Foundations of Hybrid and Embedded Software and Systems: Project Overview

Edited and presented by
S. Shankar Sastry, PI
UC Berkeley

Chess Review
October 4, 2006
Alexandria, VA

NSF-ITR Investigators

Ruzena Bajcsy, Ras Bodik, Bela Bollobas, Gautam Biswas,Tom Henzinger, Kenneth Frampton, Gabor Karsai, Kurt Keutzer, John Koo, Edward Lee, George Necula, Alberto Sangiovanni Vincentelli, Shankar Sastry, Janos Sztipanovits, Claire Tomlin, Pravin Varaiya.

ITR-Center Mission

- The goal of the ITR is to provide an environment for graduate research on the design issues necessary for supporting next-generation embedded software systems.
  - The research focus is on developing model-based and tool-supported design methodologies for real-time fault-tolerant software on heterogeneous distributed platforms.

- The Center maintains a close interaction between academic research and industrial experience.
  - A main objective is to facilitate the creation and transfer of modern, "new economy" software technology methods and tools to "old economy" market sectors in which embedded software plays an increasingly central role, such as aerospace, automotive, and consumer electronics.

Mission of Chess

To provide an environment for graduate research on the design issues necessary for supporting next-generation embedded software systems.

- Model-based design
- Tool-supported methodologies

For

- Real-time
- Fault-tolerant
- Robust
- Secure
- Heterogeneous
- Distributed Software

We are on the line to create a "new systems science" that is at once computational and physical.

The fate of computers lacking interaction with physical processes.

Hybrid and Embedded Software:
Problem for Whom and What have we done

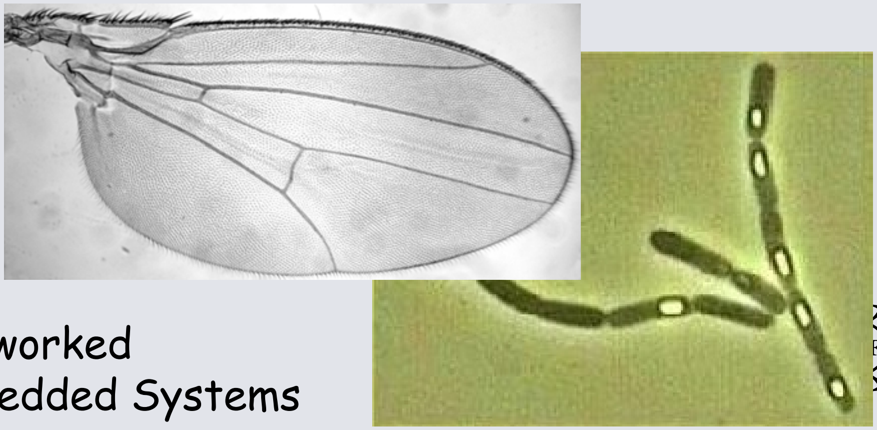
- *DoD (from avionics to micro-robots)*
  - *Essential source of functionality/superiority*
  - *UAV flight control, F-22/F-35 avionics, UAR*
- *Automotive (drive-by-wire(less)?)*
  - *Key competitive element:*
  - *Studies for Ford, GM, Toyota, Siemens*
- *Ubiquitous Computing Devices (from mobile phones to TVs to sensor webs)*
  - *Networked Embedded Systems*
  - *Several generations of Sensor Webs/Motes*
- *Plant Automation Systems*
  - *SCADA/DCS in Critical Infrastructure Protection*
  - *Closing the loop around sensor webs*

