Model-Driven Development of Embedded Real-Time Systems

by Alexandre David and Brian Nielsen

Model-driven development (MDD) has an enormous potential for improving verification, testing and synthesis of embedded systems. Our UPPAAL tool-suite for MDD of embedded real-time systems has recently been extended with components for automatic test generation and code synthesis. Presentation and demonstration at a European industrial conference on Systematic Testing spawned a lot of interest in these new techniques.

Our research centre was recently invited to participate in an industrial gathering on 'Systematic Testing' of embedded systems in Berlin, together with approximately eighty industrial participants from various sectors all over Europe ranging from the automotive industry, avionics, control systems and consumer electronics. It was observed that although these sectors vary in their technical intricacies and level of safety and reliability requirements, many common challenges exist. These include the ever-increasing size and complexity of software, demands for reduced time to market, and rapid changes in technology. At the same time we observed a change in attitude reflecting a decreased tolerance for errors, even towards zero-defects. This is also true of sectors beyond conventional safety-critical software. It is no longer accepted that systems will 'by nature' contain a certain number of errors and that the purpose of testing is to eliminate the worst of them. Quality systems are not allowed to misbehave, and certainly not in any way that users would notice. In combination, these developments make conventional quality assurance and testing techniques ineffective and too expensive.

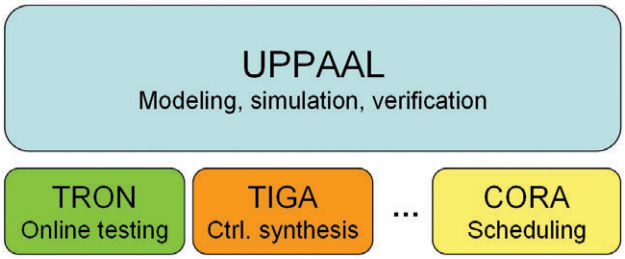

Figure 1: UPPAAL tool suite.

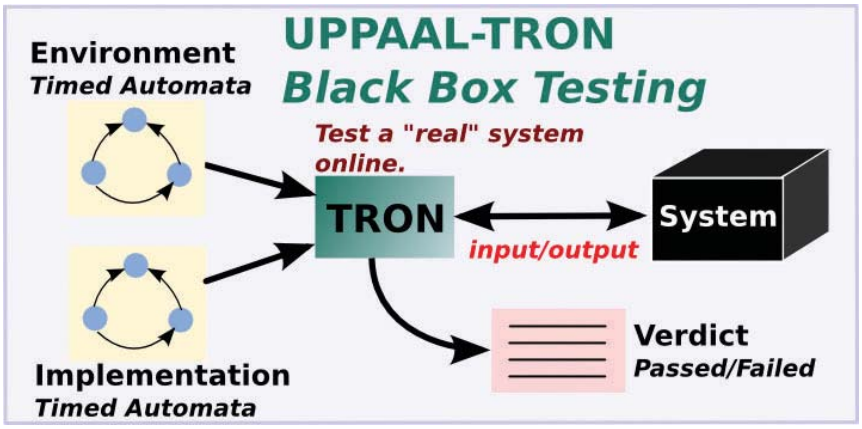

Figure 2: UPPAAL-TRON component for online testing of real-time systems.

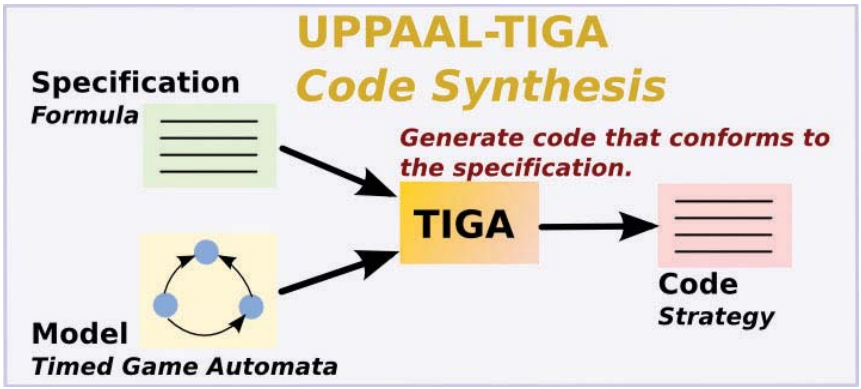

Figure 3: UPPAAL-TIGA component for controller synthesis based on timed games.

MDD is a promising approach that will contribute significantly to solving these challenges by enabling early model analysis (via simulation and model checking, for example) and design space exploration. Furthermore, model-based testing allows the test engineer to focus on the intellectual challenge of specifying and modelling the behaviour of interest at a high level of abstraction, rather than on manual test-case design, laborious scripting and manual test execution. Using model-based testing, a test generation tool generates an appropriate set of test cases and lets a test automatically execute these.

