

# Verification of Hard and Softly Timed Systems (HaaST) Final Report

ED BRINKSMA, JOOST-PIETER KATOEN AND FRITS W. VAANDRAGER

May 14, 2004

## 1 Overview of Scientific and Technological Results

### 1.1 Introduction

The HaaST project (Verification of Hard and Softly Timed Systems) has been carried out in the period from 1 September 1999 until 1 April 2004 as project TES4999 of the Dutch Programme for Research on Embedded Systems (PROGRESS). The project was carried out by researchers of the Formal Methods and Tools group at the University of Twente, the Informatics for Technical Applications at the University of Nijmegen, and Philips Research Laboratories at Eindhoven. This document gives an overview of the scientific and technological results that were obtained by the project. For more information about the HaaST project we refer to the website <http://fmt.cs.utwente.nl/HaaST/>.

#### 1.1.1 Aims and objectives

The HaaST project aimed at the development and integration of methods and tools for the verification and analysis of real-time embedded systems, with an emphasis on distributed algorithms and protocols for consumer electronics applications. The goal was not only to consider “hard” real-time constraints – those that require that a system must react in time – but also so-called “soft” real-time constraints – those that require that the system should react in time but occasionally may not. Concretely, the project had the following objectives:

1. Extension of existing verification methods for timed systems with features for stochastic properties.
2. Development of a verification tool environment for the analysis of timed and stochastic systems.
3. Application of state-of-the-art methods and tools to case studies of representative complexity.

#### 1.1.2 Context

The HaaST project can be seen as the continuation of two preceding projects, viz. the NWO project “Testing and Verification of Timed Systems” carried out by the same groups at Twente and Nijmegen, and the collaboration project “Specification, Testing, and Verification of Software for Technical Applications” of CWI, Nijmegen, Twente and Philips Research.

The HaaST project has been carried out in the context of many initiatives and activities that mark the strong and still growing interest of computer science researchers worldwide in the problems related to embedded software systems. A characteristic element is the strong interrelation between the conceptual and formal aspects of their design, and the technical and physical problems involved.

Through the participating teams the HaaST projects was firmly embedded in a number of related international research activities, such as the European IST projects VHS (Verification of Hybrid Systems; terminated) and AMETIST (Advanced Methods for Timed Systems), and the Dutch-German NWO-DFG collaborative network VOSS (Validation of Stochastic Systems). Many of the results reported here have been produced in interaction with one or more of these projects.

## 1.2 Theory and algorithms

### 1.2.1 Specification of stochastic systems

The development of a compositional modeling formalism for describing soft real-time behavior has been an important activity within HaaST. The marriage of process algebra with stochastic timing requires careful re-examination of the interpretation of some classical process algebraic constructs, in particular that of choice, synchronization and concurrent composition [40]. The standard interleaving interpretation of concurrent composition in process algebra matches well with the memorylessness of exponential distributions in continuous-time Markov chains (CTMCs). This becomes more problematic when more general distributions are allowed. A solution has been developed to treat stochastic delays here in a way similar to clocks in timed automata, with separate operators to set and test for their expiration [43, 44]. Synchronization, in the setting of stochastically delayed actions, poses the question of what is the (distribution of the) delay of the synchronizations between them. An elegant solution here is adopt interleaving for synchronizations [12, 28] which induces a delay of synchronized actions with a distribution of the maximum of the delay preceding the synchronization. A more complete overview of Markovian stochastic process algebras can be found in [40, 5]; an account of the non-Markovian case is given in [49].

One of the main achievements of the HaaST project is the design of MoDeST (MOdeling and DEscription language for Stochastic Timed systems), a formalism that is aimed to support (i) the modular description of reactive system's behavior while covering both (ii) functional and (iii) non-functional system aspects such as timing and quality-of-service constraints in a single specification. The language contains features such as simple and structured data types, structuring mechanisms like parallel composition and abstraction, means to control the granularity of transitions, and non-deterministic and random branching and timing [22, 45]. MoDeST may be viewed as an overarching notation for a wide spectrum of models, ranging from labeled transition systems, to timed automata (and probabilistic variants thereof) as well as prominent stochastic processes such as Markov chains and (continuous-time) Markov decision processes.

The “core” of the language is strongly based on stochastic process algebras for generalized semi-Markov decision processes [43, 44], whereas for the hard real-time constraints, the model of timed automata has been adopted. For dealing with discrete probabilistic branching,

Segala’s simple probabilistic automata are used. A careful selection has been made of the operators in the language such that the elegant solutions for dealing with, choice, concurrent composition, and synchronization of stochastic process algebras remain while allowing for a high expressive power.

