1 Project details

1. Title: A common framework for the analysis of reactive and timed systems.
2. Acronym: COMFORTS
3. Principle investigator: dr.ir. T.A.C. Willemse.

2 Summary

English Summary

Model checking is a popular technique for verifying the designs of real-life systems, including real-time systems, hybrid systems, probabilistic systems and data-dependent systems. Unfortunately, the results underlying the techniques from such specialised areas can be hard to translate to other areas because they rely on particular models. Moreover, systems cannot be classified as e.g. strictly real-time or strictly data-dependent: in many cases, a system is in the intersection of various classes. While properties can be verified in their separate domains using dedicated techniques, verifying the behaviour in the intersection of various domains is often impossible with the currently available techniques.

We propose to address these issues by studying the existing approaches in one framework: Parameterised Boolean Equation Systems (PBESs). Work by Mateescu, Groote and Mateescu, and Groote and Willemse has shown that the model checking problem for data-dependent systems can be translated to solving PBESs. We propose to extend these results to real-time systems, and embed results from these specialist areas in the PBESs approach.

Apart from the advantages that are brought about by studying known results in a single framework, PBESs offer a novel, and sometimes unique view on the model checking problem. For instance, Groote and Willemse showed that certain verification problems can be transformed to easier problems, simply by determining the syntactic form of the PBES and looking up its solution. While such techniques are commonplace in mathematics (e.g. for differentiation of functions), they are unique to the field of model checking and deserve further investigations.

3 Composition of the Research Team

The research team consists of the following members:

Name	**Specialism**	**Institute**
prof.dr. J.C.M. Baeten	Process Algebras, Real-time	TU/e
prof.dr.ir. J.F. Groote	Model checking, Real-time	TU/e
dr. J.C. van de Pol	Model checking	CWI & TU/e
dr.ir. T.A.C. Willemse	Model checking, Testing	RU Nijmegen
⟨postdoc⟩	Model checking	TU/e

4 Research School

The research will be conducted under the auspices of the research school IPA.

5 Description of the Proposed Research

5.1 Background

Computer controlled systems (often referred to as *embedded systems*), have rapidly become part of our daily life, and we are increasingly reliant on them. Embedded systems come in different shapes, from kitchen appliances such as food processors to connectivity devices such as mobile phones to (personal) health care systems such as pace-makers to transportation systems such as cars and trains. The increasing processing power and new software developments have been the dominating factor in allowing us to create ever more powerful features in existing systems, and computers have been the enabling technology behind novel technologies such as the mobile phone.

Paradoxically, the quality of many devices has seriously decreased over the years. Due to the increasing complexity of the software involved, and the increasing entanglement between software and hardware, the software tends to be of rather poor quality. While mechanical wear and malfunctioning was the number one cause of failures in most of the applications several decades ago, computer and software failures are rapidly taking over this position. Software issues are commonly solved by regularly releasing new software versions, replacing old software or malfunctioning parts of it. Software bugs that were present at the time of production and shipment are thus fixed on-site.

Unfortunately, patching software is only a solution when the software involved is *non-critical*, and the malfunctioning of the device is a nuisance at most. Even so, each patch that is called for is expensive, and the negative exposure of the producer also has its impact on e.g. sales figures.

For critical software, such as software contained in cars, power plants, etc. it is of utmost importance that it works as required. By means of conventional methods such as code reviews, testing, *etcetera*, the software is validated. But to ensure that the code that is validated contains no conceptual flaws, more rigorous methods are needed. The hypothesis is that this can be achieved using *formal methods* in the design trajectory.

5.2 Problem statement

The field of formal methods studies and provides the means for analysing and describing complex systems with mathematical rigour. This includes pure software systems, but also the combinations of software and hardware systems. A wealth of description languages is available to model these systems, such as I/O automata [39], process algebras such as μCRL [22, 23] and various real-time and probabilistic extensions of ACP [10, 4, 5, 9], and real-time and hybrid languages such as timed automata [2] and hybrid automata [1]. Analysing whether the specifications, written in these languages, satisfy the properties they are meant to satisfy can be done by, e.g.

- Static program analysis. This is a technique that allows to explore existing code (e.g. Java or C code). It executes an abstract version of a program's semantics on abstract data, rather than concrete data.

