1 Research Summary

The primary focus of my research is to develop formal methods and tools which support the modeling and automated analysis of complex computational systems, including software systems, embedded systems and biological systems. The main emphasis is on approaches that scale well for realistic applications. My most notable contributions are in: Establishing a noncommutative Cayley-Hamilton theorem for finite automata; Showing that minimal nondeterministic finite automata may be related via linear transformations; Automatically detecting emergent properties in networks of cardiac myocytes; Automatically learning an efficient model for excitable cells; Defining a model checking technique that allows to trade time and space for precision and confidence; Defining compositional models for discrete and hybrid hierarchic automata, together with modular proof rules and search routines; Providing compositional semantics and refinement rules for UML sequence diagrams, and their automatic translation to statecharts; Providing an algebraic foundation of UML-RT in terms of trace categories; Giving a denotational semantics for dynamically reconfigurable systems. My work resulted in a number of publicly available tools, including model checkers jMocha, Hermes, Gmc and Tempo, and hybrid systems simulators Charon and Eha. Below is a brief description of this work, classified by projects and in inverse chronological order. Ongoing projects also contain a summary of future work.

Next-Generation Model Checking and Abstract Interpretation: With a Focus on Embedded Control and Systems Biology[1] This Expedition project is focused on far-reaching and transformative research into techniques based on Model Checking and Abstract Interpretation (MCAI) for analyzing the behavior of complex embedded and dynamical systems. Traditional MCAI has a 30-year record of success at checking properties of the behavior of discrete systems automatically, and has been used to detect subtle bugs in a variety of hardware and software applications, ranging from microprocessor designs and communication protocols to railway-switching systems and satellite-control software. The purpose of this project is to extend the MCAI paradigm to reasoning about the behavior of models of physical systems that include continuous and stochastic behavior, such as those found in biological and embedded-control areas. Specific research is being undertaken in model discovery / system identification for stochastic and nonlinear hybrid systems; methods for generating sound model abstractions to simplify the reasoning process; and next-generation algorithms for analyzing the behavior of these models. Challenge problems in the area of pancreatic-cancer modeling, atrial-fibrillation detection, distributed automotive control, and aerospace control software are being used as technology drivers and testbeds for the results obtained in the course of the project.

Previous Work. Hybrid automata (HA) are a very popular modeling formalism for systems that exhibit both continuous and discrete behavior. Intuitively, HA are extended finite automata whose discrete states correspond to the various modes of continuous dynamics a system may exhibit, and whose transitions express the switching logic between these modes. The analysis of HA typically employs a combination of techniques borrowed from two seemingly disjoint domains: finite

[1]Funded by NSF CNS-Expeditions-09-26190.

automata (FA) theory and linear systems (LS) theory. As a consequence, a typical HA course first introduces one of these domains, next the other, and finally their combination. In [30, 31] we show that FA and LS can be treated in a unified way, as FA can be conveniently represented as discrete, time-invariant LS (DTLS), acting over semimodules, a generalization of vector spaces. Consequently, many techniques carry over from DTLS to FA. In particular: 1) FA minimization and FA determinization are observability and reachability transformations of their associated LS; 2) Minimal NFAs are related by linear transformations; 3) NFA satisfy a noncommutative Cayley-Hamilton theorem, which is a constructive version of the pumping lemma.

Future Work. We are currently working to obtain a hybrid, multiaffine approximation of the nonlinear model for cardiac myocytes, developed by our collaborators form the department of physiology of the Cornell University. This approximation is expected to preserve all the essential properties of the nonlinear model. Once this approximation is completed, we will work to identify the parameter-space surfaces separating a healthy cardiac tissue from a diseased one.

Survivable Software[2] Real-world software, especially the complex embedded-software systems that perform critically important functions, are prone to defects. Examples of such systems include the telephone system, fly-by-wire airplanes, and manned spacecraft. It is well-documented that these kinds of systems contain *residual defects*: software errors that show up after the code has been fully tested and delivered. Complicating matters is the fact that the amount of control software needed to, say, fly a space mission is rapidly approaching a *million lines of code*. Industrial statistics reveal that even very expensive, top-quality development processes only reduce the number of flaws in such code to somewhere in the order of 0.1 residual defects per 1,000 lines. Thus, a system with one million lines of code, may be expected to experience at least *100 defects while in operation*.

This project seeks to develop the theory, techniques and associated tools for *Survivable Software*, a new breed of software for real-world systems in general, and embedded applications in particular. The survivable-software paradigm is based on two core scientific principles: (a) Runtime software-monitoring of embedded systems that operate in dynamic, ever-changing environments is highly desirable; moreover, the monitoring can be performed cost-effectively with bounded-overhead guarantees, thereby compatible with tight resource constraints. (b) It is possible in many situations to equip embedded software with the facilities it needs to recover from runtime failures due to residual defects, so that the software can continue to function in a safe and acceptable manner.

Previous Work. Our work has focused so far on the development-automation and overhead-control of runtime-monitoring code. In [74] we introduce InterAspect, a GCC program-instrumentation framework and API, which allows instrumentation plug-ins to be developed in C, Java or GIMPLE, by using the familiar vocabulary of Aspect-Oriented Programming: pointcuts, join points, and advice functions. In [70] we present SMCO, our technique of Software Monitoring with Controllable Overhead. This technique is based on a novel combination of supervisory control theory of discrete event systems and proportional-integrative-derivative control theory of continuous systems.