Further relevant HaaST publications wrt. to this topic are: [23, 35, 42].

### 1.2.2 Verification of stochastic timed systems

Once a stochastic model has been described in the compositional formalisms mentioned above, the next step is to evaluate the measure(s) of interest such as time to failure, system throughput or utilization. To that purpose we adopt model-checking techniques. With stochastic model checking the aim is to automatically verify whether, for instance, “the chance of shut-down occurring is at most 0.01%”. This is opposed to the traditional verification question addressing pure functional behavior such as “system failures are impossible”. Stochastic model checking is based on conventional model checking, but also entails the calculation of the actual likelihoods through appropriate numerical methods.

Within the HaaST project we have worked on model-checking of Markov models, in particular, continuous-time Markov chains (CTMCs). These are an important class of stochastic processes that have been widely used in practice to determine system performance and dependability characteristics. The logic CSL (Continuous Stochastic Logic) has been developed that allows – besides the standard steady-state and transient measures – for the specification of (constraints over) probabilistic measures over paths through CTMCs. Equivalence of CSL-formulas is shown to coincide with lumping, a key concept for the aggregation of Markov chains [14] and a simulation pre-order has been developed for CTMCs that allows for the reduction of CTMC prior to model checking [15].

CSL has been adapted to stochastic process algebra [30], has been extended with rewards [16, 25], and it has been shown that checking steady-state properties reduces to solving a system of linear equations combined with standard graph analysis methods, while checking time-bounded until formulas requires the solution of a (recursive) Volterra integral equation system. The latter problem is shown, after an appropriate transformation of the CTMC, to be reducible to a transient analysis problem that can be efficiently solved [14, 31]. An overview of all these techniques can be found in [1].

A prototypical tool has been developed, in co-operation with the University of Erlangen-Nürnberg, that supports the verification of CTMCs [29, 4, 48]. This tool is publicly available<sup>1</sup> and has been applied to several case studies such as the dependability analysis of workstation clusters [26]. With the currently available technology, models of  $10^6$  –  $10^7$  states can be successfully checked [3,31].

An overview of the linking between stochastic process algebra specifications and model-checking of CTMCs has been described in [28].

---

<sup>1</sup>[www7.informatik.uni-erlangen.de/etmcc](http://www7.informatik.uni-erlangen.de/etmcc)

### 1.2.3 Verification of probabilistic systems

We worked on the theory of probabilistic (I/O) automata as initiated by Segala. Our theoretical results were applied in the verification of some probabilistic algorithms, notably the IEEE 1394 root contention protocol. Many of our results in this area ended up in the PhD thesis of Marielle Stoelinga. Two important theoretical results are:

- A paper on a testing scenario for probabilistic automata was presented at ICALP'03, where it received the EATCS prize for best track B paper [36].
- Together with Nancy Lynch and Roberto Segala, we established that on the domain of probabilistic automata, the trace distribution pre-order coincides with the simulation pre-order, thus solving a problem that had been open for quite a number of years [33].

### 1.2.4 Model checking hard real-time systems

The activities in this area have been focused on solving (hard) real-time verification and scheduling problems using model-checking techniques. Model checking is emerging as a practical tool for automated debugging of complex reactive systems such as embedded controllers and network protocols. In model checking, specifications about the system are expressed as (temporal) logic formulas, and efficient symbolic algorithms are used to traverse the model defined by the system and check if the specification holds or not. Extremely large state-spaces can often be traversed in minutes. Within HaaST, we studied two different approaches to model checking of timed systems.

In the first approach, see [2, 8,21], we showed how a hybrid control schedule can be derived, verified and optimized using the explicit (finite) state model checker Spin. The first important contribution is the use of a very lean model for real-time by "variable time advance", a technique borrowed from discrete event simulation that only adds "interesting" points in time to model, thus significantly reducing the state space. The second idea concerns the controlled introduction of non-determinism in the model that reduces the branching degree in the search trees and allows for a "guided" search of the state space, visiting the more promising parts first.

