OETCS/WP7/D7.2 – 00/01

openETCS

ITEA2 Project
Call 6 11025
2012 – 2015

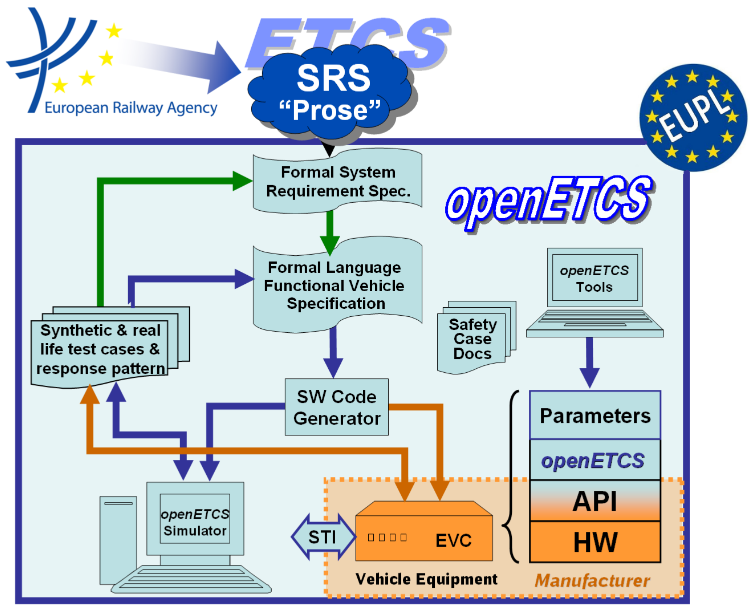
Work-Package 7: “Secondary tools”

Report on all aspects of secondary tooling

Results of T7.2

Marielle Petit-Doche, all participants of the benchmark and all participants of VnV and Safety process

March 2014



Funded by:

Federal Ministry of Education and Research

Région de Bruxelles-Capitale

GOBIERNO DE ESPAÑA
MINISTERIO DE INDUSTRIA, ENERGÍA Y TURISMO

This page is intentionally left blank

Work-Package 7: “Secondary tools” **OETCS/WP7/D7.2 – 00/01**
March 2014

Report on all aspects of secondary tooling
Results of T7.2

Document approbation

Lead author:	Technical assessor:	Quality assessor:	Project lead:
location / date	location / date	location / date	location / date
signature Marielle Petit-Doche (Systerel)	signature Cécile Braunstein (University of Bremen)	signature Cécile Braunstein (University of Bremen)	signature Klaus-Rüdiger Hase (DB Netz)

Marielle Petit-Doche
Systerel

all participants of the benchmark
WP7 partners

all participants of VnV and Safety process
WP4 partners

Deliverable

Prepared for openETCS@ITEA2 Project

Abstract: This document gives results of the evaluation and selection of the tools and methods to complete the secondary toolchain and to support verification and validation activities, safety activities, model transformation and data management for the whole project. As outlined in detail in the introduction, the choice of secondary tools is not final, as this would significantly inhibit the agility of this project. Rather, it is a knowledge base of available technologies, allowing the educated choice of suitable technologies on an "as needed" basis, during the lifetime of the project.

Modification History

Version	Section	Modification / Description	Author
00.01		19/12/2013 Document creation	M. Petit-Doche
00.02		19/12/2013 Approach (issue #236), review comments, tool descriptions	M. Petit-Doche
00.03		08/01/2014 Completing section on ProR	M. Jastram
00.04		09/01/2014 RT-tester description added	C. Braunstein
00.05		13/01/2014 Added description in the CPN Tools section; Added section on CPN Tools (copy from V and V); Added the section for ACedit and GSN (only copy from management); Completed Acedit section to give basic tool information; Added structure for section ACedit and ReqCycle	J. Welte
00.06		29/01/2014 Frama-C summary	V. Prevosto
00.07		21/02/2014 Additions to SystemC and Uppaal sections	S. Rieger
00.08		28/02/2014 Adding to Acceleo Section	A. Nitsch
01.00		03/03/2014 Finalization of the Deliverable	M. Jastram
01.01		26/11/2014 Updating of the template for compliance	M. Jastram

Table of Contents

Figures and Tables

Figures

Tables

Document information	
Work Package	WP7
Deliverable ID or doc. ref.	D7.2
Document title	Report on all aspects of secondary tooling
Document version	00.02
Document authors (org.)	Marielle Petit-Doche (Systerel)

Review information	
Last version reviewed	01.01
Main reviewers	See revision history.

Approbation			
	Name	Role	Date
Written by	Marielle Petit-Doche	WP7-T7.2 Sub-Task Leader	
Approved by	Michael Jastram	WP7 leader	3-Mar-2014

1 Introduction

The aim of this document is to report the results of the evaluation of means and tools for the secondary toolchain, i.e. the means and tools which complete the primary tool chain dedicated to formal model and software design.

This evaluation task is part of work package WP7, task 2 "Secondary tools analyses and recommendations". According to the results of WP2, especially the OpenETCS process and the requirements on language and tools [3], and the results of T7.1 on the primary toolchain [5], the aim of this task is to determine the best candidates to complete and support the primary toolchain for the following activities:

- data, function and requirement management (SSRS, WP3 and WP4), in chapter 2;
- verification and validation (WP4), in chapter 3;
- safety activities support (WP4), in chapter 4;
- model transformation and code generation (WP3 and WP4), in chapter 5.

1.1 Approach

Initially, it was planned to perform the secondary toolchain analysis in the same way the primary toolchain analysis was performed. However, it turned out not to be practical. Most importantly, team members were considering the secondary tools tactical (while the primary tools were strategic). Therefore, it was less clear what exactly was needed, and the impact of tool choice would be much more limited. Take model transformation for instance: The framework technology (EMF) was already given. It was clear that transformation would have to take place, but not yet which ones. Thus, it was unclear, which EMF model transformation framework would be the best one. Further, the choice would have little impact on the overall system (as the input and output of the model transformation would always be EMF models). Therefore, there was little motivation for an exhaustive evaluation.

As a result, this document contains the evaluated technologies, but only broadly describes those that have been evaluated or presented.

2 Data and Requirements Management

This section is dedicated to tools and means to support management of data, functions requirements and other artifacts along the openETCS process.

In total, seven tools have been proposed. Out of these, only one has been evaluated in detail (ProR). What follows is a qualitative description of the seven tools. A quantitative evaluation of ProR is included as well.

As a consequence, ProR has been adapted as the tool for requirements management. Being an open source Eclipse Project, based on EMF and an open standard for requirements exchange (ReqIF), it is a good fit.