Future Work. Current software-monitoring techniques are restricted to property-violation detec-

[2]Funded by AFOSR FA-0550-09-1-0481.

tion and possibly to rolling back. In most embedded-software applications however, this is not satisfactory. For example, after a collision detection, roll back is not possible. In purely continuous systems, the Simplex architecture is a much more desirable approach. A high-performance controller is used, as long as the state of the controlled system (the plant) resides well within the stability envelope of a high-confidence controller. Whenever the state approaches the margin of this envelope, the high-confidence controller takes over. Our current work is focused on extending this approach to hybrid and discrete systems.

Efficient Modeling and Analysis of Excitable Cell Networks using Hybrid Automata[3] An important open problem in systems biology is finding computational models that scale well for both simulation and formal analysis of biological processes. Although invaluable in revealing local interactions, most of the currently used models do not scale well, as they consist of large sets of nonlinear differential equations. By exploiting the all-or-nothing response of biological processes to external stimuli, this project seeks therefore to develop alternative, hybrid-automata (HA) models, with a particular focus on excitable (cardiac) cell networks (ECNs).

Previous Work. My results in [77, 80, 81, 75, 78, 10, 12, 11] indicate that HA models are able to capture the morphology of excitation and reproduce typical excitable cell characteristics, such as period of non-responsiveness and adaptation to pacing rates, with significantly improved computational efficiency. An important contribution is the introduction of a new kind of HA, that I call cycle-linear hybrid automata (CLHA), to highlight their cyclic structure and the fact that while in each cycle they exhibit linear dynamics, the coefficients of the corresponding linear equations and mode-transition guards may vary in interesting ways from cycle to cycle. My simulators exploit CLHA structure to obtain up to a 10-fold speedup compared to classical simulators.

In [45], I proposed the first algorithm to automatically learn, up to a prescribed error margin, the CLHA of an excitable cell from a set of training data. This algorithm first identifies the discrete states and transitions of the HA, and then for each state, it infers the corresponding linear differential equations and their associated parameters. The identification of the linear equations is based on an efficient exponential fitting algorithm, which is also used to identify the function describing the way the parameters vary according to the pacing rate. The algorithm was implemented in MATLAB, and provides the most accurate approximate model of excitable cells to date.

Using our CLHA models, I addressed the problem of specifying and detecting emergent behavior in networks of cardiac myocytes, spiral electric waves in particular, in [32]. To solve this problem I introduced a new concept of *spatial-superposition*, developed a new *spatial logic*, based on spatial-superposition, for specifying emergent behavior, devised a new method for *learning the formulae* of this logic from the spatial patterns under investigation, and applied bounded model checking to detect the onset of spiral waves. We have implemented our methodology as the Emerald tool-suite, a component of our Eha framework for specification, simulation, analysis and control of excitable hybrid automata. We illustrated the effectiveness of our approach by applying Emerald to the scalar electrical fields produced by our Cx simulator.

Future Work. My CLHA models render the formal analysis, simulation and control of excitable

[3]Funded by NSF CCF05-23863.

cells tractable. My current focus is on formal analysis [82, 79, 32]. Specifying safety properties is a nontrivial task, as these often have a spatial nature. For example, ventricular fibrillation is akin to a tornado with several vortexes. Analyzing such properties is difficult as well, as it involves detection of moving geometrical shapes and often probabilistic behavior. Scalability at organ level also demands a carefully defined notion of (bi)simulation.

Monte Carlo Model Checking[4] Model checking, the problem of deciding whether a property specified in temporal logic holds of a system model, has gained wide acceptance within the hardware and protocol verification communities. Model checking, however, is not without its drawbacks, the most prominent of which is state explosion, a phenomenon which makes traditional model checkers run out of memory and loose all the performed work. This project seeks to develop an alternative, quantifiable approach to coping with state explosion based on *Monte Carlo estimation*.

Previous Work. The main idea is to randomly sample cycles, reachable in both the system and (the negation of) the property automata. If the cycle contains an *accepting state*, then the sample is a *counter-example* to the model-checking problem. Given an error bound ϵ and confidence ratio δ, an optimal number of samples N is determined [51, 52, 54] such that, either a counter-example is found, or with confidence $1-\delta$, the probability of finding a counterexample through further sampling is less than ϵ. My preliminary experimental results with JMOCHA and GMC [51, 52, 36, 38] indicate that Monte Carlo model checking is fast, memory-efficient, and scales extremely well.

In order to preserve completeness, I introduced in [37] a randomized version of the classic depth-first-search reachability algorithm for the model-checking of safety properties. Due to its randomness, this algorithm is very robust, as it is impossible to construct a worst-case scenario. The bias towards the initial state of the system is reduced by letting the algorithm traverse each time, a random, longest loop-free path, whose first state can be any previously encountered open state (i.e., with children not yet explored). The first path uses the initial state as its starting state; afterwards, the choice is random. Experimental results on various benchmarks show that my new algorithm considerably outperforms the classic algorithm, and it is able to find counterexamples as deep as 2500 states (in these cases the classic algorithm ran out of memory).

Future Work. The uniform randomization of nondeterministic choices guarantees sample independence, a crucial property for the soundness and potentially parallel execution of the above algorithms. In some cases however, too many samples may be required for exploring error prone parts of the search space. Recently, I started to investigate search strategies, which use learning techniques to *guide* the search [76]. Sample independence for such strategies has to be guaranteed within a modified search space. On another line of research I am developing new Monte Carlo model-checking algorithms for timed and hybrid automata. The main idea is to sample the infinite space directly, and use the invariants, guards and differential equations as search heuristics. To compute the optimal number of samples I employ in this case Monte Carlo integration techniques.

