ITO Initiatives in Embedded Software

Dr. Shankar Sastry, DARPA/ITO
Control and Dynamical Systems Future Directions
University of Maryland, June 16th, 2000

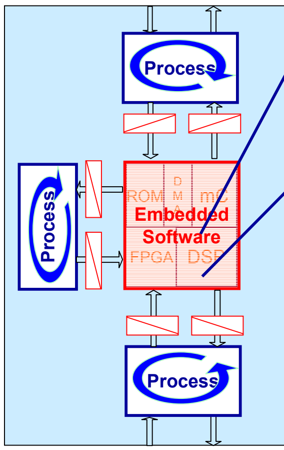
Outline

- **What is the problem?**
- **How can we solve it?**
  - Software and Physics
  - Embracing Change
  - Dynamic Structures
- **Summary of ITO Initiatives**

2

The Technology Challenge

Embedded systems: information systems tightly integrated with physical processes



Problem indicators:
- **Integration cost is too high (40-50%)**
- **Cost of change is high**
- **Design productivity crisis**

Root cause of problems is the emerging new role of embedded information systems:
- **exploding integration role**
- **new functionalities that cannot be implemented otherwise**
- **expected source of flexibility in systems**

Problem: Lack of Design Technology aligned with the new role

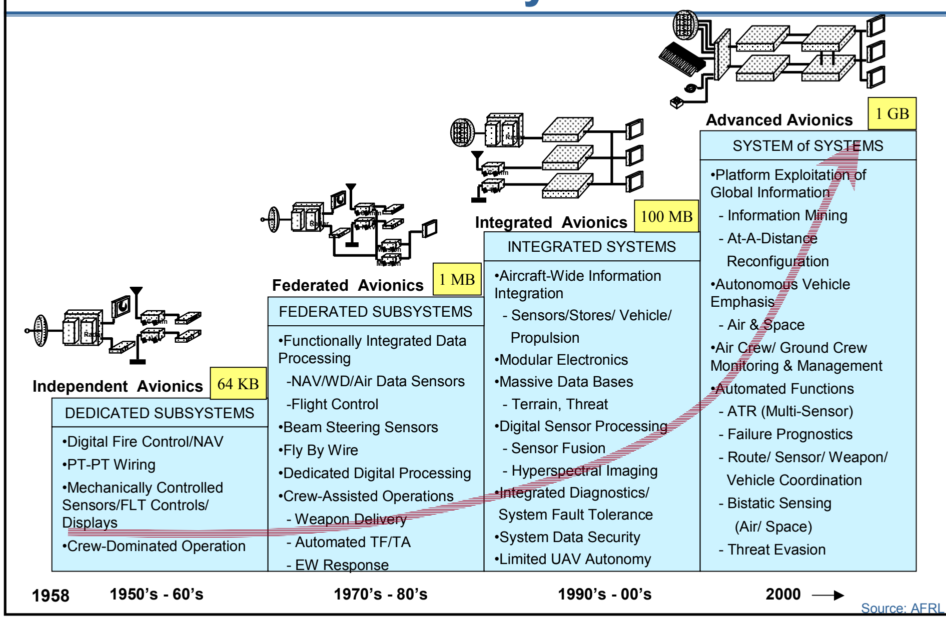
3

Problem for Whom?

- **DoD (from avionics to micro-robots)**
  - Essential source of superiority
  - Largest, most complex systems
- **Automotive (drive-by-wire)**
  - Key competitive element in the future
  - Increasing interest but low risk taking
- **Consumer Electronics (from mobile phones to TVs)**
  - Problem is generally simpler
  - US industry is strongly challenged
- **Plant Automation Systems**
  - Limited market, conservative approach

4

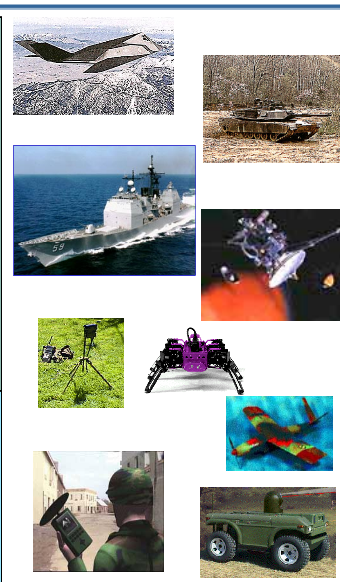
DoD Example: Avionics Systems



5

Technology Themes

- **Software and Physics**
  - Establish composability in SW for physical characteristics; System/software co-design and co-simulation environments; New methods for system/code composition
- **Embracing Change**
  - Adaptive Component Technology; Adaptable composition frameworks; QoS middleware for embedded systems
- **Dealing with Dynamic Structures**
  - Property prediction without assuming static structures; Monitoring, controlling and diagnosing variable structure systems;



6

Why Should We Do It?

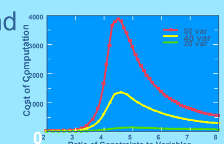
- **Themes 1 & 2:**
  - These problems hurt: cost, schedule, performance
  - The trend of IT becoming the universal integrator for systems continues and both unstoppable and necessary.
  - We have already started the work, have preliminary results and know what to do.
- **Theme 3:**
  - The wave is coming:
    - Tremendous progress in MEMS, photonics and, communication technology - **we need to build systems now!**
    - Identified applications with very high expected ROI (prognostic health management, unmanned vehicle fleets, active materials,..)

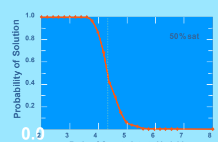
7

Why Can We Make a Difference?

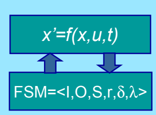
New, critical insights in fundamentals:

Phase transitions have been found in computational requirements for solving fundamental "intractable" problems.



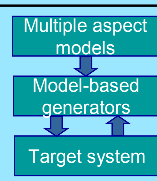


Emerging theory of hybrid systems provides a new mathematical foundation for the design and verification of embedded systems



- model checking
- compositional synthesis
- simulation

Revolutionary changes in software creation: model-based generators, aspect languages, DSL-s offer new foundation for design automation and adaptation.

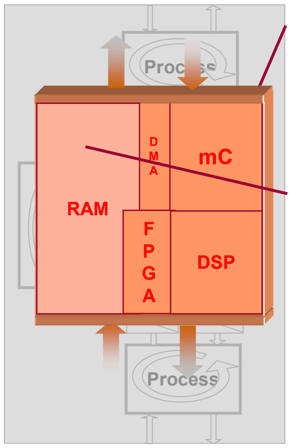


- formal modeling
- verification tools
- automated code synthesis

8

Theme 1:
Software and Physics

Embedded software: defines physical behavior of a complex nonlinear device



Embedded System: a physical process with dynamic, fault, noise, reliability, power, size characteristics

Embedded Software: designed to meet required physical characteristics

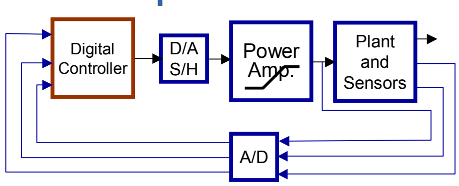
Hard Design Problem:
- Both continuous and discrete attributes (a lot)
- Every module has impact on many attributes (throughput, latency, jitter, power dissipation,..)
- Modules contend for shared resources
- Very large-scale, continuous-discrete, multi-attribute, densely-connected optimization problem

Primary challenge: Cost-cutting physical constraints destroy composability

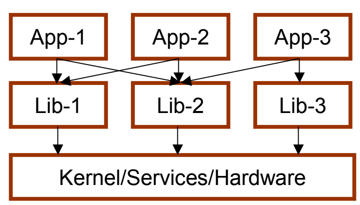
9

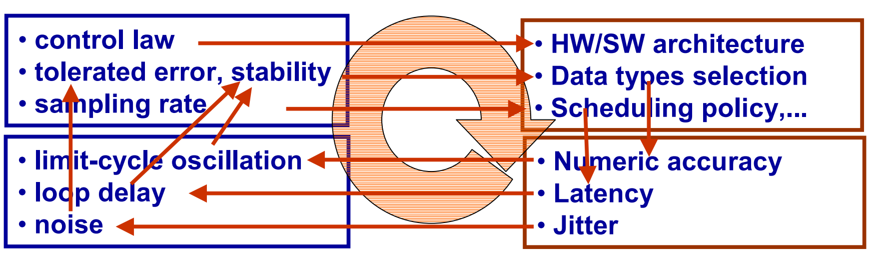
Illustrative Example for the Difficulties

Simple Controller



Embedded Software





10

Why Is this a Problem?

