

# Driving Development



With Business-Facing Tests

*Lisa Crispin*

# Takeaways

---

- Why drive development with tests
- When to write tests
- Who should write tests
- How to help customers express requirements
- Turn requirements into business-facing tests



# Why Business-Facing Tests First?

---

- Prevent business and technical disconnects
- Testers help team see big picture
- Even with TDD, missed requirements
- Define quality and success criteria, milestones



# BFTDD and Technical Debt

---

- Prevent technical debt
  - Missed requirements
  - Poorly designed, untestable code



# Benefits of BFTDD

---

- Provides advance clarity
- Provides a common language
- Collaboration
- Better quality code
- Higher business value
- Quick feedback loop

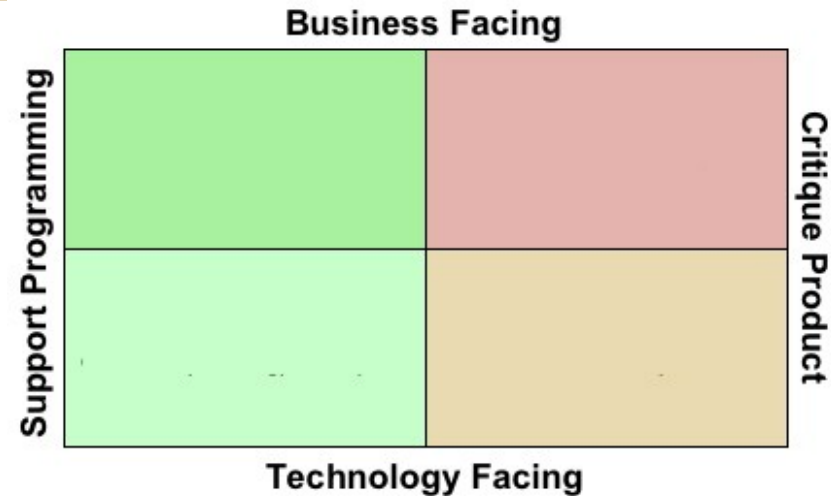


# Driving Development with Tests

---

## Test-Driven Development

- Unit tests before code
- Design tool



## Business-Facing TDD

- Examples show desired behavior
- When tests pass – are you done?



# When to Write Tests?

---

## High level tests

- Before coding starts
- Show big picture

## Detailed tests

- Concurrently with coding
- Capture details in executable tests



# Who Writes Tests?

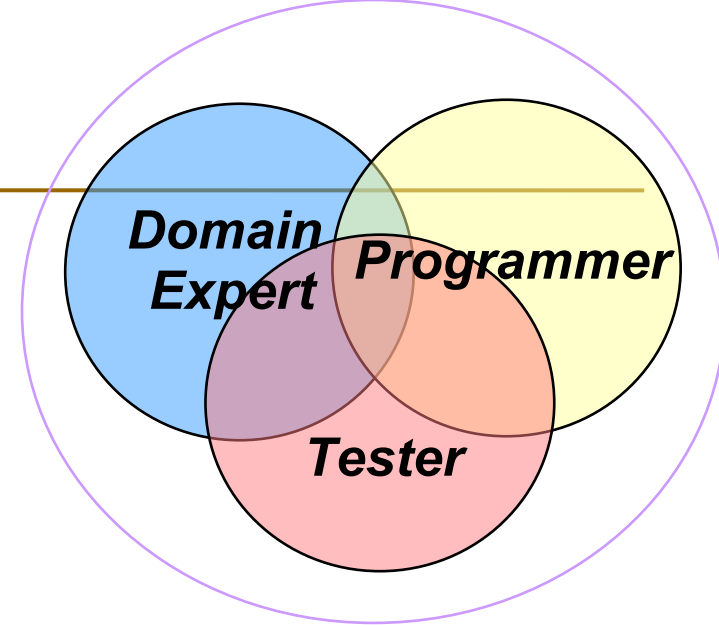
---

## Programmers

- Unit, component tests
- Help automate other tests

Testers, business experts, product owners, analysts, specialists

- Business-facing tests
- e.g. Functional, usability, performance, security...



