MISCELLANEOUS CASE STUDIES — Final Report

UT, all CRs

May 29, 2006
revised after Final Review

AMETIST DELIVERABLE 3.5.3

Project acronym: AMETIST
Project full title: Advanced Methods for Timed Systems
Project no.: IST-2001-35304

1 Introduction

This is the Ametist Final deliverable of Task 3.5 on Miscellaneous Case Studies. This Task has been created to collect and publish internal case studies of the CRs pertinent to the Ametist project. Although most of the project's resources have been directed to case studies provided by the Ametist associate partners, reported under Tasks 3.1–3.4, a number of such internal case studies have been carried out. A substantial part of the reported work has been funded from other sources than the Ametist project, but all of the reported work is strongly related to Ametist in terms of application area or techniques and tools used, as well as the researchers involved.

Being final, this version of the deliverable includes all miscellaneous case studies carried out during the 3 years of the project. For each case study we indicate its relevance to the project, the modelling formalisms and analytical tools used, and present a short summary of the work.

2 Internal Case Studies

2.1 Verification and improvement of the sliding window protocol

Relevance The contribution shows the expressiveness of the timed automaton paradigm. It also provides evidence that modelling and analysis in terms of timed automata is not restricted to model checking and scheduling analysis, which are the main techniques used in AMETIST , but that they can also provide a successful starting point for complementary techniques involving theorem proving and/or proof checking.

Modelling Timed automata, PVS models.

Tools PVS.

Summary The well-known Sliding Window protocol caters for the reliable and efficient transmission of data over unreliable channels that can lose, reorder and duplicate messages. Despite the practical importance of the protocol and its high potential for errors, it has never been formally verified for the general setting. It is attempted to fill this gap by giving a fully formal specification and verification of an improved version of the protocol. The protocol is specified by a timed state machine in the language of the verification system PVS. This allows a mechanical check of the proof by the interactive proof checker of PVS. The proposed modelling is very general and includes such important features of the protocol as sending and receiving windows of arbitrary size, bounded sequence numbers and channels that may lose, reorder and duplicate messages.[6]

Limitations This verification method can only be performed by someone who is an expert in PVS. And even for a PVS expert, verification in a new domain requires a large time investment.

2.2 Modeling and verifying a Lego car using hybrid I/O automata

Relevance The contribution shown that a natural extension of the timed automaton model can be the basis for an integrated correctness analysis of a moderate sized hybrid system.

Modelling Hybrid I/O automata.

Tools none.

Summary The application of the hybrid I/O automata framework of Lynch, Segala & Vaandrager is illustrated by using it to model and analyze the behavior of a simple Lego car with caterpillar traction. Constraints on the values of the parameters are derived that occur in the hybrid model that guarantee that the car will always move forward along a black tape, and will never get off the tape or move backward. In order to simplify the correctness proof, a transition system model is introduced that abstracts from the hybrid automaton in a rather drastic manner, but still preserves validity of the correctness properties of interest. Even though the original model does not involve any disturbances, the general parametric analysis of the system allows to exend the results in a trivial manner to a hybrid model in which several disturbances are allowed (mistakes in measurements of lengths, drift and jitter of the hardware clock, velocity, and distance between the two caterpillar treads).[8]

Limitations This method has been applied on an example that is interesting but, from a systems and control theory point of view, rather simple. Scaling the method to more realistic and complex problems is still a challenge.

2.3 On the correctness of an intrusion-tolerant group communication protocol

Relevance The contribution shows the usefulness of the timed automaton paradigm beyond the proper real-time domain. Like [6] it also confirms its usability for analysis with PVS.

Modelling Timed automata style.

Tools Murphi and PVS.

Summary Intrusion-tolerance is the technique of using fault-tolerance to achieve security properties. Assuming that faults, both benign and Byzantine, are unavoidable, the main goal of Intrusion-tolerance is to preserve an acceptable, though possibly degraded, service of the overall system despite intrusions at some of its sub-parts. In this paper, a correctness proof is presented of the Intrusion-tolerant Enclaves protocol via an adaptive combination of techniques, namely model-checking, theorem-proving and analytical mathematics. Murphi is used to verify authentication, and then PVS to formally specify and prove proper Byzantine Agreement,Agreement Termination and Integrity. Finally, robustness of the group key management module is proved mathematically.[15]

Limitations The use of PVS and Murphy requires an expert on these tools. The analysis has not been completely automated yet (this would require the addition of some special purpose theories to PVS).

