Rail Level-crossing protection systems
- Avionic flight controllers

F.3 Language

This section discusses the main element of the language.

Which are the main characteristics of the language :

	Author	Assessor 1	Assessor 2	Total
Informal language	0			
Semi-formal language	0			
Formal language	3			
Structured language	3			
Modular language	3			
Textual language	3			
Mathematical symbols or code	3			
Graphical language	3			

According WP2 requirements, give a note for the capabilities of the language (from 0 to 3) :

	Author	Assessor 1	Assessor 2	Total
Declarative formalization of properties (D.2.6-X-28)	2			
Simple formalization of properties (D.2.6-X-28.1)	2			
Scalability : capability to design large model	3			
Easily translatable to other languages (D.2.6-X-28)	3			
Executable directly (D.2.6-X-33)	3			
Executable after translation to a code (D.2.6-X-33) (precise if the translation is automatic)	3 3			
Simulation, animation (D.2.6-X-33)	3			
Easily understandable (D.2.6-X-27)	3			
Expertise level needed (0 High level, 3 few level)	2			
Standardization (D.2.6-X-29)	3			
Documented (D.2.6-X-29)	3			
Extensible language (D.2.6-01-28)	2			

Author. SCADE is a strictly textual and graphical formal language. It allows to be extended with user-defined operators. Especially the graphical representation is easy to learn and understand; nevertheless the rich tool suite covering most aspects of a EN50128 compliant process causes an appropriate learning effort by amount.

Documentation

Describe how the language is documented, the existing guidelines, coding rules, standardization...

SCADE provides a detailed documentation set:

- Getting Started
- SCADE Language Tutorial
- SCADE Suite User Manual
- SCADE Suite Technical Manual
- SCADE Suite Libraries Manual
- SCADE Language Primer
- SCADE Language Reference Manual
- Gateway Guidelines for LabView, Rhapsody, Simulink
- RTOS Adaptor Guidelines
- Timing and Stack Analysis Tools
- SCADE LifeCycle Documentation
- SCADE Suite Metamodels

- SCADE Glossary
- ...

Language usage

Describe the possible restriction on the language. SCADE is less suitable for tasks like text processing or computer graphic applications.

F.4 System Analysis

This section discusses the usage of the approach for system analysis. It can be skipped depending the results of L.8.

Acoording WP2 requirements, how the approach can be involved for the sub-system requirement specification ?

	Author	Assessor 1	Assessor 2	Total
Independent System functions definition (D.2.6-X-10.1.1)	3			
System architecture design (D.2.6-X-10.1.2)	1			
System data flow identification (D.2.6-X-10.2.3)	3			
Sub-system focus (D.2.6-X-10.2.4)	2			
System interfaces definition (D.2.6-X-10.2.5)	2			
System requirement allocation (D.2.6-X-10.3)	3			
Traceability with SRS (D.2.6-X-10.5)	3			
Traceability with Safety activities (D.2.6-X-11)	3			

Author. Although SCADE is not made for system analysis, it can be used for the following aspects on system level: Modelling of (separate) system functions, data flows, state machines and interfaces. It provides an excellent tracebility support between many different kinds of documents and other tools.

F.5 Sub-System formal design

This section discusses the usage of the approach for sub-system formal design. It can be skipped depending the results of L.8.

Two kinds of model can be planned during this phase: semi-formal models to cover the SSRS (D.2.6-X-12.1) and strictly formal models to focuss on some functional and safety aspects (D.2.6-X-14). Obviously some strictly formal means can be used to define the semi-formal model.

F.5.1 Semi-formal model

Author. SCADE models are formal. Since the following table addresses many aspects that SCADE covers in a formal way it is filled anyway. But keep in mind: it's formal - instead of semi-formal.

Concerning formal model, how the WP2 requirements are covered ?

	Author	Assessor 1	Assessor 2	Total
Consistency to SSRS (D.2.6-X-12.2)	2			
Coverage of SSRS (D.2.6-X-12.2.1)	3			
Coverage of SSHA (D.2.6-X-12.2.2)	3			
Management of requirement justification (D.2.6-X-12.2.3)	3			
Traceability to SSRS (D.2.6-X-12.2.5)	3			
Traceability of exported requirements (D.2.6-X-12.2.6)	3			
Simulation or animation (D.2.6-X-13 partial)	3			
Execution (D.2.6-X-13 partial)	3			
Extensible to strictly formal model (D.2.6-X-14.3)	3			
Easy to refine towards strictly formal model (D.2.6-X-14.4)	3			
Extensible and modular design (D.2.6-X-15)	3			
Extensible to software architecture and design (D.2.6-X-30)	3			

Concerning safety properties management, how the WP2 requirements are covered ?

	Author	Assessor 1	Assessor 2	Total
Safety function isolation (D.2.6-X-17)	2			
Safety properties formalisation (D.2.6-X-22)	2			
Logical expression (D.2.6-X-28.2.2)	3			
Timing constraints (D.2.6-X-28.2.3)	2			
Safety properties validation (D.2.6-X-23.2)	2			
Logical properties assertion (D.2.6-X-34)	2			
Check of assertions (D.2.6-X-34.1)	2			

Author. SCADE is a modelling language for functions. Therefore, only the functional aspects of properties are addressed.

Does the language allow to formalize (D.2.6-X-31):

	Author	Assessor 1	Assessor 2	Total
State machines	3			
Time-outs	3			
Truth tables	3			
Arithmetic	3			
Braking curves	3			
Logical statements	3			
Message and fields	2			

Additional comments on semi-formal model

Do you think your semi-formal model is sufficient to cover a safe design of the on-board unit until code generation ? All comments on links to other models, validation and verification activities are welcomed.

Author. *SCADE is targeted for these purposes.*

F.5.2 Strictly formal model

Concerning strictly formal model, how the WP2 requirements are covered ?

	Author	Assessor 1	Assessor 2	Total
Consistency to SFM (D.2.6-X-14.2)	3			
Coverage of SSRS (D.2.6-X-14.2)	3			
Traceability to SSRS (D.2.6-X-14.3)	3			
Extensible to software design (D.2.6-X-16)	3			
Safety function isolation (D.2.6-X-17)	3			
Safety properties formalisation (D.2.6-X-22)	2			
Logical expression (D.2.6-X-28.2.2)	3			
Timing constraints (D.2.6-X-28.2.3)	3			
Safety properties validation (D.2.6-X-23.3)	2			
Logical properties assertion (D.2.6-X-34)	2			
Proof of assertions (D.2.6-X-34.2)	2			

Does the language allow to formalize (D.2.6-X-32):

	Author	Assessor 1	Assessor 2	Total
State machines	3			
Time-outs	3			
Truth tables	3			
Arithmetic	3			
Braking curves	3			
Logical statements	3			
Message and fields	3			

Additional comments on semi-formal model

Do you think your strictly formal model can be directly defined from the SSRS ? All comments on links to other models, validation and verification activities are welcomed.

Author. *SCADE is targeted for these purposes.*

F.6 Software design

This section discusses the usage of the approach for software design. It can be skipped depending the results of L.8.

F.6.1 Functional design

How the approach allows to produce a functional software model of the on-board unit ?

	Author	Assessor 1	Assessor 2	Total
Derivation from system semi-formal model	3			
Software architecture description	3			
Software constraints	3			
Traceability	3			
Executable	3			

F.6.2 SSIL4 design

How the approach allows to produce in safety a software model ?

	Author	Assessor 1	Assessor 2	Total
Derivation from system semi-formal or strictly formal model	3			
Software architecture description	3			
Software constraints	3			
Traceability	3			
Executable	3			
Conformance to EN50128 § 7.2	3			
Conformance to EN50128 § 7.3	3			
Conformance to EN50128 § 7.4	3			

Which criteria for software architecture are covered by the methodology (see EN50128 table A.3) :

	Author	Assessor 1	Assessor 2	Total
Defensive programming	3			
Fault detection & diagnostic	0			
Error detecting code	0			
Failure assertion programming	1			
Diverse programming	0			
Memorising executed cases	0			
Software error effect analysis	0			
Fully defined interface	3			
Modelling	3			
Structured methodology	3			

F.7 Software code generation

This section discusses the usage of the approach for software code generation. It can be skipped depending the results of L.8.

Which criteria for software design and implementation are covered by the methodology (see EN50128 table A.4) :

	Author	Assessor 1	Assessor 2	Total
Formal methods	3			
Modeling	3			
Modular approach (mandatory)	3			
Components	3			
Design and coding standards (mandatory)	3			
Strongly typed programming language	3			

F.8 Main usage of the tool

This section discusses the main usage of the tool.

Which task are covered by the tool ?

	Author	Assessor 1	Assessor 2	Total
Modelling support	3			
Automatic translation	3			
Code Generation	3			
Model verification	3			
Test generation	2			
Simulation, execution, debugging	3			
Formal proof	3			

Modelling support

Does the tool provide a textual or a graphical editor ? Both.

Automatic translation and code generation

Which translation or code generation is supported by the tool ? Validated translation of SCADE models into C or ADA code.

Model verification

Which verification on models are provided by the tool? Simulation, animation, test via manual or script-based test suites. Model test coverage for structural model coverage measurement. Formal proving.

