Focus Your Testing
Using the Agile Testing Matrix

Agile Edmonton
February 4, 2009

Janet Gregory
DragonFire Inc.
Agenda

- A Bit About Testing
- Introduction to the Agile Testing Quadrants
  - Quadrant 1
  - Quadrant 2
  - Quadrant 3
  - Quadrant 4
- Planning your Testing Strategy
- References
Why Do We Test?

- To find bugs
- Make sure system is reliable
- To learn about the application
- See if UI is usable
- Feedback to future stories
- Check for doneness
- Manage technical debt
  - deferred work
  - hacks, untidy work
  - Slows the team down
Traditional Testing

Phased / Gated Projects
- Done after coding is finished
- Works from requirements documents
- Can be a fight to be involved up front
- Organizational culture – silos
- Communicates through DTS
- Few face to face conversations
- Different language than developers
Agile Testing

- **Principles / Values**
  - Communication, collaboration, interactions
- **Teams are test infected**
- **Testers are part of the team**
- **Whole team responsible for quality**
- **Tester's role**
  - Help uncover hidden assumptions
  - Provide feedback
  - Elicit and clarify requirements
  - Drive development with examples
Agile Means a Change in Mindset

Focus on the 'why' we test
Change the way we think
- not about the when
- not about the how (ex. white box, black box)

Instead of
- We’re here to break the software!

Think
- What can we do to help deliver the software successfully?
The Agile Testing Quadrants

- Can be used to ensure we accomplish all goals
  - Q1 - Technology-facing tests that support the team
  - Q2 - Business-facing tests that support the team
  - Q3 - Business-facing tests that critique the product
  - Q4 - Technology-facing tests that critique the product
The Quadrants ..... 

- Can be used as a communication tool
  - To the project team
  - To management
  - To explain testing in a common language

- Emphasize whole-team responsibility
  - Focus on collaboration
  - Whole team participation
Use to define ‘Doneness’

Defining “doneness” for release readiness

- No story is done until tested
- Customer requirements captured as passing tests
- Automated regression tests
- Delivers value
- “Doneness” in all quadrants
The Agile Testing Quadrants

Original idea by Brian Marick, www.exampler.com
Quadrant 1

Agile Testing Quadrants


Supporting the Team

Functional Tests
Examples
Story Tests
Prototypes
Simulations

Q2

Q3

Critique Product

Unit Tests
Component Tests

Q1

Q4

Automated

Technology Facing

Tools

Exploratory Testing
Scenarios
Usability Testing
UAT (User Acceptance Testing)
Alpha / Beta

Performance & Load Testing
Security Testing
“ility” Testing
Q1 Kinds of Tests

- **Unit Tests**
  - Tests developer intent - program design
  - Tests a small piece of code
  - Makes sure it does what it should

- **Component Tests**
  - Tests architect intent – system design
  - Tests that components work together correctly
Reasons / Benefits for Q1 Testing

- Focus on internal code quality
- Go faster, do more
  - Unit tests provides safety net and refactoring support
- Builds quality in
- Provides instant feedback
- TDD increases confidence in design
- Forms the foundation of automation suite
- Builds testability into code
Quadrant 2

Agile Testing Quadrants

Automated & Manual
Supporting the Team
Automated
Functional Tests
Examples
Story Tests
Prototypes
Simulations

Business Facing
Q2
Unit Tests
Component Tests

Q3
Exploratory Testing
Scenarios
Usability Testing
UAT (User Acceptance Testing)
Alpha / Beta

Q1
Performance & Load Testing
Security Testing
"ility" Testing

Q4

Critique Product
Tools

Copyright 2009 Janet Gregory, DragonFire
Q2: Kinds of Tests

- Elicit requirements
- SDD or ATDD (Acceptance Test Driven Dev)
  - allows developers to code until the tests pass
  - Fit / Fitnesse
- Examples
- User Experience
  - wire frames
  - Mock-ups / prototypes
- Pair Testing

Business-facing Tests that Support the Team
Reasons / Benefits for Q2 Testing

- Drive development with business-facing tests
- Obtain enough requirements to start coding
- Help customers achieve advance clarity
- Capture examples, express as executable tests
- Focus is external quality
- Know when we're done
- Customer – developer – tester collaboration
## Toolkit – Turning Examples into Tests

Set-up: Create a variety of users that start with the first initial, have the same name, similar email address, and unique ids.

