Statement of Work for NSF ITR "Foundations of Hybrid and Embedded Software Systems"

The attached statement of work and timeline of the progress to be made on the ITR is a first draft for the cooperative agreement to be executed between NSF and the University of California, Berkeley. We expect to refine the statement of work and deliverables during the course of a 3-month planning period prior to the kick off of the project. Before the kickoff of the ITR we expect to have the different advisory bodies for the project and the centers at Berkeley and Vanderbilt to be appointed and appropriate intellectual property agreements worked out with industrial partners. The general principle guiding the intellectual property agreements is based on the research on this project being open source. We expect to work with the NSF Program Manager to finalize these details during the planning period.

The proposed research program has four focus areas: hybrid systems theory, model-based design, advanced tool architectures, and experimental research. *Hybrid systems theory* will build the mathematical foundation of Moderns Systems Science (MSS). This foundation needs to be grounded both in the continuous mathematics of physical processes and the discrete mathematics of computational processes. *Model-based design* will build a scalable methodology for systems design and analysis based on hybrid systems theory. Model-based design controls complexity by supporting the manipulation and integration of models for multiple design aspects. *Advanced tool architectures* will provide software support for model-based design. An open software infrastructure will accommodate design and analysis tools as inter-operating components. *Experimental research* will guide the theory and tool development. Special emphasis will be on applications with societal impact, such as networked embedded systems for environmental monitoring and embedded control systems that address national and homeland security needs. The overarching theme in our research is *compositionality*. We will pursue compositionality in hybrid system theory, we will use composable models and model manipulation methods in model-based design, and we will investigate composable tool architectures that enable the rapid integration of domain-specific design environments. Compositionality enables the separation of orthogonal concerns, and the integration and reuse of solutions (theories, models, tools), and thus makes the significant objectives of this project feasible.

The document below summarizes the tasks of the four focus areas plus the education and outreach effort, with problem areas/objectives for each year, followed by a timeline for deliverables.

1. Hybrid Systems Theory (Faculty Lead: Henzinger)

A *hybrid system* is a system that contains both physical and computational processes. A typical example of a hybrid system is embedded control software, which interacts with a physical environment through sensors and actuators. Hybrid systems theory, therefore, lies at the foundation of any effort to systematically build a modern systems science (MSS). So far, hybrid systems theory has been developed, largely at UCB, only for small, idealized systems, such as *hybrid automata*. A scalable MSS must

address the following key issues. *Key faculty participation here will be from Biswas, Bollobas, Henzinger, Lee, Sangiovanni-Vincentelli, Sastry, Sztipanovits, and Varaiya*

a. Deep Compositionality

i. Assume Guarantee Techniques for Hybrid Systems
Deliverable: assume-guarantee rules for real-time and hybrid systems

ii. Practical Hybrid Systems Modeling Language
Deliverable: hybrid systems modeling language with parameterization, inheritance, and dynamic process creation

iii. Interface Theory for Hybrid Components
Deliverable: specifying and compatibility checking for multi-aspect interfaces of hybrid components

b. Robust Hybrid Systems

i. Bundle Properties of hybrid systems
Deliverable: robust compositional model for hybrid behavior

ii. Topologies for Hybrid Systems
Deliverable: topological model of hybrid systems based on discounted games

iii. Stochastic Hybrid Systems
Deliverable: modeling and analysis of stochastic hybrid systems

c. Computational Hybrid Systems

i. Zeno behavior of hybrid systems
Deliverable: algorithms and tools for detecting Zeno phenomena

ii. Approximation techniques for Hamilton Jacobi Equations
1. Fast Wavefront Methods
2. Ellipsoidal Methods
3. Linear Hyperplane Methods

Deliverable: tools for reachability and controllability analysis of hybrid behavior

iii. Synthesis of safe and live controllers for hybrid systems
Deliverable: algorithms and tools for multi-modal controller synthesis

d. Phase Transitions
Deliverable: extension and application of phase transition theory to hybrid systems behavior

2. Model Based Design (Faculty Lead: Sztipanovits)

While hybrid systems theory provides a semantic, mathematical foundation for the integrated modeling of physical and information systems, 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. To do this well, they must emphasize concurrency, communication abstractions, and temporal properties, rather than procedural interfaces. For example, domain-specific semantic frameworks 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.
Key faculty involved is Aiken, Henzinger, Karsai, Keutzer, Lee, Necula, Sangiovanni-Vincentelli and Sztipanovits.