2.4 Testing conformance of real-time applications by automatic generation of observers

Relevance The contribution shows the usefulness of the timed automaton paradigm as a basis for test case generation and test result interpretation on the basis of a real-life case study. Testing techniques offer essential methods for validation complementary to the main Ametist methods.

Modelling Network of timed automata.

Tools IF + TTG.

Summary A methodology for testing conformance of an important class of real-time applications in an automatic way, is proposed in [3]. The class includes all applications for which a specification is available and can be translated into a network of timed automata. The method relies on the automatic generation of an observer from the specification, on one hand, and on the instrumentation of the system to be tested, on the other hand. The testing process consists in feeding the traces generated by the instrumented system to the observer, which is a testing device, used to check conformance of a trace to the specification. This approach was applied to the NASA K9Rover case study.

Limitations The IF tool is yet an academic prototype and a large investment would be needed to turn it into an industrially acceptable tool. The instrumentation of the system to be tested is time consuming.

2.5 Notes on a UPPAAL model of the Welch/Lynch Clock Synchronization Protocol

Relevance This application presents model simplification techniques for TA-based analysis of clock synchronization protocols, and identifies useful extensions to the Uppaal input language.

Modelling Timed automata.

Tools Uppaal.

Summary A simplified algorithm for synchronized clock by Welch and Lynch is modelled in Uppaal. This model is too complex to let the Uppaal tool prove the essential properties of the model. It is therefore reduced to a simpler model that can be used to let Uppaal prove the properties of interest. [1]

Limitations The model reduction is not yet a generally applicable technique but requires much insight into the problem domain.

2.6 Modelling and Analysis of a Leader Election Algorithm for Mobile Ad Hoc Networks

Relevance The analysis with Uppaal of this algorithm has provided a show case for the usefulness and effectiveness of AMETIST technology for domain experts (Leslie Lamport).

Modelling Timed automata.

Tools Uppaal.

Summary This article shows the correctness of an algorithm that, while very simple, exhibits the interesting features of more complicated distributed algorithms. It uses modelling and model-checking with UPPAAL. The problem was posed by Leslie Lamport, and is inspired by a classic algorithm of Radia Perlman. The original algorithm constructs a spanning tree and maintains that tree by having the root periodically propagate an I am alive" message down it. A new tree is constructed if a failure caused some node to time out before receiving the message. The simple algorithm assumes an arbitrary network of nodes. Each node can send messages to its neighbors. The network need not be connected. The leader of a connected component is defined to be the lowest-numbered node in the component. The goal of the algorithm is for each node n to learn the leader of its connected component. Correctness of this algorithm means that if no failure or repair has occurred for a sufficiently long period of time, then every node knows its leader. [2]

Limitations The success of this verification was partly due to the fact the algorithm is not extremely complex and was defined already in a precise way, such that the modelling was not very time consuming.

2.7 Synthesis of Safe, QoS Extendible, Application Specific Schedulers for Heterogeneous Real-Time Systems

Relevance This generic application of controller synthesis methods to scheduling synthesis complements the other approaches taken in the AMETIST project, which rely on model checking or direct search algorithms. A disadvantage is that the entire state space must be generated to apply the method, an advantage its capacity to deal with heterogeneity and larger models. Direct comparison to AMETIST technology is difficult because of the prototypical nature of the implementation.

Modelling Timed automata.

Tools eCos application.

Summary This article presents a new scheduler architecture, which permits adding QoS policies to the scheduling decisions. Also a new scheduling synthesis method is presented that allows a designer to obtain a safe scheduler for a particular application and at the same time helps him in analysing the task interactions and the overall system behaviour. The scheduler architecture and scheduler synthesis method have not been developed for a particular application model and, therefore, can be used for heterogeneous applications, where there are periodic tasks, event-driven ones and tasks which are always enabled and where the tasks communicate through various synchronisation primitives. Finally, a prototype implementation of this scheduler architecture and related mechanisms on top of an open-source OS for embedded systems is presented.[14]

Limitations As already remarked, the entire state space needs to be generated, and the tool is still an academic prototype.

2.8 Description and Schedulability Analysis of the Software Architecture of an Automated Vehicle Control System

Relevance A nice example of a classical schedulability analysis in a new application context, which can provide a benchmark for AMETIST technology.

