SnT
securityandtrust.lu

Automated System Testing of Real-Time Embedded Systems Based on Environment Models

Lionel Briand

Software V&V Laboratory

uni.lu
UNIVERSITÉ DU LUXEMBOURG

Acknowledgements

- Work done at Simula Research Laboratory
- Thesis work of Zohaib Iqbal
- Shaukat Ali
- Andrea Arcuri

securityandtrust.lu

New affiliation

- Interdisciplinary centre for ICT security, reliability, and trust
- National priority
- Young university (2003) and new interdisciplinary center (2009), now the size of Simula
- International university and centre at all levels: students, research scientists, faculty, management
- Three official languages: English, French, German

UNIVERSITÉ DU LUXEMBOURG

Facts about ES

- By 2020: 40 billion embedded devices
- Annual budget for embedded systems: 160 billion Euros with annual growth of 9%
  - Average size: 1 million LOC
  - A premium class automobile: 100 million LOC
  - Boeing 787 Dreamliner: 6.5 million LOC to operate on-board support systems and avionics

Testing Considerations

- Independent system testing
  - Limited knowledge of system design
  - Thorough knowledge of application domain
- Environment simulation
  - Testing on development platform
  - Testing early

Types of Simulation

- 1) Model and simulate the SUT, its hardware and its environment
- **2)** SUT is tested on development platform with simulated environment and adapter/simulator for hardware platform
- 3) SUT deployed on actual hardware and tested with simulated environment
- Model-in-the-loop, hardware-in-the-loop, processor-in-the-loop, and **software-in-the-loop**

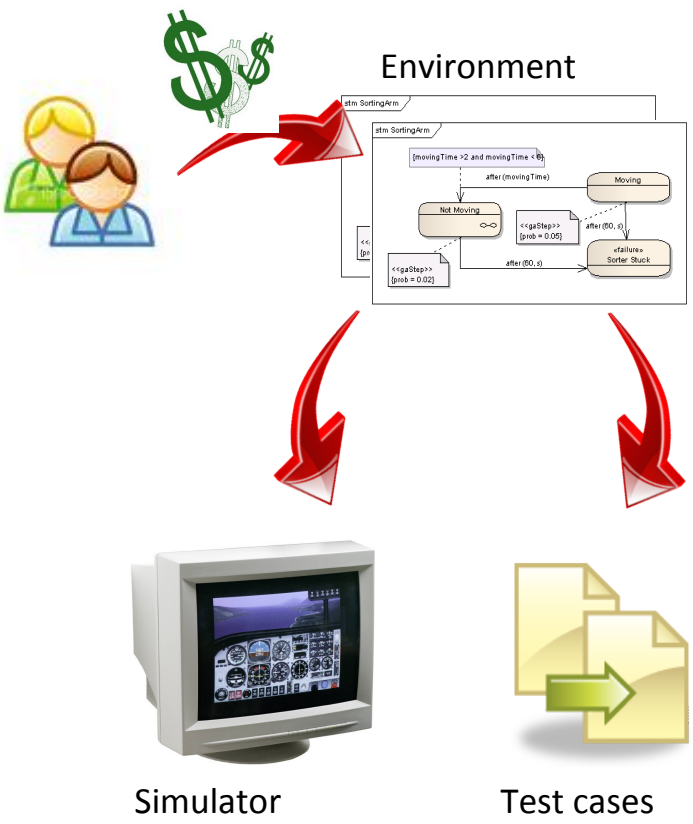
Testing Strategies

- Coverage of nominal behavior
- Robustness testing, e.g., hardware failures
- Stress testing: Target unsafe or critical states

Partners and Case Studies

- Recycling machines, marine seismic acquisition system
- Soft real time requirements in the order of seconds
- Complex environment, many communicating components

Our Approach



- Black-box (BB) testing
- Model-based testing
- Same model for generation of environment simulator and test cases

SnT
securityandtrust.lu

Decomposing the Problem

- Modeling notation and methodology
- Code generation for simulator
- Test strategy
- Test harness generation