- Process algebraic reasoning. Using abstraction and axiomatic reasoning a particular relation between a specification and an implementation is established. Usually the specification is well-understood and satisfies desirable properties. The relation between the specification and the implementation then reflects that those properties are preserved in the implementation.

- Model Checking. This technique essentially relies on building and analysing a finite model of the specification and checking whether the desired (qualitative or quantitative) property holds.

Model checking is by far the most popular means for verification. It is also used as a tool for other validation methods such as testing [18]. Its popularity can be explained by the fact that the process of verification using model checking is fairly straightforward: once the basic ingredients are present, it is *push-button technology*. Research into model checking techniques has made great strides since its introduction in the early '80s. This is mostly due to the introduction of advanced techniques such as, e.g. partial order reduction and symmetry reduction and the availability of clever representation techniques such as BDDs [14]. These have improved the applicability of model checking for real-life systems dramatically.

However, much remains to be done. In particular, when we focus on extra-functional properties, such as data-dependencies, security, real-time behaviours and probabilistic behaviours, we feel that model checking has not yet reached its full potentials. The results in model checking in this setting are diverse and scattered: each specialist field has its own gems, for example:

- Real time systems. In the seminal work of Alur and Dill [2] it was shown that for a restricted set of timed systems, called *Timed Automata*, a finite model could be extracted that preserves the properties of interest. Their work has been at the basis for many (un)decidability results in the years that followed. The theory has also led to several competitive tools for model checking real-time systems, such as Uppaal [36] and Kronos [13].

- Hybrid systems. Following the works of Alur and Dill on Timed Automata, attention quickly shifted to hybrid systems. Many of the (un)decidability results that were obtained for real-time systems were generalised (or disproved) for hybrid systems. The model, most frequently used, is that of *Hybrid Automata* [1], which is a generalisation of Timed Automata. Typical examples of tools that have come forth from these results are HyTech [30] and D/dt [7].

- Probabilistic and stochastic systems. Various model checkers exist for probabilistic and stochastic systems, such as ETMCC [31] and PRISM [35]. The models underlying these tools are in general finite-state models extended with various stochastic extensions (e.g. Markov Chains and Semi-Markov Processes).

- Data-dependent systems. Systems in which data that is communicated or manipulated can influence the flow of control and, vice versa, the flow of control has its impact on data. Tools and languages have been defined to verify classes of such data-dependent systems. An example of such a language is the language of Counter arithmetic with Lambda expressions and Uninterpreted functions (CLU) [15]. Verifications of expressions in this language using the tool UCLID have been restricted to safety properties. Another example is μCRL [22] where data and processes live on an equal footing.

Unfortunately, the intersection between these sets of systems is not empty. On the contrary, most systems are data-dependent, and contain aspects of probabilistic, real-time or hybrid behaviour. For instance, Internet protocols, such as TCP, portray real-time behaviour in combination with data dependent behaviour, where timing issues and data issues are entangled (e.g. timeouts that are dependent on data that has been communicated). While both types of extra-functional behaviours can be verified separately using their dedicated techniques, it is often the area where these extra-functional behaviours meet that is of crucial importance.

We observe that the diverse, and relatively separate visions and solutions for model checking within various problem domains have led to a situation in which results that are obtained in one domain are not easily transfered to other domains. There is a serious risk that this will hamper the applicability of model checking as a technique in the near future.

5.3 Goals

The problem we identified in the previous section is not easily solved. It requires the identification of a technique that unifies existing model checking techniques into a single formalism. In fact, the identified problem can be split in two separate problems:

1. Provide suitable candidate languages for specifying aspects of data dependent, real-time systems. We distinguish between the following two types of languages:

   - A specification language for denoting the behaviour of a system.
   - a property language for expressing the desirable properties over the studied systems.

   The specification language must be flexible enough to allow for future extensions in the direction of probabilities and hybrid phenomena. The same requirements are imposed on the property language.

2. Find a generally applicable technique that can be used to solve the model checking problem for the chosen languages. Moreover, this technique must be such that it can be studied in isolation.