# Building Requirements: Conversations

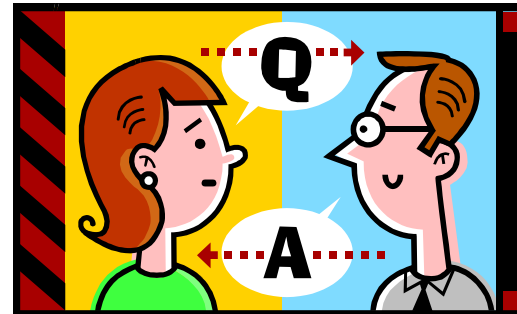
---

Ask open-ended questions:

*“Who will use the exported data? How will they use it?”*

Not

*“Should there be a link on the TPA summary page to export a .csv file?”*

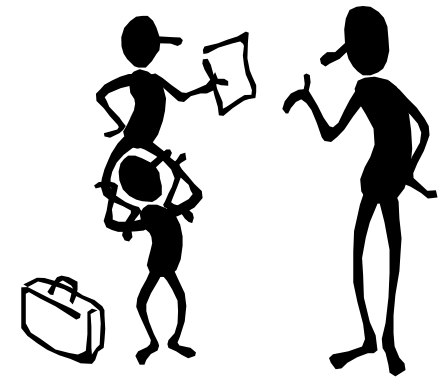


# Eliciting Requirements

---

Flush out hidden assumptions

- What business problem does this feature solve?
- How will end users use the feature?
- Simplest programming implementation?
- Any mismatches?

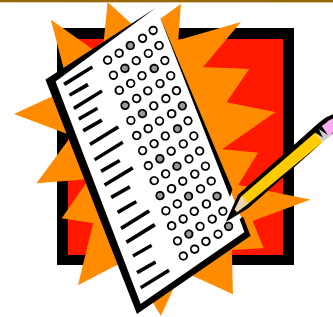


# Eliciting Requirements

---

## Whiteboard discussions

- Examples
  - Appropriate to domain, e.g. spreadsheets for financial apps
- Examples of undesirable behavior, too
- Pictures, diagrams, charts help!
- Story + use case + example/test = requirement



# Capturing Details in Executable Tests

---

- Illustrate examples
- Programmers may automate
- Form useful documentation
- Collaborative, iterative process
- Flushes out misunderstandings
- Provide daily feedback to business, development team

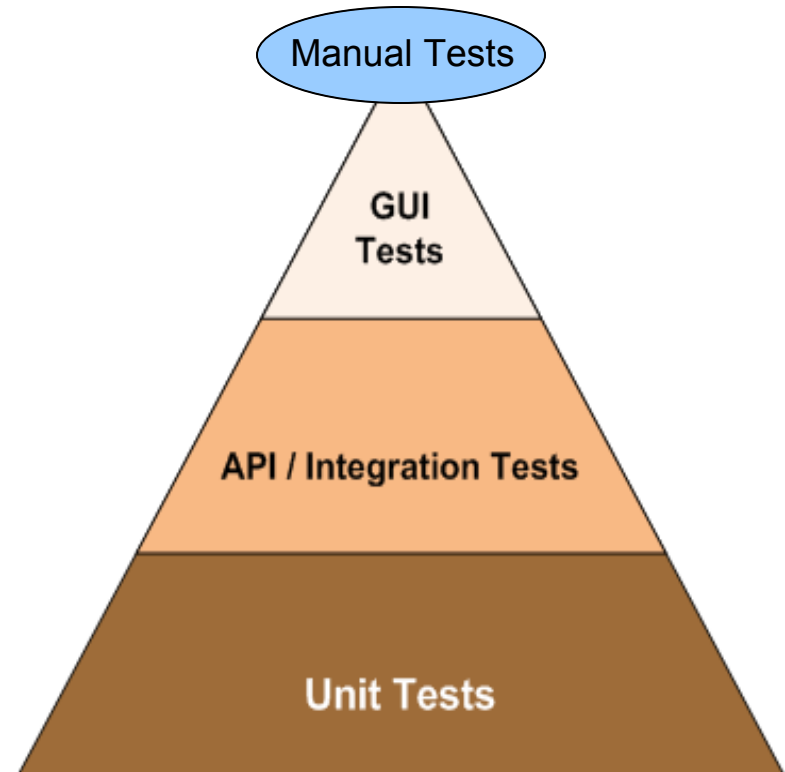


# Test Design

---

## Agile principles apply to tests

- Start simple
- Refactor as needed
- Pair
- Testable code design



# How My Team Drives Coding with Tests

---

Prior to or first day of iteration – product owner explains stories

- Examples
  - Spreadsheets, mockups
  - Whiteboard
- Story checklist
  - May provide a few high level tests