Further, traceability has been identified as a crucial element of data and requirements management. Currently, the strongest candidate is Polarsys ReqCycle, which fulfills many required criteria. The biggest concern with respect to ReqCycle is its relatively low level of maturity.

2.1 Candidates

The list of initial candidates is:

Scade Suite. Scade includes Reqtify as the requirements traceability solution. It allows to create traceability directly to Word, thereby making traceability to Subset-26 easy. However, there is no clear solution for authoring additional requirements (except using Word). Further, it is not clear how traceability to model artifacts should be realized. Last, this is a closed source solution and therefore only a last resort.

Eclipse ProR. See 2.1.2. ProR is an Eclipse based open-source tools for requirement engineering that supports the ReqIF 1.0.1 Standard .

Rodin and Plugin. See 2.1.3. Rodin is an open source tool based on Eclipse dedicaded to formal design. It includes ProR as plugin to manage direct traceability between requirements and formal models.

Assurance Case Argument Editor (ACedit) using Goal Structuring Notation (GSN). See 2.1.4. The Editor for Assurance Cases is an Eclipse based tool to use the Goal Structuring Notation (GSN) and the OMG Argumentation Metamodel (ARM) to build argument structure models and check their overall completeness and consistency. Such model help to develop, manage and document the reasons and fulfillment for every requirement.

Eclipse EMF Store. Slides at `https://github.com/openETCS/model-evaluation/blob/master/Telco_Secondary_slides/EMFStore.pdf`.

Eclipse EMF Client Platform. Slides at `https://github.com/openETCS/model-evaluation/blob/master/Telco_Secondary_slides/EMF%20Client%20Platform.pdf`

Matelo. Dropped due to lack of further engagement.

2.1.1 Tools Identified During Evaluation

During the evaluation phase, a number of additional tools were identified that were not clearly defined before. The list of challengers discussed during the evaluation is:

Ecore model + XML files. The (already chosen) Eclipse Modeling Framework supports the definition of datamodels and inter-model-traceability.

UML library. The creation of UML libraries is an attractive approach for integrating various data models on SysML-Level. This has been already realized in a prototype for the Data Dictionary.

ReqCycle ReqCycle can create requirements, import/update or reference textual requirements existing in different data sources or formats. Thereby, it can define data models to classify requirements in scopes (team organization) and in types (list of attributes). Furthermore, ReqCycle can define traceability link types from requirements to a model element or code and display all traceability links as treeviews or tables. Slides are available on github `https://github.com/openETCS/model-evaluation/blob/master/Telco_Secondary_slides/WP7%20-%20ReqCycle%20overview.pdf`

2.1.2 ProR Evaluation

Name Eclipse ProR (part of the Eclipse Requirements Modeling Framework (RMF))

Web site http://eclipse.org/rmf/pror

Licence Eclipse Public License

Abstract

ProR is a tool for requirements engineering that supports the ReqIF 1.0.1 Standard natively. It is part of the Eclipse RMF project and is built for extensibility. ProR is the result of the European Union FP7 Research Project Deploy, where it is used to provide traceability between requirements and formal models.

Vision: Our vision is to provide a tool that supports interoperability by operating directly of the ReqIF data standard, and that supports integration by taking advantage of the extendability of the Eclipse platform. ProR is targeted to academic and industrial users alike.

Publications

A number of publications about RMF and ProR are available. The following is a selection.

Wikipedia: Requirements Modeling Framework. The Requirements Modeling Framework (RMF) is an Open-Source-Framework for working with requirements based on the ReqIF standard. RMF consists of a core allowing reading, writing and manipulating ReqIF data, and a user interface allowing to inspect and edit request data.

RMF is the first and, currently, the only open source reference implementation of the ReqIF standards. Noteworthy is the fact that RMF has already been deployed in the ProStep ReqIF

Implementor Forum in order to ensure the interoperability of the commercial implementation. Since 2011 there have been reports in the German and in the international press about RMF. (more)

The Eclipse Requirements Modeling Framework. In Managing Requirements Knowledge, Springer, 2013. Michael Jastram.

This chapter presents the the Requirements Modeling Framework (RMF), an Eclipse-based open source platform for requirements engineering. The core of RMF is based on the emerging Requirements Interchange Format (ReqIF), which is an OMG standard. The project uses ReqIF as the central data model. At the time of this writing, RMF was the only open source implementation of the ReqIF data model.

By being based on an open standard that is currently gaining industry support, RMF can act as an interface to existing requirements management tools. Further, by based on the Eclipse platform, integration with existing Eclipse-based offerings is possible.

In this chapter, we will describe the architecture of the RMF project, as well as the underlying ReqIF standard. Further, we give an overview of the GUI, which is called ProR. A key strength of RMF and ProR is the extensibility, and we present the integration ProR with Rodin, which allows traceability between natural language requirements and Event-B formal models.

Openness in Systems Engineering with Eclipse ProStep Symposium, 2013. Michael Jastram.

Eclipse is an open source framework for building platform-independent GUI applications. It is managed by the Eclipse Foundation (a non-profit organization), which ensures that official Eclipse projects are interoperable and follow certain intellectual property guidelines. Open Source in general allows organizations to remedy the risk of being dependent on one single vendor. This includes the risk of the feature set provided: users can add missing features themselves or commission their inclusion to any competent party, rather than having to rely on the vendor to implement it. It further includes the risk of maintenance and long-term support. Eclipse in particular provides a solid, mature and open platform for desktop applications with a rich ecosystem. Many Eclipse offerings are ready to be used “as is”, thereby offering great cost savings.

In this talk, we demonstrate how Eclipse can be used as an integration platform for systems engineering. We focus on RMF (Requirements Modeling Framework) as a case study on how the Eclipse ecosystem can be leveraged in a business environment. RMF is a clean-room implementation of the open ReqIF standard, which is currently being adopted by various tool vendors: The currently ongoing ReqIF Implementor Forum , which is organized by ProSTEP iViP, will ensure that the various ReqIF implementations will properly function together. We will look at both the technical and business implications.

From a business point of view, this approach promises cost savings and prevents vendor lock-in. To understand the value, we will look at the openETCS project , which is an ITEA2 EU-funded project. The purpose of this project is the development of an integrated modeling, development, validation and testing framework for leveraging the cost-efficient and reliable implementation of the European Train Control System (ETCS), based on open source technologies. While the technology choice has not yet been fi¬nalized, Eclipse is a strong candidate for realizing this project, and it being open source is a core requirement. We will present the implications of such an open platform from a business point of view for the parties involved, which are customers (e.g. Deutsche Bahn), equipment manufacturers (e.g. Siemens) service providers (e.g. Formal Mind) and, of course, the EU and its citizens.