Test generation

Does the tool allow to generate tests ? For which purpose ? SCADE offers test suites to be built via test scripts. For automatic model based test case generation tools like RT-Tester are applicable.

Simulation, execution, debugging

Does the tool allow to simulate or to debug step by step a model or a code ? It allows simulation on a clock by clock base by executing the generated code while the model behaviour is visualized graphically. Graphical model debugging with breakpoint capabilities. Playback function for logfiles from the field.

Formal proof

Does the tool allow formal proof ? How ? SCADE integrates the Prover design verifier. The provable properties have to be modelled with SCADE Suite and connected to the target model in an observer configuration.

F.9 Use of the tool

According WP2 requirements, give a note for characteristics of the use of the tool (from 0 to 3) :

	Author	Assessor 1	Assessor 2	Total
Open Source (D2.6-X-36)	0			
Portability to operating systems (D2.6-X-37)	3			
Cooperation of tools (D2.6-X-38)	3			
Robustness (D2.6-X-41)	3			
Modularity (D2.6-X-41.1)	3			
Documentation management (D.2.6-X-41.2)	3			
Distributed software development (D.2.6-X-41.3)	2			
Simultaneous multi-users (D.2.6-X-41.4)	1			
Issue tracking (D.2.6-X-41.5)	0			
Differences between models (D.2.6-X-41.6)	3			
Version management (D.2.6-X-41.7)	3			
Concurrent version development (D.2.6-X-41.8)	2			
Model-based version control (D.2.6-X-41.9)	3			
Role traceability (D.2.6-X-41.10)	0			
Safety version traceability (D.2.6-X-41.11)	0			
Model traceability (D.2.6-01-035)	3			
Tool chain integration	3			
Scalability	3			

F.10 Certifiability

This section discusses how the tool can be classified according EN50128 requirements (D.2.6-X-50).

Author. SCADE is targeted to be certifiable. Validation documentation is available.

	Author	Assessor 1	Assessor 2	Total
Tool manual (D.2.6-01-42.02)	3			
Proof of correctness (D.2.6-01-42.03)	3			
Existing industrial usage	3			
Model verification	3			
Test generation	2			
Simulation, execution, debugging	3			
Formal proof	3			

Other elements for tool certification

F.11 Other comments

Please to give free comments on the approach.

Appendix G: EventB and Rodin

Author Author of the approaches description Matthias Gudemann (Systerel)

Assessor 1 First assessor of the approaches David Mentré (MERCE)

Assessor 2 Second assessor of the approaches Stanislas Pinte (ERTMS Solutions)

In the sequel, main text is under the responsibilities of the author.

Author. Author can add comments using this format at any place.

Assessor 1. First assessor can add comments using this format at any place.

Assessor 2. Second assessor can add comments using this format at any place.

When a note is required, please follow this list :

0 not recommended, not adapted, rejected

1 weakly recommended, adapted after major improvements, weakly rejected

2 recommended, adapted (with light improvements if necessary) weakly accepted

3 highly recommended, well adapted,strongly accepted

***** difficult to evaluate with a note (please add a comment under the table)

All the notes can be commented under each table.

G.1 Presentation

This section gives a quick presentation of the approach and the tool.

Name %%Name of the approach and the tool%%

Web site %%if available, how to find information%%

Licence %%Kind of licence%%

Abstract

Short abstract on the approach and tool (10 lines max)

Publications

Short list of publications on the approach (5 max)

G.2 Main usage of the approach

This section discusses the main usage of the approach.

According to the figure 1, for which phases do you recommend the approach (give a note from 0 to 3) :

	Author	Assessor 1	Assessor 2	Total
System Analysis				
Sub-system formal design				
Software design				
Software code generation				

According to the figure 1, for which type of activities do you recommend the approach (give a note from 0 to 3) :

	Author	Assessor 1	Assessor 2	Total
Documentation				
Modeling				
Design				
Code generation				
Verification				
Validation				
Safety analyses				

Known usages

Have you some examples of usage of this approach to compare with the OpenETCS objectives ?

G.3 Language

This section discusses the main element of the language.

Which are the main characteristics of the language :

	Author	Assessor 1	Assessor 2	Total
Informal language				
Semi-formal language				
Formal language				
Structured language				
Modular language				
Textual language				
Mathematical symbols or code				
Graphical language				

According WP2 requirements, give a note for the capabilities of the language (from 0 to 3) :

	Author	Assessor 1	Assessor 2	Total
Declarative formalization of properties (D.2.6-X-28)				
Simple formalization of properties (D.2.6-X-28.1)				
Scalability : capability to design large model				
Easily translatable to other languages (D.2.6-X-30)				
Executable directly (D.2.6-X-33)				
Executable after translation to a code (D.2.6-X-33) (precise if the translation is automatic)				
Simulation, animation (D.2.6-X-33)				
Easily understandable (D.2.6-X-27)				
Expertise level needed (0 High level, 3 few level)				
Standardization (D.2.6-X-29)				
Documented (D.2.6-X-29)				
Extensible language (D.2.6-01-28)				

Documentation

Describe how the language is documented, the existing guidelines, coding rules, standardization...

Language usage

Describe the possible restriction on the language

G.4 System Analysis

This section discusses the usage of the approach for system analysis. It can be skipped depending the results of L.8.

Acoording WP2 requirements, how the approach can be involved for the sub-system requirement specification ?

	Author	Assessor 1	Assessor 2	Total
Independent System functions definition (D.2.6-X-10.2.1)				
System architecture design (D.2.6-X-10.2)				
System data flow identification (D.2.6-X-10.2.3)				
Sub-system focus (D.2.6-X-10.2.4)				
System interfaces definition (D.2.6-X-10.2.5)				
System requirement allocation (D.2.6-X-10.3)				
Traceability with SRS (D.2.6-X-10.5)				
Traceability with Safety activities (D.2.6-X-11)				

G.5 Sub-System formal design

This section discusses the usage of the approach for sub-system formal design. It can be skipped depending the results of L.8.

Two kinds of model can be planned during this phase: semi-formal models to cover the SSRS (D.2.6-X-12.1) and strictly formal models to focuss on some functional and safety aspects (D.2.6-X-14). Obviously some strictly formal means can be used to define the semi-formal model.

G.5.1 Semi-formal model

Concerning semi-formal model, how the WP2 requirements are covered ?

	Author	Assessor 1	Assessor 2	Total
Consistency to SSRS (D.2.6-X-12.2)				
Coverage of SSRS (D.2.6-X-12.2.1)				
Coverage of SSHA (D.2.6-X-12.2.2)				
Management of requirement justification (D.2.6-X-12.2.3)				
Traceability to SSRS (D.2.6-X-12.2.5)				
Traceability of exported requirements (D.2.6-X-12.2.6)				
Simulation or animation (D.2.6-X-13 partial)				
Execution (D.2.6-X-13 partial)				
Extensible to strictly formal model (D.2.6-X-14.3)				
Easy to refine towards strictly formal model (D.2.6-X-14.4)				
Extensible and modular design (D.2.6-X-15)				
Extensible to software architecture and design (D.2.6-X-30)				

Concerning safety properties management, how the WP2 requirements are covered ?

	Author	Assessor 1	Assessor 2	Total
Safety function isolation (D.2.6-X-17)				
Safety properties formalisation (D.2.6-X-22)				
Logical expression (D.2.6-X-28.2.2)				
Timing constraints (D.2.6-X-28.2.3)				
Safety properties validation (D.2.6-X-23.2)				
Logical properties assertion (D.2.6-X-34)				
Check of assertions (D.2.6-X-34.1)				

Does the language allow to formalize (D.2.6-X-31):

	Author	Assessor 1	Assessor 2	Total
State machines				
Time-outs				
Truth tables				
Arithmetic				
Braking curves				
Logical statements				
Message and fields				

Additional comments on semi-formal model

Do you think your semi-formal model is sufficient to cover a safe design of the on-board unit until code generation ? All comments on links to other models, validation and verification activities are welcomed.

G.5.2 Strictly formal model

Concerning strictly formal model, how the WP2 requirements are covered ?

	Author	Assessor 1	Assessor 2	Total
Consistency to SFM (D.2.6-X-14.2)				
Coverage of SSRS (D.2.6-X-14.2)				
Traceability to SSRS (D.2.6-X-14.3)				
Extensible to software design (D.2.6-X-16)				
Safety function isolation (D.2.6-X-17)				
Safety properties formalisation (D.2.6-X-22)				
Logical expression (D.2.6-X-28.2.2)				
Timing constraints (D.2.6-X-28.2.3)				
Safety properties validation (D.2.6-X-23.3)				
Logical properties assertion (D.2.6-X-34)				
Proof of assertions (D.2.6-X-34.2)				

Does the language allow to formalize (D.2.6-X-32):

	Author	Assessor 1	Assessor 2	Total
State machines				
Time-outs				
Truth tables				
Arithmetic				
Braking curves				
Logical statements				
Message and fields				

Additional comments on semi-formal model

Do you think your strictly formal model can be directly defined from the SSRS ? All comments on links to other models, validation and verification activities are welcomed.

G.6 Software design

This section discusses the usage of the approach for software design. It can be skipped depending the results of L.8.

G.6.1 Functional design

How the approach allows to produce a functional software model of the on-board unit ?