Runtime Monitoring and Model Checking for High-Confidence Systems[5] System software, such as middleware and operating system kernels, is difficult to develop and maintain because

[4]Funded by NSF CAREER CCR01-33583.
[5]Funded by NSF CSR-AES05-09230

it is asynchronous and event-driven, often written in type-unsafe C, and complicated by caches, locks, and reference counts, all intended to improve performance. Being the critical infrastructure for all other applications, it should also elicit a sense of high confidence for its developers and users. The goal of this project is to apply advanced model-checking and code-instrumentation techniques, to achieve a balanced, confidence-driven monitoring of system software.

Previous Work. My overall approach is presented in [69]. Concurrent class machines, which account for class-based data structures, e.g., file systems, are introduced in [43]; they come equipped with an observational trace semantics, and compositional refinement rules. Initial results on my GCC tool suite for Monte Carlo static/runtime analysis of software systems are presented in [36, 25, 24].

The goal of monitoring is to detect when system behavior deviates from its specification, and when appropriate, initiate recovery actions or terminate system execution. Usually, this is achieved by instrumenting source code with checks that are always on, which considerable degrades system performance. To improve this situation, I developed a Monte Carlo, confidence-driven approach in [25, 24]. This starts with a two sided coin to decide whether to sample a monitoring event or not; if N error free samples are taken, the reliability-confidence δ is set accordingly, and a four sided coin is used; and so on. The larger the number of error-free samples, the closer is δ to one.

In [26, 23], I investigate a control-theoretic approach to the run-time verification problem. This work takes advantage of the new hardware and software features of the Linux OS, which allow to protect memory pages in a selective way and to trigger an interrupt in case of access. These interrupts are used by a model-predictive controller whose role is to minimize the monitoring overhead when profiling memory accesses or detecting memory leaks. All the Linux-OS work is based on the GCC-based platform for static and runtime verification I developed at Stony Brook.

In [72, 73], I investigate the transmission-power assignment problem for k-connected mobile ad hoc networks (MANETs), the problem of optimizing the lifetime of a MANET at a given degree k of connectivity by minimizing power consumption. Our proposed solution is fully distributed and uses a model-based transmission power adaptation strategy based on model-predictive control.

Future Work. There are several questions to be formally answered and practically tested in the above approaches. First, how appropriate is the Monte Carlo approach when environment distribution evolves in time? Second, what is a general notion of sample, within the ongoing computation of a reactive system, and in particular the OS? Third, given source-code I, specification S, and mapping C of locations in I to method calls in S, what calls can be eliminated to achieve a minimal instrumentation I_S? Fourth, what are the limits of model-predictive control for run-time monitoring? To assess progress and help transition results into practice, we will implement tool support and apply our techniques to the Embedded Linux OS.

Model Based Design and Verification of Embedded Systems[6] Computer aided verification (CAV) and embedded software design automation (ESDA) emerged from academia and industry, respectively, as two promising, model-checking-based approaches to developing the high-confidence software demanded by embedded-system applications. This project aims to push the limits of CAV

[6]Funded by NSF CAREER CCR01-33583.

towards modular ESDA models and increase confidence in ESDA by applying CAV techniques.

Models analysis. In [3, 1, 29] I developed hierarchic reactive modules (HRM), the first modeling language supporting *architectural hierarchy* as well as *behavioral hierarchy*, not only syntactically, but semantically, too: HRM comes equipped with a compositional notion of refinement, and compositional and assume/guarantee proof rules. In [8] I report on HERMES, a model checker for HRM which exploits hierarchy for efficient analysis, and in [5] I present JMOCHA, a model checker for Reactive Modules, that automates modular reasoning for architectural hierarchy. In [4, 6, 7] I developed CHARON, the analog of HRM and the first modeling language for hybrid systems, that uses hierarchy for efficient simulation, and comes equipped with a compositional notion of *refinement* and associated compositional proof rules. In [4] I also report on a simulator for CHARON. Various techniques for automated refinement checking of asynchronous processes are reported in [9].

Requirements analysis. Scenario-based descriptions (SDs) of interaction among distributed systems components, play a key role in ESDA. However, little progress was made to answer a main question about SDs: *how to assign them a formal meaning without compromising refinement?* In [53] I provide an automata-theoretic solution to this question for UML 2.1, and show that refinement in this setting is compositional. In [22] I provide a solution to another important question, namely how to synthesize the (hierarchic) state machines corresponding to a set of SDs. In [2] I introduce *shared variables interaction diagrams* (SVID) as the counterpart of SDs for the shared variables communication paradigm, and in [60, 59, 42] I extend SVIDs to hybrid systems.

Future Work of this project is performed in conjunction with the previously described projects.

A Framework for Modeling and Analyzing Complex Distributed Systems[7] Modern distributed systems deal with large numbers of heterogeneous computing and networking components to solve difficult computation and control problems. Despite the availability of distributed platforms, developing dependable systems for these platforms continues to be challenging. It is difficult to specify distributed systems and to reason about their correctness, efficiency, and fault-tolerance. The aim of this collaborative project between MIT, UConn and Stony Brook, is to produce a comprehensive computer-supported framework for modeling and analyzing distributed systems, based on sound mathematical principles, and to apply this framework to nontrivial examples.

