Oracle Strategies for Automated Testing

Quality Week 2000

Douglas Hoffman
Software Quality Methods, LLC.
24646 Heather Heights Place
Saratoga, California 95070-9710
Phone 408-741-4830
Fax 408-867-4550
doug.hoffman@acm.org

Copyright © 2000, Software Quality Methods, LLC. No part of these graphic overhead slides may be reproduced, or used in any form by any electronic or mechanical duplication, or stored in a computer system, without written permission of the author.
Describe Your Typical Automated Test

• What is being tested
• How is the test set up
• Where are the inputs coming from
• What is being checked
• Where are the expected results
• How do you know pass or fail
Your Typical Automated Test
Regression Strategy

• Write the test exercise
• Run it
• Capture the results
• “Verify” the results
• Run it again as a regression test
• Compare current with captured results
A Regression Test Model

![Diagram showing a regression test model with User, GUI Test Tool, SUT GUI, Scripts, and Results]
Regression Advantages

• Straightforward
• Same approach for all tests
• Fast implementation
• Variations easy
• Repeatable tests
Regression Disadvantages

• “20 Questions”
• Master validation
• Analysis of failures
• Limited scope
There’s A Universe Beyond Regression Automation
Automated Software Tests

- No intervention needed after launching tests
- Automatically sets-up and/or records relevant test environment
- Runs test exercise
- Captures relevant results
- Evaluates actual against expected results
- Reports analysis of pass/fail
Levels of Automation

- Fully automated software testing
- Semi-automated software testing
- Manual software testing
Simple Testing Model
(Black Box)

Test Inputs → System Under Test → Test Results
Expanded Testing Model (Black Box)
Implications of the Expanded Model

- We don’t control all inputs
- We don’t verify everything
- Multiple domains are involved
- Test exercise may be the easy part
- We can’t verify everything
- We don’t know all the factors
Size Of The Testing Problem

- Input one value in a 10 character field
- 26 UC, 26 LC, 10 Numbers
- Gives $62^{10}$ combinations
- How long at 1,000,000 per second?

What is your domain size?
We can only run a vanishingly small portion of the possible tests
Choosing The Subset

• High value tests
• Exploratory testing
• Automated tests
• More powerful exercises
More Powerful Exercises

• Increasing the number of combinations
• Self-verifying tests and diagnostics
• More frequency, intensity, duration
• Increasing the variety in exercises
Pseudo Random Numbers

• Used for selection or construction of inputs
  – With and without weighting factors
  – Selection with and without replacement
• Statistically “random” sequence
• Randomly generated “seed” value
• Requires oracles to be useful
Random Selection Among Alternatives

- Pseudo random numbers
- Partial domain coverage
- Small number of combinations
- Use an oracle for verification
Where Do We Fit In The Oracle?

• Identify what to verify
• How do we know the “right answer”
• How close to “right” do we need
• Decide when to generate the expected results
• Decide how and where to verify results
• Get the oracle(s)
Testing With An Oracle

Test Oracle

System Under Test

Test Inputs
Precondition Data
Precondition Program State
Environmental Inputs

Test Results
Postcondition Data
Postcondition Program State
Environmental Results
Automation Architecture

- Model for SUT and environment
- Break down software testing problem
- Decide on location(s) of automation
- Decide on level(s) of automation
- Describe automation architecture
A Model For SUT

System Under Test

User → GUI → Functional Engine → Remote GUI → User

API

Data Set
Automated Software Testing Process Model

Tester → Test List

Test List → Automation Engine

Automation Engine → SUT

SUT → Data Set

Test Results ← Automation Engine

Test Results ← Test List
Process Model Example

1. Testware creation, version control, and configuration management
2. Selecting the subset of test cases to run
3. Set-up and/or record environmental variables
4. Run the test exercises
5. Monitor test activities
6. Capture relevant results
7. Compare actual with expected results
8. Report analysis of pass/fail
Oracle Characteristics