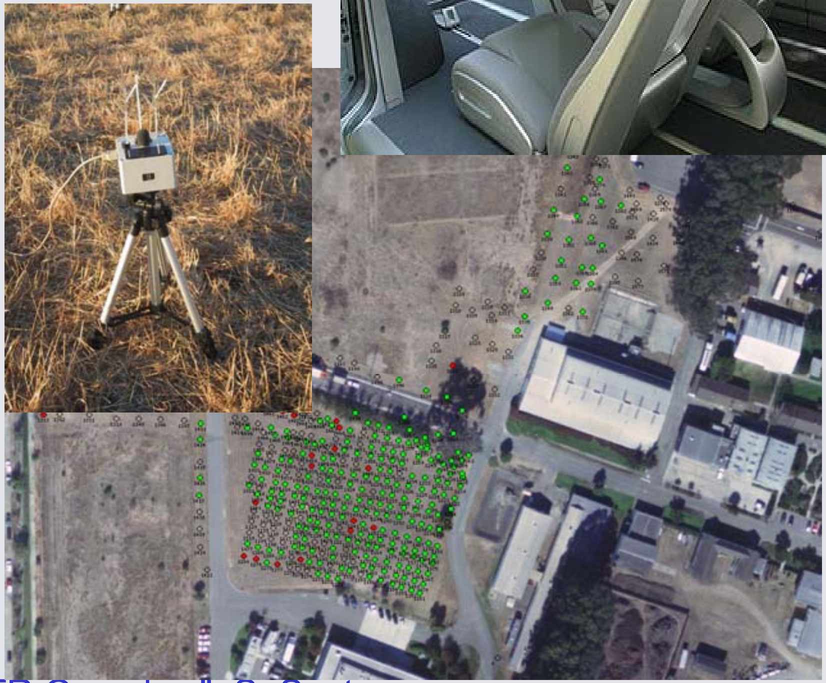
Some Applications Addressed

Automotive

Avionics: UAVs

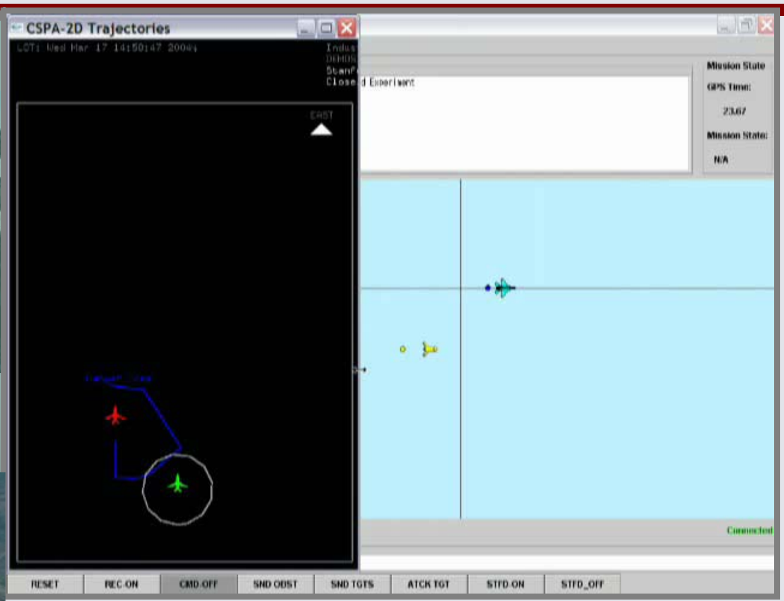
Systems Biology





Networked
Embedded Systems

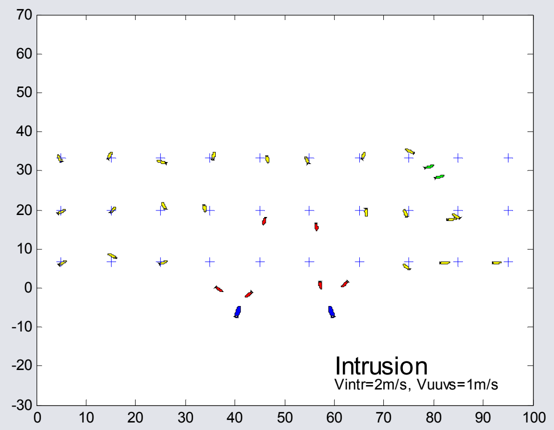
More Applications



Conflict Detection and Resolution for Manned and Unmanned Aircraft

Shooter Localization using Vanderbilt Algorithms

Pursuit Evasion for UUVS



Project Approach

- Model-Based Design (the view from above)
  - principled frameworks for design
  - specification, modeling, and design
  - manipulable (mathematical) models
  - enabling analysis and verification
  - enabling effective synthesis of implementations
- Platform-Based Design (the view from below)
  - exposing key resource limitations
  - hiding inessential implementation details
- Tools
  - concrete realizations of design methods

Key Properties of Hybrid & Embedded Software Systems

- Computational systems
  - but not first-and-foremost a computer
- Integral with physical processes
  - sensors, actuators
- Reactive
  - at the speed of the environment
- Heterogeneous
  - hardware/software, mixed architectures
- Networked
  - adaptive software, shared data, resource discovery
  - Ubiquitous and pervasive computing devices

Foundational Research

- The science of computation has systematically abstracted away the physical world. The science of physical systems has systematically ignored computational limitations. Embedded software systems, however, engage the physical world in a computational manner.
- We believe that it is time to construct an Integrated Systems Science (ISS) that is simultaneously computational and physical. Time, concurrency, robustness, continuums, and resource management must be remarried to computation.
- Mathematical foundations: Hybrid Systems Theory: Integrated Systems Science.

… and Embedded Software Research

- Models and Tools:
  - Model-based design (platforms, interfaces, meta-models, virtual machines, abstract syntax and semantics, etc.)
  - Tool-supported design (simulation, verification, code generation, inter-operability, etc.)
- Applications:
  - Flight control systems
  - Automotive electronics
  - National experimental embedded software platform
- From resource-driven to requirements-driven embedded software development.