In the academic world, UPPAAL is a well-known and widely used model-checking tool for real-time systems. It is jointly developed by Aalborg University, Denmark, and Uppsala University, Sweden. The behaviour of timed systems is graphically modelled using the timed automata formalism extended with various modelling features such as concurrency and C-like functions and data structures, to make it practically expressive and user-friendly. UPPAAL contains a graphical editor and animator/simulator, and an efficient model-checker. The latter performs an exhaustive symbolic analysis of the model and provides either a proof that the model satisfies a property, or a counter-example consisting of a trace of actions and

delays exemplifying how the property is violated. It has been applied successfully to numerous industrial cases.

UPPAAL was recently extended with components for test generation and controller synthesis. This is an important step towards the vision of being an integrated tool suite for MDD of embedded systems, where the entire development lifecycle – from specification to running code – is driven by successive refinements and iterations.

UPPAAL can now handle both online and offline test generation. Specifically, UPPAAL-TRON is an online tool for black-box conformance testing of real-time systems. In an online tool, test events are generated and executed simultaneously in real time, on the physical implementation being tested. A further verdict on the observed interaction sequences may also be made online. TRON has well-defined formal semantic and correctness criteria defining how a correct implementation should behave compared to the modelled behaviour. The tool implementation extends the efficient algorithms and data structures of the model-checker engine to enable real-time testing of many real-time systems. TRON is connected to the system under test via an adaptor component (defined by the user), translating physical I/O or signalling to the abstract events of the model. TRON has for example been applied to test an embedded refrigeration controller for industrial cooling plants.

UPPAAL-TIGA is an extension of UPPAAL used to solve two-player timed games. Its main application is for controller synthesis. In a timed game automata model, actions are partitioned as being either controllable or uncontrollable. The idea is to have a controller playing with controllable actions and an environment playing with uncontrollable actions. The model-checker tries to compute a so-called winning strategy that tells the controller which actions to take (and when) to ensure that a property holds, no matter what the environment does. The tool implements a state-of-the-art algorithm that computes such strategies on the fly, ie while exploring states. The tool, in combination with Matlab-Simulink, has been successfully used to generate executable code of a climate controller for pig stables. Recently it was also applied to generate an optimal controller for a hydraulic pump.

However, despite the relative maturity of these techniques more work is needed to integrate with industrial tool-chains like Matlab-Simulink or UML-based tools, to further scale the techniques and to deal with other quantitative constraints beyond real time.

Link:
http://www.uppaal.com/

Please contact:
Brian Nielsen
Centre of Embedded Software Systems, CISS
Aalborg University, Denmark
Tel: +45 9940 8883
E-mail: bnielsen@cs.aau.dk

Quasimodo

by Brian Nielsen

Existing Model-Driven Development (MDD) tools and methods for real-time embedded systems are rather poor at handling the relevant quantitative constraints. Quasimodo (Quantitative System Properties in Model-Driven Design of Embedded Systems) is a new EU FP7 project whose main goal is to extend current MDD techniques and tools for modelling, verification, testing and code generation with the ability to satisfy these quantitative constraints.

A key characteristic of embedded systems is that they must meet a multitude of quantitative constraints. These involve the resources that a system may use (computation resources, power consumption, memory usage, communication bandwidth, costs etc), assumptions about the environment in which it operates (arrival rates, hybrid behaviour), and requirements on the services that the system must provide (timing constraints, quality of service, availability, fault tolerance etc). Existing MDD tools for real-time embedded systems are quite sophisticated in their handling of functional requirements, but their treatment of quantitative constraints is still very limited. Hence MDD will not realize its full potential in the embedded systems area unless the ability to handle quantitative properties is drastically improved.

To focus on aspects central to embedded systems such as performance, timeliness and efficient usage of resources, the models must provide quantitative information on timing, cost, data, stochastics and hybrid phenomena. A challenge is to develop notations that are expressive, have precise formal semantics, and that can be analysed efficiently by automated tools. Quasimodo mostly works with probabilistic, priced, timed game automata and looks at how to link these to industrial tool chains.

The analysis methods developed in Quasimodo include data structures for symbolic exploration of the behaviour of models, abstraction and compositionality principles that relate design models and help to control the size and complexity of the models, exploitation of approximate analysis techniques for partial analysis of very complex models and, orthogonally, optimal utilization of the given computing platform on which the algorithms are implemented.

