MBT for Large-Scale Enterprise Information Systems

Challenges and Quality Issue

Bruno Legeard

CTO
legeard@smartesting.com
Introduction - Smartesting

Model-based testing to industrialize the functional testing process

Challenges and Quality Issues

MBT in practice – Lessons learned
Smartesting in a nutshell

• Independent Software Vendor, model-based testing solution provider, founded in 2003.
• Spin-off of a Computer Science Lab in France – University of Franche-Comté/LIFC
• 32 people - R&D Center and HQ in Besançon (France), Offices in Paris, NL and India (Bangalore)
• Focus on IT applications (Large-scale Enterprise Information Systems, Packaged applications – SAP, Oracle…) and Secured transaction Systems
Agenda

Model-based testing to industrialize the functional testing process

Challenges and Quality Issues

MBT in practice – Lessons learned
Context of large-scale Enterprise Information Systems

- System of systems
  - Several applications
  - Mix of Bespoke and Packaged applications
  - Mix of data-oriented and process-oriented applications

- Several levels of functional testing
  - Application/system testing
  - Multi-application/Acceptance/End-to-end testing
Functional testing of IT applications – Current situation

“Functional Testing is still 80% manual”

Business Needs Analysis

- How to ensure the quality of the test coverage?
- How to increase the percentage of automated testing?
- How to master costs and risks of functional testing?
Software testing industrialization based on MBT

• Continuous process from business requirements to the test execution
• Systematic coverage of expected behaviors
• Accelerate test design and test maintenance
• Compliant with Requirements & Risk-based testing approaches.
Smarttesting MBT process

**Requirement management**
- Requirements
  - Dedicated tool or
  - Business Expert

**Smarttesting Test Designer™**
- Test Model

**Test management environment**
- Requirements
  - link
- Manual tests
  - Test scripts
    - call
- Test scripts
  - Keywords automation
  - Test result

**Test execution environment**
- Test result

**Test automation engineer**
Test Model – 2/3

Test Model

- Business Process Model (BPMN)
- Behavior Model (UML/OCL)
- Business Domain Model (UML)

Test Strategy

- Test Suite
  - Init State
  - Scope
  - Strategies
Test Model – 3/3

With OCL to formalize the expected behavior
### Example - SAP integration testing

**REGRESSION TEST SD – Sales order - invoice**

<table>
<thead>
<tr>
<th>Business process steps</th>
<th>Transaction, input</th>
<th>Expected results</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Creation of a sales order in reference to a contract</td>
<td>VA01</td>
<td></td>
</tr>
<tr>
<td>Order type</td>
<td>ZC-S4</td>
<td></td>
</tr>
<tr>
<td>Sales organization</td>
<td>0056</td>
<td></td>
</tr>
<tr>
<td>Distribution channel</td>
<td>00</td>
<td></td>
</tr>
<tr>
<td>Division</td>
<td>00</td>
<td></td>
</tr>
<tr>
<td>Sales office</td>
<td>F005</td>
<td></td>
</tr>
<tr>
<td>Sales group</td>
<td>520</td>
<td></td>
</tr>
<tr>
<td>Note: Create with reference, go to the “contract” tab enter contract</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Click on “Copy”</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Press enter</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1. Item line</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Material</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Order quantity</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Amount</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Select the line and click on the “display item details” + go to “account assignment” tab</td>
<td></td>
<td></td>
</tr>
<tr>
<td>WDZ element</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Save</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2. Creation of a sales order</td>
<td>VA01</td>
<td></td>
</tr>
<tr>
<td>(Sales ordertype doesn’t exist)</td>
<td>ZC-CC</td>
<td></td>
</tr>
<tr>
<td>Enter type</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Press enter</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3. Creation of a sales order</td>
<td>VA01</td>
<td></td>
</tr>
<tr>
<td>(Sales ordertype not linked to sales area)</td>
<td>ZC-S1</td>
<td></td>
</tr>
<tr>
<td>Order type</td>
<td>0035</td>
<td></td>
</tr>
<tr>
<td>Sales organization</td>
<td>00</td>
<td></td>
</tr>
<tr>
<td>Distribution channel</td>
<td>00</td>
<td></td>
</tr>
<tr>
<td>Press enter</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4. Creation of a sales order in reference to a contract</td>
<td>VA01</td>
<td></td>
</tr>
<tr>
<td>(Mandatory data not filled)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Click on “create with reference”</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Error 000.58: Entry ZC-CC does not exist in TVAR - check your entry</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Specifications documentation**

**Sales and Delivery SAP Module**
Business flows Modeling for the SD SAP module (with BPMN)

Zoom on PreSales Process

- Customer
  - Customer requests quotation
  - Customer approval?
    - Yes
    - No

- Sales administration
  - Sales Quotation Entry (VA21)
    - No
    - Yes
  - Sales Quotation Rejection (VA22)
    - Long term supply?
      - Yes
      - No
      - Purchase info record creation (ME11)
    - Sales contract creation (VA43)
Business entities as a class diagram

Business Domain model
Business rules as a behavior model (in OCL)

