ANNUAL REPORT

FOUNDATIONS OF HYBRID AND EMBEDDED SYSTEMS AND SOFTWARE

NSF/ITR PROJECT – AWARD NUMBER: CCR-00225610

UNIVERSITY OF CALIFORNIA AT BERKELEY
VANDERBILT UNIVERSITY
UNIVERSITY OF MEMPHIS

MAY 31, 2005

PERIOD OF PERFORMANCE COVERED: JUNE 1, 2004 – MAY 31, 2005

Contents

1. Participants

1.1. People

PRINCIPAL INVESTIGATORS:

THOMAS HENZINGER, (UC BERKELEY, EECS)
EDWARD A. LEE, (UC BERKELEY, EECS)
ALBERTO SANGIOVANNI-VINCENTELLI, (UC BERKELEY, EECS)
SHANKAR SASTRY, (UC BERKELEY, EECS)
JANOS SZTIPANOVITS, (VANDERBILT, ELECTRICAL AND COMPUTER ENGINEERING)

FACULTY INVESTIGATORS:

ALEX AIKEN, (UC BERKELEY, CS)
RUZENA BAJCSY, (UC BERKELEY, EECS)
GAUTAM BISWAS, (VANDERBILT, CS)
RASTISLAV BODIK, (UC BERKELEY, EECS)
BELLA BOLLOBAS, (UNIVERSITY OF MEMPHIS, MATHEMATICS)
JEROME A. FELDMAN (UC BERKELEY, EECS)
KENNETH FRAMPTON, (VANDERBILT, ME)
J. KARL HEDRICK, (UC BERKELEY, ME)
GABOR KARSAI, (VANDERBILT, ELECTRICAL AND COMPUTER ENGINEERING)
KURT KEUTZER, (UC BERKELEY, EECS)
WAGDY H. MAHMOUD (TENNESSEE TECH. UNIVERSITY)
GEORGE NECULA, (UC BERKELEY, EECS)
SRINI RAMASWAMY (TENNESSEE TECH. UNIVERSITY)
PRAVIN VARAIYA, (UC BERKELEY, EECS)

POST DOCTORAL RESEARCHERS:

MASSIMO FRANCESCHETTI (UC BERKELEY)
CHRISTOPH KIRSH (UC BERKELEY)
CLAUDIO PINELLO (UC BERKELEY)
MARCO SANVIDO (UC BERKELEY)
JONATHAN SPRINKLE (UC BERKELEY)

GRADUATE STUDENTS:

ALESSANDRO ABATE (UC BERKELEY)
AARON AMES (UC BERKELEY)
SAURABH AMIN (UC BERKELEY)
DANIEL BALASUBRAMANIAN (VANDERBILT)

LUCA CARLONI (UC BERKELEY)
ADAM CATALDO (UC BERKELEY)
DENNIS CHANG (UC BERKELEY)
KAI CHEN (VANDEREBILT)
ELAINE CHEONG (UC BERKELEY)
ABHIJIT DAVARE (UC BERKELEY)
DOUG DENSMORE (UC BERKELEY)
ANDREW D. DIXON (VANDERBILT)
BRANDON EAMES (VANDERBILT)
MATTHEW J. EMERSON (VANDERBILT)
HUINING FENG (UC BERKELEY)
JOYTI GANDHE (VANDERBILT)
SUMITRA GANESH (UC BERKELEY)
CSABA GARAY (VANDERBILT)
MATTHEW HARREN (UC BERKELEY)
ETHAN JACKSON (VANDERBILT)
FARINAZ KOUSHANFAR (UC BERKELEY)
NARAYANAN KRISHNAN (UC BERKELEY)
ALEXANDER KURZHANSKIY(UC BERKELEY)
JONGHO LEE (UC BERKELEY)
XIAOJUN LIU (UC BERKELEY)
GABOR MADL (VANDERBILT)
DAVID P. MANDELIN (UC BERKELEY)
ELEFTERIOUS MATSIKOUDIS (UC BERKELEY)
MARK MCKELVIN (UC BERKELEY)
BHARATHWAJ MUTHUSWAMY (UC BERKELEY)
STEPHEN NEUENDORFFER (UC BERKELEY)
SONGHWAI OH (UC BERKELEY)
WILLIAM PLISHKER (UC BERKELEY)
KAUSHIK RAVINDRAN (UC BERKELEY)
PANNAG SANKETI (UC BERKELEY)
PETER SCHMIDT (VANDERBILT)
TIVADAR SZEMETHY (VANDERBILT)
GUANG YANG (UC BERKELEY)
HAIBO ZENG (UC BERKELEY)
YANG ZHAO (UC BERKELEY)
HAIYANG ZHENG (UC BERKELEY)
WEI ZHENG (UC BERKELEY)
GANG ZHOU (UC BERKELEY)
YE ZHOU (UC BERKELEY)

QI ZHOU (UC BERKELEY)

UNDERGRADUATE STUDENTS:

PHILLIP BALDWIN (UC BERKELEY)
CHRIS BEERS (VANDERBILT)
TREVOR BROWN (VANDERBILT)
COLIN COCHRAN (UC BERKELEY)
NICKOLIA COOMBS (VANDERBILT)
RACHAEL DENNISON (VANDERBILT)
BASIL ETEFIA (UC BERKELEY)
ELIZABETH FATUSIN (UC BERKELEY)
DAVID GARCIA (VANDERBILT)
RAFAEL GARCIA (UC BERKELEY)
SHANTEL HIGGINS (VANDERBILT)
MARY HILLIARD (VANDERBILT)
IYIBO JACK (UC BERKELEY)
JOHN KILBY (VANDERBILT)
SHIRLEY (XUE) LI (VANDERBILT)
PRAVEEN MUDINDI (VANDERBILT)
ANTONIO YORDAN-NONES (UC BERKELEY)
EFOSA OJOMO (VANDERBILT)
MIKE OKYERE (UC BERKELEY)
JAMESON PORTER (VANDERBILT)
RAKESH REDDY (UC BERKELEY)
MICHAEL RIVERA-JACKSON (VANDERBILT)
ISMAEL SARMIENTO (UC BERKELEY)
BINA SHAH (VANDERBILT)
SINITHRO TAVERAS (VANDERBILT)
EDWIN VARGAS (VANDERBILT)
TIRONE VINCENT (VANDERBILT)
JOHN WILLIAMSON (VANDERBILT)

TECHNICAL STAFF, PROGRAMMERS:

CHRISTOPHER HYLANDS BROOKS (UC BERKELEY)
NATHAN JEW (UC BERKELEY)
BRADLEY A. KREBS (UC BERKELEY)
PHILLIP LOARIE (UC BERKELEY)
MARVIN MOTLEY (UC BERKELEY)
GUNNAR PROPPE (UC BERKELEY)
MARY STEWART (UC BERKELEY)

AARON WALBURG (UC BERKELEY)
BRIAN WILLIAMS (VANDERBILT)

BUSINESS ADMINISTRATORS:
SUSAN B. GARDNER (UC BERKELEY)
ROBERT BOXIE (VANDERBILT, SIPHER COORDINATOR)

1.2. Partner Organizations

- University of California at Berkeley
- Vanderbilt University
- University of Memphis

1.3. Collaborators

- Albert Benveniste (IRISA INRIA, Rennes)
- Hermann Kopetz (Technical University of Vienna, Austria)
- Manfred Morari (ETH, Zurich, Switzerland)
- Gabor Peceli (Technical University of Budapest, Hungary)
- Joseph Sifakis (CNRS VERIMAG, Grenoble, France)
- Kim Larsen (University of Aalborg, Aalborg, Denmark)
- Henrik Christensen (Royal Institute of Technology, Stockholm, Sweden)

2. Activities and Findings

2.1. Project Activities

This is the second Annual Report for the NSF Large ITR on "Foundations of Hybrid and Embedded Systems and Software". This research activity is primarily organized through a Center at Berkeley CHESS (the Center for Hybrid and Embedded Systems and Software, http://chess.eecs.berkeley.edu), the Vanderbilt ISIS (Institute for Software Integrated Systems, http://www.isis.vanderbilt.edu), and the Department of Mathematical Sciences, (http://msci.memphis.edu) at Memphis.

The web address for the overall ITR project is:

http://chess.eecs.berkeley.edu/projects/ITR/main.htm

This web site has links to the proposal and statement of work for the project.

Main events for the ITR project in its third year were:

1. NSF Onsite Review, November 18th, 2004, UC Berkeley. The program and the presentations are available at http://chess.eecs.berkeley.edu/conferences/04/NSFonsiteReview11-04
2. The annual Chess Review for our industrial partners, advisory board members, and friends of the project. The program and presentations are available at http://chess.eecs.berkeley.edu/conferences/04/05/index.htm
3. A weekly Chess workshop was held at Berkeley. Presentations for the workshop are available at http://chess.eecs.berkeley.edu/workshop.htm.

We organize this section by thrust areas that we established in the statement of work.

2.1.1. Hybrid Systems Theory

We have proposed to build the theory of mixed discrete and continuous hybrid systems into a mathematical foundation of embedded software systems. For this purpose we have been pursuing four directions:

1. We have been designing models of computation that permit the composition of non-functional properties. While previously we had focused on real-time and resource-constrained systems, in the past year we developed a general theory of composing quantitative aspects of systems. We also formalized composition as a non-zero-sum game where the players (components) have different objectives.

2. We have been designing robust models of computation, where small perturbations of the system description cause only small changes in the system behavior. Previously we had identified discounting as a paradigm for achieving robustness in discrete and hybrid models, and in the past year we developed model checking algorithms for discounted

properties. We also studied affine hybrid systems as an approach to robust modeling. We have also been studying different functorial representations of hybrid systems with a view to characterizing Zeno properties of hybrid systems.

3. We have been developing and evaluating several methods for the computational treatment of hybrid systems. In particular, in the past year we designed and implemented a deterministic operational semantics for the simulation of hybrid systems, as well as ellipsoid-based algorithms for the efficient reach-set analysis of hybrid systems.

4. We have been developing stochastic models that combine hybrid dynamics with sources of uncertainty. For controlling such stochastic systems, we improved the best known algorithms for solving stochastic games. We also pursued the application of stochastic hybrid models in systems biology and for other classes of small noise perturbation of deterministic hybrid systems.

2.1.1.a. Deep Compositionality

Non-Zero-Sum Games as Compositional System Models

The components of a complex system can be viewed as players in a game with different objectives. Each component attempts to satisfy its own specification, but the specifications of different components may be neither identical nor complementary; so the right formalization is that of a non-zero-sum games. Classically, Nash equilibria capture the notion of rationality for such games. We argue, however, that for achieving compositionality in system design, a special kind of Nash equilibrium is appropriate. We study infinite stochastic games played by two-players on a finite graph with goals specified by sets of infinite traces. The games are concurrent (each player simultaneously and independently chooses an action at each round), stochastic (the next state is determined by a probability distribution depending on the current state and the chosen actions), infinite (the game continues for an infinite number of rounds), nonzero-sum (the players' goals are not necessarily conflicting), and undiscounted. We show that if each player has an ω-regular objective expressed as a parity objective, then there exists an ϵ-Nash equilibrium, for every ε >0. However, exact Nash equilibria need not exist. We study the complexity of finding values (payoff profile) of an ε-Nash equilibrium. This work is reported in [23][24][25][26].

Hybrid Systems Modeling Tools: a Survey

Las year in a collaborative project with the European Columbus project, we evaluated a set of tools, languages and formalisms for the simulation, verification and specification of hybrid systems. In this survey we evaluated the following languages and tools: Simulink, Modelica, HyVisual, Scicos, Charon, CheckMate, Masaccio, SHIFT, HSIF, Metropolis and Hysdel. This year we continued this work with a sets of surveys by Lee on the state of embedded systems, formal versus informal methods and concurrent models of computation for embedded software. These survey results are reported in [49][50][51].

.

2.1.1.b. Robust Hybrid Systems

Design and Verification of Robust System Models

In previous work, we had identified discounting as a mechanism for moving from a discrete, brittle paradigm of boolean-valued property satisfaction to a continuous, robust paradigm of real-valued property estimation. In this past year, we developed algorithms for computing the real value of discounted properties expressed in temporal logic over state transition systems. This work is reported in [25].

Affine Hybrid Systems

Affine hybrid systems are hybrid systems where the discrete domains are affine sets, and the transition maps between discrete domains are affine transformations. This definition differs from other definitions of hybrid systems that have been proposed, but the underlying ideas involved in the definition of affine hybrid systems have been seen in the literature. We give a formal framework to these ideas. Affine hybrid systems are simple, and it is this simplicity that allows us to say some useful things about them. The structure of affine hybrid systems contains a wealth of intrinsic information. Affine sets can be described in terms of matrix inequalities, and affine transformations are characterized by elements of SE(n). In the paper [10], we use the geometric information intrinsic in affine hybrid systems to develop the idea of *spatial equivalence* between an affine hybrid system **H** and an affine hybrid system **G.** More specifically we give conditions for what is known as blowing up affine hybrid systems.

A Homology Theory for Hybrid Systems

Using a categorical framework in [12] we have developed a homology theory for hybrid systems. This theory enables us to characterize some intrinsic properties of the hybrid systems structure, include whether or not it admits of Zeno behavior [13] and it allows for the constructive methods of algebraic topology to give full characterizations of when hybrid systems do not admit of solutions on the infinite time horizon in [11].

2.1.1.c. Computational Hybrid Systems

Algorithms for the Control of Stochastic Systems

The problem of designing a controller can be viewed as the problem of finding a winning strategy of the control player in a game against the plant player. We improved on the best known algorithms for finding such strategies in the case that there is also a probabilistic source of uncertainty in the game. This work is reported in [69].

Reach Set Calculations using Ellipsoidal Approximations

Linear dynamic systems are considered. We studied the application of the external ellipsoidal approximations of the reach sets to the internal point problem. An algorithm was developed that allows us to calculate control and initial state that bring the system to the given point at the given

time. A closed-form solution is obtained, or else, if the target point is unreachable, then the closest reachable point is found.The ellipsoidal toolbox provides the implementation of the ellipsoidal methods used for the computation of the reach sets. It includes external and internal ellipsoidal approximation algorithms as well as visualization routines. The API allows the user to run the algorithms with given precision. Most recently this work has been continued to allow for reachability approximations for hybrid systems. The work by Kurzhanskiy and Varaiya is in progress.

A Deterministic Operational Semantics for Hybrid System Simulations

We have developed a deterministic operational semantics for hybrid system simulations. We have implemented this semantics in HyVisual, a domain-specific hybrid system modeling framework built on Ptolemy II. HyVisual includes a simulator that gives a well-defined execution by removing unnecessary non-deterministic behaviors when dealing with the discontinuities of piecewise continuous signals and when dealing with simultaneous discrete events.

We interpret the piecewise continuous signals as a set of continuous signals and discrete events, and the discontinuities as the effects of discrete events. In particular, we developed a mechanism to accurately detect the time point when a state transition is enabled and force the transition to take place immediately. By this mechanism, an enabled transition is treated as a discrete event. Multiple discrete events can occur at the same time point in a model, but the semantics gives them a well-defined ordering. For example, when a transient state is entered, where incoming transitions and outgoing transitions are enabled simultaneously, two ordered discrete events with the same stamp represent the the transition in and the transition out.

Based on this signal interpretation, a simulation of hybrid system models is divided into two kinds of interleaved execution phases: a continuous phase and a discrete phase. Each phase uses its own fix-point semantics to find the model's behavior. In particular, the fix point of the discrete phase of execution is reached only when all events at the current time have been handled.

We resolved a few subtleties with this operational semantics, including ways to generate piecewise continuous signals, operations on continuous-time signals with discontinuities such as sampling and level-crossing detection, and execution of transitions between transient states.

We have developed a formal model of this operational semantics, studying its relationship with those of synchronous languages and discrete-event languages, and unifying these into a general operational semantics for executing heterogeneous models. This work is available in [17]

2.1.1.d. Stochastic Hybrid Systems

Stochastic hybrid systems are a natural extension of the deterministic counterpart. We also develop strategies for characterizing the stability of stochastic hybrid systems in [5]. Additionally we have used stochastic hybrid systems as approximations to simulation models of deterministic hybrid systems with non-deterministic switching [1].

Application of Stochastic Hybrid Systems to Communication Networks.

In [2][3] the objective of is to use the theory of stochastic hybrid systems to introduce two original flow control schemes for wireless networks. The mathematical underpinnings lie on the recently-developed congestion control models for Transmission-Control-Protocol(TCP)-like schemes; more precisely, the model proposed by Kelly for the wired case is taken as a template, and properly extended to the more involved wireless setting. We introduce two ways to modify a part of the model; the first is through a static law, and the second via a dynamic one. In both cases, we prove the global stability of the schemes, and present a convergence rate study and a stochastic analysis.

Application of Stochastic Hybrid Systems in Systems Biology.

Last year, we applied the stochastic hybrid system to the modeling of genetic regulatory network. This modeling approach can demonstrate the inherent randomness in molecular interactions and protein production in a cell, phenomena that are observed in biological systems but are not realized adequately using conventional macroscopic modeling techniques. A stochastic hybrid system approach was used to model the genetic network regulating the biosynthesis of an antibiotic called subtilin in Bacillus subtilis. This year we have continued this work in determining the behavior of other more complex mechanisms for regulation of other signaling mechanisms and have begun the study of what may be called stochastic resonance in biological systems.

2.1.2. Model-Based Design

Model-based design focuses on the formal representation, composition, and manipulation of models during the design process. It addresses system specification, model transformation, synthesis of implementations, model analysis and validation, execution, and design evolution. The semantic frameworks in which these models are applied may be domain-specific, offering embedded system designers methods and syntaxes that are closer to their application domain. The project team started off with three different notions for embedded system and software design: platform-based design (developed by Sangiovanni-Vincentelli's group), actor-based design (investigated by Lee's group) and model-based design (advocated by Sztipanovits' group). These approaches emphasize different, complementary aspects of the design process. Platform-based design focuses on the creation of abstraction layers in the design flow and investigates the semantic properties of mapping across these layers. Actor-based design investigates component interaction semantics and theories of composition on different layers of abstractions. Model-based design focuses on the specification and composition of domain-specific modeling languages (DSML-s) and model transformations via metamodeling. As a result of interaction among the research groups a synergistic view is emerging:

- Abstractions and design constraints play central role in the definition of platforms. DSML-s offer a formal way for capturing these abstractions and constraints in metamodels. The required semantic clarity for expressing the mapping across platforms challenges the DSML technology with the need of expanding the abstract syntax oriented metamodeling toward explicit representation of formal semantics.
- A core concept in actor-based design is component interaction semantics defined by models of computation (MoC). New results have been achieved in the semantic

foundations for heterogeneous systems, which will be the underpinning for the safe composition of heterogeneous reactive systems.

- Mapping across platforms has fundamental role in platform-based design flow. A new development in the technology of model transformations will contribute to the platform-based design vision. See for example the work reported in [78].

Our extensive experimental work on networked embedded systems (see applications and papers [39][43][59][68] have revealed new challenges in extending model-based design to distributed models of embedded systems. We have reached significant progress in developing a new theory for the design of this system category. Some recent new work in this area has been in the area of semantic anchoring of models, see for example [29].

2.1.2.a. Composition of Domain Specific Modeling Languages

Metamodeling

The modeling languages in which models are expressed are domain-specific, offering embedded system designers modeling constructs and syntax that are closer to their application domain. Domain-specific modeling languages (DSMLs) must capture the structural and behavioral aspects of embedded software and systems. Their semantics must emphasize concurrency, communication abstractions, temporal and other physical properties. For example, a DSML framework (i.e. a set of related modeling aspects) for embedded systems might represent physical processes using ordinary differential equations, signal processing using dataflow models, decision logic using finite-state machines, and resource management using synchronous models. The languages that are used for defining components of DSMLs are called *meta-languages* and the formal specifications of DSMLs are called *metamodels*. The specification of the abstract syntax of DSMLs requires a meta-language that can express concepts, relationships, and integrity constraints. The specification of the semantic domain and semantic mapping is more complicated, because models might have different interesting interpretations; therefore DSMLs might have several semantic domains and semantic mappings associated with them. For example, the *structural semantics* of a modeling language describes the meaning of the models in terms of the structure of model instances: all of the possible sets of components and their relationships, which are consistent with the well-formedness rules in defined by the abstract syntax. Accordingly, the semantic domain for structural semantics is defined by a *set-valued semantics*. The *behavioral semantics* may describe the evolution of the state of the modeled artifact along some time model. Hence, the behavioral semantics is formally captured by a mathematical framework representing the appropriate form of dynamics. See for example [44].

The specification of the abstract syntax of DSMLs requires a meta-language that can express concepts, relationships, and integrity constraints. In our work in Model-Integrated Computing (MIC), we first adopted UML class diagrams and the Object Constraint Language (OCL) as meta-language. This selection was consistent with UML's four layer meta-modeling architecture, which uses UML class diagrams and OCL as meta-language for the abstract syntax specification of UML. Last year, we have developed a MOF-based metamodeling approach and developed a new metamodeling environment using our GME tool suite. This year, our work on MIC (see [63]) has helped to clarify technical details on the Model Driven Architecture (MDA) concept of OMG and we have started up a meaningful interaction with the OMG MDA community. See

also [37][38]. Additional work that was begun this year is on doman specific visual languages for domain model evolution (see [76]).

Compositional Metamodeling

The GME-based metamodeling environment provides support for specifying DSML-s via metamodel composition. There are three characteristics of the GME that make it a valuable tool for the construction of domain-specific modeling environments. First, the GME provides generic modeling primitives that assist an environment designer in the specification of new graphical modeling environments. Second, these generic primitives are specialized to create the domain-specific modeling concepts through meta-modeling. The meta-models explicitly support composition enabling the creation of composite modeling languages supporting multiple paradigms. Third, several ideas from prototype-based programming languages have been integrated with the inherent model containment hierarchy, which gives the domain expert the ability to clone graphical models. Currently, we are exploring characteristics of the new MOF-based meta-modeling environment for DSML composition. See the work in [37][38] for the new results here.

Integration of Metamodeling with Hybrid Systems

We have developed a metamodel for the abstract syntax of the Hybrid System Interchange Format (HSIF). This work, which was part of our earlier projects, was lately extended with developing a translator between HSIF and Simulink/ Stateflow models This work helped us understanding the role of model transformations for defining semantics of DSML-s via the formal specification of translators. This work has been demonstrated successfully in a joint project with Northrup Grumann to successfully land UAVs (in a flight test in the Mojave Desert in July 2004), see [77].

Semantic Foundations for Heterogeneous Systems

Agent Algebra is a formal framework that can be used to uniformly present and reason about the characteristics and the properties of the different models of computation used in a design, and about their relationships. This is accomplished by defining an algebra that consists of a set of denotations, called agents, for the elements of a model, and of the main operations that the model provides to compose and to manipulate agents. Different models of computation are constructed as distinct instances of the algebra. However, the framework takes advantage of the common algebraic structure to derive results that apply to all models in the framework, and to relate different models using structure-preserving maps.

Relationships between different models of computation are described as conservative approximations and their inverses. A conservative approximation consists of two abstractions that provide different views of an agent in the form of an over- and a under-approximation. When used in combination, the two mappings are capable of preserving refinement verification results from a more abstract to a more concrete model, with the guarantee of no false positives. Conservative approximations and their inverses are also used as a generic tool to construct a correspondence between two models. Because this correspondence makes the correlation between an abstraction and the corresponding refinement precise, conservative approximations

are useful tools to study the interaction of agents that belong to heterogeneous models. A detailed comparison also reveals the necessary and sufficient conditions that must be satisfied for the well established notions of abstract interpretations and Galois connections (in fact, for a pair thereof) to form a conservative approximation. Conservative approximations are illustrated by several examples of formalization of models of computation of interest in the design of embedded systems.

While the framework of Agent Algebra is general enough to encompass a variety of models of computation, the common structure is sufficient to prove interesting results that apply to all models. In particular, we focus on the problem of characterizing the specification of a component of a system given the global specification for the system and the context surrounding the component. This technique, called Local Specification Synthesis, can be applied to solve synthesis and optimization problems in a number of different application areas. The results include sufficient conditions to be met by the definitions of system composition and system refinement for constructing such characterizations. The local specification synthesis technique is also demonstrated through its application to the problem of protocol conversion.

This work is reported in the recently completed PhD dissertation of Neuendorffer [64]. See also the survey paper of Lee [51] and two key reports of Lee and Neuendorffer and Zheng [54][55].

Generalized causality analysis

Causality properties of components in a hybrid system model are important to ensuring that a unique behavior is defined by the model. By introducing a functional dependency, which describes the causality relationship between the inputs and outputs of a component, we have developed a mechanism to analyze the causality properties of a hybrid system model without flattening the hierarchies. These causality properties guide execution of a simulator, ensuring deterministic behavior for deterministic models. Because the causality properties of a hybrid system may change dynamically due to the state change, our mechanism supports a dynamic re-calculation of the causality properties.

Dynamic re-calculation of causality properties can be costly and not practical for some applications. We are developing a static analysis mechanism that infers the common causality properties of a modal model from those of its modes. The result of the static analysis is conservative, but provides safety guarantees. One of our objectives is to analyze the tradeoffs between the conservative static analysis and the more costly run-time analysis. This work is surveyed in the papers [49], [50].

