Agile Test Planning with the Agile Testing Quadrants

ADP Testing Workshop 2009

Lisa Crispin

With Material from Janet Gregory

and Brian Marick's Agile Testing Matrix

Introduction

- Me: Coding, testing
- Joined first agile team in 2000
  - Tester's place in agile unclear!
- Many years on agile teams developing web applications in Java and .Net
- Help agile teams/testers

Goals - Takeaways

- When you leave, you'll know how to use the agile testing quadrants to:
  - ❑ Identify the types of testing needed
  - ❑ Identify who should do each type, and when
  - ❑ How best to accomplish each type
  - ❑ Where to start

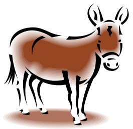
Goals

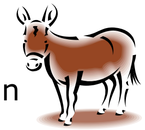
How about you?

- What areas of testing does your team need to improve?

Test Planning Includes:

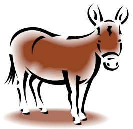
- Unit testing/TDD
- Continuous Integration
- Getting correct requirements
- ATDD, functional testing
- Test automation
- Non-functional testing
  - Performance, load, reliability, stability
  - Usability, security, other “ilities”
- Exploratory testing, tours
- …?

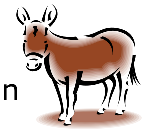




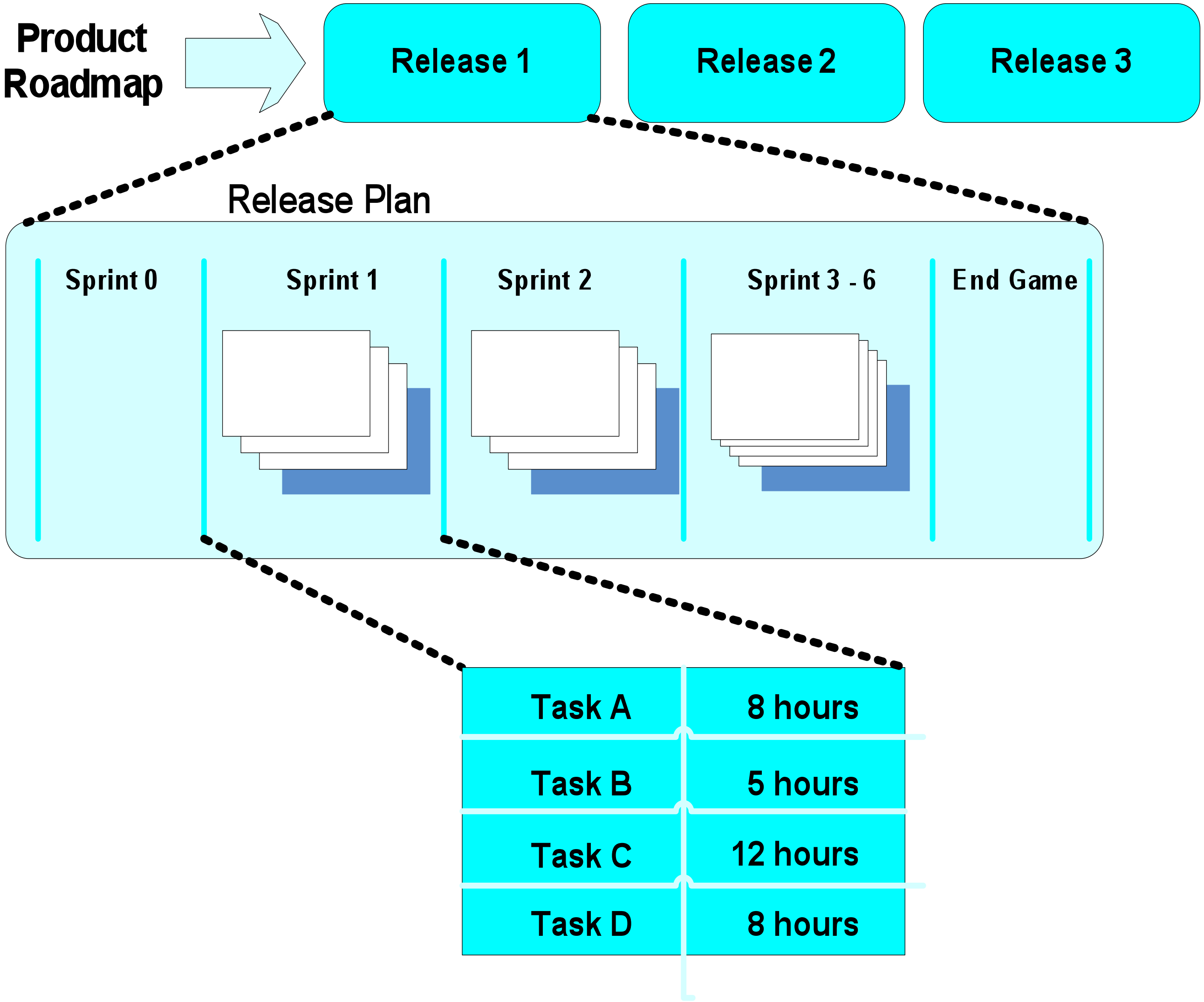
Agenda

- Overview of Quadrants
  - Purpose of testing
- Quadrant 1:
  - Technology-facing tests that support the team
- Quadrant 2:
  - Business-facing tests that support the team
- Quadrant 3:
  - Business-facing tests that critique the product
- Quadrant 4:
  - Technology-facing tests that critique the product
- Planning your strategy

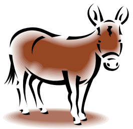


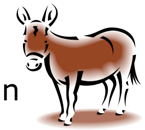


Levels of Planning – Product, Release, Iteration

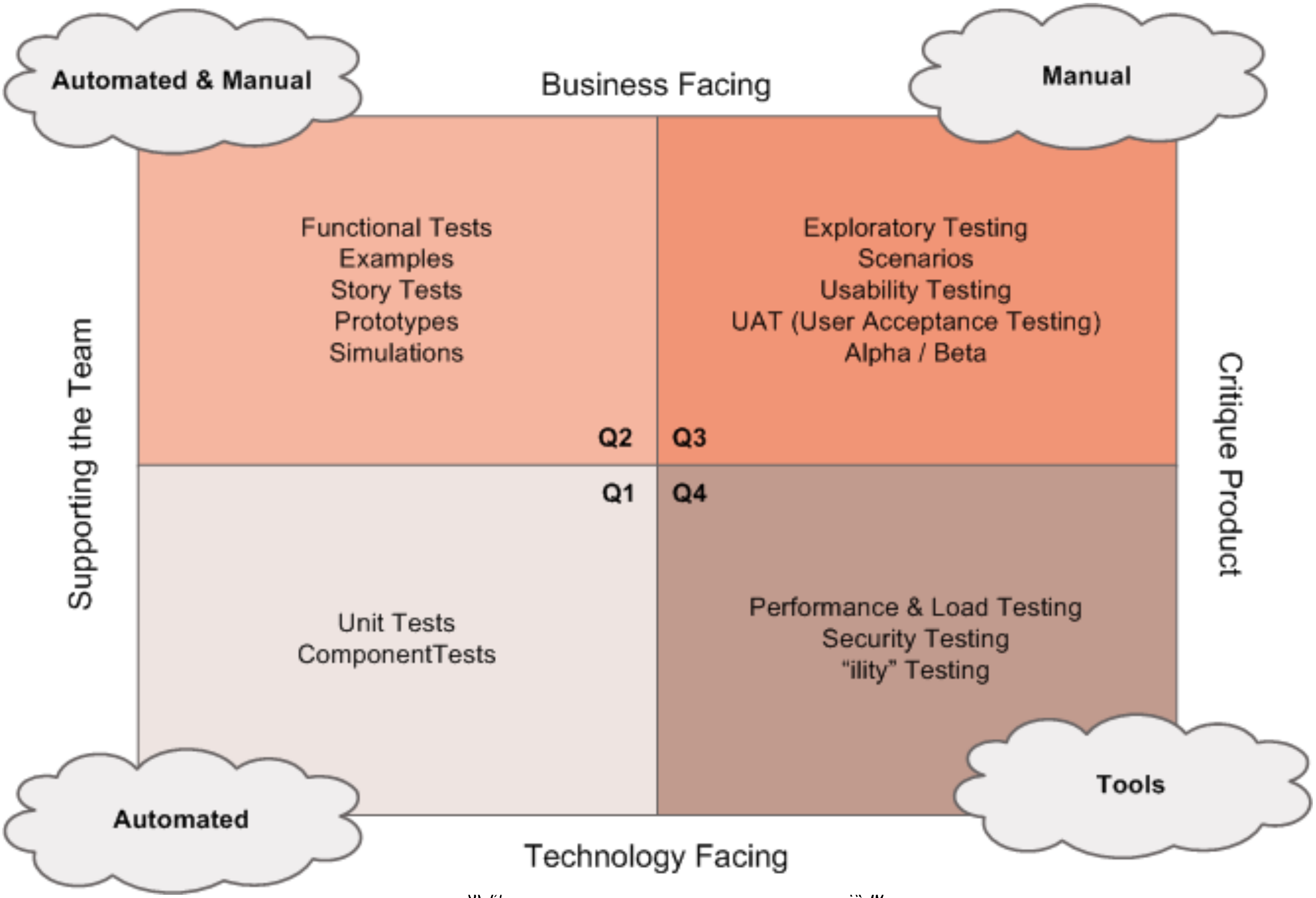


Task A	8 hours
Task B	5 hours
Task C	12 hours
Task D	8 hours





The Agile Testing Quadrants



Automated & Manual

Business Facing

Manual

Supporting the Team

Critique Product

Q2

Functional Tests
Examples
Story Tests
Prototypes
Simulations

Q3

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

Q1

Unit Tests
ComponentTests

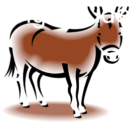
Q4

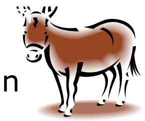
Performance & Load Testing
Security Testing
“ility” Testing