Previous Work. My specification effort was devoted to the modeling of excitable-cell networks, a very complex and especially relevant kind of distributed systems [80, 81, 75, 78, 45]. This work is described in detail above (see NSF CCF05-23863). The implementation effort resulted in the the development of TEMPO, an Eclipse-based framework supporting the Timed Input/Output Automata (TIOA) modeling language of Nancy Lynch. So far, TEMPO features: (i) A front-end processor for TIOA, incorporating syntax and type checking, composition of automata, and various intermediate languages for the design and analysis tools; (ii) A simulation tool for individual TIOA or for pairs of requirements/implementation TIOAs; (iii) A seamless integration with the UPPAAL model checker; (iv) A smooth integration with the PVS theorem prover. While my main

[7] Funded by AFOSR STTR AF-2004-023

responsibility within the above effort was the UPPAAL integration, my previous experience with JMOCHA, GMC and CHARON was instrumental for every other part of the tool development.

Future Work. The analysis of excitable-cell networks and the synthesis of a distributed controller for such networks is going be performed in conjunction with the above described projects NSF CCF05-23863, CEWIT, NSF EFRI-CBE07-10312 and NSF ERC-1453378. The implementation effort will be devoted to the development of tool support for hybrid TIOA and their associated analysis and controller synthesis problems.

Methodologically Founded Development of RT-Systems[8] In response to industrial demand, OMG is standardizing UML-RT, a real-time (RT) variant of UML. As an expected, leading modeling language for RT-systems, a formal foundation of its various visual notations is necessary, to avoid incompatible interpretations and usages. This project aims towards such a foundation.

Actor diagrams and statecharts. In a cooperation between ObjecTime Limited, TU München and University of Bucharest [61, 34, 35, 44, 27] I developed an algebraic semantics for these diagrams, as *interaction graphs* (IG) interpreted multiplicatively and additively, respectively. IGs are related to Abramski's geometry of interaction, and built on top of causality capturing *flow graphs*.

Sequence diagrams and statecharts. In [17, 16] I provide a compositional *trace semantics* for (positive) SDs and in [71] I use this semantics to automatically *synthesize* the corresponding statecharts. The synthesis algorithm has been patented in both Germany and the the EU.

Hybrid and real-time extensions. In [56, 55, 58, 57] I developed HyCharts, a hybrid extension of flow graphs. Real time and its interaction with asynchronous message passing is studied in [50, 39].

Semantics and methodical use of UML. In [14, 33, 15] I study the relation between events, messages and methods and devise a method for specifying the dynamic behavior of objects. A coherent set of refinement techniques based on the UML standard is proposed in [40, 41, 18, 19, 20].

Dynamic Dataflow Networks[9] Motivated by Internet and OO-languages, the study of mobile systems has become a very popular research area. However, little has been accomplished so far, in developing a denotational understanding of mobility, which is an essential prerequisite for compositional development and analysis. This project aims towards such an understanding.

Deterministic systems. In my PhD thesis [28], I have developed a denotational model for concurrent OO-programs. This model regards an object configuration as a network of functions interacting over streams of messages, and is based on an implicitly typed λ-calculus, exhibiting union, intersection, conditional and recursive types, together with subtyping and parametric polymorphism.

Nondeterministic systems. Subsequently [63, 62, 64, 68, 67], I extended the functions to timed, input/output relations on streams, and assembled networks by parallel composition. Mobility was achieved by allowing components to communicate channel ports. The main emphasis in these models was on capturing, within the parallel composition operators, the special kind of dynamic hiding which characterizes mobile systems. Compositional refinement is addressed in [66, 65] and an extension with true mobility is obtained in [13], by distinguishing location components.

[8] Funded by BMR Br887/12-1

[9] Funded by NATO HTECH.CRG97-2948

Deduction-Oriented Specification and Design of Software[10] Reliable software development and analysis demands the correct, complete and precise capture of system requirements and designs, within a formal language. The goal of this project is to develop an axiomatic language, incorporating a notion of refinement, and associated tools to satisfy such a demand.

The language. An informal description and methodological use of the language SPECTRUM I have designed is given in [21]. Its static semantics is presented in [46] and its underlying logical calculus is discussed in [49, 48]. In contrast to other algebraic languages, SPECTRUM permits the use of partial functions, including non-strict, high-order and non-continuous functions, and of predicative polymorphism with type classes [47]. The language is especially designed for the modularization of large projects via hierarchical specifications. It is also powerful enough to express refinement which can be proven by logical deduction within a theorem prover like Isabelle.

References

[1] R. Alur and R. Grosu. Modular refinement of hierarchic reactive machines. In *Proc. of POPL'00, the 27th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, pages 390–402, Boston, Massachusetts, January 2000. ACM Press.

[2] R. Alur and R. Grosu. Shared variables interaction diagrams. In *Proc. of ASE'01, the 16th IEEE International Conference on Automated Software Engineering*, pages 281–289, San Diego, USA, November 2001. IEEE Press.

[3] R. Alur and R. Grosu. Modular refinement of hierarchic reactive machines. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 26(2):339–360, 2004.