• Completeness of information from an oracle
• Accuracy of information from an oracle
• Usability of results
• Temporal relationships
• Supportability
Completeness

- Sufficiency
- Input Coverage
- Result Coverage
- SUT environments
- Types of errors possible
Accuracy

• How similar
  – Arithmetic accuracy
  – Statistically similar

• How independent

• Types of possible errors
Usability

• Form of information
• Location of information
• Availability of comparators
• Support in SUT environments
• Cost
Temporal Relationships

• How fast to generate results
• How fast to compare
• When is it run
• When are results compared
Supportability

- COTS or custom
- Correspondence through SUT changes
  - Test exercises
  - Tools
- Ancillary support activities required
### Oracle Approaches

<table>
<thead>
<tr>
<th>True Oracle</th>
<th>Heuristic Strategy</th>
<th>Consistency</th>
<th>Self Referential</th>
<th>No Oracle</th>
</tr>
</thead>
<tbody>
<tr>
<td>Definition</td>
<td>-Independent generation of all expected results</td>
<td>-Verifies some values, as well as consistency of remaining values</td>
<td>-Embeds answer within data in the messages</td>
<td>-Doesn’t check correctness of results, (only that some results were produced)</td>
</tr>
<tr>
<td>Advantages</td>
<td>-Faster and easier than True Oracle</td>
<td>-Fastest method using an oracle</td>
<td>-Allows extensive post-test analysis</td>
<td>-Can run any amount of data (limited only by the time the SUT takes)</td>
</tr>
<tr>
<td></td>
<td>-Much less expensive to create and use</td>
<td>-Verification is straightforward</td>
<td>-Verification is based on message contents</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>-Can generate and verify large amounts of data</td>
<td>-Can generate and verify large amounts of complex data</td>
<td></td>
</tr>
<tr>
<td>Disadvantages</td>
<td>-Expensive to implement</td>
<td>-Original run may include undetected errors</td>
<td>-Must define answers and generate messages to contain them</td>
<td>-Only spectacular failures are noticed.</td>
</tr>
<tr>
<td></td>
<td>-Complex and often time-consuming when run</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
True Oracle

- Independent implementation
- Complete coverage over domains
  - Input ranges
  - Result ranges
- “Correct” results
- Usually expensive
Heuristic Strategy

• Rules of thumb
  – Estimates
  – Approximations
  – Trends

• Levels of abstraction
  – General characteristics
  – Statistical properties
Consistency Strategy

• A / B compare
• Check for changes
• Regression checks
  – Validated
  – Unvalidated
• Alternate versions or platforms
• Foreign implementations
Self-Referential Strategy

• Embed results in the data
• Cyclic algorithms
• Shared keys with algorithms
‘No Oracle’ Strategy

• Easy to implement
• Tests run fast
• Only spectacular errors are noticed
• False sense of accomplishment
Choosing Which Strategy

• Decide how the oracle fits in
• Identify the oracle characteristics
• Prioritize testing risks
• Choose a combination of approaches
Heuristic Oracle Examples

• Live data base
  – selected records using specific criteria
  – selected characteristics for known records
  – standard characteristics for new records

• Data base engine
  – correlated field values (time, order number)

• Data communications
  – CRC

• Sine function
Self-Referential Oracle Examples

- Data base
  - embedded linkages
- Data communications
  - value patterns (start, increment, number of values)
- Noel Nyman’s “Self Verifying Data”*

* “Self Verifying Data - Validating Test Results Without An Oracle,” STAR East 1999
Mutating Automated Tests

- Closely tied to instrumentation and oracles
- Using pseudo random numbers
- Positive and negative cases possible
- Diagnostic drill down on error
Mutating Tests Examples

• Data base contents (Embedded)
• Processor instruction sets (Consistency)
• Compiler language syntax (True)
• Stacking of data objects (None)
Summary

• Decide what you are testing
• Decide how to exercise it
• Decide what can tell pass from fail
• Analyze technical factors, costs, and risks
• Select oracle strategy