Embedded Software: Opportunities and Challenges

Dr. Janos Sztipanovits, DARPA/ITO
The Technology Challenge

Embeded 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
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
DoD Example: Avionics Systems

1958 1950’s - 60’s 1970’s - 80’s 1990’s - 00’s 2000

J. Sztipanovits

Source: AFRL
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.
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.

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

- Model checking
- Compositional synthesis
- Simulation

- Formal modeling
- Verification tools
- Automated code synthesis
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
We have focused on functional composition...

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

But cross-cutting physical constraints weaken or destroy composability.
Current Technology: Functional Composition

Functional composition does not address physical constraints

- Infrastructure Services
- Operating System
- Board Support Package
- Hardware CPU, MEM, I/O

Reuseable Component Library

- Essential Common Components
- Desired Pluggable Common Components

Application Software (currently integrated manually)

- Essential Project Specific Plug-ins
- Project Specific Components - potentially including legacy

Custom Project Library

J. Sztpanovits
Goal: Design Automation Tools for Embedded Systems

◆ Compose model-based design frameworks:
  - Use existing CAD, EDA, CASE and systems Engineering frameworks as seeds
  - Add customizable design views and notations
  - Provide multi-resolution simulation
  - Add automated analysis and system/software synthesis

◆ Capabilities:
  - Co-evolve integrated physical and information system MODELS
  - Synthesize/customize software and system components directly from models
  - Establish composability for physical behavior
ITO: Program Composition for Embedded Systems (PCES)

Aspect languages will change programming:

- Infrastructure Services
- Operating System
- Board Support Package
- Hardware (CPU, MEM, I/O)

Core Code (e.g., DSP)
- "Clean" Core Code

Aspect Code
- synch.
- fault t.
- secur.

Reusable Aspect Code

ANALYZER

WEAVER
Model-based integration will change system design and integration:
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
Goal: Adaptive Component Technology for Embedded SW

- Builds on object component technology (CORBA, COM) but provides:
  - Internal mechanisms to respond to changes
  - Physically and computationally “self-aware” components

- Capabilities:
  - Insulates software from hardware with small performance penalty
  - Increases tolerance to unexpected changes
  - Optimizes performance
  - Increases tolerance to faults
Theme 3: Dealing With Dynamic Structures

A new category of systems:

Embedding +
Distribution +
Coordination

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

Why should we work on this?
The wave is coming:
• Tremendous progress in MEMS, photonics, communication technology. **We need to build systems now from these.**
• Identified applications with **very high ROI: strong application pull**
• Almost total lack of design theory technology: The problem is extremely hard.
Model: Locally and globally relevant information for global coordination

Reasoner/Adapter: Adaptation of local structure and parameters, coordination

Controller: Discrete or hybrid control of local physics

Distribution:
- Heterogeneous, simple components ($10^2$-$10^5$)
- Changing interconnection topology
- **Embedded synthesis** for dynamic distribution, reconfiguration

Coordination:
- Global **coordination** of local interactions
- Consistency of globally relevant information
- Requirements are determined by locality of physics
Applications determine the type of services required
- Physical characteristics of the system determine dynamics, accuracy and required fault behavior of services
- Services are built in layers with rich interdependence
- Algorithms used in components depend on the distributed computation model

Hard Problems: Hybrid self-stabilization, customizable design, predictable dynamics, time bounded synthesis, automated composition.
**Approach:** Transition-aware, sub critical problem solver

**Challenge:** Problem statistics, order parameter.

1. Dynamically adjust the problem to keep it “left of the phase transition”.
3. Constraint solver rapidly solves sub-critical problem instances.
What Are We Doing?

**Software and Physics**
- System/Software co-[design,simulation, analysis]-
- New methods for system/code composition--------
- Frameworks and middleware for embedded SW---
- Hybrid optimization,analysis-----------------------

**Embracing Change**
- Adaptive components for embedded systems------
- Methods for controlling flexibility----------------
- Adaptable frameworks and QoS middleware-------
- Programming methods to achieve flexibility------

**Networked Embedded Systems**
- Predicting global properties from local component descriptions without assuming static structures---
- Monitoring, controlling and diagnosing of variable structure hybrid systems------------------------
- Dynamic composition frameworks and middleware for networked embedded systems------------------
- Controlling physical, chemical and biological properties via embedded information processing---

**Relevant existing programs:**
(MoBIES, PCES, SEC)
- Coordinate efforts
- Leverage to increase common technology base
- Primary impact on Themes 1-2

**New-start program:**
- Networked Embedded Systems Technology (NEST)

**Planned program:**
- Adaptive and Reflective Middleware Systems
Conclusion

◆ Embedded Software is an important area for DARPA due 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 able to absorb change in physical systems,
  – Build, integrate physical systems dynamically from spacially and temporarily distributed components.

◆ To do this means changing culture. DARPA’s focused investment is critical to catalyze and accelerate this process.