ReqIF-OLUTION: Mit Eclipse und ReqIF zur Open-Source ALM-Werkzeugkette ObjektSpektrum, 3, 2013. Michael Jastram.

Exchange of requirements in the past either results in data loss or requires a proprietary solution. But with ReqIF, there is no an international OMG standard that solves this problem. This triggered an avalanche of activities, including the development of an Open Source reference implementation (Eclipse RMF). This article demonstrates, how with ReqIF, Eclipse and RMF, an ALM can be created with litte effort.

The ProR Approach: Traceability of Requirements and System Descriptions. CreateSpace, 2012. Michael Jastram.

Creating a system description of high quality is still a challenging problem in the field of requirements engineering. Creating a formal system description addresses some issues. However, the relationship of the formal model to the user requirements is rarely clear, or documented satisfactorily.

This work presents the ProR approach, an approach for the creation of a consistent system description from an initial set of requirements. The resulting system description is a mixture of formal and informal artefacts. Formal and informal reasoning is employed to aid in the process. To achieve this, the artefacts must be connected by traces to support formal and informal reasoning, so that conclusions about the system description can be drawn.

The ProR approach enables the incremental creation of the system description, alternating between modelling (both formal and informal) and validation. During this process, the necessary traceability for reasoning about the system description is established. The formal model employs refinement for further structuring of large and complex system descriptions. The development of the ProR approach is the first contribution of this work.

This work also presents ProR, a tool platform for requirements engineering, that supports the ProR approach. ProR has been integrated with Rodin, a tool for Event-B modelling, to provide a number of features that allow the ProR approach to scale.

The core features of ProR are independent from the ProR approach. The data model of ProR builds on the international ReqIF standard, which provides interoperability with industrial tools for requirements engineering. The development of ProR created enough interest to justify the creation of the Requirements Modeling Framework (RMF), a new Eclipse Foundation project, which is the open source host for ProR. RMF attracted an active community, and ProR development continues. The development of ProR is the second contribution of this work.

This work is accompanied by a case study of a traffic light system, which demonstrates the application of both the ProR approach and ProR.

A Systems Engineering Tool Chain Based on Eclipse and Rodin In Forms/Format, 2012. Michael Jastram.

Formal methods are experiencing a renaissance, especially in the development of safety-critical systems. An indicator for this is the fact that more and more standards either recommend or prescribe the use of formal methods.

Using formal methods on an industrial scale requires their integration into the system engineering process. This paper is exploring how an integrated tool chain that supports formal methods may look like. It thereby focusses on our experience with tool chains that are based on the open source Eclipse platform in general, and the Rodin formal modeling environment in particular.

Open Source allows organisations to remedy the risk of being dependent on one single vendor. This includes the risk of the feature set provided: users can add missing features themselves or commission their inclusion to any competent party, rather than having to rely on the vendor to implement it. It further includes the risk of maintenance and long-term support.

We see industrial interest in open source for systems engineering in general, and Eclipse in particular. Eclipse is attractive, because its license is business-friendly. Further, its modular architecture makes it easy to seamlessly integrate the various Eclipse-based tools for systems engineering.

This paper focuses on an ecosystem that is accumulated around two Eclipse-based platforms, First, the Rodin platform is an open source modeling environment for the Event-B formalism. Second, the Requirements Modeling Framework (RMF) is a platform for working with natural language requirements, supporting the international ReqIF standard.

Requirements Modeling Framework Eclipse Magazin, 6.11, 2011. Michael Jastram, Andreas Graf.

In August 2011, the Requirements Modeling Framework (RMF) has been instantiated as a new Eclipse Project. RMF consists of a core that processes data in the Requirements Interchange Format (RIF/ReqIF), and a user interface called ProR, that allows the comfortable editing of requirements data. proR provides an Extension Point that allows the integration with other tools.

Requirement Traceability in Topcased with the Requirements Interchange Format (RIF/ReqIF) First Topcased Days Toulouse, 2011. Michael Jastram, Andreas Graf.

One important step of the systems engineering process is requirements engineering. Parallel to the development of Topcased, which includes tooling for requirements engineering, a new standard for requirements exchange is emerging at the OMG under the name “ReqIF” (formally called RIF). In our talk we introduce the activities of two research projects and their tool developments, VERDE (Yakindu Requirements) and Deploy (ProR) and discuss possible synergies with Topcased.

Requirements, Traceability and DSLs in Eclipse with the Requirements Interchange Format (RIF/ReqIF) Technical Report, Dagstuhl-Workshop MBEES 2011: Modellbasierte Entwicklung eingebetteter Systeme, 2011.

Requirements engineering (RE) is a crucial aspect in systems development and is the area of ongoing research and process improvement. However, unlike in modelling, there has been no established standard that activities could converge on. In recent years, the emerging Requirements Interchange Format (RIF/ReqIF) gained more and more visibility in industry, and research projects start to investigate these standards. To avoid redundant efforts in implementing the standard, the VERDE and Deploy projects cooperate to provide a stable common basis for RIF/ReqIF that could be leveraged by other research projects too. In this paper, we present an Eclipse-based extensible implementation of a RIF/ReqIF-based requirements editing platform. In addition, we are concerned with two related aspects of RE that take advantage of the common platform. First, how can the quality of requirements be improved by replacing or complementing natural language requirements with formal approaches such as domain specific languages or models. Second, how can we establish robust traceability that links requirements and model constructs and other artefacts of the development process. We present two approaches to traceability and two approaches to modelling. We believe that our research represents a significant contribution to the existing tooling landscape, as it is the first clean-room implementation of the RIF/ReqIF standard. We believe that it will help reduce gaps in often heterogeneous tool chains and inspire new conceptual work and new tools.

Added value for OpenETCS project

ProR fulfills the formal requirements to be part of the openETCS toolchain: It is based on Eclipse and EMF, and it is licensed under an appropriate open source license. It is mature enough to be used for managing natural language requirements.

Integration in OpenETCS process and toolchain

As of this writing, ProR is already integrated into the openETCS tool. As it is based on EMF, data integration should be reasonably straight forward.

2.1.3 EventB, Rodin and plugins

Name Event-B and the Rodin platform

Web site `http://www.event-b.org`

Licence Common Public License Version 1.0 (CPL)

Abstract

Rodin is an open source tool for formal modeling and verification on the system level using the Event-B formalism. Event-B is based on set-theoretic notation of first-order logic (FOL) and has its roots in the B method which has a long history of successful application in industry on software level development.

Rodin is fully integrated into the Eclipse platform and is therefore fully extensible through plug-ins. Existing plug-ins include graphical modeling using state-machines, model simulators, modern state-of-the art SMT solvers and Rational DOORS interoperable requirements tracing using ReqIf documents and ProR.