In the second approach, systems are described as networks of timed automata in the sense of Alur and Dill. During the last decade, there has been an immense progress in the area of timed model checking, and tools such as Uppaal and Kronos are now routinely used for industrial case studies. We closely collaborated with the builders of Uppaal to add new functionality and algorithms to boost performance of the tool, and on applying the tool to challenging problems from industry. More specifically, we proposed four extensions of Uppaal:

- Distributed model checking: we demonstrated that the combined processing and memory resources of multi-processor computers can be effectively utilized [18].
- Parametric model checking: we developed algorithms that can synthesize linear parameter constraints for the correctness of timed systems [6].
- Guided model checking: here the aim is to guide the search to those parts of the state space which are most interesting, using branch-and-bound techniques and by associating

costs to both states and transitions [32, 17]. Using these ideas, we were able to solve realistic scheduling problems in a competitive manner and drastically reduce the time to find error states in several protocol models. Many of our results in this area ended up in the PhD thesis of Ansgar Fehnker.

- An enhancement of Uppaal with symmetry reduction using the scalarset datatype known from Murphi. For all examples that we experimented with (both academic toy examples and industrial cases), we obtained a drastic reduction of both computation time and memory usage, exponential in the size of the scalar sets used. Also, we discovered that the Bang & Olufsen audio/video protocol that was previously proven to be correct for two processes actually exhibits faulty behavior for three processes [27].

### 1.2.5 Hybrid systems

Hybrid systems are systems that intermix discrete and continuous components, typically computers that interact with the physical world. Due to the rapid development of processor and circuit technology, such systems are becoming more and more common in all application domains, ranging from avionics and process control to robotics and consumer electronics. The specification, design and analysis of hybrid systems require a synthesis of ideas, concepts, mathematical theories and tools that are currently spread over distinct disciplines, most notably Computer Science and Control Theory.

In close collaboration with MIT (Nancy Lynch), and the University of Verona (Roberto Segala), we further developed the hybrid I/O automata (HIOA) framework [34, 7]. This framework has been used, amongst others, to analyze examples of automated transportation systems, intelligent vehicle highway systems, air traffic control systems, automotive control systems, consumer electronics applications, and a small Lego car [24]. At RTSS'03 we presented a paper [37] in which we show how the special case of the HIOA framework in which systems synchronize via shared actions only is expressive enough to describe complex timing behavior, and to express the important ideas of previous timed automata frameworks. This restricted HIOA framework is e.g. a most natural setting to model and analyse networks of communicating processes with unreliable clocks.

## 1.3 Tools

### 1.3.1 MOTOR

In order to facilitate the analysis and evaluation of MoDeST specifications, the prototype tool MOTOR (MoDeST Tool EnviRonment) has been developed. The main purpose of MOTOR is to interpret or translate MoDeST specifications in such a way that other, external tools can be used to do the actual analysis of the MoDeST specification. This approach has the advantage to provide access to state-of-the art analysis engines with moderate effort. MOTOR is written in the C++ programming language. The reason for this choice was the good trade-off between speed of the implementation, modularity of the code, and easy extendability. The tool architecture has been discussed in [19].

The tool contains the basic functionality to handle MoDeST specifications: a parser and an implementation of the Structural Operational Semantics (SOS). A state-space generator has

been implemented that allows to generate state-based representations of MoDeST specifications in different formats. Most notably among these formats is the bcg format, which allows to interface to CADP, a model checker tool suite. The implementation of the SOS and the state-space explorer has been developed in the better part of 2002.

The next and, for HaaST, last addition to MoDeST is the integration of MOTOR in Möbius, a powerful performance evaluation tool suite developed at the University of Illinois at Urbana-Champaign. Möbius provides a state-of-the-art discrete-event simulator, which is utilized to simulate MoDeST specifications. The effort to integrate MoDeST/MOTOR into Möbius started with the begin of 2003 with a three-week visit at the University of Illinois at Urbana-Champaign, and continued from there through most of 2003. In spring 2003, a student has been sent to Urbana-Champaign in order to enhance the MoDeST language with structured data-types, and to make the necessary changes in Möbius in order to facilitate the new features of MoDeST [46].

Up til now, the combination of MOTOR/Möbius has been used for two case-studies now. The joint efforts of the MoDeST/Möbius integration has been reported in [39].

### 1.3.2 Extensions to Uppaal

As discussed already in Section 1.2.4, we studied model checking of networks of timed automata as a means to verify hard real-time properties. Our ideas were successfully implemented in prototype extensions of the widely used model checker Uppaal:

- Distributed model checking: [18].
- Parametric model checking: [6].
- Guided model checking: [32, 17].
- Symmetry reduction: [27].