We have focused on functional composition…

Subsystem A | Subsystem B | Subsystem C

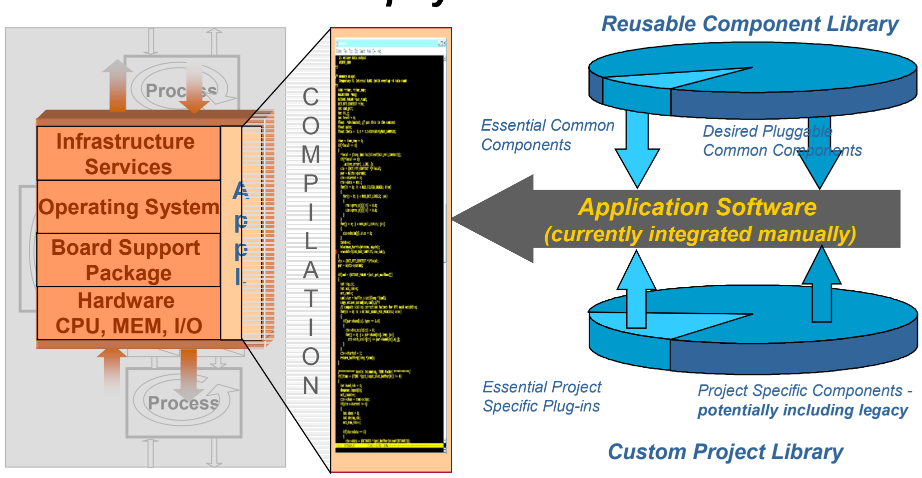
Composability: Ability to link subsystems so that properties established at subsystem levels hold at the system level

Subsystem D | Subsystem E | Subsystem F

But cross-cutting physical constraints weaken or destroy composability

Current Technology:
Functional Composition

Functional composition does not addresses physical constraints



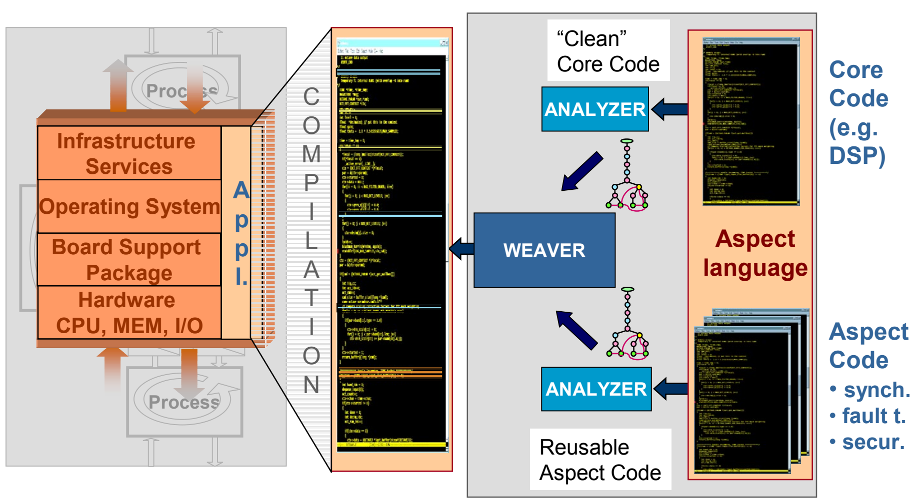
BOLDSTROKE (BOEING)

Technology Gaps

1. System/software co-design, co-simulation and analysis environments
   a. Composition of design automation tool environments
   b. Reusable components for design automation tools
2. New methods for system/code composition
   a. Model-based system composition
   b. Aspect-oriented programming
   c. Domain-specific languages
3. Frameworks and middleware to provide higher level programming abstractions
4. Hybrid optimization and analysis methods

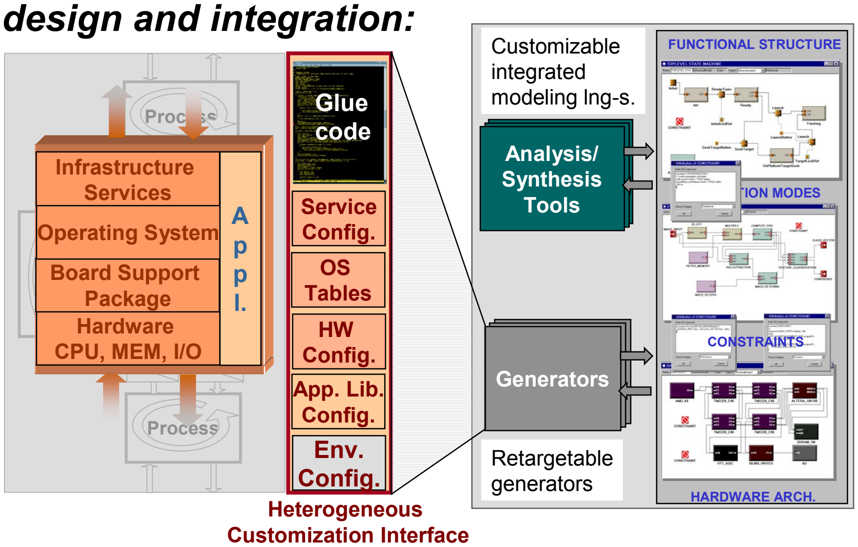
ITO: Program Composition for Embedded Systems (PCES)

Aspect languages will change programming:



ITO: Model-Based Integration of Embedded Software (MoBIES)

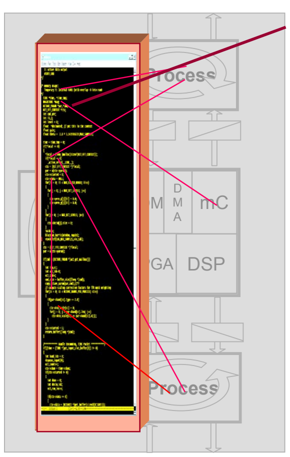
Model-based integration will change system design and integration:



15

Theme 2: Embracing Change

Source of change: environment, requirements



Hard Problem: due to its integration role, system-wide constraints accumulate in software:
- process properties - algorithms, speed, data types
- algorithms, speed, data types - resource needs
- shared resources - speed, jitter,..

..scattered all over the software.