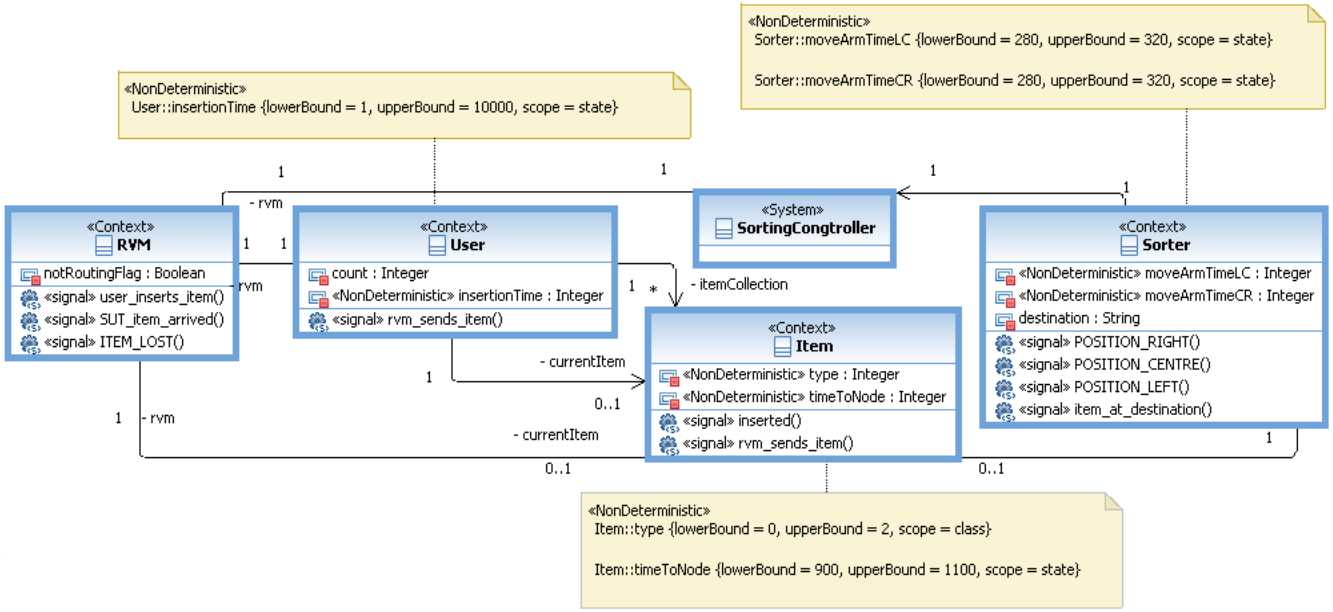
uni.lu
UNIVERSITÉ DU LUXEMBOURG

Modeling Notation and Methodology

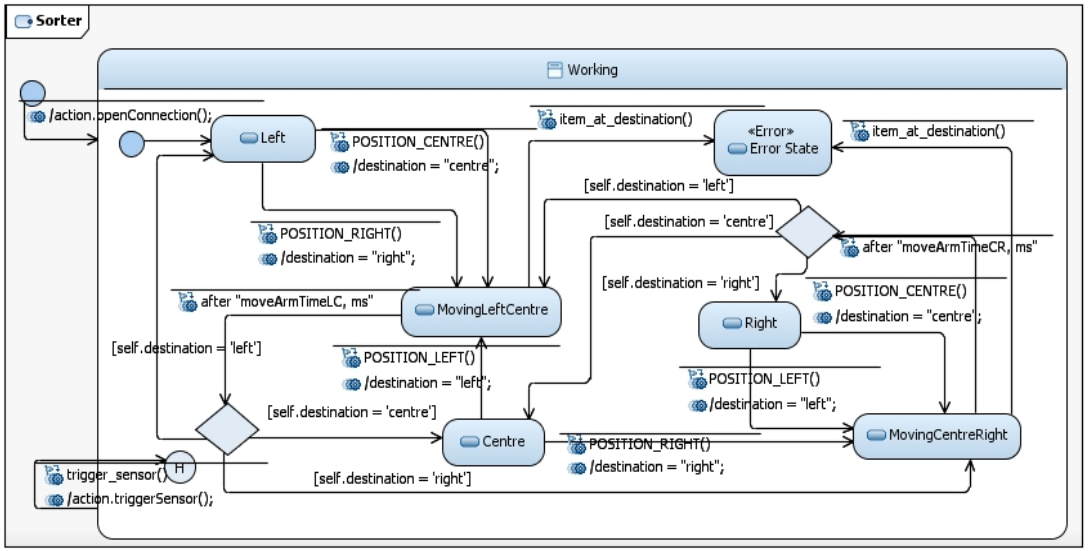
- Environment modeling, not system modeling, focused on BB test automation
- Use of standard notations, preferably software standards
- Tailor system modeling methodologies to environmental modeling needs
- Adopted Notation: UML + MARTE, and a profile

