Project no.:	**ICT-FP7-STREP-214755**
Project full title:	**Quantitative System Properties in Model-Driven Design**
Project Acronym:	**QUASIMODO**

Deliverable no.: D1.4

Title of Deliverable: Modeling Tools

Contractual Date of Delivery to the CEC:	**Month 36**
Actual Date of Delivery to the CEC:	**Month 40 (June 7, 2011)**
Organisation name of lead contractor for this deliverable:	**P02 ESI/RU**
Author(s):	**Frits Vaandrager**
Participant(s):	**P01 AAU, P02 ESI/RU, ESI/UT, P04 RWTH, P05 SU**
Work package contributing to the deliverable:	**WP1**
Nature:	**R**
Version:	**1.0**
Total number of pages:	**9**
Start date of project:	1 Jan. 2008 **Duration:** 36 month

Project co-funded by the European Commission within the Seventh Framework Programme (2007-2013)

Dissemination Level

PU	Public	X
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	

Abstract:

This deliverable presents the research on modeling tools that has been carried out within Task 1.3 of the QUASIMODO project.

Keyword list:

Contents

Abbreviations

AAU: Aalborg University, DK

ESI: Embedded Systems Institute, NL

ESI/RU: Radboud University Nijmegen, NL

ESI/UT: University of Twente, NL

RWTH: RWTH Aachen University, D

SU: Saarland University, D

1 Introduction

This deliverable presents an overview of the work on modeling tools that has been carried out within Task 1.3 of the QUASIMODO project.

The main goal of QUASIMODO is to develop new techniques and tools for model-driven design, analysis, testing and code-generation for advanced embedded systems where ensuring quantitative bounds on resource consumption is a central problem. Since the project operates at the forefront of knowledge, several of these tools (such as the Fortuna tool, which can handle the combination of probabilistic, real-time and cost features) are academic prototypes which focus on analysis, and limited support for modeling industrial sized designs. The mathematical models that are manipulated by these tools (e.g. semi Markov chains, Markov decision processes, or probabilistic timed automata) are too fine-grained to be directly used as specification means by an embedded systems designer. The modeling of realistic embedded systems directly in terms of these flat automata-based models is unmanageable, and even for simple systems, the size and complexity gets out of hand. To enable the use of such techniques by system engineers, the modeling notation must be expressive, simple, easy-to-use, and closely fit with the techniques aready used. The goal of Task 1.3 is to explore the integration of our techniques into the embedded software engineering life-cycle, by linking them to design notations and tools that are widely used within the embedded systems industry.

The *Description of Work* of QUASIMODO is centered around extensions of UML statecharts, a modeling technique that is heavily used in embedded system design such as the automotive and aerospace industry. Task 1.3 has originally been tailored to the needs of the industrial tool provider Inchron GmbH, a partner in the consortium who decided to drop out at the start of the project. While we continued to explore statechart-based notations, this drop implied that the concrete goals and especially the notation to be used were no longer definite.

QUASIMODO has been actively pursuing four different routes towards integration of its techniques within industrial design notations:

1. Study the extension of Statechart dialects with timed, cost, stochastic or hybrid features.
2. Further increase the industrial impact of one tool, Uppaal, to the point where it is used easily and routinely by embedded systems engineers.
3. Develop lightweight approaches for integration, driven by the needs of industrial sized case studies.
4. Investigate translations from industrial modeling languages to QUASIMODO tools.

We will elaborate on results obtained in each of these four areas in the next sections of this deliverable.

2 Quantitative Extensions of Statecharts

Results Current industrial practice of model-based analysis is supported by state-transition diagrammatic notations such as Statecharts. State-of-the-art modelling tools like STATEMATE

support safety and failure-effect analysis at design time, but restricted to qualitative properties. In [3], we report on a (plug-in) extension of STATEMATE enabling the evaluation of quantitative dependability properties at design time. The extension is compositional in the way the model is augmented with probabilistic timing information. This fact is exploited in the construction of the underlying mathematical model, a uniform continuous-time Markov decision process, on which we are able to check requirements of the form: "The probability to hit a safety-critical system configuration within a mission time of 3 hours is at most 0.01." We give a detailed explanation of the construction and evaluation steps making this possible, and report on a nontrivial case study of a high-speed train signalling system where the tool has been applied successfully.

Perspective We consider this work as a possible blueprint for an extension of an almost arbitrary given Statechart dialect, with timed, cost, stochastic or hybrid features. This is possible since the approach leaves the chosen original Statechart semantics untouched. Instead, it interfaces on the level of the underlying transition system semantics and allows for a notationally crisp and lightweight specification of quantitative information.