Condition for managing change:
- constraints need to be explicitly represented
- effects of changes need to propagated by tracking constraints

Flexibility is essentially a
SYSTEM-WIDE CONSTRAINT MANAGEMENT PROBLEM

16

Technology Gaps

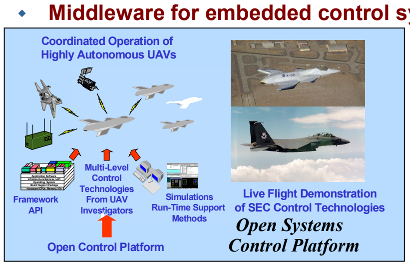
1. **Adaptive Components for Embedded Systems**
   a. embedded, active models, constraints, generators
   b. adaptive,self-monitoring, embedded software
2. **Methods to control flexibility**
   a. parametric design
   b. constraint languages
3. **Adaptable composition frameworks and QoS middleware**
4. **Programming Methods to achieve flexibility**
   a. dynamic languages
   b. programming through learning

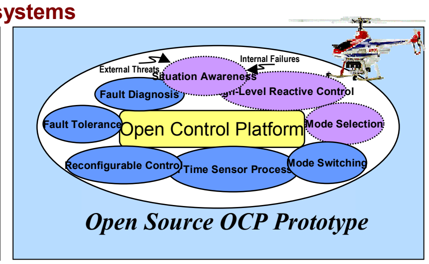
17

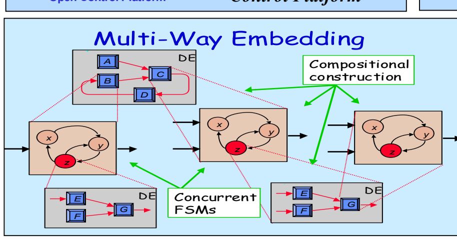
ITO: Software Enabled Control (SEC)

TECHNOLOGY GOALS:

- Control systems that we haven't been able to control before
- Increase automation for extreme maneuvers, tightly coordinated actions
- Middleware for embedded control systems







Coordinated Multi-Modal Control:

- Control middleware (reusable)
- Open systems, open source
- Reconfigurable hybrid (discrete and continuous) control loops
- Real-time data services for active (predictive) state models

18

What are We Doing? Why?

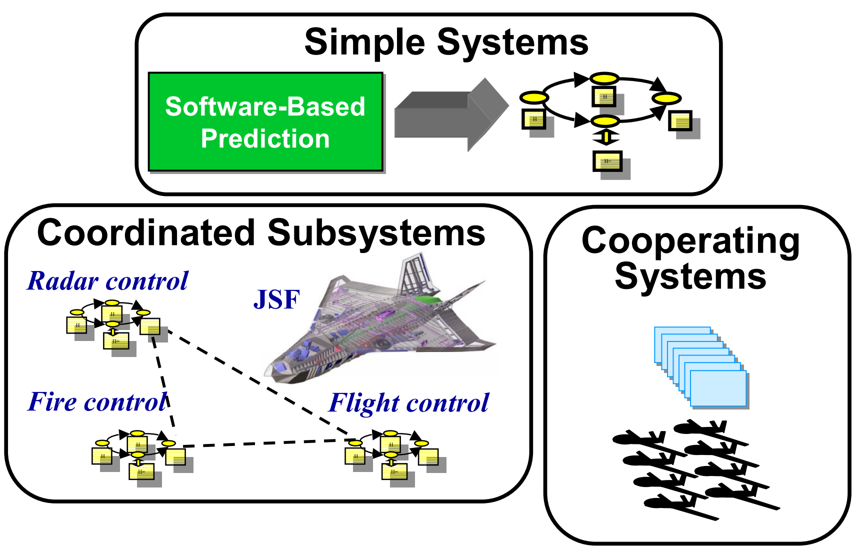
Objectives:

- Control Systems that we haven't been able to control before
- Increase automation for extreme maneuvers, improve disturbance rejection
- Provide reusable middleware for embedded software control

Current Limitations:

- Conservative, limited in capability
- Human operators "close the loop" for extreme disturbances, and high performance
- Old computational assumptions
  - Limited Resources
  - Fixed, static software designs and schedules
  - Loose integration of supervisory and "inner loop" control
  - Limited prediction (continuous only, not discrete)

Program Objectives



SEC Program Tasks

- **Active State Models**
  On-line adaptive & predictive models (e.g., of system, environment, sensors & actuators) that maximize information to controller
- **Coordinated Multi-Modal Control**
  Framework that supports precise coordination of discrete and continuous control in a single system & supports scalable coordination of subsystems and multi-systems
- **On-Line Control Customization**
  Framework for adapting controller (parameters, objective function, model configuration, discrete-continuous control structure) for current conditions
- **Open Control Platform**
  Open systems support for control systems: reusable middleware that implements core data and process services for reconfigurable, adaptive hybrid controllers

21

Active State Models

Maximal information exploitation: adaptive & predictive models

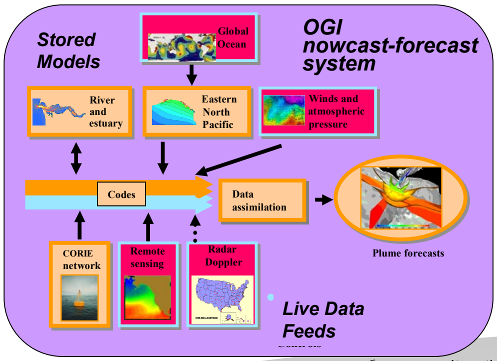
- **Control elements of the solution:**
  - Models that combine on-line information and prediction
  - High-fidelity real-time adaptive models (e.g., system dynamics, environmental data, sensor/actuator performance)
  - Dynamically aggregated models of influences on system
  - Coupled models for managing physical interaction of separate systems
  - Hybrid models for joint discrete and continuous prediction
- **Software elements of the solution:**
  - On-line support for acquisition, adaptation, and composition of active models
  - Support for on-line data stream & process configuration
  - Middleware for model acquisition, adaptation, presentation, validation, and prediction services

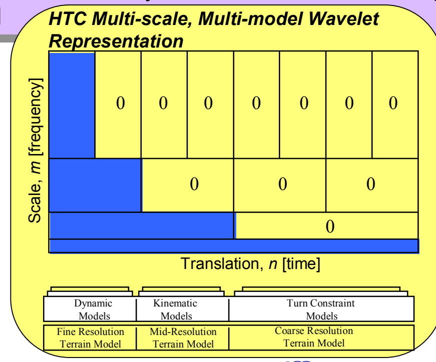
22

Active State Models

Maximal exploitation of on-line information, system dynamics

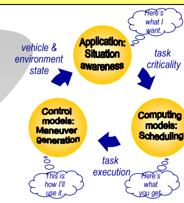
Seedling Progress: Concepts for improved (environmentally informed, multi-resolution) predictive models for model-referenced control



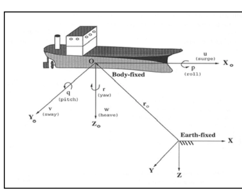


What would a task do with more time?

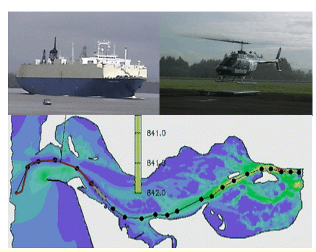
- more iterations
- more maneuvers
- longer horizon
- higher resolution
- higher fidelity model
- more samples



Active Models



High speed Container Ship



Dynamic Equations of Motion

$$M\dot{v}_r = C(v_r)v_r + D(v_r)v_r + g(\eta) = u$$

$$\dot{\eta} = J(\eta)v_r + v_c^E \qquad v_r = v - v_c$$

M = inertial matrix (including added mass)
$C(v)$ = matrix of Coriolis and centripetal terms
$D(v)$ = damping matrix
$g(\eta)$ = vector of gravitational forces and moments
u = vector of control inputs

η = position and orientation vector (earth fixed)
v = linear and angular velocities (body - fixed)
v_c = current velocities (body - fixed)

Three Types of Models:

- Single vessel model
  - Dynamics of a powered hull
- Fluid Flow in an estuary
  - Finite element model of flow in a constrained, 3D channel
  - Interacts with vessel model by superposition of velocities
- Neighboring objects model
  - Positions and velocities of nearby vessels and obstacles.