Some Current Research Focus Areas

- Software architectures for actor-oriented design
- Interface theories for component-based design
- Virtual machines for embedded software
- Semantic models for time and concurrency
- Design transformation technology (code generation)
- Visual syntaxes for design
- Approximate Solutions to H-J equations and controller synthesis
- Autonomous vehicles: Time triggered architectures for rotorcraft, pursuit evasion games for UUVs
- Automotive systems design
- Networked Embedded Systems
- Systems Biology

Tool Development Efforts

- GME
- GReAT
- DESERT
- Fresco
- Giotto/Massaccio
- Ptolemy
- HyVisual
- Metropolis
- Hyper
- MESCAL

NSF ITR Organization

- PI: Shankar Sastry
- coPIs: Tom Henzinger, Edward Lee, Alberto Sangiovanni-Vincentelli, Janos Sztipanovits
- Participating Institutions: UCB, Vanderbilt, Memphis
- Five Thrusts:
  - Hybrid Systems Theory (Tomlin/Henzinger)
  - Model-Based Design (Sztipanovits)
  - Advanced Tool Architectures (Lee)
  - Applications: automotive (ASV), unmanned systems (Tomlin/Sastry), biology (Tomlin), mechanical systems (Frampton)
  - Education and Outreach (Karsai, Lee, Varaiya)
- Five year project: kick-off meeting November 14th , 2002. Reviews May 8th, 2003, Dec 3rd, 2003, May 10th, 2004, Nov 18th 2004, May 12th, 2005, November 21st , 2005, October 4th 2006
  - Weekly seminar series
  - Ptolemy workshop May 9th, 2003, April 27th 2004,
  - Networked Embedded Systems CHESS Workshop May 9th, 2003
  - BEARS Open House, February 27th 2004, February 25th 2005, February 23rd 2006

Thrust 1 Hybrid Systems

- Deep Compositionality
  - Assume Guarantee Reasoning for Hybrid Systems
  - Practical Hybrid System Modeling Language
  - Interface Theory for hybrid components
- Robust Hybrid Systems
  - Bundle Properties for hybrid systems
  - Topologies for hybrid systems
  - Stochastic hybrid systems
- Computational hybrid systems
  - Approximation techniques for H-J equations
  - Synthesis of safe and live controllers for hybrid systems
- Phase Transitions and Network Embedded Systems

Thrust II: Model Based Design

- Composition of Domain Specific Modeling Languages
  - Meta Modeling
  - Components to manipulate meta-models
  - Integration of meta-modeling with hybrid systems
- Model Synthesis Using Design Patterns
  - Pattern Based Modal Synthesis
  - Models of Computation
  - Design Constraints and Patterns for MMOC
- Model Transformation
  - Meta Generators
  - Semantic Anchoring
  - Construction of Embeddable Generators

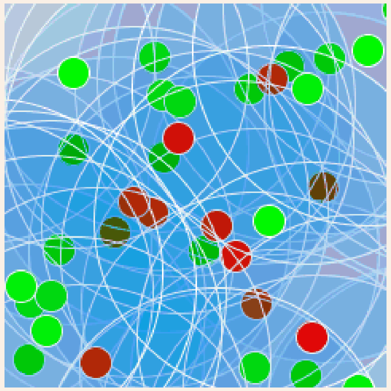
Thrust III: Advanced Tool Architectures

- Syntax and Synthesis
  - Semantic Composition
  - Visual Concrete Syntaxes
  - Modal Models
- Interface Theories
- Virtual Machine Architectures
- Components for Embedded Systems

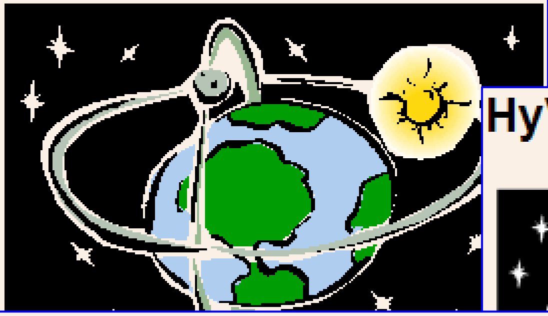
Software Releases

VisualSense

Visual editor and simulator for wireless sensor network systems

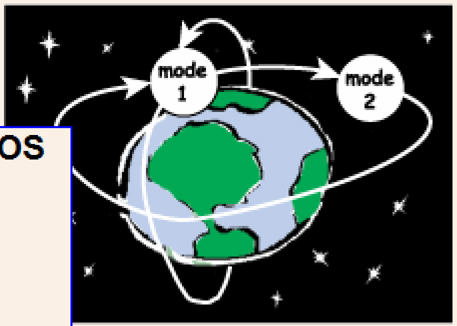


Ptolemy II



Metropolis: Design Environment for Heterogeneous Systems

HyVisual - Hybrid System Visual Modeler



Viptos - Visual interface between Ptolemy and TinyOS

The Hyper toolbox (in development)