---@REQ: VA41 - Create a contract
---@AIM: For the creation of a standard contract you should not use an inter-company customer otherwise, error: ZNS086: The donneur d’ordre / client livré is a company client. Cross-company flow out of impossible
---@NAME: Standard contract could not have an inter-company customer
  
  self.message = MESSAGE::ERR_STANDARD_CONTRACT_COULD_NOT_HAVE_AN_INTER_COMP_CUSTOMER and
  self.tobecontinued = false
  
  ...
Automated test generation

- Requirements traceability
- Smart coverage
- Risk based testing
- Automated impact analysis
Test Generation in Smartesting Test Designer

Generated tests

Test contents

Test attributes
Test publication: manual and automated tests

Test Repository

Manual testing + Automated testing

Test Analyst

Publication

Automation engineer
Publication - HP Quality Center

[Image of HP Quality Center interface]

© SMARTESTING 2009 – This document is the property of Smartesting. It may not be reproduced in whole or in part
Smartesting Test Designer

**key points**

- Based on standard UML 2.x and BPMN 1.x or 2.0
  - Integrated with modeling tools under Eclipse/UML (IBM RSM, open-source, Borland/MicroFocus Together)

- Integration into existing test environments
  - Test management & test execution tools (HP QC/QTP, IBM RQM/RFT, TestLink/Selenium)

- Test generation strategies
  - Structural model coverage criteria
  - Scenario & Schema-based testing
Agenda

- Model-based testing to industrialize the functional testing process
- Challenges and Quality Issues
- MBT in practice – Lessons learned
Challenges & Quality Issues #1

 Alignment of the test repository with the business requirements
  - How to facilitate the communication between business analysts and testers?
  - How to prioritize the test design with the critical business processes (particularly for end-to-end testing) in a risk-based approach?
Roles and collaboration (1)

- Business Analyst
- Test Analyst

Test Model

- Business Process Model (BPMN)
- Business Domain Model (UML)
- Behavior Model (UML/OCL)

Test Strategy

- Test Suite
  - Init State
  - Scope
  - Strategies
Roles and collaboration (2)

Business Analyst

Test Analyst

Automation engineer

Behavior model

Business Models and flows

BPMN

UML

‘Multi layers model’

Test components

BPT
Challenges & Quality Issues #2

How to manage the complexity and re-usability of the test model

- Collaborative work (large testing team for large-scale enterprise IT)
- Several levels of testing
  - Application testing
  - End-to-End testing
Test model - Layered structure

System of systems layer

Information System Business Process Model

Application layer

Application A

Application B

Services and Business Entities layer

Business Entities – Groupe 1

Business Entities – Groupe 2

Service 1

Service 2
Challenges & Quality Issues #3

MBT should manage both manual and automated testing

- Testers in the IT field reserve test automation for regression testing
- How to manage both manual and automated test execution in the MBT process?

MBT enhances the Key-word testing approach
Fully documented test repository

From the description of operations in the Test model to the documentation of each steps in the test repository.
Agenda

Model-based testing to industrialize the functional testing process

Challenges and Quality Issues

MBT in practice – Lessons learned
Lesson learned #1: Modeling paradigms

❖ Multi-paradigm approach (see W. Grieskamp – 2006 FATES invited paper)
  – Diagrammatic and Textual notations
  – Pre-post, State-based and Process-based modeling paradigms

❖ Supporting model sharing and re-use
  – Structuring models using packages
  – Model composition
Lesson learned #2: Dedicated test model

Model-based testing: the inversed Y model

Reusability of business process models and business domain models from the Req analysis phase..

→ Dedicated behavior test model for automated test generation
Lesson learned #3: Test generation strategies

Two complementary approaches:

– Structural model coverage
  • Using classical coverage criteria (Transition & decision coverage, path coverage, data coverage)
  • Based on risk analysis

– Scenario-based test generation
  • Generation based on high level scenario or use cases
  • Symbolic animation of scenario on the model to compute the expected results and input parameters
  • Generating tests from test schemas
Lesson learned #4: Integration issues

MBT must be integrated with test management and test automation tools.
Agenda

Model-based testing to industrialize the functional testing process

Challenges and Quality Issues

MBT in practice – Lessons learned

Conclusion – Where is MBT
Conclusion

MBT is now emerging in industry for several reasons:

– The complexity of software applications continues to increase, and the user’s aversion to software defects is greater than ever;

– The cost and time of testing is already a major proportion of many projects; MBT can decrease the overall cost of test by generating manual and automated tests;

– The MBT approach and the associated tools are now mature enough to be applied in many application areas.
Technology adoption life cycle

Innovators (techies)  Early adopters (visionaries)  Early majority (pragmatists)  Late majority (conservatives)  Laggards (sceptics)

MBT
Some Research Challenges in MBT

- Fitting MBT into existing processes (including agility) and tool chains
- How to make MBT modeling easier (behavioral part)?
  - Link with requirements analysis activities?
  - Dedicated Domain Specific Modeling languages?
- Link between MBT approach and Code analysis
- Effectiveness of test generation strategies (Theoretical & Experimental works needed)
- MBT for non functional properties (security, usability, performance, availability…)

© SMARTESTING 2009 – This document is the property of Smartesting. It may not be reproduced in whole or in part