Active State Models

Representative Topics

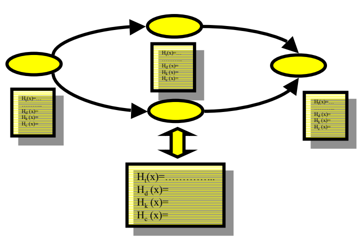
- **Nowcast/forecast software technology**
- **Wavelet multimodel support for Open Control Platform**
- **Active model representation for locality of access**
- **Symbolic methods for model reduction, manipulation**
  - Hybrid systems for model abstraction
  - Linear parameter varying models
- **On-line sensor/actuator performance modeling & prediction**
- **Uncertainty modeling for single, multi-systems**

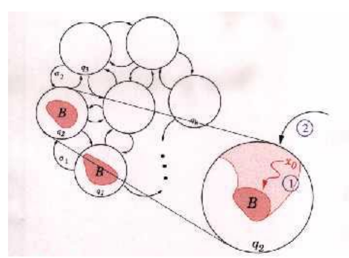
Active State Models

Example: Hybrid and Continuous Prediction

Hybrid prediction:

- *Discrete evolution (mode selection and switching logic)*
- *Continuous evolution (control law)*



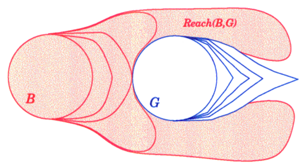


Procedure:
- start from an unsafe state (in *B*);
- approximate all states that (1) evolve to unsafe state
- approximate all states that (2) switch to unsafe state
- iterate

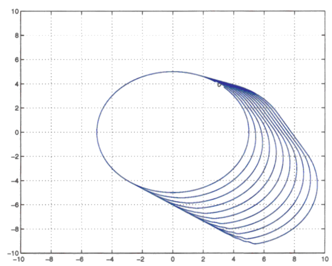
Active State Models

Example: Hybrid and Continuous Prediction

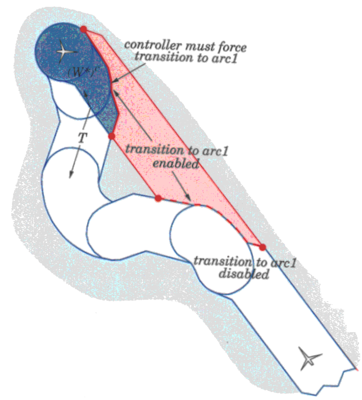
Discrete Approximation



Evolution of Reachable Sets



Constrained Transition



27

Coordinated Multi-Modal Control

Scalable single & multi-system control

- **Control elements of the solution:**
  - Mathematical foundations for hybrid discrete + continuous control (e.g., for safe mode transition), coordinated multi-mode control systems
  - Dynamic control configuration & coordination for cooperating systems
  - Hierarchical control strategies for scalable coordination of multi-systems
  - Fast methods for checking coordinated hybrid stability
- **Software elements of the solution:**
  - Systems services, middleware for coordinated hybrid controllers
  - Techniques for mapping: hybrid control models to computational models **(e.g., event-driven, time-triggered)**, and then to open system platform **(e.g., threads/schedulable units, periodic or sporadic schedules, hard & soft real time, deadlines & guarantees, QoS)**

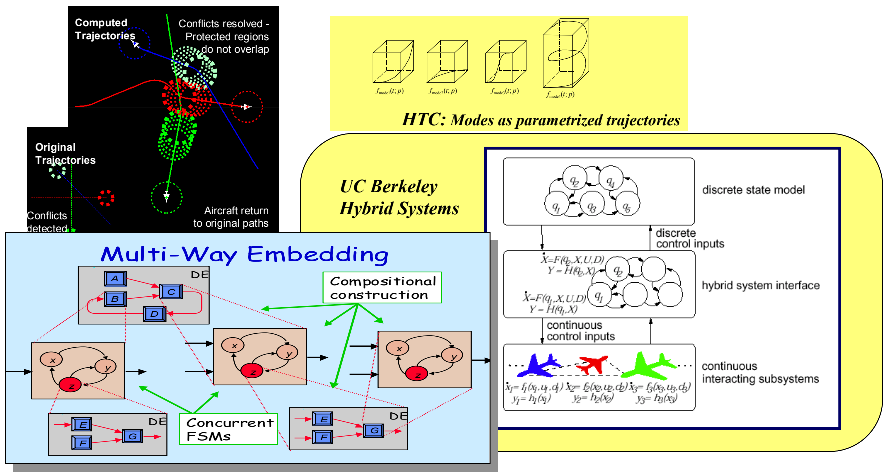
28

Coordinated Multi-Modal Control

Scalable framework for coordinated control of subsystems, multi-systems

Seedling progress :

- Hierarchical hybrid control concept for coordinated multi-agent, multi-modal control systems
- Coherent framework for discrete and continuous control, local hybrid stability
- Computational model concepts for control implementation

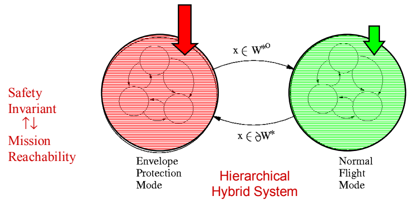


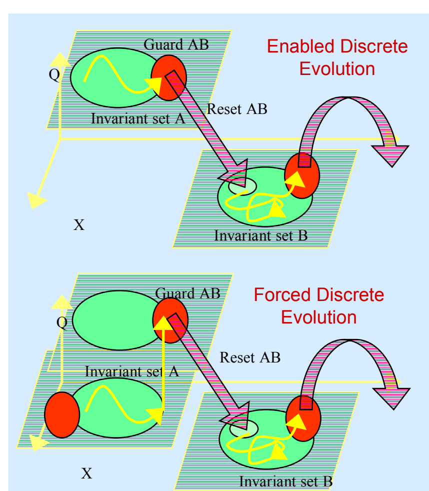
29

Coordinated Multi-Modal Control

Representative Topics

- **Multi-system hybrid control**
- **Optimal hybrid control**
- **Interoperable computational models for hybrid control design, software**
- **Asynchronous hybrid control**
- **Functional reactive programming (timed), hybrid mode switching**





30

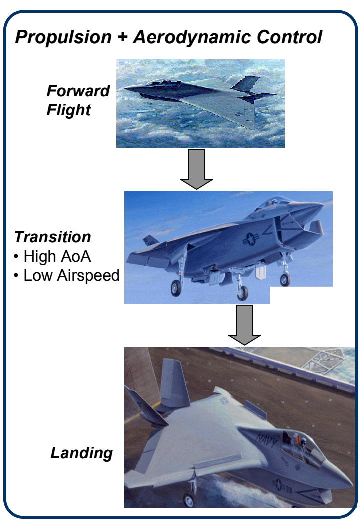
Coordinated Multi-Modal Control

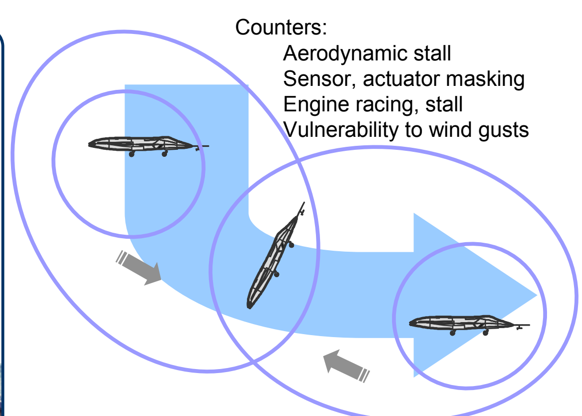
Representative Topics

- **Failure detection, isolation, and recovery (FDIR)**
  - Hybrid failure detection, isolation, and recovery
  - Multi-system hybrid FDIR
  - Validation methods for FDIR
- **Mode design**
  - Receptivity for coordination
  - Aggressive maneuver design from empirical data
  - Modes as parameterized maneuvers
- **Transition control**
  - Fuzzy mode transition control

