Multi-University Research Initiative on
High-Confidence Design for Distributed Embedded Systems

Frameworks and Tools for High-Confidence Design of Adaptive, Distributed Embedded Control Systems

Year 1 Progress Report

Janos Sztipanovits

Vanderbilt University
Institute for Software Integrated Systems
2035 Terrace Place, Nashville, TN 37203
(615) 322-3455 (office)
(615) 343-7440 (fax)
janos.sztipanovits@vanderbilt.edu

TEAM MEMBERS:

Vanderbilt: J. Sztipanovits (PI), G. Karsai and P. Volgyesi
UC Berkeley: C. Tomlin (Lead and co-PI), Edward Lee, George Necula and S. Sastry
CMU: Bruce Krogh (Lead and co-PI) and Edmund Clarke
Stanford: (Lead and co-PI) Stephen Boyd

FA9550-06-0312

1. Objectives

This project aims to develop a comprehensive approach to the model-based design of high-confidence distributed embedded systems. We will take advantage and fully leverage a shared theoretical foundation and technology infrastructure in four focus areas: hybrid and embedded systems theory, model-based software design, composable tool architectures and experimental testbeds. The objectives of our research in the focus areas are the following:

1. Develop theory of deep composition of hybrid systems with attributes of computational and communication platforms. We will address compositionality, concurrency, heterogeneity and resource, robustness, approximate verification and adaptive control architectures for uncertainty handling.
2. Develop foundations of model-based software design for high-confidence, networked embedded systems applications. We will investigate new semantic foundations for modeling languages and model transformations, precisely architected software and systems platforms that guarantee system properties via construction, and new methods for static source code verification and testing, as well as for dynamic runtime verification and testing.
3. Develop composable tool architecture that supports high-level reusability of modeling, model-analysis, verification and testing tools in domain-specific tool chains. We create new foundation for tool integration that goes beyond data modeling and data transfer.
4. Demonstrate the overall effort by creating an end-to-end design tool chain prototype for the model-based generation and verification of embedded controller code for experimental platforms.

2. Status of the Effort

We have started developing the basic components of the project along the four objectives.

Status of Objective 1:

1. We are developing a method for designing low-complexity controllers that achieve some desired performance level despite implementation errors. Most of these design problems are known to be hard and research was conducted to produce algorithms, which produce low-complexity controllers and which use Lyapunov certificates to guarantee the desired performance level. The method has been applied mostly in the context of linear systems, with some extensions to simple nonlinear ones. It has been shown to be very successful in producing controller designs of very low complexity.
2. We have developed new abstraction and iterative/adaptive refinement techniques to verify correctness of control software and hybrid dynamic systems. We are currently developing tools and real-scale applications to demonstrated and evaluate the effectiveness of these methods for design-time verification.
3. We completed initial research on new approaches to verification of floating point computations using model checker and methods to generate test cases for implemented embed-

ded systems from design models. We also initiated work on verification of translators for auto-code generation from design models.

Status of Objective 2:

1. We have started developed core elements of the prototype tool chain, FRED (placeholder name). The tool chain incorporates various modeling and verification tools for Controller Design, Platform and Component Modeling, and Code Generation. The tool chain is designed to be re-directable toward different platforms. Currently our two target platforms are TTP/C and RTAI/RTnet.
2. We have been working on experimenting with and building interface (for models and code) between time triggered platforms, TTP/C and TT-Ethernet and FRED.
3. We have been implementing high confidence code generator for the Ptolemy II actor languages using partial evaluation mechanisms. The code generator transforms an actor-oriented model into target code while preserving the model's semantics.

Status of Objective 3:

1. We have been developing model-transformation methods and implement model transformations for integrating tool chains. The research is built on the model transformation technology of the MIC tool suite (GReAT).
2. We work on the precise and formal specification of modeling languages using Abstract State Machine formalism. The current effort targets the compositional specification of semantics using elementary "semantic units".
3. We have developed PTIDES: Programming Temporally Integrated Distributed Embedded Systems. For components for embedded systems, we have further refined the Ptolemy II code generation environment and are targeting the quadrotor effort.

Status of Objective 4:

1. We have designed and built several successful first prototypes of the Stanford Testbed of Autonomous Rotorcraft for Multiple Agent Control (STARMAC) quadrotor aircraft. STARMAC is a fully autonomous aircraft with capability for trajectory tracking, hover, and with multiple kinds of sensors built on (laser range finder, stereo vision cameras).
2. We have begun the process of interfacing the Ptolemy toolkit with the embedded software control architecture on board our autonomous quadrotor aircraft.

3. Accomplishments and New Findings

3.1 Hybrid and Embedded Systems Theory

3.1.1 Embedded Systems Modeling and Deep Compositionality (Krogh)

During the past year, we have developed an abstraction technique for real-time systems. The main advantage of this technique is that it reduces the complexity of checking properties of certain types of real-time systems. This enabled applying verification techniques to larger software systems, like the real-time code needed to control UAVs.

3.1.2 Hierarchies of Robust Hybrid and Embedded Systems (Tomlin, Krogh, Sastry)

We have applied reachable set technologies to the analysis and design of collision avoidance schemes for multiple autonomous quadrotor aircraft, and to the very close formation flying of multiple fixed wing UAVs.