2.1.2.b. Extensions to Distributed Models of Embedded systems

Compositional Theory of Heterogeneous Reactive Systems

We have been working on a compositional theory of heterogeneous reactive systems in collaboration with A. Benveniste (INRIA), B. Caillaud (INRIA), and P. Caspi (VERIMAG). The approach is based on the concept of tags marking the events of the signals of a system. Tags can be used for multiple purposes from indexing evolution in time (time stamping) to expressing relations among signals like coordination (e.g., synchrony and asynchrony), and causal

dependencies. The theory provides flexibility in system modeling because it can be used both as a unifying mathematical framework to relate heterogeneous models of computations and as a formal vehicle to implement complex systems by combining heterogeneous components.

We have derived a set of theorems that support effective techniques to generate automatically correct-by-construction adaptors between designs formulated using different coordination paradigms [28]. We have applied these concepts to two scenarios that are of particular relevance for the design of embedded systems: the deployment of a synchronous design over a globally-asynchronous locally-synchronous (GALS) architecture and over a loosely time-triggered architecture (LTTA). The idea followed in these applications is to abstract away from the synchronous specifications the constraints among events of different signals due to the synchronous paradigm and, then, to map the unconstrained design into a different architecture characterized by a novel set of intended behavior of the system is retained. This is achieved by relying on a formal notion of semantics-preserving transformations that we developed based on the idea of morphisms over tag sets. We have introduced GALSC a language designed for programming event driven embedded systems such as sensor networks. GALSC implements the Tiny GALS programming model and has been implemented on the Berkeley motes and is compatible with the TinyOS/nesC component library.

2.1.2.c. Model Transformation

Meta Generator Technology

We have developed a language and a suite of supporting tools for the formal, high-level, yet executable specification of model transformations (GREAT (Graph Rewriting and Transformations) The language is based on graph transformation technology, where a complex model transformation is specified in terms of sequenced graph rewriting steps consisting of rewriting rules. This high-level specification facilitates not only the compact, formal representation of what a model translator should do, but also allows formal reasoning about the transformations. The semantics of GREAT has been formally defined in previous year's work and it comes with a set of supporting tools. See also some additional contributions this year in [37][38]. The tools suite includes an interpreter for GREAT ("Meta-Programmable Transformation Tool"), a code generator for compiling GREAT rules into C++, and a debugger (coupled with the interpreter). This year we have been using this to verify distributed real time properties of embedded systems in [58].

Pattern-Based Model Synthesis

We have introduced patterns in modeling on two levels. First, our meta-model composition technology enables to define abstract metamodeling constructs such as *templates* and *hierarchy* as language design patterns and compose those with DSML . Second, using the *template* construct, we are able to build large design spaces centered around domain architecture and automatically synthesize models that satisfy user defined constraints. The graph transformation formalism also enables the introduction of reusable idioms and patterns in model transformations We plan to further explore this opportunity in our future research. A new direction in model synthesis is aspect weaving. We have experimented with the development and application of

model weaving technology for generating complex, integrating models by merging different modeling aspects according to formally represented rules. This work has not yet been published.

2.1.2.d. Real-Time Programming Models

Event-Triggered Programming

In previous work, we had developed a time-triggered language, called Giotto, based on the Logical Execution Time (LET) model. In Giotto, software tasks are invoked and terminated strictly by clock events. This achieves deterministic behavior, which is of paramount importance in safety-critical systems. Last year, we extended the LET model to non-clock events, and called the resulting language xGiotto. This language permits the invocation and termination of tasks by arbitrary events in a way that still maintains determinism. This year we applied Giotto to the control of Unmanned Aerial Vehicles. This work has been completed but it has yet to be published.

Advanced Tool Architectures

A premise of this project is that many foundational results are best expressed through software. Academic papers can gloss over scalability, practicality, and design issues, and frequently are much harder to understand than a software application that embodies the concepts. We view software as a publication medium that for some research results is more complete, more understandable, and more rigorous than papers. To maximize impact, we distribute source code, and we put considerable effort into making sure that code is readable. Moreover, our copyright policies encourage re-use of the code, even in commercial products, in order to maximize the probability of significant impact.

Institutionally, we have a long history of producing high-quality pioneering tools (such as Spice, Espresso, MIS, Ptolemy, Polis, and HyTech from UCB, and GME, SSAT, and ACE from Vanderbilt) to disseminate the results of our research. The conventional notion of "tool," however, does not respond well to the challenges of deep compositionality, rapid construction and composition of DSMLs, and model-based transformation and generation. In this project, we have shifted the emphasis to tool architectures and tool components—that is, software modules that can be composed in flexible ways to enable researchers with modest resources to rapidly and (most importantly) correctly construct and experiment with sophisticated environments for hybrid and embedded systems.

We currently have three key frameworks that we use for this purpose, GME, which emphasizes meta modeling, Metropolis, which emphasizes codesign of architecture and functionality, and Ptolemy II, which emphasizes concurrent models of computation. The cores of these three frameworks predate this project. They are evolving together into a more coherent view of what frameworks and toolkits for hybrid and embedded software systems require.

All three frameworks share a focus on what we call "actor-oriented design," where components are conceptually concurrent and interaction between components is via the flow of data through ports. This contrasts with (and complements) prevailing object-oriented methods, where

components bundle data with methods and interaction between components is through procedure calls (method invocations, which semantically involve a transfer of control). Many commercial tools, such as Simulink, Labview, Modelica, Opnet, VHDL, and many others, use actor-oriented componentization (and some, like Modelica, use it together with object-oriented componentization). Thus, our work has promise of building the fundamentals behind high-profile and high-impact tools. For example, one key innovation of the last year is the introduction of "classes" and "inheritance" in an actor-oriented sense to complement such mechanisms that have long existed in the object-oriented sense. We report on some of the work from [45][46].

Conceptual concurrence between our frameworks is evolving. Whereas previously GME had mastered meta modeling of abstract syntactic properties of actor-oriented models, today it is moving aggressively towards meta modeling of their semantic properties, focusing initially on the the concurrent models of computation that have been implemented in Ptolemy II. Metropolis and Ptolemy II are both seeking to abstract these semantic properties in a somewhat different (and complementary) way than meta modeling. They both define an "abstract semantics" that represents the common features of families of models of computation, and they both realize models of computation through specialization (concretization) of this abstract semantics. They do so, however, in different ways, and by pursuing these different ways, the team is gaining an understanding of the fundamentals behind the use of such abstract semantics in frameworks.

A number of more specialized tools (vs. frameworks) are also being built to illustrate, refine, and publish research results. For example, Chic supports definitions of interfaces and interface theories and provides compatibility checking with respect to these interface theories. We have demonstrated that Chic can be used as a component within the Ptolemy II framework, and are using this integration to try to identify which interface theories are most productive and useful to designers. NP-Click, which was first prototyped within Ptolemy II, explores models of computation that appear to be particularly well-suited to high-speed networking systems design. Giotto, which has both a stand-alone textual syntax and a graphical syntax built in Ptolemy II, elevates the semantic principles of time-triggered architectures to the programming language and modeling level. GReAT builds on GME's meta modeling of abstract syntax to synthesize model transformers that bridge distinct abstract syntaxes. Desert builds on GME to provide systematic exploration of families of designs where components have several distinct available implementations. Blast and CCured are focused on improving the reliability of the embedded C code that ultimately emerges from these tools. Streambit explores a model of computation for computation on streams of bits, such as that typically found in networking applications. As these tools evolve, the most useful ones will be integrated into one or more of our frameworks, providing the community with a coherent view of best practices methods.

This year a number of survey papers gave coverage to this suite of tools (see for example [34][37][51][52][54][57][62][72][76][78][81]).

2.1.3.a. Syntax and Semantics

Modularity Mechanisms in Actor-Oriented Design

Concurrent, domain-specific languages such as Simulink, LabVIEW, Modelica, VHDL, SystemC, and OPNET are widely used in the design of embedded software. They provide modularization mechanisms that are significantly different from those in prevailing object-oriented languages such as C++ and Java. In these languages, components are concurrent objects that communicate via messaging, rather than abstract data structures that interact via procedure calls. Although the concurrency and communication semantics differ considerably between languages, they share enough common features that we consider them to be a family. Included in this family are our own hybrid systems modeling languages (like HyVisual) and embedded software design languages (like Giotto). We call them actor-oriented languages, and have been studying their properties as a family of languages.

Actor-oriented languages, like object-oriented languages, are about modularity of software. We argue that we can adapt for actor-oriented languages many (if not all) of the innovations of OO design, including concepts such as the separation of interface from implementation, strong typing of interfaces, subtyping, classes, inheritance, and aspects. We have realized some preliminary implementations of these mechanisms in Ptolemy II and published them in [17][34].

Code Generation from Actor-Oriented Models

We have been developing technology for code generation from actor-oriented models in Ptolemy II. This code generation system, called *Copernicus*, is designed to make maximum use of the same generic actor specifications used for simulation. The system is based on the concept of component specialization: generic actor specifications are transformed according the model context in which they are used. This model context includes information such as the connections between actor ports, assignments of values to actor parameters, and the model of computation that governs component interaction. Each aspect of a component's context, which doesn't change presents an opportunity for specialization to improve the execution performance of the component.

Recently we have focused on analysis techniques for analyzing the reconfiguration of parameter values that goes beyond simply determining whether parameter value changes or not. The analysis determines a bound on how often during the execution of a model particular parameters are reconfigured. We interpret this analysis as a behavioral type system capable of checking constraints on reconfiguration. Although reconfigured parameters cannot be specialized during code generation, behavioral type constraints on reconfiguration can enable scheduling optimization of the interaction between components. During code generation, this optimization allows threads and dynamic communication buffers to be replaced with statically scheduled code and statically allocated communication buffers. See [18][19][20].

Metropolis Framework

Features and efficiency are critical factors for simulators, which are still the dominating tools for design validation. During the past year, we improved the Metropolis simulator in several ways (see [32][33]):

1. *Add quantity annotation support*. Quantities can be used to model various kinds of performance indices, such as time, and power. They can also be generalized to model scheduling policies. Both of the usages usually appear in architecture models. A simulator's job is to ensure the quantity annotation requests are made at the proper time, and quantity resolution algorithms are executed when necessary to reach a fixed point solution.

2. *Add mapped behavior simulation support*. This capability is essential to evaluate behavior-architecture mappings, which is the key idea to explore design space in platform-based design. With this feature, a user's behavioral model can run simultaneously with a particular architecture under evaluation. Then, performance of the architecture running this behavior can be determined by simulation. During simulation, not only the events in both behavior and architecture models need to be synchronized, but also values need to be passed between. Both requirements present challenges to efficient simulation.

3. *Improve simulation efficiency*. In today's design, both behavior models and architecture models can be huge; combining them only makes the scale problem worse. We applied several techniques to improve the Metropolis simulator's performance. Simulation results show orders of magnitude performance improvement. The optimization techniques include named event reduction, a medium-centric simulation algorithm, mapped behavior handling with equivalent classes, efficient hierarchy traversal and an interleaving-concurrency-based simplification.

4. The current implementation of Metropolis simulator is based on the SystemC simulation kernel, which is an interleaving concurrent single kernel process environment. We detached the Metropolis simulator from this foundation and realized the parallelism with the pthreads library, which gives kernel-level processes (true concurrent) on some SMP machines. The preliminary experiments show sub-linear speed up on SMPs. Experiments show sub-linear speed up on the SMPs.

2.1.3.b. Interface Theories

A Component Model for Heterogeneous Systems

We have developed a new component model for timed models of computation such as discrete event, continuous time, hybrid systems, and synchronous/reactive models. Using the tagged signal model (developed by Lee and Sangiovanni-Vincentelli) as the basis to analyze the computational requirements of these models of computation, we developed a unified scheme to simulate heterogeneous timed models. The scheme relies on the proposed component model that aims to minimize the interface complexity between components and their operating environment. A generic component in this model is similar to a Mealy state machine, having an output function that computes the input-output relation at the current simulation time, and a next state function that computes the new state of the component. The simulation of a timed model goes through a sequence of time steps. In each step the system of equations formed by the components in the model is solved. This unified scheme provides a solid foundation for building correct

simulators of heterogeneous timed models. Extensions to the generic component model are developed to satisfy the requirements of specific models of computation, while still keeping the interface complexity of the components minimal. A small component interface makes component composition easier and more flexible.

The component model also makes it easier to study certain formal properties of the components. For example, we can classify discrete event components as either reactive or proactive. A DE component is reactive if all output events and state changes are triggered by input events and proactive otherwise. From the output and next state functions of a DE component, we can derive the relations among the time stamps of its input and output signals, and use these relations to analyze whether a DE composite component is reactive. Current work includes defining a behavioral type system based on pattern matching as the component programming model. See the papers [32][60][61][33].

Interface Modeling and Models of Computation

One research effort on interface modeling was centered around the fundamental principles of interface theories and the evaluation of their pragmatic impact on component-based designs. We have constructed a general experimentation framework within Ptolemy II, and integrated the Checker for Interface Compatibility (Chic) tool into Ptolemy II. We have experimented with the various supported formalisms and evaluated their applicability to the wide spectrum of computational models supported in Ptolemy II. The overall outcome stressed the inherent subjectivity of interface theories, and suggested a more model-of-computation oriented approach in their development.

An extended effort has been made towards the specification of behavioral types at the dynamic interaction level through interface automata[15]. We focused on augmenting the interface automata theory in order to handle explicitly more elaborate forms of concurrency. Based on a variant of interface automata we were able to develop an assume guarantee reasoning for mutual exclusion in open systems. In the process of generalizing our results, we revealed the fundamental limitations of the interface automata theory associated with the inherent computational capacity of finite automata.

2.1.3.c. Virtual Machine Architectures

Types for Real-Time Programs

We developed a type system for Embedded Machine code, which is assembly-like hard real-time code, with the property that well-typed programs are efficiently schedulable. A report has been submitted for publication on the use of embedded machine code for time triggered controllers for UAVs.

Separating Reactivity from Schedulability

We implemented two virtual machines, one to react to environment events (the E or Embedded machine), and the other to react to CPU events (the S or Scheduling Machine), and their interaction. This architecture allows great flexibility (portability, extensibility, and composibility) in the implementation of reactive software.

Implementing Event Scoping

We extended the E (Embedded) Machine, a virtual machine with reactive real-time behavior, to accommodate the event scoping mechanism of xGiotto. Publication is still pending.

Real-Time Software using Ptolemy-II, Giotto, and the E and S Virtual Machines

We have created a software infrastructure for designing hard real-time systems using Ptolemy II, Giotto, and an implementation of the E and S machines on KURT Linux (from University of Kansas). We use the Giotto domain within Ptolemy II, which offers a graphical syntax for Giotto models. Systems with periodic tasks are specified with a timing resolution of milliseconds. The first stage of our implementation takes a graphical model as an input and generates C code, E code, and S code. The second stage is an implementation of the Embedded Machine interpreter, which reads in the E Code and interprets it to release the tasks for execution as per the timing requirements stated. The released tasks can either be assigned to a standard scheduler such as EDF, or the scheduling can be preformed by our implementation of the Scheduling Machine, an interpreter for S code. See also [47].

2.1.3.d. Components for Embedded Systems

Mapping Network Applications to Multiprocessor Embedded Platforms

We formulated and solved the task allocation problem for a popular multithreaded, multiprocessor embedded system, the Intel IXP1200 network processor. This method proves to be computationally efficient and produces results that are within 5% of aggregate egress bandwidths achieved by hand-tuned implementations on two representative applications: IPv4 Forwarding and Differentiated Services. We are currently exploring extensions to this work by considering multiple target platforms: a reconfigurable multiprocessor system on the Xilinx Virtex-II Pro and the IXP2400. The results are reported in the paper [66].

2.1.3.e Verification of Embedded Software

Model Checking Quantitative Properties of Systems

We developed model checking algorithms for automata whose states are not labeled with boolean-valued propositions, but with natural-number valued quantities. These numbers might express, for example, power consumption or memory usage. A report on this work has been submitted for publication

Run-Time Error Handling

It is difficult to write programs that behave correctly in the presence of run-time errors. Existing programming language features often provide poor support for executing clean-up ode and for restoring invariants in such exceptional situations. e present a data flow analysis for finding a certain class of error-handling mistakes: those that arise from a failure to release resources or to clean up properly along all paths. Many real-world programs violate such resource safety policies because of incorrect error handling. Our flow-sensitive analysis keeps track of outstanding obligations along program paths and does a precise modeling of control flow in the presence of exceptions. Using it, we have found over 800 error handling mistakes almost 4

million lines of Java code[18][19][20]. Among the systems that we have debugged with our tool is the Ptolemy software.

Memory Safety Enforcement:

In previous work we have developed Ccured, a static analysis tool for C programs. CCured discovers for each pointer what kind of run-time checks, if any, are necessary in order to ensure memory safe execution. On average, Ccured discovers that 80% of the pointers used in a typical C program do not require any run-time checks. However, CCured has serious difficulties for programs that use libraries whose source is not available. For those libraries, CCured must not only make conservative assumptions about how they manipulate pointers, but must also refrain from changing the run-time representation of pointers and objects they point to. These difficulties arise in all static analysis and instrumentation systems. We have developed a solution based on user-defined polymorphic wrapper functions. These wrappers act both as a proxy for the missing library source code for the purpose of the static analysis, and also as the basis for generating code that changes the representation of pointers at the library boundary, to allow the co-existence of Ccured-controlled pointers and standard backwards-compatible pointers for the library objects. See [62] and [42].

2.1.4 Experimental Research

The main emphasis of our research is on the foundations of hybrid systems theory and of embedded system design. However, in the best tradition of our groups, a strong application program is necessary to verify the viability of the theory and to uncover difficult problems that provide appropriate motivation to develop new methods and theories. Most of the applications studied are distributed systems where scarce and fragile resources have to be used to provide reliable behavior. Wireless sensor networks, distributed systems for automotive electronics, embedded systems for national and homeland defense, are but a few examples that attracted the attention of our research groups because of their complexity and of their objective importance. We argue that the distributed nature of the applications poses additional challenges to overcome with an appropriate design methodology and supporting tools.

In particular, during this period, we have focused on the application to UAVs for air borne combat, Unmanned Underwater Vehicles, wireless sensor networks to control and monitoring, on fault-diagnosis, fault-adaptive and fault-tolerant approaches for distributed systems, and finally, on a multi-media problem as a test vehicle for the methodology and the tools embedded in Metropolis.

2.1.4.a. Embedded Control Systems

Automated Landing for Unmanned Aerial Vehicles (UAVs)

In work using hybrid control, we have partnered with Northrup Grumann to devise and actually fly (on a surrogate UAV, a Boeing T-35 with special avionics to allow it to be autonomous) algorithms for automated landing of UAVs with special contingency maneuvers for waving off landing. This work is reported in [73]

2.1.4.b. Embedded Software for National and Homeland Security

Aerial Pursuit Evasion Games for Aircraft

We have used techniques from stochastic hybrid systems along with templates of aerial combat techniques to build model predictive controllers (a form of approximate Hamilton Jacobi approximation controllers for safe operation) for enabling a UAV to participate in air to air combat. This work in partnership with Boeing Phantom Works was flown as a contest between a T-35 surrogate UAV and a human flown F-15E in the Mojave desert in July 2004. There were several instances of our controller defeating the skilled test pilot which led him to quip that the UAVs was just as good as a human pilot. See[74].

Distributed Control for Networks of Unmanned Underwater Vehicles (UUVs)

A decentralized control scheme for large packs of UUVs with communication constraints has been proposed and tested at our water tank facility at Richmond Field Station. See [36].

Softwalls for Collision Avoidance

In the last year, we have studied several methods for the Soft Walls controller, looking for a method which will work for a more realistic method of the aircraft. We have looked at a discrete-time formulation of the game theory formulation. Unfortunately, the first theoretical result did not lend itself to a practical, computable algorithm. With Claire Tomlin of Stanford and George Pappas of the University of Pennsylvania, we have begun to investigate collision avoidance methods based on computational geometry methods. Some intriguing connections are to be made between model predictive control and the solution of Hamilton Jacobi equations. See also [4].

Shooter Localization in Urban Terrain

The world's best localization algorithms for sensor networks have been developed this year at Vanderbilt and have been used to efficiently localize shooters in an urban environment (see [59]). This work is being transitioned by Raytheon Systems, Inc.

2.1.4.c. Networks of Distributed Sensors

VisualSense: Visual Editor and Simulator for Wireless Sensor Network Systems

Modeling of wireless sensor networks requires sophisticated modeling of communication channels, sensor channels, ad-hoc networking protocols, localization strategies, media access control protocols, energy consumption in sensor nodes, etc. VisualSense is a software framework designed to support a component-based construction of such models. It is intended to enable the research community to share models of disjoint aspects of the sensor nets problem and to build

models that include sophisticated elements from several aspects. VisualSense can be downloaded from http://ptolemy.eecs.berkeley.edu/visualsense.

Large Scale Sensor Networks

We looked at the problem of routing in large scale sensor networks, where only local connectivity information is used at each sensor node to decide the next-hop destination. The problem is studied in the context of small-world networks, providing a stochastic model that bridges between the well known 'six degrees of separation' property in social science and network routing protocols. Results are summarized in and have also been presented in different seminars and workshops. Some of this work began at Caltech and some under the DARPA NEST project, but it has been refined and revised under this project and is being carried forward under this project. Some of the references on this topic due to Bollobas and co-workers are not yet published and will be cited in the next report.

Another important issue in sensor networks regards how global connectivity is achieved in the network. Links are inherently unreliable and probabilistic connectivity models based on random connection models have been developed. In some recent work with Franceschetti, we obtain basic percolation results for different random connection models that account for the unreliability of the wireless channel.

The unreliability of the wireless channel is also the motivation for the work by Sinopoli, Franceschetti, Sastry et al (to be submitted to a special issue of the IEEE Proceedings), with the difference that in this case the effect on the control feedback loop is explicitly taken into account. The estimation problem is central for control systems (e.g. in pursuit evasion games with autonomous robots) and is based on the automatic refinement of the estimation as new observation data are collected in real time. When some of these data are lost due to the unreliability of the link the performance of such estimator obviously starts degrading. We explicitly characterize such degradation giving analytic bounds and showing a sharp transition to instability as the probability of link failure exceeds a given threshold. This is a substantial and first generalization to the centralized LQG paradigm of traditional control theory.

The data collection and distribution problems in sensor networks are studied in using a discrete mathematical model that allows to obtain basic limits on the time delay performance of different distribution and retrieval algorithms.

In [68] we addressed the basic problem of wireless signal propagation. Characterizing the physics of propagation is one of the key problems in wireless, as fundamental information theory limits strongly depend on the accuracy of the channel model. At the same time, this is a very complex task, due to complexity of the propagating environment that does not allow to solve Maxwell equations explicitly. Hence, often stochastic models based on few tunable parameters that can be analytically treated are employed. We proposed a stochastic model based on the theory of random walks and we obtained expressions for the path loss and power delay profile of a transmitted signal. The highlight of the research this year is the world's best localization algorithms for sensor networks developed by Vanderbilt (see for example [39]) and used in a number of applications such as shooter localization, see [59].

Distributed Control in Smart Structures

The control of Smart Structures presents a unique challenge for distributed control systems. The challenge is unique for two reasons: 1) the physics of Smart Structures are such that all nodes in a distributed sensing/control network will experience strongly connected dynamics and (2) the bandwidth of Smart Structures tends to be very high requiring high sampling rates and fast, efficient inter-node communications. In the past year we have begun experiments in the distributed control of Smart Structures experimental platforms that were constructed in the previous year. As noted, the primary challenges in these experiments center on (1) the design of distributed control systems that accommodate strongly connected inter-node dynamics and (2) the development of control systems and distributed network infrastructure that permit the relatively high sampling and communication rates needed. We have achieved some success in the distributed control development. However, the issue of sufficiently fast inter-node communication remains a bottle neck in performance. See [40][41].

Programming Models for Sensor Networks

We have created galsC [http://galsc.sourceforge.net][28], a language and compiler designed for use with the TinyGALS programming model, which uses TinyOS as the underlying component model. TinyGALS is a globally asynchronous, locally synchronous model for programming event-driven embedded systems, especially sensor networks. At the local level, software components communicate with each other synchronously via method calls. Components are composed to form actors. At the global level, actors communicate with each other asynchronously via message passing, which separates the flow of control between actors. A complementary model called TinyGUYS is a guarded yet synchronous model designed to allow thread-safe sharing of global state between actors without explicitly passing messages. The TinyGALS programming model is structured such that code for all inter-actor communication, actor triggering mechanisms, and access to guarded global variables can be automatically generated from a high level specification. By raising concurrency concerns above the level of TinyOS components, the TinyGALS programming model allows programmers to focus on the main tasks that the application must execute. Programs developed using this task-oriented model are thread safe and easy to debug.

The original TinyGALS code generation toolset was designed to be compatible with software components written for TinyOS 0.6.1. The new implementation is designed to be compatible with TinyOS 1.x, which uses the nesC programming language. Our new language, galsC, extends the nesC language, which allows for better code generation and static analysis of programs. We have also redesigned the TinyGUYS mechanism to have better scoping. Having a well-structured concurrency model at the application level greatly reduces the risk of concurrency errors, such as deadlock and race conditions. These features are especially important in severely memory-constrained and safety-critical systems.