31

Example: Coordinated Subsystem Control for STOVL/VSTOL Transition





SEC Coordinated Multi-Modal Control provides joint predictive mode transition for subsystems; control authority management; coordinated subsystems

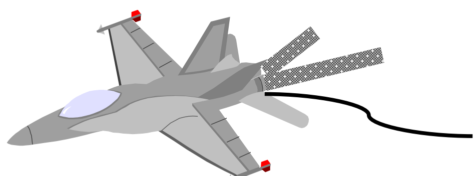
32

Coordinated Multi-System Control:
e.g., Positionable Towed Body

Now:
- Predetermined device position over entire flight envelope
- Disturbance is amplified by aircraft airstream, can go unstable

Goal:
- Detected, predict interactions
- Customize control for current flow field, aircraft dynamics, engine plume

Impact:
- Better ECM effectiveness of towed device
- Greater dynamic range (different aircraft designs)

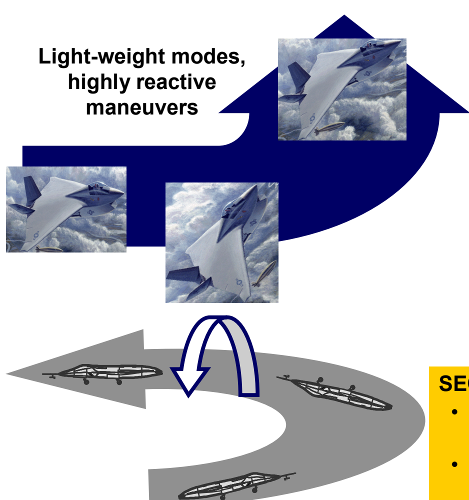


Engagement geometry:
Host aircraft position
Towed body attitude
Doppler velocity
Angular velocity
Towline motion1

SEC provides:
- ***Coordinated hybrid predictive transitions, cooperative operation***
- ***Active models of joint environment***
- ***Adaptation for conditions***

33

Mode Design for Highly Evasive UAVs



Now:
- Extreme maneuvers? Pilot closes the loop

Goal:
- Evasive, avoidance maneuvers (cobra, Immelman, jinking...)

Impact:
- Reduced human risk
- Survivable, reusable materiel
- Reconfigurable aerodynamics
- Increased rate of successful UCAV missions
- Increased applicability: SEAD, combat
- Maximized vehicle performance, range

SEC Multi-Modal Control Provides:
- ***Adaptable maneuvers & performance criteria***
- ***Escape from fixed-frequency loop design straitjacket***
- ***Real-time exploitation of information***

34

On-line Control Customization

Real-time computational, structural, parametric adaptation

- **Control elements of the solution:**
  - Bounded control recalculation to preserve safe operational envelope
  - Scalable computational control **(e.g., trajectory morphing, game-theoretic optimization, parameter-varying strategies)**
  - Symbolic computation strategies for adaptive control
  - Adaptive hybrid (discrete+continuous) control, hybrid FDIR
- **Software elements of the solution:**
  - On-line support for fast structural reconfiguration of hybrid controllers
  - Software support for timed & event-based process services, symbolic computation, process & data configuration services
  - Computational **(e.g. timing)** models for managing on-line customization
  - Resource-constrained adaptation

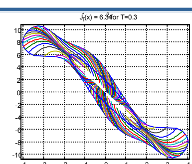
35

On-Line Control Customization

Control specialization for damage, extreme environments, extreme performance requirements

Seedling progress:

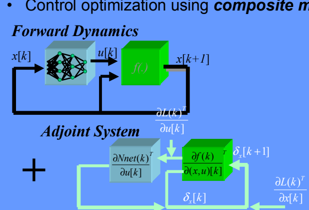
Concepts for performance improvement and disturbance rejection: on-line control recalculation, optimization, and trajectory morphing; mutable control software composition strategies



CalTech
Multilevel optimization, blends:
- receding horizon optimal control (mid- & low-level, continuous)
- dynamic programming (high level, discrete state, also receding horizon)

Control Lyapunov Function
characterization for *guaranteed* region of operation

OGI:
- Neural control: extensions to on-line (model-predictive) and receding horizon control (HJB equations + reinforcement learning)
- Control optimization using ***composite models***



*Software: **scripting approach for control customization***
formulate a control paradigm
integrate actions of diverse components (models, controllers, sensors, actuators)

Launches controlled events
by a sequence of imperative commands

Defines relations among event streams
equational specifications schedule reactions by demand

Open language format
doesn't depend upon the implementations used for operators

On-Line Control Customization Representative Topics

- Control recalculation (MPC, CLF, LPV methods)
- Aggressive maneuver/mode design
- Protocols for configurable control re-calculation
- Trajectory optimization, CLF methods
- LPV control methods for tractable calculation with reduced models
- Multi-graph methods for FDIR
- Control scripting for reconfiguring process and data stream architecture