Publications

- The leaflet [6] contains a short overview of the Rodin tool
- The book [4] explains the usage of Rodin and serves as a gentle introduction into Event-B modeling in Rodin
- The book [1] contains an extensive presentation of Event-B an several modeling examples for different system
- The scientific journal article [2] contains an in-depth look at the integration of Event-B into the Rodin platform

Slides are available on github: `https://github.com/openETCS/model-evaluation/blob/master/Telco_Secondary_slides/Systerel_Event-B.pdf`.

A quantitative evaluation is available in `https://github.com/openETCS/toolchain/blob/master/T7.2/O7.2.1_Safety/O7-2-1_Safety.pdf`

Added value for OpenETCS project

Rodin is a specialized tool to formally model and verify abstract functional behavior. Therefore data management is not in its scope, as this is clearly a lower level detail aspect, more on the implementation level.

Function Management: A Rodin model contains high level function descriptions, i.e., an abstract view of the observable system behavior and its effect on the system state. It is therefore well suited to be included in function management, by formalizing the abstract behavior of the functions, tracing any changes and observing their effect on the intended functioning of the system.

Version Management: Rodin does not contain a version management itself. Its files are based on XML, therefore any modern version control system can be used, in particular those (like svn/mercurial/git) for which an Eclipse plug-in exists. There also exists a pug-in that is compatible to model-compare in Eclipse, i.e., allows for comparison on the model level instead of text level.

Other: Rodin can provide an important support for **traceability**, which is missing here. It allows for linking formal model aspects to a requirements document, e.g., a ReqIf document in ProR. Any changes in the specification can therefore be traced in the formal Event-B model and system-level aspects can be formally verified.

Integration in OpenETCS process and toolchain

The Rodin platform is fully based on Eclipse.

The existing graphical modeling plug-ins for Rodin could be connected to Papyrus. This would require the development of a transformation of the different formats.

With SCADE there could be the possibility of interoperation via the SCADE System SysML framework.

With Classical B tools, there is the possibility to generate predicates for guards and invariants directly from the Event-B model. As classical B is based on text files and Event-B on XML file, there would be some development work to do.

2.1.4 Assurance Case Argument Editor

Name Assurance Case Argument Editor (ACedit) as an editor for the Goal Structuring Notation (GSN)

Web site https://code.google.com/p/acedit/

Licence Eclipse Public License 1.0

Abstract

A safety case is a collection of documents, which shall demonstrate that the system fulfills conditions to guaranty a safe behavior. Therefore, an argumentation chain has to be constructed to demonstrate that from the evidence available it can be reasonably concluded that a system is acceptably safe. The Goal Structuring Notation (GSN) is a known approach for constructing such argumentation chains, which refer to the explicit documentation of claims about a system, and explanation of how these claims are convincingly supported by evidence generated during different activities in design, verification and validation of that system. The ACedit is a tool implementation of this approach realized as eclipse plug-ins under the Eclipse project. The tool supports the graphical construction of argument chains in GSN and a number of model management techniques that apply to them.

The key features are the following

1. Graphical editor for the GSN, based on the GSN standard
2. Graphical editor for the ARM, based on the ARM metamodel available by sysa.omg.org
3. Model management tasks for the GSN editor like Model validation and In-place model transformation
4. Model transformation between GSN and ARM models

The graphical editors of ACedit are based on the definition of GSN and ARM metamodels, which are defined in the EMFatic language.

Publications

- George Despotou, Aris Apostolakis, Dimitris Kolovos: Assuring Dependable and Critical Systems: Implementing the Standards for Assurance Cases with ACedit
- Tim Kelly and Rob Weaver: The Goal Structuring Notation – A Safety Argument Notation

Slides are available on github in `https://github.com/openETCS/model-evaluation/blob/master/Telco_Secondary_slides/psn_20130311_Tool-Presentation-GSN_1-0_jw.pdf`

A quantitative evaluation is available in `https://github.com/openETCS/toolchain/blob/master/T7.2/O7.2.1_Safety/O7-2-1_Safety.pdf`

Added value for OpenETCS project

To structure the safety case with GSN allows us to present a clear outline of the argumentation chain and the specific relations between the numerous documents. In this way a consistent generic safety case for the overall development of an on-board system based on the openETCS development process can be presented as a GSN model. All documents already created in the openETCS work can be linked to the GSN argumentation structure to present an easy overview about the safety case and the status the documents and their relation. This creates a common understanding of the openETCS safety argumentation and helps to show the specific context for their work to all different working groups.

Integration in OpenETCS process and toolchain

The safety case task will establish based on the ongoing work of design and verification and validation activities the underlying argumentation chain, which shall demonstrate that the openETCS development process fulfills required quality and safety aspects. This will first be done by modeling the theoretical argumentation chain given by the development methodology. As the work goes on the general steps in the argumentation chain are enhanced through the actual artifacts which are created during the design and corresponding verification steps. the same will be done with all validations results for the design artifacts. As the ACedit editor is already based on Eclipse the existing to can be integrated in the basic tool chain environment to link the argumentation chain elements to specific design elements. All purely document based argumentation and conclusion parts will be linked to the GSN model via their presentation in the git repository. Therefore, a connection between the ACedit tool and the GIT repository has to be established. To do this the already existing options in the eclipse framework shall be used.

2.2 Open issues

During the evaluation, a number of issues arose that had not been clearly identified by the WP2 requirements. We briefly outline the issues and the course of action.

2.2.1 Traceability

Traceability between the various models is crucial for many activities, including V&V and change management. Many tools already support traceabillity to a degree, but only within their models (e.g. traceability between SysML model elements). However, it is clear that the model elements from the various distinct tools need to be traced as well, e.g. between requirements, SysML elements, tests, code, etc.

A number of technologies have been idenitfied and are currently discussed in toolchain issue #216 (`https://github.com/openETCS/toolchain/issues/216`).

2.2.2 How to Deal with Subset-26

Subset-26 is available as a set of Microsoft Word documents, which are unsuitable for using directly, for various reasons. The team is currently looking at ways for establishing traceablity. The most promising approach is to convert the documents into the public ReqIF-format, which would allow reading them with the ProR component that has been chosen for requirements management.

2.3 Selected means and tools

As described in the beginning, the secondary tools have tactical, not strategic value. Further, areas in secondary tooling have been identified that are not yet required by other workpackages, e.g. code generation. Thus, selection falls into three categories:

(1) Definite secondary tool choices for areas where a solution is required right away, and where a suitable tool has been identified. This includes ProR for requirements engineering.

(2) Tentative secondary tool choices for areas where a solution is required right away, but where it is not clear whether the tool choice is suitable. This includes the use of a UML profile for making the data dictionary accessible in SysML.