Control of Communication Networks

In a series of papers Abate and co-authors have explored using stochastic hybrid systems congestion control schemes for both wired and wireless networks. These methods have

tremendous applicability to other classes of network embedded systems as well, see [2][3][4], [30].

Elder Care

We have been experimenting with three axis accelerometer attached to a person and studying the sensitivity and robustness of these measurements for activities: Change from sitting to standing, Free fall, Walking, Running. We also experimented with two different locations where the accelerometer is attached to the human body: on the chest and on the waist. The data is preliminary but it suggests that the data analysis will have to be customized. We have performed some recent deployments of the technology in old age homes in Sonoma, Aarhus, Denmark and in Tampere, Finland. Experimental data is being gathered and we expect this to be a very important source of new ideas for high confidence medical devices and systems

Networks of Distributed Sensors

Distributed acoustic sensing has received a lot of attention in sensor network work. The possible applications include self-localization of sensors, target identification and tracking as well as environmental monitoring. We have developed a distributed acoustic sensing experimental platform based on PC-104 stack hardware (the same used in the Smart Structures control experiments, see [40]) and begun testing and evaluation. The primary objective is to develop sensing and sensor fusion algorithms that are distributed in nature (i.e. the computations are carried out in a parallel fashion among the sensor nodes) and without the support of a centralized controller or processor. Initial experiments have demonstrated that accuracies comparable to more sophisticated centralized solvers are possible (see [80]). We have also applied the results to the distributed control of thermoelectric coolers in [43].

2.1.4.d. Fault-Driven Applications

Distributed Diagnosis of Complex Physical Systems

The size and complexity of present day systems motivates the need for developing distributed fault diagnosis algorithms. This work extends the TRANSCEND qualitative framework for fault diagnosis in continuous dynamic systems, to develop a methodology that partitions the set of possible fault candidates in a physical system to independent sets of faults given a set of measurements. Separate diagnosers that do not interact with each other can be constructed for each set of independent faults while maintaining complete diagnosability of the system. Our approach is to partition the set of faults into subsets in such a way that we can construct independent diagnosers for each subset. Two diagnosers are independent if they do not have to share information in establishing unique diagnosis results that are globally valid. We establish this by ensuring that the two fault subsets corresponding to the two diagnosers do not require the same set of measurements to achieve complete diagnosability. Complete diagnosability is the ability to uniquely isolate every fault candidate in the system given a set of measurements. Although such system decomposition will not always be possible, our approach can be the first step for designing distributed diagnosers. The implication of this approach is that a large computationally expensive diagnosis task is decomposed into a set of smaller tasks that can be

performed independently, thus reducing the overall complexity of online diagnosis. See for example [6].

On-line Hierarchical Fault Adaptive Control

This work extends previous work we have done on online approaches for the safety control of a general class of hybrid systems. This year we have focused on a hierarchical online fault-adaptive control approach for complex systems made up of a number of interacting subsystems. To avoid complexity in the models and online analysis, diagnosis and fault-adaptive control is achieved by local units. To maintain overall performance, the problem of resource management for contending concurrent subsystems has to be addressed. We have developed a control structure, where predefined set-point specifications for system operation are used to derive optimizing utility functions for the subsystem controllers. We apply this approach in situations where a fault occurs in a system, and once the fault is isolated and identified, the controllers use the updated system model to derive new set point specifications and utility functions for the faulty system. See for example the work in [7]

We have successfully demonstrated the use of this system for a NASA application that involves components of the Advanced Life Support Systems for long-term manned missions. This is reported in [16].

Safety-Critical Distributed Applications

We developed a design methodology and the supporting tools to address feedback control problems with fault-tolerant requirements (e.g. automotive safety-critical applications).

The flow lets the designer specify independently:

- the algorithmic solution
- the distributed execution platform
- the fault behavior

The three aspects of the design are represented using respectively

- a flavor of synchronous dataflow called fault-tolerant dataflow (FTDF)
- a bipartite graph (channels and electronic control units) with performance annotations
- a relation between failure patterns (subsets of the architecture graph that may fail in a same iteration) and the corresponding subset of the algorithm that must be guaranteed

Based on these three aspects of the specification, an automatic synthesis tool introduces redundancy in the algorithms and schedules the FTDF actors on the distributed architecture, so that the fault behavior is met. In doing so, the scheduling tool aims at minimizing latency (the critical path from sensors to actuators). Finally some verification tools analyze the solution to extract timing and to verify replica determinism and fault behavior. The work is reported in [60]

We have tested the entire flow on a drive-by-wire example from BMW and a steer-by-wire example from General Motors. The experiments support the value of the methodology and suggested directions for further improvement.

2.2. Project Findings

Abstracts for key publications representing project findings during this reporting period, are provided here. These are listed alphabetically by first author. A complete list of publications that appeared in print during this reporting period is given in section 3 below, including publications representing findings that were reported in the previous annual report.

[1] A Stochastic Approximation for Hybrid Systems

A. Abate, A. Ames, S. Sastry, In Proc. American Control Conference, (in publication), Portland, June 2005.

> **Abstract**: This paper introduces a method for approximating the dynamics of deterministic hybrid systems. Within this setting, we shall consider jump conditions that are characterized by spatial guards. After defining proper penalty functions along these deterministic guards, corresponding probabilistic intensities are introduced and the deterministic dynamics are approximated by the stochastic evolution of a continuous-time Markov process. We will illustrate how the definition of the stochastic barriers can avoid ill-posed events such as "grazing," and show how the probabilistic guards can be helpful in addressing the problem of event detection. Furthermore, this method represents a very general technique for handling Zeno phenomena; it provides a universal way to regularize a hybrid system. Simulations will show that the stochastic approximation of a hybrid system is accurate, while being able to handle "pathological cases." Finally, further generalizations of this approach are motivated and discussed.

[2] New Congestion Control Schemes over Wireless Networks: Stability Analysis

Alessandro Abate, Minghua Chen, Avideh Zakhor, Sankar Sastry, submitted to IFAC 05.

> **Abstract**: The objective of this work is to introduce two original flow control schemes for wireless networks. The mathematical underpinnings lie on the recently-developed congestion control models for Transmission-Control-Protocol(TCP)-like schemes; more precisely, the model proposed by Kelly for the wired case is taken as a template, and properly extended to the more involved wireless setting. We introduce two ways to modify a part of the model; the first is through a static law, and the second via a dynamic one. In both cases, we prove the global stability of the schemes, and present a convergence rate study and a stochastic analysis.

[3] New Congestion Control Schemes over Wireless Networks: Delay Sensitivity Analys is and Simulations

A. Abate, M. Chen, S. Sastry. In Proc. International Federation of Automatic Control World Congress, (in publication), Prague, July 2005.

Abstract: This paper proposes two new congestion control schemes for packet switched wireless networks. Starting from the seminal work of Kelly (Kelly *et al.*, Dec 1999), we consider the decentralized flow control model for a TCP-like scheme and extend it to the wireless scenario. Motivated by the presence of channel errors, we introduce updates in the part of the model representing the number of connections the user establishes with the network; this assumption has important physical interpretation. Specifically, we propose two updates: the ¯rst is static, while the second evolves dynamically. The global stability of both schemes has been proved; also, a stochastic stability study and the rate of convergence of the two algorithms have been investigated. This paper focuses on the delay sensitivity of both schemes. A stability condition on the parameters of the system is introduced and proved. Moreover, some deeper insight on the structure of the oscillations of the system is attained. To support the theoretical results, simulations are provided.

[4] Robust Model Predictive Control through Adjustable Variables: An Application to Path Planning

A. Abate, L. El Ghaoui, In Proc. International Conference on Decision and Control, Atlantis, Bahamas, December 2004.

Abstract: Robustness in *Model Predictive Control* (MPC) is the main focus of this work. After a definition of the conceptual framework and of the problem's setting, we will analyze how a technique developed for studying robustness in *Convex Optimization* can be applied to address the problem of robustness in the MPC case. Therefore, exploiting this relationship between Control and Optimization, we will tackle robustness issues for the first setting through methods developed in the second framework. Proofs for our results are included. As an application of this Robust MPC result, we shall consider a Path Planning problem and discuss some simulations thereabout.

[5] A Stability Criterion for Stochastic Hybrid Systems

A. Abate, L. Shi, S. Simic, S. Sastry, In Proc. International Symposium of Mathematical Theory of Networks and Systems, Leuven, July 2004.

Abstract: This paper investigates the notion of stability for Stochastic Hybrid Systems. The uncertainty is introduced in the discrete jumps between the domains, as if we had an underlying Markov Chain. The jumps happen every fixed time T; moreover, a result is given for the case of probabilistic dwelling times inside each domain. Unlike the more classical Hybrid Sys-stems setting, the guards here are time-related, rather than space-related. We shall focus on vector fields describing input-less dynamical systems.Clearly, the uncertainty intrinsic to the model forces to introduce a fairly new definition of stability, which can be related to the classical Lyapunov one though. Proofs and simulations for our results are provided, as well as a motivational example from finance.

[6] Hierarchical Online Control Design for Autonomous Resource Management in Advanced Life Support Systems

S. Abdelwahed, J. Wu, G. Biswas, E. J. Manders, paper no. 2005-01-2965, International Conference on Environmental Systems and European Symposium on Space Environmental Control Systems, (to appear), Rome Italy, July 11—14, 2005.

Abstract: This paper presents a distributed, hierarchical control scheme for autonomous resource management in complex embedded systems that can handle dynamic changes in resource constraints and operational requirements. The developed hierarchical control structure handles the interactions between subsystem and system-level controllers, A global coordinator at the root of the hierarchy ensures resource requirements for the duration of the mission are not violated, We have applied this approach to design a three-tier hierarchical controller for the operation of a lunar habitat that includes a number of interacting life support components.

[7] Online Fault-Adaptive Control for Efficient Resource Management in Advanced Life Support Systems

S. Abdelwahed, J. Wu, G. Biswas, J. Ramirez, E.J. Manders, Habitation: International Journal of Human Support Research, vol. 10, no. 2, pp. 105-115, 2005.

Abstract: This paper presents the design and implementation of a controller scheme for efficient resource management in Advanced Life Support Systems. In the proposed approach, a switching hybrid system model is used to represent the dynamics of the system components and their interactions. The operational specifications for the controller are represented as a utility function, and the corresponding resource management problem is formulated as a safety control problem. A limited-horizon online supervisory controller is used for this purpose. The online controller explores a limited region of the state-space of the system at each time step and uses the utility function to decide on the best action. The feasibility and accuracy of the online algorithm can be assessed at design time. We demonstrate the effectiveness of the scheme by running a set of experiments on the Reverse Osmosis (RO) subsystem of the Water Recovery System (WRS).

[8] Semantic Translation of Simulink/Stateflow models to Hybrid Automata using Graph Transformations

A. Agrawal, Gy. Simon, G. Karsai, Electronic Notes in Theoretical Computer Science, In Proc. Workshop on Graph Transformation and Visual Modeling Techniques (GT-VMT 2004), Volume 109, pp. 43-56.

Abstract: Embedded systems are often modeled using Matlab's Simulink and Stateflow (MSS), to simulate plant and controller behavior but these models lack support for formal verification. On the other hand verification techniques and tools do exist for models based on the notion of Hybrid Automata (HA) but there are no tools that can convert Simulink/Stateflow models into their semantically equivalent Hybrid Automata models. This paper describes a translation algorithm that converts a well-defined subset of the MSS modeling language into an equivalent hybrid automata. The translation has been

specified and implemented using a metamodel-based graph transformation tool. The translation process allows semantic interoperability between the industry-standard MSS tools and the new verification tools developed in the research community.

[9] Reusable Idioms and Patterns in Graph Transformation Languages

A. Agrawal, A. Vizhanyo, Z. Kalmar, F. Shi, A. Narayanan, G. Karsai, International Workshop on Graph-Based Tools, In Proc. 2004 International Conference on Graph Transformations, Rome, Italy, October, 2004.

Abstract: Software engineering tools based on Graph Transformation techniques are becoming available, but their practical applicability is somewhat reduced by the lack of idioms and design patterns. Idioms and design patterns provide prototypical solutions for recurring design problems in software engineering, but their use can be easily extended into software development using graph transformation systems. In this paper we briefly present a simple graph transformation language: GReAT, and show how typical design problems that arise in the context of model transformations can be solved using its constructs. These solutions are similar to software design patterns, and intend to serve as the starting point for a more complete collection.

[10] Blowing Up Affine Hybrid Systems

A. D. Ames, S. Sastry, 43rd IEEE Conference on Decision and Control 2004 (CDC'04), Atlantis, Paradise Island, Bahamas, Dec. 2004, pp. 473-478.

Abstract: In this paper we construct the "blow up" of an affine hybrid system H, i.e., a new affine hybrid system Bl(H) in which H is embedded, that does not exhibit Zeno behavior. We show the existence of a bijection U between periodic orbits and equilibrium points of H and Bl(H) that preserves stability; we refer to this property as P-stability equivalence.

[11] Characterization of Zeno Behavior in Hybrid Systems using Homological Methods

A. D. Ames, S. Sastry, 24th American Control Conference 2005 (ACC'05), (in publication), Portland, OR, 2005.

Abstract: It is possible to associate to a hybrid system a single topological space--its underlying topological space. Simultaneously, every hybrid system has a graph as its indexing object--its underlying graph. Here we discuss the relationship between the underlying topological space of a hybrid system, its underlying graph and Zeno behavior. When each domain is contractible and the reset maps are homotopic to the identity map, the homology of the underlying topological space is isomorphic to the homology of the underlying graph; the nonexistence of Zeno is implied when the first homology is trivial. Moreover, the first homology is trivial when the null space of the incidence matrix is trivial. The result is an easy way to verify the nonexistence of Zeno behavior.

[12] A Homology Theory for Hybrid Systems: Hybrid Homology

A. D. Ames, S. Sastry, In Proc. Hybrid Systems: Computation and Control, 8th International Workshop, Zurich, Switzerland, March 9-11, M. Morari and L. Thiele, eds., vol. 3414 of Lecture Notes in Computer Science, Springer-Verlag, pp. 86-102, 2005.

Abstract: By transferring the theory of hybrid systems to a categorical framework, it is possible to develop a homology theory for hybrid systems: hybrid homology. This is achieved by considering the underlying ``space'' of a hybrid system---its hybrid space or H-space. The homotopy colimit can be applied to this H-space to obtain a single topological space; the hybrid homology of an H-space is the homology of this space. The result is a spectral sequence converging to the hybrid homology of an H-space, providing a concrete way to compute this homology. Moreover, the hybrid homology of the H-space underlying a hybrid system gives useful information about the behavior of this system: the vanishing of the first hybrid homology of this H-space---when it is contractible and finite---implies that this hybrid system is not Zeno.

[13] Sufficient Conditions for the Existence of Zeno Behavior

A. D. Ames, S. Sastry, 44th IEEE Conference on Decision and Control and European Control Conference ECC 2005 (CDC-ECC'05), (submitted for publication), Seville, Spain, Dec., 12--15, 2005.

Abstract: In this paper, sufficient conditions for the existence of Zeno behavior in a class of hybrid systems is given; these are the first sufficient conditions on Zeno of which the authors are aware for hybrid systems with nontrivial dynamics. This is achieved by considering a class of hybrid systems termed diagonal first quadrant (DFQ) hybrid systems. When the underlying graph of a DFQ hybrid system has a cycle, we can construct an infinite execution for this system when the vector fields on each domain satisfy certain assumptions. To this execution, we can associate a single discrete time dynamical system that describes its continuous evolution. Therefore, we reduce the study of executions of DFQ hybrid systems to the study of a single discrete time dynamical system. We obtain sufficient conditions for the existence of Zeno by determining when this discrete time dynamical system is exponentially stable.

[14] A Decentralized Approach to Sound Source Localization with Sensor Networks

I. Amundson, P. Schmidt, K. D. Frampton, Presented at the 2004 ASME International Mechanical Engineering Conference and Exposition, Anaheim CA, November 2004.

Abstract: A sound source localization system has been developed based on a decentralized sensor network. Decentralization permits all nodes in a network to handle their own processing and decision-making, and as a result, reduce network congestion and the need for a centralized processor. The system consists of an array of battery operated COTS Ethernet-ready embedded systems with an attached microphone circuit. The localization solution requires groups of at least four nodes to be active within the array to return an acceptable two-dimensional result. Sensor nodes, positioned randomly

over a ten square meter area, recorded detection times of impulsive sources with microsecond resolution. In order to achieve a scalable system, nodes were organized in groups of from 4 to 10 nodes. Grouping was determined by the selecting the nodes farthest apart from each other. A designated leader of each group analyzed the sound source arrival times and calculated the sound source location based on time-differences of arrival. Experimental results show that this approach to sound source localization can achieve accuracies of about 30 cm . Perhaps more importantly though, it is accomplished in a decentralized manner which can lead to a more flexible, scalable distributed sensor network.

[15] Web Service Interfaces

D. Beyer, A. Chakrabarti, T. A. Henzinger, Proc. 14th International World Wide Web Conference (WWW 2005), Chiba, Japan, May 10—14 2005.

Abstract: We present a language for specifying web service interfaces. A web service interface puts three kinds of constraints on the users of the service.

First, the interface specifies the methods that can be called by a client, together with types of input and output parameters; these are called signature constraints. Second, the interface may specify propositional constraints on method calls and output values that may occur in a web service conversation; these are called consistency constraints. Third, the interface may specify temporal constraints on the ordering of method calls; these are called protocol constraints. The interfaces can be used to check, first, if two or more web services are compatible, and second, if a web service A can be safely substituted for a web service B. The algorithm for compatibility checking verifies that two or more interfaces fulfill each others' constraints. The algorithm for substitutivity checking verifies that service A demands fewer and fulfills more constraints than service B.

[16] Online Model-Based Diagnosis to Support Autonomous Operation of an Advanced Life Support System

G. Biswas, E.J. Manders, J.W. Ramirez, N. Mahadevan, S. Abdelwahed, Habitation: International Journal of Human Support Research, vol. 10, no. 1, pp. 21-38, 2004.

Abstract: This article describes methods for online model-based diagnosis of subsystems of the advanced life support system (ALS). The diagnosis methodology is tailored to detect, isolate, and identify faults in components of the system quickly so that fault-adaptive control techniques can be applied to maintain system operation without interruption. We describe the components of our hybrid modeling scheme and the diagnosis methodology, and then demonstrate the effectiveness of this methodology by building a detailed model of the reverse osmosis (RO) system of the water recovery system (WRS) of the ALS. This model is validated with real data collected from an experimental testbed at NASA JSC. A number of diagnosis experiments run on simulated faulty data are presented and the results are discussed.

[17] HyVisual: A Hybrid System Visual Modeler

C. Brooks, A. Cataldo, E. A. Lee, J. Liu, X. Liu, S. Neuendorffer, H. Zheng, Technical Memorandum UCB/ERL M04/18/, University of California, Berkeley, June 28, 2004.

Abstract: The Hybrid System Visual Modeler (HyVisual) is a block-diagram editor and simulator for continuous-time dynamical systems and hybrid systems. Hybrid systems mix continuous-time dynamics, discrete events, and discrete mode changes. This visual modeler supports construction of hierarchical hybrid systems. It uses a block-diagram representation of ordinary differential equations (ODEs) to define continuous dynamics, and allows mixing of continuous-time signals with events that are discrete in time. It uses a bubble-and-arc diagram representation of finite state machines to define discrete behavior driven by mode transitions.

In this document, we describe how to graphically construct models and how to interpret the resulting models. HyVisual provides a sophisticated numerical solver that simulates the continuous-time dynamics, and effective use of the system requires at least a rudimentary understanding of the properties of the solver. This document provides a tutorial that will enable the reader to construct elaborate models and to have confidence in the results of a simulation of those models. We begin by explaining how to describe continuous-time models of classical dynamical systems, and then progress to the construction of mixed signal and hybrid systems.

The intended audience for this document is an engineer with at least a rudimentary understanding of the theory of continuous-time dynamical systems (ordinary differential equations and Laplace transform representations), who wishes to build models of such systems, and who wishes to learn about hybrid systems and build models of hybrid systems.

HyVisual is built on top of Ptolemy II, a framework supporting the construction of such domain-specific tools. See Ptolemy II for more information.

[18] Heterogeneous Concurrent Modeling and Design in Java (Volume 1: Introduction to Ptolemy II)

C. Brooks, E.A. Lee, X. Liu, S. Neuendorffer, Y. Zhao, H. Zheng (eds.), Technical Memorandum UCB/ERL M04/27/, University of California, Berkeley, July 29, 2004.

Abstract: This volume describes how to construct Ptolemy II models for web-based modeling or building applications. The first chapter includes an overview of Ptolemy II software, and a brief description of each of the models of computation that have been implemented. It describes the package structure of the software, and includes as an appendix a brief tutorial on UML notation, which is used throughout the documentation to explain the structure of the software. The second chapter is a tutorial on building models using Vergil, a graphical user interface where models are built pictorially. The third chapter discusses the Ptolemy II expression language, which is used to set parameter values. The next chapter gives an overview of actor libraries. These three chapters, plus one of the domain chapters, will be sufficient for users to start building interesting

models in the selected domain. The fifth chapter gives a tutorial on designing actors in Java.The sixth chapter explains MoML, the XML schema used by Vergil to store models. And the seventh chapter, the final one in this part, explains how to construct custom applets.

Volume 2 describes the software architecture of Ptolemy II, and volume 3 describes the domains, each of which implements a model of computation.

[19] Heterogeneous Concurrent Modeling and Design in Java (Volume 2: Ptolemy II Software Architecture)

C. Brooks, E. A. Lee, X. Liu, S. Neuendorffer, Y. Zhao, H. Zheng (eds.), Technical Memorandum UCB/ERL M04/16/, University of California, Berkeley, June 24, 2004.

Abstract: This volume describes the software architecture of Ptolemy II. The first chapter covers the kernel package, which provides a set of Java classes supporting clustered graph topologies for models. Cluster graphs provide a very general abstract syntax for component-based modeling, without assuming or imposing any semantics on the models. The actor package begins to add semantics by providing basic infrastructure for data transport between components. The data package provides classes to encapsulate the data that is transported. It also provides an extensible type system and an interpreted expression language. The graph package provides graph-theoretic algorithms that are used in the type system and by schedulers in the individual domains. The plot package provides a visual data plotting utility that is used in many of the applets and applications. Vergil is the graphical front end to Ptolemy II and Vergil itself uses Ptolemy II to describe its own configuration.

Volume 1 gives an introduction to Ptolemy II, including tutorials on the use of the software, and volume 3 describes the domains, each of which implements a model of computation.

[20] Heterogeneous Concurrent Modeling and Design in Java (Volume 3: Ptolemy II Domains)

C. Brooks, E. A. Lee, X. Liu, S. Neuendorffer, Y. Zhao, H. Zheng (eds.), Technical Memorandum UCB/ERL M04/17/, University of California, Berkeley, June 24, 2004.

Abstract: This volume describes Ptolemy II domains. The domains implement models of computation, which are summarized in chapter 1. Most of these models of computation can be viewed as a framework for component- based design, where the framework defines the interaction mechanism between the components. Some of the domains (CSP, DDE, and PN) are thread-oriented, meaning that the components implement Java threads. These can be viewed, therefore, as abstractions upon which to build threaded Java programs. These abstractions are much easier to use (much higher level) than the raw threads and monitors of Java. Others (CT, DE, SDF) of the domains implement their own scheduling between actors, rather than relying on threads. This usual results in much more efficient execution. The Giotto domain, which addresses real-

time computation, is not threaded, but has concurrency features similar to threaded domains. The FSM domain is in a category by itself, since in it, the components are not producers and consumers of data, but rather are states. The non-threaded domains are described first, followed by FSM and Giotto, followed by the threaded domains. Within this grouping, the domains are ordered alphabetically (which is an arbitrary choice).

Volume 1 is an introduction to Ptolemy II, including tutorials on use of the software, and volume 2 describes the Ptolemy II software architecture.

[21] Discrete-Event Systems: Generalizing Metric Spaces and Fixed Point Semantics

A. Cataldo, E. A. Lee, X. Liu, E. Matsikoudis, H. Zheng, 16th International Conference on Concurrecy Theory (CONCUR 2005), (submitted for publication), San Francisco, CA, August 2005.

Abstract: This paper studies the semantics of discrete-event systems as a concurrent model of computation. The classical approach, which is based on metric spaces, does not handle well multiplicities of simultaneous events, yet such simultaneity is a common property of discrete-event models and modeling languages. (Consider, for example, delta time in VHDL.) In this paper, we develop a semantics using an extended notion of time. We give a generalization of metric spaces that we call tetric spaces. (A tetric functions like a metric, but its value is an element of a totally-ordered monoid rather than an element of the non-negative reals.) A straightforward generalization of the Banach fixed point theorem to tetric spaces supports the definition of a fixed-point semantics and generalizations of well-known sufficient conditions for avoidance of Zeno conditions.

[22] Verifying Quantitative Properties Using Bound Functions

A. Chakrabarti, K. Chatterjee, T. A. Henzinger, O. Kupferman and R. Majumdar, In Proc. 13th Advanced Research Working Conference on Correct Hardware Design and Verification Methods (CHARME 2005), Saarbrucken, Germany, October 3—6, 2005.