The first problem is indeed important, but it is not the primary focus of the proposed project. We feel that there are many candidate languages available that will serve the goals of the proposed project. Among these candidates are mCRL2 (the successor of (timed) μCRL [45]) and timed I/O automata [34] and hybrid I/O automata [40]. As it stands we tend to choose mCRL2 [21], and the modal μ-calculus with data as found in [26] but we are open to alternatives during the course of the project. The second problem, however, *will* be the primary target of this project.

We observe that some initial work has already been done in this direction for data dependent systems, i.e. systems in which data is present and of influence on the flow of control. In [20], Groote and Mateescu showed that the model checking problem for the language μCRL could be translated to the problem of solving *Parameterised Boolean Equation Systems* (PBESs) [20]. Their approach was influenced by the excellent treatment of Mader [41, 42] on the use of *Boolean Equation Systems* (BESs) for model checking. Parameterised Boolean Equation Systems can be seen as an intermediate formalism consisting of fixpoint equations of first order boolean formulae, e.g. systems of equations such as $(\mu X(d{:}D) = \varphi)\ (\nu Y(e_0, \ldots, e_n{:}E) = \psi)$, where φ, ψ are first order formulae and μ and ν denote the least and largest fixpoints, respectively. PBESs can be studied in their own right [26], leading to e.g. novel solution methods.

In [25, 27], Groote and Willemse showed that using some elementary techniques, PBESs can be solved automatically using a semi-decision procedure. Moreover, the required PBESs can be generated automatically from a logical property and a behavioural specification. The prototype tool that was implemented as a result from these observations illustrated that the potentials of the technique are indeed very promising. This was confirmed by the results that were obtained by conducting several case studies of varying size (see [25, 27]) and of varying complexity w.r.t. the data dependencies. The approach to model checking using PBESs is illustrated in Figure 1: it is a two-step, modular process in which first the specification and the property are combined to yield a PBES. The second step in the process is to use a solution technique for PBESs.

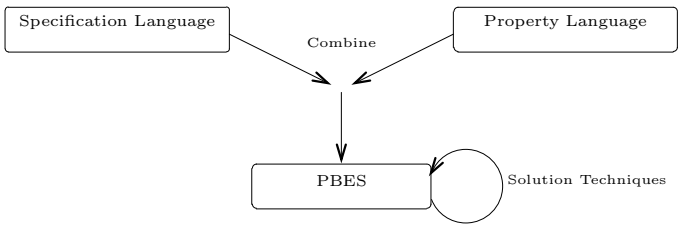

Figure 1: Verification approach using Parameterised Boolean Equation Systems

As a technique, the PBES methodology to model checking is very general and flexible: both optimisations at the level of the specification can be conducted (using e.g. process algebraic techniques to minimise the specification w.r.t. bisimulation), and different solution techniques can be used to solve the PBESs. The modelling language μCRL that has been used in combination with

the PBES methodology is quite expressive (see Luttik [38] for a study of the expressive power of μCRL, and Willemse [47] for the relation between timed μCRL and Hybrid Automata). The property language that is used in [20, 25, 27] is of similar expressive power: it is a first order extension of the well-known modal μ-calculus. The fact that the PBES methodology can be used for such expressive languages is a good indication that the technique is quite suited for analysing more involved properties, such as timing aspects and probabilities. Furthermore, the overall generality and flexibility of the PBES methodology makes it a good framework for studying (and, possibly uniting parts of) the various model checking approaches in existence.

5.4 Scientific Interest

The theory of Parameterised Boolean Equation Systems is complex. It is mainly this complexity that accounts for the relatively slow uptake of what must be considered a very promising technology. By now, it is also clear that the development of tooling for PBESs is of the utmost importance. Proof-of-concept prototype tooling has been a driving force behind the progress in theoretical PBES investigations. Furthermore, by conducting case-studies using such tooling, the required reflection on the applicability of the theory is obtained. This also helps in identifying challenges that need priority.

We propose the following lines of research:

1. Define the approach of using PBESs for timed and data-dependent, and, at a later stage, hybrid specification languages. We feel that with respect to probabilities and stochastics, the problems are currently not sufficiently well understood. Hence, we will not actively investigate the possibilities of combining probabilities and/or stochastics with the PBES methodology; of course, it may become a topic in the course of the project when increased insight shows how to deal with these.

