

# Transitioning Quality Processes to Agile Methods

Quebec City Agile User's Group  
April 15, 2009

Janet Gregory, DragonFire Inc.

With material from Lisa Crispin



Copyright 2009 Janet Gregory, DragonFire

# Introduction - Me



First agile team – 2000

Currently – coaching, training, testing

Agile Testing: A Practical Guide for Testers and Agile Teams;  
Addison Wesley 2009



Copyright 2009 Janet Gregory, DragonFire

# Quality Processes ??

---

- Change Management
- Project Management
  - Risk Management
- Requirements Management
  - Traceability
- Configuration Management
  - Development Practices
- Testing & Test Management
  - Defect Tracking
  - Release Management

.....



# Introducing Change

---

## Linear model

- Orderly, one thing at a time
- Works in stable environments

## Organic model

- Normal part of growth
- Individuals have the power to change



# Balancing Act – Why Change?

---



- Does it add value?
- Does it address a business need?
- Does it give a competitive advantage?
- Or is it just another new cool idea to try?



**It's a cultural shift needed by organizations.  
Many teams just don't take it into consideration.**



# Whole Team Commitment to Quality

---

- Cannot test quality in ...
- Produce highest quality product possible
- Everyone “test-obsessed”
- Tests drive design, coding
- Tests let us know we’re done
- Who says whether it’s good enough”?
- Attitude – honesty and trust, courage to be honest



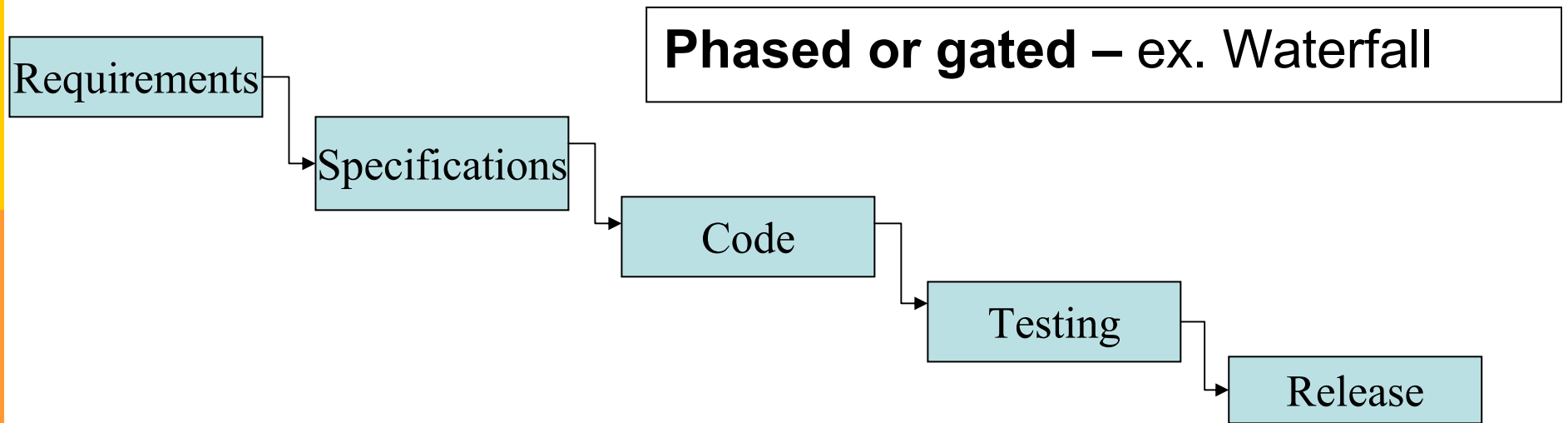
# Risk Management

---

- **Smaller Iterations**
  - Faster feedback
  - Improves estimation accuracy
- **Big Visible Charts**
  - Raises issues quickly
- **Communication**
  - Daily stand-ups
  - Iteration reviews
  - Retrospectives