Abstract: In the boolean framework of model-based specification and verification, systems are graphs where each state is labeled with boolean propositions, and properties are languages where each trace has a boolean value (i.e., a trace either satisfies a property or it does not). We define and study a quantitative generalization of this traditional setting, where propositions have integer values at states, and properties have integer values on traces. For example, the value of a quantitative proposition at a state may represent power consumed at the state, and the value of a quantitative property on a trace may represent energy used along the trace. The value of a quantitative property at a state, then, is the maximum (or minimum) value achievable over all possible traces from the state. In this quantitative framework, model checking can be used to compute, for example, the minimum battery capacity necessary for achieving a given objective, or the maximal achievable lifetime of a system with a given initial battery capacity. In the case of open systems, these problems require the solution of games with integer values.

Quantitative model-checking or game-solving is undecidable, except if bounds on the computation can be found. Indeed, many interesting quantitative properties, like minimal necessary battery capacity and maximal achievable lifetime, can be naturally specified by a quantitative-bound automaton, which consists of (1) a finite automaton with integer registers, and (2) a bound function f that maps each system K to an integer f(K). While a traditional automaton accepts or rejects a given trace, a quantitative automaton maps each trace to an integer. The bound function f(K) defines an upper bound on register values which depends on the system K. We show that every quantitative-bound automaton defines a dynamic program that provides model-checking and game-solving algorithms. Along with the linear-time, automaton-based view of quantitative verification, we present a corresponding branching-time view based on a quantitative-bound mu-calculus. We study the relationship, expressive power, and complexity of both views.

[23] Two-player Nonzero-sum ω-regular Games

Krishnendu Chatterjee, In Proc. of 16th International Conference on Concurrency Theory, (submitted for publication), San Francisco, CA, August 23—26, 2005.

Abstract: We study infinite stochastic games played by two-players on a finite graph with goals specified by sets of infinite traces. The games are concurrent (each player simultaneously and independently chooses an action at each round), stochastic (the next state is determined by a probability distribution depending on the current state and the chosen actions), infinite (the game continues for an infinite number of rounds), nonzero-sum (the players' goals are not necessarily conflicting), and undiscounted. We show that if each player has an ω-regular objective expressed as a parity objective, then there exists an ϵ-Nash equilibrium, for every ε >0. However, exact Nash equilibria need not exist. We study the complexity of finding values (payoff profile) of an ε-Nash equilibrium. We show that the values of an ε-Nash equilibrium in nonzero-sum concurrent parity games can be computed by solving the following two simpler problems: computing the values of zero-sum (the goals of the players are strictly conflicting) concurrent parity games and computing ε-Nash equilibrium values of nonzero-sum concurrent games with reachability objectives. As a consequence we establish that values of an ε-Nash equilibrium can be approximated in FNP (functional NP), and hence in EXPTIME.

[24] The Complexity of Stochastic Rabin and Streett Games

K. Chatterjee, L. de Alfaro, T. A. Henzinger, 32nd International Colloquium of Automata, Languages and Programming (ICALP 05), (submitted for publication), Lisboa, Portugal, July 11--15, 2005.

Abstract: The theory of graph games with ω-regular winning conditions

is the foundation for modeling and synthesizing reactive processes. In the case of stochastic reactive processes, the corresponding stochastic graph games have three players, two of them (System and Environment) behaving adversarially, and the third (Uncertainty) behaving probabilistically. We consider two problems for stochastic graph

games: the {\em qualitative\/} problem asks for the set of states from which a player can win with probability~1(\emph{almost-sure winning}); the {\em quantitative\/} problem asks for the maximal probability of winning (\emph{optimal winning}) from each state. We show that for Rabin winning conditions, both problems are in NP. As these problems were known to be NP-hard, it follows that they are NP-complete for Rabin conditions, and dually, coNP-complete for Streett conditions. The proof proceeds by showing that pure memoryless strategies suffice for qualitatively and quantitatively winning stochastic graph games with Rabin conditions. This insight is of interest in its own right, as it implies that controllers for Rabin objectives have simple implementations. We also prove that for every ω-regular condition, optimal winning strategies are no more complex than almost-sure winning strategies.

[25] Trading Memory for Randomness

K. Chatterjee, L. de Alfaro, T. A. Henzinger. In Proc. 1st International Conference on Quantitative Evaluation of Systems (QEST 04), University of Twente, Enschede, The Netherlands, September 27 --30, 2004.

Abstract: Strategies in repeated games can be classified as to whether or not they use memory and/or randomization. the deterministic and probabilistic varieties. We characterize when memory and/or randomization are required for winning with respect to various classes of ω-regular objectives, noting particularly when the use of memory can be traded for the use of randomization. In particular, we show that Markov decision processes allow randomized memoryless optimal strategies for all M\"uller objectives. Furthermore, we show that 2-player probabilistic graph games allow randomized memoryless strategies for winning with probability~1 those M\"uller objectives which are upward-closed. Upward-closure means that if a set α of infinitely repeating vertices is winning, then all supersets of α are also winning.

[26] Mean-Payoff Parity Games

K. Chatterjee, T. A. Henzinger, M. Jurdzinski, 20th Annual Symposium of Logics in Computer Science (LICS 05), (submitted for publication), Chicago, IL, June 26—29, 2005

Abstract: Games played on graphs may have qualitative objectives, such as the satisfaction of an ω-regular property, or quantitative objectives, such as the optimization of a real-valued reward. When games are used to model reactive systems with both fairness assumptions and quantitative (e.g., resource) constraints, then the corresponding objective combines both a qualitative and a quantitative component. In a general case of interest, the qualitative component is a {\it parity} condition and the quantitative component is a{\it mean-payoff} reward. We study and solve such mean-payoff parity games. We also prove some interesting facts about mean-payoff parity games which distinguish them both from mean-payoff and from parity games. In particular, we show that optimal strategies exist in mean-payoff parity games, but they may require infinite memory.

[27] Counter-example Guided Planning

K. Chatterjee, T. A. Henzinger, R. Jhala, R. Majumdar, 21st International Conference in Uncertainty in Artifical Intelligence (UAI 05), (submitted for publication), University of Edinburgh, Edinburgh, Scotland, July 26—29, 2005.

Abstract: Planning in adversarial and uncertain environments can be modeled as the problem of devising strategies in stochastic perfect information games. These games are generalizations of Markov decision processes (MDPs): there are two (adversarial) players, and a source of randomness. The main practical obstacle to computing winning strategies in such games is the size of the state space. In practice therefore, one typically works with {\em abstractions} of the model. The difficulty, of course, is to come up with an abstraction that is neither too coarse to remove all winning strategies (plans), nor too fine to be intractable. In verification, the paradigm of {\em counterexample-guided abstraction refinement} has been successful to construct useful but parsimonious abstractions {\em automatically}.We extend this paradigm, for the first time, to {\em probabilistic} models (namely, $\twohalf$ games and, as a special case, MDPs). This allows us to apply the counterexample-guided abstraction paradigm to the AI planning problem. As special cases, we get planning algorithms for MDPs and deterministic systems that automatically construct system abstractions.

[28] galsC: A Language for Event-Driven Embedded Systems

E. Cheong, J. Liu, Presented at Design, Automation and Test in Europe (DATE), Munich, Germany, March 7--11, 2005.

Abstract: We introduce galsC, a language designed for programming event-driven embedded systems such as sensor networks. galsC implements the TinyGALS programming model. At the local level, software components are linked via synchronous method calls to form actors. At the global level, actors communicate with each other asynchronously via message passing, which separates the flow of control between actors. A complementary model called TinyGUYS is a guarded yet synchronous model designed to allow thread-safe sharing of global state between actors via parameters without explicitly passing messages. The galsC compiler extends the nesC compiler, which allows for better type checking and code generation. Having a well-structured concurrency model at the application level greatly reduces the risk of concurrency errors, such as deadlock and race conditions. The galsC language is implemented on the Berkeley motes and is compatible with the TinyOS/nesC component library. We use a multi-hop wireless sensor network as an example to illustrate the effectiveness of the language.

[29] Toward a Semantic Anchoring Infrastructure for Domain Specific Modeling Languages

K. Chen, J. Sztipanovits, S. Neema, M. Emerson, S. Abdelwahed, Embedded Systems Software Conference (EMSOFT 2005), (submitted for publication), Jersey City, NJ, September 18—22, 2005.

Abstract: Metamodeling facilitates the rapid, inexpensive development of domain-specific modeling languages (DSML-s). However, there are still challenges hindering the wide-scale industrial application of model-based design. One of these unsolved problems is the lack of a practical, effective method for the formal specification of DSML semantics. This problem has negative impact on reusability of DSML-s and analysis tools in domain specific tool chains. To address these issues, we propose a formal well founded methodology with supporting tools to anchor the semantics of DSML-s to precisely defined and validated "semantic units". In our methodology, each of the syntactic and semantic DSML components is defined precisely and completely. The main contribution of our approach is that it moves toward an infrastructure for DSML design that integrates formal methods with practical engineering tools. In this paper we use a mathematical model, Abstract State Machines, a common semantic framework to define the semantic domains of DSML-s.

[30] New Congestion Control Schemes Over Wireless Networks: Stability Analysis

M. Chen, A. Abate, S. Sastry. In Proc. International Federation of Automatic Control World Congress, (in publication), Prague, July 2005.

Abstract: This paper proposes two new congestion control schemes for packet switched wireless networks. Starting from the seminal work of Kelly (Kelly *et al.*, Dec 1999), we consider the decentralized flow control model for a TCP-like scheme and extend it to the wireless scenario. Motivated by the presence of channel errors, we introduce updates in the part of the model representing the number of connections the user establishes with the network; this assumption has important physical interpretation. Specifically, we propose two updates: the ¯rst is static, while the second evolves dynamically. The global stability of both schemes has been proved; also, a stochastic stability study and the rate of convergence of the two algorithms have been investigated. This paper focuses on the delay sensitivity of both schemes. A stability condition on the parameters of the system is introduced and proved. Moreover, some deeper insight on the structure of the oscillations of the system is attained. To support the theoretical results, simulations are provided.

[31] Stability and Delay Considerations for Flow Control Over Wireless Networks

M.Chen, A. Abate, A. Zakhor, S. Sastry, UCB ERL Tech Report No M05/14, Berkeley, CA, 2005.

Abstract: In this paper we develop a general framework for the problem of flow control over wireless networks, evaluate the existing approaches within that framework, and propose new ones. Significant progress has been made on the mathematical modeling of flow control for the wired Internet, among which Kelly's contribution is widely accepted as a standard framework. We extend Kelly's flow control framework to the wireless scenario, where the wireless link is assumed to have a fixed link capacity and a packet loss rate caused by the physical channel errors. In this framework, the problem of flow

control over wireless can be formulated as a convex optimization problem with noisy feedback. We then propose two new solutions to the problem achieving optimal performance by only modifying the application layer. The global stability and the delay sensitivity of the schemes are investigated, and verified by numerical results. Our work advocates the use of multiple connections for flow, or congestion control, over wireless.

[32] Simulation Based Deadlock Analysis for System Level Designs

X. Chen, A. Davare, H. Hsieh, A. Sangiovanni-Vincentelli, Y. Watanabe, ACM/IEEE Design Automation Conference, (submitted for publication), Anaheim, CA, June 2005.

Abstract: In the design of highly complex, heterogeneous, and concurrent systems, deadlock detection and resolution remains an important issue. In this paper, we systematically analyze the synchronization dependencies in concurrent systems modeled in the Metropolis design environment, where system functions, high level architectures and function-architecture mappings can be modeled and simulated. We propose a data structure called the dynamic synchronization dependency graph, which captures the runtime (blocking) dependencies. A loop-detection algorithm is then used to detect deadlocks and help designers quickly isolate and identify modeling errors that cause the deadlock problems. We demonstrate our approach through a real world design example, which is a complex functional model for video processing and a high level model of function-architecture mapping.

[33] The Best of Both Worlds: The Efficient Asynchronous Implementation of Synchronous Specifications

A. Davare, K. Lwin, A. Kondratyev, A. Sangiovanni-Vincentelli. Presented at ACM/IEEE Design Automation Conference, San Diego, CA, June 7 --11, 2004.

Abstract: The desynchronization approach combines a traditional synchronous specification style with a robust asynchronous implementation model. The main contribution of this paper is the description of two optimizations that decrease the overhead of desynchronization. First, we investigate the use of clustering to vary the granularity of desynchronization. Second, by applying temporal analysis on a formal execution model of the desynchronized design, we uncover significant amounts of timing slack. These methods are successfully applied to industrial RTL designs.

[34] Overview of the Ptolemy Project

J. Davis, C. Hylands., J. Janneck, E.A. Lee, J. Liu, X.Liu, S. Neuendorffer, S. Sachs, M. Stewart, K. Vissers, P. Whitaker, Y. Xiong, Technical Memorandum UCB/ERL M01/11, EECS, University of California, Berkeley, March 6, 2001.

[35] Implementing and Testing a Nonlinear Model Predictive Tracking Controller for Aerial Pursuit Evasion Games on a Fixed Wing Aircraft

J. M. Eklund, J. Sprinkle, S. S. Sastry, American Control Conference (ACC) 2005, (In Publication), Portland, OR, Jun., 8-10, 2005.

Abstract: The capability of Unmanned Aerial Vehicles (UAVs) to perform autonomously has not yet been demonstrated, however this is an important step to enable at least limited autonomy in such aircraft to allow then to operate with temporary loss of remote control, or when confronted with an adversary or obstacles for which remote control is insufficient. Such capabilities have been under development through Software Enabled Control(SEC) program and were recently tested in the Capstone Demonstration of that program. In this paper the final simulation and flight test results are presented for a Non-linear Model Predictive Controller (NMPC) used in evasive maneuvers in three dimensions on a fixed wing UAV for the purposes of pursuit/evasion games with a piloted F-15 aircraft.

[36] Template Based Planning and Distributed Control for Networks of Unmanned Underwater Vehicles

J. M. Eklund, J. Sprinkle, S. S. Sastry, 44th IEEE Conference on Decision and Control and European Control Conference ECC 2005 (CDC-ECC'05), (submitted for publication), Seville, Spain, Dec., 12-15, 2005.

Abstract: A decentralized control scheme for large packs of unmanned underwater vehicles (UUV) is proposed and investigated. This scheme is based on shared knowledge of a template, which includes operational plans, modes of operation, contingencies including the ability to adapt individual plans within the template to changing operational conditions, and the protocols for disseminating individual state, network and command information between UUVs. This template-based control enables complex and cooperative functionality by the network within the bounds of severe communications limitations and provides for a highly scalable solution for distributed control. Simulation results of medium-sized packs of UUVs are presented and the road ahead to physical implementation, experimentation and deployment is described.

[37] A MOF-Based Metamodeling Environment

M. Emerson, J. Sztipanovits, T. Bapty, Journal of Universal Computer Science, vol. 10, No. 10, pp. 1357-1382, October, 2004.

Abstract: The Meta Object Facility (MOF) forms one of the core standards of the Object Management Group's Model Driven Architecture. It has several use-cases, including as a repository service for storing abstract models used in distributed object-oriented software development, a development environment for generating CORBA IDL, and a metamodeling language for the rapid specification, construction, and management of domain-specific technology-neutral modeling languages. This paper will focus on the use of MOF as a metamodeling language and describe our latest work on changing the MIC metamodeling environment from UML/OCL to MOF. We have implemented a functional graphical metamodeling environment based on the MOF v1.4 standard using GME and GReAT. This implementation serves as a testament to the power of formally well-defined metamodeling and metamodel-based model transformation approaches. Furthermore, our

work gave us an opportunity to evaluate sevaral important features of MOF v1.4 as a metamodeling language: (a) Completeness of MOF v1.4 for defining the abstract syntax for complex (multiple aspect) DSML-s , (b) The Package concept for composing and reusing metamodels and (c) Facilities for modeling the mapping between the abstract and concrete syntax of DSML-s.

[38] Implementing a MOF-Based Metamodeling Environment Using Graph Transformations

M. Emerson, J. Sztipanovits, In Proc. 4th Workshop on Domain-Specific Modeling, pp. 83-92, 20th Annual ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA), Vancouver, Canada, October 2004.

Abstract: Versatile model-based design demands languages and tools which are suitable for the creation, manipulation, transformation, and composition of domain-specific modeling languages and domain models. The Meta Object Facility (MOF) forms the cornerstone of the OMG's Model Driven Architecture (MDA) as the standard metamodeling language for the specification of domain-specific languages. We have implemented MOF v1.4 as an alternative metamodeling language for the Generic Modeling Environment (GME), the flagship tool of Model Integrated Computing (MIC). Our implementation utilizes model-to-model transformations specified with the Graph Rewriting and Transformation toolsuite (GReAT) to translate between MOF and the UML-based GME metamodeling language. The technique described by this paper illustrates the role graph transformations can play in interfacing MIC technology to new and evolving modeling standards.

[39] Acoustic Self-Localization in a Distributed Sensor Network

K. D. Frampton, IEEE Sensors Journal, (in publication), 2005.

Abstract: The purpose of this work is to present a technique for determining the locations of nodes in a distributed sensor network. This technique is based on the Time Difference of Arrival (TDOA) of acoustic signals. In this scheme, several sound sources of known locations transmit while each node in the sensor network records the wave front time-of-arrival. Data from the nodes are transmitted to a central processor and the nonlinear TDOA equations are solved. Computational simulation results are presented in order to quantify the solution behavior and its sensitivity to likely error sources. Experimental self-localization results are also presented in order to demonstrate the potential for this approach in solving the challenging self-localization problem.

[40] Distributed Group-Based Vibration Control with a Networked Embedded System

K. D. Frampton, Journal of Intelligent Materials Systems and Structures, Vol. 14, pp. 307--314, 2005.

Abstract: The purpose of this work is to demonstrate the performance of a distributed vibration control system based on a networked embedded system. The platform from which control is affected consists of a network of computational elements called nodes.

Each node possesses its own computational capability, sensor, actuator and the ability to communicate with other nodes via a wired or wireless network. The primary focus of this work is to demonstrate the use of existing group management middleware concepts to enable vibration control with such a distributed network. Group management middleware is distributed software that provides for the establishment and maintenance of groups of distributed nodes and that provides for the network communication within such groups. The reason for developing distributed control based on group concepts is that communication of real-time sensor and actuator data among all system nodes would not be possible due to bandwidth constraints. Group management middleware provides for inter-node communications among subsets of nodes in an efficient and scalable manner. The objective of demonstrating the effectiveness of such grouping for distributed control is met by designing distributed feedback compensators that take advantage of node groups in order to affect their control. Two types of node groups are considered: groups based on physical proximity and groups based on modal sensitivity. The global control objective is to minimize the vibrational response of a rectangular plate in specific modes while minimizing spillover to out-of-bandwidth modes. Results of this investigation demonstrate that such a distributed control system can achieve vibration attenuations comparable to that of a centralized controller. The importance of efficient use of network communications bandwidth is also discussed with regard to the control architectures considered.

[41] Vibroacoustic Control with a Distributed Sensor Network

K. D. Frampton, Presented at Applications of Graph Transformations with Industrial Relevance (AGTIVE04), Williamsburg, VA, September, 2004.

Abstract: The purpose of this work is to demonstrate the ability of a distributed control system, based on a networked embedded system, to reduce acoustic radiation from a vibrating structure. The platform from which control is affected consists of a network of computational elements called nodes. Each node possesses its own computational capability, sensor, actuator and the ability to communicate with other nodes via a wired or wireless network. The primary focus of this work is to employ existing group management middleware concepts to enable vibration control with such a distributed network. Group management middleware is distributed software that provides for the establishment and maintenance of groups of distributed nodes and that provides for the network communication among such groups. This objective is met by designing distributed feedback compensators that take advantage of node groups in order to affect their control. Two types of node groups are considered: groups based on physical proximity and groups based on modal sensitivity. The global control objective is to minimize the vibrational response of a rectangular plate in specific modes while minimizing spillover to out-of-bandwidth modes. Results of this investigation demonstrate that such a distributed control system can achieve vibration attenuations comparable to that of a centralized controller. The importance of efficient use of network communications bandwidth is also discussed with regard to the control architectures considered.

[42] Using Dependent Types to Certify the Safety of Assembly Code

M. Harren, G. C. Necula, 12th International Static Analysis Symposium (SAS '05), (in publication), London, UK, September 7-9, 2005.

Abstract: There are many source-level analyses or instrumentation tools that enforce various safety properties. In this paper we present an infrastructure that can be used to check independently that the assembly output of such tools has the desired safety properties. By working at assembly level we avoid the complications with unavailability of source code, with source-level parsing, and we certify the code that is actually deployed. The novel feature of the framework is an extensible dependently-typed framework that supports type inference and mutation of dependent values in memory. The type system can be extended with new types as needed for the source-level tool that is certified. Using these dependent types, we are able to express the invariants enforced by CCured, a source-level instrumentation tool that guarantees type safety in legacy C programs. We can therefore check that the x86 assembly code resulting from compilation with CCured is in fact type-safe.

[43] Distributed Control to Improve Performance of Thermoelectric Coolers

R. D. Harvey, D. G. Walker, K. D. Frampton, Presented at 2004 ASME International Mechanical Engineering Conference and Exposition, Anaheim CA, November 2004.

Abstract: Thermoelectric coolers (TECs) have become more popular in chip cooling applications. However, due to material properties, the scope of TEC applicability is limited. The primary reason for this limitation stems from the poor efficiency of the TEC. This low efficiency causes increased heat production resulting in a very narrow band in which the TEC is effective. Since TECs are cooling units composed of numerous individual cooling elements, this band can be expanded by implementing distributed control of the individual cooler components. Distributed control is a system for allowing each element to be powered depending on the localized heat load. Distributed control would allow for increased cooling in hot spots while minimizing excess heat generated by the TEC in areas where it is not needed. The preliminary results suggest that this type of control may be feasible, and would result in a significant increase in the TEC effectiveness. The current work provides a closer look at the increased effectiveness and improves the previous models. The current model considers lateral heat conduction in the chip, as well as variable control of the individual cooling elements proportional to heat load. By modeling different scenarios for heat distributions and exploring the application of the individual cooling units, wider applicability of the TEC for computer chip cooling can be achieved.

[44] Concerns on Separation: Modeling Separation of Concerns in the Semantics of Embedded Systems

E. Jackson, J. Sztipanovits, In Proc. Embedded Software Systems Conference (EMSOFT'2005), (in publication), Jersey City, NJ, September 18—22, 2005.

> **Abstract:** Embedded systems are commonly abstracted as collections of interacting components. This perspective has lead to the insight that component behaviors can be defined separately from admissible component interactions. We show that this separation of concerns does not imply that component behaviors can be defined in isolation from their envisioned interaction models. We argue that a type of behavior/interaction co-design must be employed to successfully leverage the separation of these concerns. We present formal techniques for accomplishing this co-design and describe tools that implement these formalisms.

[45] Graph Transformations in OMG's Model-Driven Architecture

G. Karsai, A. Agrawal, In Proc. Applications of Graph Transformations with Industrial Relevance (AGTIVE 2003), LNCS 2062. pp. 243-259, Charlottesville, VA, September 29—October 1, 2003.

> **Abstract:** The Model-Driven Architecture (MDA) vision of the Object Management Group offers a unique opportunity for introducing Graph Transformation (GT) technology to the software industry. The paper proposes a domain-specific refinement of MDA, and describes a practical manifestation of MDA called Model-Integrated Computing (MIC). MIC extends MDA towards domain-specific modeling languages, and it is well supported by various generic tools that include model transformation tools based on graph transformations. The MIC tools are metaprogrammable, i.e. they can be tailored for specific domains using metamodels that include metamodels of transformations. The paper describes the development process and the supporting tools of MIC, and it raises a number of issues for future research on GT in MDA.

[46] Design Patterns for Open Tool Integration

G. Karsai, A. Lang, S. Neema, Journal of Software and System Modeling, vol 4., no. 1, DOI: 10.1007/s10270-004-0073-y, 2004.

> **Abstract:** Design tool integration is a highly relevant area of software engineering, which can greatly improve the efficiency of development processes. Design patterns have been widely recognized as important contributors to the success of software systems. This paper describes and compares two large-grain, architectural design patterns that solve specific design tool integration problems. Both patterns have been implemented and used in real-life engineering processes.

[47] Real-Time Systems Design in Ptolemy II: A Time-Triggered Approach

V. Krishnan, Master's Report, Technical Memorandum UCB/ERL M04/22/, University of California, Berkeley, July 12, 2004.

> **Abstract**: In this report is described a software infrastructure to enable users to design hard real-time systems from Ptolemy II [1]. The Giotto [2] domain within the Ptolemy II

design environment is made use of to model systems which are then compiled and executed on KURT-Linux [3], a real time flavor of Linux.

The first stage of the software takes a graphical model as an input to generate intermediate code in the C language. This intermediate code consists of the task-code to be executed, as well as a representation of their timing requirements.

The second stage, called the Embedded Machine [5] reads in the timing information and interprets it to release the tasks for execution as per the stated requirements. The released tasks can either be assigned to a standard scheduler such as EDF, or to a scheduling interpreter called the Scheduling machine, or S Machine.