(3) Tool candidates that are strong candidates, but that are not in use yet. This includes tools like ReqCycle for traceability.

In addition, there is the library of evaluated tools that will be available if a specific tooling need is to be addressed.

3 Verification and Validation

This section is dedicated to tools and means for verification and validation.

3.1 Candidates

The list of initial candidates is shown below. Those tools without a a reference on a subsection have not been evaluated further, due to lack of engagement (no partner was willing to sponsor such an evaluation).

System C see 3.1.1.

UPPAAL see 3.1.2.

CPN tools see 3.1.3

RT-Tester see 3.1.4

Tools around Classical B (ProB, SMT solver,...) see 3.1.5

Rodin and Pluggins see 4.1.1

Frama-C see 3.1.6

Matelo

Scade Suite

Fiacre and Tina

Diversity

SPIN

3.1.1 SystemC

Name SystemC

Web site `www.accellera.org/downloads/standards/systemc/about_systemc/`

Licence SystemC Open Source License

Abstract

SystemC is a C++ library providing an event-driven simulation interface suitable for electronic system level design. It enables a system designer to simulate concurrent processes. SystemC processes can communicate in a simulated real-time environment, using channels of different datatypes (all C++ types and user defined types are supported). SystemC supports hardware and software synthesis (with the corresponding tools). SystemC models are executable.

Publications

- D. C. Black, SystemC: From the ground up. Springer, 2010.
- IEEE 1666 Standard SystemC Language Reference Manual, `http://standards.ieee.org/getieee/1666/`
- The ITEA MARTES Project, from UML to SystemC, `http://www.martes-itea.org/`
- J. Bhasker, A SystemC Primer, Second Edition, Star Galaxy Publishing, 2004
- F. Ghenassia (Editor), Transaction-Level Modeling with SystemC: TLM Concepts and Applications for Embedded Systems, Springer 2006

A quantitative evaluation is available in `https://github.com/openETCS/toolchain/blob/master/T7.2/O7.2.1_VnV/O7-2-1_VnV.pdf`

Added value for OpenETCS project

SystemC can be used for a quantitative evaluation of the specification and a system model. Although it could be used for a model of the ETCS system, it has been decided to use SCADE instead. However, the usage of SystemC models as test models will provide a significant benefit to the project. Especially for aspects where timing is important, such as braking curves computation, SystemC can be used for a quantitiative analysis. In addition, performance on different hardware platforms can be evaluated.

Integration in OpenETCS process and toolchain

SystemC can be integrated with SysML:

- Transformation from SysML, e.g., by using Acceleo
- SystemC provides executable models
- Allows performance evaluation with target hardware

3.1.2 UPPAAL

Name UPPAAL

Web site www.uppaal.org

Licence Academic free or commercial license

Abstract

Uppaal is an integrated tool environment for modeling, validation and verification of real-time systems modeled as networks of timed automata, extended with data types (bounded integers, arrays, etc.).

Publications

Short list of publications on the approach (5 max) Please refer to `http://dblp.org/search/#query=uppaal`

A quantitative evaluation is available in `https://github.com/openETCS/toolchain/blob/master/T7.2/O7.2.1_VnV/O7-2-1_VnV.pdf`

Added value for OpenETCS project

- Dedicated tool for modeling with timed automata
- Verification of real-time properties (model checking)
- Graphical simulation and modelling tool

Integration in OpenETCS process and toolchain

Integration would be possible based on model transformation from, e.g., SysML statecharts with timing annotations. This could be achieved with tools such as Acceleo or Xtend.

3.1.3 CPN Tools

Name CPN Tools

Website http://cpntools.org/

Licence Open Source (GPL/LGPL)

Abstract

CPN Tools is a tool for editing, simulating, and analyzing Colored Petri nets.

The tool features incremental syntax checking and code generation, which take place while a net is being constructed. A fast simulator efficiently handles untimed and timed nets. Full and partial state spaces can be generated and analyzed, and a standard state space report contains information, such as boundedness properties and liveness properties.

Publications

Please refer to http://cpntools.org/publications

Slides available on github `https://github.com/openETCS/model-evaluation/blob/master/Telco_Secondary_slides/b-Introduction_CPNTools.pdf`.

A quantitative evaluation is available in `https://github.com/openETCS/toolchain/blob/master/T7.2/O7.2.1_VnV/O7-2-1_VnV.pdf`

Added value for OpenETCS project

Petri Nets as a means of description are used in research an in industrial applications used for Process Modeling, Data analysis, Software design and Reliability engineering. Coloured Petri

Nets are High-level Petri Nets which are mainly used to describe, simulate and validated communication between humans and/or computers. As a means of description Coloured and Hierarchic Petri nets allow to use one uniform means of description for the entire development cycle, starting with the specification through to implementation. Coloured petri nets are standardised as part of the high level petri nets in ISO/IEC 15909 Systems and software engineering - High-level Petri nets. The use of petri nets for the system dependability analysis is standardised in IEC 62551 Analysis techniques for dependability - Petri net modeling. In addition Coloured petri nets and the CPN Tools are introduced and documented in the book *Coloured Petri Nets – Modeling and Validation of Concurrent Systems* by K. Jensen and L.M. Kristensen. CPN Tools is a mature tool suite for coloured petri nets which provides support to edit, check, simulate and analyse nets on all relevant abstraction levels. CPN Tools has a graphical editor to model nets and provides various methods to analyse the nets, most importantly a reachability analysis.

Petri nets are a strictly formal means of description suited for formal proof of behavioural and structural properties. The nets are mainly verified by generation and analysis of the state space. The tool supports the calculation and drawing of the state space, which is used to verify certain logical and temporal properties of the system. Additional model checker can be combined with the tool to provide additional functionalities. In context of the openETCS work coloured petri nets and CPN Tools provide a variable and efficient option for test models to validate the behaviour of the openETCS model. Especially, CPN Tools models can be used to specify and test timing properties.

The simulation engine of CPN tools provides a powerful simulation of petri nets and has a number of debugging functions. It does not support test generation, but provides interfaces for other tools to do so. Correspondingly, tools like SPENAT can be used to generate and manage all kinds of tests for the nets created with CPN Tools.

Integration in OpenETCS process and toolchain

CPN Tools is a mainly open source and for free tool suite which provides a number of interfaces to interact with other modeling tools and simulations. Depending on the specific use of CPN Tools to validate model behaviour or specifications these interfaces can be used co-simulated with SysML or SCADE models to run test cases. CPN Tools can also be as an independent to to just build test models and derive a number of test cases.