- Inspired by hybrid systems domain
- Consider Interchange Format Philosophy:
  - For all models which *could be* built in Tool$_1$ or Tool$_2$ (i.e., as defined by A_1) there must exist a translator to/from an Interchange Format
- Alternative philosophy:
  - For a model, *m*, built in Tool$_1$ or Tool$_2$, this model may be translated to the other tool *if* the semantics used by *m* are an intersecting subset of the semantics $S_1 \cap S_2$.

Tool$_1$ = $\langle C_1, A_1, S_1, M_{s1}, M_{c1} \rangle$

C = Concrete Syntax, A = Abstract Syntax, S = Semantics

M_s = Semantic Mapping, M_c = Concrete Syntax Mapping

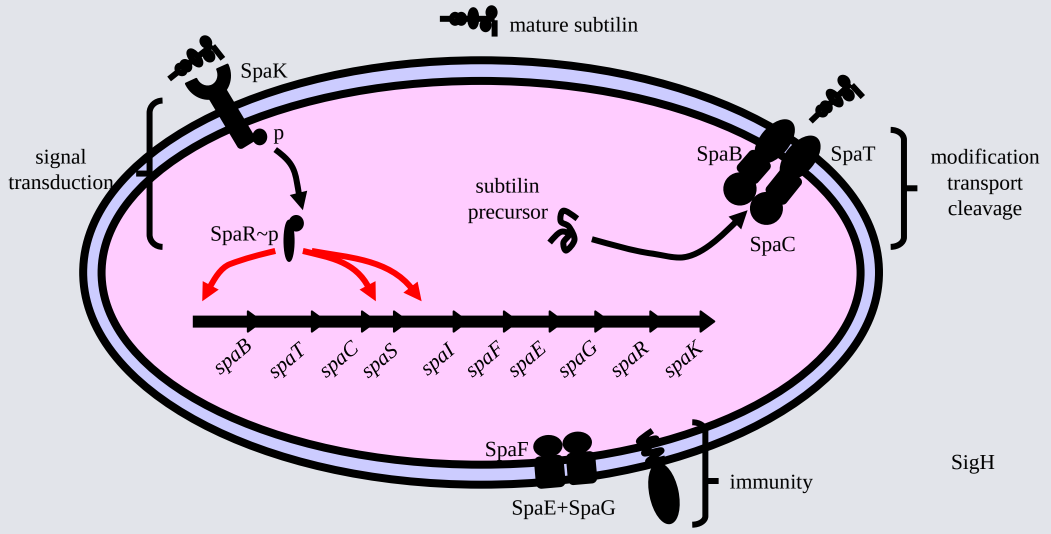
The Hyper toolbox (in development)

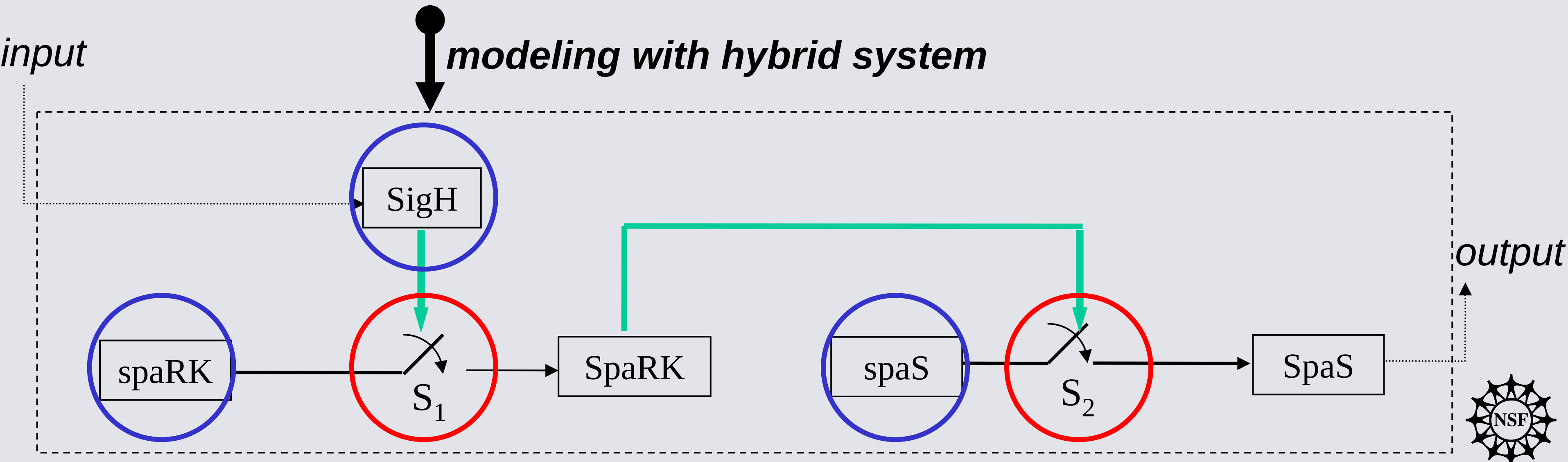
- Examine semantics used by a model to determine compatibility
- This provides several potential uses
  - Produce $\text{Tool}_{1\cap 2}$ after user request for models compatible across Tool_1, Tool_2
  - Check to see if model m_3, produced in $\text{Tool}_{1\cap 3}$ is compatible with Tool_2
  - Produce $\text{Tool}_{\text{simulate}\cap\text{verify}}$ when capability is more important than specific semantics
- Implementation strategy
  - *Strong typing, metamodeling of type structures*
  - Previous Chess work in operational semantics and Interchange Formats