It is expected that distributed model checking, guided model checking and symmetry reduction will all be implemented in the new publically available version v4.0 of Uppaal that will (hopefully) become available by the end of 2004.

## 1.4 Applications

This section gives an account of the case studies that were performed during the HaaST project in co-operation with Philips.<sup>2</sup>

**Analysis and optimisation of the IPv4 zeroconf protocol.** This case study was initiated by the home networking group of Philips Research. The study is about the IPv4 Zeroconf protocol, proposed by Cheshire/Adoba/Guttman in 2002, dedicated to the self-configuration of IP network interfaces. The task was to investigate the trade-off between reliability and effectiveness of this protocol. The problem was tackled from different directions:

---

<sup>2</sup>Further applications can be found at `/www.cs.kun.nl/ita/research/ > projects > analysis of distributed algorithms and network protocols`.

- The group in Twente developed a simple stochastic cost model of the protocol, where reliability is measured in terms of the probability to avoid an address collision after configuration, while effectiveness is viewed as the average penalty perceived by a user. The solution method was optimisation of several protocol parameters on minimal cost. [38, 20]
- The group in Nijmegen developed an Uppaal model of the protocol in order to analyze the functional correctness and the real-time behavior of the protocol. [52]

**Network on chip.** We gave a formal specification of the Aethereal protocol and its underlying architecture for packet switching networks on chip (NoC). All elements of the network and their operations were specified using the theorem prover PVS. Based on this model, we gave a (partially formalized) proof of absence of deadlock [51].

Designing a NoC that meets all the requirements for best-effort and guaranteed traffic clearly is a difficult task, and the resulting design intrinsically has to be quite complex. During the design process, the designers play around with thousands of design alternatives. It is difficult to keep track of all the design options in a systematic way, and to make sure that the choices that have been made are not inconsistent.

Our contribution [51] is that for one of the numerous design alternatives we produced (although at an abstract level) a very precise, formal model. Within this model we were able to establish a key correctness property of the system, namely absence of deadlock. It is very unlikely that the design as we have formalized it in our paper will be the one that is going to be implemented in hardware. However, since our specification is highly abstract and very modular, we think it will be relatively easy to modify our spec to reflect variations of the design.

In fact, we believe that our work illustrates that formal specification languages such as supported by PVS can be most useful to document complex designs, to force designers to clarify design choices, and to resolve problematic inconsistencies in an early stage of the design process.

**UPnP liveness protocol.** UPnP (Universal Plug and Play) is an industry standardization initiative undertaken by the UPnP Forum to “enable simple and robust connectivity among stand-alone devices and PCs from different vendors”. Typical environments for UPnP technology are homes and offices where devices communicate with wireless communication. In these networks topology changes happen frequently and UPnP should contain facilities to efficiently and robustly deal with this. To that end, Philips has proposed a liveness protocol for the quick detection which devices are still on the network, and which are not. The protocol should, however, not overload the devices by flooding them with messages. Several stochastic phenomena play an important role for the liveness protocol, such as message loss, random delays, and random behavior of devices joining and leaving the network.

Since the summer of 2003 we are using MoDeST for the formal description and analysis of the liveness protocol. Although this work has not yet been completed, it is interesting to report that some undesired phenomena in the liveness protocol have been discovered by discrete-event simulation of the MoDeST specification (using the Motor-Möbius interface). In particular, it has been shown that under certain circumstances, the system may reach an undesired equilibrium where certain control points check devices in a fast way whereas

others have a rather low frequency of checking, and the system is not able to recover from this situation [47].

## 1.5 Utilization

Verification of hard and softly timed systems is considered as a most important topic by the international research community, with great societal relevance, and many strong groups are working on it. It is evident that during the lifetime of the HaaST project enormous progress has been made in this area. With contributions from the HaaST project, the model checker Uppaal has advanced from an academic proof of concept to a tool that is now being used by thousands of researchers both in academia and in industry, and that will soon (say when version 4.0 is available) be ready for further industrial development. The MOTOR tool, which is a true product of the HaaST project, is still very much in the stage of an academic prototype, but its potential usefulness has been demonstrated already on some industrial case studies, and clearly further development of this tool will be most promising.

Altogether, the results of the HaaST project and the case studies that we carried out lead us to believe that in many cases it can be advantageous (and cost effective!) to perform formal modelling and analysis of (timing related or other) properties of embedded systems using our methods. The precise modeling of the system under consideration often already gives an important benefit, and, the combined assessment of both quantitative and qualitative system requirements using the same system model can be of great value.