# First Few Days of Iteration

---

Testers write narrative requirements, high level tests

- Wiki pages easily found for current iteration
- Review with product owner, programmers
- Too much detail can cloud picture

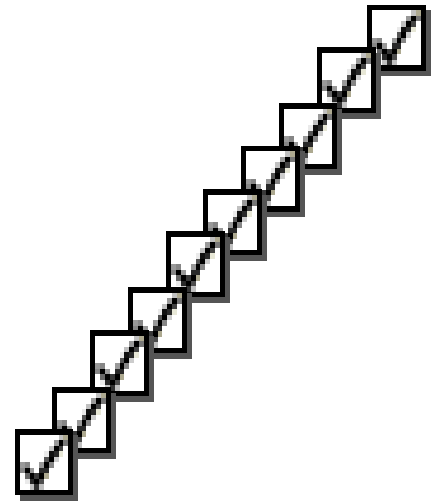


# When Coding Starts on a Story

---

Tester writes detailed test cases in FitNesse

- Simple, happy path executable test
- Programmer automates simple test
- Tester writes more test cases
  - Sad, bad, ugly paths
  - But just enough!



# Concurrently with Coding

---

Other examples/tools/guidance from testers:

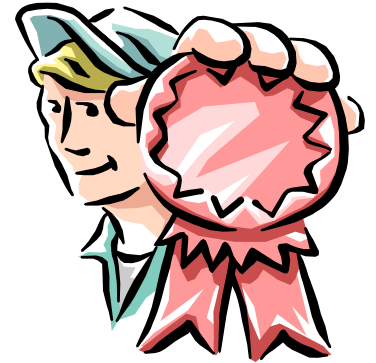
- Spreadsheet to illustrate, verify results
- Exploratory testing
- Iterate; test-code-test-code



# Development Complete

---

- All regression tests automated
- All automated tests pass
- Tests document functionality
- Exploratory testing complete
  - New tests/stories may result



# Another success story

---

## “Acceptance Test Driven Planning”, Richard Watts and David Leigh-Fellows

- QA works with customers after each iteration review to write tests for next iteration
- They use these tests to guide planning, estimation and task breakdown
- Not “big up-front design”, but “just enough”



# Test Tools, Frameworks

---

- Test automation pyramid
- Promote tester-programmer collaboration
- Combine narrative and executable tests
- Easy to understand
- Team chooses tools
- Experiment



# Types of Tools and Frameworks

---

- Unit, component testing – XUnit, BDD tools
- Behind the GUI, Collaborative – Fit, FitNesse, Rasta, home-grown frameworks
- Behavior-Driven Development – RSpec, Cucumber, easyb

```
Feature: Addition
  In order to avoid silly mistakes
  As a math idiot
  I want to be told the sum of two numbers

Scenario: Add two numbers
  Given I have entered 50 into the calculator
  And I have entered 70 into the calculator
  When I press add
  Then the result should be 120 on the screen
```



# Tools to Elicit Examples

---

Brainstorming tools – can be low tech!

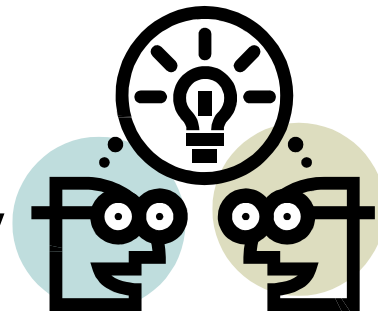
- Mind maps
- Flow diagrams, prototypes
  - Paper, whiteboard
- Conversations – power of three
- Testers facilitate communication



# “I’m Not on an Agile Team”

---

- Use business-facing tests as requirements
- Engage business people early
- Collaborate with programmers early
  - Review tests/examples together
  - Ask for small testable pieces early
- *Lots of face-to-face communication!*



# Benefits

---

- Code designed for testability
- Functionality better matches business expectations
- Better end user experience
- Time saved
- Better value to business