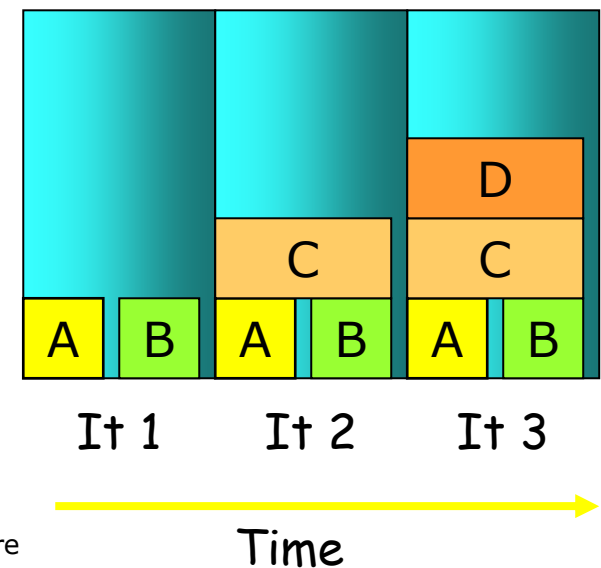


# Requirements Traceability ....



## Agile: iterative and incremental

- Each story is expanded, coded and tested
- Possible release after each iteration



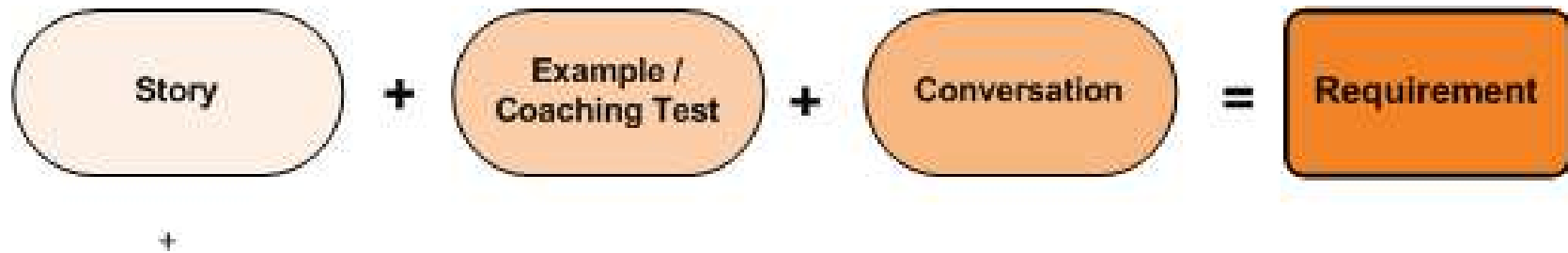
# Requirements Management

---

- Product Backlog
  - Customers list features wanted
  - Team breaks features into stories
  - Customers determine priority of each story
- Initial estimates by team
  - determines scope and tasks
  - Assess risk of stories and tasks



# Requirements ...



- Stories are not enough
- Acceptance tests – defined by customer
- Quality level – owned by customer
- Traceability



# Configuration Management

---

- Source Control
- Releasable packages
- Continuous Integration (CI)
  - Eliminates integration hell - catch errors early
  - 3<sup>rd</sup> party integration
- Release Management
  - Build once: deploy many times
  - Practice deployment