a. Composition of Domain Specific Modeling Languages

i. Meta Modeling
Deliverable: Meta-modeling methodology that allows the creation of mathematically sound and precise meta-models of modeling languages of embedded systems in support of system specification, design, synthesis, analysis, and validation.

ii. Composition to manipulate meta-models
Deliverable: Composition techniques that enforce the correct construction of complex metamodels from partial, simpler meta-models.

iii. Integration of meta-modeling with hybrid systems
Deliverable: Demonstration of the compositional meta-modeling methodology in the field of hybrid systems as applied to embedded system design

b. Model Synthesis Using Design Patterns

i. Pattern Based Model Synthesis
Deliverable: Methodology for pattern-based composition and synthesis of models for embedded systems

ii. Models of Computation
Deliverable: Models of Computations specified as design patterns for embedded systems

iii. Design Constraints and Design Patterns for Multiple Models of Computation.
Deliverable: Methodology for capturing design constraints that crosscut multiple Models of Computations, and examples demonstrating the feasibility of the approach.

c. Model Transformation

i. Meta Generators
Deliverable: Meta-generation technology and framework that allows the model-based synthesis of generators for embedded systems.

ii. Scalable Models
Deliverable: Generative extensions to meta-modeling languages that allow specification of higher-order constructs in domain modeling languages.

iii. Construction of Embeddable Generators
Deliverable: Technology for building generators that are embedded in resource-constrained embedded systems and operated at run-time.

3. **Advanced Tool Architectures (Faculty Lead: Lee)**
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. We therefore propose in this project to shift 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. Concretely, the key products of this work will be a set of toolkits, frameworks, and other software modules. We will still develop tools, but only as reference applications of the toolkits and frameworks. This approach has had high impact in other areas, most notably in compilers. and we believe that its application in the context of MSS will yield high dividends for researchers and tool implementers in hybrid and embedded systems.

The tool architecture work will be guided by the concurrent work in the other research focus areas. In particular, the concept of compositionality of hybrid systems languages and modeling languages has a profound impact on the ways in which we think about tool architectures. We propose to extend the tool architecture work performed to date at UCB and VU. To disseminate the results of this work, we propose to create a national resource consisting of a repository of open-source, well-documented, web-integrated toolkits and frameworks. Particular areas in which we will focus our efforts follow. *Key faculty involved here includes Aiken, Henzinger, Karsai, Keutzer, Lee, Messerschmitt, Necula, Sangiovanni-Vincentelli, Sastry, and Sztipanovits.*

   a. **Syntax and Semantics**
      i. Semantic Composition
      Deliverable: abstract semantics for embedded system modeling languages
      ii. Visual Concrete Syntaxes
      Deliverable: Technology and framework for experimental research on the visual concrete syntax of embedded system modeling languages.
      iii. Modal Models
      Deliverable: methods and tools for nesting dataflow and control-flow oriented models of computation
   b. **Interface Theories**
   Deliverable: algorithms and tools for checking interface compatibility and interface refinement for embedded software components
   c. **Virtual Machine Architectures**
   Deliverable: definition, implementation, and evaluation of embedded machines for software with resource and timing constraints
   d. **Components for Embedded Systems**
   Deliverable: development of component-based frameworks for embedded software reuse

4. **Experimental Research (Faculty Lead: Sangiovanni-Vincentelli and Sastry)**

To be relevant, a new design technology and systems science has to be informed by the real problems of real systems. The investigators in this project have strong track records in developing, fielding, and supporting large-scale applications, including highway transportations systems, air traffic management, electronic design automation systems, aerial robotics, vehicular electronics, sensor nets, and smart structures. This ITR project will serve as a focal point for the problems and solutions raised by these systems efforts. A sub-theme here is that the ITR project will offer "new economy" technologies to "old technology" sectors.