3.1.4 RT-Tester

Name RT-Tester

Web site http://www.verified.de/en/products/rt-tester

Licence Mixed:

- Generator: closed
- SMT Solver: open
- GUI (eclipse plug-in): open

Abstract

The RT-Tester test automation tool, made by Verified, performs automatic test generation, test execution and real-time test evaluation. It supports different testing approach such as unit testing, software integration testing for component, hardware/software integration testing and system integration testing. The tool generates automatically behavioral tests that covers the classical criteria such as reachable state, branch and MC/DC. The RT-Tester also implements the so-called model-based testing approach: starting from a test model design with UML/SysML, the RT-tester fully automatically generates test cases. Moreover the test model may be directly linked to the requirements, thus the requirement coverage may also be ensured.

The RT-tester provides the following features :

- Automated Test Case Generation
- Automated Test Data and Test Oracles Generation
- Automated Test Procedure Generation
- Automated Requirement Tracing
- Test Management system
- Test Report Generation

Note that the RT-tester model based testing has be qualified according to ISO 26262.

Publications

- Jan Peleska, Elena Vorobev, and Florian Lapschies. Automated test case generation with smt-solving and abstract interpretation. In Mihaela Bobaru, Klaus Havelund, GerardJ. Holzmann, and Rajeev Joshi, editors, NASA Formal Methods, volume 6617 of Lecture Notes in Computer Science, pages 298–312. Springer Berlin Heidelberg, 2011.
- Jan Peleska, Elena Vorobev, Florian Lapschies, and Cornelia Zahlten. Automated model-based testing with RT-Tester. Technical report, Universität Bremen, 2011.
- Jörg Brauer, Jan Peleska, and Uwe Schulze. Efficient and trustworthy tool qualification for model- based testing tools. In Brian Nielsen and Carsten Weise, editors, Testing Software and Systems, volume 7641 of Lecture Notes in Computer Science, pages 8–23. Springer Berlin Heidelberg, 2012.
- Verified Systems International GmbH. Verified :: Products. http://www.verified.de/en/products.

Added value for OpenETCS project

RT-Tester will add an automated test case and test data generator that interprets test models directly describes in SysML. The test models may specify concurrent sub-components of SUT (System Under Test) and TE (Test Environment), and timing conditions using dense time (i. e., continuous physical time) and an arbitrary number of timers.

Another added value is that the tool also provides a simulation environment that can directly generate C code from the SysML model and run the tests. The code generator may also be adapted to other purposes.

Integration in OpenETCS process and toolchain

- GUI via an eclipse plug-in already exist,
- read SysML model
- Can run SCADE generated code

3.1.5 ClassicalB Tools

Name AtelierB, ProB, SMT solver

Web site Various, see Publciations below.

Licence Mixed:

- Proof Generator: closed
- AtelierB checkers and translators open
- ProB, SMT solvers: open

Abstract

A B model is a textual and formal specification covering the functional be- haviour of a safety critical software. It is usually written based on a high-level specification (informal or formal specification, for example SysML or a natural language). It is gradually refined, starting at the top with an abstract notation and ending at the bottom with sequential instructions — which are then auo- matically translated in a target language such as C or Ada. Thus, we define three objects of verification and validation: the specification, the B model and the generated source code. Validation consists of:

- guaranteeing the functional adequacy between the specification and the model (this can be achieved, for example, through review and proofread- ing),
- building a test environment around the generated source code and test it.

Publications

- AtelierB `http://www.atelierb.eu/en`
- ProB `http://www.formalmind.com`

Slides are available on github `https://github.com/openETCS/model-evaluation/blob/master/Telco_Secondary_slides/ClassicalB_VnV.pdf`.

Detailled use of the tool and results on an example are available on github `https://github.com/openETCS/validation/blob/master/VnVUserStories/VnVUserStorySysterel/04-Results/b-ClassicalB-VnV/BmodelVnV.pdf`.

Added value for OpenETCS project

A set of tools are available to proceed on formal design and verification and validation of critical systems. Some of these tools are :

- open-source
- mature, with large use in railway domain
- qualified for SIL4 design
- allow to cover all the process from software design to executable code generation

Integration in OpenETCS process and toolchain

Such kind of approach is a mature open-source alternative to the Scade toolchain.

3.1.6 Frama-C

Name Frama-C

Web site `http://frama-c.com/`

Licence LGPL 2.1

- Some plug-ins outside of the main distribution are closed-source.

Abstract

Frama-C is a framework dedicated to the analysis of C programs. It comes with a formal specification language, ACSL, that allows to describe the contracts that each function is supposed to fulfill It features a number of plug-ins that perform various verification tasks. In particular, Value analysis is an abstract interpretation-based plugin that can verify the absence of run-time error for any execution of the program, and WP is an Hoare-logic based plug-in that can modularly check that an implementation is conforming to its ACSL specification with the help of automated theorem provers. Frama-C is also meant to be extensible, and it is easy to tailor the generic plugins toward specific VnV tasks.

Publications

- Pascal Cuoq, Florent Kirchner, Nikolai Kosmatov, Virgile Prevosto, Julien Signoles and Boris Yakobowski. Frama-C: a Software Analysis Perspective. Proceedings of SEFM 2012.
- Loïc Correnson and Julien Signoles. Combining Analysises for C Program Verification. Proceedings of FMICS 2012
- Jochen Burghardt, Jens Gerlach, Kerstin Hartig, Hans Pohl and Juan Soto. ACSL by Example, a fairly complete tour of ACSL features through various functions inspired from C++ STL.
- Patrick Baudin, Loïc Correnson and Zaynah Dargaye, WP plugin manual. `http://frama-c.com/download/wp-manual-Fluorine-20130601.pdf`

- Pascal Cuoq, Boris Yakobowski and Virgile Prevosto. Frama-C's value analysis plugin manual. `http://frama-c.com/download/frama-c-value-analysis.pdf`

Slides of Frama-C presentation are available on github `https://github.com/openETCS/model-evaluation/blob/master/Telco_Secondary_slides/c-frama-c-secondary-tools-presentati pdf`.

Specification and verification activities over the `bitwalker` code from Siemens have been conducted by Fraunhofer FOKUS with support from CEA LIST. The development is available on github: `https://github.com/openETCS/validation/tree/master/VnVUserStories/VnVUserStoryFraunhoferFOKUS`

Added value for OpenETCS project

Frama-C can be used mainly for C code verification (either for manually written code or code generated from a non-qualified tool). Abstract interpretation and deductive verification plug-ins allow checking for absence of run-time error (arithmetic overflows, division by zero, invalid pointer dereference,...), as well as verifying functional properties expressed as ACSL contracts.

Although Frama-C has not been formally qualified, there has been some reflection on how a qualification process in the context of DO-178 verification activities would be conducted.