Inspired by the success of counterexample guided abstraction refinement (CEGAR) techniques in the verification of discrete systems, we developed an iterative relaxation algorithm for linear hybrid automata. Each iteration of our algorithm computes reachability of a low-dimensional linear hybrid automaton. We then analyze the reachability information to produce possible counterexamples. We use linear programming as an efficient counterexample validation procedure. If the counterexample is spurious, linear programming techniques identify an irreducible infeasible subset (IIS) of constraints from which a set of continuous variables is selected for the construction of the next relaxation abstraction. We have applied this technique to safety verification of an Automated Highway System and a nuclear plant control model. This approach outperforms state-of-the-art reachability engines by a factor of 1000 on some examples.

3.1.3 Verification and Validation of Conservative Approximations (Clarke)

We analyzed the limits of approximation techniques for continuous image computation in model checking hybrid systems. Even for a very restricted class of functions, our results prove that continuous image computation is not numerically semidecidable. These negative results are complemented by our discovery that symbolic insight about derivative bounds can be used for approximation refinement based model checking. We also show that numerical algorithms can perform continuous image computation with arbitrarily high probability. We successfully applied our theoretical and algorithmic ideas for the analysis of prerequisites for a safe operation of the roundabout maneuver in air traffic collision avoidance.

3.1.4 Adaptive Control Architectures for Uncertainty Handling (Boyd, Krogh)

We developed a method for truncating the coefficients of a linear controller while guaranteeing that a given set of relaxed performance constraints is met. Our method sequentially and greedily truncates individual coefficients, using a Lyapunov certificate, typically in linear matrix inequality (LMI) form, to guarantee performance. Numerical examples show that the method is surprisingly effective at finding controllers with aggressively truncated coefficients, that meet typical performance constraints. We also gave an example showing how the basic method can be extended to handle nonlinear plants and controllers.

In previous work, we developed a technique for testing embedded control software systematically. This type of software is crucial in the development of low level control for autonomous systems. In the past year, we have enhanced this approach, making it possible to prove the validity of critical software properties.

3.2 Model-Based Software Design and Verification

3.2.1 Model-Integrated Computing (Sztipanovits, Karsai, Volgyesi)

The Model-Integrated Computing approach advocates the use of domain-specific modeling languages in the development process, which naturally includes a multitude of such languages. For instance, design models are typically expressed in a different language than analysis models. Considering this fact, transformations on models are necessary that integrate the various tools.

These transformations must preserve the semantics of the models being transformed, thus their correctness is essential. Continuing our previous effort on verifying the correctness of such transformations, we have developed some initial approaches and tools for solving this verification problem. Our approach is based on verifying the transformation instance (which is feasible), and not on verifying the transformation in general (which is infeasible, if not impossible).

We have developed a technique that uses semantic anchoring (and thus a common semantic framework) for showing that the source and target of a model transformation are behaviorally equivalent (technically: bisimilar). The example we have used is based on translating models between two variants of Statecharts (a modeling approach often used in flight-control software). The result of model transformations is often source code (for instance, in C). Verification of such transformations is especially important because of the widespread use of 'autocode' generators in the industry. Note that this is not a general code verification problem, rather verification of generated code that is highly idiomatic. This means, that if we know these idioms (that the code generator uses) we can analyze the code (generated by that specific generator) and discover its properties. In a follow-up publication we have shown how extended hierarchical automata can be constructed from the auto-generated C code, whose behavior could be checked against the original model: a Stateflow-style behavioral model.

In a related effort, we have made improvements to our model transformation tool (GReAT) to support complex transformation operations. An example for such operation is as follows: suppose we want to allocate software components of a distributed embedded system to specific processors based on some criteria. We have developed a high-level operator 40 based on graph transformations that allows the easy specification of such a transformation step. Without such an operator, writing the transformation is difficult and error prone.

For the toolchain that we are constructing (discussed below), we have developed a number of code generators; for generating executable code from Simulink and Stateflow models. In the Stateflow code generator we have carefully followed the Stateflow semantics (as documented in the Mathworks documentation). Unfortunately, the generated code is complex, not easy to review, although it follows a clear structure. From the superficial review of such code one could – falsely- conclude that there is an infinite recursion in the code (which could lead to stack overflow, and thus a complete failure of the software). To demonstrate that the code does not exhibit such behavior, we have developed a dedicated static code analysis tool that independently verifies that the generated code does not get into an infinite recursion. The tool constructs a conditional control flow graph for the code and checks that no loops are present in the graph.

3.2.2 Embedded Software Composition Platform (Lee)

Our two primary contributions so far have been: the principled design of embedded software for challenging UAV control applications, and development of automatic code generation from a model based tool.

We are considering both a single sensor-rich UAV platform, and a multiple UAV platform, with search, surveillance, and collision avoidance applications. We have already designed embedded software and control for these platforms using "traditional" approaches of point-based designs, to demonstrate proof of concept. Now, we are in the process of coding models for dynamics, sensors, and control algorithms in Ptolemy II in order to automatically generate C code, which we can then test against our existing code. We believe that this process will lead us to a much better understanding of automatic code generation (and verification and validation) for challenging embedded applications.