Modelling adhoc.

Tools none.

Summary This contribution describes the software architecture of an automated vehicle control system implemented in the PATH (Partners for Advanced Transit and Highways) is a research lab administered by the Institute of Transportation Studies (ITS), University of California, Berkeley, in collaboration with Caltrans. The system is responsible for automatic lateral and longitudinal control of a set of vehicles traveling in a *platoon* formation at close distance and at high speeds. The software architecture consists of a set of processes running concurrently and communicating through a *publish/subscribe* database. Some processes are triggered periodically by external inputs (e.g., from sensors) while others are triggered by events from other (internal) processes. The architecture is modelled as a set of periodic em tasks each consisting of a sequence of *sub-tasks* with varying priorities. A schedulability analysis is performed to check whether a set of timing requirements expressed as *deadlines* are met.[17]

Limitations This research is of exploratory nature, implying that there is yet no generally applicable theory or tool support.

2.9 Verification of Asynchronous Circuits using Timed Automata

Relevance This contribution shows the applicability of AMETIST technology to the hardware domain.

Modelling Timed automata.

Tools OpenKronos.

Summary This work [5] applies the timing verification tool OpenKronos, which is based on timed automata, to verify correctness of numerous asynchronous circuits. The desired behavior of these circuits is specified in terms of signal transition graphs (STG) and it is checked whether the synthesized circuits behave correctly under the assumption that the inputs satisfy the STG conventions and that the gate delays are bounded between two given numbers. The results demonstrate the viability of the timed automaton approach for timing analysis of certain classes of circuits.

Limitations The current approach needs to be enhanced with special heuristics in order to enlarge the scale of the examples on which it can be applied.

2.10 Analysis of a Protocol for Dynamic Configuration of IPv4 Link Local Addresses using Uppaal

Relevance This contribution presents an application of AMETIST technology to dynamic configuration protocols. Our conclusion is that Uppaal, which combines extended finite state machines, C-like syntax and concepts from timed automata theory, is able to model Zeroconf in a faithful and intuitive way, using notations that are familiar to protocol engineers. Our modeling efforts revealed some errors (or at least ambiguities) in the RFC that no one else spotted before. We also identify a number of points where Uppaal still can be improved. Its further interest lies in the complexity of the formalization; at the time of experimentation Uppaal could only handle the case for three hosts, which identifies this problem as a potential benchmark for timed model checking.

Modelling Timed automata.

Tools Uppaal.

Summary There are many situations in which one would like to use the Internet Protocol for local communication, for instance in the setting of in home digital networks or to establish communication between laptops. For these type of applications one would like to have a plug-and-play network in which new hosts automatically configure an IPv4 address. In this paper [9], a protocol is studied that has been proposed by the IETF to achieve this. Within this protocol a host needs 8 or 10 seconds to get an IP address, which is too long for users in the real world. And even after such a long waiting time it may occur that two hosts configure the same IP address. So how to improve the performance is a meaningful work. This protocol is formally specified as a network of timed automata, and use the Uppaal toolset to verify some properties. From a modelling perspective this case study is interesting since it involves the modelling of broadcast communication. The goal of the analysis focuses on two points: (1) no two hosts can have the same IP address (2) a host spends less time to acquire a new IP address. Some parametric constraints are given to achieve this goal. Even though full analysis of this protocol clearly requires a model in which probabilistic information can be incorporated, it is possible to establish some interesting correctness properties, even in a setting with more than 2 hosts. Due to state space explosion, Uppaal cannot explore the complete state space of the unrestricted model for more than two hosts. Thus this work presents a clear challenge for the timed model checking community.

Limitations Due to state space explosion only a model with three hosts can be handled. The challenge is thus to either improve UPPAAL or to find meaningful abstractions to reduce the size of the model.

2.11 Model checking dependabiliy attributes of wireless group communication

Relevance This contribution shows that stochastic extensions of the AMETIST technology can be used to apply model checking techniques for the dependability analysis of non-trivial group communication protocols.

Modelling Stochastic Activity Networks (SAN), Continuous Stochastic Logic.

Tools UltraSAN and ETMCC.