A quantitative evaluation is available in `https://github.com/openETCS/toolchain/blob/master/T7.2/O7.2.1_VnV/O7-2-1_VnV.pdf`

Integration in OpenETCS process and toolchain

An Eclipse plug-in is available (FCDT: `http://gforge.enseeiht.fr/projects/fcdt/`) for launching some analyses from the Eclipse framework.

An interesting subject for the integration of Frama-C in the OpenETCS process concerns the translation from SysML, Scade and/or B models into ACSL specifications against which code could be verified. Such translation from high-level models would also encompass traceability information, in order to relate Frama-C proofs (or alarms) to the appropriate high-level property.

3.2 Selected means and tools

As of this writing, no final tools have been selected. The V&V team will take advantage of the evaluation, as the need for tools arises.

4 Safety support

This section is dedicated to tools and means to support safety analyses.

4.1 Candidates

The list of initial candidates is shown below. Those tools without a a reference on a subsection have not been evaluated further, due to lack of engagement (no partner was willing to sponsor such an evaluation).

Rodin and Plugins see 4.1.1

CPN tools see 4.1.2

Goal Structuring Notation (GSN) see 4.1.3

Safety Architect

4.1.1 EventB, Rodin and plugins

Name Event-B and the Rodin platform

Web site `http://www.event-b.org`

Licence Common Public License Version 1.0 (CPL)

Abstract

Rodin is an open source tool for formal modeling and verification on the system level using the Event-B formalism. Event-B is based on set-theoretic notation of first-order logic (FOL) and has its roots in the B method which has a long history of successful application in industry on software level development.

Rodin is fully integrated into the Eclipse platform and is therefore fully extensible through plug-ins. Existing plug-ins include graphical modeling using state-machines, model simulators, modern state-of-the art SMT solvers and Rational DOORS interoperable requirements tracing using ReqIf documents and ProR.

Publications

- The leaflet [6] contains a short overview of the Rodin tool
- The book [4] explains the usage of Rodin and serves as a gentle introduction into Event-B modeling in Rodin
- The book [1] contains an extensive presentation of Event-B an several modeling examples for different system

- The scientific journal article [2] contains an in-depth look at the integration of Event-B into the Rodin platform

A quantitative evaluation is available in `https://github.com/openETCS/toolchain/blob/master/T7.2/O7.2.1_Safety/O7-2-1_Safety.pdf`

Added value for OpenETCS project

Rodin and Event-B do not directly support a hazard or risk analysis. Their goal is to strengthen the confidence in the correctness of an external safety analysis, by providing means to represent safety requirements (in particular functional requirements) in a formal model and to verify them there or to validate the intended behavior wrt. safety by simulating and observing the model.

Sub-system requirements can be specified and verified, if the formal model contains a representation of the sub-systems. While this can be achieved by refinement, it should be kept in mind that Event-B aims at system-level modeling and analysis, and therefore there could be better alternatives to analyze a very detailed model on implementation level.

The main application of Rodin is to formalize and verify the safety requirements where applicable. This supports the verification of the correctness of the arguments in the safety case, therefore strengthening the confidence in these arguments, but also to provide insight into probably lacking aspects of the safety case.

The Event-B approach is based on iterative refinements form the most abstract model to the desired level of detail. It is therefore a to-down approach, a bottom-up approach does not make sense using Event-B.

And database connection would require the development of additional plug-ins, but would be possible.

VnV of safety requirements is achieved by formal proof and simulation to validate correct functionality.

Traceability is achieved by the connection to ProR.

Generation of some documentation is already supported, as Latex documents can be generated from models. For more extensive documentation, e.g., links with safety requirements, some additional functionality would have to be developed.

Integration in OpenETCS process and toolchain

The Rodin platform is fully based on Eclipse.

The existing graphical modeling plug-ins for Rodin could be connected to Papyrus. This would require the development of a transformation of the different formats.

With SCADE there could be the possibility of interoperation via the SCADE System SysML framework.

With Classical B tools, there is the possibility to generate predicates for guards and invariants directly from the Event-B model. As classical B is based on text files and Event-B on XML file, there would be some development work to do.

4.1.2 CPN Tools

Name CPN Tools

Website http://cpntools.org/

Licence Open Source (GPL/LGPL)

Abstract

CPN Tools is a tool for editing, simulating, and analyzing Colored Petri nets.

The tool features incremental syntax checking and code generation, which take place while a net is being constructed. A fast simulator efficiently handles untimed and timed nets. Full and partial state spaces can be generated and analyzed, and a standard state space report contains information, such as boundedness properties and liveness properties.

Publications

Please refer to http://cpntools.org/publications

Slides available on github `https://github.com/openETCS/model-evaluation/blob/master/Telco_Secondary_slides/b-Introduction_CPNTools.pdf`.

A quantitative evaluation is available in `https://github.com/openETCS/toolchain/blob/master/T7.2/O7.2.1_VnV/O7-2-1_VnV.pdf`

Added value for OpenETCS project

Petri Nets as a means of description are used in research an in industrial applications used for Process Modeling, Data analysis, Software design and Reliability engineering. Coloured Petri Nets are High-level Petri Nets which are mainly used to describe, simulate and validated communication between humans and/or computers. As a means of description Coloured and Hierarchic Petri nets allow to use one uniform means of description for the entire development cycle, starting with the specification through to implementation. Coloured petri nets are standardised as part of the high level petri nets in ISO/IEC 15909 Systems and software engineering - High-level Petri nets. The use of petri nets for the system dependability analysis is standardised in IEC 62551 Analysis techniques for dependability - Petri net modeling. In addition Coloured petri nets and the CPN Tools are introduced and documented in the book *Coloured Petri Nets – Modeling and Validation of Concurrent Systems* by K. Jensen and L.M. Kristensen. CPN Tools is a mature tool suite for coloured petri nets which provides support to edit, check, simulate and analyse nets on all relevant abstraction levels. CPN Tools has a graphical editor to model nets and provides various methods to analyse the nets, most importantly a reachability analysis.

Petri nets are a strictly formal means of description suited for formal proof of behavioural and structural properties. The nets are mainly verified by generation and analysis of the state space. The tool supports the calculation and drawing of the state space, which is used to verify certain logical and temporal properties of the system. Additional model checker can be combined with the tool to provide additional functionalities. In context of the openETCS work coloured petri nets and CPN Tools provide a variable and efficient option for test models to validate the behaviour of the openETCS model. Especially, CPN Tools models can be used to specify and test timing properties.

The simulation engine of CPN tools provides a powerful simulation of petri nets and has a number of debugging functions. It does not support test generation, but provides interfaces for other tools to do so. Correspondingly, tools like SPENAT can be used to generate and manage all kinds of tests for the nets created with CPN Tools.