Automated

Tools

Technology Facing



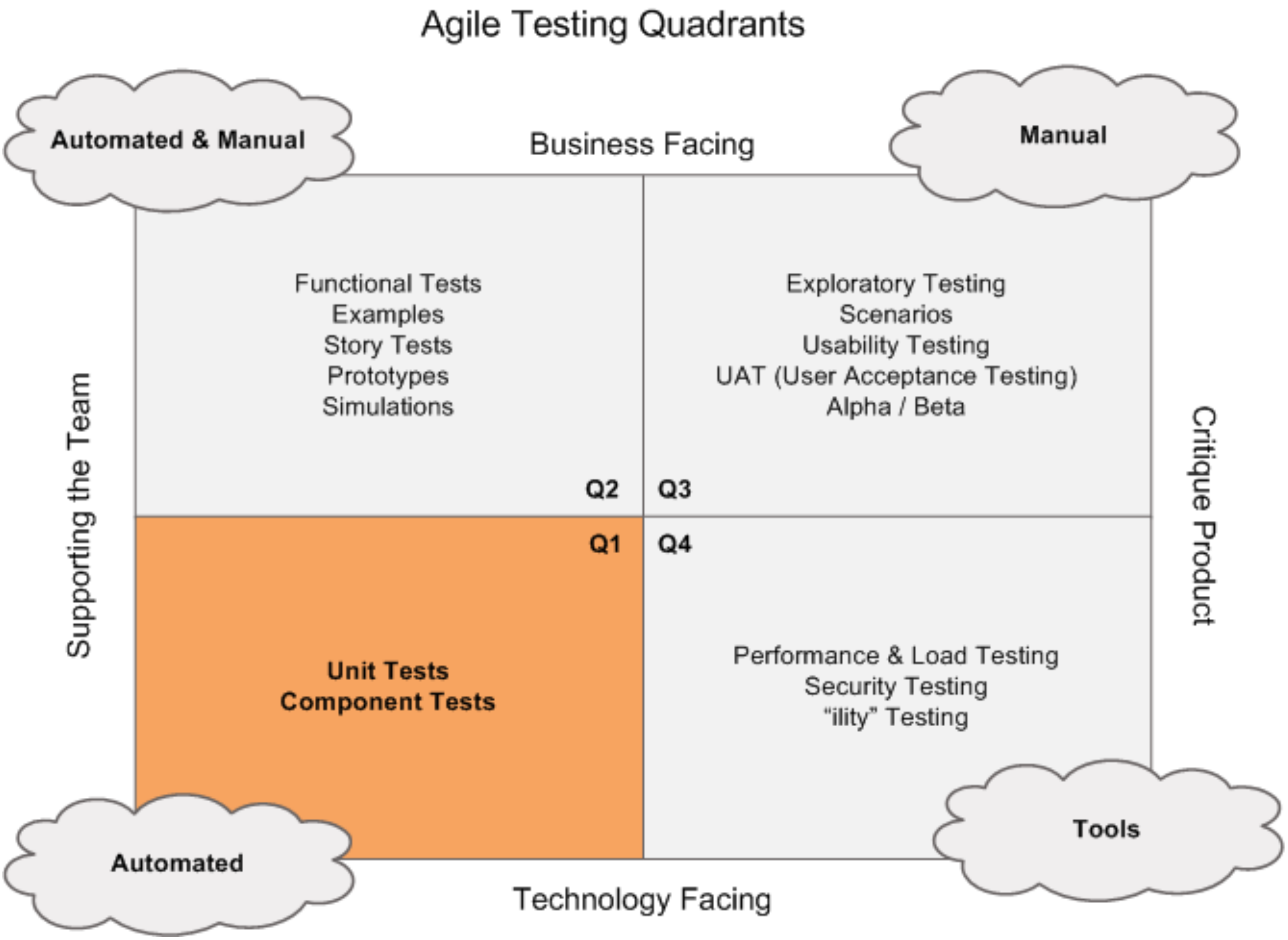


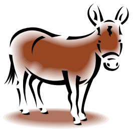
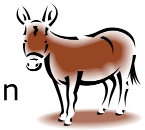
Using the Quadrants

- Quadrants help ensure we accomplish all goals
  - Support team
  - Critique product
  - Ensure business needs met
  - Ensure technological needs met

- Shared responsibility
  - Special skills may be needed
  - Focus on collaboration

Quadrant 1

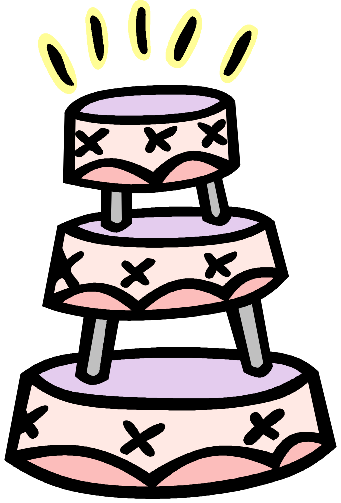



Copyright 2009: Lisa Crispin


Goal of Quadrant One Tests

Testability

- Layered or “componentized”
  - APIs, Ports and Adapters
- Test database access, updates
- Business logic and presentation separated
- Isolate tests
  - allows isolating problems
- Internal quality
- Infrastructure

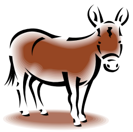


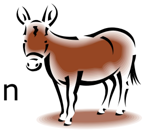
Quadrant OneTest Benefits

- Go faster, do more
  - Unit tests provide safety net
  - Refactoring support
    - Improve design & maintainability without changing functionality
- Quality, not speed, is goal
- Courage
- Confidence in design

What, Who, When

- Unit Tests
  - Developer intent – program design
  - Small piece of code does what it should
- Component Tests
  - Architect intent – system design
  - Components work together correctly
- Programmer tests/codes
- Continually refactor
- Run in CI



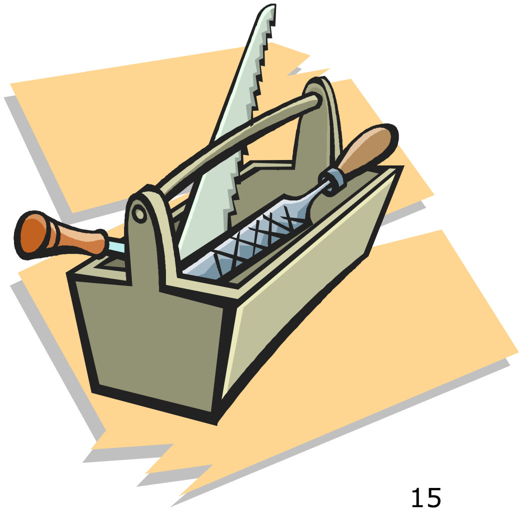


If Your Team Doesn't Do These …

- It's a team problem
- Find areas of greatest pain
- Testers writing unit tests isn't the answer
- Managers must provide time to learn
- Without Quadrant One,
    - the other quadrants will be much harder