3 Uppaal

Results A major step in the dissemination of timed automata model checking technology towards industry was the founding of Up4All, the company that aims at make the user-friendly formalism of timed automata available to engineers, through a commercial version of the Uppaal tool. Although originally developed in academia, the maturity of the Uppaal tool has much improved over the years. The graphical syntax for EFSMs and the C-like syntax are easy to understand for engineers, and very close to notations they use anyway. For a detailled report on the various extensions and improvements to the Uppaal tool that have been realized by QUASIMODO, we refer to Deliverable 5.9.

The main contribution of QUASIMODO here consists of a series of industrial sized case studies (the Chess WSN, the Chessway, the Hydac case, the Herschell/Planck satellite software architecture case from Terma, verification of printer datapaths at Océ, the Zeroconf protocol, the ASML case study) described in more detail in Deliverable 5.10. These case studies shed more light on the scope of applicability of the Uppaal toolset. The Uppaal tool and several of these case studies are discussed at length in the Industrial Handbook. Numerous tutorials and courses on Uppaal were presented at international meetings for both academic and industrial audiences.

Perspective Uppaal has reached the point where it can be used routinely to solve industrial sized problems. Some of the extensions of Uppaal studied within QUASIMODO (testing with Tron, controller synthesis with Tiga, and probabilistic verification with Pro) are still less mature than classic Uppaal, but their integration within the commercial version of Uppaal can be accomplished within years. Real industrial take-up of the QUASIMODO techniques and tools requires tight integration with existing industrial modelling approaches such as UML and Matlab/Simulink. Although prototypes translations and lightweight approaches have been studied

within QUASIMODO, full integration with industrial quality tools will require a major investment. By continuing to demonstrate (and further improve) the effectiveness of Uppaal, we generate the necessary motivation for achieving full integration with other industrial tools.

4 Lightweight Approaches

4.1 Hydac Case Study

The design of controllers for embedded systems is a difficult engineering task. In the case study proposed by Hydac, we managed to enforce safety properties in an efficient way, such that they consume the nearly least possible amount of energy. We devised a systematic way to develop models and to use a chain of automatic tools for the synthesis, verification and simulation of a provably correct and near optimal controller for a real industrial equipment. To solve the HYDAC control problem, we use three complementary tools for three different purposes: UPPAAL-TIGA for synthesis, PHAVER for verification, and SIMULINK for simulation. For more details on this successful case study, we refer to Deliverable 5.7.

4.2 Scenario-Based Analysis and Synthesis

In [8], we propose an automated, tool-supported approach to scenario-based analysis and synthesis of real-time embedded systems. The inter-object behaviors of a system are modeled as a set of live sequence charts (LSCs), and the scenario-based user requirement is specified as a separate LSC. By translating the set of LSC charts into a behavior-equivalent network of timed automata (TA), we reduce the problems of model consistency checking and property verification to classical CTL real-time model checking problems, and reduce the problem of centralized synthesis for open systems to a timed game solving problem. We implement a prototype LSC-to-TA translator, which can be linked to existing real-time model checker UPPAAL and timed game solver UPPAAL-TIGA. Preliminary experiments on a number of examples show that it is a viable approach.

5 Integration with Industrial Modeling Languages

In this section, we report on two translations from industrial modeling languages to QUASIMODO tools, which have been carried out by project members RWTH and ESI/RU. These efforts were largely funded by other projects (COMPASS and OCTOPUS, respectively), but were carried out in close collaboration with and involvement of QUASIMODO researchers, have been discussed extensively during QUASIMODO meetings and working visits, and provided inspiration for further research and development within QUASIMODO.

5.1 AADL

AADL, the standardized Architecture Analysis and Design Language (AADL) modelling framework, is gaining widespread acceptance in aerospace, automobile and avionics industries for comprehensively specifying safety-critical systems by capturing functional, probabilistic and hybrid aspects. Partner RWTH has been involved in a project, funded by the European Space Agency, on correctness, modelling, and performance of aerospace systems (COMPASS). Within the COMPASS project a tight integration has been established between the AADL framework and the MRMC probabilisticmodel checker that has been further developed within QUASIMODO.