Summary Models used for the analysis of dependability and performance attributes of communication protocols often abstract considerably from the details of the actual protocol. These models often consist of concurrent sub-models and this may make it hard to judge whether their behaviour is faithfully reflecting the protocol. In this paper [16], it is shown how model checking of continuous-time Markov chains, generated from high-level specifications, facilitates the analysis of both correctness and dependability attributes. This is illustrated by revisiting a dependability analysis [8] of a variant of the central access protocol of the IEEE 802.11 standard for wireless local area networks. This variant has been developed to support real-time group communication between autonomous mobile stations. Correctness and dependability properties are formally characterised using Continuous Stochastic Logic and are automatically verified by the ETMCC model checker. The models used are specified as Stochastic Activity Nets.

Limitations Only partial properties have been checked. In order to do a more complete verification, abstraction techniques would be necessary.

2.12 Cost-Optimisation of the IPv4 Zeroconf Protocol

Relevance This contribution shows that stochastic extensions of the AMETIST technology can be used for solving cost-optimization problems for configuration-management protocols.

Modelling Discrete Time Markov Reward Models (DRM).

Tools Maple.

Summary This paper [4] investigates the tradeoff between reliability and effectiveness for the IPv4 Zeroconf protocol, proposed by Cheshire/Adoba/Guttman in 2002, dedicated to the self-configuration of IP network interfaces. A simple stochastic cost model of the protocol is developed, 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. An analytical expression for the user penalty is derived that is used to derive optimal configuration parameters of the network, restricting to those parameters which are under the control of a consumer electronics manufacturer. In particular, it is shown that minimal cost and maximal reliability are qualities that cannot be achieved at the same time.

Limitations The method itself is general and abstract, but the quality of its results is dependent on the quality of the estimation or measurement of certain parameters, which in practice may be rather hard.

2.13 Model Checker Aided Design of a Controller for a Wafer Scanner

Relevance This contribution shows that AMETIST technology can be used to obtain industrially relevant results (in embedded systems design) with a modest investment of time. It also demonstrates its compatibility with the complementary model checking technique SMV.

Modelling SMV model, timed automata.

Tools SMV and Uppaal.

Summary For a case-study of a wafer scanner from the semiconductor industry it is shown how model checking techniques can be used to compute (i) a simple yet optimal deadlock avoidance policy, and (ii) an infinite schedule that optimizes throughput. Deadlock avoidance is studied based on a simple finite state model using SMV, and for throughput analysis a more detailed timed automaton model has been constructed and analyzed using the UPPAAL tool. The SMV and UPPAAL models are formally related through the notion of a stuttering bisimulation. The results were obtained within two weeks, which confirms once more that model checking techniques may help to improve the design process of realistic, industrial systems. Methodologically, the case study is interesting since two models (and in fact also two model checkers) were used to obtain results that could not have been obtained using only a single model (tool).[11]

Limitations Due to some restrictions of the UPPAAL tool, the relation between the UPPAAL and the SMV models had to be established by some special purpose heuristics.

2.14 Analysis of a Biphase Mark Protocol with Uppaal and PVS

Relevance This contribution shows the applicability of AMETIST technology to the analysis of bit signal encoding. Moreover, it demenstrates how (Uppaal) timed automata models and analysis can be used to produce general parametric proofs with PVS.

Modelling Timed automata.

Tools Uppaal and PVS.

Summary The biphase mark protocol is a convention for representing both a string of bits and clock edges in a square wave. The protocol is frequently used for communication at the physical level of the ISO/OSI hierarchy, and is implemented on microcontrollers such as the Intel 82530 Serial Communications Controller. An important property of the protocol is that bit strings of arbitrary length can be transmitted reliably, despite differences in the clock rates of sender and receiver, and variations of the clock rates (jitter), and distortion of the signal after generation of an edge. In this article [18], it is shown how the protocol can be modelled naturally in terms of timed automata. We use the model checker Uppaal to derive the maximal tolerances on the clock rates, for different instances of the protocol, and to support the general parametric verification that we formalized using the proof assistant PVS. Based on the derived parameter constraints we propose instances of BMP that are correct (at least in this model) but have a faster bit rate than the instances that are commonly implemented in hardware.

Limitations The Uppaal verification only applies to specific instances of the protocol, whereas we would like to be able to derive constraints on the parameters. PVS allows us to derive such constraints, but use of PVS requires much expertise and is very time consuming.