The S Machine was developed to gain fine grained control over the scheduling of tasks. The S Machine requires as input scheduling information that specifies a time line for the tasks involved thus giving the designer maximum flexibility over task scheduling, and consequently greater resource utilization. The E&S Machines when compiled along with the generated task and timing code for the KURT-Linux platform forms an executable that delivers predictable real-time performance. The benefit this approach offers is that the real-time tasks can run along with ordinary Linux tasks without the timing properties of the real-time tasks being affected.

An audio application was designed to illustrate the effectiveness of this tool-flow, which achieved a timing uncertainty of less than 130 microseconds in its task execution times.

[48] Engineering Education: A Focus on System, in Advances in Control, Communication Networks, and Transportation Systems: In Honor of Pravin Varaiya

E. A. Lee, E.H. Abed (Ed.), Systems and Control: Foundations and Applications Series, Birkhauser, Boston, 2005.

Abstract: Engineers have a major advantage over scientists. For the most part, the systems we analyze are of our own devising. It has not always been so. Not long ago, the principle objective of engineering was to coax physical materials to do our bidding by leveraging their intrinsic physical properties. The discipline was of "applied science." Today, a great deal of engineering is about coaxing abstractions that we have invented. The abstractions provided by microprocessors, programming languages, operating systems, and computer networks are only loosely linked to the underlying physics of electronics.

[49] Absolutely Positively On Time: What Would It Take?

E. A. Lee, Editorial, February 19, 2005, Available at http://ptolemy.eecs.berkeley.edu/publications/papers/05/EmbeddedSoftwareColumn/

Editorial, March 8, 2005: Despite considerable progress in software and hardware techniques, when embedded computing systems absolutely must meet tight timing constraints, many of the advances in computing become part of the problem rather than part of the solution. Although synchronous digital logic delivers precise timing determinacy, advances in computer architecture have made it difficult or impossible to estimate the execution time of software. Moreover, networking techniques introduce variability and stochastic behavior, and operating systems rely on best effort techniques. Worse, programming languages lack time in their semantics, so timing requirements are only specified indirectly. In this column, I examine the following question, "if precise timeliness in a networked embedded system is absolutely essential, what has to change?" The answer, unfortunately, is "nearly everything."

Twentieth century computer science has taught us that everything that can be computed can be specified by a Turing machine. "Computation" is accomplished by a terminating sequence of state transformations. This core abstraction underlies the design of nearly all computers, programming languages, and operating systems in use today. But unfortunately, this core abstraction does not fit embedded software very well.

This core abstraction fits reasonably well if embedded software is simply "software on small computers." In this view, embedded software differs from other software only in its resource limitations (small memory, small data word sizes, and relatively slow clocks). In this view, the "embedded software problem" is an optimization problem. Solutions emphasize efficiency; engineers write software at a very low level (in assembly code or C), avoid operating systems with a rich suite of services, and use specialized computer architectures such as programmable DSPs and network processors that provide hardware support for common operations. These solutions have defined the practice of embedded software design and development for the last 25 years or so.

Of course, thanks to the semiconductor industry's ability to follow Moore's law, the resource limitations of 25 years ago should have almost entirely evaporated today. Why then has embedded software design and development changed so little? It may be that extreme competitive pressure in products based on embedded software, such as consumer electronics, rewards only the most efficient solutions. This argument is questionable, however. There are many examples where functionality has proven more important than efficiency. It is arguable that resource limitations are not the only defining factor for embedded software, and may not even be the principal factor.

There are clues that embedded software differs from other software in more fundamental ways. If we examine carefully why engineers write embedded software in assembly code or C, we discover that efficiency is not the only concern, and may not even be the main concern. The reasons may include, for example, the need to count cycles in a critical inner loop, not to make it fast, but rather to make it predictable. No widely used programming language integrates a way to specify timing requirements or constraints.

Instead, the abstractions they offer are about scalability (inheritance, dynamic binding, polymorphism, memory management), and, if anything, further obscure timing (consider the impact of garbage collection on timing). Counting cycles, of course, becomes extremely difficult on modern processor architectures, where memory hierarchy (caches), dynamic dispatch, and speculative execution make it nearly impossible to tell how long it will take to execute a particular piece of code. Worse, execution time is context dependent, which leads to unmanageable variability. Still worse, programming languages are almost always Turing complete, and as a consequence, execution time is undecidable in general. Embedded software designers must choose alternative processor architectures such as programmable DSPs, and must use disciplined programming techniques (e.g. avoiding recursion) to get predictable timing.

Another reason engineers stick to low-level programming is that embedded software typically has to interact with hardware that is specialized to the application. In conventional software, interaction with hardware is the domain of the operating system. Device drivers are not typically part of an application program, and are not typically created by application designers. But in the embedded software context, generic hardware interfaces are rarer. The fact is that creating interfaces to hardware is not something that higher level languages support. For example, although concurrency is not uncommon in modern programming languages (consider threads in Java), no widely used programming language includes in its semantics the notion of interrupts. Yet the concept is not difficult, and it can be built into programming languages (consider for example nesC and TinyOS, which are widely used for programming sensor networks).

It becomes apparent that the avoidance of so many recent improvements in computation is not due to ignorance of those improvements. It is due to a mismatch of the core abstractions and the technologies built on those core abstractions. In embedded software, time matters. In the 20th century abstractions of computing, time is irrelevant. In embedded software, concurrency and interaction with hardware are intrinsic, since embedded software engages the physical world in non-trivial ways (more than keyboards and screens). The most influential 20th century computing abstractions speak only weakly about concurrency, if at all. Even the core 20th century notion of "computable" is at odds with the requirements of embedded software. In this notion, useful computation terminates, but termination is undecidable. In embedded software, termination is failure, and yet to get predictable timing, subcomputations must decidably terminate.

Embedded systems are integrations of software and hardware where the software reacts to sensor data and/or issues commands to actuators. The physical system is an integral part of the design and the software must be conceptualized to operate in concert with that physical system. Physical systems are intrinsically concurrent and temporal. Actions and reactions happen simultaneously and over time, and the metric properties of time are an essential part of the behavior of the system. Prevailing software methods abstract away time, replacing it with ordering. In imperative languages such as C, C++, and Java, the order of actions is defined by the program, but not their timing. This prevailing imperative abstraction is overlaid with another, that of threads or processes, typically provided by the operating system, but occasionally by the language (as in Java).

The lack of timing in the core abstraction is a flaw, from the perspective of embedded software, and threads as a concurrency model are a poor match for embedded systems. They are mainly focused on providing an illusion of parallelism in fundamentally sequential models, and they work well only for modest levels of concurrency or for highly decoupled systems that are sharing resources, where best-effort scheduling policies are sufficient. Indeed, several recent innovative embedded software frameworks, such as Simulink (from The MathWorks), nesC and TinyOS (from Berkeley), and Lustre/SCADE (from Esterel Technologies) are concurrent programming languages with no threads or processes in the programmer's model.

Embedded software systems are generally held to a much higher reliability standard than general purpose software. Often, failures in the software can be life threatening (e.g., in avionics and military systems). The prevailing concurrency model in general purpose software that is based on threads does not achieve adequate reliability. In this prevailing model, interaction between threads is extremely difficult for humans to understand. Although it is arguable that concurrent computation is inherently complex, threads make it far more complex because between any two atomic operations (a concept that is rarely well defined), any part of the state of the system can change. The basic techniques for controlling this interaction use semaphores and mutual exclusion locks, methods that date back to the 1960s. Many uses of these techniques lead to deadlock or livelock. In general-purpose computing, this is inconvenient, and typically forces a restart of the program (or even a reboot of the machine). However, in embedded software, such errors can be far more than inconvenient. Moreover, software is often written without sufficient use of these interlock mechanisms, resulting in race conditions that yield nondeterministic program behavior. In practice, errors due to misuse (or no use) of semaphores and mutual exclusion locks are extremely difficult to detect by testing. Code can be exercised for years before a design flaw appears. Static analysis techniques can help (e.g. Sun Microsystems' LockLint), but these methods are often thwarted by conservative approximations and/or false positives, and they are not widely used in practice.

It can be argued that the unreliability of multi-threaded programs is due at least in part to inadequate software engineering processes. For example, better code reviews, better specifications, better compliance testing, and better planning of the development process can help solve the problems. It is certainly true that these techniques can help. However, programs that use threads can be extremely difficult for programmers to understand. If a program is incomprehensible, then no amount of process improvement will make it reliable. Formal methods can help detect flaws in threaded programs, and in the process can improve the understanding that a designer has of the behavior of a complex program. But if the basic mechanisms fundamentally lead to programs that are difficult to understand, then these improvements will fall short of delivering reliable software. Incomprehensible software will always be unreliable software.

Prevailing industrial practice in embedded software relies on bench testing for concurrency and timing properties. This has worked reasonably well, because programs are small, and because the software gets encased in a box with no outside connectivity that can alter the behavior of the software. However, applications today demand that

embedded systems be feature-rich and networked, so bench testing and encasing become inadequate. In a networked environment, it becomes impossible to test the software under all possible conditions, because the environment is not known. Moreover, general-purpose networking techniques themselves make program behavior much more unpredictable.

What would it take to achieve concurrent and networked embedded software that was absolutely positively on time (say, to the precision and reliability of digital logic)? Unfortunately, everything would have to change. The core abstractions of computing need to be modified to embrace time. Computer architectures need to be changed to deliver precisely timed behaviors. Networking techniques need to be changed to provide time concurrence. Programming languages have to change to embrace time and concurrency in their core semantics. Operating systems have to change to rely less on priorities to (indirectly) specify timing requirements. Software engineering methods need to change to specify and analyze the temporal dynamics of software. And the traditional boundary between the operating system and the programming language needs to be rethought. What is needed is nearly a reinvention of computer science.

Fortunately, there is quite a bit to draw on. To name a few examples, architecture techniques such as software-managed caches promise to deliver much of the benefit of memory hierarchy without the timing unpredictability. Operating systems such as TinyOS provide simple ways to create thin wrappers around hardware, and with nesC, alter the OS/language boundary. Programming languages such as Lustre/SCADE provide understandable and analyzable concurrency. Embedded software languages such as Simulink provide time in their semantics. Network time synchronization methods such as IEEE 1588 provide time concurrence at resolutions (tens of nanoseconds) far finer than any processor or software architectures can deal with today. The time is ripe to pull these techniques together and build the 21st Century (Embedded) Computer Science.

Thanks to helpful comments from Elaine Cheong and Douglas Niehaus.

[50] Balance between Formal and Informal Methods, Engineering and Artistry, Evolution and Rebuild

E. A. Lee, Technical Memorandum UCB/ERL M04/19 /, University of California, Berkeley, July 4, 2004.

Abstract: This paper is the result of a workshop entitled "Software Reliability for FCS" that was organized by the Army Research Office, held on May 18-19, 2004, and hosted by: Institute for Software Integrated Systems (ISIS), Vanderbilt University. I was given the charge of leading one of four topic areas, and was assigned the title. This is my summary of the results of the workshop on this topic.

It may well be that established approaches to software engineering will not be sufficient to avert a software disaster in FCS and similarly ambitious, software-intensive efforts. This topic examines the tension between informal methods, particularly those that focus on the human, creative process of software engineering and the management of that

process, and formal methods, specifically those that rely on mathematically rooted systems theories and semantic frameworks. It is arguable that, as these approaches are construed today by their respective (largely disjoint) research communities, neither offers much hope of delivering reliable FCS software. Although certainly these communities have something to offer, the difficulties may be more deeply rooted than either approach can address. In this workshop, we took an aggressive stand that there are problems in software that are intrinsically unsolvable with today's software technology. This stand asserts that no amount of process will fix the problems because the problems are not with the process, and that today's formal techniques cannot solve the problem as long as they remain focused on formalizing today's software technologies. A sea change in the underlying software technology could lead to more effective informal and formal methods. What form could that take?

[51] Concurrent Models of Computation for Embedded Software

E. A. Lee, Technical Memorandum, University of California, Berkeley, UCB/ERL M05/2/, January 4, 2005.

Abstract: This document collects the lecture notes that I used when teaching EECS 290n in the Fall of 2004. This course is an advanced graduate course with a nominal title of Advanced Topics in Systems Theory. This instance of the course studies models of computation used for the specification and modeling of concurrent real-time systems, particularly those with relevance to embedded software. Current research and industrial approaches are considered, including real-time operating systems, process networks, synchronous languages (such as used in SCADE, Esterel, and Statecharts), timed models (such as used in Simulink, Giotto, VHDL, and Verilog), and dataflow models (such as a used in Labview and SPW). The course combines an experimental approach with a study of formal semantics. The objective is to develop a deep understanding of the wealth of alternative approaches to managing concurrency and time in software.

The experimental portion of the course uses Ptolemy II as the software laboratory. The formal semantics portion of the course builds on the mathematics of partially ordered sets, particularly as applied to prefix orders and Scott orders. It develops a framework for models of computation for concurrent systems that uses partially ordered tags associated with events. Discrete-event models, synchronous/reactive languages, dataflow models, and process networks are studied in this context. Basic issues of computability, boundedness, determinacy, liveness, and the modeling of time are studied. Classes of functions over partial orders, including continuous, monotonic, stable, and sequential functions are considered, as are semantics based on fixed-point theorems.

[52] What are the Key Challenges in Embedded Software?

E. A. Lee, Guest Editorial in System Design Frontier, Shanghai Hometown Microsystems Inc., Volume 2, Number 1, January 2005.

Abstract: Embedded software has traditionally been thought of as "software on small computers." In this traditional view, the principal problem is resource limitations (small memory, small data word sizes, and relatively slow clocks). Solutions emphasize efficiency; software is written at a very low level (in assembly code or C), operating systems with a rich suite of services are avoided, and specialized computer architectures such as programmable DSPs and network processors are developed to provide hardware support for common operations. These solutions have defined the practice of embedded software design and development for the last 25 years or so.

Of course, thanks to the semiconductor industry's ability to follow Moore's law, the resource limitations of 25 years ago should have almost entirely evaporated today. Why then has embedded software design and development changed so little? It may be that extreme competitive pressure in products based on embedded software, such as consumer electronics, rewards only the most efficient solutions. This argument is questionable, however, since there are many examples where functionality has proven more important than efficiency. We will argue that resource limitations are not the only defining factor for embedded software, and may not even be the principal factor.

Resource limitations are an issue to some degree with almost all software. So generic improvements in software engineering should, in theory, also help with embedded software. There are several hints, however, that embedded software is different in fundamental ways. For one, object-oriented techniques such as inheritance, dynamic binding, and polymorphism are rarely used in practice with embedded software development. In another example, processors used for embedded systems often avoid the memory hierarchy techniques that are used in general purpose processors to deliver large virtual memory spaces and faster execution using caches. In a third example, automated memory management, with allocation, deallocation, and garbage collection, are largely avoided in embedded software. To be fair, there are some successful applications of these technologies in embedded software, such as the use of Java in cell phones, but their application remains limited and is largely providing services that are actually more akin to general-purpose software applications (such as database services in cell phones).

Embedded systems are integrations of software and hardware where the software reacts to sensor data and issues commands to actuators. The physical system is an integral part of the design and the software must be conceptualized to operate in concert with that physical system. Physical systems are intrinsically concurrent and temporal. Actions and reactions happen simultaneously and over time, and the metric properties of time are an essential part of the behavior of the system. Prevailing software methods abstract away time, replacing it with ordering. In imperative languages such as C, C++, and Java, the order of actions is defined by the program, but not their timing. This prevailing imperative abstraction is overlaid with another, that of threads or processes, typically provided by the operating system, but occasionally by the language (as in Java).

The lack of timing in the core abstraction is a flaw, from the perspective of embedded software, and threads as a concurrency model are a poor match to embedded systems. They are mainly focused on providing an illusion of concurrency in fundamentally sequential models, and they work well only for modest levels of concurrency or for highly decoupled systems that are sharing resources, where best-effort scheduling policies are sufficient. Indeed, several recent innovative embedded software frameworks, such as Simulink (from the MathWorks), TinyOS (from Berkeley), and SCADE (from Esterel Technologies) have no threads or processes.

Embedded software systems are generally held to a much higher reliability standard than general purpose software. Often, failures in the software can be life threatening (e.g., in avionics and military systems). The prevailing concurrency model based on threads does not achieve adequate reliability. In this prevailing model, interaction between threads is extremely difficult for humans to understand. The basic techniques for controlling this interaction use semaphores and mutual exclusion locks, methods that date back to the 1960s. These techniques often lead to deadlock or livelock conditions, where all or part of a program cannot continue executing. In general-purpose computing, this is inconvenient, and typically forces a restart of the program (or even a reboot of the machine). However, in embedded software, such errors can be far more than inconvenient. Moreover, software is often written without sufficient use of these interlock mechanisms, resulting in race conditions that yield nondeterministic program behavior.

In practice, errors due to misuse (or no use) of semaphores and mutual exclusion locks are extremely difficult to detect by testing. Code can be exercised for years before a design flaw appears. Static analysis techniques can help (e.g. Sun Microsystems' LockLint), but these methods are often thwarted by conservative approximations and/or false positives, and they are not widely used in practice.

It can be argued that the unreliability of multi-threaded programs is due at least in part to inadequate software engineering processes. For example, better code reviews, better specifications, better compliance testing, and better planning of the development process can help solve the problems. It is certainly true that these techniques can help. However, programs that use threads can be extremely difficult for programmers to understand. If a program is incomprehensible, then no amount of process improvement will make it reliable. Formal methods can help detect flaws in threaded programs, and in the process can improve the understanding that a designer has of the behavior of a complex program. But if the basic mechanisms fundamentally lead to programs that are difficult to understand, then these improvements will fall short of delivering reliable software.

The key challenge in embedded software is to invent (or apply) abstractions that yield more understandable programs that are both concurrent and timed. These abstractions will be very different from those widely used for the design and development of general-purpose software.

[53] Classes and Subclasses in Actor-Oriented Design

E. A. Lee, S. Neuendorffer, invited paper, Conference on Formal Methods and Models for Codesign/ (MEMOCODE), San Diego, CA, USA, June 22-25, 2004.

Abstract: Actor-oriented languages provide a component composition methodology that emphasizes concurrency. The interfaces to actors are *parameters* and *ports* (vs. members and methods in object-oriented languages). Actors interact with one another through their ports via a messaging schema that can follow any of several concurrent semantics (vs. procedure calls, with prevail in OO languages). Domain-specific actor-oriented languages and frameworks are common (e.g. Simulink, LabVIEW, and many others). However, they lack many of the modularity and abstraction mechanisms that programmers have become accustomed to in OO languages, such as classes, inheritance, interfaces, and polymorphism. This extended abstract shows the form that such mechanisms might take in AO languages. A prototype of these mechanisms realized in Ptolemy II is described.

[54] Concurrent Models of Computation for Embedded Software

E. A. Lee, S. Neuendorffer, Technical Memorandum UCB/ERL M04/26/, University of California, Berkeley, July 22, 2004.

Abstract: The prevailing abstractions for software are better suited to the traditional problem of computation, namely transformation of data, than to the problems of embedded software. These abstractions have weak notions of concurrency and the passage of time, which are key elements of embedded software. Innovations such as nesC/TinyOS (developed for programming very small programmable sensor nodes called motes), Click (created to support the design of software-based network routers), Simulink with Real-Time Workshop (created for embedded control software), and Lustre/SCADE (created for safety-critical embedded software) offer abstractions that address some of these issues and differ significantly from the prevailing abstractions in software engineering. This paper surveys some of the abstractions that have been explored.

[55] Operational Semantics of Hybrid Systems

E. A. Lee, H. Zheng, invited paper, In Proc. Hybrid Systems: Computation and Control (HSCC) LNCS TBD, Zurich, Switzerland, March 9-11, 2005.

Abstract: This paper discusses an interpretation of hybrid systems as executable models. A specification of a hybrid system for this purpose can be viewed as a program in a domain-specific programming language. We describe the semantics of HyVisual, which is such a domain-specific programming language. The semantic properties of such a language affect our ability to understand, execute, and analyze a model. We discuss several semantic issues that come in defining such a programming language, such as the interpretation of discontinuities in continuous-time signals, and the interpretation of discrete-event signals in hybrid systems, and the consequences of numerical ODE solver techniques. We describe the solution in HyVisual by giving its operational semantics.

[56] Scientific Workflow Management and the KEPLER System

B. Ludäscher, I. Altintas, C. Berkley, D. Higgins, E. Jaeger, M. Jones, E. A. Lee, J. Tao, Y. Zhao, Concurrency & Computation: Practice & Experience, (in publication), draft version, March 2005.

Abstract: Many scientific disciplines are now data and information driven, and new scientific knowledge is often gained by scientists putting together data analysis and knowledge discovery "pipelines." A related trend is that more and more scientific communities realize the benefits of sharing their data and computational services, and are thus contributing to a distributed data and computational and community infrastrucutre (a.k.a. "the Grid"). However, this infrasctructure is only a means to an end and scientists ideally should be bothered little with its existence. The goal is for scientists to focus on development and use of what we call *scientific workflows*. These are networks of analytical steps that may involve, e.g., database access and querying steps, data analysis and mining steps, and many other steps including computationally intensive jobs on high performance cluster computers. In this paper we describe characteristics of and requirements for scientific workflows as identified in a number of our application projects. We then elaborate on KEPLER, a particular scientific workflow system, currently under development across a number of scientific data management projects. We describe some key features of KEPLER and its underlying PTOLEMY II system, planned extensions, and areas of future research. KEPLER is a community-driven, open source project, and we always welcome related projects and new contributors to join.

[57] Automatic Verification of Component-based Real-time CORBA Applications

G. Madl, S. Abdelwahed, G. Karsai, In Proc. 25th IEEE International Real-Time Systems Symposium (RTSS'04), Lisbon, Portugal, Dec. 2004, pp. 231—240.

Abstract: Distributed real-time embedded (DRE) systems often need to satisfy various time, resource and faulttolerance constraints. To manage the complexity of scheduling these systems many methods use Rate Monotonic Scheduling assuming a time-triggered architecture. This paper presents a method that captures the reactive behavior of complex time- and event-driven systems, can provide simulation runs and can provide exact characterization of timed properties of component-based DRE applications that use the publisher/subscriber communication pattern. We demonstrate our approach on real-time CORBA avionics applications.

[58] Verifying Distributed Real-Time Properties of Embedded Systems via Graph Transformations and Model Checking

G. Madl, S. Abdelwahed, D., Real-Time Systems Journal, (in publication), 2005.

Abstract: quality of service (QoS) needs of distributed real-time embedded (DRE) systems. However, component middleware also introduces challenges for DRE system developers, such as evaluating the predictability of DRE system behavior and choosing the right design alternatives before committing to a specific platform. Model-based technologies help address these issues by enabling design-time analysis and providing the means to automate the development, configuration, and integration of component-based

DRE systems. This paper provides three contributions to research on model-based design and analysis of component-based DRE systems. First, we apply model checking techniques to DRE design models using model transformations to verify key QoS properties of component-based DRE systems developed using Real-time CORBA. Second, we implemented a property-preserving model-transformation method and used it to define formal semantics to component-based modeling languages. Third, we develop a formal description of the Boeing Bold Stroke architecture from which abstract behavioral models can be constructed and verified. Our results show that model-based techniques enable design-time analysis of timed properties and can be applied to effectively predict, simulate , and verify the event-driven behavior of component-based DRE systems.

[59] Shooter Localization in Urban Terrain

M. Maroti, G. Simon, A. Ledeczi, J. Sztipanovits, IEEE Computer, pp. 60-61, August, 2004.

Abstract: The paper describes PinPtr, an acoustic sensor network-based shooter localization system. Instead of using a few expensive acoustic sensors, a low-cost ad-hoc acoustic sensor network measures both the muzzle blast and shock wave to accurately determine the location of the shooter and the trajectory of the bullet. The basic idea is simple: using the arrival times of the acoustic events at different sensor locations, the shooter position is calculated using the speed of sound and the location of the sensors. The robust sensor fusion algorithm, which is running on the base station, is based on a search on a hyper-surface defined by a consistency function The consistency function, which provides the number of sensor measurements consistent with hypothetical shooter positions and shot time, automatically classifies measurements and eliminates those, which are erroneous or the result of multipath effects. The highly redundant ad-hoc sensor field ensures that enough good measurements are left to determine the shooter's position. The global maximum of the surface, corresponding to the shooter position, is guaranteed to be found by a fast search algorithm.

[60] Fault Tolerant Data Flow Modeling Using the Generic Modeling Environment

M. L. McKelvin, Jr, J. Sprinkle, C. Pinello, A. Sangiovanni-Vincentelli, 12th Annual IEEE International Conference and Workshop on the Engineering of Computer Based Systems, Greenbelt, Maryland, Apr. 4--5, 2005, pp. 229–235.