# Agile Testing Resources

---

- [www.lisacrispin.com](http://www.lisacrispin.com)
- [www.janetgregory.ca](http://www.janetgregory.ca)
- [www.agilealliance.org](http://www.agilealliance.org)
- [www.exampler.com](http://www.exampler.com)
- [www.testobsessed.com](http://www.testobsessed.com)
- [www.satisfice.com](http://www.satisfice.com)
- [www.testingreflections.com](http://www.testingreflections.com)
- [agile-testing@yahoogleroups.com](mailto:agile-testing@yahoogleroups.com)



Local course: Applied Agile Testing - Prototest



# Tool Links

---

- [softwareqatest.com](http://softwareqatest.com)
- [www.fitnessse.org](http://www.fitnessse.org)
- [webtest.canoo.com](http://webtest.canoo.com)
- [fit.c2.com](http://fit.c2.com)
- [watir.com](http://watir.com)
- [seleniumhq.org](http://seleniumhq.org)
- [cukes.info](http://cukes.info)
- [easyb.org](http://easyb.org)



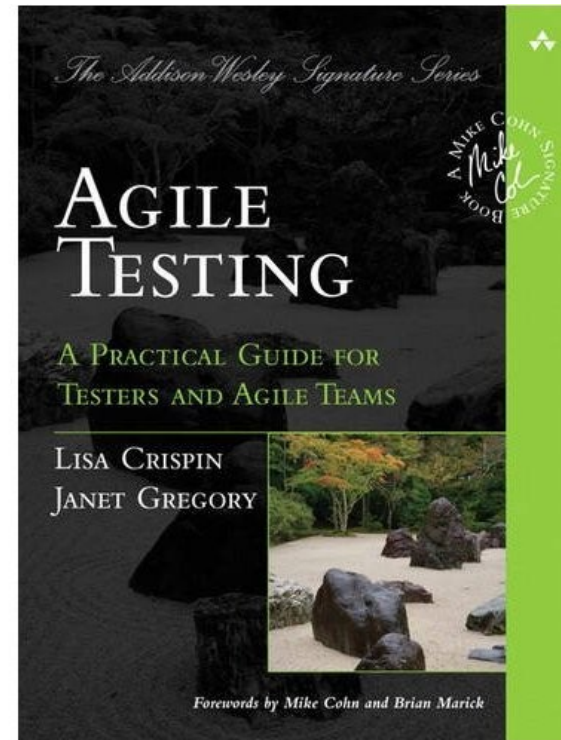
# Now Available

---

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

By Lisa Crispin and Janet Gregory

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



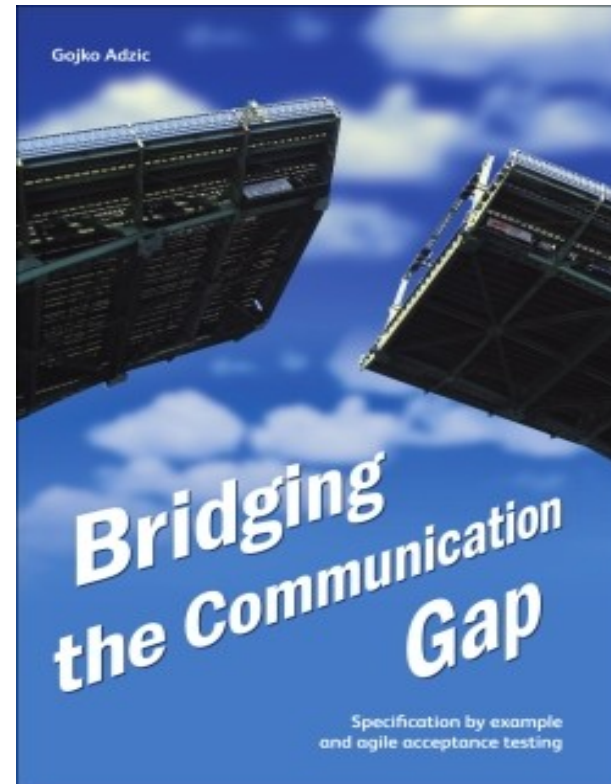
# Now Available

---

## *Bridging the Communication Gap: Specification by Example and Agile Acceptance Testing*

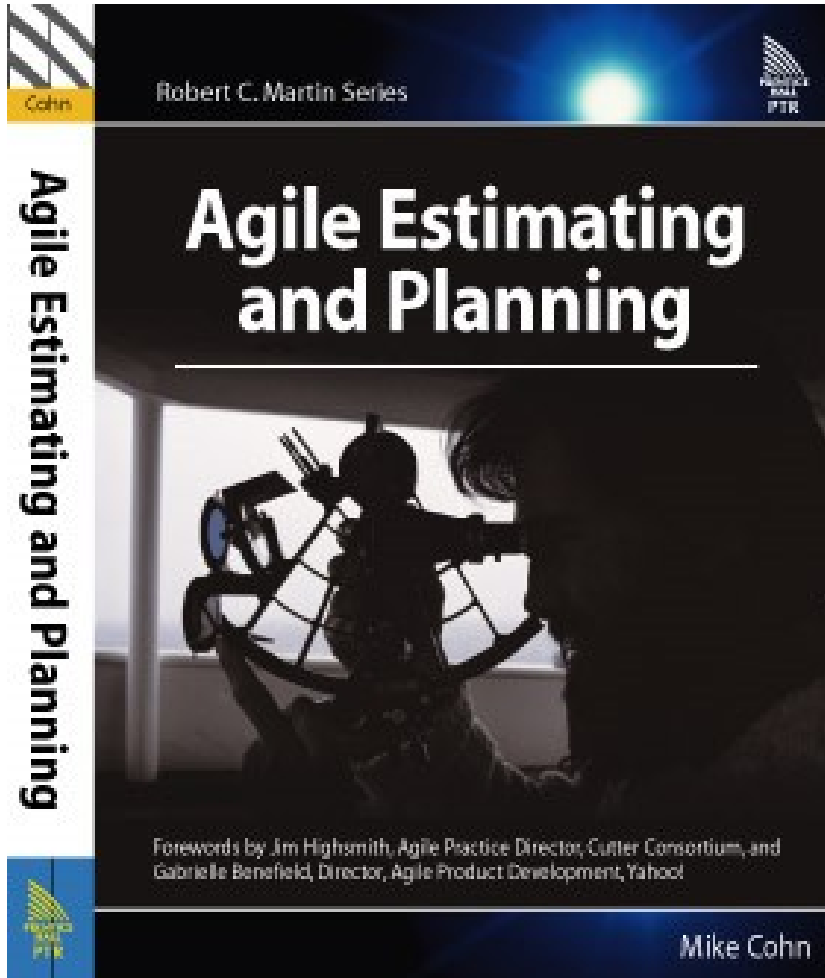
By Gojko Adzic

[www.gojko.com](http://www.gojko.com)