Thrust IV: Applications

- Embedded Control Systems
  - Avionics: F-22, F-35, UAV flight control, Open Control Platform
  - Veitronics: Engine control, Braking control, architectures
- Embedded Systems for Unmanned Systems
  - Air Traffic Control; Smart Walls, Sector Control
  - UAVs: flight control, autonomous navigation, landing
  - UUVs; pursuit evasion games
- Networks of Distributed Sensors and Networked Embedded Systems
- Stochastic Hybrid Systems in Systems Biology
- Hybrid Models in Structural Engineering
  - Active Noise Control
  - Vibration damping of complex structures

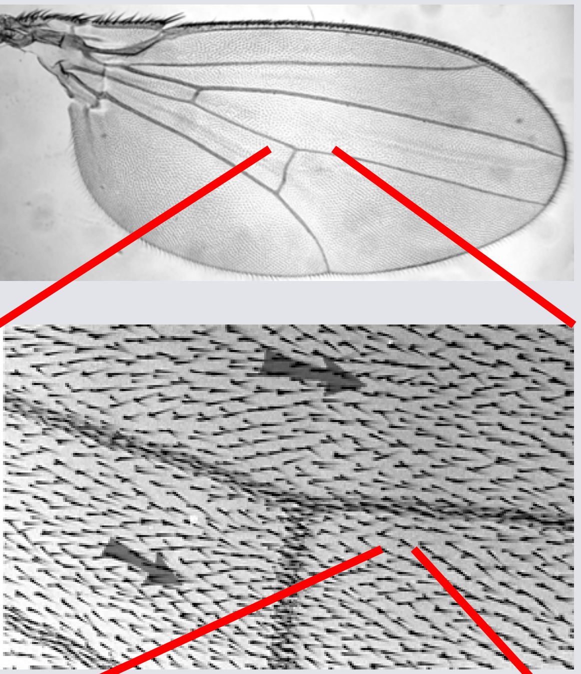
Antibiotic biosynthesis in *Bacillus subtilis*





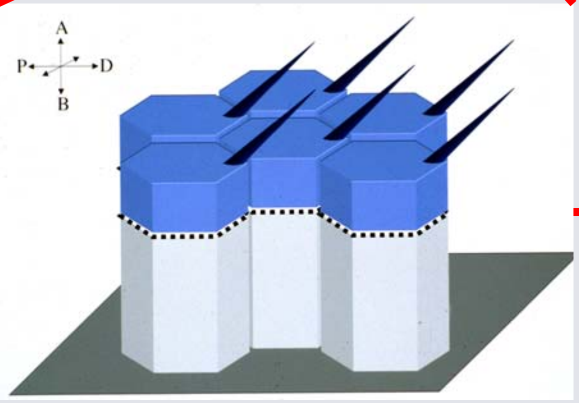
○ = *discrete states (with randomness)* ○ = *continuous states*