	Author	Assessor 1	Assessor 2	Total
Derivation from system semi-formal model				
Software architecture description				
Software constraints				
Traceability				
Executable				

G.6.2 SSIL4 design

How the approach allows to produce in safety a software model ?

	Author	Assessor 1	Assessor 2	Total
Derivation from system semi-formal or strictly formal model				
Software architecture description				
Software constraints				
Traceability				
Executable				
Conformance to EN50128 § 7.2				
Conformance to EN50128 § 7.3				
Conformance to EN50128 § 7.4				

Which criteria for software architecture are covered by the methodology (see EN50128 table A.3) :

	Author	Assessor 1	Assessor 2	Total
Defensive programming				
Fault detection & diagnostic				
Error detecting code				
Failure assertion programming				
Diverse programming				
Memorising executed cases				
Software error effect analysis				
Fully defined interface				
Modelling				
Structured methodology				

G.7 Software code generation

This section discusses the usage of the approach for software code generation. It can be skipped depending the results of L.8.

Which criteria for software design and implementation are covered by the methodology (see EN50128 table A.4) :

	Author	Assessor 1	Assessor 2	Total
Formal methods				
Modeling				
Modular approach (mandatory)				
Components				
Design and coding standards (mandatory)				
Strongly typed programming language				

G.8 Main usage of the tool

This section discusses the main usage of the tool.

Which task are covered by the tool ?

	Author	Assessor 1	Assessor 2	Total
Modelling support				
Automatic translation				
Code Generation				
Model verification				
Test generation				
Simulation, execution, debugging				
Formal proof				

Modelling support

Does the tool provide a textual or a graphical editor ?

Automatic translation and code generation

Which translation or code generation is supported by the tool ?

Model verification

Which verification on models are provided by the tool?

Test generation

Does the tool allow to generate tests ? For which purpose ?

Simulation, execution, debugging

Does the tool allow to simulate or to debbug step by step a model or a code ?

Formal proof

Does the tool allow formal proof ? How ?

G.9 Use of the tool

According WP2 requirements, give a note for characteristics of the use of the tool (from 0 to 3) :

	Author	Assessor 1	Assessor 2	Total
Open Source (D2.6-X-36)				
Portability to operating systems (D2.6-X-37)				
Cooperation of tools (D2.6-X-38)				
Robustness (D2.6-X-41)				
Modularity (D2.6-X-41.1)				
Documentation management (D.2.6-X-41.2)				
Distributed software development (D.2.6-X-41.3)				
Simultaneous multi-users (D.2.6-X-41.4)				
Issue tracking (D.2.6-X-41.5)				
Differences between models (D.2.6-X-41.6)				
Version management (D.2.6-X-41.7)				
Concurrent version development (D.2.6-X-41.8)				
Model-based version control (D.2.6-X-41.9)				
Role traceability (D.2.6-X-41.10)				
Safety version traceability (D.2.6-X-41.11)				
Model traceability (D.2.6-01-035)				
Tool chain integration				
Scalability				

G.10 Certifiability

This section discusses how the tool can be classified according EN50128 requirements (D.2.6-X-50).

	Author	Assessor 1	Assessor 2	Total
Tool manual (D.2.6-01-42.02)				
Proof of correctness (D.2.6-01-42.03)				
Existing industrial usage				
Model verification				
Test generation				
Simulation, execution, debugging				
Formal proof				

Other elements for tool certification

G.11 Other comments

Please to give free comments on the approach.

Appendix H: Classical B and Atelier B

Author Author of the approaches description Marielle Petit-Doche (Systerel)

Assessor 1 First assessor of the approaches Peter Mahlmann (DB)

Assessor 2 Second assessor of the approaches %%Name - Company%%

In the sequel, main text is under the responsibilities of the author.

Author. Author can add comments using this format at any place.

Assessor 1. First assessor can add comments using this format at any place.

Assessor 2. Second assessor can add comments using this format at any place.

When a note is required, please follow this list :

0 not recommended, not adapted, rejected

1 weakly recommended, adapted after major improvements, weakly rejected

2 recommended, adapted (with light improvements if necessary) weakly accepted

3 highly recommended, well adapted,strongly accepted

***** difficult to evaluate with a note (please add a comment under the table)

All the notes can be commented under each table.

H.1 Presentation

This section gives a quick presentation of the approach and the tool.

Name %%Name of the approach and the tool%%

Web site %%if available, how to find information%%

Licence %%Kind of licence%%

Abstract

Short abstract on the approach and tool (10 lines max)

Publications

Short list of publications on the approach (5 max)

H.2 Main usage of the approach

This section discusses the main usage of the approach.

According to the figure 1, for which phases do you recommend the approach (give a note from 0 to 3) :

	Author	Assessor 1	Assessor 2	Total
System Analysis				
Sub-system formal design				
Software design				
Software code generation				

According to the figure 1, for which type of activities do you recommend the approach (give a note from 0 to 3) :

	Author	Assessor 1	Assessor 2	Total
Documentation				
Modeling				
Design				
Code generation				
Verification				
Validation				
Safety analyses				

Known usages

Have you some examples of usage of this approach to compare with the OpenETCS objectives ?

H.3 Language

This section discusses the main element of the language.

Which are the main characteristics of the language :

	Author	Assessor 1	Assessor 2	Total
Informal language				
Semi-formal language				
Formal language				
Structured language				
Modular language				
Textual language				
Mathematical symbols or code				
Graphical language				

According WP2 requirements, give a note for the capabilities of the language (from 0 to 3) :

	Author	Assessor 1	Assessor 2	Total
Declarative formalization of properties (D.2.6-X-28)				
Simple formalization of properties (D.2.6-X-28.1)				
Scalability : capability to design large model				
Easily translatable to other languages (D.2.6-X-30)				
Executable directly (D.2.6-X-33)				
Executable after translation to a code (D.2.6-X-33) (precise if the translation is automatic)				
Simulation, animation (D.2.6-X-33)				
Easily understandable (D.2.6-X-27)				
Expertise level needed (0 High level, 3 few level)				
Standardization (D.2.6-X-29)				
Documented (D.2.6-X-29)				
Extensible language (D.2.6-01-28)				

Documentation

Describe how the language is documented, the existing guidelines, coding rules, standardization...

Language usage

Describe the possible restriction on the language

H.4 System Analysis

This section discusses the usage of the approach for system analysis. It can be skipped depending the results of L.8.

Acoording WP2 requirements, how the approach can be involved for the sub-system requirement specification ?

	Author	Assessor 1	Assessor 2	Total
Independent System functions definition (D.2.6-X-10.2.1)				
System architecture design (D.2.6-X-10.2)				
System data flow identification (D.2.6-X-10.2.3)				
Sub-system focus (D.2.6-X-10.2.4)				
System interfaces definition (D.2.6-X-10.2.5)				
System requirement allocation (D.2.6-X-10.3)				
Traceability with SRS (D.2.6-X-10.5)				
Traceability with Safety activities (D.2.6-X-11)				

H.5 Sub-System formal design

This section discusses the usage of the approach for sub-system formal design. It can be skipped depending the results of L.8.

Two kinds of model can be planned during this phase: semi-formal models to cover the SSRS (D.2.6-X-12.1) and strictly formal models to focuss on some functional and safety aspects (D.2.6-X-14). Obviously some strictly formal means can be used to define the semi-formal model.

H.5.1 Semi-formal model

Concerning semi-formal model, how the WP2 requirements are covered ?

	Author	Assessor 1	Assessor 2	Total
Consistency to SSRS (D.2.6-X-12.2)				
Coverage of SSRS (D.2.6-X-12.2.1)				
Coverage of SSHA (D.2.6-X-12.2.2)				
Management of requirement justification (D.2.6-X-12.2.3)				
Traceability to SSRS (D.2.6-X-12.2.5)				
Traceability of exported requirements (D.2.6-X-12.2.6)				
Simulation or animation (D.2.6-X-13 partial)				
Execution (D.2.6-X-13 partial)				
Extensible to strictly formal model (D.2.6-X-14.3)				
Easy to refine towards strictly formal model (D.2.6-X-14.4)				
Extensible and modular design (D.2.6-X-15)				
Extensible to software architecture and design (D.2.6-X-30)				

Concerning safety properties management, how the WP2 requirements are covered ?

	Author	Assessor 1	Assessor 2	Total
Safety function isolation (D.2.6-X-17)				
Safety properties formalisation (D.2.6-X-22)				
Logical expression (D.2.6-X-28.2.2)				
Timing constraints (D.2.6-X-28.2.3)				
Safety properties validation (D.2.6-X-23.2)				
Logical properties assertion (D.2.6-X-34)				
Check of assertions (D.2.6-X-34.1)				

Does the language allow to formalize (D.2.6-X-31):

	Author	Assessor 1	Assessor 2	Total
State machines				
Time-outs				
Truth tables				
Arithmetic				
Braking curves				
Logical statements				
Message and fields				

Additional comments on semi-formal model

Do you think your semi-formal model is sufficient to cover a safe design of the on-board unit until code generation ? All comments on links to other models, validation and verification activities are welcomed.

H.5.2 Strictly formal model

Concerning strictly formal model, how the WP2 requirements are covered ?