Note that we are *not* proposing that the ITR project orbit around a single (or even a small number of) challenge problems or testbeds. The project will engage in dialog with large systems efforts, and provide technology to those systems efforts for evaluation and testing, but it will not divert resources to the large systems building efforts themselves. Instead, this ITR project will include a distributed laboratory for experimentation with location-aware computing, robotics, aerobotics, vehicle electronics, control systems, security systems, smart environments, smart structures, and even whimsical embedded systems. This laboratory will be used to test the concepts emerging from the center, to evaluate evolving technology from elsewhere, to demonstrate results from the project, and to keep the research grounded in practical, implementable methods. The design of the laboratory, applications, and reference solutions will be made available to the research community. *All ITR investigators will participate in this activity*.

a. Embedded Control Systems

i. Embedded Software for Avionics
Deliverable: High Level Languages and Frameworks for designing embedded control systems for avionics applications. Experiments will be performed on the aerobot testbed at Berkeley.

ii. Embedded Software for Vehicular Electronics
Deliverable: High Level Languages and Frameworks for designing embedded control systems for vetronics applications. Experiments will be performed on vehicular testbeds at Vanderbilt/Berkeley.

iii. Embedded Software for Wireless Embedded Systems
Deliverable: Technology and algorithms for the representation and constraint-based pruning of complex design spaces for embedded systems.

b. Embedded Software for National and Homeland Security

i. Embedded Software for Air Traffic Control
Deliverable: Embedded Software for conflict resolution and other safety critical maneuvers for safer air transportation. Work will be coordinated with certification efforts at FAA/NASA/Eurocontrol.

ii. Embedded Software for Onboard Avionics
Deliverable: "*Softwalls"* for automatic prevention of aircraft from entering restricted use airspace. Work will be coordinated with industry partners Boeing and Honeywell.

c. Networks of distributed sensors

i. Sensorwebs of embedded networked systems
Deliverable: Programing languages for Networked Embedded System

ii. Applications
Deliverable: Specific Implementations of building environment monitoring, tracking and surveillance.

d. Hybrid Models in Structural Engineering

i. Active Noise Control
Deliverable: Experimental verification of tools in the domain of noise control of structures.

ii. Vibration damping of complex structures
Deliverable: Experimental verification of the techniques and tools developed in the vibration control problem domain.

5. Education and Outreach (Faculty Lead: Lee and Varaiya (undergraduate), Sastry and Henzinger (graduate))

Our agenda is 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. Based on the ongoing, groundbreaking effort at UCB, we propose to deliberately re-architect and retool undergraduate teaching at the participating institutions, and to make the results widely available to encourage critical discussion and facilitate adoption. In addition, will be recruiting new undergraduate students (usually juniors) from minority institutions through the established REU programs SUPERB-IT at UCB (8 more) and SURGE at VU (5 more) to participate in the research of the project.

a. Curriculum Development for Modern Systems Science (MSS)

i. Lower Division Courses
Deliverable: lower-division course that combines the mathematical foundations of MSS (discrete and continuous math) with applications from signal processing and computer science

ii. Upper Division Courses
Deliverable: upper-division course that focuses on system design principles, including concurrency modeling and abstraction

iii. Graduate Course Offerings
Deliverable: graduate courses on hybrid systems, concurrency modeling, and embedded software

b. Undergraduate Curriculum Insertion and Transfer

i. Goals and ABET requirements
Deliverable: High-level goals documented according to ABET requirements, including evaluation criteria.

ii. New courses for partner institutions and community colleges
Deliverable: Courseware materials for new courses.

iii. Introduction of new courses into curriculum
Deliverable: Documentation for courses placed in the curricula and taught.

iv. Evaluation and revision
Deliverable: Evaluation report and revised courseware materials.
v. New Elective Courses
Deliverable: Courseware materials for new courses.
vi. Expansion of SUPERB program
Deliverable: engage SUPERB students in CHESS research

c. Summer Internship Program in Hybrid and Embedded Software Research (SIPHER)
Deliverable: Summer internship programs in Hybrid and Embedded Software Research for students and faculty from partnering institutions: community colleges and HBCU-s.

Timeline for deliverables

The deliverables will be in the form of papers and scholarly output (working papers), software releases and experiments. All material will be made available in the public domain. We also will publicize the materials through regular meetings and CHESS/ISIS reviews in addition to the ITR review process.

1. Hybrid Systems Theory

1a. Deep Compositionality

- Year 1 Development and evaluation of a method for temporal interface specification and compatibility checking for hybrid systems. Development and evaluation of a method for assume-guarantee reasoning about temporal properties of hybrid systems
- Year 2 Development and evaluation of a method for resource interface specification and compatibility checking for hybrid systems. Development and evaluation of a method for assume-guarantee reasoning about resource properties of hybrid systems
- Year 3 Development and evaluation of a method for performance interface specification and compatibility checking for hybrid systems, assume-guarantee reasoning about performance properties of hybrid systems.
- Year 4 Definition and evaluation of a first version of a specification language for hybrid systems with deep compositionality.
- Year 5 Final version of a specification language for hybrid systems with deep compositionality, parameterization, inheritance and dynamic process creation.