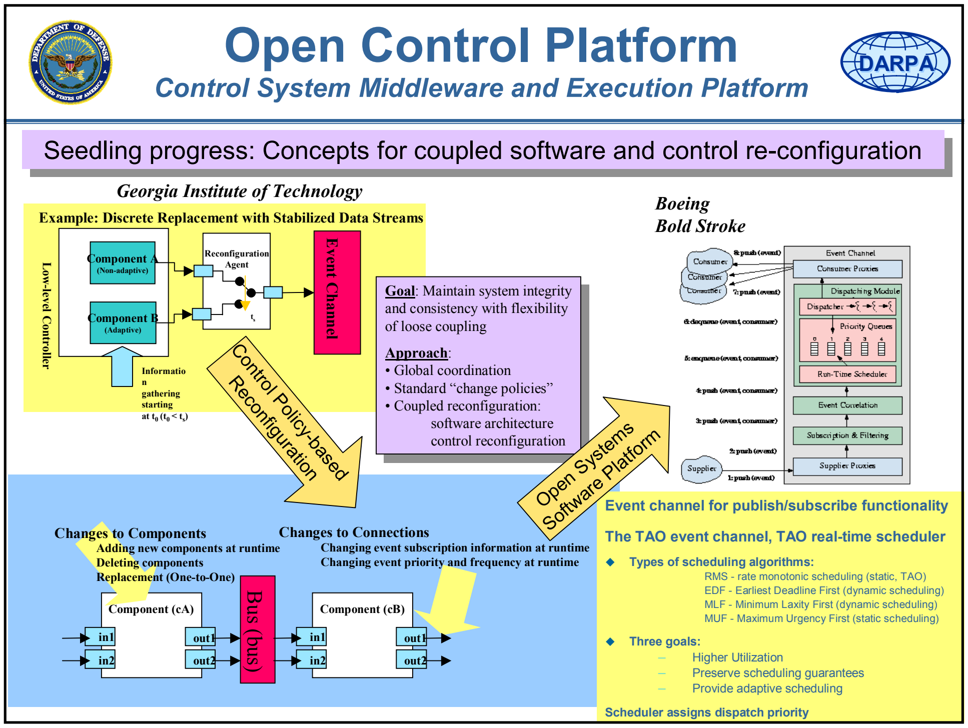
37

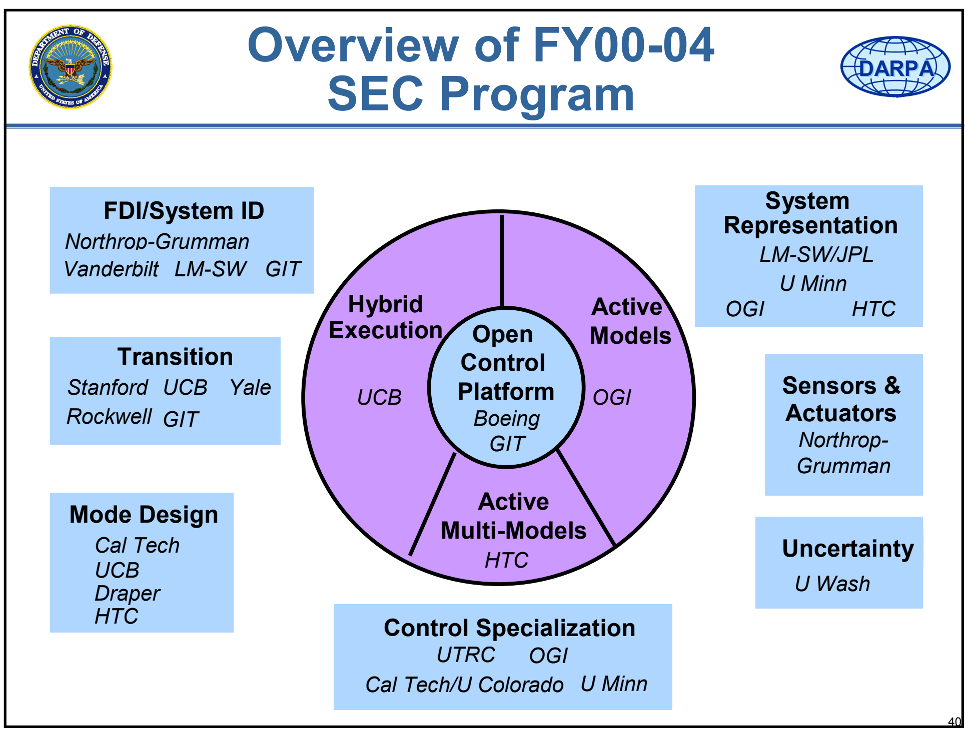
Open Control Platform

Middleware for open, software-enabled control

- Control elements of the solution:
  - Open control architectures
- Software elements of the solution:
  - System software support
    - Dynamic control configuration & scheduling
    - Distributed control coordination services
    - Real-time data and process services
    - Porting services
    - QoS resource management
  - Middleware framework for hybrid & adaptive control
  - SEC open source control platform implementation
  - SEC technology integration, demonstration, transition

38


Open Control Platform
Control System Middleware and Execution Platform
DARPA
Seedling progress: Concepts for coupled software and control re-configuration
Georgia Institute of Technology
Example: Discrete Replacement with Stabilized Data Streams
Low-level Controller
Component A (Non-adaptive)
Component B (Adaptive)
Reconfiguration Agent
Event Channel
Information gathering starting at t_0 ($t_0 < t_s$)
Control Policy-based Reconfiguration
Goal: Maintain system integrity and consistency with flexibility of loose coupling
Approach:
• Global coordination
• Standard "change policies"
• Coupled reconfiguration: software architecture control reconfiguration
Boeing
Bold Stroke
Event Channel
Consumer Proxies
Dispatching Module
Priority Queues
Run-Time Scheduler
Event Correlation
Subscription & Filtering
Supplier Proxies
Open Systems Software Platform
Changes to Components
Adding new components at runtime
Deleting components
Replacement (One-to-One)
Changes to Connections
Changing event subscription information at runtime
Changing event priority and frequency at runtime
Component (cA)
Bus (bus)
Component (cB)
in1
in2
out1
out2
Event channel for publish/subscribe functionality
The TAO event channel, TAO real-time scheduler
Types of scheduling algorithms:
RMS - rate monotonic scheduling (static, TAO)
EDF - Earliest Deadline First (dynamic scheduling)
MLF - Minimum Laxity First (dynamic scheduling)
MUF - Maximum Urgency First (static scheduling)
Three goals:
Higher Utilization
Preserve scheduling guarantees
Provide adaptive scheduling
Scheduler assigns dispatch priority


Overview of FY00-04
SEC Program
DARPA
FDI/System ID
Northrop-Grumman
Vanderbilt LM-SW GIT
Transition
Stanford UCB Yale
Rockwell GIT
Mode Design
Cal Tech
UCB
Draper
HTC
Hybrid Execution
UCB
Active Models
OGI
Open Control Platform
Boeing
GIT
Active Multi-Models
HTC
System Representation
LM-SW/JPL
U Minn
OGI HTC
Sensors & Actuators
Northrop-Grumman
Uncertainty
U Wash
Control Specialization
UTRC OGI
Cal Tech/U Colorado U Minn
40

Relevant DARPA Efforts: *Software and Physics*

Technology Recommendations:

1. System/software co-design, co-simulation and analysis environments
2. Optimization methods and tools for
   Very-large Scale,
   Densely-Connected,
   Multiple-Attribute,
   Continuous & Discrete Problems
3. Frameworks and middleware to provide higher level programming abstractions
4. Programming technology
5. Model-based code generation

SEC Mapping & Emphasis:

1. Mid: hybrid system design environment (extending)
2. High: hybrid discrete and continuous; composition and analysis; modeled uncertainty, system coupling
3. High: Open Control Platform middleware encapsulates: control switching and scheduling, data & prediction, on-line specialization
4. Low: apply functional reactive programming languages for control switching
5. Low-Mid: map hybrid system computational models to open systems platform

41

Relevant DARPA Efforts: *Embracing Change*

Technology Recommendations:

1. Adaptive Components for Embedded Systems
   - embedded, active models, constraints, constraint management
   - adaptive,self-monitoring, embedded software
2. Adaptable composition frameworks and QoS middleware
3. Methods to achieve flexibility
   - aspect oriented programming
   - domain specific languages
   - dynamic languages
4. Methods to control flexibility
   - parametric design
   - constraint languages

SEC Mapping & Emphasis:

1. High: Active state models, on-line control redesign, constraint management
2. Mid: Layered above real-time QoS middleware; open control platform middleware
3. Low: Functional reactive programming, control-specific reconfiguration
4. Mid: Control-specific solutions for resource & authority management services

42

Relevant DARPA Efforts: *Networked Embedded Systems*

Technology Recommendations:

1. Develop methods for predicting global properties from local component descriptions without assuming static structure.
2. Develop monitoring, controlling, diagnosis techniques for large, variable structure, hybrid systems.
3. Develop technology to replace, enhance or extend material, mechanical and biological properties via computation.
4. Explore biologically inspired distributed control and adaptation methods for networked embedded systems.

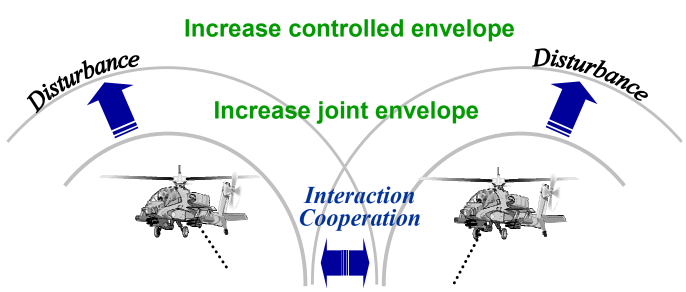
SEC Mapping & Emphasis:

1. Mid: Active State Models, hybrid prediction for reconfigurable, distributed hybrid systems
2. High: SEC primary focus
3. TBD: Apparent potential for applying hybrid systems
4. None: Not precluded

43

Experiment *Cooperative airlift*

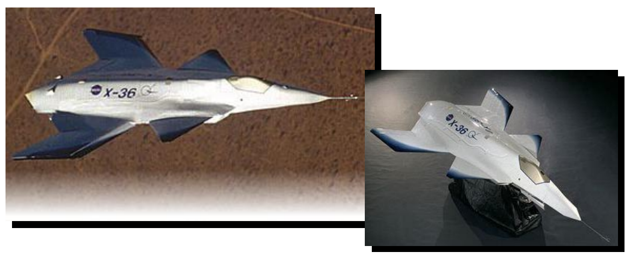
- **Baseline: Improved stability maintenance, performance in operation of single vehicle (e.g., university lab VTOL Rotorcraft)**
- **Challenge: Cooperative operation of systems with coupled dynamics (e.g. joint airlift, simple lift mode coordination)**
- **Goal: Increase the envelope of operation for cooperative (including physically coupled) system control**