Results In [4], we present a component-based modelling approach to system-software co-engineering of real-time embedded systems, in particular aerospace systems. Our method is centred around the AADL modelling framework. We formalize a significant subset of AADL, incorporating its recent Error Model Annex for modelling faults and repairs. The major distinguishing aspects of this component-based approach are the possibility to describe nominal hardware and software operations, hybrid (and timing) aspects, as well as probabilistic faults and their propagation and recovery. Moreover, it supports dynamic (i.e. on-the-fly) reconfiguration of components and inter-component connections. The operational semantics gives a precise interpretation of specifications by providing a mapping onto networks of event-data automata. These networks are then subject to different kinds of formal analysis such as model checking, safety and dependability analysis and performance evaluation. Mature tool support realizes these analyses.

In [5], we present a graphical toolset for verifying AADL models. Analyses are implemented on top of mature model checking tools and range from requirements validation to functional verification, safety assessment via automatic derivation of FMEA tables and dynamic fault trees, to performability evaluation, and diagnosability analysis. The COMPASS approach and toolset was intensively tested on serious industrial cases by Thales Alenia Space in Cannes (France). These cases include thermal regulation in satellites and satellite mode management with its associated FDIR strategy. It was concluded that the modeling approach based on AADL provides sufficient expressiveness to model all hardware and software subsystems in satellite avionics. The hierarchical structure of specifications and the component-based paradigm enables the reuse of models. Also incremental modeling is very well supported. The RAMS analyses as provided by the toolset were found to be mature enough to be adopted by industry, and the corresponding results allowed the evaluation of design alternatives. Current investigations indicate that the integrated COMPASS approach significantly reduces the time and cost for safety analysis compared to traditional on-board design processes [9].

Perspective Whereas the current AADL tool is graphical based, the input is a textual AADL description. The ESA has funded an extension of the COMPASS project in which a graphical version of AADL will be used as input.

5.2 The Octopus Toolset

Octopus is a joint project of a consortium of Dutch industrial and academic partners. The Embedded Systems Institute (ESI) has the responsibility for project management and knowledge dissemination. Océ-Technologies B.V., the carrying industrial partner, provides the industrial challenge, expert knowledge in the domain of digital document printing systems (DDS).

A common challenge during embedded system development is the need to explore extremely large design spaces, involving multiple metrics of interest (timing, resource usage, energy usage, or cost). The number of design parameters (number and type of processing cores, sizes and organization of memories, interconnect, scheduling and arbitration policies) is typically very large and the relation between parameter settings and design choices on the one hand and metrics of interest on the other hand is often difficult to determine. Given these observations, embedded-system design trajectories require a systematic approach that is automated as far as possible. One of the goals of Octopus is to develop tools and techniques to support Design-Space Exploration (DSE) for the datapath of printers.

Results In [2], we introduce the Octopus DSE toolset that aims to leverage existing modeling, analysis, and DSE tools to support model-driven DSE for embedded systems. The current toolset integrates Uppaal and CPN Tools, and is centered around the DSE Intermediate Representation (DSEIR) that is specifically designed to support DSE. The toolset architecture allows: (i) easy reuse of models between different tools, while providing model consistency, and the combined use of these tools in DSE; (ii) domain-specific abstractions to support different application domains and easy reuse of tools across domains.

The Octopus DSE toolset follows the popular Y-chart philosophy [1, 7]. This philosophy is based on the observation that embedded systems development typically involves the co-development of an application, a platform, and the mapping of the application onto the platform. In the Y-chart philosophy, specification of applications, platforms and mappings are separated. This allows independent evaluation of various alternatives of one of these system aspects while fixing the others.

Inspired by work on model-based design of printers, the DSEIR language supports the use parametrized partial orders (PPO) for describing applications. PPOs are a simple extension of partial orders, expressive enough to compactly represent large task graphs with repetitive behavior. In [6], we present a translation from a subclass of PPOs to Uppaal together with a proof that the transition system induced by the Uppaal model is isomorphic to the configuration structure of the original PPO. Moreover, we report on a series of experiments which demonstrates that Uppaal models obtained through this translation are more tractable than handcrafted models of the same systems used in earlier case studies. Our results boost the verification power of the Octopus design-space exploration toolset

Figure 1 gives an example of how applications are specified in DSEIR. Rectangles encode tasks which contain the instance number between the square brackets and the task duration between parentheses. The `s` and `e` subrectangles represent the start and end event types. Arrows indicate the precedence edges between event types. Their labels show the update function and the resources handed over. For simplicity, we did not depict the arrow between the `s` and the `e`