1b. Robust Hybrid Systems

- Year 1 Development and evaluation of a methodology for the robust modeling of discrete systems based on discounted notions of property satisfaction.
- Year 2 Development and evaluation of a methodology for the robust and discounted modeling of hybrid systems based on trace topologies.
- Year 3 Development of a methodology for robust and discounted refinement of hybrid systems.
- Year 4 Development and evaluation of a methodology for compositional modeling for stochastic hybrid systems.
- Year 5 Development of a unified framework for modeling real-time, stochastic and other quantitative interaction behavior using discounted games.

1c. Computational Hybrid Systems

- Year 1 Development and evaluation of a methodology for the detection of Zeno phenomena in hybrid systems
- Year 2 Development and evaluation of compositional algorithms for the model checking of hybrid systems based on fast-wavefront, ellipsoidal and linear hyper-plane methods.
- Year 3 Development and evaluation of compositional techniques for the single mode controllability of hybrid systems based on fast-wavefront, ellipsoidal and linear hyper-plane methods.
- Year 4 Development and evaluation of compositional alogirthms for multi-mode controllability of hybrid systems.
- Year 5 Development and evaluation of compositional algorithms for the optimal control of hybrid systems.

1d Phase Transitions

This work is of a basic nature aimed at elucidating the role of phase transitions in constraint solvers for embedded and networked embedded systems. No specific timeline for results is feasible to give.

2. Model Based Design Timetable

2.a Composition of Domain Specific Modeling Languages

- Year 1 We will develop the meta-modeling methodology and language.
- Year 2 We will develop composition techniques for correct by construction meta-models and apply compositional metamodeling to hybrid models of embedded systems.
- Year 3 Revision and new versions of Year 1, 2 outcomes.
- Year 4 We will develop enhanced compositional meta-modeling tools applied to hybrid systems.

- Year 5 Integration of compositional meta-modeling tools in repository NEPHEST (National Experimental Platform for Hybrid and Embedded Systems Technology)

2b. Model Synthesis Using Design Patterns

- Year 1 No activity expected.
- Year 2 Development of pattern based composition and model synthesis.
- Year 3 Models of computation expressed as design patterns, development of a methodology for capturing cross cutting design constraints.
- Year 4 Development of advanced pattern based composition and model synthesis, continuation of the expression of models of computation as design patterns.
- Year 5 Enhanced methodology for capturing cross cutting design constraints and integration of model synthesis tools.

2c. Model Transformation

- Year 1 Development of meta generation framework and technology.
- Year 2 Generative extensions to meta-modeling languages.
- Year 3 New versions of Years 1,2 items and methodology for constructing embeddable generators.
- Year 4 Continuation of embeddable generators.
- Year 5 Integration of embeddable generators on NEPHEST.

3. Advanced Tool Architectures

3a. Syntax and Semantics

- Year 1 Abstract semantics for continuous and discrete models
- Year 2 Extensions for mixing time and untimed models, construction of experimental toolkit for actor-oriented models.
- Year 3 Fixed Point Semantics, prototype visual actor-oriented framework, interface definition of modal models.
- Year 4 High order components: visual representations, coupling of modal mechanisms with analysis tools.
- Year 5 Delivery of integrated tool suite with integration of composition, visual framework design and modal modeling into NEPHEST.

3b. Interface Theories

- Year 1 Identification of compatibility checks
- Year 2 Definition of interfaces for compatibility checks.
- Year 3 Prototype compatibility and support infrastructure.
- Year 4 Refinement of toolkit
- Year 5 Delivery of demonstration software for compatibility checking.

3c. Virtual Machine Architectures

- Year 1 Evaluation of E-machine and Java Virtual Machine.
- Year 2 Next generation virtual machine definition.
- Year 3 Application of virtual machine in test systems.
- Year 4 Revision of virtual machine architecture.

- Year 5 Delivery of virtual machine architecture definition.

3d. Components for Embedded Systems

- Year 1 Evaluation of a toolkit for embedded systems
- Year 2 Definition of a toolkit architecture and repository technology.
- Year 3 Conversion of existing component architecture.
- Year 4 Integration of toolkits and toolkit architecture.
- Year 5 Delivery of toolkit based framework.