In the implementation step, executable code running on given physical devices must be provided. While the theoretical framework of the quantitative models assumes infinitely fast hardware, infinitely precise clocks, unbounded memory and so on, real CPUs are subject to hard limitations in terms of frequency and memory size. Being able to guarantee that properties established by a

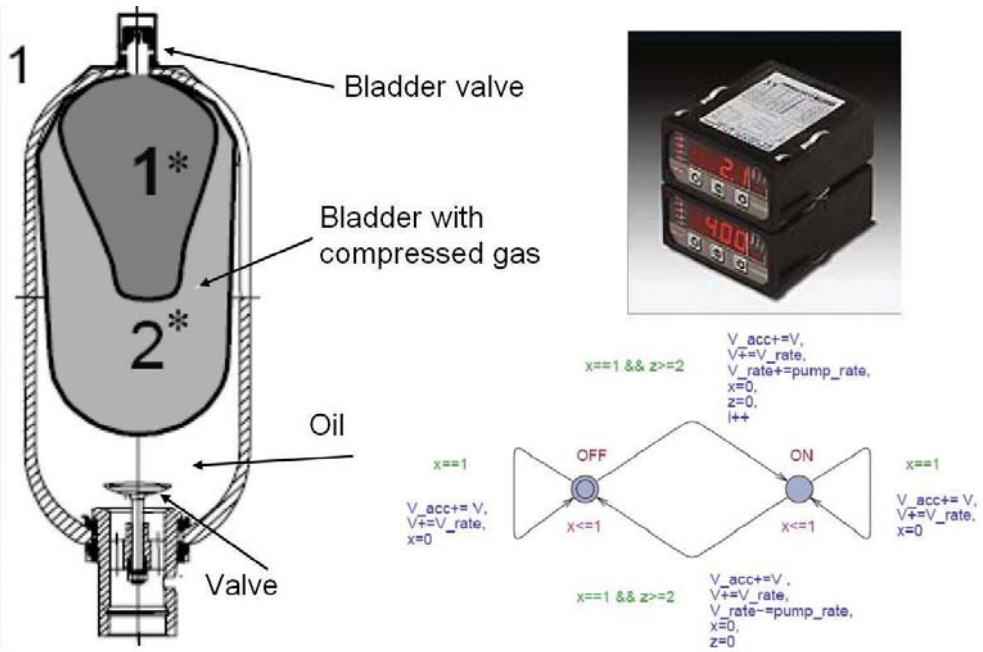

Figure 1: Accumulator bladder for a hydraulic pump (left), its electronic controller (top right), and part of the model of the control algorithm of accumulator control system (bottom right).

given model are also valid in its implementation is therefore a major challenge.

Current industrial testing is often manual and without effective automation, and consequently is rather error prone and costly: it is estimated that 30-70% of the total development cost is related to testing. Model-based testing is a novel approach to testing with significant potential to improve both cost and efficiency. We intend to extend the model-based testing technology to the setting of quantitative models, allowing generation, selection, execution and provision of coverage measures to be made. Another challenge, similar to implementation synthesis, is to develop a sound and theoretically well-founded framework and tools for testing quantitative systems when these quantities (eg timing) cannot be observed and controlled accurately.

In order to demonstrate the usefulness of our techniques, we will apply them to several complex industrial case studies and provide a collection of unique tool components to be used as plug-ins in industrial tools or tool chains like Matlab/Simulink.

One case study concerns a controller for an oil accumulator system for a hydraulic pump. The controller must maintain the gas pressure in the bladder within a safe pressure range, and should be robust against fluctuations in the energy consumption of the machine. The goal is then to find an optimal controller that minimizes the (long-term) energy consumption. Our approach consists of modelling the accumulator system as timed game automata in our game-solving tool Uppaal Tiga, and using this to automatically synthesize the optimal controller. We have produced a controller that has a 40% gain compared to classical controllers.

A second case concerns the modelling and analysis of medium-level access protocols for a specific type of sensor network. The interesting properties of the network will be modelled as probabilistic (timed) automata and linearly priced timed automata, and our model checking techniques and tools will be used to determine important system properties like collision-freeness, transmission rate and energy consumption.

Quasimodo will also model, analyse and test the application software for the Attitude Control Computer for the Hershel/Planck satellites. In particular we will evaluate the techniques for model-based online real-time testing. This work will include producing timed automata models of central aspects of both the control software and its environment, to ensure that only feasible and realistic tests are generated. Moreover, to facilitate automatic execution, test adaptation software will need to be developed to allow successful translation between abstract-model events and concrete messages to be sent/received from the system under test.

Partners in the project are Aalborg University, Denmark (coordinator); Embedded Systems Institute, the Netherlands; RWTH Aachen University, Germany; Universität des Saarlandes, Germany; Université Libre de Bruxelles, Belgium; ENS-Cachan/CNRS, France; Terma A/S, Denmark; Hydac GmbH, Germany; and Chess Beheer B.V, the Netherlands. The total budget is €2 696 000 and the project, which started in January 2008, will have a duration of 36 months.

Link:
http://www.quasimodo.aau.dk

Please contact:
Brian Nielsen
Centre of Embedded Software Systems, CISS
Aalborg University, Denmark
Tel: +45 9940 8883
E-mail: bnielsen@cs.aau.dk