Planar cell polarity in *Drosophila*

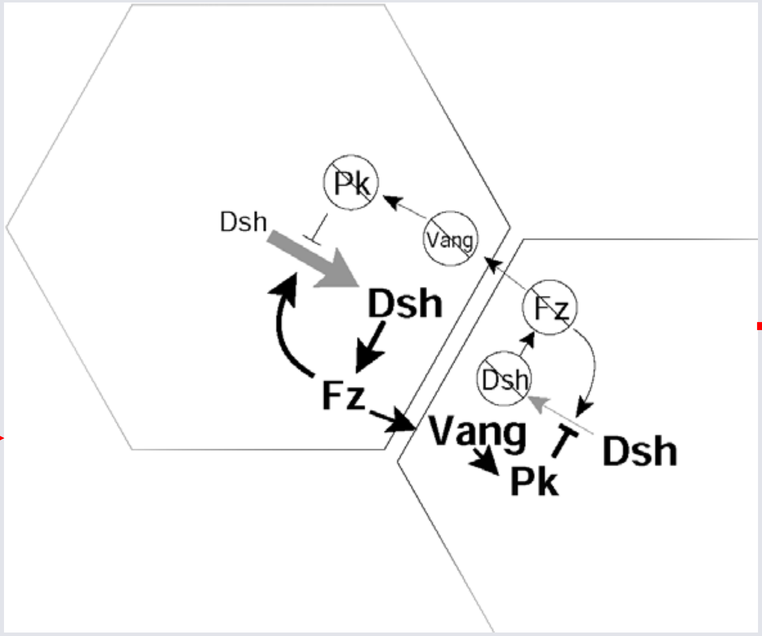


phenotype

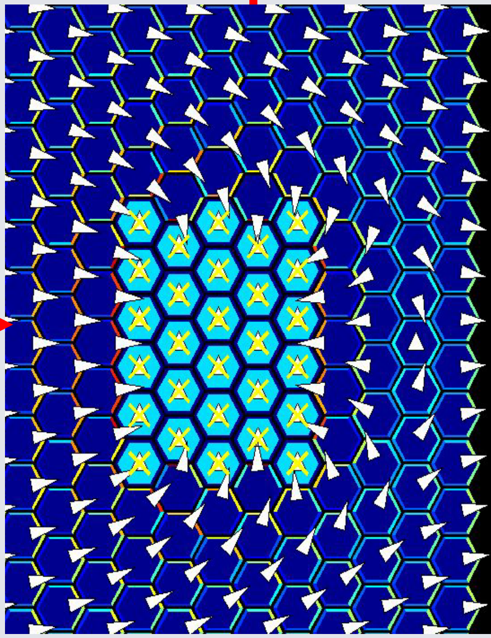
•Simulations
•Parameters estimation
•Study of mutants



cell model



proteins feedback network



Thrust V: Education and Outreach

- Curriculum Development for MSS
  - Lower Division
  - Upper Division
  - Graduate Courses
- Undergrad Course Insertion and Transfer
  - New courses for partner institutions (workshops held March 1st 2003, Summer 2004), ABET requirements
  - Introduction of new undergrad control course at upper division level by embedded control course coordinated with San Jose State
  - CHESS-SUPERB/ Summer Program in Embedded Software Research SIPHER program (6 + 4 students in Summer 03, 3 + 5 in Summer 04, 6+4 students in Summer 05, 6 students in Summer 06)
- Graduate Courses
  - EECS 249 Design of Embedded Systems: Models, Validation, and Synthesis
  - EECS 290N Concurrent Models of Computation for Embedded Software
  - Vanderbilt EECE 395 / EECS 291E/ME 290S Hybrid Systems
  - Repositorying using VanTH software

Knowledge Transfer Continued

- Three NITRD-HCSS studies
  - High Confidence Medical Devices and Systems: Philadelphia, June 2005. Sastry, Sztipanovits organizing committee members, follow up meeting at Vanderbilt: Dec. 2005
  - Aviation Safety and Certification: Planning Meeting Seattle Nov 9, 10th 2005, full meeting Oct 5, 6, 2006 Tomlin study leader.
  - High Confidence SCADA systems: Planning meeting, Washington, DC March 21-23, 2006, full meeting Pittsburgh, November
- OSD Software Producibility Workshops for Dr. Robert Gold
  - Berkeley Planning Workshop July 2005
  - Arlington: OSD + Industry + Academia Workshop May 2006

