Usage Profile and Platform Independent
Automated Validation of
Service Behavior Specifications

Henning Groenda,
QUASOSS,
04.10.2010
Context

Round-trip Performance Engineering

Legend

- Requirements
- Implementation
- Descriptive Performance Specifications
- Normative Performance Specification
Motivation

Is the specification valid for the parameters in the system?

How accurate is the specification?

Does the specification match an implementation?

Implementation 1
  Version 1

Implementation 1
  Version 2

Implementation 2
  Version 1
Parameterized Specification Example

- Configuration
- Usage Profile
- Required Services
- Hardware Environment
Specification Validation Process

<table>
<thead>
<tr>
<th>&lt;&lt;Component Owner / Creator&gt;&gt;</th>
<th>&lt;&lt;Certification Authority Evaluator&gt;&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>request validation</td>
<td>Specification(s)</td>
</tr>
<tr>
<td>Component Model</td>
<td></td>
</tr>
<tr>
<td>Performance Specification</td>
<td></td>
</tr>
<tr>
<td>Quality Annotations</td>
<td></td>
</tr>
<tr>
<td>Mapping Model</td>
<td></td>
</tr>
<tr>
<td>Code</td>
<td></td>
</tr>
<tr>
<td>Deployment Instructions</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Assessment Step
Activity Diagram

for
create Testcase Specifications (1) → instrument (2)
while
check Stopcondition (3)
do
create Testcase Instance(s) (4) → measure Testcase Instance(s) (5) → check Validity (6)
Validation Process Example

**Synthetization Strategy**
One Partition, Random

**Testcases**
\{param=5,...\},
\{param=1,...\}

**Validity Ranges**

\text{param} = [0..100]
\text{calculate}() = [0..10]

**Implementation**

**Instrumented Implementation**

**Accuracy**
relative=5%, absolute=12

**Stop Condition**
\#Testcases = 100

Henning Groenda - Usage Profile and Platform Independent Automated Validation of Performance Specifications 04.10.2010
Quality Annotation (for PCM)

Henning Groenda - Usage Profile and Platform Independent Automated Validation of Performance Specifications

8

04.10.2010
Validation

- Accuracy of validation:
  Performance validation based on Bytecode

- Instrumentation not mixed with source code
  Use annotation model

- Mapping specification elements and implementation
  Use General Abstract Syntax Tree (GAST)
Implementation

```java
public class FibonacciAlgorithm {

    /** Calculates a Fibonacci number given the number of rounds the algorithm should run. *
     * @param rounds Number of calculation rounds. *
     * @return Fibonacci number. *
     */
    public long fibonacci(long rounds) {
        long i1 = 0;
        long i2 = 1;
        long i3 = 0;
        // normalized loop
        for (long i = 0; i < rounds; i++) {
            long i3 = i1 + i2;
            i2 = i1;
            i1 = i3;
        }
        return i3;
    }
}
```

GAST

```
Package de
  Package de.fzi
    Package de.fzi.ByCounter
      Package de.fzi.ByCounter.example
        Package de.fzi.ByCounter.example.Fibonacci
          GAST Class de.fzi.ByCounter.example.Fibonacci.FibonacciAlgorithm
            Comment _-i8V8UYUEDi-Yb1IWQK_sg
            Position FILE:C:
            Constructor _-i4VYEUDi-Yb1IWQK_sg
            Method fibonacci(long):long
              Comment _-i8V8kYUEDi-Yb1IWQK_sg
              Comment _-i8V80YUEDi-Yb1IWQK_sg
              Position FILE:C:
              Block Statement _-IkPQYUEDi-Yb1IWQK_sg
              Local Variable _-jjwZEYUEDi-Yb1IWQK_sg
              Local Variable _-jjwZYUEDi-Yb1IWQK_sg
              Local Variable _-jjwaEYUEDi-Yb1IWQK_sg
              Local Variable _-jjwakYUEDi-Yb1IWQK_sg
              Declaration Type Access _-h3uR0YUEDi-Yb1IWQK_sg long
              Formal Parameter _-jKxGfYLUed-iYb1IWQK_sg
              Inheritance Type Access _-kWalkYUEd-iYb1IWQK_sg java.lang.Object
```
public class FibonacciAlgorithm {