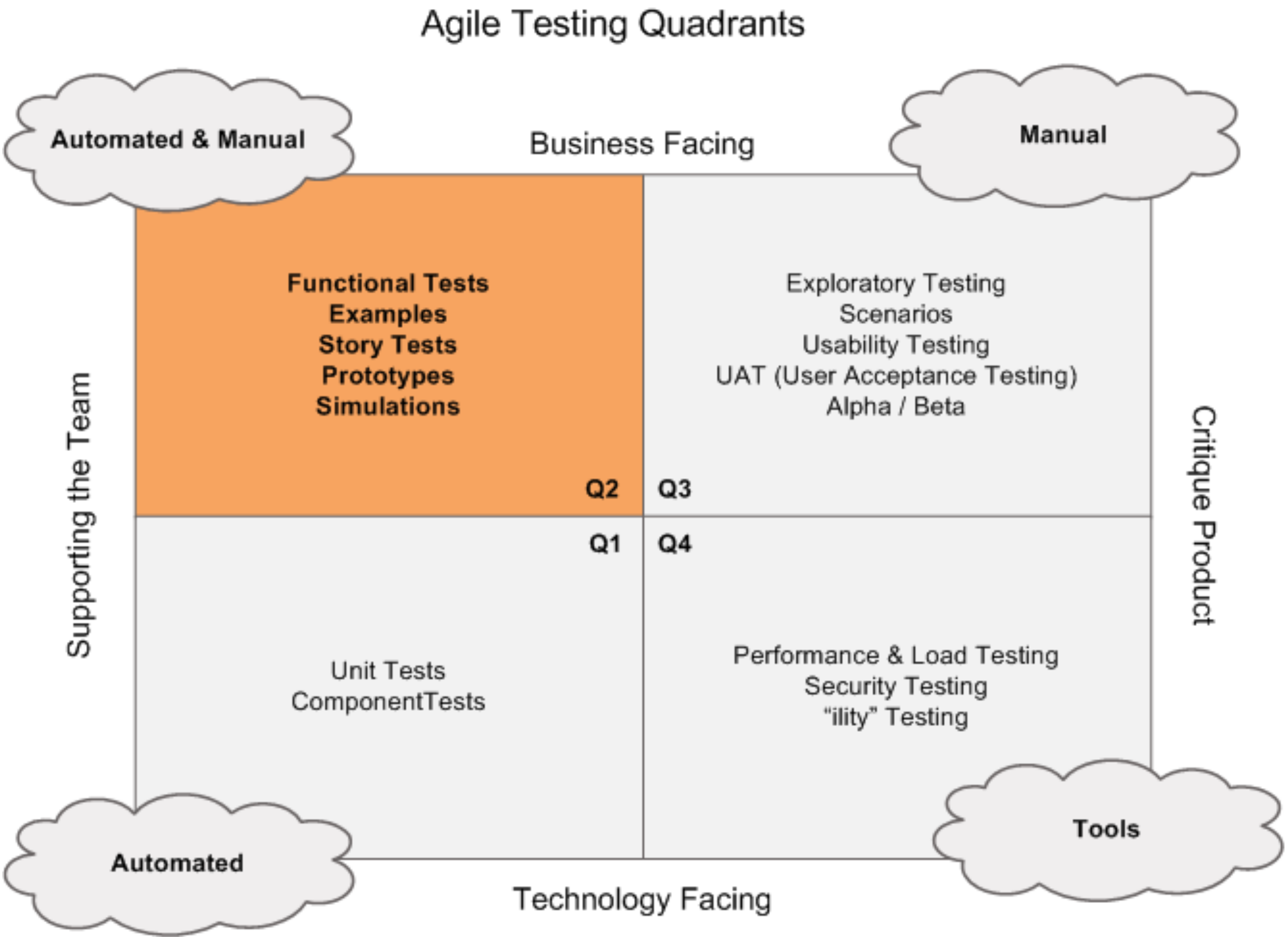
Quadrant One Toolkit

- Source code management
  - Version control
  - Know what has been changed, by whom
  - Be able to restore earlier version
- Integrated development environment
  - compile, debug, build GUI, refactor
  - eg. Eclipse, IntelliJ Idea, NetBeans
- Build/CI tools
  - eg. CruiseControl, Hudson. TeamCity
- Unit test tools
  - xUnit
  - Mocking tools



Questions?

Quadrant 2



Agile Testing Quadrants

Business Facing

Technology Facing

Supporting the Team

Critique Product

Automated & Manual

Manual

Automated

Tools

Q2

Functional Tests
Examples
Story Tests
Prototypes
Simulations

Q3

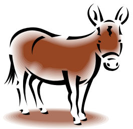
Exploratory Testing
Scenarios
Usability Testing
UAT (User Acceptance Testing)
Alpha / Beta

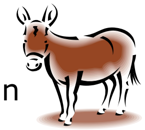
Q1

Unit Tests
ComponentTests

Q4

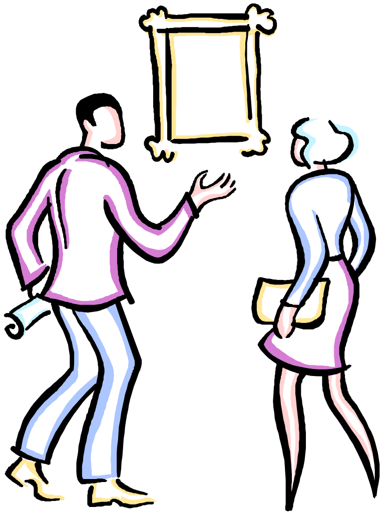
Performance & Load Testing
Security Testing
"ility" Testing





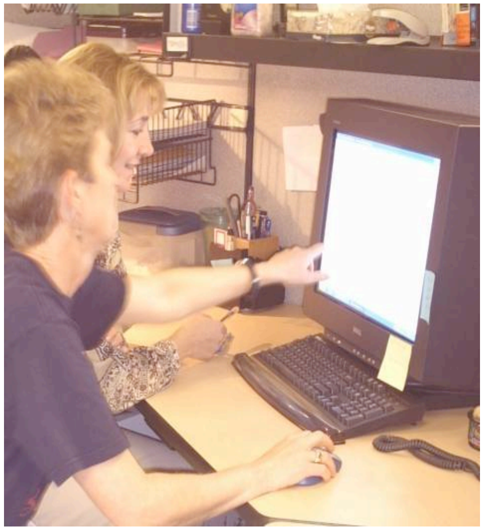
Purpose of Quadrant Two

- Drive development with business-facing tests
- Ask the right questions
- Help customers achieve advance clarity
- Capture examples, express as executable tests
- External quality
- Know when we're done



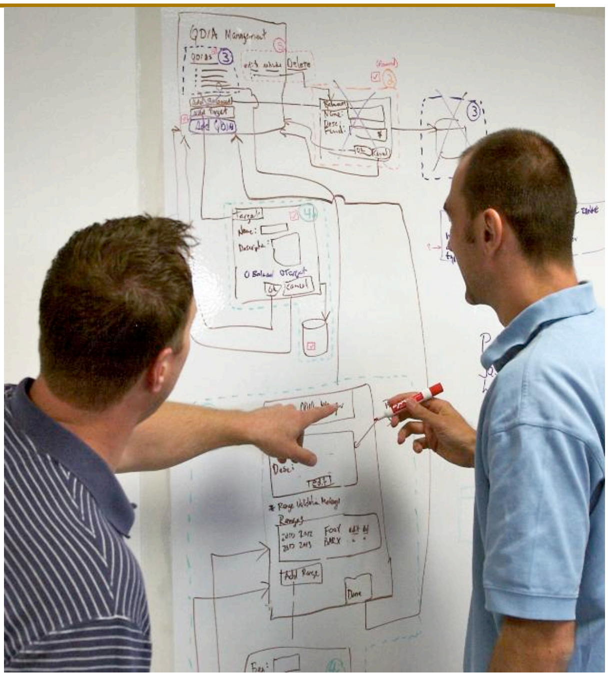
Who Does Quadrant 2 Tests, When?

- Testers have special expertise
- Collaboration with customers



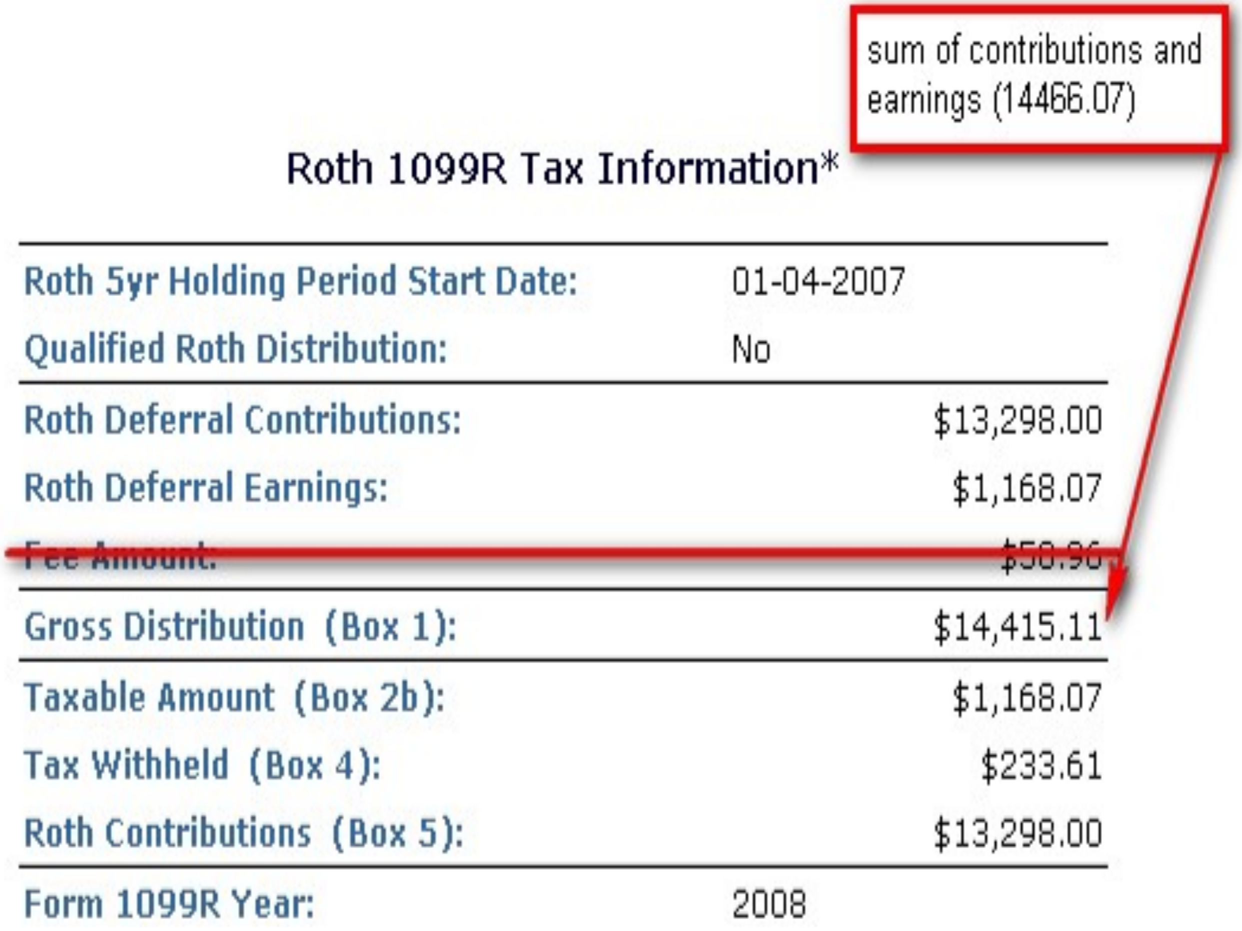
- Team responsibility
  - Programmers
  - DBAs, analysts, ...
- Start of iteration
  - Business-facing tests drive development
- Throughout iteration
  - No story done until tested