We are considering both a single sensor-rich UAV platform, and a multiple UAV platform, with search, surveillance, and collision avoidance applications. We have already designed embedded software and control for these platforms using "traditional" approaches of point-based designs, to demonstrate proof of concept. Now, we are in the process of coding models for dynamics, sensors, and control algorithms in Ptolemy II in order to automatically generate C code, which we can then test against our existing code. We believe that this process will lead us to a much better understanding of automatic code generation (and verification and validation) for challenging embedded applications.

Our second main contribution has been significant development of the Ptolemy II C code generation facility. This facility allows Ptolemy II users to generate C code from Synchronous Dataflow (SDF) models that optionally use Finite State Machines (FSMs). The Ptolemy II code generation facility first shipped as part of Ptolemy II 6.0 on February, 2007. Code generation from high level models allows us to provide scalable, modular and composable designs on embedded systems. Our work in interface theories allows us to formalize key correctness properties.

3.2.3 Automated Source Code Verification and Testing (Clarke)

In previous work, we developed a technique for testing embedded control software systematically. This type of software is crucial in the development of low level control for autonomous systems. In the past year, we have enhanced this approach, making it possible to prove the validity of critical software properties.

Ongoing work on translator validation. Model-based development, e.g., with UML statecharts or Stateflow/Simulink, is an integral part of embedded system design. Automatic translations between meta-level design descriptions and actual code representations are of great interest due to reduced costs and efforts. Therefore, the validation of translators forms a significant technological challenge to enable failure-free high quality software. We are using our expertise in software verification and compilers to build solutions for this problem.

Test case generation from design models. Test cases for embedded control implementations are typically constructed manually, using engineering insight to identify extreme conditions and corner cases. We are looking at formal methods for generating test cases from design models with the objectives of (1) guaranteeing coverage of critical conditions that can arise from the unmodelled variability introduced by the target platform and the external environment; and (2) minimizing test cases taking into account the results of design-level verification. Our initial work has shown that it is possible to apply methods developed for protocol conformance using timed automata to this new domain. We are currently developing extensions of these methods to handle a broader class of variations that can arise in embedded control systems.

Verification of floating point computations. Current control law verification tools do not account for effects of floating point computations. We are developing verification techniques to account for numerical errors of the programs executing in the target processor environment by leveraging existing verification technology and by constructing methods for round-off over-approximations. An initial implementation for iterative linear programs has been done using Phaver, a hybrid system verification tool. We are currently developing methods for dealing with a wide class of specifications relevant to the performance of embedded control code.

Timing properties of distributed embedded control systems. We are developing methods for providing design-time guarantees for the stability and performance of distributed embedded control systems. These performance certificates will enable designers to make quantitative assumptions about the behavior of the final implemented controller, even in the presence of communication delays, jitter, and other timing uncertainties. Our initial research has focused on extensions of results for time-delay systems to the distributed case, and the application of simulation tools for numerical evaluation.

3.3 Composable Tool Architectures

3.3.1 Advanced Open Tool Integration Framework (Sztipanovits, Karsai)

Based on our earlier work, we have continued our efforts on the formal specification of behavioral semantics for domain specific modeling languages.

We have developed a new approach in incrementally defining semantic units in the Abstract State Machine framework. The main contribution in the last year was the establishment of a simplified basic library of semantic units for transition systems and for basic component interaction categories.

We progressed in understanding the compositional specification of semantics using semantic units. During the last year we have developed several case studies for understanding the principle of compositionality and for specifying a composition framework.

3.3.2 Prototype Tool Chain (Volgyesi, Karsai, Sztipanovits)

The primary goal of the prototype tool chain (FRED for now) is to provide a baseline transformation path from high level control models to executable code on significantly different platforms while providing open access through well defined interfaces to the intermediate steps along the path. These interfaces will enable other participants of the MURI project to integrate their tools and to be able to demonstrate their contributions in a complete end-to-end solution.

The tool chain divides the design and implementation process to roughly three partitions as it is shown in Figure 1. Initially, the components of the system are built in the MATLAB **Simulink/Stateflow** environment. Due to its widespread acceptance in the control community and its rich support for various problem domains this seemed to be a natural choice for modeling platform used for defining and verifying the dynamics of embedded controllers. Note that that are no strong couplings between the Simulink/Stateflow environment and the rest of the tool chain components, it is a relatively easy task to take input models from other similar modeling environment (eg.: from Ptolemy II, Labview or other tools). Currently, we import topology and parameter information from Simulink/Stateflow with automatic type inferencing on the signal paths with the **MDL2MGA** tool.

At the core of the next design partition is a domain-specific modeling language implemented in the GME meta programmable modeling environment. The role of this visual language (**ECSL-DP**) is twofold: it decouples the input modeling environment (ie: Simulink/Stateflow) from the rest of the toolchain and adds important aspects which are not addressed in the high level controller models. The most important aspect is the *component model*, where dataflow blocks can be assigned to software components, tasks and hardware nodes and the decisions are made on task scheduling and communication patterns. Although the ECSL-DP component language is rich enough to support different execution platforms, component model instances are built around a selected platform. The selected platform (somewhat similar to the Director concept in Ptolemy) determines the subset of the ECSLP-DP language that can be used in the given model along with quantitative parameters (eg.: time granularity, jitter, latency, message sizes and number representation). The insertion of these **platform properties** into the visual design environment is a key concept in the tool chain. The component model closely reflects the concepts of the adoption layer, described in later sections.

