# openETCS Toolchain WP Description of Work

September 11, 2012

This Work Package will provide the tool chain that is necessary to formalise the ETCS system specification. The formal specification will be used further for verification and code generation. The tool chain must support the following tasks:

1. Support the authoring of the formal ETCS system specification.
2. Support the creation of a formal ETCS software specification (while the ETCS specification describes the *problem*, the software specification describes the *solution*).
3. Support code generation from the software specification.
4. Support execution, debugging and simulation of the software specification.
5. Support test case generation from a software test specification.
6. Support the verification and validation of the various artefacts, including the formal ETCS specification against the textual ETCS specification.
7. Comply with the EN 50128 requirements to tools.
8. Support requirements tracing across all tools and build steps.
9. Provide seamless integration of the tools into one tool chain.

The tool chain definition will benefit from other R&D projects and off-the-shelves tools. The semantics of the modelling languages shall be carefully studied.

The first goal of this WP is to identify sets of consistent languages and tools enabling the design of the system. This will be done in close collaboration with WP2.

In order to be able to progress without depending too much on WP2 requirements and other deliverables, the subtasks shall make use of prototyping in order to gain knowledge regarding the possible modelling languages and tool platforms.

# 1 Tool Chain Architecture Analyses and Recommendations

The first subtask is the tool chain architecture analyses and recommendations. This tool chain will encompass all description levels of the system design from holistic viewpoint to code generation. Specific attention shall be paid to the semantics and the traceability of each part of the chain. This task is composed of the following activities:

> (jonasHelming) For both, 1.1 as well as 1.2 it would be helpful to have a reference set of ERTMS requirements from WP2. This should be small sub-set of the specification, but be somehow representative. Using this sub-set would ake it much easier to compare different languages and tools. It can be used as a test set. I think this might be even more useful then detailed requirements for the toolchain, as we have quite some compentence on that in WP3a I think. Of course we will work on the requirements from WP2, but to get startedt with a comparison of existing tools, I think a test set might be the most required piece. Any opinions on that?

## 1.1 Identify/define the potential modelling languages

(mj) I think there is a huge overlap with WP2. Their findings may reduce the possible target languages drastically, and by their requirement must take everything written here into account anyway.

(SP) I agree with michael, so I removed the language analysis, and replaced it by prototyping using the candidate languages selected by WP2.

WP2 shall identify the candidate languages suitable to model the Subset-026.

The following prototypes shall be developed, one for each candidate language outlined by WP2 State of the Art Analysis, and this for a representative subset of Subset-026 requirements.

1. ... (insert here one prototype per modelling language) ...

| | | |
|---|---|---|
| Input: | WP2 State of the Art Analysis | Oct-12 |
| | WP2 Requirements (incomplete) | Oct-12 |
| | WP2 Reference set of ERTMS requirements | Oct-12 |
| Output: | Modeling Language(s) (and reasoning) | ??? |

## 1.2 Identify and compare existing tools

> (jonasHelming) I think we had this task in the FPP before, was it removed? I think that is already a task we could get started with. I think there was already a proposal on the mailing list to create wiki pages for the tools suggested by the partners. It would make sense, tough to prepare a template for this.

## 1.3 Decide which existing tool platform shall be used as the foundation

Based on the state of the art findings and requirements from WP2, an existing tool platform will be chosen for implementing the tool chain.

The following prototypes shall be developed:

1. ... (insert here useful prototypes) ...

| | | |
|---|---|---|
| Input: | WP2 State of the Art Analysis | Oct-12 |
| | WP2 Requirements (incomplete) | Oct-12 |
| | WP2 Reference set of ERTMS requirements | Oct-12 |
| Output: | Tool Platform (and reasoning) | ??? |

## 1.4 Identify the modelling tools architecture

(mj) I assume that the previous task will pretty much dictate the platform architecture.

These tools enable designers to model using the languages defined before. The interoperability of the tools is a key point to address traceability of the different models in the design process. Tools may already exist and may be used "off-the-shelves" or may be developed in the WP. As example, if a standard modelling language is selected such as UML/SysML, open tools already exists (Topcased); if a specific modelling language is needed, WP actors have to develop a grammar or meta model to develop such particular modelling tool.

## 1.5 Analyse requirement elicitation techniques

Analyse requirement elicitation techniques in order to define a strategy (mj: process?) to derive OpenETCS formal model requirements from ERTMS FRS/SRS

Addition Christophe.GASTON@cea.fr:

> A classical modelling process starts by defining high level models of systems that shall be refined step by steps in order to make implementation choices. From a syntactical point of view, the model transformation techniques described below are good technoligical candidates. However, at the sematical level, refining a model into a more concrete one requires first to define a refinement correctness relation in order to ensure properties preservation. Moreover, systems that will be considered in the project will be based on concepts of interacting processes (since those systems are distributed). Therefore we need to identify techniques allowing us to start from system level properties (*i.e.* specifying global behaviors) and to deduce what properties each

> process should satisfy (*i.e.* what behavior should have each process) so that the global cooperation of all processes realize the system properties. In this subtask we will identify potential solutions to that problem.

|   |   |   |
|---|---|---|
| Input: | WP2 Requirements (incomplete) | Oct-12 |
| Output: | Requirement Elicitation Techniques | ??? |

(SP) I would suggest to remove that subtask. - This belongs to WP2. - Requirement elicitation technique is not a tool, and moreover - ERTMS SRS (Subset-026) already include the requirements.