2.15 Automatic verification of the IEEE 1394 root contention protocol with KRONOS and PRISM

Relevance This contribution shows the combination of AMETIST technology (timed automata and KRONOS), a probabilistic extension of timed automata, and the PRISM tool for analysing Markov decision processes, can be succesfully applied to the automated verification of protocols that depend on stochastic choice mechanisms.

Modelling Probabilistic timed automata, Probabilistic Timed Computation Tree Logic.

Tools KRONOS and PRISM.

Summary This article [7] reports on the automatic verification of timed probabilistic properties of the IEEE 1394 root contention protocol combining two existing tools: the real-time model-checker Kronos and the probabilistic model-checker Prism. The system is modelled as a probabilistic timed automaton. Kronos is first used to perform a symbolic forward reachability analysis to generate the set of states that are reachable with non-zero probability from the initial state, and before the deadline expires. This information is then encoded as a Markov decision process to be analyzed with Prism. This technique is applied to compute the minimal probability of a leader being elected before a deadline, for different deadlines, and to study the influence on this minimal probability, of using a biased coin, and considering different wire lengths.

Limitations For the general case, more techniques for compacting the state space are necessary, in order for the verification to be effective.

2.16 Model Checking the Time to Reach Agreement

Relevance This contribution shows the applicability of AMETIST technology to the field of partially synchronous systems.

Modelling Timed automata.

Tools Uppaal.

Summary The timed automaton framework of Alur and Dill is a natural choice for the specification of partially synchronous distributed systems. The past has shown, however, that verification of these systems by model checking usually is very difficult. Therefore, model checking techniques have thus far not really been used for their design, even though these techniques are widely used in other areas, e.g., hardware verification. The present paper [10] demonstrates that the revolutionary development of both the usability and the efficiency of model checking tools may change this. It is shown that a complex partially synchronous distributed algorithm can easily be modeled with the uppaal model checker, and that it is possible to analyze some interesting and non-trivial instances with reasonable computational resources. Clearly, such analysis results can greatly support the design of these systems: model checking tools may provide valuable early feedback on subtle design errors and hint at system invariants that can subsequently be used in the general correctness proof.

Limitations Because of limitations of the model checking tool, it is still necessary to investigate much energy into finding suitable abstractions of the system.

2.17 Timed Automata Based Analysis of Embedded System Architectures

Relevance This contribution shows the applicability of AMETIST technology to the (automated) evaluation of architectures for embedded systems. It is demonstrated that the use of standard AMETIST technology can compare favourably to existing analytical tools (MOSES) for timeliness analysis.

Modelling Timed automata.

Tools Uppaal.

Summary This article [12] shows that timed automata can be used to model and analyze timeliness properties (response time and throughput) of embedded system architectures. Using a case study inspired by industrial practice, it presents in detail how such a timed automata model is composed. This construction is elaborated upon because the modeling strategy often has a great influence on the efficiency of the performance model evalutation. The model is verified using the UPPAAL model checker. Based on the results, it is argued that timed automata can sometimes be a useful alternative to the stochastic and simulation based techniques that are prevalent in this domain. The proposed modeling strategy can easily be automated, which alleviates the difficulty of constructing timed automata models. Furthermore, is the size of the timed automata model allows analysis, then the results will be in general more accurate than, for instance, simulation results. This is particularly true for finding hard and exact upper bounds for the timeliness properties. The results are compared to earlier work where the same case study was explored using Modular Performance analysis which is based on real-time calculus.

Limitations Manual construction and maintenance of the models is error prone and should be automated to be useful in an industrial setting. Uppaal lacks the features that are necessary to conveniently perform a parameter sweep.

2.18 From StoCharts to MoDeST: a comparative reliability analysis of train radio communications

Relevance This contribution shows the applicability of stochastic extensions of AMETIST technology to the automated reliability analysis of wireless communication protocols.

Modelling StoCharts, MoDeST.

Tools MOTOR/Moebius.

Summary StoCharts have been proposed as a UML statechart extension for performance and dependability evaluation, and were applied in the context of train radio reliability assessment to show the principal tractability of realistic cases with this approach. In this paper [13], this bare feasibility result is extended in two important directions. First, we sketch the cornerstones of a mechanizable translation of StoCharts to MoDeST. The latter is a process algebra-based formalism supported by the Motor/Moebius tool tandem. Second, this translation is exploited for a detailed analysis of the train radio case study.