2. Study solution techniques for PBESs in isolation. We distinguish between fundamental studies and pragmatic studies:

   - Fundamental issues to be solved focus in particular on the open questions that are raised in Groote and Willemse [26]. These include the quest for a more intuitive basic (semantic) theory for PBESs than the one that is used in [26], and finding an answer to the question whether there is a method to solve all PBESs of the form $\sigma X(d{:}D) = \varphi$ (where $\sigma \in \{\nu, \mu\}$ is a largest or least fixpoint) by replacing φ by a first order formulae in which X does not occur, meaning that PBESs are of equal power as first order logic. Answering such questions, either positively or negatively, gives fundamental insight into model checking as a technique and is bound to lead to new decidability results for many specialist areas, as these solution techniques are not application specific.
   - Pragmatic issues. These include finding original and novel types of solution techniques such as *pattern matching*. In [26], Groote and Willemse showed that it is possible to solve particular classes of PBESs, simply by looking up the answer. This means that the analysis of certain systems or certain properties using model checking can become elementary, where, currently, they are infeasible. The technique can best be compared to the well-established branch of mathematics that deals with differentiation: using a set of *patterns*, the derivative of many functions can be determined quickly. Many modern computer algebra systems incorporate such techniques.

     So far, for PBESs only four patterns have been determined. These patterns have already proved helpful in side-stepping some undecidable problems by transforming them to easier problems (see the examples of [26]).

     The quest is to find a large set of patterns that can ultimately be used in finding solutions to most popular model checking questions such as deadlock detection. The benefits are profound: pattern matching can vastly improve on model checking performance and memory usage: complex computations (such as e.g. approximating solutions) are reduced to cheap syntactical transformations.

3. An important line of research is to focus on transferring established decidability results from various settings to the PBES setting. This includes e.g.

   - the use of *zones* from Timed Automata [2], and the related decidability results for e.g. reachability. The same can be done for classes of Hybrid Automata [1].
   - relate symmetry reduction and partial order reduction techniques of finite-state model checking to the PBES approach. These techniques have been successfully applied in the setting of security [16], where they have led to effective tooling, see e.g. the tool Athena [46]. While some work on this has been done in the setting of real-time model checking [17], we expect to gain insights in some of the open problems in this setting. One of the conjectures is that many partial order reduction techniques are tied to pattern matching problems in PBESs. Such conjectures must be further investigated.
   - abstract interpretation, which has been used successfully for several classes of systems, including timed systems [29] and data dependent systems (see e.g. [33] and [44]).

   Studying these results in a single formalism will have a positive impact on problems from domains other than the ones for which they have been defined.

4. Additionally, techniques that are commonplace in process algebraic verification can be incorporated in the setting of model checking. For instance, in [37], Lin showed that the strong bisimulation relation of two systems can be calculated by solving a *predicate equation system*. These predicate equation systems resemble the PBESs of [20, 26], although the exact relationship must still be investigated. We expect to obtain similar results when we focus on other equivalence relations such as branching bisimulation. This means that genuine marbles such as the *Cones and Foci* method for branching bisimulation of Groote and Springintveld [24] may be reflected in the PBES approach. In turn, this will lead to a better understanding of the technique (and of process algebraic verification in general) and may result in similar techniques for other equivalence relations such as the ones described in [9, 5, 8]. Over time, this is expected to provide a flourishing exchange of ideas from the traditionally separated areas of model checking and process algebraic verification. It may also help tackle some of the long-standing open issues such as the use of abstraction in real-time systems which is still not well-understood.

The study of model checking techniques of several specialist fields in a single framework is expected to accelerate cross-fertilisations between the diverse and specialist approaches in model checking. It is especially in the use of novel solution techniques for PBESs (such as the use of patterns and invariants [26]) and in the combinations of known results that we expect to be able to push the limits of model checking.

At the end of the project, we expect to have tooling and a single methodology for model checking various classes of systems. These would include data-dependent systems and real-time systems, and systems that can be found in the intersection of these two classes. If everything goes well, steps towards hybrid systems and possibly even probabilistic systems can be expected. The challenge is to apply the developed techniques and verification methodology to complex real-life systems, including security protocols and Internet control protocols.