	Author	Assessor 1	Assessor 2	Total
Consistency to SFM (D.2.6-X-14.2)				
Coverage of SSRS (D.2.6-X-14.2)				
Traceability to SSRS (D.2.6-X-14.3)				
Extensible to software design (D.2.6-X-16)				
Safety function isolation (D.2.6-X-17)				
Safety properties formalisation (D.2.6-X-22)				
Logical expression (D.2.6-X-28.2.2)				
Timing constraints (D.2.6-X-28.2.3)				
Safety properties validation (D.2.6-X-23.3)				
Logical properties assertion (D.2.6-X-34)				
Proof of assertions (D.2.6-X-34.2)				

Does the language allow to formalize (D.2.6-X-32):

	Author	Assessor 1	Assessor 2	Total
State machines				
Time-outs				
Truth tables				
Arithmetic				
Braking curves				
Logical statements				
Message and fields				

Additional comments on semi-formal model

Do you think your strictly formal model can be directly defined from the SSRS ? All comments on links to other models, validation and verification activities are welcomed.

H.6 Software design

This section discusses the usage of the approach for software design. It can be skipped depending the results of L.8.

H.6.1 Functional design

How the approach allows to produce a functional software model of the on-board unit ?

	Author	Assessor 1	Assessor 2	Total
Derivation from system semi-formal model				
Software architecture description				
Software constraints				
Traceability				
Executable				

H.6.2 SSIL4 design

How the approach allows to produce in safety a software model ?

	Author	Assessor 1	Assessor 2	Total
Derivation from system semi-formal or strictly formal model				
Software architecture description				
Software constraints				
Traceability				
Executable				
Conformance to EN50128 § 7.2				
Conformance to EN50128 § 7.3				
Conformance to EN50128 § 7.4				

Which criteria for software architecture are covered by the methodology (see EN50128 table A.3) :

	Author	Assessor 1	Assessor 2	Total
Defensive programming				
Fault detection & diagnostic				
Error detecting code				
Failure assertion programming				
Diverse programming				
Memorising executed cases				
Software error effect analysis				
Fully defined interface				
Modelling				
Structured methodology				

H.7 Software code generation

This section discusses the usage of the approach for software code generation. It can be skipped depending the results of L.8.

Which criteria for software design and implementation are covered by the methodology (see EN50128 table A.4) :

	Author	Assessor 1	Assessor 2	Total
Formal methods				
Modeling				
Modular approach (mandatory)				
Components				
Design and coding standards (mandatory)				
Strongly typed programming language				

H.8 Main usage of the tool

This section discusses the main usage of the tool.

Which task are covered by the tool ?

	Author	Assessor 1	Assessor 2	Total
Modelling support				
Automatic translation				
Code Generation				
Model verification				
Test generation				
Simulation, execution, debugging				
Formal proof				

Modelling support

Does the tool provide a textual or a graphical editor ?

Automatic translation and code generation

Which translation or code generation is supported by the tool ?

Model verification

Which verification on models are provided by the tool?

Test generation

Does the tool allow to generate tests ? For which purpose ?

Simulation, execution, debugging

Does the tool allow to simulate or to debbug step by step a model or a code ?

Formal proof

Does the tool allow formal proof ? How ?

H.9 Use of the tool

According WP2 requirements, give a note for characteristics of the use of the tool (from 0 to 3) :

	Author	Assessor 1	Assessor 2	Total
Open Source (D2.6-X-36)				
Portability to operating systems (D2.6-X-37)				
Cooperation of tools (D2.6-X-38)				
Robustness (D2.6-X-41)				
Modularity (D2.6-X-41.1)				
Documentation management (D.2.6-X-41.2)				
Distributed software development (D.2.6-X-41.3)				
Simultaneous multi-users (D.2.6-X-41.4)				
Issue tracking (D.2.6-X-41.5)				
Differences between models (D.2.6-X-41.6)				
Version management (D.2.6-X-41.7)				
Concurrent version development (D.2.6-X-41.8)				
Model-based version control (D.2.6-X-41.9)				
Role traceability (D.2.6-X-41.10)				
Safety version traceability (D.2.6-X-41.11)				
Model traceability (D.2.6-01-035)				
Tool chain integration				
Scalability				

H.10 Certifiability

This section discusses how the tool can be classified according EN50128 requirements (D.2.6-X-50).

	Author	Assessor 1	Assessor 2	Total
Tool manual (D.2.6-01-42.02)				
Proof of correctness (D.2.6-01-42.03)				
Existing industrial usage				
Model verification				
Test generation				
Simulation, execution, debugging				
Formal proof				

Other elements for tool certification

H.11 Other comments

Please to give free comments on the approach.

Appendix I: Petri Nets

Author Author of the approaches description Jan Welte (TU-BS)

Assessor 1 First assessor of the approaches %%Name - Company%%

Assessor 2 Second assessor of the approaches %%Name - Company%%

In the sequel, main text is under the responsibilities of the author.

Author. Author can add comments using this format at any place.

Assessor 1. First assessor can add comments using this format at any place.

Assessor 2. Second assessor can add comments using this format at any place.

When a note is required, please follow this list :

0 not recommended, not adapted, rejected

1 weakly recommended, adapted after major improvements, weakly rejected

2 recommended, adapted (with light improvements if necessary) weakly accepted

3 highly recommended, well adapted,strongly accepted

***** difficult to evaluate with a note (please add a comment under the table)

All the notes can be commented under each table.

I.1 Presentation

This section gives a quick presentation of the approach and the tool.

Name %%Name of the approach and the tool%%

Web site %%if available, how to find information%%

Licence %%Kind of licence%%

Abstract

Short abstract on the approach and tool (10 lines max)

Publications

Short list of publications on the approach (5 max)

I.2 Main usage of the approach

This section discusses the main usage of the approach.

According to the figure 1, for which phases do you recommend the approach (give a note from 0 to 3) :

	Author	Assessor 1	Assessor 2	Total
System Analysis				
Sub-system formal design				
Software design				
Software code generation				

According to the figure 1, for which type of activities do you recommend the approach (give a note from 0 to 3) :

	Author	Assessor 1	Assessor 2	Total
Documentation				
Modeling				
Design				
Code generation				
Verification				
Validation				
Safety analyses				

Known usages

Have you some examples of usage of this approach to compare with the OpenETCS objectives ?

I.3 Language

This section discusses the main element of the language.

Which are the main characteristics of the language :

	Author	Assessor 1	Assessor 2	Total
Informal language				
Semi-formal language				
Formal language				
Structured language				
Modular language				
Textual language				
Mathematical symbols or code				
Graphical language				

According WP2 requirements, give a note for the capabilities of the language (from 0 to 3) :

	Author	Assessor 1	Assessor 2	Total
Declarative formalization of properties (D.2.6-X-28)				
Simple formalization of properties (D.2.6-X-28.1)				
Scalability : capability to design large model				
Easily translatable to other languages (D.2.6-X-30)				
Executable directly (D.2.6-X-33)				
Executable after translation to a code (D.2.6-X-33) (precise if the translation is automatic)				
Simulation, animation (D.2.6-X-33)				
Easily understandable (D.2.6-X-27)				
Expertise level needed (0 High level, 3 few level)				
Standardization (D.2.6-X-29)				
Documented (D.2.6-X-29)				
Extensible language (D.2.6-01-28)				

Documentation

Describe how the language is documented, the existing guidelines, coding rules, standardization...

Language usage

Describe the possible restriction on the language

I.4 System Analysis

This section discusses the usage of the approach for system analysis. It can be skipped depending the results of L.8.

Acoording WP2 requirements, how the approach can be involved for the sub-system requirement specification ?

	Author	Assessor 1	Assessor 2	Total
Independent System functions definition (D.2.6-X-10.2.1)				
System architecture design (D.2.6-X-10.2)				
System data flow identification (D.2.6-X-10.2.3)				
Sub-system focus (D.2.6-X-10.2.4)				
System interfaces definition (D.2.6-X-10.2.5)				
System requirement allocation (D.2.6-X-10.3)				
Traceability with SRS (D.2.6-X-10.5)				
Traceability with Safety activities (D.2.6-X-11)				

I.5 Sub-System formal design

This section discusses the usage of the approach for sub-system formal design. It can be skipped depending the results of L.8.

Two kinds of model can be planned during this phase: semi-formal models to cover the SSRS (D.2.6-X-12.1) and strictly formal models to focuss on some functional and safety aspects (D.2.6-X-14). Obviously some strictly formal means can be used to define the semi-formal model.

I.5.1 Semi-formal model

Concerning semi-formal model, how the WP2 requirements are covered ?

	Author	Assessor 1	Assessor 2	Total
Consistency to SSRS (D.2.6-X-12.2)				
Coverage of SSRS (D.2.6-X-12.2.1)				
Coverage of SSHA (D.2.6-X-12.2.2)				
Management of requirement justification (D.2.6-X-12.2.3)				
Traceability to SSRS (D.2.6-X-12.2.5)				
Traceability of exported requirements (D.2.6-X-12.2.6)				
Simulation or animation (D.2.6-X-13 partial)				
Execution (D.2.6-X-13 partial)				
Extensible to strictly formal model (D.2.6-X-14.3)				
Easy to refine towards strictly formal model (D.2.6-X-14.4)				
Extensible and modular design (D.2.6-X-15)				
Extensible to software architecture and design (D.2.6-X-30)				

Concerning safety properties management, how the WP2 requirements are covered ?