# Estimating

---



*Agile Estimating  
and Planning*

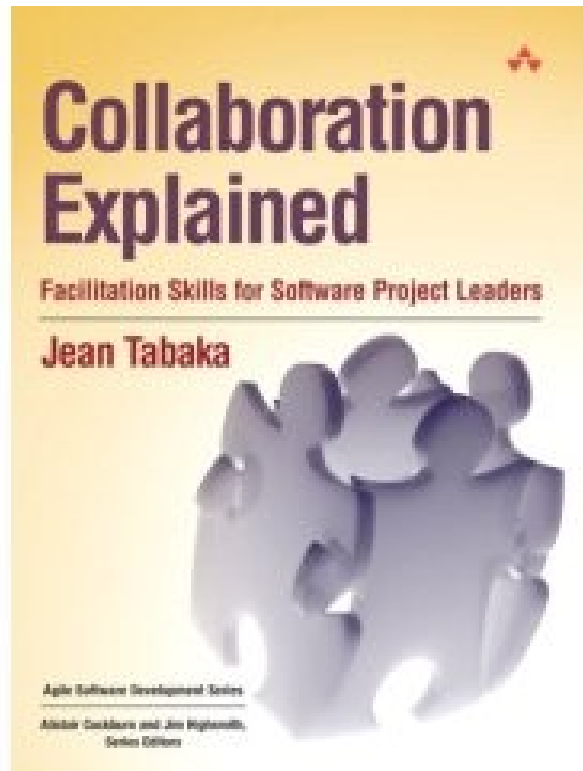
By Mike Cohn

Available on  
Amazon



# Collaboration

---



*Collaboration Explained :  
Facilitation Skills for  
Software Project Leaders*

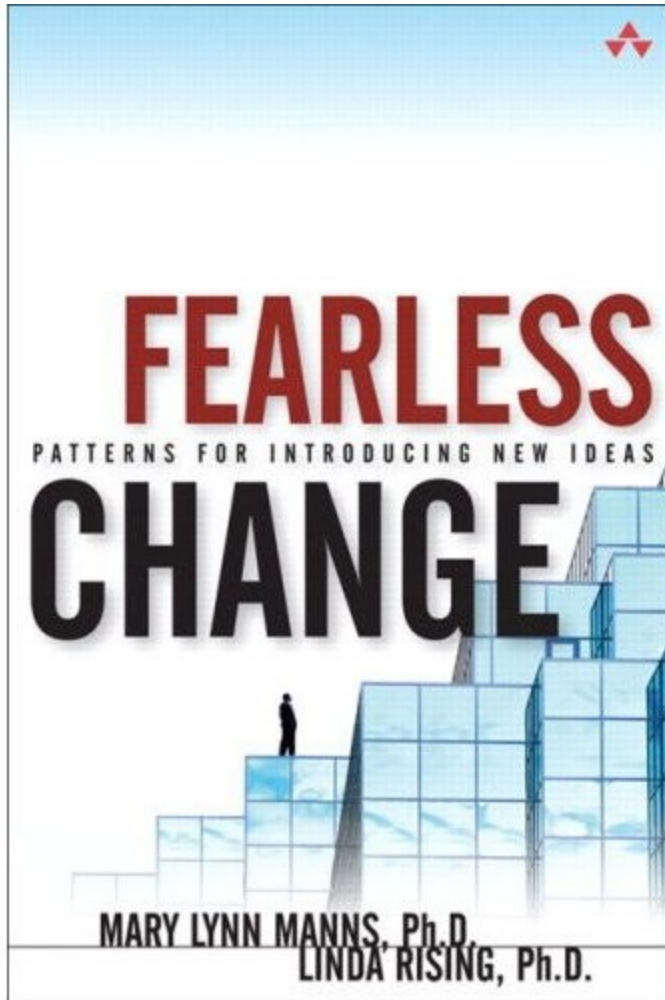
By Jean Tabaka

Available on Amazon



# Implementing Change

---



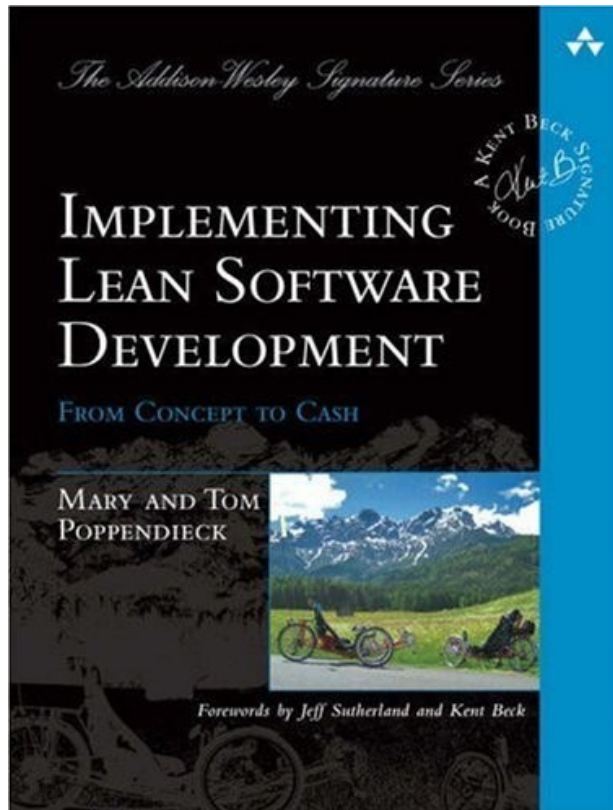
*Fearless Change:  
Patterns for introducing  
new ideas*

By Linda Rising and  
Mary Lynn Manns



# Lean Development

---



*Implementing Lean Software Development: From Concept to Cash*

By Mary and Tom Poppendieck

Available on Amazon



# Questions?

---

