Hierarchical Design and Analysis of Timed Systems

Wang Yi
Uppsala University

1 Summary

This document is an application for two Ph.D. projects: (1) *An Object-Oriented Language for Hierarchical Modelling and Analysis* and (2) *Test Generation: From Requirement Specifications to Executable Tests*. Research activities will be conducted within the joint effort on *tool-based system development environment* for real-time embedded systems, between Uppsala University, and Mälardalen University with Mecel AB, ABB Industrial systems AB and Volvo Technological Development Corporation being industrial partners. The goal is to further develop the UPPAAL tool to a *coherent tool environment* for the development of embedded real-time systems, which supports each step of the system development process, from *specification, design, simulation, verification, runnable code generation, to test generation.*

2 Ph.D. Projects

Languages for Hierarchical Design and Analysis

Currently, the tool UPPAAL provides no support for modelling hierarchical structures, not to mention hierarchical analysis. The user can describe only a network of automata in the very flat form, which restricts the size of problems that UPPAAL can deal with. Recently we started a joint case-study project with ABB Industrial Products to model and verify a bus protocol (AF 100) developed by ABB Industrial Systems AB, which is a commercial product. It is our experience with model-checkers like UPPAAL that a large part of the total time in most of the case studies is spent on constructing the system model, i.e., modeling. This has lead to a strong demand for a modelling language as well as a graphical user interface, with features supporting abstractions, such as hierarchical descriptions of models. There have been several modelling languages e.g. State-Chart for Statemate, developed particularly for modelling and simulation. However, rather little work has been done on static analysis for such languages to provide static information, e.g. types, hierarchical structures etc to guide verification.

We plan to develop an object–oriented modelling language as well as a graphical user interface for the tool UPPAAL. It supports two types of hierarchical structures:

- the system hierarchy where a system may be a collection of sub–systems (nested networks).
- the state hierarchy where a state may be a collection of states (nested automata).

We should mention that this project is naturally related to the existing *code-generation* project supported by ARTES as both address language design issues and more importantly the code-generation project is to generate runnable codes from design specifications.

The future plan is to develop verification techniques that utilize the hierarchical information in order to guide the state–space exploration algorithm that the UPPAAL engine is based on and to introduce *abstraction* in the UPPAAL simulator and diagnostic trace–generator.

Test Generation

Testing real-time properties for embedded systems is an area where few general techniques are available. In industry, most of the techniques are ad hoc and product–oriented. There have been work on testing sequential pograms, addressed in the area of software engineering and conformance testing in the area of telecommunication, but little work on real time testing.

In addition to the difficulties in testing sequential programs such as input coverage, testing embedded real time systems must also deal with following aspects:

1. **Environment Simulation**: due to the fact that many applications are safety critical, the testing machinery for embedded systems can not be put in the target environment. Testing must be carried with a simulated model of the physical environment, which simulates the reasonable inputs, and in addition can be directed to play critical scenarios. The creation of such simulation environments is non-trivial when the environment is a complicated physical process. A related problem is to supply representative inputs.

2. **Real–Time Behaviours**: to test real-time properties, test cases must cover not only all the sequences of significant events, but also significant time points. It is a non–trivial task to cover all time points in particular working in a dense time model. One needs to find all the critical time points to represent time zones instead of points as in the case of real time model checking.

3. **Non-determinism**: for a given input, the behavior of the system may still not be predictable, especially for event-driven systems. In order to be able to replay a test execution, the testing machinery may record the sequence of significant events, sufficient to regenerate the execution.

A survey on problems and possible solutions on testing real time systems can be found in

[Sch95] *W. Schüutz. Testing distributed real-time systems: An overview. In Randel l, Laprie, Kopetz, and Littlewood, editors, Predictably Dependable Computing Systems, pages 283-298. Springer Verlag, 1995.*

[RNHW98] *P. Raymond, X. Nicollin, N. Halbwachs, and D. Weber. Automatic testing of reactive systems. In RTSS98, 1998.*

In this project, we will concentrate on *test generation*. It will be a joint project with Volvo technological development Corporation which is involved in testing control systems for cars. The company (Volvo) has built up a testing environment simulating physical components of cars, such as motor etc. In particular, we will study the following issues:

1. **Test Description Language**: To describe abstract tests which may be transformed to executable test–sequences to run together with the software/hardware components to be tested in the testing environment at Volvo.