	Author	Assessor 1	Assessor 2	Total
Safety function isolation (D.2.6-X-17)				
Safety properties formalisation (D.2.6-X-22)				
Logical expression (D.2.6-X-28.2.2)				
Timing constraints (D.2.6-X-28.2.3)				
Safety properties validation (D.2.6-X-23.2)				
Logical properties assertion (D.2.6-X-34)				
Check of assertions (D.2.6-X-34.1)				

Does the language allow to formalize (D.2.6-X-31):

	Author	Assessor 1	Assessor 2	Total
State machines				
Time-outs				
Truth tables				
Arithmetic				
Braking curves				
Logical statements				
Message and fields				

Additional comments on semi-formal model

Do you think your semi-formal model is sufficient to cover a safe design of the on-board unit until code generation ? All comments on links to other models, validation and verification activities are welcomed.

I.5.2 Strictly formal model

Concerning strictly formal model, how the WP2 requirements are covered ?

	Author	Assessor 1	Assessor 2	Total
Consistency to SFM (D.2.6-X-14.2)				
Coverage of SSRS (D.2.6-X-14.2)				
Traceability to SSRS (D.2.6-X-14.3)				
Extensible to software design (D.2.6-X-16)				
Safety function isolation (D.2.6-X-17)				
Safety properties formalisation (D.2.6-X-22)				
Logical expression (D.2.6-X-28.2.2)				
Timing constraints (D.2.6-X-28.2.3)				
Safety properties validation (D.2.6-X-23.3)				
Logical properties assertion (D.2.6-X-34)				
Proof of assertions (D.2.6-X-34.2)				

Does the language allow to formalize (D.2.6-X-32):

	Author	Assessor 1	Assessor 2	Total
State machines				
Time-outs				
Truth tables				
Arithmetic				
Braking curves				
Logical statements				
Message and fields				

Additional comments on semi-formal model

Do you think your strictly formal model can be directly defined from the SSRS ? All comments on links to other models, validation and verification activities are welcomed.

I.6 Software design

This section discusses the usage of the approach for software design. It can be skipped depending the results of L.8.

I.6.1 Functional design

How the approach allows to produce a functional software model of the on-board unit ?

	Author	Assessor 1	Assessor 2	Total
Derivation from system semi-formal model				
Software architecture description				
Software constraints				
Traceability				
Executable				

I.6.2 SSIL4 design

How the approach allows to produce in safety a software model ?

	Author	Assessor 1	Assessor 2	Total
Derivation from system semi-formal or strictly formal model				
Software architecture description				
Software constraints				
Traceability				
Executable				
Conformance to EN50128 § 7.2				
Conformance to EN50128 § 7.3				
Conformance to EN50128 § 7.4				

Which criteria for software architecture are covered by the methodology (see EN50128 table A.3) :

	Author	Assessor 1	Assessor 2	Total
Defensive programming				
Fault detection & diagnostic				
Error detecting code				
Failure assertion programming				
Diverse programming				
Memorising executed cases				
Software error effect analysis				
Fully defined interface				
Modelling				
Structured methodology				

I.7 Software code generation

This section discusses the usage of the approach for software code generation. It can be skipped depending the results of L.8.

Which criteria for software design and implementation are covered by the methodology (see EN50128 table A.4) :

	Author	Assessor 1	Assessor 2	Total
Formal methods				
Modeling				
Modular approach (mandatory)				
Components				
Design and coding standards (mandatory)				
Strongly typed programming language				

I.8 Main usage of the tool

This section discusses the main usage of the tool.

Which task are covered by the tool ?

	Author	Assessor 1	Assessor 2	Total
Modelling support				
Automatic translation				
Code Generation				
Model verification				
Test generation				
Simulation, execution, debugging				
Formal proof				

Modelling support

Does the tool provide a textual or a graphical editor ?

Automatic translation and code generation

Which translation or code generation is supported by the tool ?

Model verification

Which verification on models are provided by the tool?

Test generation

Does the tool allow to generate tests ? For which purpose ?

Simulation, execution, debugging

Does the tool allow to simulate or to debbug step by step a model or a code ?

Formal proof

Does the tool allow formal proof ? How ?

I.9 Use of the tool

According WP2 requirements, give a note for characteristics of the use of the tool (from 0 to 3) :

	Author	Assessor 1	Assessor 2	Total
Open Source (D2.6-X-36)				
Portability to operating systems (D2.6-X-37)				
Cooperation of tools (D2.6-X-38)				
Robustness (D2.6-X-41)				
Modularity (D2.6-X-41.1)				
Documentation management (D.2.6-X-41.2)				
Distributed software development (D.2.6-X-41.3)				
Simultaneous multi-users (D.2.6-X-41.4)				
Issue tracking (D.2.6-X-41.5)				
Differences between models (D.2.6-X-41.6)				
Version management (D.2.6-X-41.7)				
Concurrent version development (D.2.6-X-41.8)				
Model-based version control (D.2.6-X-41.9)				
Role traceability (D.2.6-X-41.10)				
Safety version traceability (D.2.6-X-41.11)				
Model traceability (D.2.6-01-035)				
Tool chain integration				
Scalability				

I.10 Certifiability

This section discusses how the tool can be classified according EN50128 requirements (D.2.6-X-50).

	Author	Assessor 1	Assessor 2	Total
Tool manual (D.2.6-01-42.02)				
Proof of correctness (D.2.6-01-42.03)				
Existing industrial usage				
Model verification				
Test generation				
Simulation, execution, debugging				
Formal proof				

Other elements for tool certification

I.11 Other comments

Please to give free comments on the approach.

Appendix J: System C

Author Author of the approaches description Stefan Rieger (TWT)/ Frank Golatowski (Uni Rostock)

Assessor 1 First assessor of the approaches Cecile Braunstein (Uni. Bremen)

Assessor 2 Second assessor of the approaches Silvano DalZilio / LAAS

In the sequel, main text is under the responsibilities of the author.

Author. Author can add comments using this format at any place.

Assessor 1. First assessor can add comments using this format at any place.

Assessor 2. Second assessor can add comments using this format at any place.

When a note is required, please follow this list :

0 not recommended, not adapted, rejected

1 weakly recommended, adapted after major improvements, weakly rejected

2 recommended, adapted (with light improvements if necessary) weakly accepted

3 highly recommended, well adapted,strongly accepted

***** difficult to evaluate with a note (please add a comment under the table)

All the notes can be commented under each table.

J.1 Presentation

This section gives a quick presentation of the approach and the tool.

Name %%Name of the approach and the tool%%

Web site %%if available, how to find information%%

Licence %%Kind of licence%%

Abstract

Short abstract on the approach and tool (10 lines max)

Publications

Short list of publications on the approach (5 max)

J.2 Main usage of the approach

This section discusses the main usage of the approach.

According to the figure 1, for which phases do you recommend the approach (give a note from 0 to 3) :

	Author	Assessor 1	Assessor 2	Total
System Analysis				
Sub-system formal design				
Software design				
Software code generation				

According to the figure 1, for which type of activities do you recommend the approach (give a note from 0 to 3) :

	Author	Assessor 1	Assessor 2	Total
Documentation				
Modeling				
Design				
Code generation				
Verification				
Validation				
Safety analyses				

Known usages

Have you some examples of usage of this approach to compare with the OpenETCS objectives ?

J.3 Language

This section discusses the main element of the language.

Which are the main characteristics of the language :

	Author	Assessor 1	Assessor 2	Total
Informal language				
Semi-formal language				
Formal language				
Structured language				
Modular language				
Textual language				
Mathematical symbols or code				
Graphical language				

According WP2 requirements, give a note for the capabilities of the language (from 0 to 3) :

	Author	Assessor 1	Assessor 2	Total
Declarative formalization of properties (D.2.6-X-28)				
Simple formalization of properties (D.2.6-X-28.1)				
Scalability : capability to design large model				
Easily translatable to other languages (D.2.6-X-30)				
Executable directly (D.2.6-X-33)				
Executable after translation to a code (D.2.6-X-33) (precise if the translation is automatic)				
Simulation, animation (D.2.6-X-33)				
Easily understandable (D.2.6-X-27)				
Expertise level needed (0 High level, 3 few level)				
Standardization (D.2.6-X-29)				
Documented (D.2.6-X-29)				
Extensible language (D.2.6-01-28)				

Documentation

Describe how the language is documented, the existing guidelines, coding rules, standardization...

Language usage

Describe the possible restriction on the language

J.4 System Analysis

This section discusses the usage of the approach for system analysis. It can be skipped depending the results of L.8.

Acoording WP2 requirements, how the approach can be involved for the sub-system requirement specification ?

	Author	Assessor 1	Assessor 2	Total
Independent System functions definition (D.2.6-X-10.2.1)				
System architecture design (D.2.6-X-10.2)				
System data flow identification (D.2.6-X-10.2.3)				
Sub-system focus (D.2.6-X-10.2.4)				
System interfaces definition (D.2.6-X-10.2.5)				
System requirement allocation (D.2.6-X-10.3)				
Traceability with SRS (D.2.6-X-10.5)				
Traceability with Safety activities (D.2.6-X-11)				

J.5 Sub-System formal design

This section discusses the usage of the approach for sub-system formal design. It can be skipped depending the results of L.8.