Knowledge Transfer Industry Initiatives

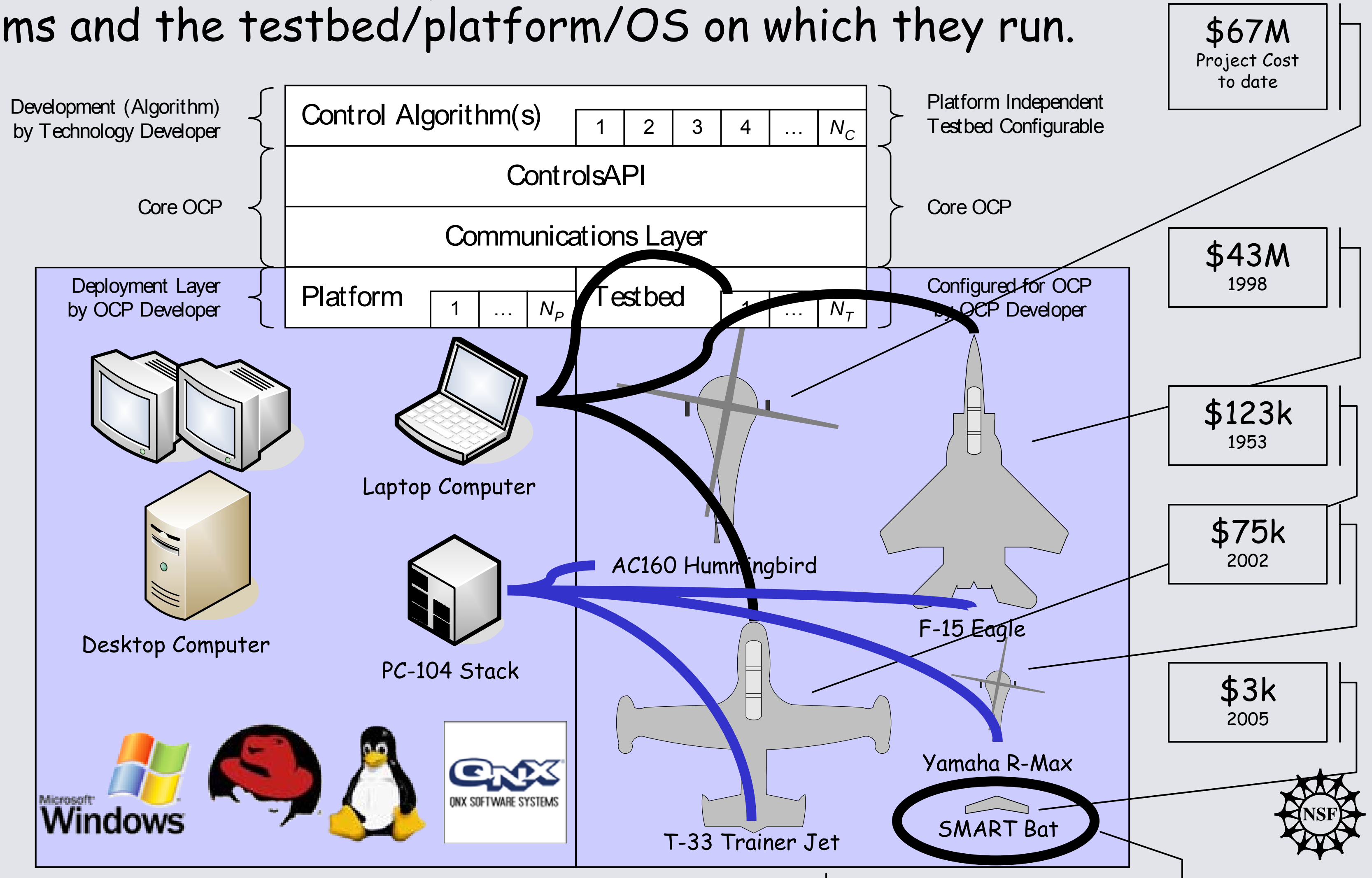
- GM through ESCHER and GM Scientific Board (Sangiovanni Vincentelli)
- Boeing and Raytheon (Sztipanovits, Sprinkle, Sastry, Tomlin)
- United Technologies (Sastry and Sangiovanni)
- Siemens (Lee and Tomlin)
- Ford (Sastry with MIT-Ford Alliance)
- Telecomm Italia (Sangiovanni Vincentelli)
- Lockheed (Sprinkle, Schmidt)
- Bosch, Toyota (Sprinkle, Lee)

Knowledge Transfer

- Interaction with EU-IST programs
  - Columbus (with Cambridge, l'Aquila, Rome, Patras, INRIA)
  - Hybridge, Hycon (with Cambridge, Patras, NLR, Eurocontrol, Brescia, KTH)
  - ARTISTE, ARTIST-2: Educational Initiatives (Grenoble, INRIA, ETH-Zurich)
  - RUNES EU-IST program in network embedded systems (Ericsson, KTH, Aachen, Brescia, Pisa, Patras, …)
  - EU-US Embedded Systems meeting, Paris, July 2005 organized by Sztipanovits, June 2006 organized by Sastry
- Foundation of non-profit ESCHER
  - Interaction with F-22/JSF design review teams
  - Secure Networked Embedded Systems: TinyOS, Tiny DB, TinySec,
  - Bio-SPICE repository

The Embedded Open Control Platform (EOCP)