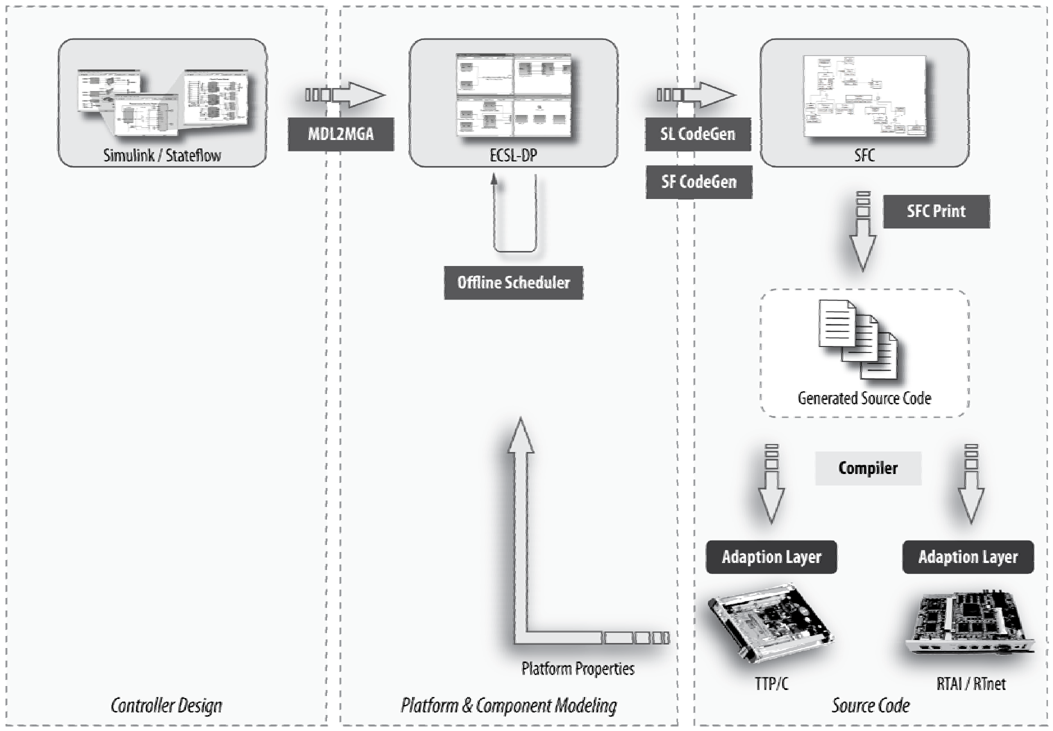

Figure 1: Initial architecture of the tool chain

As one of the most important interfaces for tool integration, the ECSL-DP models can be accessed by model transformators to support different aspects of the componentization process. Currently, we provide an **Offline Scheduler** for task and message scheduling on time triggered platforms. Based on the apriori assigned task rates and communication dependencies the scheduler computes the proper static schedule using constraint optimization techniques.

Target code generation is carried out in two phases. First, the ECSL-DP model is morphed into an intermediate language model using graph transformation techniques. This intermediate representation (**SFC**) is similar to an abstract syntax tree (AST) and source code generation is almost trivial from it. Due to the abstract representation it is relatively easy to target different imperative languages (C, C++, Java) and it provides a standard interface for source code verification tools. Note, that the SFC language is not completely general (like general C) but is tuned for realizing models with state machines and dataflow graphs. Due to technical reasons two graph transformation tools provide the mapping from ECSL-DP to SFC, dataflow structures are processed by **SL CodeGen** while state machines are transformed by **SF CodeGen**. Currently, we support C source file generation from SFC models with the help of the **SFCPrint** tool.

A unified **adoption layer** enables us to use a generic code generator tool suite on different platforms. The adoption layer is similar to the so called Board Support Packages: it hides the quirks and tricks of the underlying platforms and provides a unified API for the generated code. This layer is a very thin one: it only supports those API primitives which can be trivially mapped to the underlying platform. The supported subset of the API and its performance properties (Platform Profile) are the most important characteristics of the platform in the component model. Currently, the adoption layer supports event triggered tasks, event triggered message channels and periodically or periodically scheduled time triggered tasks and message channels. The orthogonal and symmetrical classification of event and (perdiocal or aperiodical) time triggered tasks and messaging and their well defined compositions in the real-time target is the most important result so far in this area of our work.

The tool chain supports the RTAI Linux environment and the TTP/C platform. We plan to add new execution platforms in the future, but the real strength of the tool chain lies in its support to add new potential platforms with deep understanding of its semantics and quantitative properties.

The tool chain is implemented in C++ and is heavily built on our existing modeling infrastructure. The GME tool provides a robust but flexible meta-programmable visual modeling environment, graph transformations (SL CodeGen, SF CodeGen) are implemented with the GReAT graph transformation engine, several of our interpreters processes model contents through domain specific APIs provided by the UDM tool suite. Currently the tool chain elements are command line programs, which provides a batch processing interface. Later, we would like to develop a graphical interface shell which encapsulates these tools and provides an IDE-like environment.