[4] R. Alur, R. Grosu, Y. Hur, I. Lee, and V. Kumar. Modular specification of hybrid systems in Charon. In *Proc. of HSCC'00, the 3rd International Workshop on Hybrid Systems: Computation and Control*, volume 1790 of *LNCS*, pages 6–19, Pittsburgh, PA, March 2000. Springer Verlag.

[5] R. Alur, R. Grosu, M. Kang, B.Y. Wang, L.di Alfaro, T. Henzinger, R. Majumdar, C. Meyer, and F. Mang. JMOCHA: A model checking tool that exploits design structure. In *Proc. of ICSE'01, the 23rd International Conference on Software Engineering*, pages 835–836, Toronto, Canada, May 2001. IEEE Press.

[6] R. Alur, R. Grosu, I. Lee, and O. Sokolsky. Compositional refinement for hierarchical hybrid systems. In *Proc. of HSCC'01, the 4th International Workshop on Hybrid Systems: Computation and Control*, volume 2034 of *LNCS*, pages 33–48, Roma, Italy, March 2001. Springer Verlag.

[7] R. Alur, R. Grosu, I. Lee, and O. Sokolsky. Compositional modeling and refinement for hierarchic hybrid systems. *The Journal of Logic and Algebraic Programming (JLAP)*, 68(1):105–128,

[10] Funded by DFG (German Research Community) under codename SPECTRUM

2006. As of Oct. 31 2006, Ranked 3rd Among the Hottest Top 25 Articles of JLAP by ScienceDirect.

[8] R. Alur, R. Grosu, and M. McDougall. Efficient reachability analysis of hierarchic reactive machines. In *Proc. of CAV'00, the 12th International Conference on Computer-Aided Verification*, volume 1855 of *LNCS*, pages 280–295, Chicago, USA, July 2000. Springer Verlag.

[9] R. Alur, R. Grosu, and B.-Y. Wang. Automated refinement checking for asynchronous processes. In *Proc. of FMCAD'00, the 3rd International Conference on Formal Methods in Computer-Aided Design*, volume 1954 of *LNCS*, pages 55–72, Austin Texas, November 2000. Springer Verlag.

[10] E. Bartocci, F. Corradini, M.R. Di Berardini, E. Entcheva, S.A. Smolka, and R. Grosu. Modeling and simulation of cardiac tissue using hybrid i/o automata. *Theoretical Computer Science (TCS)*, 410(33–34):3149–3165, August 2009.

[11] E. Bartocci, F. Corradini, E. Entcheva, R. Grosu, and S.A. Smolka. CellExcite: an efficient simulation environment for excitable cells. *BMC Bioinformatics*, 9(Suppl 2):1–13, March 2008.

[12] E. Bartocci, F. Corradini, R. Grosu, E. Merelli, O. Riganelli, and S.A. Smolka. StonyCam: a formal framework for modeling, analyzing and regulating cardiac myocytes. In *Concurrency, Graphs and Models*, volume 5065 of *LNCS*, pages 493–502. Springer Verlag, June 2008.

[13] K. Bergner, R. Grosu, A. Rausch, A. Schmidt, P. Scholz, and M. Broy. Focussing on mobility. In *Proc. of the 32nd Annual Hawaii International Conference on System Sciences*, pages 8030–8039, Hawaii, January 1999. IEEE Press.

[14] R. Breu and R. Grosu. Modeling the dynamic behavior of objects: On events, messages and methods. In *Proc. Euro-Par'97, the 3rd International Euro-Par Conference*, volume 1300 of *LNCS*, pages 572–587, Passau, Germany, August 1997. Springer Verlag.

[15] R. Breu and R. Grosu. Relating events, messages and methods of multiple threaded objects. *The Journal of Object Oriented Programming (JOOP)*, 12(8):8–14, 2000.

[16] R. Breu, R. Grosu, C. Hofmann, F. Huber, I. Krueger, B. Rumpe, M. Schmidt, and W. Schwerin. Exemplary and complete object interaction descriptions. *Computer Standards and Interfaces*, 19(7):335–345, November 1998.

[17] R. Breu, R. Grosu, C. Hofmann, F. Huber, I. Krüger, B. Rumpe, M. Schmidt, and W. Schwerin. Exemplary and complete object interaction descriptions. In *Proc. of the OOPSLA'97 Workshop on Object-oriented Behavioral Semantics*, pages 1–11, Vancouver, Canada, October 1997.

[18] R. Breu, R. Grosu, F. Huber, B. Rumpe, and W. Schwerin. Towards a precise semantics for object-oriented modeling techniques. In *Proc. of the ECOOP'97 Workshop on Precise Semantics for Object-Oriented Modeling Techniques*, pages 53–61, Jyvaeskylae, Finland, May 1997. TU Munich. Longer version.

[19] R. Breu, R. Grosu, F. Huber, B. Rumpe, and W. Schwerin. Towards a precise semantics for object-oriented modeling techniques. In *OO-Technology, ECOOP'97 Workshop Reader*, volume 1357 of *LNCS*, pages 205–210. Springer Verlag, June 1997. As of Oct. 31 2006, it has 35 citations.

[20] R. Breu, R. Grosu, F. Huber, B. Rumpe, and W. Schwerin. Systems, views and models of UML. In *The Unified Modeling Language, Technical Aspects and Applications*, pages 93–109. Physica Verlag, Heidelberg, 1998.

[21] M. Broy, C. Facchi, R. Grosu, R. Hettler, H. Humann, D. Nazareth, F. Regensburger, O. Slotosch, and K. Stølen. The requirements and design specification language Spectrum. An informal introduction (V 1.0). Technical Report TUM-I9311,TUM-I9312, Technische Universität München, May 1993.