Domain Model

- Class diagram, Specific guidelines for environment modeling
- Hardware: Sensors, Actuators, …; Other (sub)systems; Users; Abstract concepts: temperature …
- Associations: Communicating components, physical connections, …
- Properties: affect the SUT, controlled by simulation, constrain behavior, used in state invariants



Behavioral Modeling

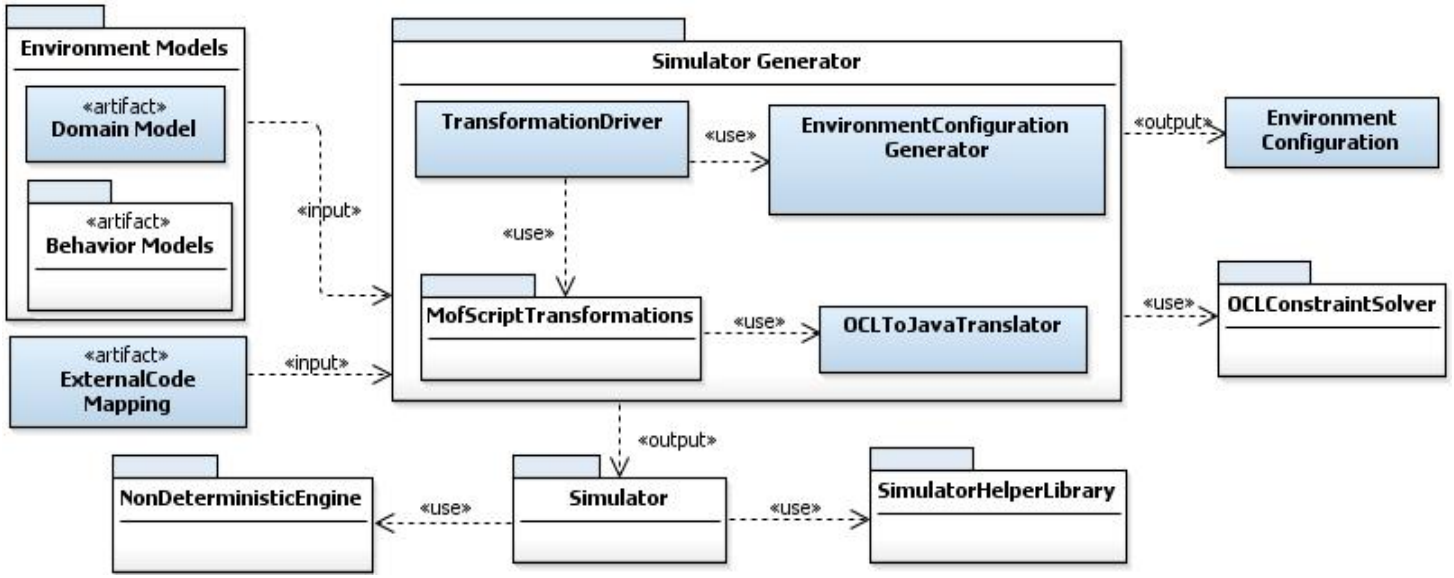


Sorting Arm

- For all env. components, including users
- Communicating UML state machines, OCL
- Profile: Error and Failure state stereotypes
- Error: Oracle
- Level of abstraction: Only include refinements that impact the SUT
- MARTE: Non-determinism (probabilities, timed events)

Simulator Code Generation

- Developed a simulation framework
- Resolved UML variation points, e.g., object concurrency model, time semantics, execution semantics, and order of events
- Extended version of the state design pattern to address the requirements for simulation and testing: Concurrency, time events, change events, effects.
- Java as action and target language, since time requirements in seconds. Other target languages and technologies can be used.



Test Harness

- Test goal: Reach Error states
- Environment models and simulator used to generate test cases and oracles
- Test case is a setting of the environment simulator
  - Relations among instances of environment components (Environment configuration)
  - Input values for non-deterministic transitions (Simulation configuration)

Test Strategies

- System testing: Thousands of LOCS executed, seconds/minutes, multi-threading
- Random testing: Baseline
- Adaptive Random Testing
  - Distance function, Generate diverse test suite
- Search-based testing to converge towards Error states