Abstract: Designing embedded software for safety-critical, real-time feedback control applications is a complex and error prone task. Fault tolerance is an important aspect of safety. In general, fault tolerance is achieved by duplicating hardware components, a solution that is often more expensive than needed. In particular applications, such as automotive electronics, a subset of the functionalities has to be guaranteed while others are not crucial to the safety of the operation of the vehicle. In this case, we must make sure that this subset is operational under the potential faults of the architecture. A model of computation called Fault-Tolerant Data Flow (FTDF) was recently introduced to describe at the highest level of abstraction of the design the fault tolerance requirements on the functionality of the system. Then, the problem of implementing the system efficiently on a platform consists of finding a mapping of the FTDF model on the

components of the platform. A complete design flow for this kind of application requires a user-friendly graphical interface to capture the functionality of the systems with the FTDF model, algorithms for choosing an architecture optimally, (possibly automatic) code generation for the parts of the system to be implemented in software and verification tools. In this paper, we use the Generic Modeling Environment (GME) developed at Vanderbilt University to design a graphical design capture system and to provide the infrastructure for automatic code generation. The design flow is embedded into the Metropolis environment developed at the University of California at Berkeley to provide the necessary verification and analysis framework.

[61] A Visual Language for Describing Instruction Sets and Generating Decoders

T. Meyerowitz, J. Sprinkle, A. Sangiovanni-Vincentelli, 20th Annual ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA), Vancouver, BC, Oct., 25, 2004, pp. 23–32.

Abstract: We detail the syntax and semantics of ISA_ML, a visual modeling language for describing Instruction Set Architectures of microprocessors, and an accompanying tool that takes a description in the language and generates decoders from it in the form of a disassembler and a micro-architectural trace interfacer. The language and tool were built using the Generic Modeling Environment (GME), and leverage the concepts of meta-modeling to increase productivity and to provide extensive error checking to the modeler. Using this tool, we were able to construct a model of significant subsets of the MIPS, ARM, and PowerPC instruction sets each in 8 hours or less. This language can be retargeted for other purposes, such as generating synthesizable instruction decoders.

[62] CCured: Type-Safe Retrofitting of Legacy Software

G. C. Necula, J. Condit, M. Harren, S. McPeak, W. Weimer, ACM Transactions on Programming Languages and Systems, vol. 27, no. 3, May, 2005.

Abstract: This article describes CCured, a program transformation system that adds type safety guarantees to existing C programs. CCured attempts to verify statically that memory errors cannot occur, and it inserts run-time checks where static verification is insufficient. CCured extends C's type system by separating pointer types according to their usage, and it uses a surprisingly simple type inference algorithm that is able to infer the appropriate pointer kinds for existing C programs. CCured uses physical subtyping to recognize and verify a large number of type casts at compile time. Additional type casts are verified using run-time type information. CCured uses two instrumentation schemes, one that is optimized for performance and one in which metadata is stored in a separate data structure whose shape mirrors that of the original user data. This latter scheme allows instrumented programs to invoke external functions directly on the program's data without the use of a wrapper function. We have used CCured on real-world security-critical network daemons to produce instrumented versions without memory-safety vulnerabilities, and we have found several bugs in these programs. The instrumented code is efficient enough to be used in day-to-day operations.

[63] Model-Integrated Computing for Heterogeneous Systems

S. Neema, A. Dixon, T. Bapty, J. Sztipanovits, In Proc. International Conference on Computing, Communications and Control Technologies (CCCT'04), Austin, TX, August 14-17, 2004.

Abstract: Modern embedded and networked embedded system applications are demanding very high performance from systems with minimal resources. These applications must also be flexible to operate in a rapidly changing environment. High performance with limited resources needs application-specific architectures, while flexibility requires adaptation capabilities. Design of these systems creates unique challenges, since the traditional decomposition of the design space to hardware and software components and to functional and non-functional requirements do not give acceptable performance. Model-Integrated Computing (MIC) is an emerging design technology, which integrates all essential aspects of system design in a general, but highly customizable framework. This paper provides an overview of MIC and shows its application in the design of reconfigurable processing system.

[64] Actor-Oriented Metaprogramming

S. Neuendorffer, PhD Thesis, University of California, Berkeley, December 21, 2004.

Abstract: Robust design of concurrent systems is important in many areas of engineering, from embedded systems to scientific computing. Designing such systems using dataflow-oriented models can expose large amounts of concurrency to system implementation. Utilizing this concurrency effectively enables distributed execution and increased throughput, or reduced power usage at the same throughput. Code generation can then be used to automatically transform the design into an implementation, allowing design refactoring at the dataflow level and reduced design time over hand implementation.

This thesis focuses particularly on the benefits and disadvantages that arise when constructing models from generic, parameterized, dataflow-oriented components called *actors*. A designer can easily reuse actors in different models with different parameter values, data types, and interaction semantics. Additionally, during execution of a model actors can be reconfigured by changing their connections or assigning new parameter values. This form of reconfiguration can conveniently represent adaptive systems, systems with multiple operating modes, systems without fixed structure, and systems that control other systems. Ptolemy II is a Java-based design environment that supports the construction and execution of hierarchical, reconfigurable models using actors.

Unfortunately, allowing unconstrained reconfiguration of actors can sometimes cause problems. If a model is reconfigured, it may no longer accurately represent the system being modeled. Reconfiguration may prevent the application of static scheduling analysis to improve execution performance. In systems with data type parameters, reconfiguration may prevent static analysis of data types, eliminating an important form of error

detection. In such cases, it is therefore useful to limit which parameters or structures in a model can be reconfigured, or when during execution reconfiguration can occur.

This thesis describes a reconfiguration analysis that determines when reconfiguration occurs in a hierarchical model. Given appropriate formulated constraints, the analysis can alert a designer to potential design problems. The analysis is based on a mathematical framework for approximately describing periodic points in the behavior of a model. This framework has a lattice structure that reflects the hierarchical structure of actors in a model. Because of the lattice structure of the framework, this analysis can be performed efficiently. Models of two different systems are presented where this analysis helps verify that reconfiguration does not violate the assumptions of the model.

Run-time reconfiguration of actors not only presents difficulties for a system modeler, but can also impede efficient system implementation. In order to support run-time reconfiguration of actors in Java, Ptolemy II introduces extra levels of indirection into many operations. The overhead from this indirection is incurred in all models, even if a particular model does not use reconfiguration.

In order to remove the indirection overhead, we have developed a system called Copernicus which transforms a Ptolemy II model into self-contained Java code. In performing this transformation the Java code for each actor is specialized to its usage in a particular model. As a result, indirection overhead only remains in the generated code if it is required by reconfiguration in the model. The specialization is guided by various types of static analysis, including data type analysis and analysis of reconfiguration. In certain cases, the generated code runs 100 times faster and with almost no memory allocation, compared to the same model running in a Ptolemy II simulation. For small examples, performance close to handwritten Java code has been achieved.

[65] Modeling Real-World Control Systems: Beyond Hybrid Systems

S. Neuendorffer, In Proc. Winter Simulation Conference (WSC), Washington, DC, USA, December 5--8, 2004.

Abstract: Hybrid system modeling refers to the construction of system models combining both continuous and discrete dynamics. These models can greatly reduce the complexity of a physical system model by abstracting some of the continuous dynamics of the system into discrete dynamics. Hybrid system models are also useful for describing the interaction between physical processes and computational processes, such as in a digital feedback control system. Unfortunately, hybrid system models poorly capture common software architecture design patterns, such as threads, mobile code, safety, and hardware interfaces. Dealing effectively with these practical software issues is crucial when designing real-world systems. This paper presents a model of a complex control system that combines continuous-state physical system models with rich discrete-state software models in a disciplined fashion. We show how expressive modeling using multiple semantics can be used to address the design difficulties in such a system.

[66] Automated Task Allocation for Network Processors

W. Plishker, K. Ravindran, N. Shah, K. Keutzer. In Proc. Network System Design Conference, October, 2004, pp. 235-245.

> **Abstract**: Network processors have great potential to combine high performance with increased flexibility. These multiprocessor systems consist of programmable elements, dedicated logic, and specialized memory and interconnection networks. However, the architectural complexity of the systems makes programming difficult. Programmers must be able to productively implement high performance applications for network processors to succeed. Ideally, designers describe applications in a domain specific language (DSL). DSLs expedite the development process by providing component libraries, communication and computation semantics, visualization tools, and test suites for an application domain. An integral aspect of mapping applications described in a DSL to network processors is allocating computational tasks to processing elements. We formulate this task allocation problem for a popular network processor, the Intel IXP1200. This method proves to be computationally efficient and produces results that are within 5% of aggregate egress bandwidths achieved by hand-tuned implementations on two representative applications: IPv4 forwarding and DiffServ.

[67] Designing Distributed Diagnosers for Complex Physical Systems

I. Roychoudhury, G. Biswas, X. Koutsoukos and S. Abdelwahed, Intl. Workshop on Principles of Diagnosis (DX-05), (in publication) Monterey, CA, June 2005.

> **Abstract:** Online diagnosis methods require large computationally expensive diagnosis tasks to be decomposed into sets of smaller tasks so that time and space complexity constraints are not violated. This paper defines the distributed diagnosis problem in the Transcend qualitative diagnosis framework, and then develops heuristic algorithms for generating a set of local diagnosers that solve the global diagnosis problem without a coordinator. Two versions of the algorithm are discussed. The time complexity and optimality of these algorithms are compared and validated through experimental results.

[68] A Distributed Algorithm for Acoustic Localization Using a Distributed Sensor Network

P. Schmidt, I. Amundson, K.D. Frampton, Journal of the Acoustical Society of America, Vol. 115, No. 5, Pt. 2, pp. 2578, 2004.

> **Abstract:** An acoustic source localization algorithm has been developed for use with large scale sensor networks using a decentralized computing approach. This algorithm, based on a time delay of arrival (TDOA) method, uses information from the minimum number of sensors necessary for an exactly determined solution. Since the algorithm is designed to run on computational devices with limited memory and speed, the complexity of the computations has been intentionally limited. The sensor network consists of an array of battery operated COTS Ethernet ready embedded systems with an integrated microphone as a sensor. All solutions are calculated as distinct values, and the same TDOA method used for solution is applied for ranking the accuracy of an individual solution. Repeated for all combinations of sensor nodes, solutions with accuracy

equivalent to complex array calculations are obtainable. Effects of sensor placement uncertainty and multipath propagation are quantified and analyzed, and a comparison to results obtained in the field with a large array and a centralized computing capability using a complex, memory intensive algorithm is included.

[69] Optimal Control for a class of Stochastic Hybrid Systems

L. Shi, A. Abate, S. Sastry, In Proc. International Conference on Decision and Control, Atlantis, Bahamas, December 2004.

Abstract: In this paper, an optimal control problem over a "hybrid Markov Chain" (hMC) is studied. A hMC can be thought of as a traditional MC with continuous time dynamics pertaining to each node; from a different perspective, it can be regarded as a class of hybrid system with random discrete switches induced by an *embedded MC*. As a consequence of this setting, the index to be maximized, which depends on the dynamics, is the expected value of a non deterministic cost function. After obtaining a closed form for the objective function, we gradually suggest how to device a computationally tractable algorithm to get to the optimal value. Furthermore, the complexity and rate of convergence of the algorithm is analyzed. Proofs and simulations of our results are provided; moreover, an applicative and motivating example is introduced.

[70] Generative Components for Hybrid Systems Tools

J. Sprinkle, Journal of Object Technology, vol. 4, no. 3, April 2005, Special issue: 6th GPCE Young Researchers Workshop 2004, pp. 35-39, Available at *http://www.jot.fm/issues/issue_2005_04/article5*

Abstract: Generative techniques, while normally associated with programming languages or code generation, may also be used to produce non-executable artifacts (e.g., configuration or toolchain artifacts). Coupled with domain-specific modeling, generative techniques provide a way to consolidate toolchains and complex domains into a relatively compact space, by providing generators that produce artifacts in the appropriate semantic domain. This paper describes the motivation and usage of one such environment, in the domain of hybrid systems, as well as discussion and goals for future research.

[71] On the Partitioning of Syntax and Semantics For Hybrid Systems Tools

J. Sprinkle, A. D. Ames, S. S. Sastry, 44th IEEE Conference on Decision and Control and European Control Conference ECC 2005 (CDC-ECC'05), (submitted for publication), Seville, Spain, Dec., 12--15, 2005.

Abstract: Interchange formats are notoriously difficult to finish. That is, once one is developed, it is highly nontrivial to prove (or disprove) generality, and difficult at best to gain acceptance from all major players in the application domain. This paper addresses such a problem for hybrid systems, but not from the perspective of a tool interchange format, but rather that of tool availability in a toolbox. Through the paper we explain why we think this is a good approach for hybrid systems, and we also analyze the domain of hybrid systems to discern the semantic partitions that can be formed to yield a

classification of tools based on their semantics. These discoveries give us the foundation upon which to build semantic capabilities, and to guarantee operational interaction between tools based on matched operational semantics.

[72] A Paradigm for Teaching Modeling Environment Design

J. Sprinkle, J. Davis, G. Nordstrom, 20th Annual ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA), Educators Symposium (Poster Session), ACM, Vancouver, BC, Oct., 24--28, 2004.

Abstract: Model-Integrated Computing (MIC) is a generic term for the practice of coupling models of a complex system with the execution of that system. Although MIC has been in use for years by experts who learn its techniques in an ad hoc manner, it is only recently that system modeling-as a science-has begun to be taught as a subject. This paper presents a paradigm in use in a course designed specifically to teach the design of domain-specific modeling environments. The work describes how this paradigm grows throughout the course to complement the expanding nomenclature, mapping technologies, visitor and object-oriented technologies, and design decisions encountered by the students.

[73] Deciding to Land a UAV Safely in Real Time

J. Sprinkle, J. M. Eklund, S. S. Sastry, In Proc. American Control Conference (ACC) 2005, (in publication), Portland, OR, Jun., 8--10, 2005.

Abstract: The difficulty of autonomous free-flight of a fixedwing UAV is trivial when compared to that of takeoff and landing. There is an even more marked difference when deciding whether or not a UAV can capture or recapture a certain trajectory, since the answer depends on the operating ranges of the aircraft. A common example of requiring this calculation, from a military perspective, is the determination of whether or not an aircraft can capture a landing trajectory (i.e., glideslope) from a certain initial state (velocity, position, etc.). As state dimensions increase, the time to calculate the decision grows exponentially. This paper describes how we can make this decision at flight time, and guarantee that the decision will give a safe answer before the state changes enough to invalidate the decision. We also describe how the computations should be formulated, and how the partitioning of the state-space can be done to reduce the computation time required. Flight testing was performed with our design, and results are given.

[74] Encoding Aerial Pursuit/Evasion Games with Fixed Wing Aircraft into a Nonlinear Model Predictive Tracking Controller

J. Sprinkle, J. M. Eklund, H. J. Kim, S. S. Sastry, IEEE Conference on Decision and Control, pp. 2609--2614 , Dec., 2004.

Abstract: Unmanned Aerial Vehicles (UAVs) have shown themselves to be highly capable in intelligence gathering, as well as a possible future deployment platform for munitions. Currently UAVs are supervised or piloted remotely, meaning that their

behavior is not autonomous throughout the flight. For uncontested missions this is a viable method; however, if confronted by an adversary, UAVs may be required to execute maneuvers faster than a remote pilot could perform them in order to evade being targeted. In this paper we give a description of a non-linear model predictive controller in which evasive maneuvers in three dimensions are encoded for a fixed wing UAV for the purposes of this pursuit/evasion game.

[75] Toward Design Parameterization Support for Model Predictive Control

J. Sprinkle, J. M. Eklund, S. S. Sastry, IEEE 4th International Conference on Intelligent Systems Design and Application, IEEE, IEEE Press, Budapest, Hungary, Aug., 26--28, 2004.

Abstract: Research into the autonomous behavior of Unmanned Aerial Vehicles (UAVs) requires concise and dependable specification techniques in order to provide behavioral descriptions for the controllers of these aircraft. Practical issues with autonomous aircraft involve the safety and reliability of the controller (e.g., guarantee of stability), as well as verification of the high-level intention of the autonomous behavior. Since most behaviors are implemented orthogonally to their high-level specification, improvements in the ability to rapidly specify the models that govern the behavior are certainly welcome. This paper describes the framework for decreasing the abstraction required to specify the behavior of an autonomous controller implemented through model predictive control.

[76] A Domain-Specific Visual Language for Domain Model Evolution

J. Sprinkle, G. Karsai, J. Vis. Lang. and Comp., vol. 15, no. 3-4, pp. 291-307, Jun., 2004.

Abstract: Domain-specific visual languages (DSVLs) are concise and useful tools that allow the rapid development of the behavior and/or structure of applications in well-defined domains. These languages are typically developed specifically for a domain, and have a strong cohesion to the domain concepts, which often appear as primitives in the language. The strong cohesion between DSVL language primitives and the domain is a benefit for development by domain experts, but can be a drawback when the domain evolves---even when that evolution appears insignificant. This paper presents a domain-specific visual language developed expressly for the evolution of domain-specific visual languages, and uses concepts from graph-rewriting to specify and carry out the transformation of the models built using the original DSVL.

[77] Using the Hybrid Systems Interchange Format to Input Design Models to Verification & Validation Tools

J. Sprinkle, O. Shakernia, R. Miller, S. S. Sastry, IEEE Aerospace Conference, Big Sky, MT, Mar., 2005.

Abstract: The domain of hybrid systems lacks the set of mature tools which can reliably apply verification and validation (V&V) techniques for all kinds of systems. Currently, no single tool supports analysis, simulation, verification, validation, and code synthesis of controllers for hybrid systems. As such, it becomes necessary to depend on several tools

for analysis of different aspects of the system. This paper , describes the utilization of the definition of a system in more than one toolsuite, while reducing the required effort for the same engineers that design the controllers to interface to the V&V toolsuites.

[78] Platform Modeling and Model Transformations for Analysis

T. Szemethy, G. Karsai, Journal of Universal Computer Science, vol. 10, no. 10, pp 1383-1406, 2004.

Abstract: The model-based approach to the development of embedded systems relies on the use of explicit models in the design process. If these models faithfully represent the components of the system with respect to their properties as well as their interactions, then they can be used to predict the dynamic behavior of the system under construction. In this paper we argue for modeling the execution platform that facilitates the component interactions, and show how models of the application and the knowledge of the platform can be used to translate system configurations into another abstract formalism (timed automata, in our case) that allows system verification through model checking.

[79] Introducing Embedded Software and Systems Education and Advanced Learning Technology in an Engineering Curriculum

J. Sztipanovits, G. Biswas, K. Frampton, A. Gokhale, L. Howard, G. Karsai, T. J. Koo, X. Koutsoukos, D. Schmidt, ACM Transactions on Embedded Systems, (in publication), 2005.

Abstract: Embedded software and systems are at the intersection of electrical engineering, computer engineering, and computer science, with increasing importance in mechanical engineering. Despite the clear need for knowledge of systems modeling and analysis (covered in electrical and other engineering disciplines) and analysis of computational processes (covered in computer science), few academic programs have integrated the two disciplines into a cohesive program of study. This paper describes the efforts conducted at Vanderbilt University to establish a curriculum that addresses the needs of embedded software and systems. Given the compartmentalized nature of traditional engineering schools, where each discipline has an independent program of study, we have had to devise innovative ways to bring together the two disciplines. The paper also describes our current efforts in using learning technology to construct, manage, and deliver sophisticated computer-aided learning modules that can supplement the traditional course structure in the individual disciplines through out-of-class and in-class use.

[80] Experiments on the Decentralized Vibration Control with Networked Embedded Systems

T. Tao, K.D. Frampton, Presented at the 2004 ASME International Mechanical Engineering Conference and Exposition, Anaheim CA, November 2004.

Abstract: The early promise of centralized active control technologies to improve the performance of large scale, complex systems has not been realized largely due to the

inability of centralized control systems to "scale up"; that is, the inability to continue to perform well when the number of sensors and actuators becomes large. Now, recent advances in Micro-electro-mechanical systems (MEMS), microprocessor developments and the breakthroughs in embedded systems technologies, decentralized control systems may see these promises through. A networked embedded system consists of many nodes that possess limited computational capability, sensors, actuators and the ability to communicate with each other over a network. The aim of this decentralized control system is to control the vibration of a structure by using such an embedded system backbone. The key attributes of such control architectures are that it be scalable and that it be effective within the constraints of embedded systems. Toward this end, the decentralized vibration control of a simply supported beam has been implemented experimentally. The experiments demonstrate that the reduction of the system vibration is realized with the decentralized control strategy while meeting the embedded system constraints, such as a minimum of inter-node sensor data communication, robustness to delays in sensor data and scalability.

[81] Proceedings of the 4th OOPSLA Workshop on Domain-Specific Modeling (DSM'04)

J. Tolvanen, J. Sprinkle, M. Rossi, eds., Jyvavaskyla, Finland, 20th Annual ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA), University of Jyvavaskyla, Oct., 2004.

Abstract: Domain-Specific Modeling aims at raising the level of abstraction beyond programming by specifying the solution directly using domain concepts. In a number of cases the final products can be generated from these high-level specifications. This automation is possible because of domain-specificity: both the modeling language and code generators fit to the requirements of a narrow domain only, often in a single company. This is the fourth workshop on Domain-Specific Modeling, following the encouraging experiences from the earlier workshops at past OOPSLA conferences (Tampa 2001, Seattle 2002 and Anaheim 2003). During the time the DSM workshops have been organized, interest in domain-specific modeling languages, metamodeling and supporting tools has increased greatly.

[82] Towards Generation of Efficient Transformations

A. Vizhanyo, A. Agrawal, F. Shi, Generative Programming and Component Engineering, 3rd International Conference, October, 2004, LNCS 3286, pp 298-316, 2004.

Abstract: In this paper we discuss efficiency related constructs of a graph rewriting language, called Graph Rewriting and Transformation (GReAT), and introduce a code generator tool, which together provide a programming framework for the specification and efficient realization of graph rewriting systems. We argue that the performance problems frequently associated with the implementation of the transformation can be significantly reduced by partial evaluation and adopting language constructs that allow algorithmic optimizations.

[83] Distributed Control of Thermoelectric Coolers

D. G. Walker, K. D. Frampton, R.D. Harvey, In Proc. 9th Intersociety Conference on Thermal and Thermomechanical Phenomena in Electronic Systems (ITherm), paper no. 151, Las Vegas, NV, June 1-4, 2004, pp. 361--366.

Abstract: Thermoelectric refrigeration has been studied for use in electronics cooling applications. Because of their low efficiency, a significant amount of additional heat is produced that must be removed from the device by passive means. However, even well-designed passive heat removal systems are faced with physical limitations and can not dissipate additional energy. Therefore, thermoelectric coolers often are not a viable alternative to chip cooling. Distributed control refers to a concept where individual devices operate based on their local condition and that of their nearest neighbors. With this approach a large number of sensors and actuators can be bundled with minimal compute power and communication. In the case of electronics cooling, a thermoelectric cooler can be constructed as many separate coolers, each with its own thermocouple and minimal control system. With this architecture, only those regions that generate heat will be cooled thermoelectrically reducing the impact on the heat removal system and increasing the viability of thermoelectric coolers. Through analytic TEC heat models, the present work seeks to evaluate the possibility of using distributed controlled thermoelectric coolers for microelectronics cooling applications. Results indicate that TEC coolers can provide a 2-fold increase in efficiency when distributed control is used for nonuniformly heated chips.

[84] The Design and Application of Structured Types in Ptolemy

Y. Xiong, E.A. Lee, X. Liu, Y. Zhao, L.C. Zhong, IEEE Int. Conf. on Granular Computing (Grc 2005), (in publication), Beijing, China, July 25-27, 2005.

Abstract: Ptolemy II is a component-based design and modeling environment. It has a polymorphic type system that supports both the base types and structured types, such as arrays and records. The base type support was reported in [12]. This paper presents the extensions that support structured types. In the base type system, all the types are organized into a type lattice, and type constraints in the form of inequalities can be solved efficiently over the lattice. We take a hierarchical and granular approach to add structured types to the lattice, and extend the format of inequality constraints to allow arbitrary nesting of structured types. We also analyze the convergence of the constraint solving algorithm on an infinite lattice after structured types are added. To show the application of structured types, we present a Ptolemy II model that implements part of the IEEE 802.11 specifications. This model makes extensive use of record types to represent the protocol messages in the system.

[85] Dynamic Dataflow Modeling in Ptolemy II

G. Zhou, Master's Report, Technical Memorandum No. UCB/ERL M05/2/, University of California, Berkeley, December 21, 2004.

Abstract: Dataflow process networks are a special case of Kahn process networks (PN). In dataflow process networks, each process consists of repeated firings of a dataflow actor, which defines a quantum of computation. Using this quantum avoids the complexities and context switching overhead of process suspension and resumption incurred in most implementations of Kahn process networks. Instead of context switching, dataflow process networks are executed by scheduling the actor firings. This scheduling can be done at compile time for synchronous dataflow (SDF) which is a particularly restricted case with the extremely useful property that deadlock and boundedness are decidable. However, for the most general dataflow, the scheduling has to be done at run time and questions about deadlock and boundedness cannot be statically answered. This report describes and implements a dynamic dataflow (DDF) scheduling algorithm under Ptolemy II framework based on original work in Ptolemy Classic. The design of the algorithm is guided by several criteria that have implications in practical implementation. We compared the performance of SDF, DDF and PN. We also discussed composing DDF with other models of computation (MoC). Due to Turing-completeness of DDF, it is not easy to define a meaningful iteration for a DDF submodel when it is embedded inside another MoC. We provide a suite of mechanisms that will facilitate this process. We give several application examples to show how conditionals, data-dependent iterations, recursion and other dynamic constructs can be modeled in the DDF domain.