### Query users - return all: no criteria set (sort by login id)

<table>
<thead>
<tr>
<th>query users with login id</th>
<th>first name</th>
<th>last name</th>
<th>email address</th>
<th>phone number</th>
<th>roles</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Administrator</td>
<td>Administrator</td>
<td><a href="mailto:id_ad@dragonfire.xxx">id_ad@dragonfire.xxx</a></td>
<td>403-974-4957</td>
<td>Employee, Operations, Production Support, Quality Assurance</td>
</tr>
<tr>
<td>Bill</td>
<td>Bia</td>
<td>bibbo</td>
<td><a href="mailto:bill_bia@dragonfire.xxx">bill_bia@dragonfire.xxx</a></td>
<td></td>
<td>Employee</td>
</tr>
<tr>
<td>Darien</td>
<td>Filpot</td>
<td>Bibbo</td>
<td><a href="mailto:filpot_bibbo@dragonfire.xxx">filpot_bibbo@dragonfire.xxx</a></td>
<td></td>
<td>Quality Assurance</td>
</tr>
<tr>
<td>Frank</td>
<td>Bill</td>
<td>Frank</td>
<td><a href="mailto:frank_bill@dragonfire.xxx">frank_bill@dragonfire.xxx</a></td>
<td></td>
<td>Quality Assurance</td>
</tr>
<tr>
<td>Jane</td>
<td>Roberts</td>
<td>Jane</td>
<td><a href="mailto:jane_roberts@dragonfire.xxx">jane_roberts@dragonfire.xxx</a></td>
<td></td>
<td>Operations</td>
</tr>
<tr>
<td>Juno</td>
<td>Will</td>
<td>June</td>
<td><a href="mailto:junebug@dragonfire.xxx">junebug@dragonfire.xxx</a></td>
<td></td>
<td>Employee</td>
</tr>
<tr>
<td>Juno</td>
<td>Williams</td>
<td>Junebug</td>
<td><a href="mailto:junebug@dragonfire.xxx">junebug@dragonfire.xxx</a></td>
<td></td>
<td>Quality Assurance</td>
</tr>
</tbody>
</table>

### Query users based on first name

<table>
<thead>
<tr>
<th>query users with login id</th>
<th>first name</th>
<th>last name</th>
<th>J</th>
<th>email address</th>
<th>phone number</th>
<th>roles</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jane</td>
<td>Roberts</td>
<td>Jane</td>
<td>J</td>
<td><a href="mailto:jane_roberts@dragonfire.xxx">jane_roberts@dragonfire.xxx</a></td>
<td></td>
<td>Operations</td>
</tr>
<tr>
<td>Juno</td>
<td>Will</td>
<td>June</td>
<td>J</td>
<td><a href="mailto:junebug@dragonfire.xxx">junebug@dragonfire.xxx</a></td>
<td></td>
<td>Quality Assurance</td>
</tr>
<tr>
<td>Juno</td>
<td>Williams</td>
<td>Junebug</td>
<td>J</td>
<td><a href="mailto:junebug@dragonfire.xxx">junebug@dragonfire.xxx</a></td>
<td></td>
<td>Quality Assurance</td>
</tr>
</tbody>
</table>

### Query users based on last name

<table>
<thead>
<tr>
<th>query users with login id</th>
<th>first name</th>
<th>last name</th>
<th>B</th>
<th>email address</th>
<th>phone number</th>
<th>roles</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bill</td>
<td>Bia</td>
<td>Bibbo</td>
<td>B</td>
<td><a href="mailto:bill_bia@dragonfire.xxx">bill_bia@dragonfire.xxx</a></td>
<td></td>
<td>Employee</td>
</tr>
<tr>
<td>Frank</td>
<td>Bill</td>
<td>Frank</td>
<td>B</td>
<td><a href="mailto:frank_bill@dragonfire.xxx">frank_bill@dragonfire.xxx</a></td>
<td></td>
<td>Quality Assurance</td>
</tr>
</tbody>
</table>

### Query users based on email

<table>
<thead>
<tr>
<th>query users with login id</th>
<th>first name</th>
<th>last name</th>
<th>email address</th>
<th>phone number</th>
<th>roles</th>
</tr>
</thead>
<tbody>
<tr>
<td>Darien</td>
<td>Filpot</td>
<td>Bibbo</td>
<td><a href="mailto:filpot_bibbo@dragonfire.xxx">filpot_bibbo@dragonfire.xxx</a></td>
<td></td>
<td>Quality Assurance</td>
</tr>
</tbody>
</table>
Set-up: Create a variety of users that start with the first initial, have the same name, similar email address, and unique ids.