5.5 Related Work

Apart from recent work conducted by Groote and Mateescu [20], Groote and Willemse [25, 26, 27, 28], and Zhang and Cleaveland [48], and research that is conducted on finding new or more efficient solution techniques for *Boolean Equation Systems* (BESs, a rudimentary form of PBESs in which there is no explicit notion of data and data-dependency), research in PBESs is still in its infancy. Most works on model checking focus on furthering the bounds on decidability within their own specialist areas. In particular in the area of real-time systems [2, 29, 3, 32] and data dependent systems [33, 15], work is continuing to find new decidability results. Boundary-crossing techniques and investigations are rare, which makes the proposed project rather unique. There are, however,

several noteworthy influences to the project, such as the works conducted on *Parametric Timed Automata* [3] (and follow-up work conducted in [32]), and the works underlying the tool TReX [6]. However, we feel that the problems tackled by these approaches are only the tip of the iceberg. For instance, Parametric Timed Automata are merely an example of how data affects timing for a small class of systems. Also, these works lack the generality that we foresee by using PBESs.

Another inspiration to the project is the use of Boolean Equation Systems (BESs) for model checking and equivalence checking. A prime example of a quite successful tool-suite that is based on BESs is CADP [19], which uses BESs in its model checking tools [43] and equivalence checking tools [11]. BESs are also used within the VeriGem project, a joint project by the TU/e, the CWI and the University of Twente. The aim of this project is to solve BESs in a distributed fashion using a *grid*. The results of this project are likely to break new speed and size records for model checking. At the same time, the VeriGem project, and tool-suites such as CADP, can benefit greatly from the techniques that will be investigated in the proposed project, as the techniques that will be developed within the proposed project, such as structural rewriting, are complementary to the techniques used within e.g. CADP or VeriGem.

To conclude, the effort to study such a wide range of existing model checking techniques in a single framework is unique in the Netherlands, and appears to be unique even worldwide.

5.6 Local Embedding

The project will be carried out in two groups of the the Eindhoven University of Technology, viz. the Formal Methods Group and the Design and Analysis of Systems Group.

The Formal Methods group has a strong reputation in the area of process algebras. In particular, the work focuses on the use of process algebra for semantics and verification of security, real-time, hybrid and probabilistic systems. Furthermore, the group is actively exploring the use of formal methods for software development, using, among others, techniques from model checking (see e.g. [12, 17]).

The Design and Analysis of Systems Group is directed towards the transformation of basic techniques such that they become effective tools in the development of actual computer controlled systems. Currently, their main activity is the development of the specification language mCRL2 [21], as a successor of μCRL, together with the associated tools, theory and working practice.

Currently, both groups have various projects that are running, of which we only list the ones that are relevant for this project and briefly explain their relevance to this project.

1. BRICKS: Transmission Control Protocols for the Internet. This project runs within the theme of "Algorithms and Formal Methods" of BRICKS and it focuses on the use of techniques from process algebraic verification for Internet control protocols such as the TCP. Such protocols are typically in the intersection of data-dependent and real-time systems.

2. BRICKS: Protocols for secure infrastructure and e-commerce. This project runs within the theme of "Parallel Distributed Computing" of BRICKS. The focus of this project is to develop a system and methodology for the engineering of provably secure (multicast) security protocols. One of its aims is to improve the tool support for the verification of security protocols.

3. Tools and techniques for integrating performance analysis and system verification (TIPSy, see `http://www.niwi.knaw.nl/nl/oi/nod/onderzoek/OND1298386/toon`). The focus of this project is on developing and enhancing existing algorithms for system verification.

4. "Vernieuwingsimpuls": Improving the Quality of Protocol Standards (for more information, see `http://www.win.tue.nl/oas/index.html?iqps/index.html`). This project aims to participate in the standardisation committees for international communication standards and to improve the quality of these standards by applying formal techniques.

5. NWO project: Visualization of Large Transition Graphs (VoLTS). This project aims to visualise the structure of transition systems with millions of states that arise when analysing models of computer controlled systems.

6. Focus Project VeriGem: The availability of cluster computers induces the question whether these can be used for the verification of modal formulas. Techniques for doing so are developed within this project.

The output of the TIPSy project and the running BRICKS projects will serve as input for case studies for the proposed project. Vice versa, the model checking tools and techniques developed within the proposed project are expected to result in rapid feedback to these projects, thereby providing a valuable contribution to these three projects.