3.4 Testing and Experimental Validation (Tomlin, Sastry)

Testing and experimental validation of the design flow will be accomplished on a single UAV and multiple UAV platforms, with search, surveillance, and collision avoidance functions. The baseline embedded software and control for these platforms has been completed using "traditional" approaches of point-based designs. We will incrementally migrate the baseline controller design to the emerging model-based design tool suite.

4. Personnel Supported

Vanderbilt:

1. Professor Janos Sztipanovits (PI)
2. Professor Gabor Karsai
3. Peter Volgyesi (research scientist)
4. Joe Porter (Graduate Student, funded by this contract)
5. Ryan Thibodeaux (Graduate Student, funded elsewhere)

Associated but not supported:

1. Himansu Neema (senior engineer)
2. Sandeep Neema (senior research scientist)
3. Harmon Nine (senior engineer)

Berkeley:

1. Professor Claire Tomlin
2. Professor Edward A. Lee

3. Professor Shankar Sastry
4. Humberto Gonzales (Graduate Student, funded by this contract)
5. Gabe Hoffmann (Graduate Student at Stanford, funded elsewhere)
6. Gang Zhou (Graduate Student, funded by this contract)
7. Man-kit (Jackie) Leung (Graduate Student, funded elsewhere)
8. Christopher Brooks (Staff Programmer/Analyst, funded 25% starting May 1, 2007)
9. Jonathan Sprinkle (Research Associate, funded elsewhere)

<u>CMU</u>

1. Professor Bruce Krogh
2. Professor Edmund Clarke
3. Himanshu Jain, PhD candidate, Computer Science Dept., CMU
4. Flavio Lerda, PhD candidate, Computer Science Dept., CMU

Associated but not supported:

5. Sumit Jha, PhD candidate, Computer Science Dept., CMU
6. Stephen Magill, PhD candidate, Computer Science Dept., CMU
7. Bryant Lee, PhD candidate, Computer Science Dept., CMU
8. Nishant Sinha, PhD candidate, Computer Science Dept., CMU
9. Constantios Bartzis, Post Doc, Computer Science Dept., CMU
10. Tamir Heyman, Post Doc, Computer Science Dept., CMU
11. Azideh Farzan, Post Doc, Computer Science Dept., CMU
12. Ingo Feinerer, Visiting Researcher, Computer Science Dept., CMU
13. Stacey Ivol, PhD candidate, Dept. of ECE, CMU
14. Hitashyam Maka, PhD candidate, Dept. of ECE, CMU
15. Ajinkya Y. Bhave, PhD candidate, Dept. of ECE, CMU
16. James Weimer, PhD candidate, Dept. of ECE, CMU

<u>Stanford</u>

1. Professor Stephen P. Boyd,
2. Jo¨elle Skaf, Ph.D. Candidate
3. Siddharth Joshi, Ph.D. Candidate
4. Almir Mutapcic, Ph.D. Candidate
5. Seung Jean Kim, Consulting Professor

5. Publications

1. Gang Zhou, Man-Kit Leung, and Edward A. Lee, "A Code Generation Framework for Actor-Oriented Models with Partial Evaluation", in Proceedings of International Conference on Embedded Software and Systems 2007, LNCS 4523,

pp. 786-799, Daegu, South Korea, May 14-16, 2007, Y.-H. Lee et al. (Eds.) http://ptolemy.eecs.berkeley.edu/publications/papers/07/codegen/

2. Antoon Goderis, Christopher Brooks, lkay Altintas, Edward A. Lee, "Composing Different Models of Computation in Ptolemy II and Kepler," 2007 Proceedings, International Conference on Computational Science (ICCS), May, 2007; http://chess.eecs.berkeley.edu/pubs/193.html

3. Yang Zhao, Jie Liu and Edward A. Lee, "A Programming Model for Time-Synchronized Distributed Real-Time Systems", in Proceedings of the 13th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS 07), Bellevue, WA, United States, April 3-6, 2007. http://ptolemy.eecs.berkeley.edu/publications/papers/07/RTAS/

4. Ye Zhou and Edward A. Lee. " A Causality Interface for Deadlock Analysis in Dataflow," In Proceedings of the 6th ACM & IEEE Conference on Embedded Software (EMSOFT '06), Seoul, Korea, October 22-25, 2006. http://ptolemy.eecs.berkeley.edu/publications/papers/06/causalityInterfaces_EMSOFT06/

5. Edward A. Lee. "Concurrent Semantics without the Notions of State or State Transitions," in Proceedings of the International Conference on Formal Modeling and Analysis of Timed Systems (FORMATS), Paris, (LNCS 4202, Springer-Verlag, E. Asarin and P. Bouyer, Eds.), September 25-27, 2006. http://ptolemy.eecs.berkeley.edu/publications/papers/06/concurrentsemantics/

6. Xiaojun Liu, Eleftherios Matsikoudis, and Edward A. Lee. " Modeling Timed Concurrent Systems," in Proceedings of the 17th International Conference on Concurrency Theory (CONCUR), Bonn, Germany, August 27-30, C. Baier and H. Hermanns (Eds.), LNCS 4137, Springer-Verlag, pp. 1-15, 2006. http://ptolemy.eecs.berkeley.edu/publications/papers/06/timedconcurrentsystems/

7. J. Mikael Eklund, Jonathan Sprinkle, Todd Templeton, S. Shankar Satry, "Transitioning Intelligence to Embedded Platforms", AVT-146 Symposium on "Platform Innovations and System Integration for Unmanned Air, Land and Sea Vehicles", series in Applied Vehicle Technology, NATO, (in publication),Florence, Italy, May., 14--17, 2007.