The results of the HaaST project are being used in several European research projects on embedded systems, notably the IST-project *Advanced methods for timed systems (AMETIST)*, and the network of excellence ARTIST. Also at the national level, there are currently several projects that are strongly related to HaaST and that are (partially) based on its results: the NWO projects *Systematic testing of real-time embedded systems (STRESS)*, *Model checking infinite-state Markov chains (MC = MC)*, *Fault-tolerant real-time algorithms analyzed incrementally (FRAAI)* and the NWO-Vernieuwingsimpuls project *Verification of performance and dependability*. Finally, both research groups participate in the DFG-NWO bilateral programme *Validation of Stochastic Systems (VOSS)*.

## 1.6 Concluding remarks

**Conclusions.** The HaaST project had the following specific objectives:

1. Extension of existing verification methods for timed systems with features for stochastic properties.
2. Development of a verification tool environment for the analysis of timed and stochastic systems.
3. Application of state-of-the-art methods and tools to case studies of representative complexity.

These objectives have been realized: (1) scientifically, the project significantly contributed to advance the state-of-the-art for the analysis of stochastic and hard timed systems (Sections 1.2.1, 1.2.2 and 1.2.3), (2) as a technological contribution, we developed a prototype verification tool environment MOTOR for the analysis of timed and stochastic systems, and actively participated in the further development of the Uppaal tool for the analysis of hard real-time

systems (Section 1.3), (3) although the co-operation with Philips has not been optimal during the entire project (amongst others, due to re-organizations at Philips) three case studies have been conducted in close cooperation with Philips, two applications in the area of consumer electronics, and one on chip design. The results of these case studies have been considered valuable, both for Philips, as well as for the verification techniques and tools developed in the project (Section 1.4).

**The future.** The MOTOR tool is only a first prototype and initial case studies conducted with it indicate several shortcomings. In addition, there are several important open problems in the field of soft and hard real-time verification. To mention a few:

- Algorithms for the verification of stochastic aspects need to be further improved (efficiency), and effective means have to be developed to allow for abstraction such that systems of larger size can be automatically checked;
- To take aspects like power consumption and memory usage into account, reward extensions of stochastic models seem to be highly appropriate. Model checking of stochastic reward models has, however, received scant attention so far. Available techniques are inefficient, restricted to particular kinds of rewards, and suffer from computational problems (e.g., numerical instability). There is a significant impulse needed to turn this approach into a technique of practical relevance.
- Timeliness and interaction with the environment are main characteristics of embedded software. Stochastic models that faithfully represent embedded software thus need to be timed and open. Openness entails that the behavior of the environment is underspecified, and that interactions may be abstracted from. Nondeterminism is the technique par excellence for this purpose. Verification techniques for stochastic timed decision processes are, however, not yet available.

Given these points, the project partners will strive for a follow-up project of HaaST to generate a full academic prototype of the MOTOR tool and to bridge the gap towards design notations that are common practice in the design of embedded systems, such as the UML.

Also the further improvement and industrial exploitation of timed automata technology — and its implementation in the Uppaal toolset — is an obvious topic for future research.

## 2 HaaST publications

### 2.1 Journal papers

1. C. BAIER, B. HAVERKORT, H. HERMANN AND J.-P. KATOEN. *Model-checking algorithms for continuous-time Markov chains*. IEEE Transactions on Software Engineering, vol. **29**, no. 6, pages 524–541, 2003.
2. E. BRINKSMA, A. MADER AND A. FEHNER. *Verification and optimization of a PLC control schedule*. International Journal on Software Tools for Technology Transfer, vol. **4**(1), pages 21–3, 2002.
3. P. BUCHHOLZ, J.-P. KATOEN, P. KEMPER AND C. TEPPER. *Model-checking large structured Markov chains*. Journal of Logic and Algebraic Programming, vol. **56**, no. 1-2, pages 69–97, 2003.