Limitations The translation from StoCharts to MoDeST has not yet been fully automated.

3 Conclusion

The reported internal case studies have been useful in one or more of the following respects:

- they illustrate the applicability of AMETIST technology to areas not covered by the core case studies;
- they illustrate the usefulness of certain extensions of the timed automata model and/or supporting tools;
- they provide (potential) benchmarks for AMETIST technology and its future evolutions.

Due to the limited effort that was invested in these case studies some possibilities for deeper comparison were necessarily left unexplored. It is expected, however, that the work on these internal case studies will be very useful for possible future AMETIST follow-up projects.

Out of the 18 case studies discussed in this deliverable 11 are concerned with the modelling and analysis of distributed algorithms and protocols, 5 with (control) software/scheduling for embedded systems, and one with verification of asynchronous circuits. If we would consider all case studies that have been carried out using timed automata based tools then we would probably observe similar proportions:

1. the key application area for timed automata technology was (and still is) analysis of distributed algorithms and protocols. In this domain, timed automata are applied frequently and almost routinely to industrially relevant verification problems. Ametist definitely contributed to the applicability (both in terms of easy/convenient modelling and verification

power) of timed automata technology for this type of systems. In fact, without the progress achieved in Ametist several of the case studies (e.g. [10, 9]) could not have been dealt with.

2. an application area of growing importance is the analysis and design of (control) software for embedded systems, both at the architectural level (see e.g. [12]), the level of task scheduling (see e.g. [11, 14]) and at the (hybrid) control level (see e.g. [8]). It was one of the original ambitions of Ametist to push the applicability of timed automata technology to the areas of scheduling and control, and based on the four main Ametist case studies and the five miscellaneous case studies, we may conclude that this goal has been achieved.

3. there are a few other areas, e.g. verification of asynchronous circuit [5], where the viability of timed automata approach has been shown.

It is interesting to note that in 7 out of the 18 case studies more than one tool has been used: timed automata tools are combined with theorem provers, computer algebra tools, finite state model checkers, tools that support analysis of stochastic models, testing tools, etc We see this as a very natural and clear trend.

References

[1] L. Aceto, G. Behrmann, J.F. Groote, and K. Larsen. Notes on a uppaal model of the welch/lynch clock synchronization protocol. Unpublished note, http://www.cs.auc.dk/ kgl/AcetoBehrmannGrooteLarsen.pdf, http://www.cs.auc.dk/ kgl/ClockSync.xml, 2004.

[2] G. Behrmann, K. Larsen, and A. Skou. Modelling and analysis of a leader election algorithm for mobile ad hoc networks. Modelling carried for on request by Leslie Lamport, links http:://www.cs.auc.dk/ kgl/Lamport1.pdf, http:://www.cs.auc.dk/ kgl/Lamport2.pdf, http:://www.cs.auc.dk/ kgl/leader.xml, http:://www.cs.auc.dk/ kgl/leader.q, 2003.

[3] S. Bensalem, M. Bozga, M. Krichen, and S. Tripakis. Testing conformance of real-time applications by automatic generation of observers. In *Runtime Verification (RV'04)*, 2004.

[4] H. Bohnenkamp, P. van der Stok, H. Hermanns, and F.W. Vaandrager. Cost-optimisation of the IPv4 zeroconf protocol. In *Proceedings of the International Conference on Dependable Systems and Networks (DSN2003), em San Fransisco*, pages 531–540, Los Alamitos, California, 2003. IEEE Computer Society. Available from World Wide Web: `http://www.cs.kun.nl/ita/publications/papers/fvaan/IPDS.html`.

[5] M. Bozga, H. Jianmin, O. Maler, and S. Yovine. Verification of asynchronous circuits using timed automata. In *Proceedings of TPTS'02 Workshop*. Elsevier, April 2002. Available from World Wide Web: `http://www-verimag.imag.fr/PEOPLE/Oded.Maler/Papers/async.ps`.

[6] D. Chkliaev, J. Hooman, and E. de Vink. Verification and improvement of the sliding window protocol. In *Proceedings TACAS'03*, pages 113–127. Lecture Notes in Computer Science 2619, Springer-Verlag, 2003. Available from World Wide Web: `http://www.cs.kun.nl/~hooman/SWP.html`.

