Accelerated Agile Testing

Harnessing the Power of Exploratory Testing

CAMUG, March 6th, 2014
Christin Wiedemann, PQA
Outline

• Agile Testing
• The Objective of Testing
• Exploratory Testing
• Time-Boxed Test Sessions
• Reporting and Metrics
• Agile Testing Workflow
• Christin Wiedemann
  ‣ Regional VP/Chief Scientist
  ‣ PQA Vancouver
  ‣ Ph.D. in Physics in 2007 (Stockholm University, Sweden)
  ‣ 2 years as a developer before I found my true vocation – testing
  ‣ Moved to Canada in 2011
Testing on Agile Projects

- Iterative
- Continuous changes
- Fast pace
- Testing early

- Team effort
- Less focus on artifacts

http://gardenofeaden.blogspot.ca
Cheap, Good, Fast

- You can get it *fast* and *cheap*, but you will not get high quality
- Reducing cost and time typically means increasing *risk*
- There are always constraints
- You can only pick two attributes

```
Quality

Time (schedule)  Cost (resources)
```

©2014 Professional Quality Assurance Ltd. (PQA) -- http://www.pqatesting.com
Objective of Testing

- In testing, we:
  - Execute tests and observe how the software responds
  - Record and store test results

- However, the goal of testing is not the execution or the artifacts, but what we learn about the product

- *Software testing is a quest for information and knowledge*

- Knowledge must be efficiently communicated to stakeholders

- Knowledge allows for informed decision making
Risk-Driven Testing

- Complete test coverage is impossible
  ‣ We must prioritize
- We can never prove that software is working – only find proof of ways in which it fails
- A test process should help us understand and control the risk of product failure
- In risk-driven testing, we:
  ‣ Select tests that we think are the most likely to expose serious problems
  ‣ Skip tests that we think are unlikely to expose problems, or likely to expose problems that have very little impact
- Risk-driven testing is an approach – not a technique
Risk-Driven Testing

- Risk-driven testing:
  - Organize testing to reduce the level of *product* risk when shipping software
  - Doesn’t deal with *project* risk

- Risks are used to:
  - Select test approach/technique
  - Design tests by considering risks at an early stage and letting risks steer the design
  - Prioritize tests during execution

- Risk analysis is an important part of test planning
Exploratory Testing

• What is Exploratory Testing?
  ‣ A software testing approach
  ‣ Simultaneous learning, test design and test execution
  ‣ Introduced by Cem Kaner in 1983

• Why the name Exploratory Testing?
  ‣ To distinguish it from ad hoc testing
  ‣ To emphasize the exploration
• What is Exploratory Testing not?
  ‣ Ad-hoc testing
  ‣ Sloppy testing
  ‣ Careless testing
  ‣ Unstructured testing
  ‣ Undocumented testing
  ‣ Unskilled testing
• Why Exploratory Testing?
  ‣ Less preparation
  ‣ Do not need complete specifications or requirements
  ‣ Bugs found quickly
  ‣ Adaptable and flexible
  ‣ Creative, fun and stimulating
Scripted Versus Exploratory Testing

- Testing requires specialized skills
- Which takes more time?
  - Maintaining test suites
  - Training testers
- Testers need to be aware and observant
  - http://youtu.be/ubNF9QNEQLA
  - http://youtu.be/Ahg6qcgoay4

Transport for London
http://www.tfl.gov.uk
How Much Detail?

- How much detail is too much?
- Focus on business objects – not attributes
- Use preconditions

"The invoice status is approved"

```
business object
attribute
```

"The invoice is approved"

```
business object
```
Don’t confuse testing and checking!

Checking:
- Confirming something we believe to be true
- Confirm, verify, validate
- Pass/fail

Testing:
- Exploring with the motivation of finding new information
- Exploration, investigation, discovery, learning
Testing Versus Checking

- Checking:
  - Suitable for automation
  - Detailed test cases/scripts

- Testing:
  - Should be done by humans
  - High-level guidelines rather than detailed steps
  - Use-case level
Figure adapted from original by Jon Bach
Session-Based Test Management (SBTM)

- **What SBTM is:**
  - Tool-supported testing approach
  - Introduced by Jonathan Bach and James Bach in 2000
  - Structured and documented exploratory testing

- **Why use it:**
  - Management control
  - Metrics reporting
  - Accountability
  - Documentation
  - Rapid defect discovery
  - Flexibility
Session-Based Test Management (SBTM)

- Work in sessions
- Time-box
- Uninterrupted
- Reviewable
- Feedback (debriefing)

- Test charter: Mission for the session
- Session report
Test Charter

• Test Charter:
  ‣ Mission for the session
  ‣ How to test
  ‣ What kind of problems to look for
  ‣ Often created in advance
  ‣ Extent and level of detail flexible

• Structure:
  ‣ Risk
  ‣ Coverage
  ‣ Time frame

Credit: Michael D. Kelly
Session Report

- Session Report:
  - Date, Time & Tester
  - Test Charter
  - Function Area
  - Time break-down:
    - Test design
    - Test execution
    - Test reporting
    - Other (e.g. interruptions and setup)
  - Bugs found
  - Issues found
  - Opportunity
  - Notes
• Estimate number of charters, time for each charter is given