8. G. M. Hoffmann, S. L. Waslander, and C. J. Tomlin, "STARMAC: The Stanford Testbed of Autonomous Rotorcraft for Multiple Agent Control", in preparation for submission to the AIAA Journal of Guidance, Control, and Dynamics.

9. The paper "Controller Coefficient Truncation Using Lyapunov Performance Certificate" was submitted to IEEE Transactions on Automatic Control in December 2006. It is currently under review. A shorter version of the paper will appear in Proceedings of the European Control Conference, July 2007.

10. A. Mutapcic, S.-J. Kim, and S. Boyd, "Beamforming with uncertain weights", published in IEEE Signal Processing Letters, May 2007.

11. L. Vandenberghe, S. Boyd, and K. Comanor, "Generalized Chebyshev bounds via semi definite programming", published in SIAM Review, March 2007.

12. L. Xiao, S.Boyd, and S.-J. Kim, "Distributed average consensus with least-mean-square deviation" published in Journal of Parallel and Distributed Computing, 2007.

13. S.-J. Kim, K. Koh, S. Boyd, and D. Gorinesvky, "ℓ1 trend filtering", submitted to SIAM Review, problems and techniques section, May 2007.

14. S. Boyd, S.-J. Kim, L. Vandenberghe, and A. Hassibi, "A tutorial on geometric programming", published online in Optimization and Engineering, Springer, April 2007.

15. R. Panicker, J. Kahn, and S. Boyd, "Compensation of multimode fiber dispersion using adaptive optics via convex optimization", submitted to IEEE Journal of Lightwave Technology, May 2007.

16. S. Samar, S. Boyd, and D. Gorinevsky, "Distributed estimation via dual decomposition", to appear in Proceedings of European Control Conference, Kos, Greece, July 2007.

17. A. Mutapcic, S. Boyd, A. Farjadpour, S. Johnson, and Y. Avniel, "Robust design of slow-light tapers in photonic crystals", submitted to SIAM Journal on Optimization, December 2006.

18. S. Joshi and S. Boyd, "An efficient method for large-scale gate sizing" submitted to IEEE Transactions on Circuits and Systems I, December 2006.

19. S.-J. Kim and S. Boyd, "A minimax theorem with applications to machine learning, signal processing, and finance", submitted to SIAM Journal on Optimization, December 2006.

20. A. Ben Tal, S. Boyd, and A. Nemirovski, "Extending scope of robust optimization", published in Math. Programming, Series B, February 2006.

21. S. Boyd and B. Wegbreit, "Fast computation of optimal contact forces", submitted to IEEE Trans. Robotics, December 2006.

22. S.-J. Kim, A. Magnani, A. Mutapcic, S. Boyd, and Z.-Q. Luo, "Robust beam forming via worst-case SINR maximization", submitted to IEEE Transactions on Signal Processing, December 2006.

23. A. Ghosh and S. Boyd), “Upper bounds on algebraic connectivity via convex optimization”, published in Linear Algebra and its Applications, October 2006.

24. J. Sun, S. Boyd, L. Xiao, and P. Diaconis, “The fastest mixing Markov process on a graph and a connection to a maximum variance unfolding problem”, published in SIAM Review, problems and techniques sections, November 2006.

25. K. Koh, S.-J. Kim, and S. Boyd, “An interior-point method for large-scale ℓ1-regularized logistic regression”, to appear in Journal of Machine Learning Research, 2007.

26. S. Samar, D. Gorinevsky, and S. Boyd, “Embedded estimation of fault parameters in an unmanned aerial vehicle” published in Proceedings of IEEE International Conference on Control Applications, October 2006.

27. M. S. Lobo, M. Fazel, and S. Boyd, “Portfolio optimization with linear and fixed transaction costs”, to appear, Annals of Operations Research, special issue on financial optimization, December 2006.

28. A. Mutapcic, S.-J. Kim, and S. Boyd, “Array signal processing with robust rejection constraints via second-order cone programming”, published in Proceedings of the 40th Asilomar Conference on Signals, Systems, and Computers (ACSSC’06), October-November 2006.

29. A. Ghosh and S. Boyd, ”Growing well-connected graphs”, published in Proc. 45th IEEE Conference on Decision and Control (CDC), December 2006.

30. Sumit Jha, Bruce Krogh, James Weimer and Edmund Clarke , “Reachability for Linear Hybrid Automata Using Iterative Relaxation Abstraction” in Proceedings of HSCC 2007.

31. André Platzer and Edmund M. Clarke, “The Image Computation Problem in Hybrid Systems Model Checking”, in Proceedings of HSCC 2007.