<table>
<thead>
<tr>
<th>create user with login id</th>
<th>first name</th>
<th>last name</th>
<th>email address</th>
<th>phone number</th>
<th>roles</th>
</tr>
</thead>
<tbody>
<tr>
<td>frankb</td>
<td>Frank</td>
<td>Billian</td>
<td><a href="mailto:frankb@dragonfire.xx">frankb@dragonfire.xx</a></td>
<td></td>
<td>Quality Assurance</td>
</tr>
<tr>
<td>junebug</td>
<td>Juno</td>
<td>Williams</td>
<td><a href="mailto:junebug@dragonfire.xx">junebug@dragonfire.xx</a></td>
<td></td>
<td>Quality Assurance</td>
</tr>
<tr>
<td>fillpot</td>
<td>Darien</td>
<td>Fillpot</td>
<td><a href="mailto:fillpot_darien@dragonfire.xx">fillpot_darien@dragonfire.xx</a></td>
<td></td>
<td>Quality Assurance</td>
</tr>
<tr>
<td>jonero</td>
<td>June</td>
<td>Roberts</td>
<td><a href="mailto:June.Roberts@dragonfire.xx">June.Roberts@dragonfire.xx</a></td>
<td></td>
<td>Operations</td>
</tr>
<tr>
<td>billboa</td>
<td>Bill</td>
<td>Bia</td>
<td><a href="mailto:bill_bia@dragonfire.xx">bill_bia@dragonfire.xx</a></td>
<td></td>
<td>Employee</td>
</tr>
<tr>
<td>junebug1</td>
<td>Juno</td>
<td>Will</td>
<td><a href="mailto:junebug1@dragonfire.xx">junebug1@dragonfire.xx</a></td>
<td></td>
<td>Quality Assurance</td>
</tr>
</tbody>
</table>

Query users - return all: no criteria set (sort by login id)

<table>
<thead>
<tr>
<th>first name</th>
<th>last name</th>
<th>login id</th>
<th>email address</th>
<th>phone number</th>
<th>roles</th>
</tr>
</thead>
<tbody>
<tr>
<td>Idi</td>
<td>Administrator</td>
<td>administrator</td>
<td>id <a href="mailto:administer@ngx.com">administer@ngx.com</a></td>
<td>403-974 4957</td>
<td>Employee, Operations, Production Support, Quality Assurance</td>
</tr>
<tr>
<td>Bill</td>
<td>Bia</td>
<td>billboa</td>
<td><a href="mailto:bill_bia@dragonfire.xx">bill_bia@dragonfire.xx</a></td>
<td></td>
<td>Employee</td>
</tr>
<tr>
<td>Darien</td>
<td>Fillpot</td>
<td>fillpot</td>
<td><a href="mailto:fillpot_darien@dragonfire.xx">fillpot_darien@dragonfire.xx</a></td>
<td></td>
<td>Quality Assurance</td>
</tr>
<tr>
<td>Frank</td>
<td>Billian</td>
<td>frankb</td>
<td><a href="mailto:frankb@dragonfire.xx">frankb@dragonfire.xx</a></td>
<td></td>
<td>Quality Assurance</td>
</tr>
<tr>
<td>Jonero</td>
<td>Roberts</td>
<td>jonero</td>
<td><a href="mailto:June.Roberts@dragonfire.xx">June.Roberts@dragonfire.xx</a></td>
<td></td>
<td>Operations</td>
</tr>
<tr>
<td>Juno</td>
<td>Williams</td>
<td>junebug</td>
<td><a href="mailto:junebug@dragonfire.xx">junebug@dragonfire.xx</a></td>
<td></td>
<td>Employee</td>
</tr>
<tr>
<td>Juno</td>
<td>Williams</td>
<td>junebug</td>
<td><a href="mailto:junebug@dragonfire.xx">junebug@dragonfire.xx</a></td>
<td></td>
<td>Quality Assurance</td>
</tr>
</tbody>
</table>

Query users based on first name

<table>
<thead>
<tr>
<th>first name</th>
<th>last name</th>
<th>login id</th>
<th>email address</th>
<th>phone number</th>
<th>roles</th>
</tr>
</thead>
<tbody>
<tr>
<td>June</td>
<td>Roberts</td>
<td>jonero</td>
<td><a href="mailto:June.Roberts@dragonfire.xx">June.Roberts@dragonfire.xx</a></td>
<td></td>
<td>Operations</td>
</tr>
<tr>
<td>Juno</td>
<td>Williams</td>
<td>junebug</td>
<td><a href="mailto:junebug@dragonfire.xx">junebug@dragonfire.xx</a></td>
<td></td>
<td>Employee</td>
</tr>
<tr>
<td>Juno</td>
<td>Williams</td>
<td>junebug</td>
<td><a href="mailto:junebug@dragonfire.xx">junebug@dragonfire.xx</a></td>
<td></td>
<td>Quality Assurance</td>
</tr>
</tbody>
</table>

