Integration of Architecture Specification, Testing and Dependability Analysis

Swapna S. Gokhale
Joseph R. Horgan
Applied Research
Telcordia Technologies
Morristown, NJ 07960
{swapna,jrh}@research.telcordia.com

Kishor S. Trivedi.
Dept. of Electrical & Computer Engg.
Duke University
Durham, NC 27708
kst@ee.duke.edu
Outline

- Introduction and motivation
  - Discussion of the methodology
  - Demonstration of methodology using a case study
  - Conclusions and future research
Introduction and Motivation

• Software architecture is increasingly appreciated as a method of understanding and analysis as software systems continue to grow in size and complexity.

• Software architecture represents early design decisions:
  – Have a profound impact on the non-functional attributes of a system.
  – Difficult to change or reverse.

• Architecture analysis is one of the best vehicles to assess important quality attributes such as reliability, reusability, maintainability and performance.
Introduction and Motivation (contd..)

- Languages used to specify software architectures:
  - Focus on the high-level structure rather than the implementation details of a particular source module.
  - Play an important role in the development of software by composing source modules rather than individual statements.

- Development of tools to support understanding, testing, debugging, reengineering, and maintaining architecture specifications is gaining prominence.
Introduction and Motivation (contd..)

• Software architectures specified in architecture specification languages can be used for performance and dependability analysis:
  – Performance and dependability models can be constructed from such specifications to enable quantitative analysis.

• Lack of appropriate information to parameterize the quantitative models constructed from software specifications.

• Trace data generated during simulation/execution of architecture specifications can provide a rich source of information for model parameterization.
  – Collection and analysis of such trace data is likely to be supported by many tools.

• Similar approach has been demonstrated at the source code level.
Introduction and Motivation (contd..)

• Demonstrate a methodology to parameterize the performance and dependability models constructed from architecture specifications using trace data collected during simulation/execution of architectural specifications.

• Three-way integration between:
  – Architecture specification,
  – Specification simulation/execution and
  – Performance and dependability analysis.

• Methodology facilitated by Telcordia Software Visualization and Analysis Tool Suite (TSVAT) developed to support architectural specifications in SDL.
Outline

- Introduction and motivation

- Discussion of the methodology

- Demonstration of methodology using a case study

- Conclusions and future research
Methodology

1. SDL Specification
2. Simulation/Execution
3. Transformation to SRN Model
4. Model Parameterization
5. Reach. Graph Generation
6. Perf. & Depend. Analysis

Info. from other sources
Step I: System Specification in SDL

• Specification and Description Language (SDL) chosen as a Communicating Extended Finite State Machine (CEFSM) specification language.

• Choice of SDL motivated by the following reasons:
  – ITU standard, many telecom systems are specified in SDL.
  – Well-defined semantics.
  – Many commercial tools available to investigate architectural specifications in SDL.
  – Allows dynamic creation and termination of process instances and their corresponding communication paths during execution.

• First step is to specify the system in SDL.
**Step I: System Specification in SDL (contd..)**

- SDL provides a hierarchical abstraction of the system structure.
  - Top level is a system level specification.
  - System includes blocks.
  - Blocks include additional blocks or processes.

- Blocks communicate through channels.
  - Channels can be either delaying or non-delaying.

- Process in a block is defined by an extended finite state machine.

- Processes in a block communicate via signal routes.
  - Signal routes have no delay.

- SDL specification provides a process view of a software system.
Step II: Specification Simulation/Execution

- Simulate/execute the system specified in SDL.
- Simulator from Telelogic to simulate the SDL specification.
- Simulator instrumented with TSVAT used to collect trace data during simulation.
- Telcordia Software Visualization and Analysis Tool Suite developed to support architecture specification, debugging and testing and to collect trace data.
Step II: Specification Simulation/Execution (contd.)

- TSVAT based on the creation of a flow graph of the specification, laying out its execution structure.
- Trace files indicate the number of times a given part of the specification, such as a process, a transition, a decision, a state input, or a data flow has been exercised in a single simulation run or at the end of testing.
- Reports coverage with respect to the following criteria:
  - Functions (Processes in SDL).
  - Basic transitions (Statement sequence in SDL that is always executed sequentially, no internal branching constructs).
  - Decisions (Conditional branches from one transition to the other.)
- Execution traces can be used to extract branching probabilities of the various decisions in the specification.
- Simulation guided by an operational profile, then branching probabilities would be a characteristic of field usage.
Step II: Specification Simulation/Execution (contd..)

```plaintext
input ngt;
output busyT;
nexstate waitHangUp;
input hangUp;
nexstate idle;
endstate;

state waitHangUp;
input hangUp;
nexstate idle;
endstate;

state talking;
input hangUp;
output relC;
output done to callee;
nexstate idle;
input done;
output relC;
nexstate waitHangUp;
endstate;

state waitRing;
input ring;
output ringT;
nexstate waitAnswer;
input busy;
output relC;
output busyT;
nexstate waitHangUp;
```
Step III: Translation from SDL Specification to SRN Model

- SDL specification of a system translated to a Stochastic Reward Net (SRN) model.
  - SRN model facilitates quantitative performance and dependability analysis.
- SRNs are a generalization of Generalized Stochastic Petri Nets (GSPNs), which in turn are a generalization of stochastic Petri Nets (SPNs).
- Stochastic Petri Net (SPN):
  - Allows exponential firing times with the transitions.
- Generalized Stochastic Petri Net (GSPN):
  - Exponential as well as zero firing times with transitions.
  - Allows the definition of conditions to inhibit the firing of a transition.
**Step III: Translation from SDL Specification to SRN Model**

- **Stochastic Reward Net (SRNs):**
  - Substantially increase the modeling power of GSPNs by adding guard functions, marking dependent arc multiplicities, general transition priorities, and reward rates at the net level.