2. **Testing Requirment Specificaitons**: To generate abstract tests (and then to executable test–sequences) from logical specifications (requirement specifications) to justify that the target implementation satisfies the original requirements.

3 Work Plan

Research Environment

The following ARTES nodes will participate in the project:

1. Uppsala University

2. Mälardalen University

3. Mecel AB in Göteborg (Mikeal Strömberg, Tel: 031 703 3200) and

4. Volvo Technological Development Corporation (Ola Lundqvist, Tel: 031 772 4407)

Complementary Activities

There have been several related activities in Uppsala and Mälardalen within the proposed project, notably the following projects:

1. Code-Generation: From High-Level Design Model to Executable Code (at Mälardalen, supported by ARTES)

2. The UPPAAL Model Checker (i.e. the UPPAAL project) (at Uppsala, supported by NUTEK)

Work Plan and Budget

We apply for funding to support 2 Ph.D. students (the Budget should be calculated according to the ARTES standard) at Uppsala University to work on (as thesis projects):

1. An Object-Oriented Language for Hierarchical Design and Analysis. Currently, we have an excellent student (Alexander David) working on the project. A prototype language (named UL^{++}) is under development. During the next academic year, we plan to study a field bus protocol AF100 (a commercial product) developed by ABB to obtain feedback on the design of UL^{++}. In two years from now, a licentiate thesis based on the development of UL^{++} should be finished.

2. Test-Generation: From Requirement Specification to Executable Tests. This project is planned to start in the beginning of 1999. A visiting researcher working on real time testing from Twente University, Holland is currently visiting our group. A joint work on timed testing is in progress, which is to compile *determinizable* models in UPPAAL to tests. We plan to have a new Ph.D. student to join the project in the beginning of 1999 in collaboration with Volvo.

Industrial Cooperation

The success of the UPPAAL project have generated a number of industrial interests. However, we will conduct the project mainly in collaboration with ABB Automation Products (on the verification of AF100 protocol), Mecel AB and Volvo Technological Development Corporation.

1. ABB Industrial Products AB (Ulf Hammar, Tel: 021 34 3059)
2. Mecel AB in Göteborg (Mikeal Strömberg, Tel: 031 703 3200) and
3. Volvo Technological Development Corporation (Ola Lundqvist, Tel: 031 772 4407)

References

[1] Johan Bengtsson, David Griffioen, Kåre Kristoffersen, Kim G. Larsen, Fredrik Larsson, Paul Pettersson, and Wang Yi. Verification of an Audio Protocol with Bus Collision Using UPPAAL. In Rajeev Alur and Thomas A. Henzinger, editors, *Proc. of 9*th *Int. Conf. on Computer Aided Verification*, number 1102 in Lecture Notes in Computer Science, pages 244–256. Springer–Verlag, July 1996.

[2] E.M. Clarke and J.M. Wing. Formal methods: State of the art and future directions. *Computing Surveys*, 28(4):626–643, Dec. 1996.

[3] Klaus Havelund, Arne Skou, Kim G. Larsen, and Kristian Lund. Formal Modeling and Analysis of an Audio/Video Protocol: An Industrial Case Study Using UPPAAL. In *Proc. of the 18*th *IEEE Real-Time Systems Symposium*, December 1997.

[4] Kim G. Larsen, Paul Pettersson, and Wang Yi. UPPAAL in a Nutshell. *International Journal on Software Tools for Technology Transfer*, 1997.

[5] Magnus Lindahl, Paul Pettersson, and Wang Yi. Formal Design and Analysis of a Gear-Box Controller. In *Proc. of the 4*th *Workshop on Tools and Algorithms for the Construction and Analysis of Systems*, Lecture Notes in Computer Science, March 1998.