[7] C. Daws, M.Z. Kwiatkowska, and G. Norman. Automatic verification of the IEEE 1394 root contention protocol with KRONOS and PRISM. *STTT*, 5(2-3):221–236, 2004. Available from World Wide Web: `http://www.springerlink.com/index/10.1007/s10009-003-0118-5`.

[8] A. Fehnker, F.W. Vaandrager, and M. Zhang. Modeling and verifying a Lego car using hybrid I/O automata. In *Third International Conference on Quality Software (QSIC 2003),* Dallas, Texas, USA, November 6 - 7, pages 280–289. IEEE Computer Society Press, 2003. Available from World Wide Web: `http://www.cs.kun.nl/ita/publications/papers/fvaan/LEGO.html`.

[9] B. Gebremichael, F.W. Vaandrager, and M. Zhang. Analysis of a protocol for dynamic configuration of IPv4 link local addresses using Uppaal. Report NIII-R06XX, Nijmeegs Instituut voor Informatica en Informatiekunde, University of Nijmegen, 2006. Available from World Wide Web: `http://www.cs.kun.nl/ita/publications/papers/fvaan/zeroconf`. To appear.

[10] M. Hendriks. Model checking the time to reach agreement. Technical Report ICIS-R05014, Institute for Computing and Information Sciences, Radboud University Nijmegen, 2005. Available from World Wide Web: `http://www.cs.ru.nl/research/reports/info/ICIS-R05014.html`. Submitted to the

International Conference on Formal Modelling and Analysis of Timed Systems (FORMATS'05).

[11] M. Hendriks, N.J.M. van den Nieuwelaar, and F.W. Vaandrager. Model checker aided design of a controller for a wafer scanner. Report NIII-R0430, Nijmegen Institute for Computing and Information Sciences, University of Nijmegen, 2004. Available from World Wide Web: `http://www.cs.kun.nl/ita/publications/papers/fvaan/HNV04.html`.

[12] M. Hendriks and M. Verhoef. Timed automata based analysis of embedded system architectures, 2005. Submitted to the 2nd Conference on the Quantitative Evaluation of Systems (QEST'05).

[13] H. Hermanns, D.N. Jansen, and Y.S. Usenko. From stocharts to modest: a comparative reliability analysis of train radio communications. In *Proc. 5th International Workshop on Software and Performance (WOSP'05)*, Palma de Mallorca, Spain, July 2005. To appear. Also appeared as a SFB/TR 14 AVACS Technical Report ATR-002.

[14] Ch. Kloukinas and S. Yovine. Synthesis of safe, qos extendible, application specific schedulers for heterogeneous real-time systems. In *Proceedings of '5th Euromicro Conference on Real-Time Systems (ECRTS'03)'*, Porto, Portugal, July 2003. Available from World Wide Web: `http://www-verimag.imag.fr/PEOPLE/Christos.Kloukinas/IN2P3/kloukinas_yovine.pdf`.

[15] M. Layouni, J. Hooman, and S. Tahar. On the correctness of an intrusion-tolerant group communication protocol. In *Proceedings 12th Conference on Correct Hardware Design and Verification Methods (CHARME 2003)*, pages 231–246. Lecture Notes in Computer Science 2860, Springer-Verlag, 2003. Available from World Wide Web: `http://www.niii.kun.nl/~hooman/CHARME03.html`.

[16] M. Massink, J.-P. Katoen, and D. Latella. Model checking dependabiliy attributes of wireless group communication. In *Dependable Systems and Networks - Performance and Dependability Symposium (DSN 2004)*, pages 711–720, Firenze, Italy, 2004. IEEE CS Press. Available from World Wide Web: `http://ametist.cs.utwente.nl/INTERNAL/PUBLICATIONS/UTPublications/MKL04.ps`.

[17] S. Tripakis. Description and schedulability analysis of the software architecture of an automated vehicle control system. In *EMSOFT*, 2002. Available from World Wide Web: `http://www-verimag.imag.fr/~tripakis/emsoft02-full.ps.gz`.

[18] F.W. Vaandrager and A.L. de Groot. Analysis of a biphase mark protocol with Uppaal and PVS. Technical Report NIII-R0445, NIII, Radboud University Nijmegen, 2004. Available from World Wide Web: `http://www.cs.kun.nl/ita/publications/papers/fvaan/BMP.html`.