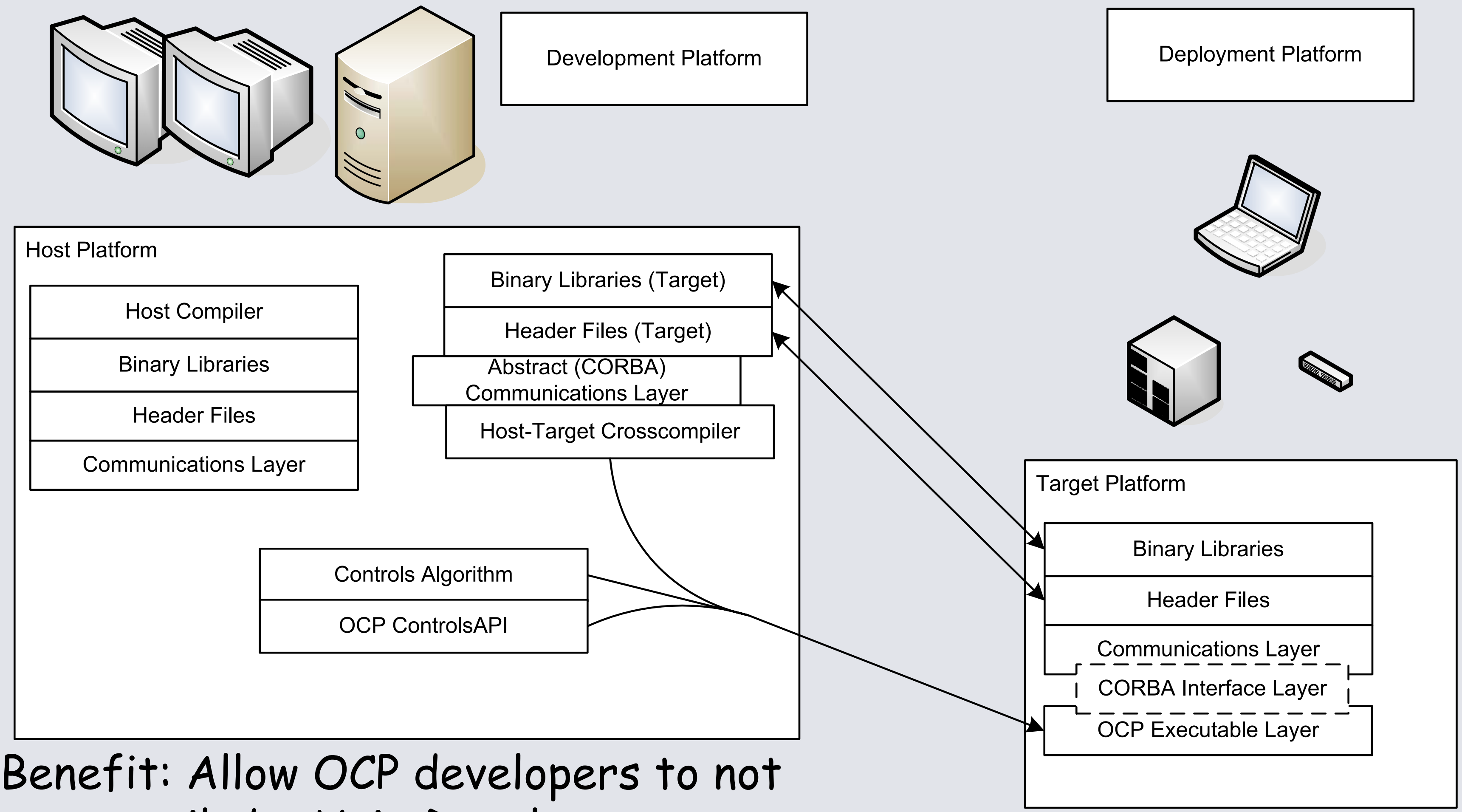
OCP provides an insulation layer between software-based control algorithms and the testbed/platform/OS on which they run.



No existing platform of appropriate form-factor/weight

Development, Deployment, and Demystification

Objective: Separate development and deployment platforms, provide out-of-the-box self-configuration scripts for new dev/deploy platforms



Benefit: Allow OCP developers to not necessarily be Unix Developers

Summary

- Foundations of Hybrid and Embedded Systems ITR focus for new approaches:
  - Theory
  - Modeling Framework
  - Tool Architectures
  - Applications with specific stakeholder interest
- Catalyst for new initiatives
  - Software Producibility in OSD
  - Network Embedded Systems → Cyber Physical Systems
  - High Confidence Embedded Systems → TRUST

Summary (continued)

- Education
  - Undergrad Courses introduced
  - Grad Courses developed
  - Remaining to do
    - Modularize courses
    - Repository content (VanTh: CAPE/eLMS)
- Outreach
  - Major element SUPERB and SIPHER (average of 8 undergrad researchers from HBCU/MI institutions per year)
  - Outreach to California Community College System and San Jose State, Fisk
  - Some outreach to Oakland highschools BFOIT in years 1, 2 of ITR