<table>
<thead>
<tr>
<th>Charters planned</th>
<th>78</th>
</tr>
</thead>
<tbody>
<tr>
<td>Charters run</td>
<td>12</td>
</tr>
<tr>
<td>New charters added</td>
<td>5</td>
</tr>
</tbody>
</table>

(78 + 5) - 12 = 71 charters left to run
71 * 90 min = 106 hours

<table>
<thead>
<tr>
<th>Charter</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Charter #1</td>
<td>John</td>
</tr>
<tr>
<td>Charter #2</td>
<td>Lisa</td>
</tr>
<tr>
<td>Charter #3</td>
<td>Anne</td>
</tr>
</tbody>
</table>

Charter assigned to
Blocked
Run
Waiting
Reporting

- Distinguish between reporting on *status* and *quality*
- Know your audience
- Beware of misunderstandings and misinterpretation
• *What* are you trying to measure
• *Why* are you trying to measure it
• *When* are you measuring it
• Qualitative or quantitative

• Example – defects found
  ‣ How is the metric affected by the measurement?
  ‣ Does “defects found” accurately reflect:
    • Status
    • Quality

• Many metrics tend to reflect learning rather than status/quality
SBTM Metrics

- Metrics
  - Bugs found
  - Issues found
  - On-charter vs opportunity
  - Session vs non-session work
  - Number of sessions over time

Time break-down

Test Sessions
Coverage

• Measuring coverage of manual testing:
  ‣ Test cases
  ‣ Test charters/sessions

<table>
<thead>
<tr>
<th>Function Area</th>
<th>Charters</th>
<th>Risk</th>
</tr>
</thead>
<tbody>
<tr>
<td>Area 1</td>
<td>12</td>
<td>1</td>
</tr>
<tr>
<td>Area 2</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>Area 3</td>
<td>7</td>
<td>4</td>
</tr>
</tbody>
</table>
• Each iteration adds new functionality
• You want to run same *checks* over and over again (regression test)
• Hard if not impossible to do for humans
• Humans test
Traditional Testing Workflow

- Waterfall
  - Consecutive phases

- Agile
  - All phases repeated in every iteration
• Planning
  ‣ Test ideas in mind map
  ‣ Function areas and/or test techniques
  ‣ Group test ideas into sessions (optional)
  ‣ Estimate number of charters needed (optional)
Agile/Exploratory Testing Workflow

Planning ➔ Design ➔ Execution ➔ Reporting

<table>
<thead>
<tr>
<th>Function Area</th>
<th>Charters</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>Installation</td>
<td>Upgrade</td>
<td>Medium</td>
</tr>
<tr>
<td>Installation</td>
<td>New installation</td>
<td>High</td>
</tr>
<tr>
<td>File export</td>
<td>File validation</td>
<td>Low</td>
</tr>
</tbody>
</table>
Agile/Exploratory Testing Workflow

- Planning
- Design
- Execution
- Reporting

- Design, Execution, Reporting
  - Update mind map
  - Session reports (optional)
  - Update test charters (optional)
  - Add test ideas
  - Create additional test charters (optional)
Agile/Exploratory Testing Workflow

Planning | Design | Execution | Reporting

- File name
- File header
- File compression
- File validation

1. File Export
2. Installation
3. Stress Testing
4. Domain Testing

AUT

- New installation
- Upgrade

- File size
- Processing time

- Decimals
- Special characters

Bug ID: 31415
File containing "&" cannot be processed.

Press 'F4' to edit in Notes View

©2014 Professional Quality Assurance Ltd. (PQA) -- http://www.pqatesting.com
Do We Fix All bugs?

- Do we always have to fix all bugs?
  - Which bugs can be left in the software?
  - Why would we want to leave bugs unfixed?
  - What are the risks of fixing bugs?
Do We Fix All Bugs?

- **Not fixing** a bug is associated with a cost:
  - Customer dissatisfaction and potentially lost revenue
  - Cost of releasing patches
  - Providing support

- **Fixing** a bug is associated with a cost:
  - Development time to make fix
  - Test time to retest bug
  - Test time to regression test
  - Slower time to market
  - Resources tied up

- **Postponing** a bug is associated with same costs as fixing a bug, as well as:
  - Additional administration time
Can You Be Too Safe?

- Are we being over-cautious?
- Can we be too safe?
- How can we tell if we are?
Contact Information

- Christin Wiedemann
- Christin.Wiedemann@pqatesting.com
- http://www.pqatesting.com
- http://christintesting.wordpress.com
- @c_wiedemann
Links - Mindmapping

- **XMind**: Powerful tool with a lot of nice features, used in examples
  - [http://www.xmind.net](http://www.xmind.net)

- **mindmeister**: Collaborative tool.
  - [http://www.mindmeister.com](http://www.mindmeister.com)

- **FreeMind**: The simpler of the mind mapping tools, but still very useful.
• Rapid Reporter: A note taking tool for exploratory testing sessions.
  › http://testing.gershon.info/reporter/

• Session Tester: A tool for recording and managing exploratory testing session.
  › http://session-tester.software.informer.com/

• SBTestExecute: A tool that produces summary reports and calculates metrics from an Excel session report template.
  › http://www.addq.se/test-och-kvalitetssakring/produkter/sbtexecute (scroll down to bottom of page for English)