Toolkit – Eliciting Requirements



- Checklists
- Mind maps
  - brainstorming
  - words, ideas, tasks
- Mockups / paper prototypes
  - User-centered design
- Flow diagrams
- Whiteboards (physical and virtual)
- Thin slice/steel thread exercise
- Behavior-driven development

Mock-Up Example

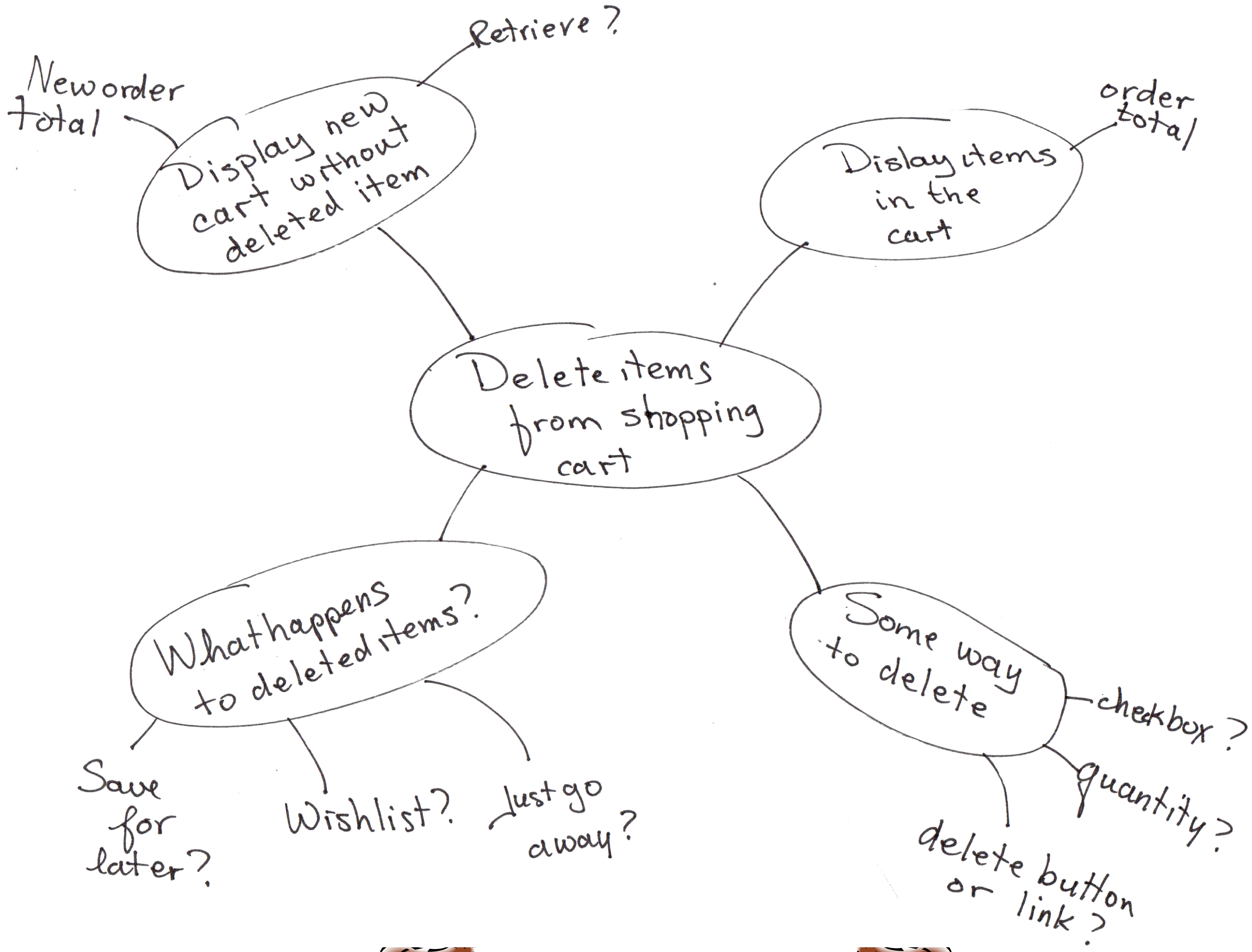


sum of contributions and earnings (14466.07)

Roth 1099R Tax Information*

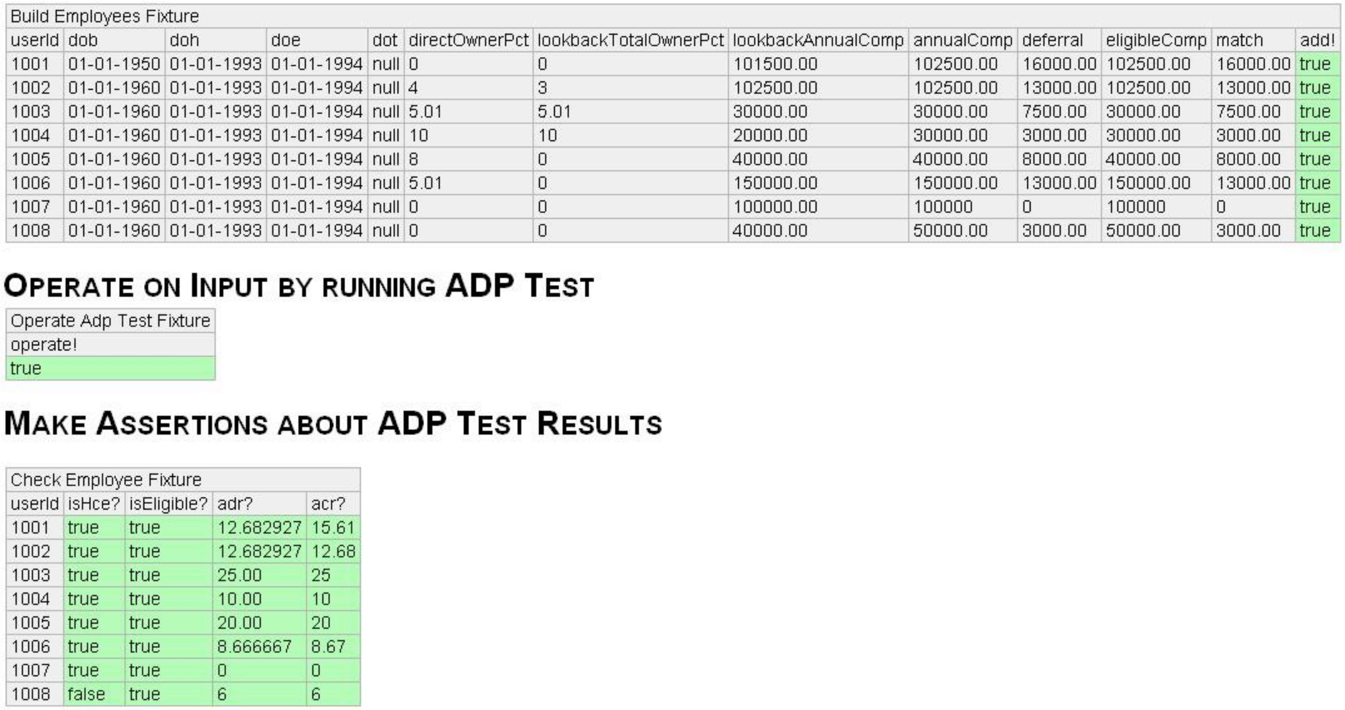
Roth 5yr Holding Period Start Date:	01-04-2007
Qualified Roth Distribution:	No
Roth Deferral Contributions:	$13,298.00
Roth Deferral Earnings:	$1,168.07
~~Fee Amount:~~	~~$50.96~~
Gross Distribution (Box 1):	$14,415.11
Taxable Amount (Box 2b):	$1,168.07
Tax Withheld (Box 4):	$233.61
Roth Contributions (Box 5):	$13,298.00
Form 1099R Year:	2008

Mind Map Example



Toolkit – Turning Examples into Tests

- Fit/FitNesse
  - collaboration in software development
  - Takes place of regular UI



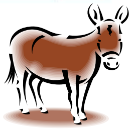
Build Employees Fixture												
userId	dob	doh	doe	dot	directOwnerPct	lookbackTotalOwnerPct	lookbackAnnualComp	annualComp	deferral	eligibleComp	match	add!
1001	01-01-1950	01-01-1993	01-01-1994	null	0	0	101500.00	102500.00	16000.00	102500.00	16000.00	true
1002	01-01-1960	01-01-1993	01-01-1994	null	4	3	102500.00	102500.00	13000.00	102500.00	13000.00	true
1003	01-01-1960	01-01-1993	01-01-1994	null	5.01	5.01	30000.00	30000.00	7500.00	30000.00	7500.00	true
1004	01-01-1960	01-01-1993	01-01-1994	null	10	10	20000.00	30000.00	3000.00	30000.00	3000.00	true
1005	01-01-1960	01-01-1993	01-01-1994	null	8	0	40000.00	40000.00	8000.00	40000.00	8000.00	true
1006	01-01-1960	01-01-1993	01-01-1994	null	5.01	0	150000.00	150000.00	13000.00	150000.00	13000.00	true
1007	01-01-1960	01-01-1993	01-01-1994	null	0	0	100000.00	100000	0	100000	0	true
1008	01-01-1960	01-01-1993	01-01-1994	null	0	0	40000.00	50000.00	3000.00	50000.00	3000.00	true

