Tool Support for Engineering Certifiable Software[2]

László Gönczy[1], István Majzik[1], Ákos Horváth[1], Dániel Varró[1], András Pataricza[1]

Department of Measurement and Information Systems
Budapest University of Technology and Economics
Budapest, Hungary

Abstract

Formal methods can effectively support the model driven develoment and analysis of IT applications in many domains. Typically, the domain-specific engineering models are transformed to formal analysis models (to compute measures that help the designer in verifying the design decisions) and verified models are mapped to test and implementation related software artefacts. An overview of four European projects demonstrates the use of support tools and tool integration facilities in development processes of systems having in sight the demand of certification according to domain-specific standards.

Keywords: Keywords: Model based design, model transformation, tool integration.

1 Introduction

The *Fault Tolerant Systems Research Group* of the Dept. of Measurement and Information Systems at the Budapest University of Technology and Economics has long tradition in model driven develoment and analysis of IT applications in many domains. Formal methods are considered as integral part of the analysis methods: the engineering models (primarily UML and domain-specific models) are transformed to formal analysis models (e.g., Petri-nets, Kripke structures) to compute measures that can be back-annotated to the designer. Verified models are then mapped to test and implementation related software artefacts (e.g., deployment configuration, source code). The integration of the suporting tools into the design process required the development of means for efficient tool integration.

In the recent years the Group participated in several European projects that have the aim to develop *methods* or specific *applications* that have the potential to be included in development processes or systems certifiable according to domain-specific

[1] Email: [gonczy,majzik,ahorvath,varro,pataric]@mit.bme.hu

[2] This work was partially supported by the following European projects: DECOS (IST-511764), DIANA (AERO1-030985), SAFEDMI (SUSTDEV-031413) and SENSORIA (IST-3-016004).

This paper is electronically published in
Electronic Notes in Theoretical Computer Science
URL: www.elsevier.nl/locate/entcs

standards. In this paper we give an overview of four projects and demonstrate their results that are related to the engineering of certifiable software:

- In the DECOS (Dependable Embedded Components and Systems) project (see Section 2) we elaborated a technology for the integration of tools in a certifiable verification and validation process.
- In the SAFEDMI (Safe Driver Machine Interface for ERTMS Automatic Train Control) project (see Section 3) we developed tools that support design and analysis methods prescribed by the standards for railway software development.
- In the SENSORIA (Software Engineering in Service-Oriented Overlay Computers) project (see Section 4) we contributed to the development method that supports the use of formal methods for creating certifiable service orchestrations.
- The DIANA (Distributed, equipment Independent environment for Advanced avioNic Applications) project (see Section 5) defines the development and certification means needed to support an Integrated Modular Electronics platform.

2 Tool Integration for Verifying Component-Based Dependable Embedded Systems

The DECOS project [1] aimed at creating a model-based development framework for critical embedded applications (e.g. in automotive, avionics or industrial process control systems). The toolchain architecture was designed for certifiability: it supports systematic verification and validation (V & V) actions in a customizable way. Verification workflows are derived from the requirements of the related standards, e.g., ISO/IEC 61508.

Our role in the toolchain development was i) *developing model transformations* in the VIATRA2 framework [6], ii) *design of the automated testbench* and iii) *implementing the underlying message-based infrastructure* which provided an asynchronous, reliable and automated communication environment for tool integration.

DECOS followed an MDA-conform approach where the system was first designed in a high level modeling language, UML (using a standard, DECOS-specific extension), SCADE [4] or in a domain-specific editor. This Platform Independent Model is then mapped to Platform Specific Models which are the basis of deployment configuration and code generation. These transformations are implemented within an Eclipse-based environment. During the development, requirements are attached to each models (or DECOS-artefacts), which implicate the appropriate V-plans to be executed in order to comply with a *safety case*.

Predefined, standards-compliant V-plans contain separate V and V activities for different models. Such activities include *formal validation and verification*, *fault injection*, *FMEA*, *simulation*, systematic *testing* (white box, black box, coverage, etc.) or even *radiated emission* test for dependable embedded hardware. Some of these activities need user interaction or are executed on a hardware artefact, therefore the test bench support different levels of automation for V-plans.