6 Expected Use of Instrumentation

The project can be carried out using standard equipment and does not require additional equipment.

References

[1] R. Alur, C. Courcoubetis, N. Halbwachs, T.A. Henzinger, P.-H. Ho, X. Nicollin, A. Olivero, J. Sifakis, and S. Yovine. The algorithmic analysis of hybrid systems. *Theoretical Computer Science*, 138:3–34, 1995.

[2] R. Alur and D.L. Dill. A theory of timed automata. In *Theoretical Computer Science*, volume 126, pages 183–235, 1994.

[3] R. Alur, T.A. Henzinger, and M.Y. Vardi. Parametric Real-Time Reasoning. In *Proceedings of the 25th Annual Symposium on Theory of Computing (STOC)*, pages 592–601. ACM Press, 1993.

[4] S. Andova. Time and probability in process algebra. In T. Rus, editor, *Proceedings of AMAST 2000*, volume 1816 of *LNCS*, pages 323–338. Springer-Verlag, 2000.

[5] S. Andova and J.C.M. Baeten. Abstraction in probabilistic process algebra. In T. Margaria and W. Yi, editors, *Proceedings of TACAS 2001*, volume 2031, pages 204–219. Springer-Verlag, 2001.

[6] A. Annichini, A. Bouajjani, and Mihaela Sighireanu. TReX: A Tool for Reachability Analysis of Complex Systems. In *Proceedings of Computer Aided Verification*, volume 1855 of *LNCS*, pages 368–372. Springer-Verlag, 2001.

[7] E. Asarin, O. Bournez, T. Dang, and O. Maler. Approximate reachability analysis of piecewise-linear dynamical systems. In *Hybrid Systems: Computation and Control, Third International Workshop*, volume 1790 of *LNCS*, pages 21–31. Springer-Verlag, 2000.

[8] S. Andova and T.A.C. Willemse. Equivalences for Silent Transitions in Probabilistic Systems. In *Proceedings of the 11th International Workshop on Expressiveness in Concurrency, August 2004, London, Great Britain*, Elsevier (to appear).

[9] J.C.M. Baeten and C.A. Middelburg. Process Algebra with Timing. EATCS Monograph, Springer Verlag 2002.

[10] J.C.M. Baeten and W.P. Weijland. *Process Algebra*. Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, 1990.

[11] D. Bergamini, N. Descoubes, C. Joubert and R. Mateescu. BISIMULATOR: A Modular Tool for On-the-Fly Equivalence Checking. In *Proceedings of the 11th International Conference on Tools and Algorithms for the Constructions and Analysis of Systems TACAS'2005 (Edinburgh, Scotland)*, volume 3440 of *Lecture Notes in Computer Science*, pages 581–585, 2005.

[12] , D. Bošnaški, D. Dams and L. Holenderski. Symmetric SPIN. *International Journal on Software Tools for Technology Transfer*, 4(1):92-106, Springer-Verlag, 2002

[13] M. Bozga, C. Daws, O. Maler, A. Olivero, S. Tripakis, and S. Yovine. Kronos: A Model-Checking Tool for Real-Time Systems. In A.J. Hu and M.Y. Vardi, editors, *Proceedings of the 10th International Conference on Computer Aided Verification, Vancouver, Canada*, volume 1427 of *LNCS*, pages 546–550. Springer-Verlag, 1998.

[14] R.E. Bryant. Graph-based algorithms for Boolean function manipulation. *IEEE Transactions on Computers*, C-35(8):677–691, 1986.

[15] R.E. Bryant, S.K. Lahiri, and S.A. Seshia. Modeling and verifying systems using a logic of counter arithmetic with lambda expressions and uninterpreted functions. In *CAV 2002*, volume 2404 of *Lecture Notes in Computer Science*, pages 78–92. Springer-Verlag, 2002.

[16] C.J.F. Cremers and S. Mauw. Checking secrecy by means of partial order reduction. In D. Amyot and A.W. Williams, editors, *Proceedings of the fourth SDL and MSC Workshop, SAM 2004: Security Analysis and Modelling*, volume 3319 of *LNCS*, pages 177-194. Springer-Verlag, 2004.