Operate on Input by running ADP Test

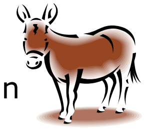
Operate Adp Test Fixture
operate!
true

Make Assertions about ADP Test Results

Check Employee Fixture				
userId	isHce?	isEligible?	adr?	acr?
1001	true	true	12.682927	15.61
1002	true	true	12.682927	12.68
1003	true	true	25.00	25
1004	true	true	10.00	10
1005	true	true	20.00	20
1006	true	true	8.666667	8.67
1007	true	true	0	0
1008	false	true	6	6

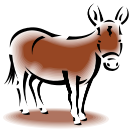


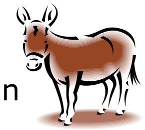
Copyright 2009: Lisa Crispin



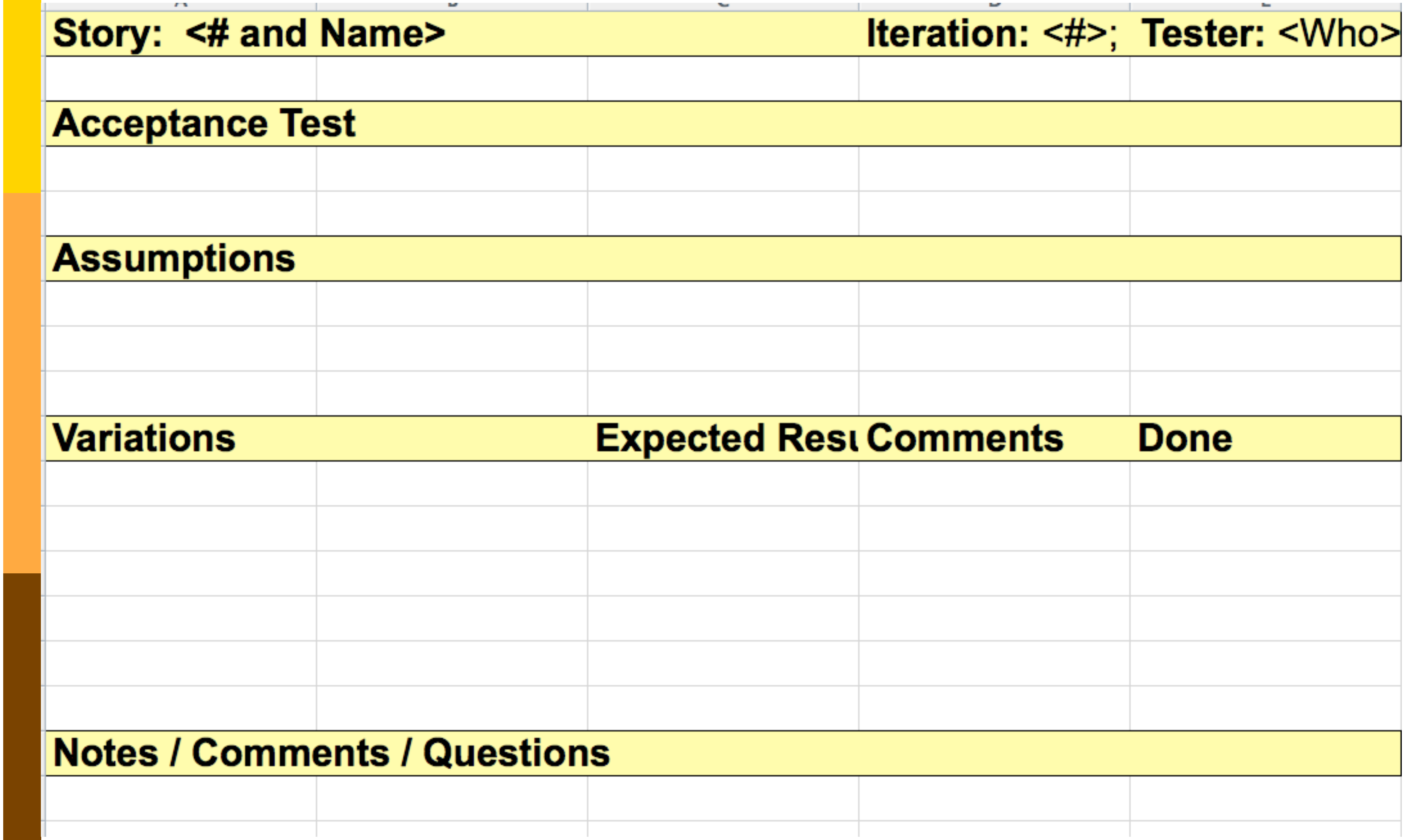
More Tools to Turn Examples into Tests

- BDD frameworks
  - Cucumber, easyB, nbehave, rspec
- GUI test tools/libraries/frameworks
  - Selenium
  - Watir/Watin/Watij, Cucumber, Rasta, Taza
  - Canoo WebTest
  - Robot Framework
  - SWAT
  - QTP





Sample Story test Template



Story: <# and Name>			**Iteration: <#>;**	**Tester: <Who>**
Acceptance Test				
Assumptions				
Variations		**Expected Resu**	**Comments**	**Done**
Notes / Comments / Questions				

Questions?

Exercise

Story: As an Agile Testing Toys shopper, I want the ability to delete items from my shopping cart, so I don't buy items I don't want.

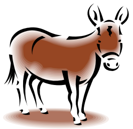
Additional information:

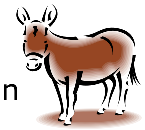
- The business isn't picky about how to implement this: change quantity to 0, click a checkbox, click a button – whatever is easiest to implement and intuitive to the shopper.

In small groups, discuss

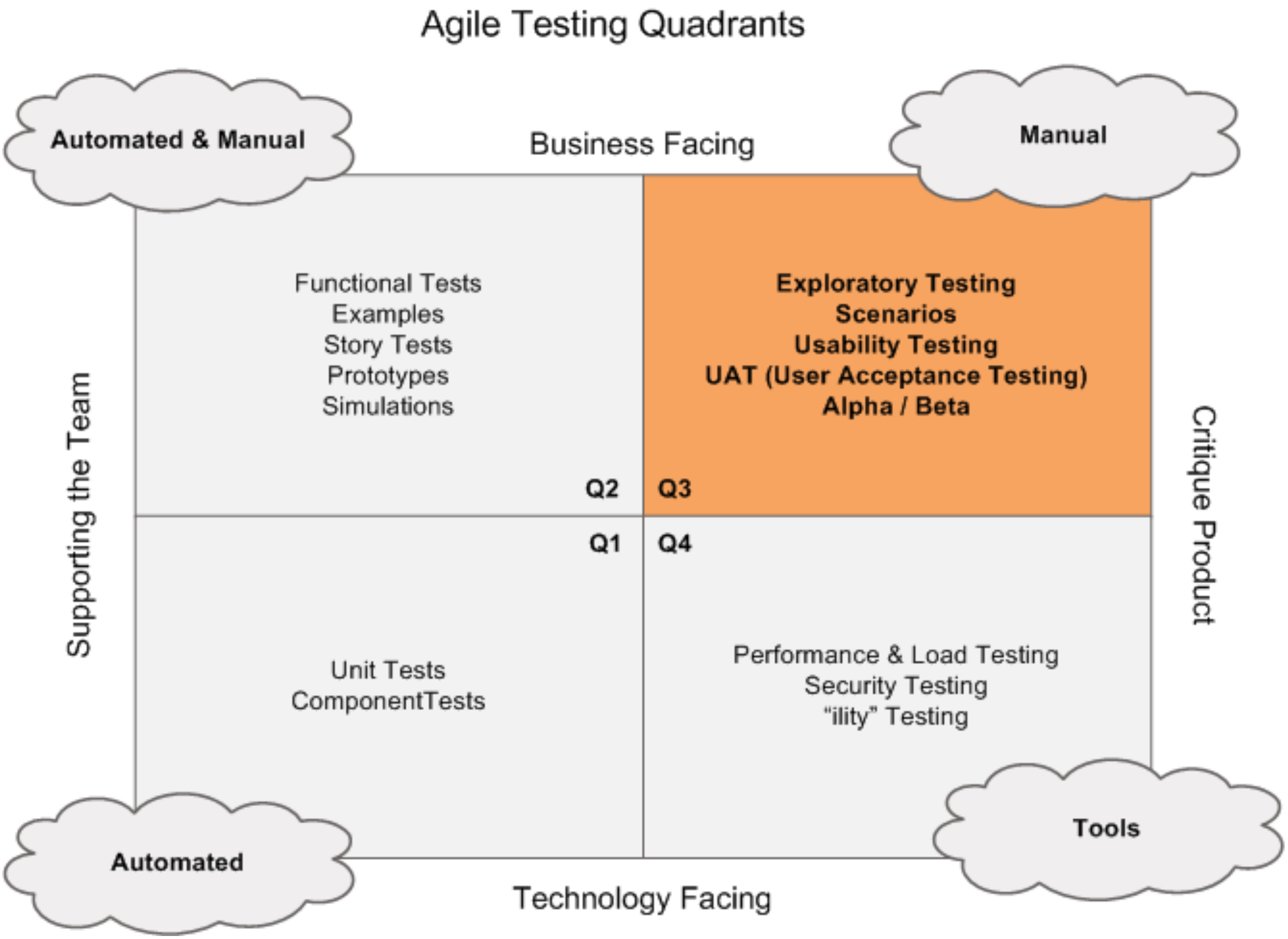
1. What would an acceptance test(s) look like?