4.1.3 Assurance Case Argument Editor

Name Assurance Case Argument Editor (ACedit) as an editor for the Goal Structuring Notation (GSN)

Web site https://code.google.com/p/acedit/

Licence Eclipse Public License 1.0

Abstract

A safety case is a collection of documents, which shall demonstrate that the system fulfills conditions to guaranty a safe behavior. Therefore, an argumentation chain has to be constructed to demonstrate that from the evidence available it can be reasonably concluded that a system is acceptably safe. The Goal Structuring Notation (GSN) is a known approach for constructing such argumentation chains, which refer to the explicit documentation of claims about a system, and explanation of how these claims are convincingly supported by evidence generated during different activities in design, verification and validation of that system. The ACedit is a tool implementation of this approach realized as eclipse plug-ins under the Eclipse project. The tool supports the graphical construction of argument chains in GSN and a number of model management techniques that apply to them.

The key features are the following

1. Graphical editor for the GSN, based on the GSN standard
2. Graphical editor for the ARM, based on the ARM metamodel available by sysa.omg.org
3. Model management tasks for the GSN editor like Model validation and In-place model transformation
4. Model transformation between GSN and ARM models

The graphical editors of ACedit are based on the definition of GSN and ARM metamodels, which are defined in the EMFatic language.

Publications

- George Despotou, Aris Apostolakis, Dimitris Kolovos: Assuring Dependable and Critical Systems: Implementing the Standards for Assurance Cases with ACedit

- Tim Kelly and Rob Weaver: The Goal Structuring Notation – A Safety Argument Notation

Slides are available on github in `https://github.com/openETCS/model-evaluation/blob/master/Telco_Secondary_slides/psn_20130311_Tool-Presentation-GSN_1-0_jw.pdf`

A quantitative evaluation is available in `https://github.com/openETCS/toolchain/blob/master/T7.2/O7.2.1_Safety/O7-2-1_Safety.pdf`

Added value for OpenETCS project

To structure the safety case with GSN allows us to present a clear outline of the argumentation chain and the specific relations between the numerous documents. In this way a consistent generic safety case for the overall development of an on-board system based on the openETCS development process can be presented as a GSN model. All documents already created in the openETCS work can be linked to the GSN argumentation structure to present an easy overview about the safety case and the status the documents and their relation. This creates a common understanding of the openETCS safety argumentation and helps to show the specific context for their work to all different working groups.

Integration in OpenETCS process and toolchain

The safety case task will establish based on the ongoing work of design and verification and validation activities the underlying argumentation chain, which shall demonstrate that the openETCS development process fulfills required quality and safety aspects. This will first be done by modeling the theoretical argumentation chain given by the development methodology. As the work goes on the general steps in the argumentation chain are enhanced through the actual artifacts which are created during the design and corresponding verification steps. the same will be done with all validations results for the design artifacts. As the ACedit editor is already based on Eclipse the existing to can be integrated in the basic tool chain environment to link the argumentation chain elements to specific design elements. All purely document based argumentation and conclusion parts will be linked to the GSN model via their presentation in the git repository. Therefore, a connection between the ACedit tool and the GIT repository has to be established. To do this the already existing options in the eclipse framework shall be used.

4.2 Selected means and tools

As of this writing, no final tools have been selected. The V&V team will take advantage of the evaluation, as the need for tools arises.

5 Model transformation and Code generation

This section is dedicated to tools and means for model transformation and code generation.

5.1 Candidates

The list of initial candidates is shown below. Those tools without a a reference on a subsection have not been evaluated further, due to lack of engagement (no partner was willing to sponsor such an evaluation).

Scade Suite

Rodin and Pluggins

Acceleo see 5.1.1.

ATL

QVTO and SmartQVT

Xtend

5.1.1 Acceleo

Name Acceleo

Web site http://www.eclipse.org/acceleo/

Licence Eclipse

Abstract

Acceleo is an implementation of the Object Management Group (OMG) MOF Model to Text Language (MTL) standard. Based on a special template language model to text transformations can be defined. It is fully integrated with Eclipse and also part of Polarsys.

Publications

- Most information is available on the homepage http://www.eclipse.org/acceleo/

Added value for OpenETCS project

Acceleo as a model-to-text (M2T) transformation language supports generating textual artifacts from templates. Therefore, M2T transformation is ideally suited for code generation purposes e.g. form abstract SysML models to make them executable and testable. Providing executable specification will provide a momentous benefit to the project. In the openETCS project, Acceleo will be used to have a transformation from SysML models to IEEE standardized language SystemC.

Integration in OpenETCS process and toolchain

- Acceleo is based on the Eclipse Modeling Framework (EMF)
- This brings compatibility with any tool that produces EMF compatible models.

5.2 Selected means and tools

We did not explicitly evaluate the Eclipse Modeling Framework (EMF), which has already been chosen as part of the primary toolchain (D7.1). EMF has powerful model transformation capabilities which may be sufficient for many tasks. Beyond that, no other final tools have been selected.

6 Conclusion

The challenges posed by evaluating secondary technologies turned out to be quite different than those from the primary technologiees (see D7.1). The primary tools created some strong opinions and discussions at time, due to the central nature of the primary tools. In contrast, it was hard to engage partners in evaluating secondary tools: First, because the imporance was not as visible, and second, because there was no immediate need. Due to the availability of the primary tools, work could get started.

In spite of these challenges, we believe that we created a solid reference that will make selection of secondary tools on an “as needed” basis easy and quick. There is a good chance that new, unexpected tool requirements will apear during the reminder of this project. In that case, this report will be amended with additional evaluations.

Appendix: References

[1] Jean-Raymond Abrial. *Modeling in Event-B: System and Software Engineering*. Cambridge University Press, New York, NY, USA, 1st edition, 2010.

[2] Jean-Raymond Abrial, Michael Butler, Stefan Hallerstede, Thai Son Hoang, Farhad Mehta, and Laurent Voisin. Rodin: an open toolset for modelling and reasoning in Event-B. *STTT*, 12(6):447–466, 2010.

[3] Sylvain Baro and Jan Welte. Requirements for openETCS. Technical Report D2.6, OpenETCS, 2013.

[4] Michael Jastram (Ed.). Rodin user's handbook. `http://handbook.event-b.org`, 2012.

[5] Marielle Petit-Doche and WP7 Participants. D7.1: Report on the final choice of the primary toolchain. Primary Toolchain OETCS/WP7/D7.1, openETCS, July 2013.

[6] Systerel. `http://sourceforge.net/projects/rodin-b-sharp/files/Doc_Rodin_General/Rodin_Leaflets/Leaflet_Rodin_E.pdf/download`, 2012.