Search-based Testing

- Search problem: Find simulation configurations reaching Error states in environment models
- Search techniques: GA, (1+1) EA
- Approach level (A), branch-distance (B) in state machines
  - Approach level (to get close to error state)
  - Branch distance (to solve OCL constraints on guards)
- Extension to handle time-based transition triggers (T)
- Original formula for fitness f:
  - $f(m) = min_e ((A_e(m) + nor(T_e(m)) + nor(B_e(m)))$

Case Studies (I)

- Research questions:
  - RQ1: What is the effect of test case representation on fault detection effectiveness of the testing strategies?
  - RQ2: Which testing strategy is best in terms of failure detection amongst RT, ART, GA, and (1+1)EA?
  - RQ3: Is environment model-based system testing an effective approach in detecting faults for industrial RTES?
- Evaluation criteria: success rate (Fisher exact test, odds ratio), # test cases executed when no difference (Mann-Whitney U-test, Vargha-Delaney A statistic)
- Industrial RTES + artificial case studies derived from actual systems to obtain various properties
- At most two error states, hand seeded non-trivial fault (multithreaded, network features)
- Artificial RTES: 96 experiments, sampled at most 1000 test cases for each algo, 10 seconds per test case, run each algo 100 times, 653 days of execution on a single computer -> cluster
- Industrial RTES: run simulator for 60 seconds, 1000 fitness evaluations can take 16 hours, single dedicated computer, 55 days

Results

- ART best, but search techniques worse than RT!
- Industrial case study: ART yielded a success rate of 100%
- But on some artificial problems, no approach was successful (seeded fault hard to detect)
- Plausible explanation for low performance of search:
  - Most difficult: The transition to the error state, whose trigger and guard usually depend on the entire state of the environment and SUT
  - Little diversity was generated during search
  - Favored time distance over branch distance
  - Branch distance only computed if time distance was zero
  - Made it very difficult to solve branch distance
- Found two critical faults in the (already tested) production code of the industrial case study

Case Study (II)

- Improved the fitness function for search techniques
- Compute branch distance even when time distance not zero
- Favor cases where branch distance is shorter over cases where time distance is shorter (used order function ITD, not fitness)
- Exploit more properties of the environment models: time (TIR) and transitions to "risky" states, model coverage
- Overall, (1+1) EA with TIR and ITD proved to be the best algorithm
- But search-based algorithms perform significantly worse than RT for the artificial problems where the approach to risky states was trivial.
- Recommendation: Start by applying RT and evaluating whether the risky states are easy to reach. If this is the case, then RT is most likely trigger the transition to the error state. In case the approach is not trivial, then one should use (1+1)EA-ITD-TIR

SnT
securityandtrust.lu

Conclusions

- (1+1)EA once again better than GA
- Using more information from models and simulation (risky states) helped improve search
- But the same technique is not the best on all case studies
- Finding the right fitness function was, as usual, a very exploratory process
- Running such experiments on RTES presents practical challenges (scale, time)

uni.lu
UNIVERSITÉ DU LUXEMBOURG

SnT
securityandtrust.lu

Final Thoughts

- An example research project driven by precise, concrete industrial considerations
- Practical aspects weigh heavily on how one defines the problem and on the applicability of the solution
- Applicability is not an afterthought after the research is completed, not in engineering disciplines
- A complete solution is necessarily multidisciplinary: Modeling, search, testing
- Though a common industrial problem, limited material could be reused from research
- SE researchers often address problems that are, as defined, unlikely to be encountered in practice. Vast areas of practical importance are not addressed.

References

- Z. Iqbal et. al, Automated System Testing of Real-Time Embedded Systems Based on Environment Models, Simula Research Laboratory, Technical Report (2011-19) - submitted
- Z. Iqbal et. al, Code Generation from UML/MARTE/OCL Environment Models to Support Automated System Testing of Real-Time Embedded Software, Simula Research Laboratory, Technical Report (2011-04) - submitted
- S. Ali et. al, Solving OCL Constraints for Test Data Generation in Industrial Systems with Search Techniques, Simula Research Laboratory, Technical Report(2010-16) – submitted. Short version in QSIC 2011.
- A. Arcuri et. al, "Black-box System Testing of Real-Time Embedded Systems Using Random and Search-based Testing", In: IFIP International Conference on Testing Software and Systems (ICTSS), 2010.
- Z. Iqbal et. al, "Environment Modeling with UML/MARTE to Support Black-Box System Testing for Real-Time Embedded Systems: Methodology and Industrial Case Studies", In: ACM/IEEE International Conference on Model Driven Engineering Languages and Systems (MODELS), 2010.