Model Based Test Automation
Avoiding the maintenance trap
Hans Planken
Vice President Alliances EMEA
January 2015
Automation level below 25% even after 30 years of software testing. Why?

Do we need a different approach?
Evolution of Complexity - Testing Challenges

- Mainframe
- Client/Server
- Web
- Mobile

Complexity
What’s the problem? A typical test project.

If you don’t know how much test coverage you have, you’ll end up with far too many test cases!

Not effective enough

Automation levels are low because of it’s difficult to automate!

Not efficient enough

Source: TRICENTIS® customer projects

© 2014 Tricentis GmbH. All rights reserved.
Tricentis Tosca Testsuite

Virtualize – Orchestrated Service Virtualization

Optimize
Risk Coverage Optimizer

Manage
Test Data Provisioning

Automate
Model-Based Test Automation

Target Applications
- HTML
- Android
- iOS
- Java
- .NET
- XML
- SAP
- ORACLE

Reporting & Dashboard

Technology Integration
- HP
- RALLY
- JIRA
- Jenkins
- Tasktop
- Office
Tricentis Tosca Testsuite

Aligns **business priorities** to test planning & execution

Generates the **fewest test cases** to maximize **risk coverage**

Shifts focus from **Test Coverage** to **Risk Coverage**
Synthetic Test Data Engine

Total effort/costs

100%

75%

50%

25%

0%

Test Case Specification

Manual Test Execution

Finding and Preparing Test Data

Execution of Test Cases

Automation

insufficient control of test data

50 - 75% of manual execution costs!

Source: Customer Projects TRICENTIS®
Tricentis Tosca Testsuite

Provides tailored **test data basis** for test execution

Reduces effort in **manual test execution** by 40 – 50 %

**Stateful test data management** is key for high automation rates
Enables **extreme automation** (rates > 90%)

Covers **UI and nonUI test automation**

Enables seamless transition from **manual** to **automated testing**
What makes it different?


Model-Based Test Automation: Robust, Cost-Effective, Scalable
Model-Based Test Automation
Robust, Cost-Effective, Scalable
What is the Problem?

Efficiency: Why are automation levels so low?

What keeps us from higher automation levels...

- Prohibitive Maintenance of Test Scripts
- Missing or inadequate Object Recognition
- GUI and nonGUI tests require different teams/skills...
- Autom. Tests require dynamic and thorough Test Data
The Evolution of Test Automation

1st Generation

Record/Replay
FRAGILE
Example:

“…select invoice with the car registration number VIC-3456”

<table>
<thead>
<tr>
<th>Checkbox</th>
<th>Name</th>
<th>Reg Number</th>
<th>Invoice</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>John Smith</td>
<td>NSW-1234</td>
<td>90035560</td>
</tr>
<tr>
<td></td>
<td>Jim Courier</td>
<td>NSW-2345</td>
<td>90035561</td>
</tr>
<tr>
<td>✓</td>
<td>John Cook</td>
<td>VIC-3456</td>
<td>90035562</td>
</tr>
<tr>
<td></td>
<td>Tim Rehorn</td>
<td>VIC-4567</td>
<td>90035563</td>
</tr>
<tr>
<td></td>
<td>Dale Howard</td>
<td>WA-5678</td>
<td>90035564</td>
</tr>
<tr>
<td></td>
<td>Frank Rose</td>
<td>WA-6789</td>
<td>90035565</td>
</tr>
</tbody>
</table>

Does your business team understand this command?
Is this command stable?

Conclusion: Capture/Replay does not work!
The Evolution of Test Automation

Script-Based

1st Generation
Record/Replay
FRAGILE

2nd Generation
Test Automation Frameworks
EXPENSIVE

Model-Based

Next Generation
Model-Based
Robust
Cost Effective
• Cognitive Test Cases: business language - not scripts!
• Requires only business SME skills
• Highly **stable** from a technical as well as from a business point of view
• **Independent from technologies**, HTML, Java, .NET, SAP, Manual execution, …
Example:
Do you understand the **business** instruction?
Do you remember it?

**CheckBox Click**,
“/usr/cntlCONTAINER/shellcont/shell[2]/chbx[1,3]”

“…select invoice with the car registration number VIC-3456”
Model Based Test Automation
How does it work?
Model-Based Automation

System Under Test

Test Cases

Model(s)

Business Technology
de coupling

UI
non UI
OneView: UI/nonUI; Robustness over Change

Test Cases in plain English

Model SCAN & SYNCHRONIZE

Systems

© 2014 Tricentis GmbH. All rights reserved.
Consistent User Experience & On-Demand Staffing

Test Logic

Technology

Base Engine \textit{decouples}

- HTML
- Web Services
- SWIFT
- Petrol
- Diesel
- Gas

DB

© 2014 Tricentis GmbH. All rights reserved.
TOSCA allows for easy tests across **any technology** and/or platform!

---

**Technologies**
- xBrowser
- Mobile
- Java
- .NET, WPF
- Centura/Gupta
- Win32, C++, VB
- PowerBuilder, Delphi
- 3270, Emulators
- Word, XL, Outlook
- PDF-Documents
- Databases
- WebServices, XML
- REST JSON
- SAP
- Avaloq
- Siebel, Oracle

**Customer-specific extensions**

---

**Model Based Test Automation**

---

> 30 technology specific adapters
Traditional automation from a **manual** to an **automated** Test Case…

**Manual (Prose)**

**TestCase KNU01**

Login as User „alpha“ and the password „xytz12“

Set date to the first Monday following today.

Click continue

<next action>

<next action>

<next action>

Automated (Code)

```plaintext
' check all pas and aggregate actionText
With pa
  If useDefinition Then
    .loadDefinition
    omType = LCase(defName,OMTranslationName)
  If omType = OM_TABLE Then
    ' create a new table-rep to fetch actiontext
    Set paCtrl = paramAction[TABLE_IDENTIFIER]
    If paCtrl Is Nothing Then
      Set paCtrl = paramAction[FIELD_IDENTIFIER]
    End If
    If paCtrl Is Nothing Then
      ' try to use standard-text
  End If
  If useDefinition Then
```

© 2014 Tricentis GmbH. All rights reserved.
Only Tosca offers seamless test automation

Model

Traditional testing: two sets; typically out of sync!

You decide!

75%
The Bottom Line

Pre Tosca Cost

Test Case Specification

Manual Test Execution

Finding and Preparing Test Data

Execution of Test Cases

Post Tosca Cost

Risk-Based Test Structure

Test Case Design

Manual Testing

Test Automation

Total costs

100%

75%

50%

25%

0%

Source: Customer Projects Tricentis®

Poor automation coverage due to (i) maintenance trap (ii) insufficient control of test data
Thank You!