Ontology-based semantical validation of models is a sample V & V activity. Here UML metamodels and models are transformed to the input language of RACER to execute semantical queries. These queries can validate metamodel consistency,

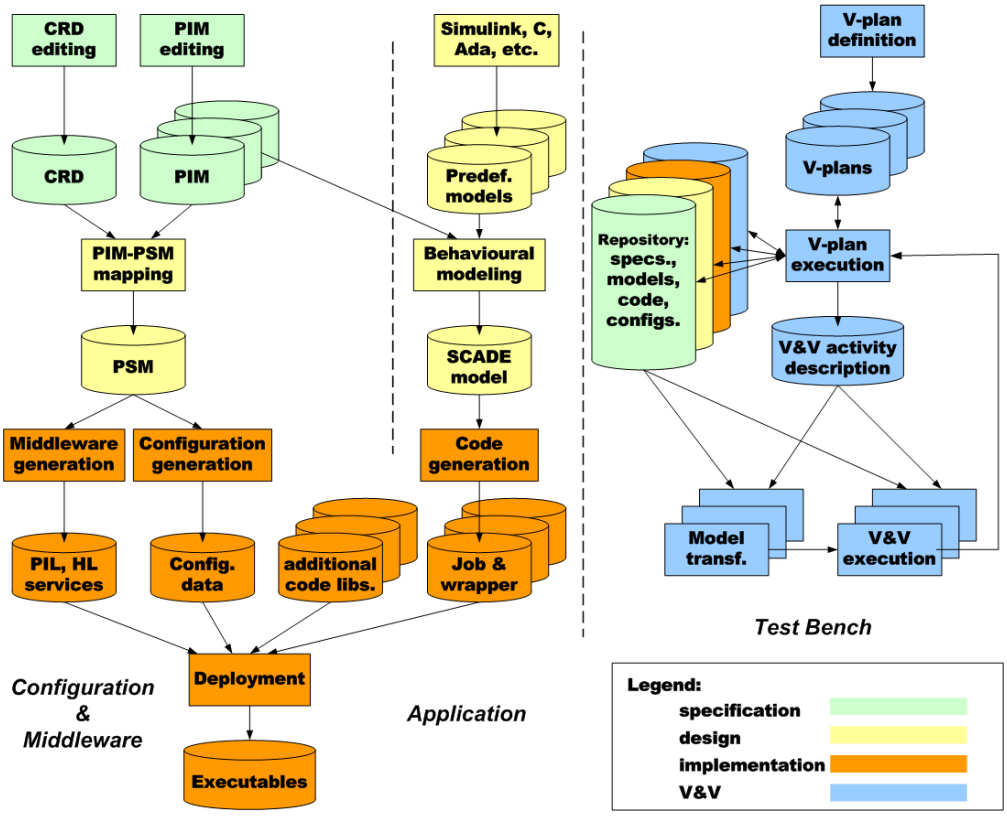

Fig. 1. DECOS Test bench configuration

completeness and correctness of models (i.e., metamodel-conformance) and domain specific requirements on model instances.

3 Tools for the Development of a Safety-Critical Railway Application

The objective of the SAFEDMI project [3] is to design and develop a Driver-Machine Interface (DMI) compatible with the European Train Control System. The DMI shall be able to satisfy at least SIL2 (Safety Integrity Level 2) according to European standards for railway applications. The safety issues to be tackled by the SAFEDMI project are related to visualization, driver input data acquisition, data communication between on-board system components, and wireless communication for configuration and maintenance.