[22] M. Broy, R. Grosu, and I. Krüger. Automatically generating a program. *Electronic Version, United States Patent and Trademark Office*, 2002. http://patft.uspto.gov/netahtml/search-bool.html.

[23] S. Callanan, D.J. Dean, M. Gorbovitski, R. Grosu, J. Seyster, S.A. Smolka, and E. Zadok. Software monitoring with bounded overhead. In *Proc. of NGS'08, the Next Generation Software Workshop at IPDPS*, pages 1–8, Miami, Florida, USA, April 2008. IEEE Press.

[24] S. Callanan, R. Grosu, X. Huang, S.A. Smolka, and E. Zadok. Compiler-assisted software verification using plug-ins. In *Proc. of NGS'06, the Next Generation Software Workshop at IPDPS*, pages 1–8, Rhodes Island, Greece, April 2006. IEEE Press.

[25] S. Callanan, R. Grosu, A. Rai, , S.A. Smolka, M.R. True, and E. Zadok. Runtime verification for high-confidence systems: A monte carlo approach. In *Proc. of MBT'06, the 2nd Workshop on Model Based Testing*, volume 164(4) of *ENTCS*, pages 41–52, Vienna, Austria, 2006. Springer Verlag.

[26] S. Callanan, R. Grosu, J. Seyster, S.A. Smolka, and E. Zadok. Model predictive control for memory profiling. In *Proc. of NGS'07, the Next Generation Software Workshop at IPDPS*, pages 1–7, Long Beach, California, USA, March 2007. IEEE Press.

[27] R. Grosu. A formal foundation for UML for Real Time. Book project in progress. Started as habilitation thesis.

[28] R. Grosu. A formal foundation for concurrent object oriented programming. Technical Report TUM-I9444, Technische Universität München, January 1995.

[29] R. Grosu. And/Or hierarchies and round abstraction. In *Proc. of MFCS'00, the 25th International Symposium on Mathematical Foundations of Computer Science*, volume 1893 of *LNCS*, pages 52–67, Bratislava, Slovak Republic, August 2000. Springer Verlag.

[30] R. Grosu. Finite automata as time-invariant linear systems: Observability, reachability and more. In *Proc. of HSCC'09, the 12th International Conference on Hybrid Systems: Computation and Control*, volume 5469 of *LNCS*, pages 194–208, San Francisco, USA, April 2009. Springer Verlag.

[31] R. Grosu. The Cayley-Hamilton theorem for noncommutative semirings. In *Proc. of CIAA'10, the 15th International Conference on Implementation and Application of Automata*, volume 5469 of *LNCS*, pages 194–208, Winnipeg, Canada, August 2010. Springer Verlag.

[32] R. Grosu, E. Bartocci, F. Corradini, E. Entcheva, S.A. Smolka, and A. Wasilewska. Learning and detecting emergent behavior in networks of cardiac myocytes. In *Proc. of HSCC'08, the 11th International Conference on Hybrid Systems: Computation and Control*, volume 4981 of *LNCS*, pages 229–243, St. Louis, USA, April 2008. Springer Verlag.

[33] R. Grosu and R. Breu. Modeling the dynamic behavior of objects: On events, messages and methods. Technical Report TUM-I9804, Technische Universität München, February 1998.

[34] R. Grosu, M. Broy, B. Selic, and G. Stefanescu. Towards a calculus for UML-RT specifications. In *Proc. of the 7th OOPSLA Workshop on Behavioral Semantics of OO Business and System Specifications*, pages 1–18, Vancouver, Canada, October 1998. ACM Press.

[35] R. Grosu, M. Broy, B. Selic, and G. Stefanescu. What is behind UML-RT? In *Behavioral Specifications of Businesses and Systems*, pages 73–88. Kluwer Academic Publishers, 1999.

[36] R. Grosu, X. Huang, S. Jain, and S. A. Smolka. Open source model checking. In *Proc. of SoftMC'05, the 3rd Workshop on Software Model Checking*, volume 144(3) of *ENTCS*, pages 27–44, Edinburgh, Scotland, July 2005. Springer Verlag.

[37] R. Grosu, X. Huang, S.A. Smolka, W. Tan, and S. Tripakis. Deep random search for efficient model checking of timed automata. In F. Kordon and O. Sokolsky, editors, *Revised Selected Papers of MW'06, the 7th Monterey Workshop on Composition of Embedded Systems*, volume 4888 of *LNCS*, pages 111–124. Springer Verlag, Paris, France, October 2008.

[38] R. Grosu, X. Huang, S.A. Smolka, and P. Yang. Monte Carlo analysis of security protocols: Needham-Schroeder revisited. In *Proc. of DIMACS Workshop on Security Analysis of Protocols*, pages 1–10, Rutgers University, June 2004.

[39] R. Grosu, C. Klein, and M. Broy. Reconciling real-time with asynchronous message passing. In *Proc. of FME'97, the 4th International Symposium Formal Methods Europe*, volume 1313 of *LNCS*, pages 182–200, Graz, Austria, September 1997. Springer Verlag.

[40] R. Grosu, K. Klein, and B. Rumpe. Enhancing the SysLab system-model with state. Technical Report TUM-I9631, Technische Universität München, July 1996.