    /**
     * Calculates a Fibonacci number given the number of rounds the algorithm should run.
     * @param rounds Number of calculation rounds.
     * @return Fibonacci number.
     */
    public long fibonacci(long rounds) {
        long i1 = 0;
        long i2 = 1;
        long i3 = 0;
        // normalized loop
        for (long i = 0; i < rounds; ++i) {
            i3 = i1 + i2;
            i2 = i1;
            i1 = i3;
        }
        return i3;
    }

}
Parameter

- rounds: 10

Specification

Measurement

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>LCONST_0</td>
<td>✓</td>
</tr>
<tr>
<td>11</td>
<td>LCONST_1</td>
<td>✓</td>
</tr>
<tr>
<td>73</td>
<td>LLOAD</td>
<td>✓</td>
</tr>
<tr>
<td>44</td>
<td>LSTORE</td>
<td>✓</td>
</tr>
<tr>
<td>20</td>
<td>LADD</td>
<td>✓</td>
</tr>
<tr>
<td>11</td>
<td>LCMP</td>
<td>✓</td>
</tr>
<tr>
<td>11</td>
<td>IFLT</td>
<td>✓</td>
</tr>
<tr>
<td>1</td>
<td>GOTO</td>
<td>✓</td>
</tr>
<tr>
<td>1</td>
<td>LRETURN</td>
<td>✓</td>
</tr>
</tbody>
</table>
- No need to compensate warmup and teardown effects
- No need to identify and exclude outliers

<table>
<thead>
<tr>
<th>rounds</th>
<th>Response Time (ns)</th>
<th>Response Time (ns)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>pure</td>
<td>instrumented</td>
</tr>
<tr>
<td>1</td>
<td>6 161</td>
<td>141 197 445</td>
</tr>
<tr>
<td>10</td>
<td>3 920</td>
<td>105 979 160</td>
</tr>
<tr>
<td>100</td>
<td>3 360</td>
<td>105 463 856</td>
</tr>
<tr>
<td>1000</td>
<td>10 082</td>
<td>137 462 045</td>
</tr>
</tbody>
</table>

- Significant overhead for validation
  - Vs. re-use of specifications
  - Vs. creating a new specification for each use-case
  - Vs. integrating and measuring a service in a test system
Assumptions & Limitations

- Focus on
  - Business services
  - CPU resource demand

- Internal state analysis not implemented yet
- Runtime-weaving is not supported yet
- Currently supports validation of internal actions
Conclusion Future Work

- Automated validation of performance specifications
  - Exact measurements without disturbances
  - Validation of parameterized specifications
  - Support for fine-grained regions of validation
  - Supports validation of more than one method
  - Documented specification quality: Trust, error propagation

- Future Work
  - Support the use of required services
  - Improved test case synthetization strategies
    - Block coverage (on specification)
    - Condition coverage (on specification)
Next Steps

- **Theory**
  - WP 5: Test Strategies for Performance Specifications
    - Overview on control-flow based strategies
    - Discoverable error types on functional level
    - Semantic translation to performance specifications
    - Value of applicability on performance specifications

- **Testbed**
  - Support for external calls
  - Support for component parameters
  - Support for complex parameters
  - Support for parameter characterizations
  - Dynamic testcase synthetization (pre- vs on-demand-creation)
```java
public class FibonacciAlgorithm {
    public long fibonacci(long rounds) {
        long i1 = 1;
        long i2 = 1;
        long i3 = 0;
        // normalized loop
        for (long i = 0;
             i < rounds;
             i++) {
            i3 = i1 + i2;
            i2 = i1;
            i1 = i3;
        }
        return i3;
    }
}
```
Example Input

GAST Model

Package de
  Package de.fzi.ByCounter
    Package de.fzi.ByCounter.example
      Package de.fzi.ByCounter.example.Fibonacci
        GAST Class de.fzi.ByCounter.example.Fibonacci.RunExample
        GAST Class de.fzi.ByCounter.example.Fibonacci.FibonacciAlgorithm
        Comment -_l8VUYUEd-Yb1WQk_sg
        Position FILE:C:\data\wrk\Projekte\Part III - Certification of Performance Specifications\fu
        Constructor -h4VUYUEd-Yb1WQk_sg
        Method fibonacci(long)long
          Comment -_B8kYUEd-Yb1WQk_sg
          Comment -_B80YUEd-Yb1WQk_sg
          Position FILE:C:\data\wrk\Projekte\Part III - Certification of Performance Specifications\fu
          Block Statement -KiPQYUEd-Yb1WQk_sg
          Local Variable -_jwZEYUEd-Yb1WQk_sg
          Local Variable -_jvzKUEd-Yb1WQk_sg
          Local Variable -_jvzZeYUEd-Yb1WQk_sg
          Local Variable -_jvzakYUEd-Yb1WQk_sg
          Declaration Type Access _h3uRlUYUEd-Yb1WQk_sg long
          Formal Parameter -jXgEYUEd-Yb1WQk_sg
          Inheritance Type Access _kWolUEYUEd-Yb1WQk_sg java.lang.Object