3.1 Tool support for a SIL 2 development process

The SIL 2 assessment and certification requires a rigorous development process according to EN 50129 and EN 50128. These standards prescribe a combination of methods and techniques that are categorized as mandatory, highly recommended or recommended. Several methods can be effectively supported by automated tools. In this paper we mention two tools that were developed or adapted in the SAFEDMI project to support the following methods:

- *Quantitative evaluation of availability and safety*: According to the standards, the quantified evaluation of random failure integrity shall be carried out by means of probabilistic calculations. To do this, a so-called dependability model is constructed. It is a mathematically precise model (in the form of a Stochastic Activity Network) representing the failure behaviour of system components, error propagation among them according to the given architecture, and the developed error detection and recovery mechanisms. Accordingly, the dependability model is composed of smaller sub-models belonging to system components, interactions, and the additional mechanisms. Although the definition of these sub-models is a task of dependability experts, their assembly can be performed by an automated tool on the basis of the UML architecture (class and object diagrams) of the DMI. The solution of the dependability model provides (i) the system-level availability and (ii) the hazardous failure rate that must not exceed the allowed value (tolerable hazard rate, THR).
- Robustness testing: Robustness of the DMI and its software components can be characterised by testing the responses to exceptional scenarios like extreme parameters in API calls, invalid sequence of interactions (in communication protocols, internal interactions or mode changes). Robustness tests are constructed automatically on the basis of the UML class diagrams (annotated with the ranges of acceptable values in the parameters of methods) and the sequence diagrams specifying typical interactions. The tool generates the test calls with combinations of invalid and valid parameter values, and applies pre-defined mutation operators (e.g., omission, change) in test sequences derived from the sequence diagrams.

4 Developing Service Oriented Architecture with Support of Formal Methods

A primary goal of the Sensoria project [5] is to provide support for different stakeholders and actors during the entire project lifecycle for developing service-oriented overlay systems of a justifiable quality. SENSORIA proposes a model-driven approach for the entire development cycle of services based applications and infrastructures including the design, the formal analysis, the deployment and re-engineering of services. The core ideas of the SENSORIA engineering approach are illustrated in Fig. 4.

The development method supports the use of precise formal methods for creating certifiable service orchestrations. The *SENSORIA Development Environment* is an Eclipse-based framework which enables the creation of tool repositories, a shared data pool and provides access for tool GUIs as well. Tool features are provided by OSGI services which also facilitate the wrapping of remote Web services.

4.1 Service certification

Although service-oriented systems generally do not have such strict certification requirements as embedded devices, some areas (e.g. banking processes) have domain-specific safety criteria. Deep semantic analysis of service orchestration can validate the fulfillment of such criteria. SENSORIA offers a set of sound model-based formal

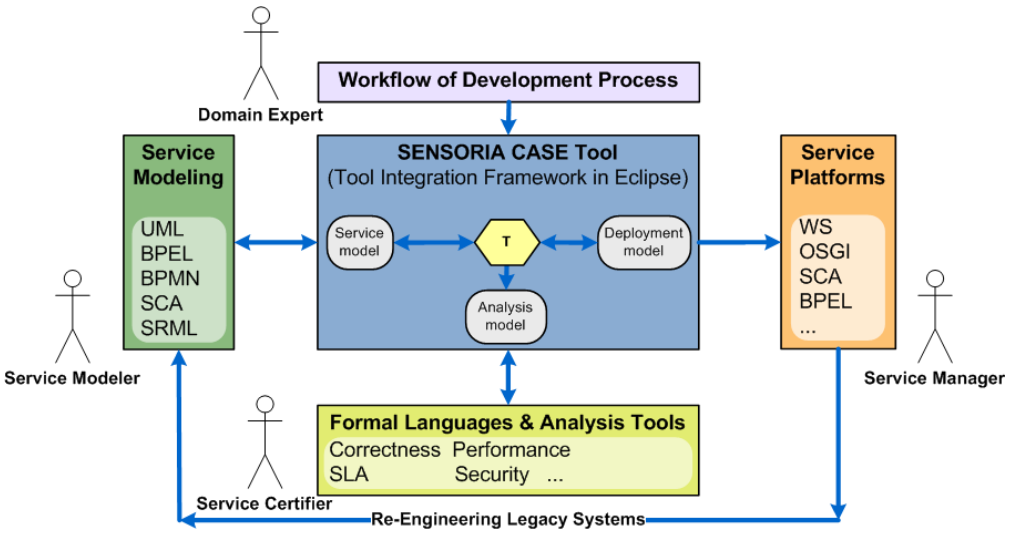

Fig. 2. The SENSORIA engineering approach

analysis features.

