Traffic-aware Stress Testing of Distributed Systems Based on UML Models

Vahid Garousi, Lionel Briand, and Yvan Labiche

28th International Conference on Software Engineering (ICSE 2006)

2006. 8. 9
Ah-Rim Han
28th ICSE 2006 in Shanghai (1/4)

Keynote talks – Barry Boehm
A view of 20th and 21st century SE

Registration site

Tea time
28th ICSE 2006 in Shanghai (2/4)
Banquet

Lab members in banquet

Performance in banquet
28th ICSE 2006 in Shanghai (4/4)

Night seeing

Poster session with Dr. Ma
Mutation: A mutation for Java

Shopping mall

With Paul Luo Li and Hyunjung
Contents

- Introduction
- Overview approach
- Building test models
- Derivation of stress test requirements
- Case study
- Conclusion/ future work
- Discussion
Introduction (1/2)

- Distributed system (DS)
  - Development and testing are difficult
    - Physical distribution of objects
    - Parallel execution of objects
    - Hard real-time constraints
  - Even seldom failure costs are quite expensive
    - Public switched telephone network in the United States
      - Study period: 1992-1994
      - Main results
        » Only 6% of outages were overloads and led to 44% of PSTN’s service downtime

Need to develop methodologies and tools for testing DS under stress condition w.r.t. overloads
Introduction (2/2)

- Develop UML model-driven stress testing techniques
  - Focus on
    - Maximize the chances of discovering network traffic faults
      - Not feasible to test all possible interactions
  - Based on
    - UML 2.0 model
      - Augmented with timing information
    - Control flow analysis in sequence diagrams
      - Analysis of the network traffic specific control flow paths entail
Overview approach

INPUT
- Design UML Model
  - Sequence Diagrams
  - Class Diagram
  - Context Diagram
  - Network Deployment Diagram
  - Modified Interaction Overview Diagram

Discuss in this work
- Test Model Generator
  - Test Model (TM)
    - Control Flow Model
    - Network Interconnectivity Tree
    - Traffic Usage Pattern
    - Inter-SD Constraints

OUTPUT
- Optimization Algorithm
  - Stress Test Parameters
    - Stress Test Requirements
  - Tester

Modeler

SUT
- Test Oracles
  - Test Driver
- Test Cases
Building test models

Input UML Model
- Sequence Diagram
- System Class Diagram
- System Context Diagram
- Modified Interaction Overview Diagram
- Network Deployment Diagram

Control Flow Analysis

Arrival Pattern Analysis

Inter-SD Constraint Analysis

Network Traffic Usage Analysis

NDD Path Analysis

Test Model
- Control Flow Model (Concurrent Control Flow Graph)
- Arrival Patterns Model
- Inter-SD Constraints Model
- Independent-SD Set
- Network Traffic Usage Model
- Network Interconnectivity Tree
Control flow model (CFM)

SD with asynchronous messages and timing information

Concurrent control flow graph (CCFG) of the SD

\( \rho_1 = ABC \begin{pmatrix} DE \\ FG \end{pmatrix} \)

\( \rho_2 = ABC \begin{pmatrix} DE \\ HI \\ FG \end{pmatrix} \)

\( \rho_3 = ABC \begin{pmatrix} DE \\ (HI)^2 \\ FG \end{pmatrix} \)

\( \rho_4 = ABC \begin{pmatrix} DE \\ (HI)^3 \\ FG \end{pmatrix} \)

Some concurrent control flow paths (CCFPs) of the CCFG above
Network interconnectivity tree (NIT)

Derivation of network path between two nodes

getNetworkPath(n1, n3) = <Network1, SystemNetwork, Network2>
Network traffic usage pattern (1/3)

- How CCFP entails traffic on a network
- Heuristic
  - Only consider distributed (not local) messages in SDs

\[ \rho_1 = ABC \begin{pmatrix} DE(\ ) \\ FG \end{pmatrix} \]

\[ \Rightarrow DCCFP (\rho_1) = C \begin{pmatrix} DE(\ ) \\ FG \end{pmatrix} \]

Distributed CCFP (DCCFP)
Network traffic usage pattern (2/3)

- Network traffic usage (NTU) function
  - Measure the amount of traffic entailed by a distributed message

\[
\text{NTU}(\text{msg}) = \text{CallDT}(\text{msg}) = \text{dataSize}(A) + \text{dataSize}(B) = (8 \times (100 + 500)) + (8 \times 100 + 2 \times 100) = 5.8 \text{KB (kilobytes)}
\]
Network traffic usage pattern (3/3)

- Network traffic usage pattern (NTUP) function
  - Measure values from the parameter set of
    - Distributed concurrent control flow paths (DCCFP $\rho$)
    - Networks (go through network net - use `getNetworkPath()`)
    - Time domain (particular time instant $t$)

**Network traffic usage pattern of a DCCFG**

**Network traffic usage values of the messages**

<table>
<thead>
<tr>
<th>$m$</th>
<th>$NTU(m)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>60 KB</td>
</tr>
<tr>
<td>D</td>
<td>90 KB</td>
</tr>
<tr>
<td>E</td>
<td>50 KB</td>
</tr>
<tr>
<td>F</td>
<td>30 KB</td>
</tr>
<tr>
<td>G</td>
<td>40 KB</td>
</tr>
</tbody>
</table>
Inter-sequence diagram constraints

