Model-Based Analysis of Embedded Systems

Rance Cleaveland, Dept. Comp. Sci. and ISR

ISR @25 — The Institute for Systems Research — A. James Clark School of Engineering

Collaborators and Support

ISR / UMD
Steve Marcus, James Ferlez, Peter Fontana, Sam Huang

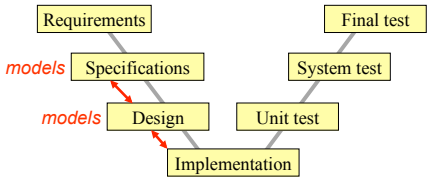
Fraunhofer
Chris Ackermann, Iris Morschhaeuser, Arnab Ray

Robert Bosch
Beth Latronico, Thomas Mandl, Chris Martin, Charles Shelton

Research support by NSF Grant CCF-0820072, NSF Expeditions in Computing Grant CCF-0926194 , and funding from Robert Bosch and Fraunhofer USA

Some Software Companies

Model-Based Development



Models formalize specifications, design
Models support V&V, testing, code generation
Models facilitate communication among teams

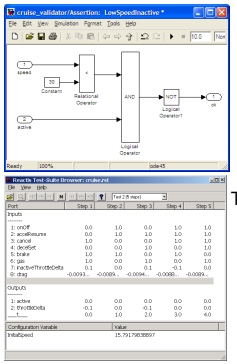
Validating Simulink®/Stateflow® Models

Problem Motivation
- Simullink / Stateflow de facto modeling standard
- Early requirement checking yields more efficient design / development
- Manual checking error-prone, costly

Approach: Instrumentation-Based Verification
- Formalize requirements as monitor models
- Instrument design model with models
- Use automated coverage-testing to check for violations

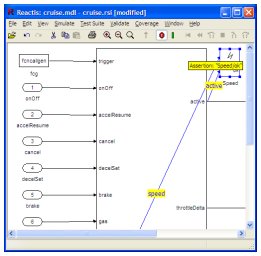
Pilot Study: Automotive Electronicsl
- Source of recalls
- Available artifiacts: production code, requirements document
- Tasks: formulate monitor models, perform testing

Instrumentation-Based Verification



Monitor model: "If speed < 30 cruise must be inactive"

Tests



Instrumented design model

Results / Conclusions

Monitor models formalize requirements "efficiently"
Reference architecture was a big factor

Formalization helps, even without formal verification
Testing-based checking uncovered inconsistencies
Tests also useful for diagnosing problems

Modeling "upstream" pays off "downstream"
Design modeling easier after requirements modeling

Requirements are not always what is required
Requirements documents are often "just another description"

Requirements Reconstruction

Problem Motivation
- System comprehension / understanding
- Inaccurate / incomplete / implicit requirements specs

Approach: Invariants via Data Mining
- Generate coverage tests from system model / code
- Use data mining to find invariants between inputs, outputs
- Double-check proposed invariants using IBV

Pilot Study Results
- Automotive electronics application
- 95% of requirements reconstructed
- Two implicit (missing) requirements discovered
- 98% of requirements were valid

Model Checking for Real-Time

Problem Motivation
- Early design verification desirable for real-time systems
- Model-checking gives definitive answers, but is slow

Approach: Predicate Equation Systems
- Convert model-checking problems for real-time into queries in recursive predicate logic (PES)
- Use efficient solvers for PES to answer model-checking questions

Pilot Study Results
- Test bed of >40 real-time specs used
- Performance was much better in most cases than existing tools

CMACS

Project Motivation for "Computational Modeling of Complex Systems"
- Computational modeling plays increased role in science, engineering
- Techniques (model checking ,abstract interpretation) from system verification can be used to automate model analysis

Approaches
- Multi-university project involving four lead universities (incl. UMD/ISR)
- Computational biology, control-system challenge problems
- Focus: hybrid, stochastic system modeling and analysis

Results
- Five-year, $10m project started Nov. 2009
- ISR CS, ECE faculty, students exploring stochastic modeling and composition
- CS, Public Health team members working on model construction