Two kinds of model can be planned during this phase: semi-formal models to cover the SSRS (D.2.6-X-12.1) and strictly formal models to focuss on some functional and safety aspects (D.2.6-X-14). Obviously some strictly formal means can be used to define the semi-formal model.

J.5.1 Semi-formal model

Concerning semi-formal model, how the WP2 requirements are covered ?

	Author	Assessor 1	Assessor 2	Total
Consistency to SSRS (D.2.6-X-12.2)				
Coverage of SSRS (D.2.6-X-12.2.1)				
Coverage of SSHA (D.2.6-X-12.2.2)				
Management of requirement justification (D.2.6-X-12.2.3)				
Traceability to SSRS (D.2.6-X-12.2.5)				
Traceability of exported requirements (D.2.6-X-12.2.6)				
Simulation or animation (D.2.6-X-13 partial)				
Execution (D.2.6-X-13 partial)				
Extensible to strictly formal model (D.2.6-X-14.3)				
Easy to refine towards strictly formal model (D.2.6-X-14.4)				
Extensible and modular design (D.2.6-X-15)				
Extensible to software architecture and design (D.2.6-X-30)				

Concerning safety properties management, how the WP2 requirements are covered ?

	Author	Assessor 1	Assessor 2	Total
Safety function isolation (D.2.6-X-17)				
Safety properties formalisation (D.2.6-X-22)				
Logical expression (D.2.6-X-28.2.2)				
Timing constraints (D.2.6-X-28.2.3)				
Safety properties validation (D.2.6-X-23.2)				
Logical properties assertion (D.2.6-X-34)				
Check of assertions (D.2.6-X-34.1)				

Does the language allow to formalize (D.2.6-X-31):

	Author	Assessor 1	Assessor 2	Total
State machines				
Time-outs				
Truth tables				
Arithmetic				
Braking curves				
Logical statements				
Message and fields				

Additional comments on semi-formal model

Do you think your semi-formal model is sufficient to cover a safe design of the on-board unit until code generation ? All comments on links to other models, validation and verification activities are welcomed.

J.5.2 Strictly formal model

Concerning strictly formal model, how the WP2 requirements are covered ?

	Author	Assessor 1	Assessor 2	Total
Consistency to SFM (D.2.6-X-14.2)				
Coverage of SSRS (D.2.6-X-14.2)				
Traceability to SSRS (D.2.6-X-14.3)				
Extensible to software design (D.2.6-X-16)				
Safety function isolation (D.2.6-X-17)				
Safety properties formalisation (D.2.6-X-22)				
Logical expression (D.2.6-X-28.2.2)				
Timing constraints (D.2.6-X-28.2.3)				
Safety properties validation (D.2.6-X-23.3)				
Logical properties assertion (D.2.6-X-34)				
Proof of assertions (D.2.6-X-34.2)				

Does the language allow to formalize (D.2.6-X-32):

	Author	Assessor 1	Assessor 2	Total
State machines				
Time-outs				
Truth tables				
Arithmetic				
Braking curves				
Logical statements				
Message and fields				

Additional comments on semi-formal model

Do you think your strictly formal model can be directly defined from the SSRS ? All comments on links to other models, validation and verification activities are welcomed.

J.6 Software design

This section discusses the usage of the approach for software design. It can be skipped depending the results of L.8.

J.6.1 Functional design

How the approach allows to produce a functional software model of the on-board unit ?

	Author	Assessor 1	Assessor 2	Total
Derivation from system semi-formal model				
Software architecture description				
Software constraints				
Traceability				
Executable				

J.6.2 SSIL4 design

How the approach allows to produce in safety a software model ?

	Author	Assessor 1	Assessor 2	Total
Derivation from system semi-formal or strictly formal model				
Software architecture description				
Software constraints				
Traceability				
Executable				
Conformance to EN50128 § 7.2				
Conformance to EN50128 § 7.3				
Conformance to EN50128 § 7.4				

Which criteria for software architecture are covered by the methodology (see EN50128 table A.3) :

	Author	Assessor 1	Assessor 2	Total
Defensive programming				
Fault detection & diagnostic				
Error detecting code				
Failure assertion programming				
Diverse programming				
Memorising executed cases				
Software error effect analysis				
Fully defined interface				
Modelling				
Structured methodology				

J.7 Software code generation

This section discusses the usage of the approach for software code generation. It can be skipped depending the results of L.8.

Which criteria for software design and implementation are covered by the methodology (see EN50128 table A.4) :

	Author	Assessor 1	Assessor 2	Total
Formal methods				
Modeling				
Modular approach (mandatory)				
Components				
Design and coding standards (mandatory)				
Strongly typed programming language				

J.8 Main usage of the tool

This section discusses the main usage of the tool.

Which task are covered by the tool ?

	Author	Assessor 1	Assessor 2	Total
Modelling support				
Automatic translation				
Code Generation				
Model verification				
Test generation				
Simulation, execution, debugging				
Formal proof				

Modelling support

Does the tool provide a textual or a graphical editor ?

Automatic translation and code generation

Which translation or code generation is supported by the tool ?

Model verification

Which verification on models are provided by the tool?

Test generation

Does the tool allow to generate tests ? For which purpose ?

Simulation, execution, debugging

Does the tool allow to simulate or to debbug step by step a model or a code ?

Formal proof

Does the tool allow formal proof ? How ?

J.9 Use of the tool

According WP2 requirements, give a note for characteristics of the use of the tool (from 0 to 3) :

	Author	Assessor 1	Assessor 2	Total
Open Source (D2.6-X-36)				
Portability to operating systems (D2.6-X-37)				
Cooperation of tools (D2.6-X-38)				
Robustness (D2.6-X-41)				
Modularity (D2.6-X-41.1)				
Documentation management (D.2.6-X-41.2)				
Distributed software development (D.2.6-X-41.3)				
Simultaneous multi-users (D.2.6-X-41.4)				
Issue tracking (D.2.6-X-41.5)				
Differences between models (D.2.6-X-41.6)				
Version management (D.2.6-X-41.7)				
Concurrent version development (D.2.6-X-41.8)				
Model-based version control (D.2.6-X-41.9)				
Role traceability (D.2.6-X-41.10)				
Safety version traceability (D.2.6-X-41.11)				
Model traceability (D.2.6-01-035)				
Tool chain integration				
Scalability				

J.10 Certifiability

This section discusses how the tool can be classified according EN50128 requirements (D.2.6-X-50).

	Author	Assessor 1	Assessor 2	Total
Tool manual (D.2.6-01-42.02)				
Proof of correctness (D.2.6-01-42.03)				
Existing industrial usage				
Model verification				
Test generation				
Simulation, execution, debugging				
Formal proof				

Other elements for tool certification

J.11 Other comments

Please to give free comments on the approach.

Appendix K: UPPAAL

Author Author of the approaches description Stefan Rieger (TWT)

Assessor 1 First assessor of the approaches Cecile Braunstein (Uni. Bremen)

Assessor 2 Second assessor of the approaches Silvano DalZilio / LAAS

In the sequel, main text is under the responsibilities of the author.

Author. Author can add comments using this format at any place.

Assessor 1. First assessor can add comments using this format at any place.

Assessor 2. Second assessor can add comments using this format at any place.

When a note is required, please follow this list :

0 not recommended, not adapted, rejected

1 weakly recommended, adapted after major improvements, weakly rejected

2 recommended, adapted (with light improvements if necessary) weakly accepted

3 highly recommended, well adapted,strongly accepted

***** difficult to evaluate with a note (please add a comment under the table)

All the notes can be commented under each table.

K.1 Presentation

This section gives a quick presentation of the approach and the tool.

Name %%Name of the approach and the tool%%

Web site %%if available, how to find information%%

Licence %%Kind of licence%%

Abstract

Short abstract on the approach and tool (10 lines max)

Publications

Short list of publications on the approach (5 max)

K.2 Main usage of the approach

This section discusses the main usage of the approach.

According to the figure 1, for which phases do you recommend the approach (give a note from 0 to 3) :

	Author	Assessor 1	Assessor 2	Total
System Analysis				
Sub-system formal design				
Software design				
Software code generation				

According to the figure 1, for which type of activities do you recommend the approach (give a note from 0 to 3) :

	Author	Assessor 1	Assessor 2	Total
Documentation				
Modeling				
Design				
Code generation				
Verification				
Validation				
Safety analyses				

Known usages

Have you some examples of usage of this approach to compare with the OpenETCS objectives ?

K.3 Language

This section discusses the main element of the language.

Which are the main characteristics of the language :

	Author	Assessor 1	Assessor 2	Total
Informal language				
Semi-formal language				
Formal language				
Structured language				
Modular language				
Textual language				
Mathematical symbols or code				
Graphical language				

According WP2 requirements, give a note for the capabilities of the language (from 0 to 3) :

	Author	Assessor 1	Assessor 2	Total
Declarative formalization of properties (D.2.6-X-28)				
Simple formalization of properties (D.2.6-X-28.1)				
Scalability : capability to design large model				
Easily translatable to other languages (D.2.6-X-30)				
Executable directly (D.2.6-X-33)				
Executable after translation to a code (D.2.6-X-33) (precise if the translation is automatic)				
Simulation, animation (D.2.6-X-33)				
Easily understandable (D.2.6-X-27)				
Expertise level needed (0 High level, 3 few level)				
Standardization (D.2.6-X-29)				
Documented (D.2.6-X-29)				
Extensible language (D.2.6-01-28)				

Documentation

Describe how the language is documented, the existing guidelines, coding rules, standardization...

