From Manual Testing to Intelligent Test Automation

• Where Are We Today?
• Evolution of Software Testing
  • From Manual Testing to Automated Test Design
• Evolution of Model Based Testing
  • 3 Generations of Test Automation with Model-Based Testing
• Artificial Intelligence in Testing
  • Intelligent Test Automation
  • The Future of Testing
Where Are We Today?

Paris, 16-18 October 2018
Most of Today’s Testing is **Still Done Manually**

What percentage of your test execution is automated?*

<table>
<thead>
<tr>
<th>Percentage</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>More than 80%</td>
<td>10%</td>
</tr>
<tr>
<td>Between 41% and 80%</td>
<td>12%</td>
</tr>
<tr>
<td>Between 20% and 40%</td>
<td>15%</td>
</tr>
<tr>
<td>Less than 20%</td>
<td>56%</td>
</tr>
<tr>
<td>We do not automate</td>
<td>7%</td>
</tr>
</tbody>
</table>

*From an Enterprise IT Poll in Europe*
Even More Dramatic: Enterprise IT Testing Market 2017

Automated Testing: $30BN
  ... of which only tools: $3BN
  ... of which only test execution: 60%

Total Size:
$150BN (14% CAGR)

Manual Testing: $120BN (!)

Sources include:
1) Dimensional Research, Testing Trends in 2017
2) Technavio Global Software Testing Market, 2017
3) Zion Market Research, Test Automation Market by Test Type, 2017
Product Development Is Moving Faster Every Day
There Is Too Much to Test & Not Enough Time

Then...
- One computer platform
- One OS
- One app
- One browser
- One release per year

Now...
- Many platforms (PC, tablet, phone, cloud, voice ...)
- Multiple OS (Win, iOS, Android, Chrome, legacy, ...)
- More connected apps
- Multiple browsers (Chrome, IE, Firefox ...)
- Multiple releases (every month or sooner)
Integrations, Complexity & Manual Are Top Challenges

Source: 8th Edition of World Quality Report 2016-17
The Net Is Testing Needs To Be Done..

FASTER
BETTER
CHEAPER
The Evolution of Software Testing
Evolution of Software Testing

- Manual Testing
  - Test Scripts
    - Capture/Replay
  - Test Models
    - Keyword Driven Testing Frameworks
  - ATD
    - Automated Test Design
- Model Based Testing (MBT)
Manual Testing

• “It is cheap”! Anybody can do it!
• It is time consuming, ineffective, and error prone
• It is still not (yet) replaceable
  • When tests are hard (or too expensive) to automate
  • Exploratory testing (with data)
Test Scripts and Capture Replay

- Automatic execution!
- Interactions have to be manually performed or coded
- But capture/replay is still driven by manual testing 😊
- Scripting requires special skills
- One small change in SUT requires re-recording or re-coding
(Keyword Driven) Testing Frameworks

- Simpler to use, understand & maintain
- Allows reduction of required skills for test specification down to Excel!
- Keywords are reusable & tests become retargetable!
- Still requires (highly) skilled automation team for scripting keywords

Example:

```
Login(username, password)
StartVehicle()
```
Test Models (MBT)

- Generally graphical representation of control flows instead of coding
- Easier to adopt & review than scripts
- Can be generated from manual test descriptions or test plans
- Allows generation of (partial) scripts and documentation
- Still user is responsible for test design
Automated Test Design (MBT)

• Specify **functionality to be tested** and generate optimal test coverage
• Generally graphical representation of control and data aspects
• Basically full script & documentation generation from generated tests
• Now manual testers can effectively contribute to test automation

Don’t model tests; model the functionality to be tested!

When you model the functionality to be tested you can create a model capable of representing every test for that functionality. It is a much smaller task, since the number of all possible tests is very large and change behind your back, which means test models will become obsolete the moment they are made.

(adopted from Monica Anderson “Don’t model the World; Model the Mind”)
About Model-Based Testing (MBT)

- **Umbrella term** for using models in a testing context; only one approach is to use MBT for *automating test design*
  - MBT *complements* test execution
  - Recognized by worldwide industrial standards (ETSI, ISTQB)
The Evolution of Model-Based Testing
Evolution of Model Based Testing
Automated Test Design: Planning & Searching

Constructs tests from the explored part of the state space by selecting paths that leads to testing goals.

Test generation algorithm searches the state space for testing goals, defined by the end user.

Model behavior implies typically infinite state space.
Automated Test Design: Reasoning & Problem Solving

- Test generation reasons about unspecified input data and derives a concrete data based on modeled logic
- Generally, constraint solving theory is applied for computing unspecified input data

Models interact with an unspecified environment via inputs and outputs

Models describe the expected external behavior of the system under test
Automated Test Design: Finding Important Tests

- Predict the ways the system may fail
- Combine large set of heuristics
  - Equivalence classes
  - Boundary value analysis
  - Combinatorial testing
  - Mutation testing
  - And more...
- Know what do not cover!
MBT 1.0: Modeling Notation is Adoption Hurdle

From functional programming (10+ years ago) ...