44

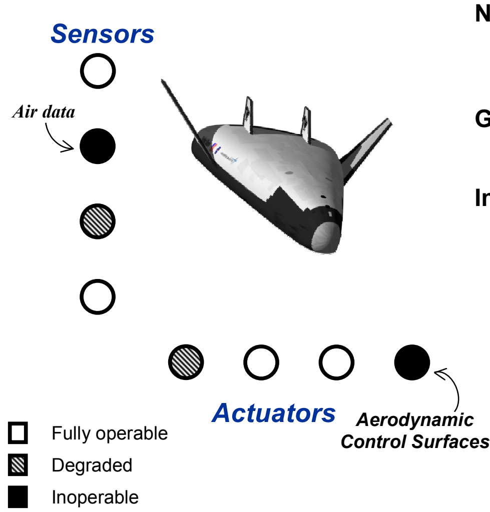
Demonstration
X-36

- **Baseline: better regulation of flight, better takeoff and landing control, for remotely-piloted, unstable vehicles**
- **Challenge: Automate highly-reactive, extreme maneuver & evasion regimes, coordinated for adversarial encounters**
- **Goal: Enable high- and extreme-performance autonomous tactical maneuvers in order to enhance survivability, capability of autonomous weapons systems**



45

Example: Military Space Vehicle



Now
- Re-entry problems (similar to shuttle)
- Lacks control in transition region

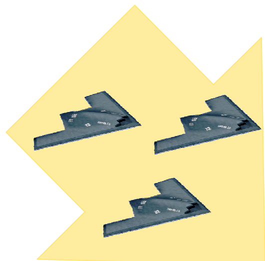
Goal:
- Permit maneuvers in upper atmosphere

Impact:
- Protect vehicle upon re-entry into threat environment
- State model supports prediction during dead reckoning

SEC On-Line Control Customization will provide:
- ***Flexible use of available sensors and control authority***
- ***Time-triggered & predictive mode transitions***

46

Example: UAV Swarms



Now

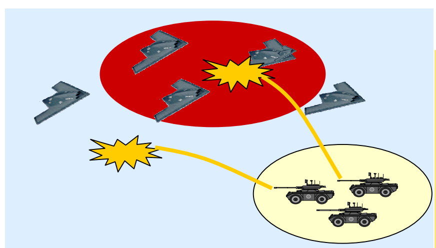
- One remote pilot per vehicle
- Static flight group configurations

Goal:

- Dynamic swarm structure
- Joint response to damage, failure

Impact:

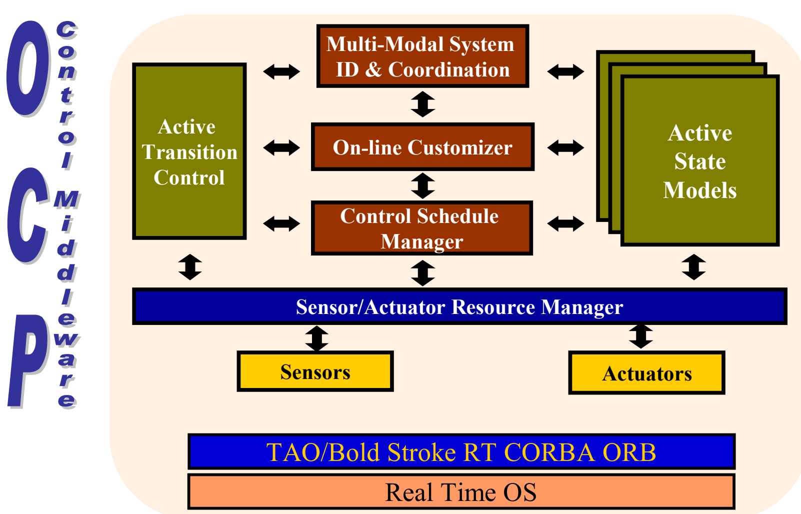
- Configurable mission systems
- Reduced multi-system operational effort
- Survivable UAV swarms



SEC On-Line Control Customization provides:

- ***Dynamically reconfigurable cooperative groups***
- ***Joint FDIR for coordinated reconfiguration after failure and damage***

Task: Open Control Platform



Open Control Platform

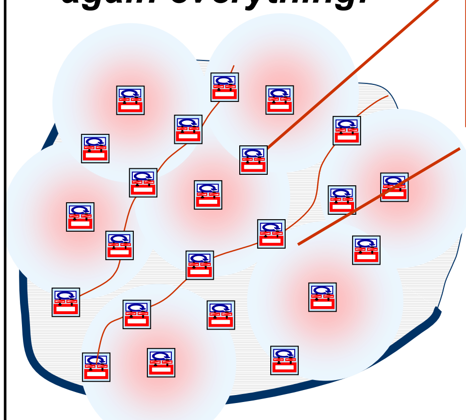
Challenges

- Provide control middleware and tool support for building commodity controllers.
- Provide parametric, structural framework necessary to support SEC active state model, hybrid/coordinated, and adaptive multi-modal control technologies.
- Provide flexible experimental platform for SEC control research and demonstration.

49

Theme 3: Dealing With Dynamic Structures

Networked embedded systems will change again everything:



LARGE number of tightly integrated, spatially and temporally distributed physical/information system components with reconfigurable interconnection.

Hard Problems:

- Design culture changes: from analysis and synthesis of static structures to dynamic structures
- Notion of correctness changes: from "getting it right" to "keeping it right
- Richness of interactions changes: from stable data oriented to dynamic hybrid interactions

Essentially a **dynamic aggregation** problem

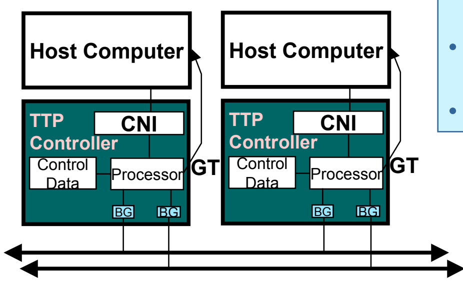
50

A European Approach to Networked Embedded Systems:

Time-Triggered Architecture

Designed for:
- Automotive, drive-by-wire apps.
- Distributed architecture
- Safety critical, hard real-time

- Time division multiple access protocol
- Synchronous
- Fault Tolerant Global Time
- Distributed redundancy management
- Composability for jitter, faults, reliability
- Low-cost implementation



TTP Bus

Limitations:
- **Static Scheduling**
- **Overhead cost**
- **No support for event-driven op.**

Same concept as Safebus (in 777 avionics) - but the money is in cars -

51

Metamorphosis of a Machine

Capabilities:

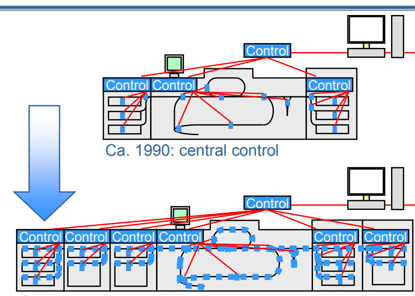
Embedding:
- MEMS-based sensing, actuation: high degree of control over physics, adaptation
- redundancy: self-diagnosis, repair
- mass production: economies of scale

~ 100,000 actuators and sensors

Distribution:
- modularity: high degree of flexibility, "macro" adaptation
- redundancy: "fail-safe" robustness, graceful degradation, resource optimization

1,000 computational nodes

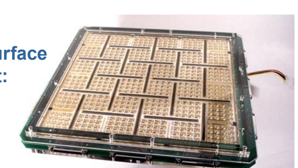
Coordination:
- plug-and-play: 100s of configurations and 1000s functions
- reconfigurability: rapid deployment, mass customization
- upgradeability: low maintenance, extended life



Ca. 1990: central control

Ca. 2010: distributed, embedded control
600 60 to pages per minute, 1-10KHz loops 1-10 Hz reconfig.