Language usage

Describe the possible restriction on the language

K.4 System Analysis

This section discusses the usage of the approach for system analysis. It can be skipped depending the results of L.8.

Acoording WP2 requirements, how the approach can be involved for the sub-system requirement specification ?

	Author	Assessor 1	Assessor 2	Total
Independent System functions definition (D.2.6-X-10.2.1)				
System architecture design (D.2.6-X-10.2)				
System data flow identification (D.2.6-X-10.2.3)				
Sub-system focus (D.2.6-X-10.2.4)				
System interfaces definition (D.2.6-X-10.2.5)				
System requirement allocation (D.2.6-X-10.3)				
Traceability with SRS (D.2.6-X-10.5)				
Traceability with Safety activities (D.2.6-X-11)				

K.5 Sub-System formal design

This section discusses the usage of the approach for sub-system formal design. It can be skipped depending the results of L.8.

Two kinds of model can be planned during this phase: semi-formal models to cover the SSRS (D.2.6-X-12.1) and strictly formal models to focuss on some functional and safety aspects (D.2.6-X-14). Obviously some strictly formal means can be used to define the semi-formal model.

K.5.1 Semi-formal model

Concerning semi-formal model, how the WP2 requirements are covered ?

	Author	Assessor 1	Assessor 2	Total
Consistency to SSRS (D.2.6-X-12.2)				
Coverage of SSRS (D.2.6-X-12.2.1)				
Coverage of SSHA (D.2.6-X-12.2.2)				
Management of requirement justification (D.2.6-X-12.2.3)				
Traceability to SSRS (D.2.6-X-12.2.5)				
Traceability of exported requirements (D.2.6-X-12.2.6)				
Simulation or animation (D.2.6-X-13 partial)				
Execution (D.2.6-X-13 partial)				
Extensible to strictly formal model (D.2.6-X-14.3)				
Easy to refine towards strictly formal model (D.2.6-X-14.4)				
Extensible and modular design (D.2.6-X-15)				
Extensible to software architecture and design (D.2.6-X-30)				

Concerning safety properties management, how the WP2 requirements are covered ?

	Author	Assessor 1	Assessor 2	Total
Safety function isolation (D.2.6-X-17)				
Safety properties formalisation (D.2.6-X-22)				
Logical expression (D.2.6-X-28.2.2)				
Timing constraints (D.2.6-X-28.2.3)				
Safety properties validation (D.2.6-X-23.2)				
Logical properties assertion (D.2.6-X-34)				
Check of assertions (D.2.6-X-34.1)				

Does the language allow to formalize (D.2.6-X-31):

	Author	Assessor 1	Assessor 2	Total
State machines				
Time-outs				
Truth tables				
Arithmetic				
Braking curves				
Logical statements				
Message and fields				

Additional comments on semi-formal model

Do you think your semi-formal model is sufficient to cover a safe design of the on-board unit until code generation ? All comments on links to other models, validation and verification activities are welcomed.

K.5.2 Strictly formal model

Concerning strictly formal model, how the WP2 requirements are covered ?

	Author	Assessor 1	Assessor 2	Total
Consistency to SFM (D.2.6-X-14.2)				
Coverage of SSRS (D.2.6-X-14.2)				
Traceability to SSRS (D.2.6-X-14.3)				
Extensible to software design (D.2.6-X-16)				
Safety function isolation (D.2.6-X-17)				
Safety properties formalisation (D.2.6-X-22)				
Logical expression (D.2.6-X-28.2.2)				
Timing constraints (D.2.6-X-28.2.3)				
Safety properties validation (D.2.6-X-23.3)				
Logical properties assertion (D.2.6-X-34)				
Proof of assertions (D.2.6-X-34.2)				

Does the language allow to formalize (D.2.6-X-32):

	Author	Assessor 1	Assessor 2	Total
State machines				
Time-outs				
Truth tables				
Arithmetic				
Braking curves				
Logical statements				
Message and fields				

Additional comments on semi-formal model

Do you think your strictly formal model can be directly defined from the SSRS ? All comments on links to other models, validation and verification activities are welcomed.

K.6 Software design

This section discusses the usage of the approach for software design. It can be skipped depending the results of L.8.

K.6.1 Functional design

How the approach allows to produce a functional software model of the on-board unit ?

	Author	Assessor 1	Assessor 2	Total
Derivation from system semi-formal model				
Software architecture description				
Software constraints				
Traceability				
Executable				

K.6.2 SSIL4 design

How the approach allows to produce in safety a software model ?

	Author	Assessor 1	Assessor 2	Total
Derivation from system semi-formal or strictly formal model				
Software architecture description				
Software constraints				
Traceability				
Executable				
Conformance to EN50128 § 7.2				
Conformance to EN50128 § 7.3				
Conformance to EN50128 § 7.4				

Which criteria for software architecture are covered by the methodology (see EN50128 table A.3) :

	Author	Assessor 1	Assessor 2	Total
Defensive programming				
Fault detection & diagnostic				
Error detecting code				
Failure assertion programming				
Diverse programming				
Memorising executed cases				
Software error effect analysis				
Fully defined interface				
Modelling				
Structured methodology				

K.7 Software code generation

This section discusses the usage of the approach for software code generation. It can be skipped depending the results of L.8.

Which criteria for software design and implementation are covered by the methodology (see EN50128 table A.4) :

	Author	Assessor 1	Assessor 2	Total
Formal methods				
Modeling				
Modular approach (mandatory)				
Components				
Design and coding standards (mandatory)				
Strongly typed programming language				

K.8 Main usage of the tool

This section discusses the main usage of the tool.

Which task are covered by the tool ?

	Author	Assessor 1	Assessor 2	Total
Modelling support				
Automatic translation				
Code Generation				
Model verification				
Test generation				
Simulation, execution, debugging				
Formal proof				

Modelling support

Does the tool provide a textual or a graphical editor ?

Automatic translation and code generation

Which translation or code generation is supported by the tool ?

Model verification

Which verification on models are provided by the tool?

Test generation

Does the tool allow to generate tests ? For which purpose ?

Simulation, execution, debugging

Does the tool allow to simulate or to debbug step by step a model or a code ?

Formal proof

Does the tool allow formal proof ? How ?

K.9 Use of the tool

According WP2 requirements, give a note for characteristics of the use of the tool (from 0 to 3) :

	Author	Assessor 1	Assessor 2	Total
Open Source (D2.6-X-36)				
Portability to operating systems (D2.6-X-37)				
Cooperation of tools (D2.6-X-38)				
Robustness (D2.6-X-41)				
Modularity (D2.6-X-41.1)				
Documentation management (D.2.6-X-41.2)				
Distributed software development (D.2.6-X-41.3)				
Simultaneous multi-users (D.2.6-X-41.4)				
Issue tracking (D.2.6-X-41.5)				
Differences between models (D.2.6-X-41.6)				
Version management (D.2.6-X-41.7)				
Concurrent version development (D.2.6-X-41.8)				
Model-based version control (D.2.6-X-41.9)				
Role traceability (D.2.6-X-41.10)				
Safety version traceability (D.2.6-X-41.11)				
Model traceability (D.2.6-01-035)				
Tool chain integration				
Scalability				

K.10 Certifiability

This section discusses how the tool can be classified according EN50128 requirements (D.2.6-X-50).

	Author	Assessor 1	Assessor 2	Total
Tool manual (D.2.6-01-42.02)				
Proof of correctness (D.2.6-01-42.03)				
Existing industrial usage				
Model verification				
Test generation				
Simulation, execution, debugging				
Formal proof				

Other elements for tool certification

K.11 Other comments

Please to give free comments on the approach.

Appendix L: GNATprove

Author Author of the approaches description David Mentré (MERCE)

Assessor 1 First assessor of the approaches %%Name - Company%%

Assessor 2 Second assessor of the approaches Matthias Gudemann (Systerel)

In the sequel, main text is under the responsibilities of the author.

Author. Author can add comments using this format at any place.

Assessor 1. First assessor can add comments using this format at any place.

Assessor 2. Second assessor can add comments using this format at any place.

When a note is required, please follow this list :

0 not recommended, not adapted, rejected

1 weakly recommended, adapted after major improvements, weakly rejected

2 recommended, adapted (with light improvements if necessary) weakly accepted

3 highly recommended, well adapted,strongly accepted

***** difficult to evaluate with a note (please add a comment under the table)

All the notes can be commented under each table.

L.1 Presentation

This section gives a quick presentation of the approach and the tool.

Name %%Name of the approach and the tool%%

Web site %%if available, how to find information%%

Licence %%Kind of licence%%

Abstract

Short abstract on the approach and tool (10 lines max)

Publications

Short list of publications on the approach (5 max)

L.2 Main usage of the approach

This section discusses the main usage of the approach.