Link Model

Performance Specification Model

GAST Model - Usage Profile and Platform Independent Automated Validation of Performance Specifications

Henning Groenda - Usage Profile and Platform Independent Automated Validation of Performance Specifications 04.10.2010
Example

Quality Annotations

Performance Specification Model

Quality Annotation

Validation Quality

Quality Annotation

Henning Groenda - Usage Profile and Platform Independent Automated Validation of Performance Specifications

04.10.2010
Example

Test case partitions / Test case instance

Validation Quality

<table>
<thead>
<tr>
<th>Feature</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Internal State Influence Analysis</td>
<td>false</td>
</tr>
<tr>
<td>Number Testcases Per Partition</td>
<td>100</td>
</tr>
<tr>
<td>Stop Condition Type</td>
<td>OnePartitionRandom</td>
</tr>
<tr>
<td>Test Case Generation Type</td>
<td>OnePartitionRandom</td>
</tr>
<tr>
<td>Acceptable Absolute Deviance</td>
<td>0.0</td>
</tr>
<tr>
<td>Acceptable Relative Deviance</td>
<td>0.0</td>
</tr>
</tbody>
</table>

Example

Model

Example

Model Validation Quality

QualityAnnotation

ValidatedSetting

ParameterRange

TestcasePartition

TestcaseInstance

ParameterPartition

Range

ParameterType

<enumeration>

InputParameter

ReturnValue

Component

Henning Groenda - Usage Profile and Platform Independent Automated Validation of Performance Specifications 04.10.2010
Instrumentation Profile

Performance Specification Model

Link Model

Instrumentation Model

Henning Groenda - Usage Profile and Platform Independent Automated Validation of Performance Specifications 04.10.2010
Example

Test case execution result

2x LCONST_0
3x LCONST_1
10x LLOAD
8x LSTORE
2x LADD
2x LCMP
2x IFLT
1x GOTO
1x LRETURN

Example

Model
Example Comparison

- 2x LCONST_0
- 3x LCONST_1
- 10x LLOAD
- 8x LSTORE
- 2x LADD
- 2x LCMP
- 2x IFLT
- 1x GOTO
- 1x LRETURN

Example fibonacci number calculation

Resource Demand
- 2 <Generic JVM> LCONST_0
- 3 <Generic JVM> LCONST_1
- 10 <Generic JVM> LLOAD
- 8 <Generic JVM> LSTORE
- 2 <Generic JVM> LADD
- 2 <Generic JVM> LCMP
- 2 <Generic JVM> IFLT
- 1 <Generic JVM> GOTO
- 1 <Generic JVM> LRETURN

Certified

Report
Next Steps

- Additional parameter types
  - Component
  - Return-value of external calls

- Parameter characterizations
  - Mapping of technical and performance-relevant interface
  - Structural-based information, e.g. nodes and edges of graph
  - Non-structural-based information, e.g. compression or clustering ratio
  - Support for synthetization and characterization detection

- Internal state
  - Detect size of internal state influence: Evaluate by re-executing with the same parameters but in other order.
Next Steps

- Testcase generation
  - Random
  - Equidistant partitions (input or output)
  - Output partitions with equal change
    - Linear
    - Integral
  - Weight depending on magnitude of change
  - Based on implementation analysis

- Stop-Conditions
  - Number of test cases per partition
  - Choose Metric, e.g. Coverage-/GAST-Analysis
    - Estimated error probability / confidence
Next Steps

- **Case Study: Fibonacci algorithm**
  - Evaluate for all values within range; Extend range until validation fails
  - Measure effort, show chosen values
  - Random vs. equidistant input partitions
Challenges

- How to validate parameterizes specifications which are based on control-flow abstractions?
- How can usage-profile and hardware environment independent validations be supported?
- How can the quality of a specification be quantified? Which metrics are useful?
- How can the quality of the validation of a specification be evaluated? Which test case generation strategies and stop conditions allow to cover which error types?
- How can certification be integration into a component-based development process?
- Which instrumentation granularity should be choosed to validate parameterized specifications?