Ten Years of Model Based Testing
A Sober Evaluation

MBT 2006
March 26 2006
Alan Hartman
IBM Israel – Haifa Research Laboratory
Joint work with Kenneth Nagin, Sergey Olkovsky, Andrei Kirshin and others

Agenda
• What is Model Based Testing
• Value of Model Based Testing
• Selected highs and lows from ten years in MBT
• Barriers to Model Based Testing
• Where do we go from here

What is Model Based Testing
• All testing is based on a model
• MBT occurs when the model is:
  – formalized
  – recorded in some form
  – used for generating test cases or oracles
• Model Driven Testing is a special case of MBT
  – Abstract models
  – Automated test transformations

The testing value proposition
• Testing is expensive
  – 30-50% of development costs
• Poor testing is VERY expensive
  – Down time
  – Maintenance costs
  – Rework
  – Law suits
• Model Based Testing promises
  – Increased effectiveness of testing
  – Similar or decreased costs
  – Reuse of design artefacts
Downtime Costs (per Hour)

- Brokerage operations $6,450,000
- Credit card authorization $2,600,000
- Ebay (1 outage 22 hours) $225,000
- Amazon.com $180,000
- Package shipping services $150,000
- Home shopping channel $113,000
- Catalog sales center $90,000
- Airline reservation center $89,000
- Cellular service activation $41,000
- On-line network fees $25,000
- ATM service fees $14,000

Costs of fixing a bug

<table>
<thead>
<tr>
<th>Percentage of Bugs</th>
<th>Cost to repair defect in this phase</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coding</td>
<td>$25</td>
</tr>
<tr>
<td>Unit Test</td>
<td>$130</td>
</tr>
<tr>
<td>Function Test</td>
<td>$250</td>
</tr>
<tr>
<td>Field Test</td>
<td>$1,000</td>
</tr>
<tr>
<td>Post Release</td>
<td>(Apar $15-40,000)</td>
</tr>
</tbody>
</table>

Classical V Process

- Specification
- Design
- Implementation
- System testing
- Function testing
- Unit testing

Offline Model Based Testing

- Specifications
- Test Objectives
- Test Architecture
- Generation Directives
- Execution Directives
- Behavior Model
- 1. Create Model
- 2. Create Directives
- 3. Generate Test Suites
- 4. Review
- 5. Execute Test
- 6. Analyze and Feedback

Test Suite

Test Results

Unit Under Test
Online Model Based Testing

Specifications → Behavior Model → Inline Generator

Test Objectives

Iteratively:
1. Consult test objectives
2. Generate a test step in the model
3. Apply the step to the UUT
4. Observe the result
5. Consult the model for validation

Unit Under Test

Values of Model Based Testing

• Starting from specification
  – Involves testers early in the development process
  – Teams testers with developers
  – Forces testability into product design

• Building the test interface
  – Finds design and specification bugs - before code exists
  – The model is the test plan - and is easily maintained

• Automated test suite generation
  – Coverage is guaranteed - increases testing thoroughness
  – Zero test suite maintenance costs

• Automated test suite execution
  – Finds code and interface bugs
  – Includes a framework for the testing of distributed applications
  – Reduces test execution costs

What is the reality?

• 1997 – GOTCHA for processor architecture verification
• 1998 – IBM US PoC with very experienced tester
• 1999 – IBM US file system test
• 2002 – AGEDIS project, FT, DB application, Messaging system
• 2001 – IBM Telephone company engagement
• 2003 – Health care ISV
• 2005 – IBM Java compliance

What is the reality?

• 1997 – GOTCHA for processor architecture verification
• Coverage driven testing
• Tools were used by Ph. D.s
  – “Successful” in the lab., not production
  – Hardware not software
What is the reality?

- 1998 – IBM US Proof of Concept with very experienced tester
- First funding for our under-cover project in SW testing
- Reality hits
  - Violent resistance
  - I can do better by hand
  - Bad UI

1999 – IBM US file system test

- Retest of functions
- Modelling and translation by testers
- Comparison
  - Original test: 18 bugs, 12 PM
  - Pilot test: 15 original bugs + 2 escapes, 10 PM (INCLUDING learning curve)
- Conclusions:
  - Efficient way to free the tester for creative testing
  - Replaces a large part of the manual test case writing
- Reality:
  - Never used again

Was the problem our modelling language?

- GDL was clunky, looked like Pascal, came from hardware, not sexy
- 2002 – AGEDIS project - move to UML
- Tool collaboration
  - Produced a camel rather than a racehorse
- Three industrial teams
  - Another Ph. D.
  - A below standard test engineer
  - A genius with the heart of a toolmaker (the NIH - Not Invented Here - problem)

What is the reality?

- 2001 – IBM Telephone company engagement
- Rapid deployment
  - Team of five crack testers with a tight deadline
  - Enormous volume of testing
- Created an automated process for converting existing Cobol artifacts into GDL models
- The reality
  - Mountains of bugs uncovered
  - 60% of the bugs were documentation bugs
  - No repeated use of the tools
What is the reality?

• 2003 – Health care ISV
• Conversion of existing testing artifacts into GDL models
• Simplification of the UI
• Modelling using a spreadsheet
• Simplified coverage criteria
• The reality
  – Our champion at the ISV got a promotion
  – No repeat business

What is the reality?

• 2005 – IBM Java compliance
• Thousands of classes with a reference implementation
• Use reflection to generate models
• The tester never sees a model

Barriers to Model Based Testing

• Process shift
  – Up front investment in test
• Personnel shift
  – Higher education and sophistication
• Tools
  – Still bleeding edge

Modelling is not easy

• What details do you leave out?
• How do you check the model?
• There are no recipes for success
So where to now?

• I still believe in model based testing
  - But keep the models away from the (average) testers
• I believe less in model based toolkits for testers
  - MBT should be a service offering not a tool product
• Hide the complexities as far as possible
  - Automate whenever possible
  - KISS
• Reuse existing artifacts to generate models
  - Creative tooling
  - Domain specific

References

• AGEDIS Papers: www.agedis.de

Advertising

• ECMDA workshop on Integration of Model Driven Development and Model Driven Testing
  
  Bilbao – July 11th 2006
  
  • Submission Deadline: April 16th 2006
  • http://www.fokus.fraunhofer.de/event/imdt/