- Modified interaction overview diagram (MIOD)
  - Model sequential and conditional constraints between SDs
  - Choose maximum number of SDs running concurrently
    - Independent SD Sets (ISDS)
      - Any two SDs in the set are independent

MIOD example

Nodes: SDs
Edges: iff two SDs are independent

One of the maximal-complete subgraphs → ISDS

\[
\text{ISDS}_1 = \{A, B, D, E\} \quad \text{ISDS}_2 = \{A, B, F\} \quad \text{ISDS}_3 = \{C, D, E\} \quad \text{ISDS}_4 = \{C, F\}\]
Where we are now

INPUT
- Design UML Model
- Sequence Diagrams
- Class Diagram
- Context Diagram
- Network Deployment Diagram
- Modified Interaction Overview Diagram

Discuss in this work
- Test Model Generator
- Test Model (TM)
- Control Flow Model
- Network Interconnectivity Tree
- Traffic Usage Pattern
- Inter-SD Constraints

OUTPUT
- Optimization Algorithm
- Stress Test Parameters
- Stress Test Requirements

Tester

Modeler

SUT
- Test Oracles
- Test Driver

Test Cases
Derivation of stress test requirements (1/3)

- **Heuristics**
  - Time-shifting stress testing techniques
    - Schedule to send all maximum stress messages concurrently
      - \( \text{DCCFP}_{i,\text{max}} : \text{DCCFP which has maximum traffic message among all DCCFPs of a SD}_i \)

---

![Diagram showing NUTP of DCCFPs of an ISDS and a stress test requirement]
Derivation of stress test requirements (2/3)

Algorithm

Step 1
1.1-Find the maximum traffic value, time and messages of each DCCFP over network net
1.2-Among all of DCCFPs of a SD, find the DCCFP with maximum traffic value over network net

Step 2
2-Among all ISDSs, choose the ISDS with maximum traffic over network net

Step 3
3-Schedule the SDs in ISDS_{max} so that all their maximum stress messages happen at the same time

Start time of maximum traffic messages will be used in step 3

\[
\text{max } DT(\text{ISDS}_1) = \sum_{SD_i \in \text{ISDS}_1 = \{A,B,D,E\}} \text{max } DT(\rho_{i,\text{max}}) \\
= \text{max } DT(\rho_{A,2}) + \text{max } DT(\rho_{B,1}) + \text{max } DT(\rho_{D,1}) + \text{max } DT(\rho_{E,1}) \\
= 140\text{ KB}
\]

max DT(\text{ISDS}_2) = 70 \quad \text{max } DT(\text{ISDS}_3) = 127 \quad \text{max } DT(\text{ISDS}_4) = 57 \quad \cdots \cdots \quad 19 / 24
Derivation of stress test requirements (3/3)

- Stress test scheduling
  Start times of maximum traffic messages

- Output stress test requirements
  \[
  < (\rho_{A,1}, 15\text{ ms}), (\rho_{B,1}, 10\text{ ms}), \text{null}, (\rho_{D,1}, 5\text{ ms}), (\rho_{E,1}, 0\text{ ms}), \text{null} >
  \]

Scheduling

\[\rho_{i,\text{max}}: \text{DCCFP which entails the maximum traffic over selected network of SD}_i\]
Case study

- Comparisons on the prototype system: SCADA-based power system (SCAPS)
  - Derived and executed of
    - 500 operational profile-based test cases (standard)
    - 500 stress test cases
  - Results
    - Stress test strategy is effective in detecting violation of hard real-time constraint

Maximum execution time by running OPT and ST
Conclusion

- Conclusion
  - UML model-based network traffic usage analysis in DS
  - Stress test strategy aiming at increasing chances of discovering faults related to network traffic

- Future work
  - Investigate stress testing with other resource types
    - CPU and memory
  - Use UML 2.0 test profile for stress test requirements
    - Improve test automation
Discussion

- Arrival-pattern constraint is not considered
  - Triggering SDs may not be allowed in any time instant
    - Ex) Triggered by a timer on a periodic basis
  - More sophisticated optimization algorithm is needed
    - Genetic algorithm-based stress technique

- Not various stress test strategies are considered
  - Only data traffic of a network in a single time instant
  - Other techniques are needed to reveal other types of faults
For my thesis

- What I have learned
  - UML model-based
    - Systematic and automatic approach
    - Identification and application of control flow in UML 2.0 SDs
    - Assessment of factors affecting system performance
      - Network traffic
  - Distributed systems
    - Similar issues and techniques related to MPSoC (Multiple Processors System on Chip)
      - Overload, communication overhead, etc.
Various stress test algorithms

- Naming conventions of functions used
  - 32 different test strategies to test different nodes and network in a DS

- Example
  - Instant stress testing towards a node
  - Period stress testing from a node
SD and CD showing the effect of classes with polymorphism in CFA