[41] R. Grosu, K. Klein, B. Rumpe, and M. Broy. State transition diagrams. Technical Report TUM-I9630, Technische Universität München, July 1996.

[42] R. Grosu, I. Krueger, and T. Stauner. Requirements specification of an automotive system with hybrid sequence charts. In *Proc. WORDS'99, the 5th International Workshop on Object-oriented Real-time Dependable Systems*, pages 149–154, Monterey, California, November 1999. IEEE Press.

[43] R. Grosu, Y.A. Liu, S.A. Smolka, S.D. Stoller, and J. Yan. Automated software engineering using concurrent class machines. In *Proc. of ASE'01, the 16th IEEE International Conference on Automated Software Engineering*, pages 297–307, San Diego, USA, November 2001. IEEE Press.

[44] R. Grosu, D. Lucanu, and G. Stefanescu. Mixed relations as enriched semiringal categories. *Journal of Universal Computer Science (JUCS)*, 6(1):112–129, 2000.

[45] R. Grosu, S. Mitra, P. Ye, E. Entcheva, IV Ramakrishnan, and S.A. Smolka. Learning cycle-linear hybrid automata for excitable cells. In *Proc. of HSCC'07, the 10th International Conference on Hybrid Systems: Computation and Control*, volume 4416 of *LNCS*, pages 245–258, Pisa, Italy, April 2007. Springer Verlag.

[46] R. Grosu and D. Nazareth. The specification language Spectrum - Core language report V1.0. Technical Report TUM-I9402, Technische Universität München, August 1994.

[47] R. Grosu and D. Nazareth. Towards a new way of parameterization. In *Proc. of the 3rd Maghrebian Conference on Software Engineering and Artificial Intelligence*, pages 383–392, Rabat, Marocco, April 1994.

[48] R. Grosu and F. Regensburger. The logical framework of Spectrum. Technical Report TUM-I9402, Technische Universität München, March 1994.

[49] R. Grosu and F. Regensburger. The semantics of Spectrum. In *Proc. of HOA'93, the 1st International Workshop on Higher-Order Algebra, Logic, and Term Rewriting*, volume 816 of *LNCS*, pages 124–145, Amsterdam, The Netherlands, September 1994. Springer Verlag.

[50] R. Grosu and B. Rumpe. Concurrent timed port automata. Technical Report TUM-I9533, Technische Universität München, October 1995.

[51] R. Grosu and S.A. Smolka. Quantitative model checking. In *Proc. of ISoLA'04, the 1st International Symposium on Leveraging Applications of Formal Methods*, pages 165–174, Pahphos, Cyprus, November 2004.

[52] R. Grosu and S.A. Smolka. Monte Carlo model checking. In *Proc. of TACAS'05, the 11th International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, volume 3440 of *LNCS*, pages 271–286, Edinburgh, Scotland, April 2005. Springer Verlag.

[53] R. Grosu and S.A. Smolka. Safety-liveness semantics for UML 2.0 sequence diagrams. In *Proc. of ACSD'05, the 5th International Conference on Application of Concurrency to System Design*, pages 6–14, St Malo, France, June 2005. IEEE Press.

[54] R. Grosu and S.A. Smolka. Monte carlo methods for process algebra. In *Proc. of the Int. Workshop on Algebraic Process Calculi: The First Twenty Five Years and Beyond*, volume 162(1) of *ENTCS*, pages 203–207, Bertinoro, Italy, September 2006. Springer Verlag.

[55] R. Grosu and T. Stauner. Modular and visual specification of hybrid systems - an introduction to hycharts. Technical Report TUM-I9801, Technische Universität München, December 1998.

[56] R. Grosu and T. Stauner. Visual description of hybrid systems. In *Proc. of WRTP'98, the 23rd IFAC-IFIP Workshop On Real Time Programming*, pages 1–6, Shantou, Guandong Province, P. R. China, June 1998. Elsevier Science Ltd.

[57] R. Grosu and T. Stauner. Modular and visual specification of hybrid systems. An introduction to HyCharts. *Formal Methods in System Design (FMSD)*, 21(1):5–38, July 2002.

[58] R. Grosu, T. Stauner, and M. Broy. A modular visual model for hybrid systems. In *Proc. of FTRTFT'98, the 5th International Symposium Formal Techniques in Real-Time and Fault-Tolerant Systems*, volume 1486 of *LNCS*, pages 75–91, Lyngby, Denmark, September 1998. Springer Verlag.

[59] R. Grosu, T. Stauner, and I. Krueger. Hybrid sequence charts. In *Proc. of ISORC'00, the 3rd IEEE International Symposium on Object-oriented Real-time distributed Computing*, pages 104–114, Newport Beach, California, March 2000. IEEE Press.

[60] R. Grosu, T. Stauner, and I. Krüger. Hybrid sequence charts. Technical Report TUM-I9914, Technische Universität München, July 1999.

[61] R. Grosu, G. Stefanescu, and M. Broy. Visual formalisms revisited. In *Proc. of ACSD'98, the 1st International Conference on Application of Concurrency to System Design*, pages 41–52, Aizu-Wakamatsu, Japan, March 1998. IEEE Press.

[62] R. Grosu and K. Stølen. A denotational model for mobile point-to-point dataflow networks. Technical Report TUM-I9527, Technische Universität München, October 1995.

[63] R. Grosu and K. Stølen. A denotational model for mobile many-to-many dataflow networks. Technical Report TUM-I9622, Technische Universität München, May 1996.