... via some graphical & a lot of general purpose programming (~7 yrs ago) ...
Today We Use Domain Specific Modeling Languages

- Designed specifically to use tester’s language, concepts & terminology
- Easy to use: drag & drop building blocks to compose functionality
- Enabling direct automation
- Flexible and extensible
- See also UCAAT 2013 tutorial
MBT 2.0: Enabling Test Process Automation

- MBT is not just about test generation!
- MBT integrates & connects with requirement & test management, versioning control, and test automation/CI integration
MBT 2.0: Still Every New Project Starts Like This ...
MBT 3.0: “Eliminating” Modeling

Complete End to End Test Process Automation with MBT
Test Reverse Engineering

- Readers normalize test input format
- Keywords are identified from test steps using Natural Language Processing (NLP)
- Redundant tests and subtests are eliminated
- Tests are converted into flows and merged
[ALM] Test Plans: Excel as Input to MBT

Import & Refine

Generate Tests

Execute!
“BDD”: Gherkin as Input to MBT

**Feature:** Shopping with esd.conformiq.com

**Scenario:** Successful shopping

Given a user has logged in successfully
When user adds 1 item(s) of "CQ0003"
Then shopping basket contains 1 item(s)

**Scenario:** Bad product id

Given a user has logged in successfully
When user adds 3 item(s) of "123"
Then application displays invalid entry

**Scenario:** Remove item

Given a user has logged in successfully
When user adds 1 item(s) of “CQ0003”
Then shopping basket contains 1 item(s)
When user clicks Remove all button for item "CQ0003"
Then shopping basket contains 0 item(s)
MBT Trends in 2017: Modeling in Prose

• Low end MBT tools are pushing very simple, prose based modeling notations into the test tool market
  • Arguably domain specific modeling still has the edge as it enables a direct path to automatic test execution (today not possible from prose)

• Users have started to perceive domain specific modeling (= creation of building blocks) as a burden when it comes to modeling
  • Model generation from test assets (reverse engineering) has addressed this issue but not removed it

• Frequent use of extension points (“custom actions”) is diluting the value proposition of domain specific languages
  • Prose and domain specific action modeling start to mix
Artificial Intelligence in Testing
AI Techniques Have Been Leveraged for (Many) Years!

- Fuzzy and Probabilistic Logic
- Classification
- Search & Optimization
- Expert Systems
- (Deep) Machine Learning (ML)

Today the term “AI” is generally reduced to (Deep) Machine Learning ...
More Recent: AI in Robustness Testing

- App store testing example: Fuzz vs search vs man+machine
- Initially, “break the system by clicking around” (= “Monkey Testing”)
- Now also combination of online MBT with learnings from previous failures
- No human required for test execution!

Source: Stuart Reid, “Smarter Testing with Artificial Intelligence”, JFTL 2017
More Recent: AI in Result Analysis of Large Test Suites

- Predictive analytics: what *could* fail?
  - “Test Less - Test Right” (M. Venkata)
  - Failure patterns, code changes, etc
- Diagnostic analytics: what is failing?
  - Data mining & visualization of defects
  - Application monitoring for anomalies
- Prescriptive analytics: what to fix?
  - Symptom vs defect
  - Root cause analysis
How Could We Use Learning in Functional Testing?

• Test automation tools have managed successfully to make creation of tests that are automatically executable more or less “dead easy”

• BUT ... automating from “arbitrary” text still has to be done manually

Manually refine into domain specific model inputs & outputs

Manually refine into concrete automation steps

Enter user name and password and click the Sign In button
Introducing Intelligent Test Automation

- From manual to automatically executable test in two steps
  1. Feed manual tests and train intelligent test automation (if any), e.g., resolve unknown text to automation rules
  2. Execute and fix unrecognized object identifiers (if any)
- The more intelligent test automation is trained - the less it needs to learn!
Intelligent Test Automation in a Nutshell

- Reader(s) normalize input format(s)
- Natural Language Processing is used to analyze test steps
- User “teaches” mappings to target test automation tool or framework
- Works with any input format, different target technologies, and any test automation framework!

Note: Conformiq patents pending
Intelligent Test Automation Naturally Extends to MBT!

- **Requirements**: Includes BAs & Testers
- **Test Design**: Includes optimization
- **Test Creation**: Includes scripts, data & validations
- **Test Execution**: Includes harnesses

Test Files (fully generated)

Custom Libraries

Exec Tool(s)

Intelligent Test Automation
The Start of a New Generation in Test Automation?
Evolution of Software Testing

- Manual Testing
- Test Scripts Capture/Replay
- Keyword Driven Testing Frameworks
- Test Models
- Automated Test Design (ATD)
- Model Based Testing (MBT)
Thank You for Your Attention!

Q&A

stephan.schulz@conformiq.com