The project also targets to cover the MDA lifecycle with *deployment configuration generation*. Recent and ongoing research activities aim at integrating WSDL (Web Service Description Language), BPEL (Business Process Execution Language) and SCA (Service Component Architecture) generation from semantically validated engineering models.

5 Certification Means for Object-Oriented Avionics Applications

The DIANA Project [2] is the first step for the implementation of an enhanced avionics platform, named AIDA (Architecture for Independent Distributed Avionics), providing secure distribution and execution on virtual machines to avionics applications. Along with this objective, DIANA also aims at contributing to the definition and standardization of the development and certification means based on Model Driven System Development needed to support this novel platform. It is important to mention that the envisaged development means for the AIDA platform inherits the ideas introduced by the DECOS tool chain introduced in Sec. 2.

In the current paper we focus on the contract based specification approach proposed in the DIANA project as potential certification means for future avionics standards.

5.1 System Requirement Specification by Contracts

System requirement specification (SRS) provides a *black box* description of what the system should do, in terms of the interaction of the system with its external environment. Based on the definition of SRS by means of *usage domains* defined by the RTCA DO297, Integrated Modular Avionics (IMA) Development, Guidance and Certification considerations. DIANA aims to adopt this approach by introducing *design-by-contract* not only on system – as defined by the usage domain – but also

on application, platform and environment level.

Defined over the concept of pre-/postcondition *design-by-contract* prescribes that system architects should define precise verifiable interface specifications (pre/-postconditions) for system components based upon the theory of abstract data types and the concept of a business contract. This means that contracts provides semantics to formally describe the behavior of a module or platform, removing potential ambiguity with regard to the concrete implementation.

By using the combination of SysML, OCL and JML for capturing contracts on different levels of design and implementation, DIANA aims to integrate contract definition, analyze and validation to its tool chain. This integration will enable to use SysML requirement diagrams and OCL to capture and map the contracts on platform independent level and – based on the platform mapping – automatically derive the platform specific JML contracts to the implementation level.

Current tool development in the project – to support this idea – focuses on (i) to enhance the model bus (based on VIATRA2) to support mapping and maintenance of contracts attached to PIM and PSM models, (ii) to evaluate the use of static checkers (ESC/ Java2) and formal theorem provers (KeY) to check the correctness of critical source code segments against its specification given by contracts, (iii) to examine the use of automatic test case generation based on the contracts, and (iv) to verify property preservation of model transformations used in the model bus to ensure that the correctness of formal analysis carried out during the development process cannot be corrupted by erroneous model transformation.

6 Conclusion

This paper demonstrated by citing four European projects that tools based on formal methods and mathematically precise model transformations can be effectively integrated into the design process of systems with the set purpose of certification according to domain-specific standards. The tools presented in this paper demonstrated the wide range of design and verification aspects that can be supported: among others ontology-based checking of models, dependability analysis of the architecture, configuration generation were mentioned. The DECOS tool integration approach demonstrated the support of a tailorable certification process.

References

[1] DECOS (Dependable Embedded Components and Systems) EU FP6 Project. http://www.decos.at/.

[2] DIANA (Distributed, equipment Independent environment for Advanced avioNic Application) EU FP6 Project. http://diana.skysoft.pl/.

[3] SAFEDMI (Safe Driver Machine Interface for ERTMS Automatic Train Control) EU FP6 Project. http://www.safedmi.org/.

[4] SCADE Suite. http://www.esterel-technologies.com/products/scade-suite/.

[5] SENSORIA (Software Engineering in Service-Oriented Overlay Computers) EU FP6 Project, 2005. http://sensoria-ist.eu.

[6] VIATRA2 Framework at Eclipse GMT. http://www.eclipse.org/gmt/.