2.3. Project Training and Development

As part of setting up CHESS (the new UCB Center for Hybrid and Embedded Software Systems), we have created a CHESS Software Lab, which is focused on supporting the creation of publication-quality software supporting embedded systems design. The lab is physically a room with wireless and wired network connections, a large table for collaborative work, a large format printer (used for UML diagrams and poster preparation), comfortable furniture supporting extended hours of collaborative work, a coffee machine, and a library that inherited a collection of software technology books from the Ptolemy Project. This room is used to promote a local version of the Extreme Programming (XP) software design practice, which advocates pair programming, design reviews, code reviews, extensive use of automated regression tests, and a collaboratively maintained body of code (we use CVS). The room began operation in March of 2003 and has been in nearly constant use for collaborative design work. The principal focus of that work has been on advanced tool architectures for hybrid and embedded software systems design.

2.4. Outreach Activities

Continuing in our mission to build a modern systems science (MSS) with profound implications on the nature and scope of computer science and engineering research, the structure of computer science and electrical engineering curricula, and future industrial practice. This new systems science must pervade engineering education throughout the undergraduate and graduate levels. Embedded software and systems represent a major departure from the current, separated

structure of computer science (CS), computer engineering (CE), and electrical engineering (EE). In fact, the new, emerging systems science reintegrates information and physical sciences. The impact of this change on teaching is profound, and cannot be confined to graduate level.

This year we have continued our work to lay the foundation for a new philosophy of undergraduate teaching at the participating institutions. We also used the summer months to foster appreciation for research in underprivileged and minority students in engineering, by continuing to sponsor and participate in the established REU programs SUPERB-IT at UCB and SIPHER at VU.

We continue the collaboration with San Jose State University to continue to develop the undergraduate embedded control course jointly between Berkeley, Vanderbilt and San Jose State University. Prof. Ping Hsu from San Jose State University teaches the class at both Berkeley and San Jose State.

2.4.1. Curriculum Development for Modern Systems Science (MSS)

Our agenda is to restructure computer science and electrical engineering curricula to adapt to a tighter integration of computational and physical systems. Embedded software and systems represent a major departure from the current, separated structure of computer science (CS), computer engineering (CE), and electrical engineering (EE). In fact, the new, emerging systems science reintegrates information and physical sciences. The impact of this change on teaching is profound, and cannot be confined to graduate level. Based on the ongoing, groundbreaking effort at UCB, we are engaged in retooling undergraduate teaching at the participating institutions, and making the results widely available to encourage critical discussion and facilitate adoption.

We are engaged in an effort at UCB to restructure the undergraduate systems curriculum (which includes courses in signals and systems, communications, signal processing, control systems, image processing, and random processes). The traditional curriculum in these areas is mature and established, so making changes is challenging. We are at the stage of attempting to build faculty consensus for an approach that shortens the pre-requisite chain and allows for introduction of new courses in hybrid systems and embedded software systems.

Undergrad Course Insertion and Transfer

At many institutions, introductory courses are quite large. This makes conducting such a course a substantial undertaking. In particular, the newness of the subject means that there are relatively few available homework and lab exercises and exam questions. To facilitate use of this approach by other instructors, we have engaged technical staff to build web infrastructure supporting such courses. We have built an instructor forum that enables submission and selection of problems from the text and from a library of submitted problems and exercises. A server-side infrastructure generates PDF files for problem sets and solution sets.

The tight integration of computational and physical topics offers opportunities for leveraging technology to illustrate fundamental concepts. We have developed a suite of web pages with applets that use sound, images, and graphs interactively. Our staff has extended and upgraded these applets and created a suite of Powerpoint slides for use by instructors.

We have begun to define an upper division course in embedded software (aimed at juniors and seniors). This new course will replace the control course at the upper division level at San Jose State. We also continued to teach at UC Berkeley the integrated course designed by Prof. Lee, which employs techniques discovered in the hybrid and embedded systems research to interpret traditional signals.

Course: Structure and Interpretation of Signals and Systems (UCB, EECS 20N)

Instructor: Prof. Edward A. Lee
Prof. Pravin Varaiya

This course is an introduction to mathematical modeling techniques used in the design of electronic systems. Signals are defined as functions on a set. Examples include continuous time signals (audio, radio, voltages), discrete time signals (digital audio, synchronous circuits), images (discrete and continuous), discrete event signals, and sequences. Systems are defined as mappings on signals. The notion of state is discussed in a general way. Feedback systems and automata illustrate alternative approaches to modeling state in systems. Automata theory is studied using Mealy machines with input and output. Notions of equivalence of automata and concurrent composition are introduced. Hybrid systems combine time-based signals with event sequences. Difference and differential equations are considered as models for linear, time-invariant state machines. Frequency domain models for signals and frequency response for systems are investigated. Sampling of continuous signals is discussed to relate continuous time and discrete time signals.

Graduate Courses

Several graduate courses were taught in the past year in the area of embedded and hybrid systems, as well as systems modeling. All of these courses are a reflection of the teaching and curriculum goals of the ITR and its affiliated faculty.

Course: Concurrent Models of Computation for Embedded Software (UCB, EECS 290N)

Instructor: Prof. Edward A. Lee

This experimental research course will study models of computation used for the specification and modeling of concurrent real-time systems, particularly those with relevance to embedded software. Current research and industrial approaches will be considered, including real-time operating systems, synchronous languages (such as used in SCADE, Esterel, and Statecharts), timed models (such as used in Simulink, OPNET, NS-2, VHDL, and Verilog), dataflow models (such as a used in Labview and SPW), process networks (such as used in SDL), and software component models (such as nesC/TinyOS, Click, and CORBA). The course will combine an experimental approach with a study of formal semantics. The objective will be to develop a deep understanding of the wealth of alternative approaches to managing concurrency and time in software.

Course: Hybrid Systems: Computation and Control (UCB, EECS 291E/ME 290S)

Instructor: Prof. S. Shankar Sastry
Dr. Jonathan Sprinkle

This course investigates modeling, analysis and verification of various classes of hybrid systems. Special attention is paid to computational and simulation tools for hybrid systems. Applications ranging from networked sensors, power electronics, avionics, and autonomous vehicles will be covered. The course consists of lectures, a handful of homework assignments, and a final project.

Course: Theory of Automata, Formal Languages, and Computation (VU, CS 352)

Instructor: Prof. Xenofon Koutsoukos

Structures, automata and formal language theory, process modeling with finite state automata, supervisory control theory, controllability and supervision, supervisory control under partial observation, modular and hierarchical supervisory control, supervisory control of real-time systems, fault diagnosis of discrete event systems, and modular diagnosis approaches.

Course: Foundations of Hybrid and Embedded Systems (VU, CS 376)

Instructor: Prof. Xenofon Koutsoukos
Prof. T. John Koo

Modeling, analysis, and design of hybrid and embedded systems. Heterogeneous modeling and design of embedded systems using formal models of computation, modeling and simulation of hybrid systems, properties of hybrid systems, analysis methods based on abstractions, reachability, and verification of hybrid systems.

Course: Control Systems I (VU, EECE 257-01)

Instructor: Prof. T. John Koo

Introduction to the theory and design of feedback control systems, steady-state and transient analyses, stability considerations, model representation, state-variable models. Prerequisite: EECE 213 or consent of instructor. The objective of this course is to develop an understanding and the ability to use basic tools for analyzing and designing linear, time-invariant control systems. Materials are primarily taught in classroom setting. A Matlab-based simulation package package Simulink will be used to design, analyze and simulate the control systems. This course is organized around the concepts of linear, time-invariant feedback control theory. Main emphasis will be on the classical methods of control engineering in the frequency and time domains. Also covered are the fundamental concepts of modern control theory including state variable modeling and solution of state equations.

Course: Model Integrated Computing (VU, CS 388 / EE 395)

Instructor: Dr. James Davis
Dr. Greg Nordstrom

Model-Integrated Computing (MIC) addresses the problems of designing, creating, and evolving information systems by providing rich, domain-specific modeling environments including model analysis and model-based program synthesis tools. MIC is used to create and evolve integrated, multiple-aspect models using concepts, relations, and model composition principles routinely used in the specific field, to facilitate systems/software engineering analysis of the models, and to automatically synthesize applications from the models.

Course: Real-Time Systems (VU, EECE 353-01)

Instructor: Prof. Aniruddha Gokhālé
Prof. Douglas Schmidt
Bala Natarajan

This course focuses on the analysis and design of real-time systems. The course covers topics on system modeling using the tagged signal models and timed models of computation, specifications and scheduling techniques for real-time tasks, simulation and verification of real-time systems, software architecture and language for constructing real-time systems. Special attention is paid to computational and simulation tools for real-time systems. Applications ranging from robotics, embedded control systems, drive-by-wire systems, space missions, telecommunication systems, industrial automation, and middleware software systems will be covered.

Course: Automated Verification (VU, EECE 315)

Instructor: Dr. Sherif Abdelwahed

Several notations and methods have been developed to help the designer specify clear and unambiguous system requirements, verify that the requirements are consistent and correct, and verify that the refined design meets its specification. However, these methods are time-consuming and error-prone, and can be applied more effectively if there are tools to check their correctness. The goal of the course is to emphasize formal notations and methods that have tool support. We will cover the basis of underlying theory for the tools.

Course: Automated Verification (VU, EECE 375)

Instructor: Dr. Sherif Abdelwahed

This course provides a detailed coverage of the diagnosis and supervisory control problem for discrete, asynchronous, nondeterministic systems like manufacturing, traffic and communication systems. The underlying theory is developed in an elementary framework of automata and formal languages, and is supported by a software package for creating applications.

2.4.2. SUPERB-IT Program

The Summer Undergraduate Program in Engineering Research at Berkeley–Information Technology (SUPERB-IT) in the Electrical Engineering and Computer Sciences (EECS) Department offers a group of talented undergraduate engineering students the opportunity to gain

research experience. The program's objective is to increase diversity in the graduate school pipeline by affirming students' motivation for graduate study and strengthening their qualifications.

SUPERB-IT participants spent eight weeks at UC Berkeley during the summer of 2004 working on exciting ongoing research projects in information technology with EECS faculty mentors and graduate students. Students who participate in this research apprenticeship explore options for graduate study, gain exposure to a large research-oriented department, and are motivated to pursue graduate study. Additional information about the program can be obtained at:

http://www.eecs.berkeley.edu/Programs/ugrad/superb/superb.html

This ITR project contributed to the support of three SUPERB-IT students in 2004, and organized projects (described below in the abstracts from their papers) in hybrid systems theory, wireless sensor networks, and aerial vehicle simulation. The students were hosted by the Chess center at Berkeley (Center for Hybrid and Embedded Software Systems).

SUPERB-IT participants received a $3,500 stipend, room and board on campus in the International House, and up to $600 for travel expenses. In addition, Chess provided these students with one laptop computer each, configured with appropriate software, plus laboratory facilities for construction of associated hardware.

The students supported at Berkeley in 2004 were Basil Etefia, Rafael Garcia, and Elizabeth Fatusin. The three students worked on individual projects which were tailored to fit their individuation needs, interests, and background, as well as to contribute to the overall goals of the ITR project. Three graduate student mentors facilitated the process, and the operation was coordinated and directed by Professor Shankar Sastry and Dr. Jonathan Sprinkle. Each student was responsible for an individual project, as described below. Their project posters and reports are available at http://chess.eecs.berkeley.edu/projects/ITR/2004/superb.htm.

Project: Routing Protocols for Wireless Sensor Networks

Student: Basil Etefia—*Loyola Marymount University*

> This paper presents an improvement on the implementation of information routing capabilities in ad hoc wireless sensor networks. Improving the protocols used by each sensor node can increase the network's localization and power conservation abilities. Furthermore, using a more efficient algorithm will improve the overall effectiveness of the entire network in terms of power efficiency. Using novel and creative schemes to generate shortest paths for information routing from source to destination nodes, we have implemented an approach to limit the inefficiencies of routing protocols used by sensor networks for information transfer.

Project: Multi-view Configuration of Flight Dynamic Playback

Student: Elizabeth (Beth) Fatusin—*Cornell University, via Ohlone Community College*

> Unmanned Aerial Vehicles (UAVs) have proven to be highly capable in reconnaissance and intelligence gathering, as well as a possible future operation platform for weapons.

Currently UAVs are supervised or piloted remotely, meaning that their behavior is not autonomous throughout the flight. Consequently, instructions are being passed to the UAVs commanding the maneuvers to execute and perform to pursue their adversary or to evade being targeted. In recent years, the security level of these aircrafts has become a critical issue because these aircraft are used for combat missions and other secure operations. For this reason and many others, designing a dual display of the pursuer/evader of UAVs using a flight simulator and displaying in a singular window, appears to be a good idea. In this paper, we give a description of how the data is being collected and transformed into friendly data for viewing its actual maneuvers as a three dimension weapon using a flight simulator and why it is important and safe to do so.

Project: Singular Event Detection

Student: Rafael S. García—*University of Puerto Rico at Mayagüez*

The main objective in the study of event location is to find when the solution to an Ordinary Differential Equation (ODE) crosses a switching surface. Due to the fact that it is almost always impossible to find the exact solution to an arbitrary ODE, the event must be located with a numerical approximation to the exact solution. Since approximate solutions are by nature inexact, the question is whether the real solution displays the same qualitative behavior as its numerical approximation. This raises the need for a number that gauges the validity of this approximation, i.e., a condition number. The purpose of this research is to show how numerical approximations of the solutions of simple systems–linear systems–with simple switching surfaces–hyperplanes–can lead to incorrect event detection. Motivated by these examples, a candidate condition number will be proposed in an attempt to quantify this failure.

Plans for 2005

The SUPERB program is dedicated to providing undergraduate students with the opportunities and experiences of research. At Chess, where our research goals are at the intersection of traditional Electrical Engineering and Computer Science, this includes a wide variety of possible topics including robotics, systems theory, programming, image processing, algorithm development, simulation, and good old mathematics. More importantly, the Chess/ITR philosophy is that new developments in traditional research areas will emerge from interdisciplinary cooperation between Electrical Engineering and Computer Science researchers. To reflect this, and the wide array of possible topics, we have chosen for the 2005 SUPERB program a set of projects which intersect in some areas, and are independent in others, and will allow for inter-student cooperation to devise new solutions to unsolved problems.

It is important to note that several projects fit into more than one category. We believe this is typical of research at Berkeley, and will be reflective of research everyone in the future. Our goal is to use this as a benefit for the student over the summer, where they will learn how to think in terms of multiple goals at once, and achieve results that can be interesting to experts in more than one field. The projects broadly fit into the areas of hybrid systems, sensor networks, computer vision, VLSI design, and simulation.

Six graduate student mentors have been identified to facilitate the process, and the operation is being coordinated and directed by Dr. Jonathan Sprinkle (a postdoc under the supervision of Professor Shankar Sastry). Although the students are working together and interacting extensively, each will be responsible for a single project's full completion. The students and mentors have already been paired for what promises to be an exciting summer.

In addition to the science of research, we will also expose students to the reporting aspects of research. These include the techniques used for writing papers, technical skills such as the use of LaTeX, and the importance of having a stock version of what exactly it is you are working on**.** Specific cross-cutting tasks include LaTeX template design, poster design and best practices, using the Concurrent Versioning System (CVS), participation in a literature reading group, and research reporting through the web.

2.4.3. Summer Internship Program in Hybrid and Embedded Software Research (SIPHER) Program

The SIPHER program (Summer Internship Program in Hybrid and Embedded Software Research) is a program similar to SUPERB-IT, but located at Vanderbilt. More information about the program can be found at:

http://fountain.isis.vanderbilt.edu/fountain/Teaching/

The Institute for Software-Integrated Systems (ISIS) at Vanderbilt University's School of Engineering in cooperation with UC Berkeley and University of Memphis has recently been awarded a grant by the National Science Foundation to conduct research in the field of Hybrid and Embedded Systems (HES). The research aims at laying the scientific and technological foundations of embedded system design. Embedded computing systems are present in all traits of modern society: in cars and airplanes, in cell phones, in household devices, in medical devices, just to name a few. This is a multi-year research project that builds the science: the principles and the math, and the technology: the tools that the next generation of engineers will use to build these systems in the future, better than ever before.

The objective of the SIPHER program is that undergraduates from underrepresented groups participate in the research program: receive training in the science and technology developed by the researchers, and work on specific research problems. The program will be coordinated with UC Berkeley's SUPERB-IT, and joint teleconferences are expected.

The undergraduate students who apply to this program are expected to be rising juniors and seniors in an Electrical or Computer Engineering or Computer Science program leading towards a BSc or BE degree. The students must have a background in elementary systems courses: signals and systems for EE, digital systems in CE/CS, and have skills in programming using a high-level language. Engineering students from other programs, like Mechanical Engineering and Chemical Engineering are also encouraged to apply, provided they have similar backgrounds (e.g. in controls).

The SIPHER program runs for 10 weeks each summer, and there are 7 (seven) positions available, with a $6,000 stipend for the period. This year, the participants will be partly funded

by the Tennessee Louis Stokes Alliance for Minority Participation (TLSAMP) project (supported by NSF). Students are expected to pay for their accommodations. Limited housing opportunities may be available on the Vanderbilt campus. Applicants are competitively selected to the program.

In the SIPHER activities, we organized a summer internship in 2004 for seven participants from underrepresented groups. The students are organized into groups who solved different embedded software development problems. The students used Vanderbilt-developed modeling tool to create models of the embedded applications, develop code for the components, and then use the model transformation tools to create the final application.

The students worked on five small, team-oriented projects related to development of embedded software. In this work they used software tools available at ISIS, and they were supervised by professors and senior graduate students. During first few weeks they underwent rigorous training to learn how to use the design tools. The training was provided by lecturers who deliver our Model-Integrated Computing classes. All of the students had backgrounds in programming, and thus were able to solve the project problems. Similarly to UCB, graduate student mentors assisted and guided the student projects. Descriptions for the projects and the students who worked on them are included below.

Project: Distributed System Implementation of the Kuvangu Running Frog's Calling Behavior

Students: Praveen Mudindi—*Alabama A&M University*
Efosa Ojomo—*Vanderbilt University*

Male frogs attract their female mates over great distances by having collective synchronized chirps. These collective chirps increase the intensity and the possibility of being heard at great distances. Initially, the frogs chirp at different rates and out of phase, but after a while, they all synchronize their chirps to those of the frog(s) with the highest chirp intensity, producing a much louder and unified chirp. The objective of this project is to mimic this behavior using a distributed, real-time, embedded (DRE) system architecture. This platform has made it possible to write a generic program that will perform different functions on different systems. One main goal of the project is to get acquainted with embedded systems and sensor networks. Sensor networks sense their environments and react according to the program specifications they have been given.

Project: Maze Discovery and Chutes & Ladders

Students: Jameson Porter—*Tennessee Tech University*
Mary Hilliard—*University of Tennessee, Chattanooga*

As the need for embedded systems increase in society, capabilities for system development must expand to better define the constraints and physical aspects of the system. By using Model Integrated Computing (MIC), engineers can better understand the embedded system as a whole as well as easily evolve the system according to its environment. The goal of both the original and additional project is to gain a better

understanding of embedded systems and model integrating computing techniques by implementing a maze mapping and Chutes & Ladders playing Lego robot, respectively.

Project: Tic-Tac-Toe Implementation

Students: Shirley (Xue) Li—*Massachusetts Institute of Technology*
Sinithro (Miguel) Taveras—*University of Florida*

From medical procedures to space explorations, embedded, autonomous computing is becoming vital in many safety-critical applications. At the same time, many industries are looking to use the model-integrated approach for the analysis of complex systems. As a part of the SIPHER program, our initial objective was to familiarize ourselves with GME and its underlying principles. For the first half of the summer, using the SMOLES paradigm, our group worked on developing control algorithms for a LEGO Mindstorm robot. The specific task included implementing low-level, reactive control algorithms, and higher-level, planning activities. For the second half of the summer, our group's general objective was to further explore the capabilities of Mindstorm robots. Our particular task involved implementing a Tic-Tac-Toe gaming algorithm and perfecting navigational movements.

Project: A Model-Integrated Computing Approach to Embedded Controller Design

Students: Christopher Beers—*Vanderbilt University*

The MACS Lab at Vanderbilt University has a three tank fluid system, which is used as a test bed system for modeling, monitoring, control, estimation, and fault detection/adaptation of a hybrid system. The sensors and actuators in the system are operated by a Network Capable Application Processor (NCAP) and a Smart Transducer Interface Module (STIM). The NCAP has a small web server, and all commands are sent as through a HTTP/CGI interface to the STIM, which either reads a physical measurement or sends out an actuator signal to the system. An initial implementation of a controller was written as platform dependent C code. The control software was handwritten and required detailed knowledge of the actual NCAP configuration. The first step for this summer project was to use the ACE framework to write a wrapper that adheres to the IEEE 1451.2 standard for the NCAP communication. This was completed by mid-summer. The next step was to implement the control software using the OCP (Open Control Platform) and modify the FACT (Fault-Adaptive Control Technology) architecture so that a controller could be built using multiple finite state machines. This also required making changes to the graphical interfaces of GME. This simplifies the controller creation process, and allows users to easily create and run experiments on the three-tank system.

Project: Visual Tracking and Control

Students: John Williamson, III—*North Carolina A&T University*
Trevor Brown—*Middle Tennessee State University*

The first part was to visually track a robot going through a maze using a web camera to capture the images. In order to achieve this we had to start off with the basics, and that was to represent the relationship between the personal computer (PC) and camera using

the Generic Modeling Environment (GME). Once we had a good visual model of the camera and PC's components, the data flow between the two using input and output ports, and timers to start the data flow, we interpreted it and produced our main C++ files. We then hand-coded supporting functions, resulting in the achievement of having a PC controlled camera tracking a moving robot, keeping the target in the middle of our display screen. The second half of our project also involved tracking a moving robot. This time, instead of having a digital camera doing the tracking, we constructed a laser holding robot to do the tracking. With the now non-moving camera positioned to view our entire maze, we gathered information from its images to find the X and Y coordinates of the moving robot and our laser. With this information, Java and C++ code coupled with the Lejos firmware on the RCX, we had a system that controlled our built robot to move and keep the laser pointed on the moving robot captured by the web camera.

Plans for 2005

Similarly to the SUPERB program, the SIPHER is also dedicated to providing undergraduate students with the opportunities and experiences of research. As the main research goals of the project are set, the selected research topics are at the intersection of traditional Electrical Engineering and Computer Science. For this reason, we have chosen a set of research projects that for the 2005 SIPHER program that reflect this goal.

The selected research projects/topics are as follows:

* *Visually guided, remotely controlled robots.* Students in this project will use LEGO robots equipped with short-range communication devices, and a visual tracking system with a desktop workstation. The goal is to help an operator to track and control robots via a camera.
* *Process control applications.* Students in this project will learn about hybrid modeling of a small industrial plant and will develop fault-adaptive controllers for it.
* *Sensor networks for monitoring and control.* Students here will learn about networks of cameras that will be used to track moving objects.
* *Embedded controllers.* Students here will work on active vibration control problems.
* *Model-based tools for low-end microcontrollers.* Students will do research on how to build model-based software development tools for 8-bit microcontrollers.

Five graduate student mentors have been identified to facilitate the process, and the operation is being coordinated and directed by Mr. Efosa Ojomo (a former SIPHER student, now a graduate engineer). The students will be working in small teams, under the tutelage of the graduate students. The students and mentors have already been paired for what promises to be an exciting summer.

In addition to the science of research, we will also expose students to the other aspects of research. During the summer programs, there will be two formal reviews, with presentations and formal reports. The students will also participate in field trips to nearby industrial and research sites (Toshiba plant, Saturn plant, and NASA Marshall Space Flight Center).

3. Publications and Products

In this section, we list published papers only. Submitted papers and in press papers are described in section 2.2.

3.1. Journal Publications

- S. Abdelwahed, J. Wu, G. Biswas, J. Ramirez, E.J. Manders, "Online Fault-Adaptive Control for Efficient Resource Management in Advanced Life Support Systems," Habitation: International Journal of Human Support Research, vol. 10, no. 2, pp. 105-115, 2005.

- G. Biswas, E.J. Manders, J.W. Ramirez, N. Mahadevan, S. Abdelwahed, "Online Model-Based Diagnosis to Support Autonomous Operation of an Advanced Life Support System," *Habitation: International Journal of Human Support Research*, vol. 10, no. 1, pp. 21-38, 2004.

- M. Emerson, J. Sztipanovits, T. Bapty, "A MOF-Based Metamodeling Environment," *Journal of Universal Computer Science*, vol. 10, No. 10, pp. 1357-1382, October, 2004.

- K. D. Frampton, "Distributed Group-Based Vibration Control with a Networked Embedded System," *Journal of Intelligent Materials Systems and Structures*, Vol. 14, pp. 307--314, 2005.

- G. Karsai, A. Lang, S. Neema, "Design Patterns for Open Tool Integration," *Journal of Software and System Modeling*, vol 4., no. 1, DOI: 10.1007/s10270-004-0073-y, 2004.