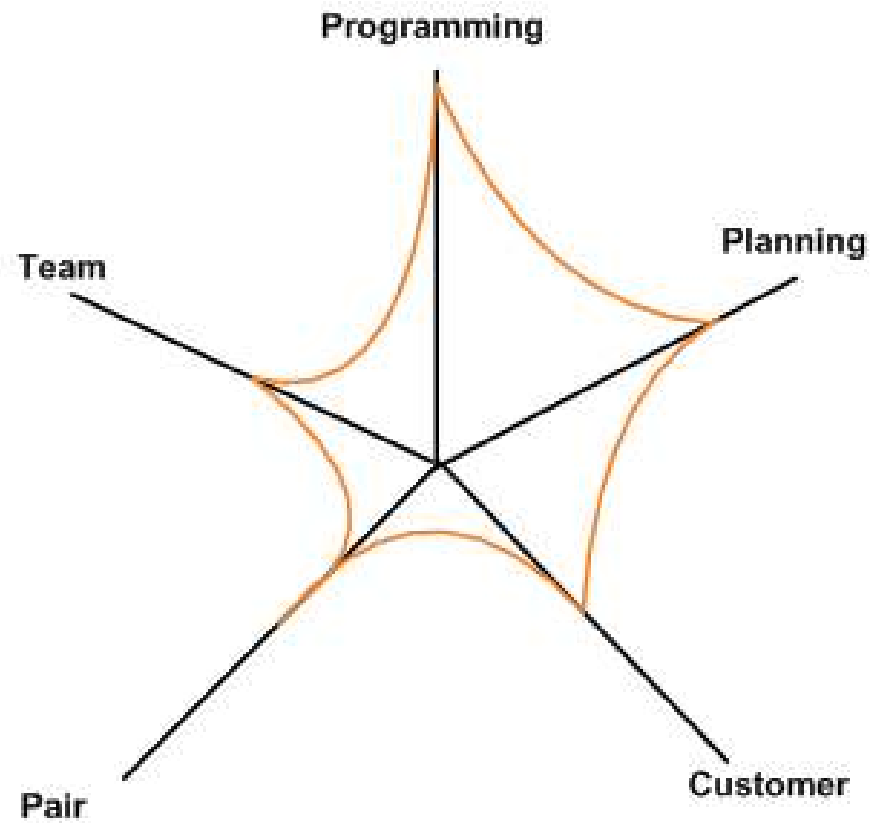
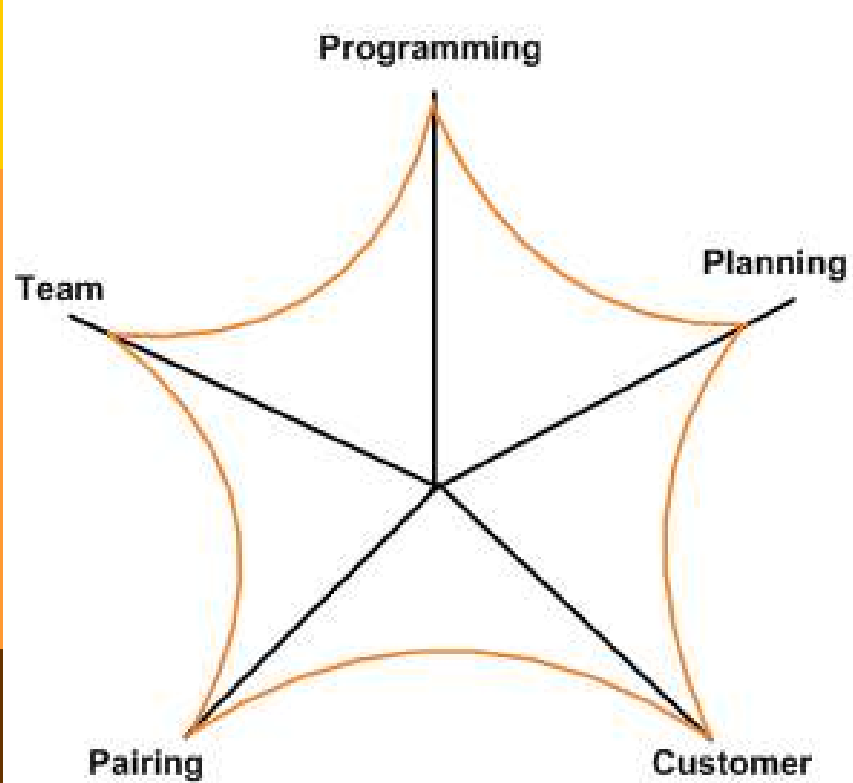
# Development Practices

---

- **Test first**
  - Reduces need for debugging
  - Creates customer confidence
  - Pair Programming
  - Reduces need for formal code reviews
- **Collective Code Ownership**
  - Coding standards means everyone is on the same page
- **Pair Programming**
  - Constant code review



