Course Outline

Write Right Agile User Stories and Acceptance Tests Right

Course Summary

Description

Everyone complains that poor requirements are the major cause of project problems. Yet, like the weather, nobody does much about it, at least not effectively. Traditional approaches advocate writing voluminous requirements documents that too often don't seem to help much and may even contribute to difficulties. Agile goes to the opposite extreme, relying on brief requirements in the form of three-line user stories that fit on the front an index card and a few user story acceptance criteria that fit on the card's back. Surprise, as Mark Twain noted, in some ways it's even harder to write Agile's brief requirements effectively. This interactive workshop reveals reasons user stories and their acceptance tests can fall short of their hype, explains critical concepts needed for effectiveness, and uses a real case to provide participants guided practice writing and evaluating user stories and their acceptance criteria/tests.

Objectives

By the end of this course, students will learn:

- Major sources of poor requirements that cause defects, rework, and cost/time overruns.
- How Agile user stories and their acceptance criteria/tests address these issues.
- Difficulties that still afflict requirements in Agile projects and why they persist.
- Writing more effective user stories and acceptance criteria/tests.
- What else is necessary to produce working software that provides real value.

Topics

- Agile, User Story Fundamentals
- Requirements are Requirements—
- Or Maybe Not
- Writing More Suitable User Stories
- Product Owner Vs. Business Analyst
- (Hidden) Backlog Issues
- Conversation Conundrum
- User Story Acceptance Tests
- Testing Concepts

Audience

This course has been designed for product owners, analysts, developers, and other Agile (and other) project team members who are or should be involved in defining requirements.

Duration

Two Days

Course Outline

Write Right Agile User Stories and Acceptance Tests Right

Course Outline

I. Agile, User Story Fundamentals

A. Agile Manifesto's relevant points
B. Characterization of traditional approaches
C. Waterfall and big up-front requirements
D. Agile's sprints and backlogs alternative
E. Agile project team roles
F. User story "As a <role>…" (Card)
G. User story acceptance criteria (Confirmation)
H. Given, when, then format
I. Estimating user story size
J. Splitting and refining
K. Prioritizing and allocating to backlogs/sprint
L. Constructing/implementing (Conversations)
M. Reviewing, retrospectives
N. Grooming backlog and reprioritizing
- Exercise: Write Needed User Stories
- Exercise: Define their Acceptance Criteria
- Exercise: Review Your User Stories/Criteria
- Exercise: Split/Groom Your User Story

II. Requirements are Requirements—Or Maybe Not

A. User stories are backlog items, features
B. Chicken and egg relation to use cases
C. Issues and inconsistencies
D. Business vs. product/system requirements
E. "Levels Model" of requirements
F. Other mistaken presumptions
G. Requirements overview
H. Where user stories should fit, do fit instead
I. Conversation conundrum
- Exercise: Write REAL Business Requirements
- Exercise: Write another User Story
- Exercise: Review Your New User Story

III. Writing More Suitable User Stories

A. Focus on what provides value
B. Users, customers, and stakeholders
C. Exercise: Identify Overlooked Stakeholders
D. Problem Pyramid tool to get on track
- Exercise: Find Value
- Exercise: Using the Problem Pyramid
- Exercise: Business Requirement User Stories
- Exercise: Size Supplied User Story
- Exercise: Split Supplied User Story
- Exercise: Define Features for User Story
- Exercise: Define Feature's 'Story'

IV. Product Owner Vs. Business Analyst

A. Business analysis discovers requirements
B. Product owner (PO) role created by Agile
C. Essential PO characteristics
D. Skills/knowledge for authority
E. Product owner viewed as the analyst
F. Should a business analyst (BA) be the PO
G. Rethinking the PO role
- Exercise: Designing a Better PO Role

V. (Hidden) Backlog Issues

A. What is a backlog item
B. What else often also are backlog items
C. Apples, oranges, and fruitcakes impacts
D. Different needs, purposes, artifacts
E. User stories for prioritization on value
F. Features for product delivery
G. Tasks for estimating, performing work

Course Outline

Write Right Agile User Stories and Acceptance Tests Right

Course Outline (cont.)

H. User story mapping
I. Addressing quality factors
J. Dealing with defects
K. Agile's difficulty scaling and integrating
L. Sprint 0
M. Spikes
- Exercise: Design a Better Backlog

VI. Conversation Conundrum

A. Placeholder metaphor
B. Developer as BA, pros and cons
C. Data gathering and analysis
D. Planning an effective interview
E. Controlling with suitable questions
F. Then a miracle occurs…
- Exercise: Design a Better Conversation

VII. User Story Acceptance Tests

A. Typical development/testing, issues
B. Test-first development/testing
C. Test-driven development issues
D. Acceptance Test-Driven Development
E. Common guidance defining criteria/tests
F. Confirming vs. defining requirements
G. Suitability of using for quality factors
H. Differences from test-first unit tests
I. Traps when tests = requirements
J. Inadequate, missed and unclear criteria
- Exercise: Write User Story Acceptance Criteria
- Exercise: Review/Revise Acceptance Criteria

K. Turning criteria into tests, issues
L. How many tests are really needed
M. Tests are main means to control risk
N. Risk elements, relation to tests
O. Test design techniques
- Exercise: Design their Tests
- Exercise: Review Your User Stories/Tests

VIII. Testing Concepts

A. Should tests equal requirements
B. How testing actually works
C. Defining correctness independently
D. Test-first illusion
E. UAT vs. User Story Acceptance Test
F. Demo illusory UAT
G. Test design techniques
H. Checklists and guidelines
- Exercise: Applying Checklists/Guidelines

I. Decision trees, decision tables
- Exercise: Applying Logic Analysis

J. Boundary testing
- Exercise: Applying Boundary Tests

K. Reactive vs. proactive risk analysis
L. Putting Agile TDD on steroids
- Exercise: Applying Proactive Risk Analysis