Modeling Languages and Methods for Distributed Embedded Systems

Laurent Michel    Alex Shvartsman
CSE Department
ldm@engr.uconn.edu  aas@cse.uconn.edu

Workshop on Research and Curriculum Development Opportunities 2013

October 1st, 2013
Student Union Rooms 304 b/c, UConn Storrs Campus, Storrs, CT 06269
Background

• **Tempo Project**
  – Toolkit for Timed Input/Output Automata Formalism

• **Partners**
  – Nancy Lynch (MIT/VeroModo)
  – Alex Shvartsman (UCONN/VeroModo)
  – Laurent Michel (UCONN)
  – Scott Smolka (Stony Brook)
  – Nancy Griffeth (Lehman)
  – Myla Archer (NRL)

• **Past Funding**
  – Air Force STTR (Phases 1 & 2)
Hard Reality

• Distributed Systems are increasingly complex
• Requirements
  – Safety (correctness)
  – Performance
  – Fault tolerance
  – Dynamic
  – Reconfiguration
• Huge gap between
  – Modeling
  – Implementation
Current Practice

• Ad-hoc modeling
• Limited use of formal methods at design stage
• Limited ability to prove key properties
  – Liveness
  – Safety
  – ...
• No continuity through life-cycle
  – Formal tools not used all the way to development
Bottom Line

• Design bugs discovered too late in the process
• Design fails to consider subtle issues
  – E.g., no modeling for continuous state changes
• Correct designs get mangled during
  – Translation
  – Implementation
Modeling with TIOA

- Timed Input / Output Automata
- What it is
  - Modeling Formalism
  - Rooted in automata theory
  - Captures discrete & continuous transitions
  - Captures interactions with its environment
  - Supports composition to deal with complex systems
  - Supports rigorous arguments about system properties
TIOA in a Nutshell

• **Fundamentally**
  – Model system as composition of a collection of interacting state machines.

• **Communication** with peers *precisely* stated

• **Prove** properties of model
  – Correctness, liveness, deadlock free,…
  – Methods: invariants, trace inclusion (“simulation”)

• **Simulate** specification

• **Generate** code from verified specification
Tempo

- Tempo is a *computer-aided design framework*
  - IDE support for complex distributed systems
  - Based on Timed I/O Automata

- Designed to support
  - Modeling and specification
  - Verification and model checking *(UPPAAL)*
  - Theorem Proving *(PVS)*
  - Simulation
  - Code generation
  - Optimal Deployment
Showcase Examples

• **DHCP Failover**
  – *Specified & verified* the protocol
  – *(Found bugs in it!)*

• **RAMBO**
  – Specified protocol
  – Solve the deployment & reconfiguration problem

• **ESDS** *(Eventually Serializable Data Services)*
  – Specified protocol
  – Solved the deployment problem *(NP-hard)*
Brief Example

automaton Thermostat(low, high, initialTemp, ambientTemp, coolingRate, heatingRate:Real)

signature
  output turnOn, turnOff

states
  isOn: Bool := initialTemp < high;
  temp: Real := initialTemp;

transitions
  output turnOn
    pre temp ≤ low ∧ ¬isOn;
    eff isOn := true;
  output turnOff
    pre temp ≥ high ∧ isOn;
    eff isOn := false;

trajectories
  trajdef heaterOff
    invariant ¬isOn;
    stop when temp = low
    evolve d(temp) = coolingRate * (ambientTemp-temp);
  trajdef heaterOn
    invariant isOn;
    stop when temp = high
    evolve d(temp) = heatingRate;
Road Ahead

• Outstanding task
  – Expand code generation from specifications
  – Look into optimization problems
    • Derived from specification
    • Related to deployment
  – Look into alternatives to UPPAAL / PVS
Potential Impact

• “Cradle to Grave” solution for Distributed Systems
  – Detect design flaws early
  – Address operational deployment issues
  – Automate code synthesis for components

• Bottom line

  Cohesive approach to complex and error prone design and implementation tasks
Many Potential Applications

- Time to Completion & Cost depend on system
  - Avionics system?
    - TCAS variant was specified and reasoned about using the IOA methodology
  - Multi-shaft elevator control?
  - Toaster-oven?
Research Team

• L. Michel (CSE)
• A. Shvartsman (CSE)