Query users based on last name

<table>
<thead>
<tr>
<th>first name</th>
<th>last name</th>
<th>login id</th>
<th>email address</th>
<th>phone number</th>
<th>roles</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bill</td>
<td>Bia</td>
<td>billboa</td>
<td><a href="mailto:bill_bia@dragonfire.xx">bill_bia@dragonfire.xx</a></td>
<td></td>
<td>Employee</td>
</tr>
<tr>
<td>Frank</td>
<td>Billian</td>
<td>frankb</td>
<td><a href="mailto:frankb@dragonfire.xx">frankb@dragonfire.xx</a></td>
<td></td>
<td>Quality Assurance</td>
</tr>
</tbody>
</table>

Query users based on email

<table>
<thead>
<tr>
<th>first name</th>
<th>last name</th>
<th>login id</th>
<th>email address</th>
<th>phone number</th>
<th>roles</th>
</tr>
</thead>
<tbody>
<tr>
<td>Darien</td>
<td>Fillpot</td>
<td>fillpot</td>
<td><a href="mailto:fillpot_darien@dragonfire.xx">fillpot_darien@dragonfire.xx</a></td>
<td></td>
<td>Quality Assurance</td>
</tr>
</tbody>
</table>
Quadrant 3

Agile Testing Quadrants

Automated & Manual

Business Facing

Manual

Exploratory Testing
Scenarios
Usability Testing
UAT (User Acceptance Testing)
Alpha / Beta

Supported by the Team

Q2

Q3

Q4

Q1

Supporting the Team

Automated

Technology Facing

Critique Product

Tools

Functional Tests
Examples
Story Tests
Prototypes
Simulations

Unit Tests
ComponentTests

Performance & Load Testing
Security Testing
“ility” Testing

Copyright 2009 Janet Gregory, DragonFire
Q3: Kinds of Testing

- **Exploratory Testing**
  - feedback into stories
  - work with customers to understand what you think
  - work with developers if you have questions

- **Test for Usability**
  - understand end users - personas
  - who will be using the system

- **Pair test with customers**

- **User Acceptance Testing**
Q3: Collaborative Testing

- Provide feedback ....
  - Discuss with technical, customer team
  - Turn what you learn into tests that drive new features
  - Change process as needed

- Iteration reviews
  - Builds confidence
  - Quick feedback loop

- Informal demos
  - Pair exploratory testing with customer
  - Even on unfinished code

Business-facing Tests that Critique the Product
Reasons / Benefits for Q3 Testing

- Feedback to Quadrants One and Two
- Evaluate the product
- Recreate actual user experiences
- Realistic use
- Learn as you test
- Context
  - What works for your situation
  - “It depends”
  - A tool, not a rule
- Constructive

Business-facing Tests that Critique the Product
Quadrant 4
Q4: Kinds of Tests

- Nonfunctional or parafunctional tests
- “ility” testing
- Performance, scalability, stress, load
- Memory management
- Security testing
  - Roles & permissions
  - System ‘hacking’
- Data migration
- Infrastructure Testing
- Recovery

Technology-facing Tests that Critique the Product
Reasons / Benefits for Q4 Testing

- “Non-functional” requirements may be higher priority than “functional”
  - Performance
  - Stability
  - Security
- Specialized expertise might be needed
  - Collaborate to transfer skills
- Makes the ‘finished’ product
- Does your application deliver the ‘right’ value

Technology-facing Tests that Critique the Product
The Agile Testing Quadrants

Original idea by Brian Marick, www.exampler.com
Planning Your Test Strategy

- Consider scope, priorities, risks
- Tools that solve the problem
- Involve customers
- Collaborate with programmers
- Consider all four quadrants
- Test matrix - big picture
  - allows whole team to understand
- Document only what is useful
- Use lessons learned to improve
Questions?
Available Now!

Agile Testing: A Practical Guide for Testers and Agile Teams

By Janet Gregory and Lisa Crispin

Available on

• Amazon.com
• Amazon.ca

www.agiletester.ca

www.janetgregory.ca
Resources

- www.exampler.com
- Collaboration Explained: Jean Tabaka
- Testing Extreme Programming, By Lisa Crispin and Tip House
- Fearless Change: Patterns for introducing new ideas, Linda Rising and Mary Lynn Manns
- agile-testing@yahoogroups.com