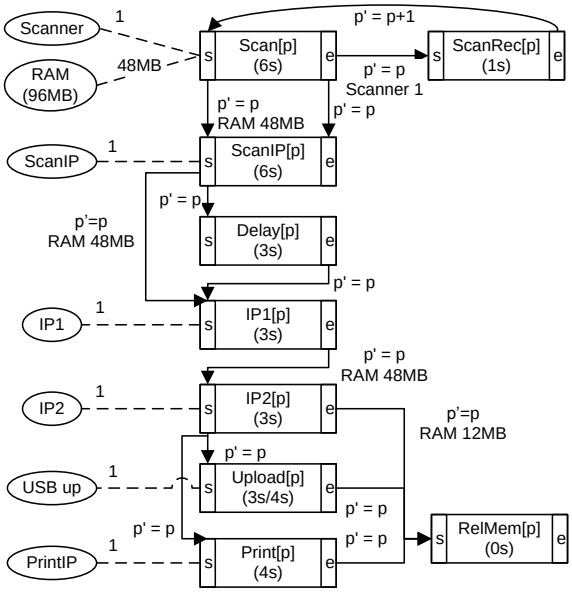

Figure 1: Process from Store Case

event types of the same task and we also assume a finite number of task instances. The circles encode resources and the parentheses contain their maximum capacity (one if not mentioned). The dashed lines represent resource claims. The difference between the claimed and handover amount is released at the end of a task.

Perspective ESI intends to continue the further development of the DSEIR toolset beyond the Octopus project, aiming at application not just within Océ, but also within other companies.

References

[1] F. Balarin. *Hardware-software co-design of embedded systems: the POLIS approach*. Kluwer, 1997.

[2] T. Basten, Benthum E. van, M. Geilen, M. Hendriks, F. Houben, G. Igna, F. Reckers, Smet S. de, L. Somers, E. Teeselink, N. Trcka, F. Vaandrager, J. Verriet, M. Voorhoeve, and Y. Yang. Model-driven design-space exploration for embedded systems: The octopus toolset. In T. Margaria and B. Steffen, editors, *Leveraging Applications of Formal Methods, Verification, and Validation - 4th International Symposium on Leveraging Applications, ISoLA 2010, Heraklion, Crete, Greece, October 18-21, 2010, Proceedings, Part I*, volume 6415 of *Lecture Notes in Computer Science*, pages 90–105. Springer, 2010.

[3] Eckard Böde, Marc Herbstritt, Holger Hermanns, Sven Johr, Thomas Peikenkamp, Reza Pulungan, Jan Rakow, Ralf Wimmer, and Bernd Becker. Compositional dependability evaluation for STATEMATE. *IEEE Trans. Software Eng.*, 35(2):274–292, 2009.

[4] Marco Bozzano, Alessandro Cimatti, Joost-Pieter Katoen, Viet Yen Nguyen, Thomas Noll, and Marco Roveri. Safety, dependability and performance analysis of extended aadl models. *The Computer Journal*, 2010.

[5] Marco Bozzano, Alessandro Cimatti, Joost-Pieter Katoen, Viet Yen Nguyen, Thomas Noll, Marco Roveri, and Ralf Wimmer. A model checker for aadl. In Tayssir Touili, Byron Cook, and Paul Jackson, editors, *Computer Aided Verification, 22nd International Conference, CAV 2010, Edinburgh, UK, July 15-19, 2010. Proceedings*, volume 6174 of *Lecture Notes in Computer Science*, pages 562–565. Springer, 2010.

[6] F. Houben, G. Igna, and Frits Vaandrager. Modeling task systems using parameterized partial orders, May 2011. Submitted.

[7] Bart Kienhuis, Ed Deprettere, Kees Vissers, and Pieter van der Wolf. An approach for quantitative analysis of application-specific dataflow architectures. In *ASAP*, pages 338–, Washington, DC, USA, 1997. IEEE Computer Society.

[8] Kim Guldstrand Larsen, Shuhao Li, Brian Nielsen, and Saulius Pusinskas. Scenario-based analysis and synthesis of real-time systems using uppaal. In *Design, Automation and Test in Europe, DATE 2010, Dresden, Germany, March 8-12, 2010*, pages 447–452. IEEE, 2010.

[9] Yuri Yushstein, Marco Bozzano, Alessandro Cimatti, Joost-Pieter Katoen, Viet Yen Nguyen, Thomas Noll, Xavier Olive, and Marco Roveri. System-software co-engineering: Dependability and safety perspective. In *4th IEEE International Conference on Space Mission Challenges in Information Technology (SMC-IT)*. IEEE CS Press, 2011.