2. What variations could you give to the developers?





Quadrant 3



Agile Testing Quadrants

Automated & Manual

Business Facing

Manual

Supporting the Team

Critique Product

Functional Tests
Examples
Story Tests
Prototypes
Simulations

Q2

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

Q3

Q1

Unit Tests
ComponentTests

Q4

Performance & Load Testing
Security Testing
"ility" Testing

Automated

Tools

Technology Facing

Evaluating 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

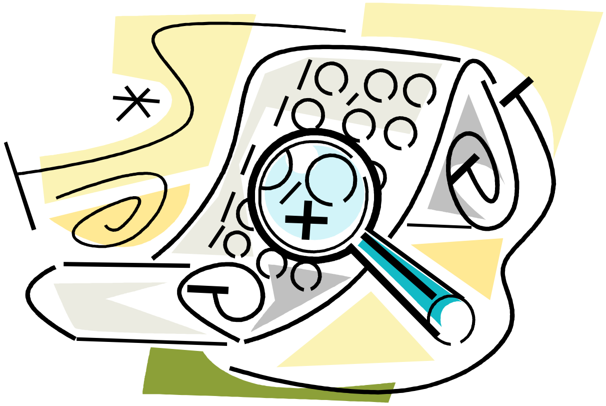
Copyright 2009: Lisa Crispin

Demos with Customers

- Iteration reviews
  - Builds confidence
  - Quick feedback loop
- Informal demos
  - Pair exploratory testing with customer
  - Even on unfinished code

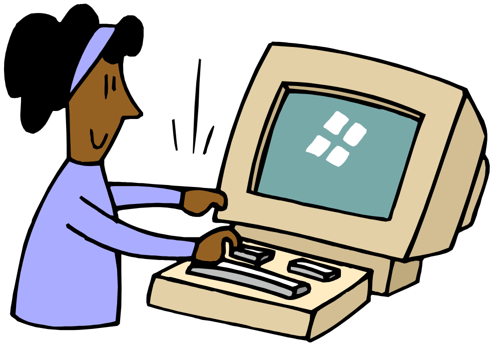
Exploratory Testing

- Simultaneous learning, test design, test execution [source: James Bach]
  - “Doing” reveals more than “thinking”
- Careful observation
- Critical thinking
- Diverse ideas
- Rich resources
  - Tools, data, people [source: Jon Hagar]



Other Types of Testing

- Scenario testing
  - Process flows
  - Realistic data
  - Soap opera testing (Hans Buwalda)
- Usability testing
  - Personas
  - Navigation
  - Observing users
- Don't forget documents, reports, help text



Behind the GUI

- API testing
  - Inputs and outputs
  - Sequence of API calls
  - Checking log files
  - Example: Test parsing of upload file
  - Example: Test shipping cost calculation
  - States and transitions
- Web Services
  - External customers
  - Levels of service
  - Validate definitions against profiles
  - Validate requests and responses

Feedback to Tests that Support Team

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

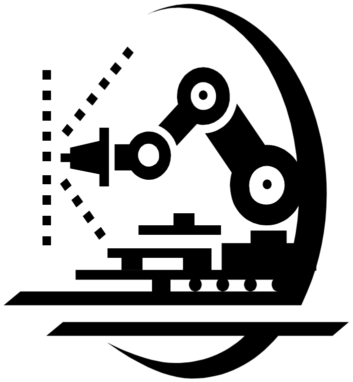
Who Does Quadrant 3 Tests, When

- Requires good skills, experience, intuition, critical thinking
- Involve the customers
- Programmers help with tools to facilitate
- Do as early as possible

Quadrant Three Toolkit

Tool Strategy

- Consider who uses tests, who writes and maintains tests
- Quadrant 2 tools may apply
- Take time to research, experiment



Tools for Exploratory Testing

- Test scenario setup
  - eg. Watir/Watij scripts
- Generate test data
  - eg. PerlClip, Ruby script
- Simulators
  - Simulate data, feed to app over time
- Monitors
  - Watch log files
- Emulators
  - Duplicate system behavior
    - eg. mobile devices

Questions?

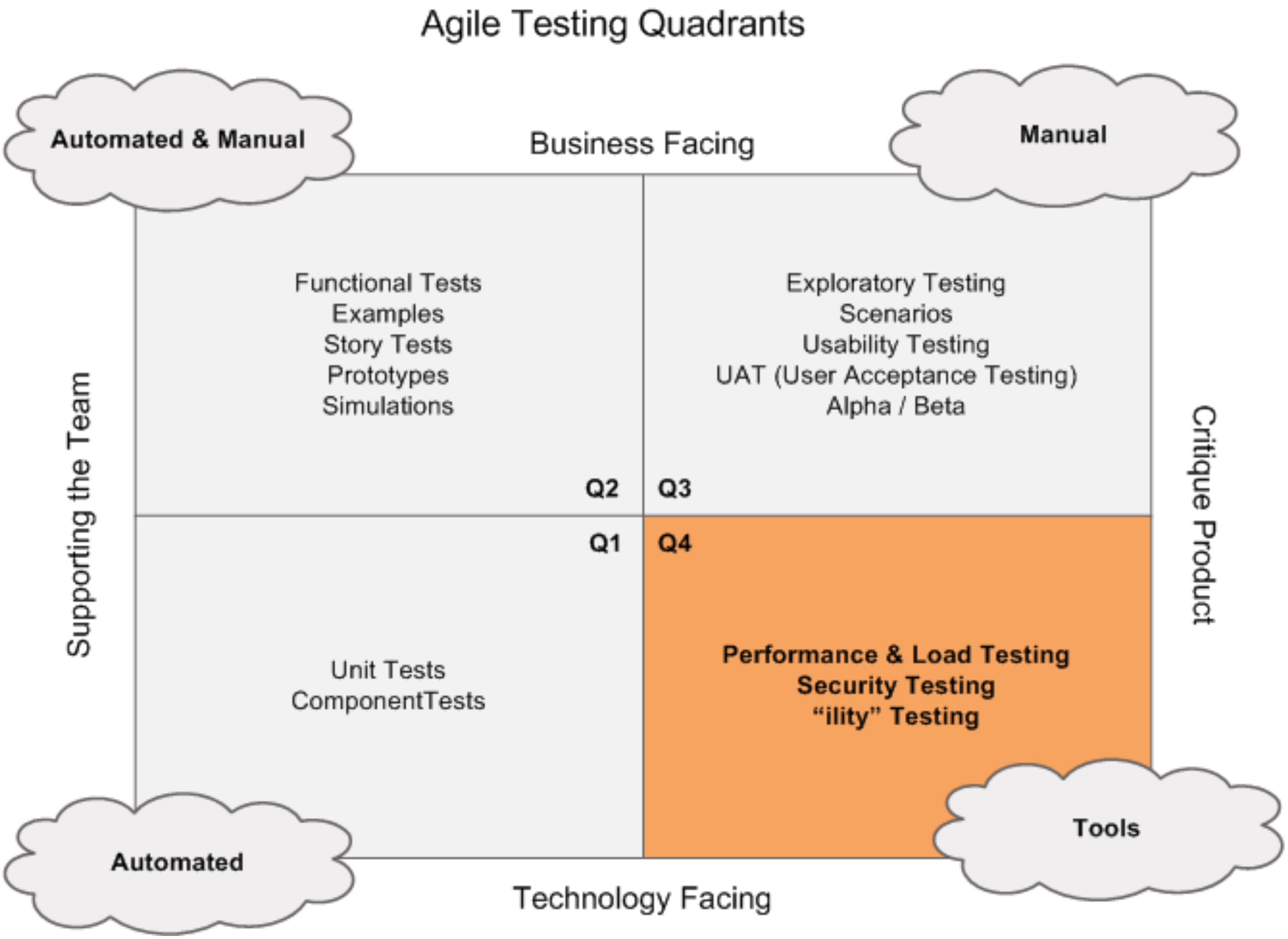
Discussion

Story: As an Agile Testing Toys shopper, I want the ability to delete items from my shopping cart, so I don't buy items I don't want.

What types of Quadrant 3 tests you think will be needed to test this story. Who might do each test?

What types of Quadrant 3 testing happen on your team?