[64] R. Grosu and K. Stølen. A model for mobile point-to-point data-flow networks without channel sharing. In *Proc. of AMAST'96, the 5th International Conference on Algebraic Methodology and Software Technology*, volume 1101 of *LNCS*, pages 504–519, München, Germany, July 1996. Springer Verlag.

[65] R. Grosu and K. Stølen. Specification of mobile systems. In M. Haveraaen and O. Owe, editors, *Proc. of NWPT'96, the 8th Nordic Workshop on Programming Theory*, pages 67–76, Oslo, Norway, December 1996. University of Oslo.

[66] R. Grosu and K. Stølen. Compositional specification of mobile systems. Technical Report TUM-I9748, SFB-342/29/97A, Technische Universität München, November 1997.

[67] R. Grosu and K. Stølen. Stream based specification of mobile systems. *Formal Aspects of Computing (FAC)*, 13(1):1–31, 2001.

[68] R. Grosu, K. Stølen, and M. Broy. A denotational model for mobile point-to-point dataflow networks with channel sharing. Technical Report SFB-342/17/97A, Technische Universität München, May 1997.

[69] R. Grosu, E. Zadok, S.A. Smolka, R. Cleaveland, and Y.A. Liu. High-confidence operating systems. In *Proc. of EW'02, the 10th ACM SIGOPS European Workshop: Can we really depend on an OS?*, pages 205–208, Saint-Emilion, France, September 2002. ACM Press.

[70] X. Huang, J. Seyster, S. Callanan, K. Dixit, R. Grosu, S.A. Smolka, S.D. Stoller, and E. Zadok. Software monitoring with controllable overhead. To appear in International Journal on Software Tools for Technology Transfer (STTT), 2010.

[71] I. Krüger, R. Grosu, P. Scholz, and M. Broy. From MSCs to statecharts. In *Distributed and Parallel Embedded Systems*, pages 61–71. Kluwer Academic Publishers, 1999.

[72] O. Riganelli, R. Grosu, S. Das, C.R. Ramakrishnan, and S.A. Smolka. Power optimization in fault-tolerant manets. In *Proc. of Mascots'08, the 16th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems*, Baltimore, USA, September 2008. IEEE Computer Society.

[73] O. Riganelli, R. Grosu, S. Das, C.R. Ramakrishnan, and S.A. Smolka. Power optimization in fault-tolerant manets. In *Proc. of Hase'08, the 11th IEEE High Assurance Systems Engineering Symposium*, Nanjing, China, December 2008. IEEE Computer Society.

[74] J. Seyster, K. Dixit, X. Huang, R. Grosu, K. Havelund, S.A. Smolka, S.D. Stoller, and E. Zadok. Aspect-oriented instrumentation with GCC. In *Proc. of RV'10, the 1st International Conference on Runtime Verification*, volume 5469 of *LNCS*, pages 194–208, Malta, November 2010. Springer Verlag.

[75] M.R. True, E. Entcheva, S.A. Smolka, P. Ye, and R. Grosu. Efficient event-driven simulation of excitable hybrid automata. In *Proc. of EMBS'06, the 28th IEEE International Conference of the Engineering in Medicine and Biology Society*, pages 3150–3153, New York City, USA, August 2006. IEEE Press.

[76] Z. Yang, B. Al-Rawi, K. Sakallah, X. Huang, S.A. Smolka, and R. Grosu. Dynamic path reduction for software model checking. In *Proc. of iFM'09, the 7th International Conference on Integrated Formal Methods*, LNCS, Düsseldorf, Germany, February 2009. Springer Verlag.

[77] P. Ye, E. Entcheva, R. Grosu, and S.A. Smolka. Efficient modeling of excitable cells using hybrid automata. In *Proc. of CMSB'05, the 3rd Workshop on Computational Methods in Systems Biology*, pages 216–227, Edinburgh, Scotland, April 2005.

[78] P. Ye, E. Entcheva, S.A. Smolka, and R. Grosu. A cycle-linear hybrid-automata model for excitable cells. *IET Systems Biology*, 2(1):24–32, January 2008.

[79] P. Ye, E. Entcheva, S.A. Smolka, and R. Grosu. Symbolic analysis of the neuron. In *Proc. of ICBBE'08, the 2nd International Conference on Bioinformatics and Biomedical Engineering*, pages 836–839, Shanghai, China, May 2008. IEEE.

[80] P. Ye, E. Entcheva, M.R. True, S.A. Smolka, and R. Grosu. A cycle-linear approach to modeling action potentials. In *Proc. of EMBS'06, the 28th IEEE International Conference of the Engineering in Medicine and Biology Society*, pages 3931–3934, New York City, USA, August 2006. IEEE Press.

[81] P. Ye, E. Entcheva, M.R. True, S.A. Smolka, and R. Grosu. Hybrid automata as a unifying framework for modeling cardiac cells. In *Proc. of EMBS'06, the 28th IEEE International Conference of the Engineering in Medicine and Biology Society*, pages 4151–4154, New York City, USA, August 2006. IEEE Press.

[82] P. Ye, R. Grosu, S.A. Smolka, and E. Entcheva. Formal analysis of abnormal excitation in cardiac tissue. In *Proc. of CMSB'08, the 6th International Conference on Computational Methods in Systems Biology*, LNBI, Rostock, Germany, October 2008. Springer Verlag.