4. Experimental Research

4a. Embedded Control Systems

- Year 1 Embedded Avionics for single aerobot using Giotto and platform based approaches.
- Year 2 Embedded Veitronics experiment using time triggered embedded controller
- Year 3 Embedded Avionics for multiple aerobots using time triggered plus event triggered architectures and design suite developed in Focus Areas 2,3 above
- Year 4 Networked embedded systems for wireless applications: light weight protocols with scalability for networked embedded systems, the use of Networked embedded systems-C (Nes-C), high level languages for programming networked embedded systems.
- Year 5 Integration of experiments and capstone experiment with 10^4 networked embedded systems.

4b. Embedded Software for National and Homeland Security

- Year 1 Embedded Software for Onboard Avionics "Softwalls", conflict detection and resolution.
- Year 2 Development of Softwalls and integration into commercial flight simulators, human factors study for air traffic management using embedded tools.
- Year 3 Secure versions of Softwalls and Air traffic software tools
- Year 4 High confidence versions of onboard and ground tools.
- Year 5 Lessons learned and certification issues for embedded software in avionics and ground based systems.

4c. Networks of Distributed Sensors

- Year 1 Testbeds of distributed sensors for tracking targets.
- Year 2 Test beds for plume detection, climate monitoring. Services for coordination: time synchronization, self-stabilization.
- Year 3 Sensorwebs and closing the loop in building environments.
- Year 4 Lessons learned for designs for networked embedded systems.
- Year 5 Technology transfer using open experimental platforms.

5. Education and Outreach

5a. Curriculum Development for Modern Systems Science

5b. Undergraduate Curriculum Insertion and Transfer
5c. Summer Internship Program in Hybrid and Embedded Software.

The activities here will be continuous and on going through out the life of the ITR and beyond with development of new courses, accreditation, training of community college and other teachers, summer programs as stated in the deliverables. No specific timeline of deliverables is given since the activities are continuous.

Management Plan

The organization chart for the project management is as in the proposal submitted. Here we give a description of the chart only owing to restrictions in including graphics in this document. The Project Director, Prof. Sastry, will be responsible for the overall management of the project and will chair the Executive Council. The role of the Executive Council is to perform strategic planning, research and outreach coordination, budget allocation, and measurement of the effectiveness of the overall research, education, and dissemination efforts. The members of the Executive Council are Profs. Henzinger, Lee and Sangiovanni-Vincentelli (UCB), and Prof. Sztipanovits (VU). The members of the Executive Council are the principal investigators of the four major research focus areas: Prof. Henzinger – Hybrid Systems Theory; Prof. Sztipanovits – Model-based Design, Prof. Lee – Advanced Tool Architectures; and Prof. Sangiovanni-Vincentelli and Sastry – Experimental Research. The research focus areas will be complemented by the Curriculum Development and Education and Outreach areas. The leadersof Curriculum Development are Prof. Lee and Varaiya (UCB); the leader of Education Outreach is Prof. Karsai (VU), in cooperation with Dr. Brian Williams (VU). Each leader will be responsible for developing the research and technology activities of the research focus area. Each area will have a number of projects; the investigators in these projects constitute the focus area committee and are responsible for executing the focus area research agenda.

The project team will be supported by a Board of Advisors chosen from industry and academia. The choice of the Board of Advisors will be discussed with NSF program staff for their input and feedback. The primary role of the Board of Advisors is to provide feedback to the team on progress and effectiveness in industrial applications. Members will be major industry stakeholders in advances in embedded software technology (GM, Ford, Boeing, Honeywell, Lockheed-Martin, Motorola, Intel), representatives from the tool industry (Cadence, Synopsys), and representatives from other academic groups such as Stanford, the University of Pennsylvania, the DARPA-SRC MARCO Research Centers (including CMU and Georgia Tech), the Technical University of Vienna, and the EU Group on Embedded Software, Artiste, headed by Prof. Sifakis from CNRS-VERIMAG, Grenoble, France.

The Board of Advisors will meet twice per year to review research activities. The semi-annual meetings will alternate between California and Tennessee and will have a retreat format with participating faculty, students, and visiting researchers presenting research results, progress, and future directions. The project team will meet immediately after each of the retreats, to collect the feedback from the retreats and determine new ideas and

directions with an emphasis on the individual focus areas. The first meeting each year will be in late October and the second meeting each year will be in early May starting with 2002. We expect that the late October/early November meeting dates will be negotiated with NSF program staff to allow for review of the progress to date. We expect the kickoff to be the first meeting.