According to the figure 1, for which phases do you recommend the approach (give a note from 0 to 3) :

	Author	Assessor 1	Assessor 2	Total
System Analysis				
Sub-system formal design				
Software design				
Software code generation				

According to the figure 1, for which type of activities do you recommend the approach (give a note from 0 to 3) :

	Author	Assessor 1	Assessor 2	Total
Documentation				
Modeling				
Design				
Code generation				
Verification				
Validation				
Safety analyses				

Known usages

Have you some examples of usage of this approach to compare with the OpenETCS objectives ?

L.3 Language

This section discusses the main element of the language.

Which are the main characteristics of the language :

	Author	Assessor 1	Assessor 2	Total
Informal language				
Semi-formal language				
Formal language				
Structured language				
Modular language				
Textual language				
Mathematical symbols or code				
Graphical language				

According WP2 requirements, give a note for the capabilities of the language (from 0 to 3) :

	Author	Assessor 1	Assessor 2	Total
Declarative formalization of properties (D.2.6-X-28)				
Simple formalization of properties (D.2.6-X-28.1)				
Scalability : capability to design large model				
Easily translatable to other languages (D.2.6-X-30)				
Executable directly (D.2.6-X-33)				
Executable after translation to a code (D.2.6-X-33) (precise if the translation is automatic)				
Simulation, animation (D.2.6-X-33)				
Easily understandable (D.2.6-X-27)				
Expertise level needed (0 High level, 3 few level)				
Standardization (D.2.6-X-29)				
Documented (D.2.6-X-29)				
Extensible language (D.2.6-01-28)				

Documentation

Describe how the language is documented, the existing guidelines, coding rules, standardization...

Language usage

Describe the possible restriction on the language

L.4 System Analysis

This section discusses the usage of the approach for system analysis. It can be skipped depending the results of L.8.

Acoording WP2 requirements, how the approach can be involved for the sub-system requirement specification ?

	Author	Assessor 1	Assessor 2	Total
Independent System functions definition (D.2.6-X-10.2.1)				
System architecture design (D.2.6-X-10.2)				
System data flow identification (D.2.6-X-10.2.3)				
Sub-system focus (D.2.6-X-10.2.4)				
System interfaces definition (D.2.6-X-10.2.5)				
System requirement allocation (D.2.6-X-10.3)				
Traceability with SRS (D.2.6-X-10.5)				
Traceability with Safety activities (D.2.6-X-11)				

L.5 Sub-System formal design

This section discusses the usage of the approach for sub-system formal design. It can be skipped depending the results of L.8.

Two kinds of model can be planned during this phase: semi-formal models to cover the SSRS (D.2.6-X-12.1) and strictly formal models to focuss on some functional and safety aspects (D.2.6-X-14). Obviously some strictly formal means can be used to define the semi-formal model.

L.5.1 Semi-formal model

Concerning semi-formal model, how the WP2 requirements are covered ?

	Author	Assessor 1	Assessor 2	Total
Consistency to SSRS (D.2.6-X-12.2)				
Coverage of SSRS (D.2.6-X-12.2.1)				
Coverage of SSHA (D.2.6-X-12.2.2)				
Management of requirement justification (D.2.6-X-12.2.3)				
Traceability to SSRS (D.2.6-X-12.2.5)				
Traceability of exported requirements (D.2.6-X-12.2.6)				
Simulation or animation (D.2.6-X-13 partial)				
Execution (D.2.6-X-13 partial)				
Extensible to strictly formal model (D.2.6-X-14.3)				
Easy to refine towards strictly formal model (D.2.6-X-14.4)				
Extensible and modular design (D.2.6-X-15)				
Extensible to software architecture and design (D.2.6-X-30)				

Concerning safety properties management, how the WP2 requirements are covered ?

	Author	Assessor 1	Assessor 2	Total
Safety function isolation (D.2.6-X-17)				
Safety properties formalisation (D.2.6-X-22)				
Logical expression (D.2.6-X-28.2.2)				
Timing constraints (D.2.6-X-28.2.3)				
Safety properties validation (D.2.6-X-23.2)				
Logical properties assertion (D.2.6-X-34)				
Check of assertions (D.2.6-X-34.1)				

Does the language allow to formalize (D.2.6-X-31):

	Author	Assessor 1	Assessor 2	Total
State machines				
Time-outs				
Truth tables				
Arithmetic				
Braking curves				
Logical statements				
Message and fields				

Additional comments on semi-formal model

Do you think your semi-formal model is sufficient to cover a safe design of the on-board unit until code generation ? All comments on links to other models, validation and verification activities are welcomed.

L.5.2 Strictly formal model

Concerning strictly formal model, how the WP2 requirements are covered ?

	Author	Assessor 1	Assessor 2	Total
Consistency to SFM (D.2.6-X-14.2)				
Coverage of SSRS (D.2.6-X-14.2)				
Traceability to SSRS (D.2.6-X-14.3)				
Extensible to software design (D.2.6-X-16)				
Safety function isolation (D.2.6-X-17)				
Safety properties formalisation (D.2.6-X-22)				
Logical expression (D.2.6-X-28.2.2)				
Timing constraints (D.2.6-X-28.2.3)				
Safety properties validation (D.2.6-X-23.3)				
Logical properties assertion (D.2.6-X-34)				
Proof of assertions (D.2.6-X-34.2)				

Does the language allow to formalize (D.2.6-X-32):

	Author	Assessor 1	Assessor 2	Total
State machines				
Time-outs				
Truth tables				
Arithmetic				
Braking curves				
Logical statements				
Message and fields				

Additional comments on semi-formal model

Do you think your strictly formal model can be directly defined from the SSRS ? All comments on links to other models, validation and verification activities are welcomed.

L.6 Software design

This section discusses the usage of the approach for software design. It can be skipped depending the results of L.8.

L.6.1 Functional design

How the approach allows to produce a functional software model of the on-board unit ?

	Author	Assessor 1	Assessor 2	Total
Derivation from system semi-formal model				
Software architecture description				
Software constraints				
Traceability				
Executable				

L.6.2 SSIL4 design

How the approach allows to produce in safety a software model ?

	Author	Assessor 1	Assessor 2	Total
Derivation from system semi-formal or strictly formal model				
Software architecture description				
Software constraints				
Traceability				
Executable				
Conformance to EN50128 § 7.2				
Conformance to EN50128 § 7.3				
Conformance to EN50128 § 7.4				

Which criteria for software architecture are covered by the methodology (see EN50128 table A.3) :

	Author	Assessor 1	Assessor 2	Total
Defensive programming				
Fault detection & diagnostic				
Error detecting code				
Failure assertion programming				
Diverse programming				
Memorising executed cases				
Software error effect analysis				
Fully defined interface				
Modelling				
Structured methodology				

L.7 Software code generation

This section discusses the usage of the approach for software code generation. It can be skipped depending the results of L.8.

Which criteria for software design and implementation are covered by the methodology (see EN50128 table A.4) :

	Author	Assessor 1	Assessor 2	Total
Formal methods				
Modeling				
Modular approach (mandatory)				
Components				
Design and coding standards (mandatory)				
Strongly typed programming language				

L.8 Main usage of the tool

This section discusses the main usage of the tool.

Which task are covered by the tool ?

	Author	Assessor 1	Assessor 2	Total
Modelling support				
Automatic translation				
Code Generation				
Model verification				
Test generation				
Simulation, execution, debugging				
Formal proof				

Modelling support

Does the tool provide a textual or a graphical editor ?

Automatic translation and code generation

Which translation or code generation is supported by the tool ?

Model verification

Which verification on models are provided by the tool?

Test generation

Does the tool allow to generate tests ? For which purpose ?

Simulation, execution, debugging

Does the tool allow to simulate or to debbug step by step a model or a code ?

Formal proof

Does the tool allow formal proof ? How ?

L.9 Use of the tool

According WP2 requirements, give a note for characteristics of the use of the tool (from 0 to 3) :

	Author	Assessor 1	Assessor 2	Total
Open Source (D2.6-X-36)				
Portability to operating systems (D2.6-X-37)				
Cooperation of tools (D2.6-X-38)				
Robustness (D2.6-X-41)				
Modularity (D2.6-X-41.1)				
Documentation management (D.2.6-X-41.2)				
Distributed software development (D.2.6-X-41.3)				
Simultaneous multi-users (D.2.6-X-41.4)				
Issue tracking (D.2.6-X-41.5)				
Differences between models (D.2.6-X-41.6)				
Version management (D.2.6-X-41.7)				
Concurrent version development (D.2.6-X-41.8)				
Model-based version control (D.2.6-X-41.9)				
Role traceability (D.2.6-X-41.10)				
Safety version traceability (D.2.6-X-41.11)				
Model traceability (D.2.6-01-035)				
Tool chain integration				
Scalability				

L.10 Certifiability

This section discusses how the tool can be classified according EN50128 requirements (D.2.6-X-50).

	Author	Assessor 1	Assessor 2	Total
Tool manual (D.2.6-01-42.02)				
Proof of correctness (D.2.6-01-42.03)				
Existing industrial usage				
Model verification				
Test generation				
Simulation, execution, debugging				
Formal proof				

Other elements for tool certification

L.11 Other comments

Please to give free comments on the approach.