4. H. HERMANNNS, J.-P. KATOEN, J. MEYER-KAYSER AND M. SIEGLE. *A tool for model checking Markov chains*. Int. Journal on Software Tools for Technology Transfer, vol. **4**, no. 2, pages 153–172, 2003.
5. H. HERMANNNS, U. HERZOG AND J.-P. KATOEN. *Process algebra for performance evaluation*. Theoretical Computer Science, vol. **274**, no. 1-2, pages 43–87, 2002.
6. T. HUNE, J.M.T. ROMIJN, M.I.A. STOELINGA AND F.W. VAANDRAGER. *Linear parametric model checking of timed automata*. Journal of Logic and Algebraic Programming, vol. **52-53**, pages 183–220, 2002.
7. N.A. LYNCH, R. SEGALA, AND F.W. VAANDRAGER. *Hybrid I/O automata*. Information and Computation, vol. **185**(1), pages 105–157, 2003.
8. A. MADER, E. BRINKSMA, H. WUPPER, AND N. BAUER. *Design of a PLC control program for a batch plant, VHS case study 1*. European Journal of Control, vol. **7**(4), pages 416–439, 2001.
9. J.G. SPRINGINTVELD, F.W. VAANDRAGER AND P.R. D’ARGENIO. *Testing timed automata*. Theoretical Computer Science, vol. **254**, Issue 1-2, pages 225-257, 2001.

## 2.2 Books and edited volumes

10. E. BRINKSMA AND K.G. LARSEN (EDITORS). *Computer Aided Verification, 14th International Conference (CAV 2002)* volume **2404** of Lecture Notes in Computer Science. Springer-Verlag, 2002.
11. E. BRINKSMA, H. HERMANNNS AND J.-P. KATOEN (EDITORS). *Lectures on Formal Methods and Performance Analysis*. volume **2090** in Lecture Notes in Computer Science (Tutorial). Springer-Verlag, 2001.
12. H. HERMANNNS. *Interactive Markov Chains: The Quest for Quantified Quality*. volume **2428** in Lecture Notes in Computer Science, 2002.
13. J.-P. KATOEN AND P. STEVENS (EDITORS). *Tools and Algorithms for the Construction and Analysis of Computer Systems*. volume **2280** in Lecture Notes in Computer Science. Springer-Verlag, 2002.

## 2.3 Conference contributions

14. C. BAIER, B. HAVERKORT, H. HERMANNNS AND J.-P. KATOEN. *Model checking continuous-time Markov chains by transient analysis*. In E.A. Emerson and A. Sistla, editors, Computer-Aided Verification (CAV), volume **1855** of Lecture Notes in Computer Science, pages 358–372, 2000.
15. C. BAIER, J.-P. KATOEN, H. HERMANNNS AND B. HAVERKORT. *Simulation for continuous-time Markov chains*. In L. Brim, P. Jancar, M. Kretinzki and A. Kucera, editors, Concurrency Theory (CONCUR), volume **2421** of Lecture Notes in Computer Science, pages 338–354, 2002.

16. C. BAIER, B. HAVERKORT, H. HERMANNNS AND J.-P. KATOEN. *On the logical characterisation of performability properties*. In U. Montanari, J.D.P. Rolim, E. Welzl, editors, Automata, Languages and Programming (ICALP), volume **1853** of Lecture Notes in Computer Science, pages 780–792, 2000.
17. G. BEHRMANN, A. FEHNER, T. HUNE, K.G. LARSEN, P. PETTERSSON, J.M.T. ROMIJN AND F. VAANDRAGER. *Minimum-cost reachability for priced timed automata*. In M.D. Di Benedetto and A.L. Sangiovanni-Vincentelli, editors, Proceedings HSCC'01, LNCS 2034, pages 147–164, 2001.
18. G. BEHRMANN, T. HUNE AND F.W. VAANDRAGER. *Distributed Timed Model Checking - How the Search Order Matters*. In E.A. Emerson and A.P. Sistla, editors. Proceedings Computer-Aided Verification (CAV), LNCS 1855, pages 216-231, 2000.
19. H. BOHNENKAMP, H. HERMANNNS, J.-P. KATOEN AND R. KLAREN. *The MoDeST modelling tool and its implementation*. In P. Kemper and W.H. Sanders, editors, Modelling Techniques and Tools for Computer Performance Evaluation (TOOLS), volume **2794** of Lecture Notes in Computer Science, pages 116–133, 2003.
20. H. BOHNENKAMP, P. VAN DER STOK, H. HERMANNNS, AND F. VAANDRAGER. *Cost-Optimisation of the IPv4 Zeroconf Protocol*. In Proceedings of the International Conference on Dependable Systems and Networks (DSN'03), IEEE CS Press, pages 531-540, 2003.
21. E. BRINKSMA AND A. MADER. *Verification and optimization of a PLC control schedule*. In K. Havelund, J. Penix, and W. Visser, editors, SPIN Model Checking and Software Verification, volume 1885 of Lecture Notes in Computer Science, pages 73–92, 2000.
22. P.R. D'ARGENIO, H. HERMANNNS, J.-P. KATOEN AND J. KLAREN. *MoDeST: A Modelling language for Stochastic Timed systems*. In L. de Alfaro and S. Gilmore, editors, Process Algebra and Probabilistic Methods, volume **2165** of Lecture Notes in Computer Science, pages 87–103, 2001.
23. P.R. D'ARGENIO, J.-P. KATOEN AND E. BRINKSMA. *Specification and analysis of soft real-time systems: quality and quantity*. IEEE Real-Time Systems Symposium (RTSS), pages 104–114, IEEE CS Press, 1999.
24. A. FEHNER, M. ZHANG AND F.W. VAANDRAGER. *Modeling and Verifying a Lego Car Using Hybrid I/O Automata*. In Third International Conference on Quality Software (QSIC 2003), IEEE CS Press, 2003 (to appear). Full version in M. Broy and M. Pizka, editors. Models, Algebras, and Logic of Engineering Software, Nato ASI Series III: Computer and Systems Sciences, Volume 191, pages 385-402, IOS Press, 2003.
25. B. HAVERKORT, L. CLOTH, H. HERMANNNS, J.-P. KATOEN AND C. BAIER. *Model-checking performability properties*. International Conference on Dependable Systems and Networks (DSN), pages 103–113, IEEE CS Press, 2002.
26. B. HAVERKORT, H. HERMANNNS, AND J.-P. KATOEN. *The use of model checking techniques for quantitative dependability evaluation*. IEEE Symposium on Reliable Distributed Systems (SRDS), IEEE Computer Society Press, pp. 228–238, 2000.