- **SRNs provide the same capabilities as Markov Reward Models:**
  - Markov Reward Model is a Markov chain with a reward rate (real number) assigned to each state.
  - Compute measures such as expected reward rate both in the steady state and at a given time, the expected accumulated reward until absorption or a given time, and the distribution of accumulated reward until absorption or until a given time

- **Define rules to translate a process-level SDL specification to a SRN model.**
Parameters of the SRN model obtained by translation from a SDL specification can be categorized into five classes depending on the sources of information used for parameterization:

- Execution time parameters.
- User inputs.
- Branching probabilities.
- Inputs from other components/processes.
- Failure and repair parameters.

Execution time parameters:

- Parameters associated with the execution of tasks and decisions.
- Heavily dependent on implementation specifics.
- Generate code semi-automatically from SDL specifications and use measurements obtained from the execution of this partial code.
Step IV: SRN Model Parameterization (contd..)

• User inputs:
  – Model the inputs representing the actions of a user.
  – Expected by the system at various stages of execution.
  – Distributions and the actual values may be derived from historical data or expert opinion.

• Branching probabilities:
  – Reflect the probabilities of occurrence of the various outcomes of a decision.
  – Extracted from the trace data collected during the simulation/execution of SDL specification.
Step IV: SRN Model Parameterization (contd..)

• Inputs from other processes:
  – Each process in an application may expect certain inputs from other processes in the application.
  – Some parameters may be obtained by considering the execution of other processes in the system.

• Failure and repair parameters:
  – Characterize the failure and repair behavior of the processes.
  – Characterize the failure and repair behavior of each task/decision within a process.
  – Necessary to compute measures such as the reliability and availability.
  – Obtained from historical data or based on expert opinion.
Step V: Reachability Graph Generation

- Reachability graph of a SRN is the set of states that are reachable from other states.
- Generated using SPNP (Stochastic Petri Net Package) developed at Duke University.
- SPNP is a versatile modeling tools for the solution of Stochastic Petri Net (SPN) models.
  - SPN models are described in an input language called CSPL (C-based SPN language).
  - CSPL is an extension of C programming language with additional constructs to facilitate the description of SPN models.
Step VI: Performance and Dependability Analysis

- Parameterized Stochastic Reward Net (SRN) model of the application may be used for performance and dependability analysis.
- SPNP used to compute various measures of interest such as performance, reliability and availability.
- Quantitative analysis based on Markov Reward Model paradigm.
• Introduction and motivation

• Discussion of the methodology

➢ Demonstration of methodology using a case study

• Conclusions and future research
Case Study: Telecom Application

Block-level SDL Specification of Telecom Application

- Two distributed blocks, CallHandler and ResManager.
- CallHandler controls the call processing functions.
- CallHandler has two processes: Caller and Callee
- ResManager involves inventory control and remote database access.
- ResManager has two processes: RMgr and TMgr.
- Communication between CallHandler and ResManager occurs over delaying channels c3 and c4, indicating that the two blocks can be implemented in a distributed fashion.
Case Study: Telecom Application

Process-level specification of the Process TMgr

- Input from other process
- Check for the availability of the resource
- Resource available
  - Send_msg
  - Sendogl
  - Reov_gfl
- Resource unavailable
  - Send_msg
  - Send_rgf
  - Reov_rgf
Case Study: Telecom Application

- Graphical SDL representation of the Telecom Application was transformed to a textual SDL representation.
  - Textual representation was approximately 300 lines of SDL code.
- Textual SDL representation was instrumented with TSVAT.
- SDL specification of the Telecom Application was simulated using the simulator from Telelogic.
- 13 simulations runs were conducted.
  - Trace data was collected during the simulation runs.
Case Study: Telecom Application

Translation from SDL Specification to SRN Model

process TMgr

SIGNAL reqT

SendMsg

Input

Task

Available

Send_gt

Send_gt_tr

Available

Available_i

available_tr

available_y available_n

Available

SendMsg

reqT

available

"yes"

Send_gt

Send_ngt

gt

ngt

Recv_gt

Recv_ngt

Decision
Case Study: Telecom Application

SRN Model of Process TMgr

![SRN Diagram]
Case Study: Telecom Application

Parameterized SRN Model of Process TMgr

- **SendMsg**
  - exp(1) → **reqT**
  - exp(1) → **Available**
  - exp(1) → **available_tr**

- **Fail_Available**
  - exp(0.001)

- **Send_gt**
  - exp(0.001) → **Fail_Send_gt**
  - exp(0.001) → **Send_gt_tr**

- **Send_ngt**
  - exp(1)

- **Send_ngt_tr**
  - exp(1)

- **Available_i**
  - 0.9 → **available_y**
  - 0.1 → **available_n**

- **Received_gt**
  - exp(1)

- **Received_ngt**
  - exp(1)

Branching probabilities extracted from trace data.
Case Study: Telecom Application

Performance and Dependability measures

Dependability measures:
- Probability of resource availability: 0.82
- Probability of resource unavailability: 0.09
- Probability of failure: 0.09

Performance measures:
- Mean Time to Absorption (MTTA): 1001.99
Outline

- Introduction and motivation
- Discussion of the methodology
- Demonstration of the methodology using a case study
  ➢ Conclusions and future research
Conclusions and Future Research

• Proposed a methodology to parameterize the performance and dependability models constructed from architectural specifications in SDL based on the trace data collected during the simulation/execution of the specification.

• Proposed methodology seeks three-way integration of:
  – Architecture specification,
  – Specification simulation/execution and
  – Performance and dependability analysis.

• Demonstrated the methodology using a case study of a Telecom Application.

• Future research includes continued development of methods to estimate parameters of analytical models from simulation, testing and historical data.