32. Edmund M. Clarke, Flavio Lerda, and Muralidhar Talupur, “An Abstraction Technique for Real-Time Verification”, in Proceedings of the GM R&D Workshop on Next Generation Design and Verification Methodologies for Distributed Embedded Control System. 2007. (Invited Paper)

33. Xuandong Li, Sumit Jha, Lei Bu, “A Tool for Bounded Model Checking of Linear Hybrid Automata”, in Proceedings of BMC 2006 (Electronic Notes in Theoretical Computer Science)

34. Edmund Clarke and Flavio Lerda, “Model Checking: Software and Beyond”, published in Computer Society of India Communications. Vol. 31, Issue 2, May 2007.

35. Kai Chen, Janos Sztipanovits and Sandeep Neema, “Compositional Specification of Behavioral Semantics,” in Proceedings of DATE 2007, Nice, France

36. Ethan Jackson and Janos Sztipanovits, “Constructive Techniques for Meta- and Model-level Reasoning,” Submitted paper

37. Ethan Jackson and Janos Sztipanovits, “Models as Structures: The Structural Semantics of Model-based Design,” Submitted paper

38. Narayanan, G. Karsai : Using Semantic Anchoring to Verify Behavior Preservation in Graph Transformations, Electronic Communications of the EASST, Volume 4, 2006: Graph and Model Transformation 2006, available from http://eceasst.cs.tu-berlin.de/index.php/eceasst

39. G. Karsai, A. Narayanan: On the Correctness of Model Transformations in the Development of Embedded Systems, Proceedings of the 2006 Monterey Workshop, Paris, France. Paper version submitted for publication in an upcoming LNCS volume.

40. Aditya Agrawal, Gabor Karsai, Sandeep Neema, Feng Shi, Attila Vizhanyo: “The Design of a Language for Model Transformations”, Journal on Software and System Modeling, pp 261-288, Volume 5, Number 3 / September, 2006.

41. D. Balasubramanian, A. Narayanan, S. Neema, F. Shi, R. Thibodeaux, G. Karsai: A Sub-graph Operator for Graph Transformation Languages, accepted for presentation at 6th International Workshop on Graph Transformation and Visual Modeling Techniques, March 31 - April 1, 2007, Braga, Portugal.

6. Interactions/Transitions

6.1 Participation/presentations at meetings, conferences, seminars

1. MURI team attended the bi-weekly MURI telcons.

2. HCDDES Planning, June 8, 2006, Berkeley.
   Christopher Brooks presented "Timeliness, with a little Scalability"
   Claire Tomlin presented "Embedded software analysis and design for aircraft systems"
   Gabor Karsai presented “Tool Integration Frameworkls”
   Janos Sztipanovits presented “Project Overview”
   Stephen Boyd presented “Robust Control Design”
   Bruce Krogh presented “Hybrid Systems Verification”

3. HCDDES Kickoff, August 7, 2006, Atlanta.
   Edmund Clarke presented “Source Code Verification and Testing”
   Edward Lee presented "Advanced Tool Architectures"
   Claire Tomlin presented "Hybrid and Embedded Systems and Control Theory"
   Gabor Karsai presented “Model-Based Software Design and Verification”
   Shankar Sastry presented “Testing and Experimental Tool Validation”

Janos Sztipanovits presented “Project Overview”

4. NASA Ames Research Center, November 2006
   Claire Tomlin - Invited by the Director of Aeronautics and the Chief Scientist to give a talk to the entire Center.

5. CDC V&V workshop, December 2006, San Diego.
   Claire Tomlin presented "Hybrid and Embedded Systems and Control Theory"

6. Ptolemy Ptutorial, February 12, 2007, Berkeley.
   Gang Zhou and Man-kit (Jackie) Leung gave a 45 minute tutorial on how to develop Ptolemy codegen actors.

7. Seventh Biennial Ptolemy Miniconference, February 13, 2007, Berkeley.
   Gang Zhou presented the code generation work to 60 attendees
   http://ptolemy.eecs.berkeley.edu/conferences/07/index.htm

8. Technical Interchange Meeting, March 30, 2007, CMU.
   Gang Zhou presented an overview of the Ptolemy codegen effort.
   Christopher Brooks and Gang Zhou discussed Ptolemy execution semantics.

9. Ptolemy/Starmac Meeting, March 9, 2007, Berkeley.
   Professor Tomlin, Professor Lee, Gabe Hoffman, Gang Zhou, Man-kit Leung and Christopher Brooks met. Gabe Hoffman presented "STARMAC 2 Hardware and Software Design"

10. Professor Boyd and Jo¨elle Skaf will be presenting the results mentioned above at the European Control Conference in Kos, Greece (July 2-5, 2007).

11. The 2006 Federated Logic Conference
    Edmund M. Clarke, Himanshu Jain, Sumit Jha: A Tool for Bounded Model Checking of Linear Hybrid Automata.

12. GM R&D Workshop on Next Generation Design and Verification Methodologies for Distributed Embedded Control System. 2007
    Edmund M. Clarke: An Abstraction Technique for Real-Time Verification.

13. International Conference on Hybrid Systems Computation and Control 2007
    Bruce H Krogh, Edmund C. Clarke: Reach ability for Linear Hybrid Automata Using Iterative Relaxation Abstraction , The Image Computation Problem in Hybrid Systems Model Checking.

14. Ninth International Conference on Theory and Applications of Satisfiability Testing (SAT 2006).
    Himanshu Jain and Edmund M. Clarke: Satisfiability Checking of Non-clausal Formulas using General Matings.