27. M. HENDRIKS, G. BEHRMANN, K.G. LARSEN, AND F. VAANDRAGER. *Adding Symmetry Reduction to Uppaal*. In P. Niebert and K.G. Larsen, editors, Proceedings First International Workshop on Formal Modeling and Analysis of Timed Systems (FORMATS 2003), volume **2791** of Lecture Notes in Computer Science, Springer Verlag, 2004. Full version available as Technical Report NIII-R0407, NIII, University of Nijmegen, February 2004.
28. H. HERMANNNS AND J.-P. KATOEN. *Performance analysis := (process algebra + model checking)  $\times$  Markov chains*. In K.G. Larsen and M. Nielsen, editors, Concurrency Theory (CONCUR), volume **2154** of Lecture Notes in Computer Science, pages 59–82. 2001 (invited tutorial).
29. H. HERMANNNS, J.-P. KATOEN, J. MEYER-KAYSER AND M. SIEGLE. *A Markov chain model checker*. In S. Graf and M. Schwartzbach, editors, Tools and Algorithms for the Construction and Analysis of Systems (TACAS), volume **1785** of Lecture Notes in Computer Science, pages 357–373, 2000.
30. H. HERMANNNS, J.-P. KATOEN, J. MEYER-KAYSER AND M. SIEGLE. *Towards model checking stochastic process algebra*. In W. Grieskamp, T. Santen and B. Stoddart, editors, Integrated Formal Methods (IFM), volume **1945** of Lecture Notes in Computer Science, pages 420–439, 2000.
31. J.-P. KATOEN, M.Z. KWIATKOWSKA, G. NORMAN AND D. PARKER. *Faster and symbolic CTMC model checking*. In L. de Alfaro and S. Gilmore, editors, Process Algebra and Probabilistic Methods, volume **2165** of Lecture Notes in Computer Science, pages 23–38, 2001.
32. K. G. LARSEN, G. BEHRMANN, H. BRINKSMA, A. FEHNER, P. PETERSSON, AND J. M. T. ROMIJN. *As cheap as possible: Efficient cost-optimal reachability for priced timed automata*. In G. Berry and P. Cousot, editors, *Computer Aided Verification (CAV)*, volume **2102** of Lecture Notes in Computer Science, pages 493–505, 2001.
33. N.A. LYNCH, R. SEGALA AND F.W. VAANDRAGER. *Compositionality for Probabilistic Automata*. In R. Amadio and D. Lugiez, editors. Proceedings 14th International Conference on Concurrency Theory (CONCUR 2003), LNCS 2761, pages 208–221. 2003.
34. N.A. LYNCH, R. SEGALA AND F.W. VAANDRAGER. *Hybrid I/O automata revisited*. In M.D. Di Benedetto and A.L. Sangiovanni-Vincentelli, editors. Proceedings HSCC'01, LNCS 2034, pages 403–417, 2001.
35. T. RUYS, R. LANGERAK, J.-P. KATOEN, D. LATELLA AND M. MASSINK. *First passage time analysis of stochastic process algebra using partial orders*. In T. Margaria and W. Yi, editors, Tools and Algorithms for the Construction and Analysis of Systems (TACAS), volume **2031** of Lecture Notes in Computer Science, pages 220–236, 2001.
36. M. STOELINGA AND F.W. VAANDRAGER. *A Testing Scenario for Probabilistic Automata*. In J.C.M. Baeten, J.K. Lenstra, J. Parrow and G.J. Woeginger, editors. Proceedings International Colloquium on Automata, Languages and Programming (ICALP), LNCS 2719, pages 407–418, 2003.