Quadrant 4



Agile Testing Quadrants

Automated & Manual

Business Facing

Manual

Supporting the Team

Critique Product

Q2

Functional Tests
Examples
Story Tests
Prototypes
Simulations

Q3

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

Q1

Unit Tests
ComponentTests

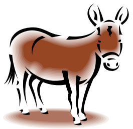
Q4

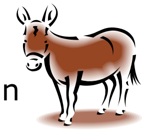
Performance & Load Testing
Security Testing
"ility" Testing

Automated

Tools

Technology Facing



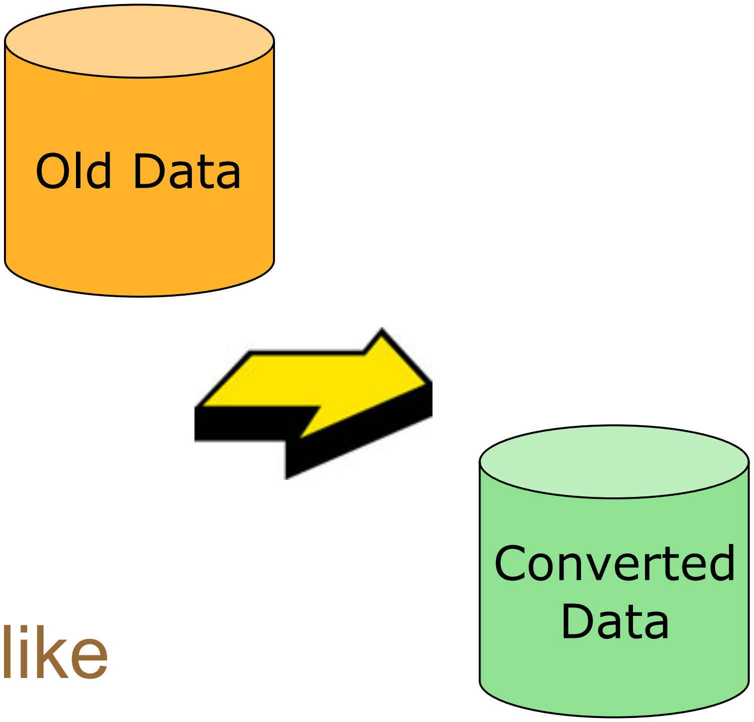


Quadrant Four Tests

- Performance
  - How fast? Identify bottlenecks
- Stability
  - How long?
- Reliability
  - How often?
- Scalability
  - How much?
- Maintainability,compatibility, installability…

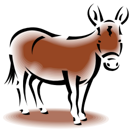
More Quadrant Four Tests

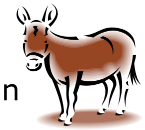
- Memory management
  - Issues such as leaks
- Data migration
  - Conversion scripts
- Recovery
  - Failover testing
- Test environments
  - Independent, production-like
- Baselines
- Write stories for these types of tests



Who Does Quadrant 4 Tests, When?

- Depends on priorities
- May need from start
- May need to test scalability early
- It pays to get a baseline
- Programmers can write multiple-thread harnesses at unit level
- Plan for specialists as needed
- Team responsibility



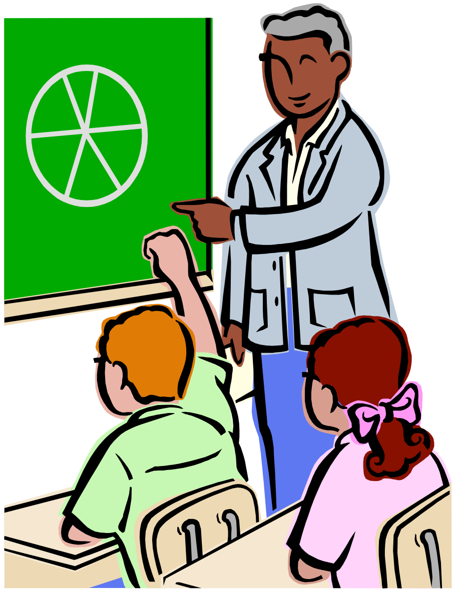


Quadrant Four Automation

- Write stories to evaluate tools
- Specialists?
- Training in Quadrant Four testing skills

Quadrant Four Automation

- Write stories to evaluate tools
- Specialists?
- Training in Quadrant Four testing skills



Quadrant Four Automation

- Native database tools
  - SQL, data import tools
- Shell scripting
- Monitoring tools examples
  - jConsole
    - Application bottlenecks, memory leaks
  - jProfiler
    - Database and bean usage

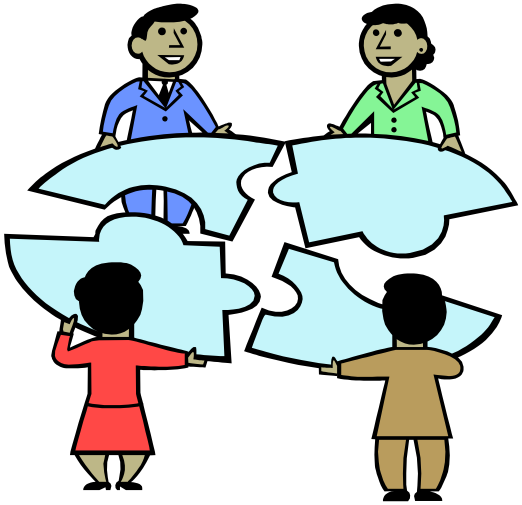
More Quadrant Four Tools

- Commercial load test tools
  - Loadrunner
  - Silk Performer
- Open source test tools
  - jMeter
  - The Grinder
  - jUnitPerf
- Performance test providers
  - Multiple sites

Questions?

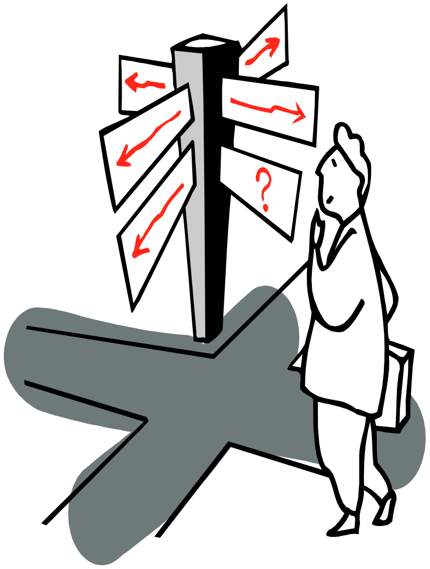
Doneness

- No story is done until testing complete
- Automated regression tests
- Customer requirements captured as passing tests
- Delivers value
- Doneness in all quadrants



Planning Your Test Strategy

- Scope
- Priorities, risks
- Tools that solve the problem
- Customers
- Document only what is useful
- Consider all four quadrants
- Use lessons learned to improve



Group Exercise

In your small groups: Draw the four quadrants on a big sheet of paper.

Make notes in each quadrant where your team lacks one or more types of tests.

Group the similar types. Are there common problem areas? What's the weakest quadrant?

What will you do to address this, when you go back? Share "Aha" moments.

Available Now!

Agile Testing: A Practical Guide for Testers and Agile Teams

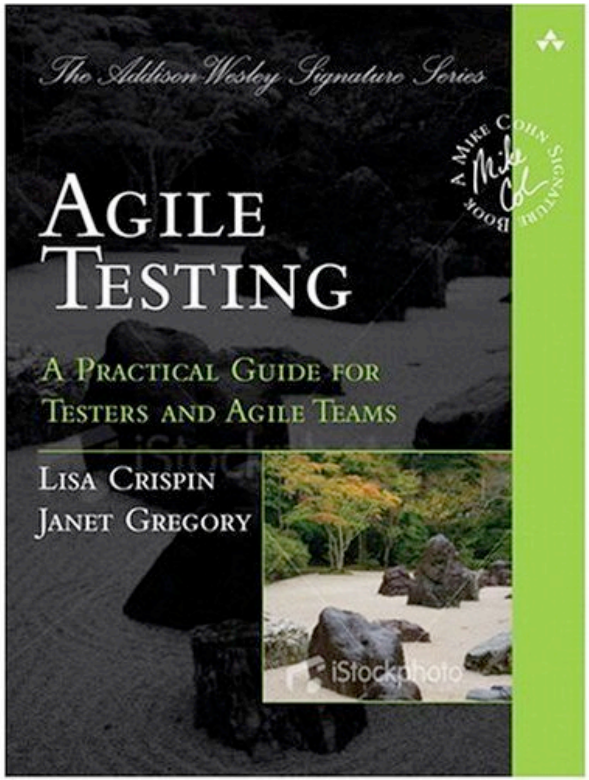
By Janet Gregory and Lisa Crispin

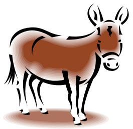
Available on

- Amazon.com
- Amazon.ca

www.agiletester.ca

www.janetgregory.ca

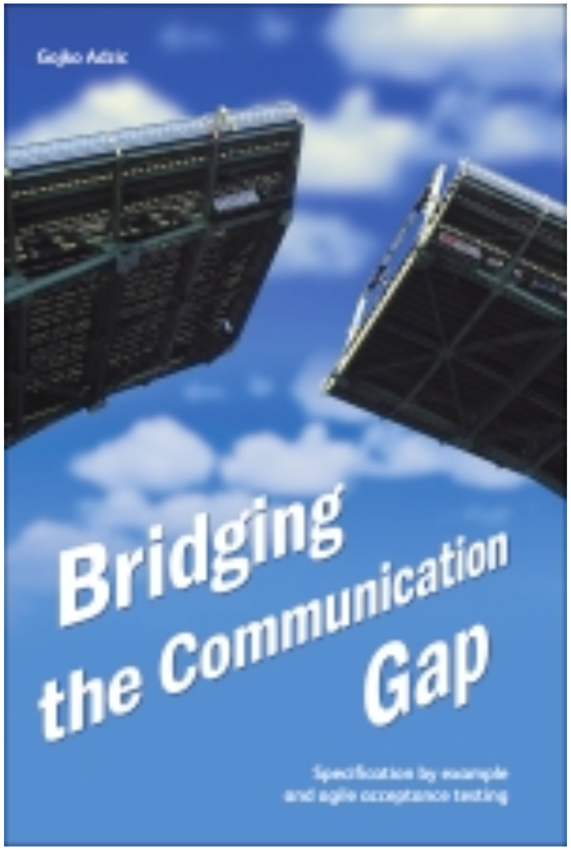

The Addison-Wesley Signature Series
A Mike Cohn Signature Book
Agile Testing
A Practical Guide for Testers and Agile Teams
Lisa Crispin
Janet Gregory
iStockphoto