# XP Radar Chart



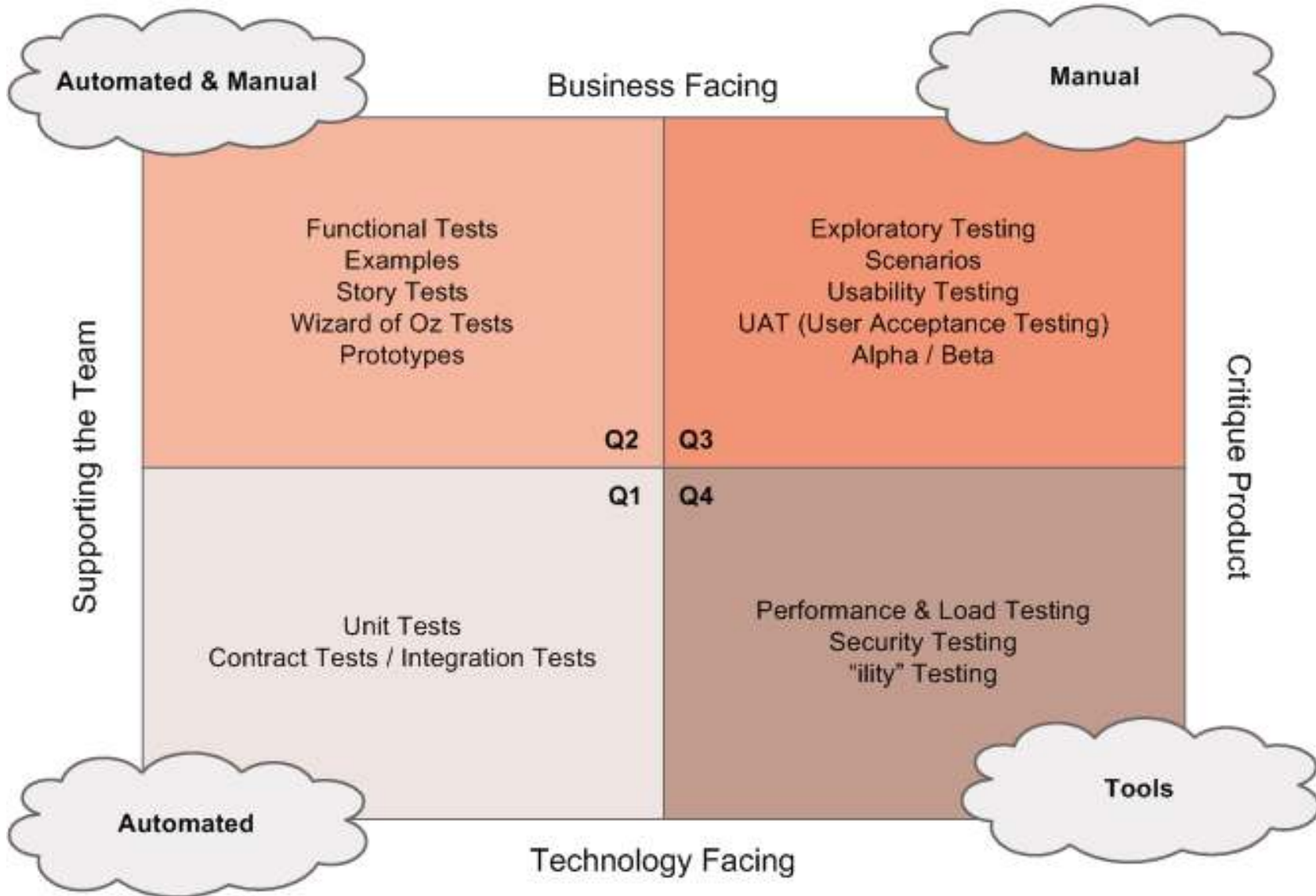
# Test & Test Management

---

- Drive development with executable tests
- Design for testability
- Test as you go
- Embrace automation
  - Team effort
  - Team chooses tools
  - Start simple



# Agile Testing Quadrants



# Automate Regression Testing

Add Employees

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



# Defect Tracking

---

## An agile approach:

- Understand the problem
- Do what works for your team
- Focus on goals: bug prevention
- Start simple, add as needed
- Explore alternatives



# Explore Alternatives

---

- Start new project with no DTS
  - See what you need
- Set rules - “no more than 10 bugs at once”
  - Analyze and address bad trends
- Fix all bugs
  - Low priority usually quick to fix
  - Unfixed bugs build technical debt quickly
- Treat bugs as stories
  - Estimate and prioritize
- Color-coded cards and stickers
  - Visible, promotes communication



# When to Fix Bugs

---

- **Fix now**
  - way cheaper
  - prevents technical debt
- **Estimate, prioritize and fix later**
  - complex, high effort
  - legacy code, may destabilize
  - customer decision
- **Never fix**
  - about to be rewritten
  - low risk
  - don't pretend you'll fix if you won't!