[17] D. Dams, R. Gerth, B. Knaack, R. Kuiper. Partial-order Reduction Techniques for Real-time Model Checking. In *Formal Aspects of Computing*, 10(5-6): 469-482, 1998.

[18] A. Engels, L.M.G. Feijs, and S. Mauw. Test generation for intelligent networks using model checking. In E. Brinksma, editor, *Proceedings of TACAS'97*, volume 1217 of *LNCS*, pages 384–398. Springer-Verlag, 1997.

[19] J-C. Fernandez, H. Garavel, A. Kerbrat, R. Mateescu, L. Mounier, and M. Sighireanu. CADP (CAESAR/ALDEBARAN Development Package): A Protocol Validation and Verification Toolbox. In *Proceedings of the 8th International Conference on Computer-Aided Verification CAV'96*, volume 1102 of *LNCS*, pages 437–440. Springer-Verlag, 1996.

[20] J.F. Groote and R. Mateescu. Verification of temporal properties of processes in a setting with data. In A.M. Haeberer, editor, *AMAST'98*, volume 1548 of *LNCS*, pages 74–90. Springer-Verlag, 1999.

[21] J.F. Groote, A. Mathijssen, S. Ploeger, M.A. Reniers, Y.S. Usenko, M. van Weerdenburg and J. van der Wulp. mCRl2: specification and analysis of process behaviour with data and time. To appear as Technical Report. Eindhoven University of Technology, 2006.

[22] J.F. Groote and A. Ponse. The syntax and semantics of μCRL. In A. Ponse, C. Verhoef, and S.F.M. van Vlijmen, editors, *Algebra of Communicating Processes '94*, Workshops in Computing Series, pages 26–62. Springer-Verlag, 1995.

[23] J.F. Groote and M.A. Reniers. Algebraic process verification. In J.A. Bergstra, A. Ponse, and S.A. Smolka, editors, *Handbook of Process Algebra*, chapter 17, pages 1151–1208. Elsevier (North-Holland), 2001.

[24] J.F. Groote and J. Springintveld. Focus points and convergent process operators: a proof strategy for protocol verification. In *The Journal of Logic and Algebraic Programming* volume 49, pages 31–60, 2001

[25] J.F. Groote and T.A.C. Willemse. A checker for modal formulas for processes with data. In F.S. de Boer, M.M. Bosangue, S. Graf, and W.-P. de Roever, editors, *Proceedings of FMCO 2003*, volume 3188 of *LNCS*, pages 223–239. Springer-Verlag, 2004.

[26] J.F. Groote and T.A.C. Willemse. Parameterised boolean equation systems (extended abstract). In P. Gardner and N. Yoshida, editors, *Proceedings of CONCUR 2004*, volume 3170 of *LNCS*, pages 308–324. Springer-Verlag, 2004.

[27] J.F. Groote and T.A.C. Willemse. Model-checking processes with data. In *Science of Computer Programming*, volume 56, pages 251-273, 2005.

[28] J.F. Groote and T.A.C. Willemse. Parameterised boolean equation systems. In *Theoretical Computer Science*, volume 343, pages 332-369, 2005.

[29] N. Halbwachs, Y.-E. Proy, and P. Roumanoff. Verification of Real-Time Systems using Linear Relation Analysis. In *Formal Methods in System Design*, 11(2):157–185, 1997.

[30] T.A. Henzinger, P.-H. Ho, and H. Wong-Toi. HyTech: A model checker for hybrid systems. *Software Tools for Technology Transfer*, 1:110–122, 1997.

[31] H. Hermanns, J.-P. Katoen, J. Meyer-Kayser, and Markus Siegle. A Markov Chain Model Checker. In *Proceedings of Tools and Algorithms for Construction and Analysis of Systems 2000*, volume 1785 of *LNCS*, pages 347–362. Springer-Verlag, 2000

[32] T.S. 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*, 2002

[33] B. Jeannet. Dynamic Partitioning In Linear Relation Analysis. Application To The Verification Of Reactive Systems. *Formal Methods in System Design*, 23(1):5-37, 2003.

[34] 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), Cancun, Mexico*, pages 166–177. IEEE Computer Society, 2003.