Bridging the Communication Gap

Specification By Example and Acceptance Testing

Gojko Adzic

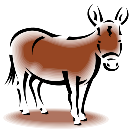

Gojko Adzic
Bridging the Communication Gap
Specification by example and agile acceptance testing

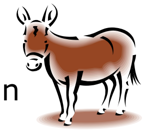
Some Agile Testing Resources

- lisacrispin.com
- janetgregory.ca
- exampler.com
- testobsessed.com
- agile-testing@yahoogroups.com
- www.fitnesse.org
- webtest.canoo.com
- fit.c2.com
- www.awta.org

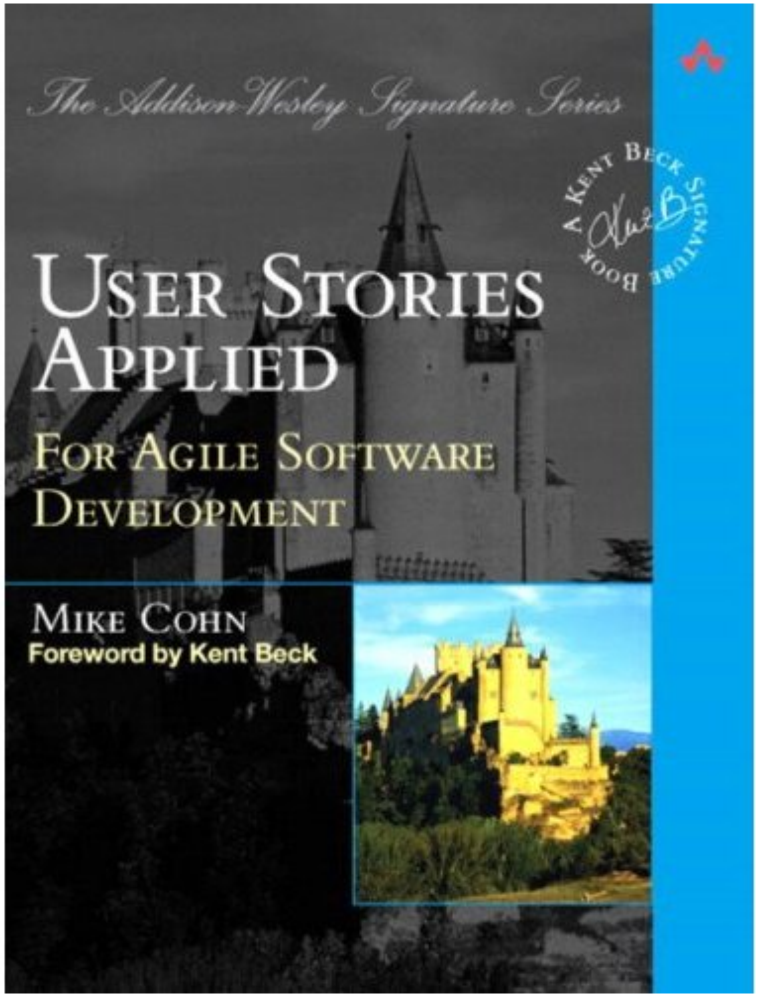
Exploratory Testing Resources

- Testing Computer Software, Kaner
- Lessons Learned in Software Testing; Kaner, Bach, Pettichord
- www.testinglessons.com
- www.developsense.com





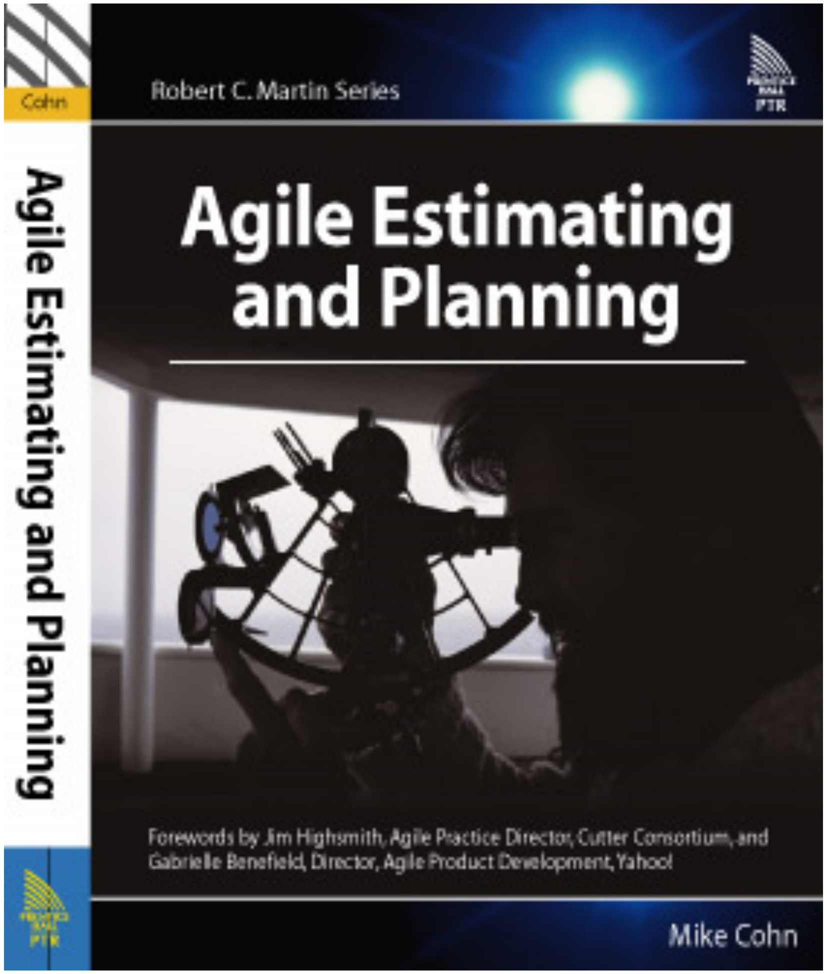
Agile Resources


The Addison-Wesley Signature Series
A Kent Beck Signature Book
User Stories Applied
For Agile Software Development
Mike Cohn
Foreword by Kent Beck

User Stories Applied

by Mike Cohn

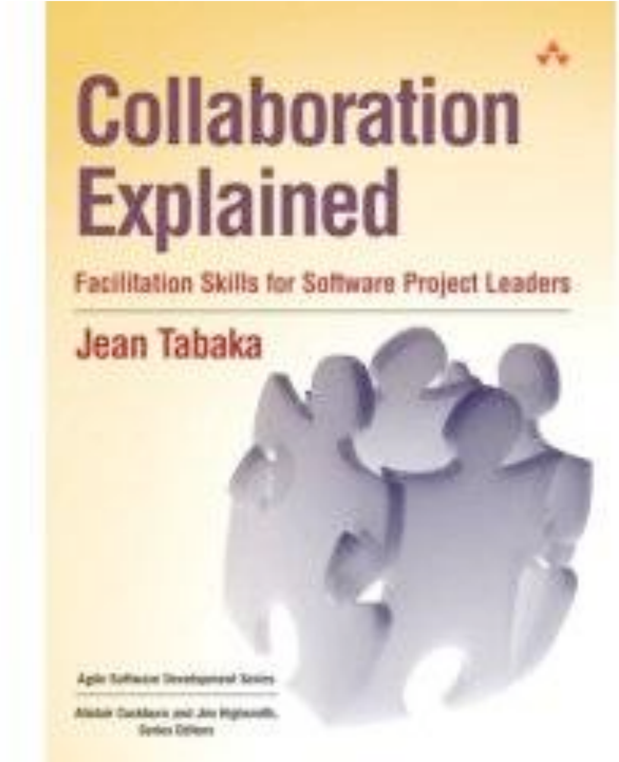
Agile Resources


Robert C. Martin Series
Agile Estimating and Planning
Forewords by Jim Highsmith, Agile Practice Director, Cutter Consortium, and Gabrielle Benefield, Director, Agile Product Development, Yahoo!
Mike Cohn
Cohn
Agile Estimating and Planning
PRENTICE HALL PTR

Agile Estimating and Planning

By Mike Cohn

Collaboration

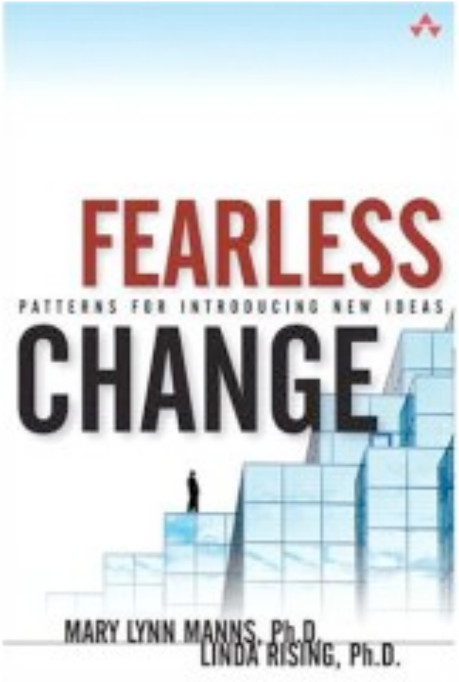

Collaboration Explained
Facilitation Skills for Software Project Leaders
Jean Tabaka
Agile Software Development Series
Alistair Cockburn and Jim Highsmith, Series Editors

Collaboration Explained : Facilitation Skills for Software Project Leaders

By Jean Tabaka

Available on Amazon

Implementing Change



Fearless Change: Patterns for introducing new ideas

By Linda Rising and Mary Lynn Manns

Available on Amazon

Goal

Have fun, whatever you do!

Questions?