# Logging Defects

---

- Unit-level bugs
  - Write a test, not a bug
- Bugs found during development
  - Fix immediately in most cases
  - Log on card rather than DTS if not quick fix
  - Log in DTS if not fixed in iteration
- Legacy system
  - Log in DTS (if not part of current dev)
- Production system
  - Log in DTS (always)



# Retrospective – Continuous Improvement

---

- Solve problems as a team
- Is there one single limiting factor?
  - something holding you back
  - something causing a blown iteration
  - anything causing other big problems
- Action items
  - Focus on 1 or 2 things to improve
  - Create task cards for future iterations
  - Need new resources?
    - E.g.. new team member with particular expertise



# What Can You Do?

---

- Try small, incremental changes
- Bring problems to team, use retrospectives
- Be patient – change is hard
- Need time to learn new way of working
- Build credibility, trust



# Take-aways....

- Consider the alternatives
- Everyone can be a change agent
- Communication is key
- Be flexible
- Involve the whole team in solutions



# Now Available

## *Agile Testing: A Practical Guide for Testers and Agile Teams*

By Lisa Crispin and Janet Gregory

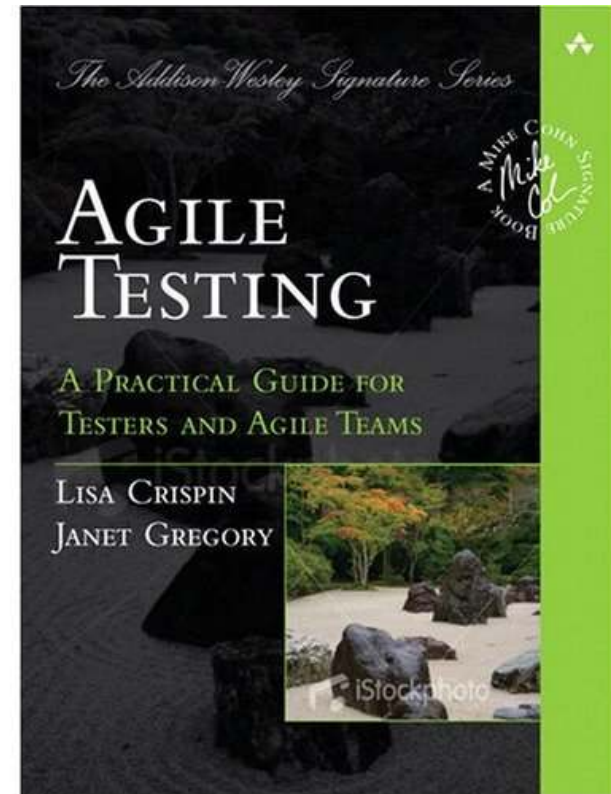
[www.agiletester.ca](http://www.agiletester.ca)

My contact info

[www.janetgregory.ca](http://www.janetgregory.ca)

<http://janetgregory.blogspot.com/>

[janet@agiletester.ca](mailto:janet@agiletester.ca)



# Resources

---

- Gerry Weinberg, *Becoming A Technical Leader*, Dorset House 1986
- Marylynn Manns and Linda Rising, *Fearless Change: Patterns for Introducing New Ideas*, Addison-Wesley 2004
- Mary Poppendieck and Tom Poppendieck, *Lean Software Development*, Addison-Wesley 2003
- [agile-testing@yahooogroups.com](mailto:agile-testing@yahooogroups.com)
- Agile Manifesto: <http://agilemanifesto.org/>
- [www.lisacrispin.com](http://www.lisacrispin.com)
- [www.mountangoatsoftware.com](http://www.mountangoatsoftware.com) — Mike Cohn's web site