Active surface transport:

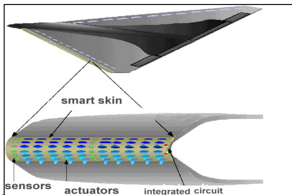


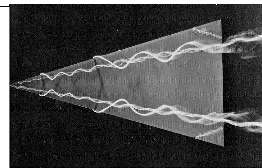
500 actuators, 30,000 sensor elements

Reconfigurability, redundancy, and local control promise improved time-to-market, robustness, and in-field customization.

52

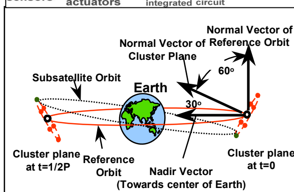
Networked Embedded Systems:
Examples

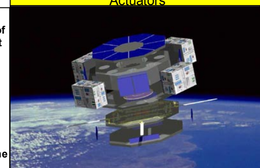




Vortex Control Using Distributed MEMS Actuators

MEMS Actuators for Vortex Control (UCLA, CalTech)
- Number of nodes: 10^4
- Loop frequency: 1 KHz
- Coordination frequency: 10Hz
- Geometric size: 30m

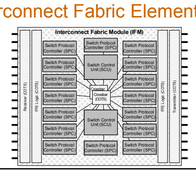


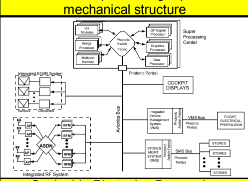


Information processing replaces mechanical structure

Pico Satellite Constellations (Aerospace Corporation)
- Number of nodes: 10^2 - 10^3
- Loop frequency: 1-2 KHz
- Coordination frequency: 1Hz
- Geometric size: 1-1000km

Key Integrating Element:
Interconnect Fabric Element (IFM)



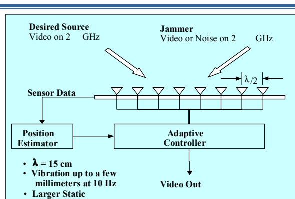


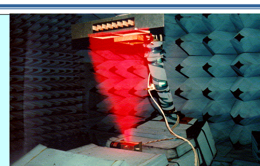
Scaleable Photonics Processing Unified Network

Unified Processing Network For Avionics
- Processing nodes: > 3*10^3
- Link Bandwidth: 2GHz
- Aggregate Bandwidth: >10^12
- Industry Standard Protocol

53

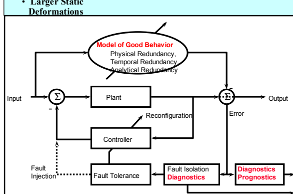
Networked Embedded Systems:
Examples

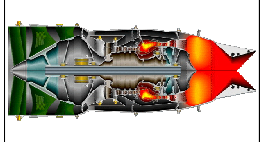




Sensors enable electronic deformation compensation

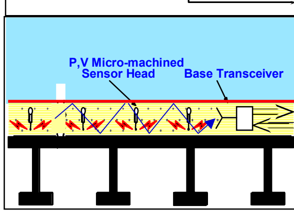
On-orbit Space-Based Phased Array Antennas (Boeing)
- Number of nodes: 10^3
- Loop frequency: 1-5Hz
- Coordination frequency: 1 Hz
- Geometric size: 6m x 22m

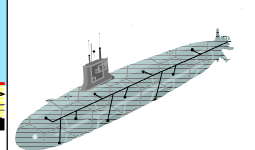




Smart Engine: Interactive, Aware and Reconfigurable

Smart Engine (UTRC)
- Number of nodes: 10^2
- Loop frequency: 2-5KHz
- Coordination frequency: 50-100Hz
- Geometric size: 5m





Information processing replaces mechanical structure

Noiseless Sonar (NRL)
- Number of nodes: 3*10^4
- Loop frequency: 30KHz
- Coordination frequency: 1Hz
- Geometric size: 300’

54

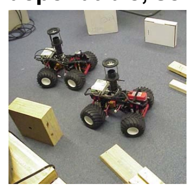
The Platforms are Coming!

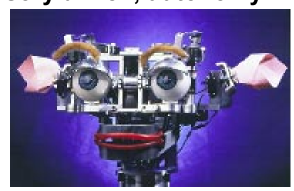
56

Mobile Autonomous Robot Software (MARS)

OBJECTIVE:

Develop the missing, (learning-based) Software Technologies **required for dependable, sensory driven, autonomy**





GOALS:

Synthesize the desirable capabilities of both **deliberative (symbol-mediated)** and **reactive (sensor mediated)** control by combining explicit, embedded programming technologies with machine learning strategies to enable mobile autonomous robots to have the capacity to function in **realistic, unpredictable environments**.

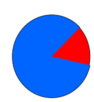
Develop domain specific, embedded software composition methodologies to enhance our capacity to create complex **embedded software for autonomous systems**.

57

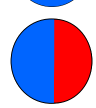
Key Program Idea

Source of Competency

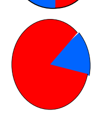
Soft Computing



Robot Shaping



Imitative Learning



■ Pre-programmed (innate)

■ Learning-based (acquired)

58

Soft Computing (USC)

- Approach:
  - Incorporate a fuzzy-neural capability whereby a fuzzy controller ensures the robot meets some predefined performance criteria, while the neural network produces additional tuning data until that criteria is met
  - Standard library of atomic "basis behaviors" containing both numeric and symbolic state information and dynamically linked into behavior networks
- Hypothesis:
  - Behavior Nets can be automatically generated through learning mechanisms
  - Online learning based on Reinforcement Learning in which the perception system is continually comparing observed behaviors to the internal representations

Robot Shaping (GaTech)

- **Approach:**
  - Reasoning in Finite State Automata (FSA) for plan generation through the use of "wizards" to guide high-level deliberative planning
  - Probabilistic situation recognition and indexing into behavior sets for opportunistic planning and reaction
  - Case-Based Reasoning: situation-dependent behavioral gains and assemblage switching at the reactive level of execution
- **Hypothesis:**
  - Q-learning methods can enable behavior assemblage selection at a level above gain adjustment
  - Specialized reinforcement learning methods ("learning momentum") can be used to adjust behavior gains at run time

The New Embedded Software Problem

DARPA

- **Work in Themes 1 & 2 change the way we construct embedded software**
  - Address current embedded software problem
- **Networking embedded systems changes the playing field again**
- **Old, closed system approach is incapable of addressing this problem**
- **Radical new approaches/technologies are needed for this new problem space**

60

Technology Gaps

1. Methods for predicting global properties from local component descriptions without assuming static structure.
2. Monitoring, controlling and diagnosing of large, variable structure, hybrid systems.
3. Dynamic, adaptable composition frameworks and middleware
4. Controlling of physical, chemical and biological properties via embedded information systems.

61

Conclusion

- The Embedded Software Initiative is DARPA's response to the exploding integration role of information technology across military platforms.
- Existing and planned programs establish a new re-integration of physical and information sciences. This will make a huge difference in our ability to:
  - design software for achieving physical behavior,
  - make software the media absorbing change in physical systems,
  - build, integrate physical systems dynamically from spatially and temporally distributed components.
- To do this means changing culture. DARPA's focused investment is critical to catalyze and accelerate this process.

62