## 1.6 Verification Tools

Identify potential verification tools with regard to modelling techniques; verification techniques shall be investigated.

CEA: we propose to agregate in this section several subsection related to the kinds of properties which is treated. We tried to identify subsections following this intuition. The idea would be to talk about technologies (*e.g.* static analysis, model-checking etc...) that could adress the verification of each sort of property.

(mj) I think this is not on the cricital path.

Comment Hardi Hungar:

> Verification tools resp. techniques are very important in the development of a safety-critical system like the ETCS OBU. According to the relevant standards (most prominently the EN 50128 of the CENELEC family), every design step has to be verified. Assuming that models will constitute artifacts oft he development – and are not just used for explanatory purposes – it is necessary to be able to establish the correctness of each refinement step. Or, to put it differently, the tool chain needs a concept for seamless verification, preferably tool-based, to be fit for its purpose.
>
> I think, this must be taken into account already early in the definition process. Therefore, while it might not be the first thing to consider (without modeling, there is no verification of models), it should definitely not be the last.

Comment Stanislas Pinte:

> I agree with Hardi.
>
> In my opinion, the model should include the tests of the model, so that it could be verifiable in a "model-in-the-loop" fashion.
>
> It doesn't have to be model proving (that I think belons more to other WPs) rather model testing.
>
> Inside our http://www.ertmssolutions.com/ertms-formalspecs/ approach, we assume the following:
>
> - Model tests are part of the model - Model should be 100% covered by tests (proved by model coverage reports) - Toolchain must support developing, executing and debugging tests

> Model verification also includes:
>
> - veryfing that the model corresponds to the orginial requirements specifications (in our case, UNISIG Subset-026 BL3). ERTMSFormalSpecs also supports marking model artifacts as "verified" against source requirements. - veryfing that 100% of source requirements are traced against one or more model artifacts (proven by traceability reports)
>
> I would think that such verifications are indeed in the critical path...i.e. if not implemented we shall not be able to have a fully functional model.

### 1.6.1 Functionnal properties

Such as all nominal behaviors of functions of a system...

### 1.6.2 Non Functionnal properties

Such as real time properties concerning executions, data dependancies...

### 1.6.3 Safety properties

Such as fault tolerance, absence of run time errors...

| | | |
|---|---|---|
| Input: | WP2 Requirements | ??? |
| Output: | Verification tool choice | ??? |

## 1.7 Code Generation Strategy

(mj) I think this is not on the cricital path.

Analyse the code generation strategy.
Comment Stanislas Pinte:

> In my opinion, the code generation strategy should be handled in T3.3 Modelling of ETCS specification, that is part of WP3b.
>
> @Fabien, could you confirm WP3b point of view about this?
>
> Additionally, prototypes should be developed for each candidate modelling language, and for each candidate target language.

| | | |
|---|---|---|
| Input: | WP2 Requirements | ??? |
| Output: | Code Generation Strategy | ??? |

## 1.8 Model Transformation

(mj) I think this is not on the cricital path.

Analyse model transformation techniques and tools in order to refine the specification from one description level to another.

Comment Stanislas Pinte:

> In our product ERTMSFormalSpecs (http://www.ertmssolutions.com/ertms-formalspecs/) we use a single, unified model, that is supporting complete Subset-026 logic, with full traceability.
>
> If that approach works, why do we need several specification levels?

| | | |
|---|---|---|
| Input: | WP2 Requirements | ??? |
| Output: | Model Transformation Strategy | ??? |

### 1.9 Schedulability

CEA: this section could disappear and the content would be dispatched in subsection "'non functionnal properties" of Section "'Verification"

(mj) I think this is not on the cricital path.

Analyse schedulability tools.

| | | |
|---|---|---|
| Input: | WP2 Requirements | ??? |
| Output: | Schedulability Strategy | ??? |

### 1.10 Capture Additional Requirements

Capture wishes/requirements on how to support the designer in their activities.

| | | |
|---|---|---|
| Input: | Designer Wishes and Requirements | ongoing |
| Output: | Captured and organised designer requiremenets | ??? |

## 2 Define and Develop Tool Chain

(mj) I see "make" not as an option. Considering the resources available for this project, we will tailor (and extend) an exisiting tool platform.

The second subtask defines and develops the tool chain and the infrastructure enabling its evolution and maintenance. First of all, a "make or reuse" decision about the components of the tool chain has to be made. Then a common development infrastructure has to be defined or chosen in order to integrate all the tools (Eclipse like infrastructure). Finally, the subtask achieves the development and the integration of the tools.

TODO: Further subtasks to be developed.