[35] M. Kwiatkowska, G. Norman, and D. Parker. Probabilistic Symbolic Model Checking with PRISM: A Hybrid Approach. *International Journal on Software Tools for Technology Transfer (STTT)*, 6(2):128–142, 2004.

[36] K. Larsen, P. Pettersson, and W. Yi. Uppaal in a nutshell. *Springer International Journal of Software Tools for Technology Transfer*, 1, 1997.

[37] H. Lin. Symbolic transition graph with assignment. In U. Montanari and V. Sassone, editors, *Proceedings of CONCUR 1996*, volume 1119 of *LNCS*, pages 50–65. Springer-Verlag, 1996.

[38] S.P. Luttik. *Choice quantification in process algebra*. PhD thesis, University of Amsterdam, April 2002.

[39] N.A. Lynch. I/O Automata: A model for discrete event systems. In *22nd Annual Conference on Information Sciences and Systems*, pages 29–38, Princeton University, Princeton, N.J., 1988

[40] N.A. Lynch, R. Segala, and F.W. Vaandrager. Hybrid I/O automata. *Information and Computation*, 185(1):105-157, 2003.

[41] A. Mader. Modal μ-calculus, model checking and gauß elimination. In E. Brinksma, R.W. Cleaveland, K.G. Larsen, T. Margaria, and B. Steffen, *Tools and Algorithms for Construction and Analysis of Systems, First International Workshop, TACAS '95, Aarhus, Denmark*, volume 1019 of *Lecture Notes in Computer Science*, pages 72–88. Springer-Verlag, 1995.

[42] A. Mader. *Verification of Modal Properties Using Boolean Equation Systems.* PhD thesis, Technical University of Munich, 1997.

[43] R. Mateescu. Local Model-Checking of an Alternation-Free Value-Based Modal Mu-Calculus. In Proceedings of the 2nd International Workshop on Verification, Model Checking and Abstract Interpretation VMCAI'98 (Pisa, Italy), September 1998.

[44] J.C. van de Pol and M.E. Valero. Modal Abstractions in mCRL. In C. Rattray, S. Maharaj and C. Shankland, editors, *Proceedings of Algebraic Methodology and Software Technology (AMAST'04), Stirling, Scotland, July 2004*, LNCS 3116, pp. 409-425.

[45] M.A. Reniers, J.F. Groote, M.B. van der Zwaag, and J. van Wamel. Completeness of Timed mCRL. *Fundamenta Informaticae*, 50(3-4):361-402, 2002.

[46] D. Song, S. Berezin, and A. Perrig. Athena, a Novel Approach to Efficient Automatic Security Protocol Analysis *Journal of Computer Security*, volume 9(1,2):47–74, 2001.

[47] T.A.C. Willemse. Embeddings of Hybrid Automata in Process Algebra. In E.A. Boiten, J. Derrick and G. Smith, editors, *Fourth International Conference on Integrated Formal Methods IFM2004*, volume 2999 of *Lecture Notes in Computer Science*, pp. 343–362, Springer-Verlag, 2004.

[48] D. Zhang and R. Cleaveland. Fast Generic Model-Checking for Data-Based Systems. In F. Wang, editor, *Formal Techniques for Networked and Distributed Systems - FORTE 2005*, volume 3731 of *Lecture Notes in Computer Science*, pp. 83–97, 2005

Selected Key Publications

[1] S. Andova and T.A.C. Willemse. Branching bisimulation for probabilistic systems: characteristics and decidability. Accepted for publication in Theoretical Computer Science, Elsevier.

[2] J.C.M. Baeten and C.A. Middelburg. Process Algebra with Timing. EATCS Monograph, Springer Verlag 2002.

[3] J.F. Groote and T.A.C. Willemse. Model-checking processes with data. In *Science of Computer Programming*, 56: 251-273, 2005.

[4] J.F. Groote and T.A.C. Willemse. Parameterised boolean equation systems. In Theoretical Computer Science, 343: 332-369, 2005.

[5] J.C. van de Pol and M.E. Valero. Modal Abstractions in mCRL. In C. Rattray, S. Maharaj and C. Shankland, editors, *Proceedings of Algebraic Methodology and Software Technology (AMAST'04), Stirling, Scotland, July 2004*, LNCS 3116, pp. 409-425.