15. 18th Computer-aided Verification (CAV 2006).
    Himanshu Jain and Edmund M. Clarke: Using Statically Computed Invariants inside the Predicate Abstraction and Refinement Loop.

16. ESMD Software Workshop (ExSoft, Nasa Exploration Program), April 10, 2007, Houston TX
    Janos Sztipanovits keynote presentation "Model-Based Software Development"

17. Safe and Secure Systems Airworthiness Certification R&D Workshop, September 12-14, 2006, Fairborn, OH (AFRL Workshop)
    Janos Sztipanovits and Jonathan Sprinkle presented "Frameworks and Tools for High-Confidence Design of Adaptive, Distributed Embedded Control Systems"

18. IEEE International Conference on Engineering of Computer Based Systems (ECBS 2007) March 26, 2007, Tucson, AZJanos Sztipanovits keynote presentation on Cyber Physical Systems

19. Workshop on Event-based Semantics, RTAS 2007 13th IEEE Real-Time and Embedded Technology and Applications Symposium, April 3 2007, Bellevue, WA
    Janos Sztipanovits presented "Defining Behavioral Semantics for Domain Specific Modeling Languages"

20. Workshop on Foundations and Applications of Component-Based Design, EMSOFT 2006, October 26, 2006, Seoul, South Korea
    Janos Sztipanovits presented "Towards Compositional Specification of Behavioral Semantics"

21. Workshop on Towards a Systematic Approach to Embedded System Design, DATE 2007, April 20, 2007, Nice, France
    Janos Sztipanovits keynote on "Towards Systematic Model Based Development of Embedded Systems – Requirements and Challenges"

22. Vanderbilt hosted Professor Hermann Kopetz on May 23, 2007 regarding TTP-C applications.

6.2 Consultative and advisory functions to other laboratories and agencies, especially Air Force and other DoD laboratories. Provide factual information about the subject matter, institutions, locations, dates, and names(s) of principal individuals involved

1. Shankar Sastry was a member of the AF Scientific Advisory Board (9/03-9/06) responsible for reviews of information technology programs in the Air Force.
2. Janos Sztipanovits serves as member of the AF Scientific Advisory Board.

6.3 Technology Assists, Transitions, and Transfers.

1. Ptolemy II 6.0 includes an update of HyVisual, the Hybrid Systems modeling tool. In addition we have:
    a. Assisted in the transfer of avionics code from Berkeley to the HCDDES team
    b. Provided consultation and research materials about the IEEE-1588 platform as a possible testbed
    c. Prototyped a vhdl target for the code generation effort
    d. Researched Hybrid Interchange formats and discussed these with researchers
    e. in Alberto Sangiovanni-Vincentelli's group and at Cadence Berkeley Labs
    f. Discussed the design of Vanderbilt's code generation effort.
2. Ptolemy II 6.0 was released on February 13, 2007. Ptolemy II includes the code generation facility. The Ptolemy source tree is available via CVS. We are actively working with Bosch and National Instruments
3. Vanderbilt's MIC tool suite (GME, GReAT, UDM, OTIF) has two major releases during the last year. The releases are available through the ESCHER and ISIS download sites
4. Vanderbilt continued working with GM, Raytheon and BAE Systems research groups on transitioning model-based design technologies into programs.
5. Vanderbilt continued working with Boeing's FCS program on applying the MIC tools for precise architecture modeling and systems integration

6.4 New discoveries, inventions, or patent disclosures.

None.

6.5 Honors and Awards

1. Claire Tomlin MacArthur Foundation Fellowship, 2006
2. Claire Tomlin Okawa Foundation Award, 2006
3. Edward Lee was named to the Robert S. Pepper Distinguished Professorship in Electrical Engineering & Computer Sciences
4. Janos Sztipanovits Elected to US Air Force Scientific Advisory Board 2006-2010
5. Stephen Boyd was granted an Honorary doctorate from the Royal Institute of Technology (KTH), Stockholm, in November 2006. The ceremony took place in Stockholm City Hall, and involved the firing of canons with the installation of each of the 5 new honorary doctors.
6. Stephen Boyd was appointed Visiting Professor at Harbin Institute of Technology, Harbin, China.
7. Stephen Boyd with student S. Samar and colleague D. Gorinevsky, he was awarded the Best student paper award, 2006 IEEE International Conference on Control Applications, for "Embedded Estimation of Fault Parameters in an Unmanned Aerial Vehicle,"
8. Professsor Boyd gave or was invited to give a number of plenary, keynote, and other

distingushed lectures:

- 2007 Plenary lecture, WSEAS Conference on System Science and Simulation in Engineering, Venice.
- 2007 Simon Stevin Lecture, Katholieke Universiteit Leuven.
- 2007 DYSCO (Dynamic systems, control, and optimization) lecture, Louvain la Neuve.
- 2006 Linneaus Center Distinguished Speaker Series, KTH, Stockholm. International Congress of Mathematicians, Madrid.
- 2006 Invited section lecture, International Congress of Mathematicians, Madrid.
- 2006 Zaborszky distinguished lecture series, Washington University, St. Louis.
- 2006 Plenary lecture, International Symposium on Nonlinear Theory and its Applications (NOLTA), Bologna.
- 2006 Plenary lecture, 25th Chinese Control Conference, Harbin.