37. D.K. KAYNAR, N.A. LYNCH, R. SEGALA, AND F.W. VAANDRAGER. *A Framework for Modelling Timed Systems with Restricted Hybrid Automata*. In Proceedings 24th IEEE International Real-Time Systems Symposium (RTSS03), 2003. IEEE Computer Society, pages 166-177, 2003.

## 2.4 Other publications

38. H. BOHNENKAMP, H. HERMANNNS, M. ZHANG AND F. VAANDRAGER. *Cost-Optimisation of the IPv4 Zeroconf Protocol*. Proceedings of the 3rd PROGRESS Workshop on Embedded Systems, 2002.
39. H. BOHNENKAMP, T. COURTNEY, D. DALY, S. DERISAWI, H. HERMANNNS, J.-P. KATOEN, J. KLAREN, V. LAM AND W.H. SANDERS. *On integrating the Möbius and Modest modeling tools*. Proceedings International Conference on Dependable Systems and Networks (DSN), 2003 (tool demo paper).
40. E. BRINKSMA AND H. HERMANNNS. *Process algebra and Markov chains*. In H. Brinksma, H. Hermannns, and J. P. Katoen, editors, *Lectures on Formal Methods and Performance Analysis*, volume 2090 of Lecture Notes in Computer Science, pages 183–232, 2001.
41. E. BRINKSMA AND J.-P. KATOEN. *Hooggespannen verwachtingen model checking*. Bits & Chips, vol. **3**, no. 4, pages 31–33, 2001 (in Dutch).
42. P.R. D'ARGENIO. *A Compositional translation of stochastic automata into timed automata*. CTIT Technical Report 00-08, 2000.
43. P.R. D'ARGENIO AND J.-P. KATOEN. *A theory of stochastic systems, part I: Stochastic automata*. 2003, submitted.
44. P.R. D'ARGENIO AND J.-P. KATOEN. *A theory of stochastic systems, part II: Process algebra*. 2003, submitted.
45. P.R. D'ARGENIO, H. HERMANNNS, J.-P. KATOEN AND J. KLAREN. *Modelling stochastic systems (extended abstract)*. In F. Karelse, editor, 2nd Workshop on Embedded Systems pages 31–39, 2001.
46. J. GORTER. *Arrays and structures in Modest and Möbius*. Stage report at University of Illinois at Urbana-Champaign, 2003.
47. J. GORTER. *Modeling and analysis of the UPnP liveness protocol*. Master's thesis, Universiteit Twente, 2004.
48. H. HERMANNNS, J.-P. KATOEN, J. MEYER-KAYSER AND M. SIEGLE. *A model checker for performance and dependability properties (extended abstract)*. In F. Karelse, editor, 2nd Workshop on Embedded Systems, pages 83–89, 2001.
49. J.-P. KATOEN AND P.R. D'ARGENIO. *General distributions in process algebra*. In E. Brinksma, H. Hermannns, and J.-P. Katoen, editors, *Lectures on Formal Methods and Performance Analysis (FMPA)*, volume **2090** of Lecture Notes in Computer Science, pages 375–424, 2001.

50. J.-P. KATOEN. *Real-time and probabilistic systems – Foreword*. Theoretical Computer Science, vol. **282**, no. 1, pages 1–3, 2002.
51. B. GEBREMICHAEL, F.W. VAANDRAGER, AND M. ZHANG. *Formal Models of Guaranteed and Best-Effort Services for Networks on Chip*. 2004.
52. M. ZHANG AND F.W. VAANDRAGER. *Analysis of a Protocol for Dynamic Configuration of IPv4 Link Local Addresses using Uppaal*. Technical Report NIII-R04XX, NIII, University of Nijmegen, 2004.