- E. A. Lee, E.H. Abed (Ed.), "Engineering Education: A Focus on System, in Advances in Control, Communication Networks, and Transportation Systems: In Honor of Pravin Varaiya," *Systems and Control: Foundations and Applications Series*, Birkhauser, Boston, 2005.

- E. A. Lee, "What are the Key Challenges in Embedded Software?" *Guest Editorial in System Design Frontier*, Shanghai Hometown Microsystems Inc., Volume 2, Number 1, January 2005.

- M. Maroti, G. Simon, A. Ledeczi, J. Sztipanovits, "Shooter Localization in Urban Terrain," *IEEE Computer*, pp. 60-61, August, 2004.

- G. C. Necula, J. Condit, M. Harren, S. McPeak, W. Weimer, "CCured: Type-Safe Retrofitting of Legacy Software," *ACM Transactions on Programming Languages and Systems*, vol. 27, no. 3, May, 2005.

- P. Schmidt, I. Amundson, K.D. Frampton, "A Distributed Algorithm for Acoustic Localization Using a Distributed Sensor Network," *Journal of the Acoustical Society of America*, Vol. 115, No. 5, Pt. 2, pp. 2578, 2004.

- J. Sprinkle, "Generative Components for Hybrid Systems Tools," *Journal of Object Technology*, vol. 4, no. 3, April 2005, *Special issue: 6th GPCE Young Researchers Workshop 2004*, pp. 35-39, Available at http://www.jot.fm/issues/issue_2005_04/article5

- J. Sprinkle, G. Karsai, "A Domain-Specific Visual Language for Domain Model Evolution," *J. Vis. Lang. and Comp*., vol. 15, no. 3-4, pp. 291-307, Jun., 2004.

- T. Szemethy, G. Karsai, "Platform Modeling and Model Transformations for Analysis," *Journal of Universal Computer Science*, vol. 10, no. 10, pp 1383-1406, 2004.

3.2. Conference Papers

- A. Abate, L. El Ghaoui, "Robust Convex Optimization through Adjustable Robust Variables: an application to Model Predictive Control," In Proc. International Conference on Decision and Control, Atlantis, Bahamas, December 2004.

- A. Abate, L. Shi, S. Simic, S. Sastry, "A Stability Criterion for Stochastic Hybrid Systems," *In Proc. International Symposium of Mathematical Theory of Networks and Systems*, Leuven, July 2004.

- A. Agrawal, Gy. Simon, G. Karsai, "Semantic Translation of Simulink/Stateflow models to Hybrid Automata using Graph Transformations," Electronic Notes in Theoretical Computer Science, In Proc. Workshop on Graph Transformation and Visual Modeling Techniques (GT-VMT 2004), Volume 109, pp. 43-56.

- A. Agrawal, A. Vizhanyo, Z. Kalmar, F. Shi, A. Narayanan, G. Karsai, "Reusable Idioms and Patterns in Graph Transformation Languages," International Workshop on Graph-Based Tools, In Proc. 2004 International Conference on Graph Transformations, Rome, Italy, October, 2004.

- A. D. Ames, S. Sastry, "Blowing Up Affine Hybrid Systems," *43rd IEEE Conference on Decision and Control 2004 (CDC'04)*, Atlantis, Paradise Island, Bahamas, Dec. 2004, pp. 473-478.

- A. D. Ames, S. Sastry, "A Homology Theory for Hybrid Systems: Hybrid Homology," *In Proc. Hybrid Systems: Computation and Control, 8th International Workshop*, Zurich, Switzerland, March 9-11, M. Morari and L. Thiele, eds., vol. 3414 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 86-102, 2005.

- I. Amundson, P. Schmidt, K. D. Frampton, "A Decentralized Approach to Sound Source Localization with Sensor Networks," *Presented at the 2004 ASME International Mechanical Engineering Conference and Exposition*, Anaheim CA, November 2004.

- D. Beyer, A. Chakrabarti, T. A. Henzinger, "Web Service Interfaces," *Proc. 14th International World Wide Web Conference (WWW 2005)*, Chiba, Japan, May 10—14 2005.

- K. Chatterjee, L. de Alfaro, T. A. Henzinger. "Trading Memory for Randomness," *In Proc. 1st International Conference on Quantitative Evaluation of Systems (QEST 04)*, University of Twente, Enschede, The Netherlands, September 27 --30, 2004.

- E. Cheong, J. Liu, "galsC: A Language for Event-Driven Embedded Systems," *Presented at Design, Automation and Test in Europe (DATE)*, Munich, Germany, March 7--11, 2005.

- A. Davare, K. Lwin, A. Kondratyev, A. Sangiovanni-Vincentelli. "The Best of Both Worlds: The Efficient Asynchronous Implementation of Synchronous Specifications," *Presented at ACM/IEEE Design Automation Conference*, San Diego, CA, June 7 --11, 2004.

- M. Emerson, J. Sztipanovits, "Implementing a MOF-Based Metamodeling Environment Using Graph Transformations" *In Proc. 4th Workshop on Domain-Specific Modeling*, *20th Annual ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA),* pp. 83-92, Vancouver, Canada, October 2004.

- K. D. Frampton, "Vibroacoustic Control with a Distributed Sensor Network," *Presented at Applications of Graph Transformations with Industrial Relevance (AGTIVE04*), Williamsburg, VA, September, 2004.

- R. D. Harvey, D. G. Walker, K. D. Frampton, "Distributed Control to Improve Performance of Thermoelectric Coolers," *Presented at 2004 ASME International Mechanical Engineering Conference and Exposition*, Anaheim CA, November 2004.

- G. Karsai, A. Agrawal, "Graph Transformations in OMG's Model-Driven Architecture," *In Proc. Applications of Graph Transformations with Industrial Relevance (AGTIVE 2003)*, LNCS 2062. pp. 243-259, Charlottesville, VA, September 29—October 1, 2003.

- E. A. Lee, S. Neuendorffer, "Classes and Subclasses in Actor-Oriented Design," invited paper, *Conference on Formal Methods and Models for Codesign/ (MEMOCODE)*, San Diego, CA, USA, June 22-25, 2004.

- E. A. Lee, H. Zheng, "Operational Semantics of Hybrid Systems" invited paper, *In Proc. Hybrid Systems: Computation and Control (HSCC),* LNCS TBD, Zurich, Switzerland, March 9-11, 2005.

- G. Madl, S. Abdelwahed, G. Karsai, "Automatic Verification of Component-based Real-time CORBA Applications," *In Proc. 25th IEEE International Real-Time Systems Symposium (RTSS'04)*, Lisbon, Portugal, Dec. 2004, pp. 231—240.

- M. L. McKelvin, Jr, J. Sprinkle, C. Pinello, A. Sangiovanni-Vincentelli, "Fault Tolerant Data Flow Modeling Using the Generic Modeling Environment," *12th Annual IEEE International Conference and Workshop on the Engineering of Computer Based Systems*, Greenbelt, Maryland, Apr. 4--5, 2005, pp. 229–235.

- T. Meyerowitz, J. Sprinkle, A. Sangiovanni-Vincentelli, "A Visual Language for Describing Instruction Sets and Generating Decoders," *20th Annual ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA)*, Vancouver, BC, Oct., 25, 2004, pp. 23–32.

- S. Neema, A. Dixon, T. Bapty, J. Sztipanovits, "Model-Integrated Computing for Heterogeneous Systems," *In Proc. International Conference on Computing, Communications and Control Technologies (CCCT'04)*, Austin, TX, August 14-17, 2004.

- S. Neuendorffer, "Modeling Real-World Control Systems: Beyond Hybrid Systems," *In Proc. Winter Simulation Conference (WSC)*, Washington, DC, USA, December 5--8, 2004.

- W. Plishker, K. Ravindran, N. Shah, K. Keutzer. "Automated Task Allocation for Network Processors," *In Proc. Network System Design Conference*, October, 2004, pp. 235-245.

- L. Shi, A. Abate, S. Sastry, "Optimal Control for a class of Stochastic Hybrid Systems," *In Proc. International Conference on Decision and Control*, Atlantis, Bahamas, December 2004.

- J. Sprinkle, J. Davis, G. Nordstrom, "A Paradigm for Teaching Modeling Environment Design," *20th Annual ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA), Educators Symposium (Poster Session)*, ACM, Vancouver, BC, Oct., 24--28, 2004.

- J. Sprinkle, J. M. Eklund, H. J. Kim, S. S. Sastry, "Encoding Aerial Pursuit/Evasion Games with Fixed Wing Aircraft into a Nonlinear Model Predictive Tracking Controller," *In Proc. IEEE Conference on Decision and Control*, pp. 2609--2614 , Dec., 2004.

- J. Sprinkle, J. M. Eklund, S. S. Sastry, "Toward Design Parameterization Support for Model Predictive Control," *IEEE 4th International Conference on Intelligent Systems Design and Application*, IEEE, IEEE Press, Budapest, Hungary, Aug., 26--28, 2004.

- J. Sprinkle, O. Shakernia, R. Miller, S. S. Sastry, "Using the Hybrid Systems Interchange Format to Input Design Models to Verification & Validation Tools," *IEEE Aerospace Conference*, Big Sky, MT, Mar., 2005.

- T. Tao, K.D. Frampton, "Experiments on the Decentralized Vibration Control with Networked Embedded Systems," *Presented at the 2004 ASME International Mechanical Engineering Conference and Exposition*, Anaheim CA, November 2004.

- J. Tolvanen, J. Sprinkle, M. Rossi, eds., "Proceedings of the 4th OOPSLA Workshop on Domain-Specific Modeling (DSM'04)," Jyvavaskyla, Finland, *20th Annual ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA)*, University of Jyvavaskyla, Oct., 2004.

- A. Vizhanyo, A. Agrawal, F. Shi, "Towards Generation of Efficient Transformations," *In Proc. Generative Programming and Component Engineering, 3rd International Conference*, October, 2004, LNCS 3286, pp 298-316, 2004.

- D. G. Walker, K. D. Frampton, R.D. Harvey, "Distributed Control of Thermoelectric Coolers," *In Proc. 9th Intersociety Conference on Thermal and Thermomechanical Phenomena in Electronic Systems (ITherm)*, paper no. 151, Las Vegas, NV, June 1-4, 2004, pp. 361--366.

3.3. Books, Reports, and Other One-Time Publications

- A. D. Ames, S. Sastry, "A Homology Theory for Hybrid Systems: Hybrid Homology," *In Proc. Hybrid Systems: Computation and Control, 8th International Workshop*, Zurich, Switzerland, March 9-11, M. Morari and L. Thiele, eds., vol. 3414 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 86-102, 2005.

- C. Brooks, A. Cataldo, E. A. Lee, J. Liu, X. Liu, S. Neuendorffer, H. Zheng, "HyVisual: A Hybrid System Visual Modeler," *Technical Memorandum UCB/ERL M04/18/*, University of California, Berkeley, June 28, 2004.

- C. Brooks, E.A. Lee, X. Liu, S. Neuendorffer, Y. Zhao, H. Zheng (eds.), "Heterogeneous Concurrent Modeling and Design in Java (Volume 1: Introduction to Ptolemy II)," *Technical Memorandum UCB/ERL M04/27/*, University of California, Berkeley, July 29, 2004.

- C. Brooks, E. A. Lee, X. Liu, S. Neuendorffer, Y. Zhao, H. Zheng (eds.), "Heterogeneous Concurrent Modeling and Design in Java (Volume 2: Ptolemy II Software Architecture)," *Technical Memorandum UCB/ERL M04/16/*, University of California, Berkeley, June 24, 2004.

- C. Brooks, E. A. Lee, X. Liu, S. Neuendorffer, Y. Zhao, H. Zheng (eds.), "Heterogeneous Concurrent Modeling and Design in Java (Volume 3: Ptolemy II Domains)," *Technical Memorandum UCB/ERL M04/17/*, University of California, Berkeley, June 24, 2004.

- M.Chen, A. Abate, A. Zakhor, S. Sastry, "Stability and Delay Considerations for Flow Control Over Wireless Networks," *UCB ERL Tech Report No M05/14*, Berkeley, CA, 2005.

- J. Davis, C. Hylands., J. Janneck, E.A. Lee, J. Liu, X.Liu, S. Neuendorffer, S. Sachs, M. Stewart, K. Vissers, P. Whitaker, Y. Xiong, "Overview of the Ptolemy Project," *Technical Memorandum UCB/ERL M01/11*, EECS, University of California, Berkeley, March 6, 2001.

- V. Krishnan, Master's Report, "Real-Time Systems Design in Ptolemy II: A Time-Triggered Approach," *Technical Memorandum UCB/ERL M04/22/*, University of California, Berkeley, July 12, 2004.

- E. A. Lee, Editorial, February 19, 2005, "Absolutely Positively On Time: What Would It Take?" Available at http://ptolemy.eecs.berkeley.edu/publications/papers/05/EmbeddedSoftwareColumn/

- E. A. Lee, "Balance between Formal and Informal Methods, Engineering and Artistry, Evolution and Rebuild," *Technical Memorandum UCB/ERL M04/19/*, University of California, Berkeley, July 4, 2004.

- E. A. Lee, "Concurrent Models of Computation for Embedded Software," *Technical Memorandum, University of California, Berkeley, UCB/ERL M05/2/*, January 4, 2005.

- E. A. Lee, S. Neuendorffer, "Concurrent Models of Computation for Embedded Software," *Technical Memorandum UCB/ERL M04/26/*, University of California, Berkeley, July 22, 2004.

- S. Neuendorffer, "Actor-Oriented Metaprogramming," *PhD Thesis*, University of California, Berkeley, December 21, 2004.

- G. Zhou, "Dynamic Dataflow Modeling in Ptolemy II," *Master's Report, Technical Memorandum No. UCB/ERL M05/2/*, University of California, Berkeley, December 21, 2004.

3.4. Dissemination

Although this is a long term project focused on foundations, we are actively working to set up effective technology transfer mechanisms for dissemination of the research results. A major part of this is expected to occur through the open dissemination of software tools.

Making these software tools useful and usable outside the research community is a significant issue. Towards this end, we have cooperated with the formation of the Escher consortium, which has begun operating (www.escherinstitute.org) . Escher has negotiated with both Berkeley and Vanderbilt specific priorities for "industrial hardening" of research tools from this project. In particular, at Berkeley, top priority will be placed on Giotto, xGiotto, and Ptolemy II in the near term. At Vanderbilt, top priority will be placed on GME, Desert, and GReAT. General Motors, Raytheon, and Boeing are signed up as charter industrial partners in Escher, and more companies are expected.

An important, emerging forum for dissemination of information and influencing industrial practice is the recently form Model-Integrated Computing Special Interest Group (MIC PSIG) of OMG. (http://mic.omg.org/) This forum is run by industry and its primary goal is the preparation and management of standardization activities related to various aspects model-based design in embedded systems. The ISIS and CHESS teams are very much involved these activities. The White Paper for a standard Open Tool Integration Framework (OTIF) is based on the work of researcher at ISIS.

The Chess website, http://chess.eecs.berkeley.edu, includes publications and software distributions. In addition, as part of the outreach effort, the UC Berkeley introductory signals systems course, which introduces hybrid systems, is available at http://ptolemy.eecs.berkeley.edu/eecs20/ and Ptolemy II software is available at http://ptolemy.eecs.berkeley.edu.

The ISIS website, http://www.isis.vanderbilt.edu, makes publications and software available.

3.5. Other Specific Product

The following software packages have been made available during this review period on the Chess website, http://chess.eecs.berkeley.edu:

- The Generic Modeling Environment (GME 4) is a configurable toolkit for creating domain-specific modeling and program synthesis environments. The configuration is accomplished through metamodels specifying the modeling paradigm (modeling language) of the application domain. The modeling paradigm contains all the syntactic, semantic, and presentation information regarding the domain; which concepts will be used to construct models, what relationships may exist among those concepts, how the concepts may be organized and viewed by the modeler, and rules governing the construction of models. The modeling paradigm defines the family of models that can be created using the resultant modeling environment. The latest release (8/25/04) can be found at: http://www.isis.vanderbilt.edu/projects/gme/.
- GReAT (Graph Rewriting And Transformation) is a component technology of GME compriseed of a metamodel based graph transformation language useful for the specification and implementation of model-to-model transformations. The most recent release, r1.3.3, can be downloaded from: http://www.isis.vanderbilt.edu/Projects/mobies/downloads.asp#GREAT
- HyVisual 5.0 is a hybrid system visual modeler, was released in March of 2005. This visual modeler supports construction of hierarchical hybrid systems. It uses a block-diagram representation of ordinary differential equations (ODEs) to define continuous dynamics. It uses a bubble-and-arc diagram representation of finite state machines to define discrete behavior. HyVisual 5.0-alpha is available at http://ptolemy.eecs.berkeley.edu/hyvisual/index.htm
- Metropolis (1.02) consists of an infrastructure, a tool set, and design methodologies for various application domains. The infrastructure provides a mechanism such that heterogeneous components of a system can be represented uniformly and tools for formal methods can be applied naturally. Metropolis 1.02 is available at http://chess.eecs.berkeley.edu/chess/forum/6.html
- Ptolemy II 5.0 beta is a set of Java packages supporting heterogeneous, concurrent modeling and design. Its kernel package supports clustered hierarchical graphs, which are collections of entities and relations between those entities. Its actor package extends the kernel so that entities have functionality and can communicate via the relations. Its domains extend the actor package by imposing models of computation on the interaction

between entities. Examples of models of computation include discrete-event systems, dataflow, process networks, synchronous/reactive systems, and communicating sequential processes. Ptolemy II includes a number of support packages, such as data, providing a type system, data encapsulation and an expression parser, plot, providing visual display of data, math, providing matrix and vector math and signal processing functions, and graph, providing graph-theoretic manipulations. The Ptolemy II V.5 beta release is available at http://ptolemy.eecs.berkeley.edu/ptolemyII/ptII5.0/index.htm

- Universal Data Model (UDM) Generates C++ API from UML class diagrams. The API can be used to read/write XML files, GME databases, etc. and is component technology for Graph Rewriting And Transformation (GReAT). V. 2.21 can be downloaded from http://www.isis.vanderbilt.edu/Projects/mobies/downloads.asp#UDM

- VisualSense is a visual editor and simulator for wireless sensor network systems. Modeling of wireless sensor networks requires sophisticated modeling of communication channels, sensor channels, ad-hoc networking protocols, localization strategies, media access control protocols, energy consumption in sensor nodes, etc. This modeling framework is designed to support a component-based construction of such models. It is intended to enable the research community to share models of disjoint aspects of the sensor nets problem and to build models that include sophisticated elements from several aspects. VisualSense-4.0.1 is available as part of the Ptolemy II 4.0.1 release at http://ptolemy.eecs.berkeley.edu/ptolemyII/ptII4.0/index.htm.

4. Contributions

This section summarizes the major contributions during this reporting period.

4.1. Within Discipline

4.1.1. Hybrid Systems Theory

- We have defined an operational semantics for hybrid systems which was an invited plenary speech at the *Hybrid Systems: Computation and Control* conference in 2005, and poised to influence the current and next generation of toolsets to reflect these semantics.

- We have developed algorithms for computing the real value of discounted properties, and continued investigation of their application.

- We have developd a theory of a functorial view of hybrid systems which enables elegant characterization of Zeno and other qualitative properties of hybrid systems.

- We have developed an introductory theory on a condition number for hybrid system simulations.

- We have improved on the best known algorithms for finding strategies for the control of stochastic hybrid systems. These have been applied to applications as diverse as air to air combat for UAVs, coordinated operations of UUVs and communication networks.

- We have constructed a toolbox using ellipsoidal methods to calculate reach sets for linear dynamic systems, and begun to apply those to hybrid systems.

- We have developed an extensive theory of two and multi person stochastic games with extensions of notions of safety and almost safety in a number of important directions.

- We have continued to apply and study stochastic hybrid systems within the domain of biological systems.

- We have begun the implementation of a broad initiative to support toolchains in hybrid systems under semantic anchoring and model transformations.

4.1.2. Model-Based Design

- Applying our ongoing work on metamodeling, we have continued development on semantic anchoring for model-based development.

- We have continued to demonstrate our defined agent algebras as a formal framework for uniformly representing and reasoning about models of computation used in the design of hybrid and embedded software systems.

- We have continued to demonstrate our theoretical and compositional framework for reasoning about causality in components which are composed under concurrent models of computation.

- We have extended our previously developed tagged-signal model for concurrent models of computation to represent the semantics of globally asynchronous, locally synchronous systems built upon loosely time-triggered architectures.

- We have continued to maintain a language and a suite of supporting tools for the specification of model transformations based on graph rewriting.

- We have continued to use our approach to model synthesis based on patterns specified formally as metamodels.

4.1.3. Advanced Tool Architectures

- We have extended modularity mechanisms (classes, subclasses, inheritance, interfaces) for actor-oriented design, complementing the prevailing object-oriented methods to also include types and subtypes for modeling-time template based design.

- We have further developed the code generation approach based on component specialization by developing a formal framework for reasoning about reconfiguration in embedded software.

- We have continued to improve the performance and feature set of the Metropolis framework.

- We have further developed our notion of interface theories to support reasoning about heterogeneous component composition and about the dynamics of models of computation.

- We have developed a modeling environment for processor instruction set decoding and used it to simulate the MIPS, PowerPC, and ARM processors.

- We have continued to investigate interests in fault-tolerant systems by developing new modeling languages which simulate and trace faults in a system.

4.1.4. Experimental Research

- We have used our work on safe set calculations to successfully land UAVs (a joint project with Northrup Grumann) in flight tests

- We have used our results in model predictive control to develop air to air combat strategies for a UAV to successfully defeat a manned aircraft (joint project with Boeing). The UAV successfully targeted the manned aircraft (of much higher speed and performance and piloted by an expert test pilot) several times.

- We have deployed the Metropolis platform-based design methodology for use on various veitronics problems of interest to Toyota, GM, and BMW.

- We have continued development, and deployed a modeling environment for wireless sensor networks. These have been used in very high profile shooter localization schemes.

- We have shown application results on the connectivity of large-scale sensor networks.

- We have developed new programming models for sensor networks that build on the popular TinyOS models.

- We have developed programming models for bit streaming applications that use sketching of strategies with code generation.

- We have continued adapting sensor networks and wireless technology to address elder care diagnosis and monitoring issues.

- We have used control theory and software based on the Time-Triggered Architecture and Giotto language to control Unmanned Aerial Rotorcraft, and also integrated Giotto with Metropolis.

4.2. Other Disciplines

- We developed new efficient algorithms for solving stochastic games, which have applications in other fields such as economics.

- We contributed to scientific interdisciplinary information sharing through collaboration and major contribution to the framework of the Kepler Scientific Workflow project.

- We continued to contribute to the simulation and understanding of biological processes, such as the piliation of *E. coli*, and the production of the *B. subtilis* antibiotics. We have now moved on to other mechanisms in other organisms.

4.3. Human Resource Development

Several panels in important conferences and workshops pertinent to embedded systems (e.g., DAC, ICCAD, HSCC, EMSOFT, CASES, and RTSS) have pointed out the necessity of upgrading the talents of the engineering community to cope with the challenges posed by the next generation embedded system technology. Our research program has touched many graduate students in our institutions and several visiting researchers from industry and other Universities so that they now have a deep understanding of embedded system software issues and techniques to address them. The industrial affiliates to our research program are increasing and we hope to be able to export in their environments a modern view of system design. Preliminary feedback from our partners has underlined the importance of this process to develop the professional talent pool.

4.4. Integration of Research and Education

In this report, we have touched multiple times on research and education especially in the outreach section. In addition, there has been a strong activity in the continued update of the undergraduate course taught at Berkeley on the foundations of embedded system design. The graduate program at Berkeley and at Vanderbilt has greatly benefited from the research work in the ITR. EE249 at Berkeley has incorporated the most important results thus far obtained in the research program. EE 290 A and C, advanced courses for PhD students, have featured hybrid system and the interface theories developed under this project. EE219C, a course on formal verification, has used results from the hybrid theory verification work in the program. Finally, many final projects in these graduate courses have resulted in papers and reports listed in this document. The course EE291E on Hybrid Systems: Computation and Control is jointly taught at Berkeley and Vanderbilt and is benefiting a great deal from comments of students as far as the development of new text book material.

In addition to the influence on graduate students, we have endeavored to show hybrid and embedded systems as emerging research opportunities to undergraduates. We have also demonstrated that for advanced undergraduates these topics are not out of place as senior design courses, or advanced topics courses, which may in the future lead to the integration of these as disciplines in engineering across a broader reach of universities.

4.5. Beyond Science and Engineering

Embedded systems are part of our everyday life and will be much more so in the future. In particular, wireless sensor networks will provide a framework for much better environmental monitoring, energy conservation programs, defense and health care. Already in the application chapter, we can see the impact of our work on these themes. Elder care, soft walls and distributed diagnosis are a few examples of this impact. In the domain of transportation systems, our research is improving safety in cars. Future applications of hybrid system technology will involve biological systems to a much larger extent showing that our approach can be exported to other field of knowledge ranging from economics to biology and medicine. At Berkeley, the Center for Information Technology Research in the Interest of Society is demonstrating the potential of our research in fields that touch all aspects of our life. Some key societal grand challenge problems where our ITR research is making a difference includes health care delivery: high confidence medical devices and systems, transportation, cybersecurity, and transportation.