Deliver Quality with Agile and Lean

Johanna Rothman
Agile and Lean Program Management: Scaling Collaboration Across the Organization
@johannarothman
www.jrothman.com
jr@jrothman.com
781-641-4046
Is Your Agile Approach Working for You?

- Easy releasing is a problem:
  - QA doesn’t “finish” the stories before the iteration ends
  - Integration testing has no home (no one’s responsibility)
  - Quality is not clear for a story, iteration, or release
  - You need to fix defects past the iteration or feature-done
- Possibly other “agile” problems…
- You don’t see that agile is helping your quality or speed of release
Our Agenda

• Agile and Lean and why they work
• Quality concerns
• What to measure
• Role of the QA person/group
Johanna’s General Agile Picture
The Principles Behind the Agile Manifesto

• Satisfy the customer through early and continuous delivery of valuable software.

• Welcome changing requirements, even late in development. Agile processes harness change for the customer’s competitive advantage.

• Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.

• Business people and developers must work together daily throughout the project.

• Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.

• The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.

• Working software is the primary measure of progress.

• Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.

• Continuous attention to technical excellence and good design enhances agility.

• Simplicity--the art of maximizing the amount of work not done--is essential.

• The best architectures, requirements, and designs emerge from self-organizing teams.

• At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.
Lean Principles

1. Eliminate waste.
3. Decide as late as possible.
4. Deliver as fast as possible.
5. Empower the team.
6. Build integrity in.
7. See the whole.

— Mary and Tom Poppendieck, *Lean Software Development: An Agile Toolkit*
Anyone Have This Problem?

Two weeks of development

Two weeks of testing

This duration is the entire time box that counts
Why Agile/Lean Works

- Small batch size
- Collaboration among everyone
- Frequent feedback
- See working product
- Ability to change is an effect of frequent delivery of value
Our Agenda

• Agile and Lean and why they work

• Quality concerns

• What to measure

• Role of the QA person/group
Where’s Quality?

• Agile and lean are supposed to be better, faster, cheaper

• “Everyone” is in charge of quality

• Agile doesn’t create quality; it creates the environment in which you can deliver quality
“Quality is value to someone”
— Jerry Weinberg
Your Someones

• Product Owner
• Customer(s)
• Developers
• Testers
• Anyone else?
The PO Decides on Quality

- Acceptance criteria
- Frequent review of work in progress and finished work
- Boards help everyone see what’s going on: done and not done
Boards Visualize the Work

Scrum Board

<table>
<thead>
<tr>
<th>Ready</th>
<th>In Progress</th>
<th>Done</th>
</tr>
</thead>
</table>

Kanban Board

<table>
<thead>
<tr>
<th>Ready</th>
<th>Develop and unit test</th>
<th>Dev-Done</th>
<th>System test</th>
<th>Done</th>
</tr>
</thead>
</table>

Ready columns contents do not change during the iteration. The team commits to the Ready column at the beginning of the iteration, for each iteration.

There is a limit on this column. You can swap out something and swap something else in at any time.
Two Major Quality Concerns

• You gave me what I asked for. It’s not what I need.

• It’s not done.
Not What I Asked For

Problem

• Use acceptance criteria
• Rapid and frequent feedback
• Keep stories small and build incrementally
Iterative and Incremental Creation

• Start with simplest thing possible
• Add in more parts until you have a whole
• Iterate on features
• Incremental on details

• Example: Secure login for shopping cart

1. Already-existing user can log in.
2. Already-existing user can add too cart
3. AE user can check out.
4. Add new user. (Add cart and check-out stories…)
5. Add security for users…
Quality Criteria

• What done means to the team
• Acceptance criteria on a story
• Release criteria for a release
Let’s Discuss “Done”

• You’ve heard of “done-done” and “done-done-done”

• What do people mean when they say that?
Done means the feature is releasable.
What Do You Need for Releasable Features?

• Possible list:
  • Collaboration on the feature so more than one person understands the issues
  • Multiple eyes on the code
  • Sufficient test automation
  • Exploratory tests
  • Code & tests checked in
  • Accepted by PO
  • Documentation completed
  • Maybe more for your environment
Releasable Features Means Release is a Business Decision
What Kind of Testing Does Your Product Need?

Unit perspective

- Work product review
- Unit level testing class or function
- Component testing (a cohesive group of related pieces)
- Feature (or module) testing
- Area testing (a related set of features)
- Integration testing (a few features)
- System testing testing the system as a whole
- Smoke testing tends to be about here
What Kind of Features Do You Have?
Role of Automation

• If you create iteratively and incrementally, you will:
  • Add features and their tests (unit, integration, system)
  • Change features and their tests (unit, integration, system)
  • Exploration & Automation!
Questions for You to Ponder

• Do you have areas of testing your team “ignores”?
  • What could you do to address that?
• Does someone ask you to write test cases?
• Do you have automation debt?
• What can you do about test debt?
Our Agenda

• Agile and Lean and why they work
• Quality concerns
• What to measure
• Role of the QA person/group
Burnups vs. Burndowns

• Burnups show you what you have done over time

• Burndowns tell you how far from your “commitment” you are

• What would you rather know?
Several Velocity Stories
“Problems” with Velocity

• Velocity is a measure of capacity, not productivity

• Not always predictable

• Individual to each team, and can vary with domain
Measure Completed Features

• Completed features (running, tested features):
  • Your customers use them
  • You can release them
  • They are valuable

• Include total and remaining features so we have a sense of where we are

• Depends on deliverables, not epics or themes
Product Backlog Burnup

• Real earned value
• Partial answer to “Where are we?”
• Shows value feature-by-feature
• Shows when features grow
What Do You Want Less of?

• Work In Progress (across project)
• Defects
• Other “Less of”:
  • Multitasking
  • ?
What is Productivity in an Agile Environment?

• Features the *team* completes over time

• Teams take features, people don’t take features

  • If people collaborate, swarm, or mob, personal “productivity” is irrelevant

• We don’t normalize features between teams
Some Other Measures

• Time between internal/external releases (trend)

• Build time

• Quality scenarios (performance, reliability, functionality) as trends from build to build
Questions for You to Ponder

• Do you measure snapshots or trends of defects?

• Does your team try to release features even if you have found defects?
Role of QA

• Consider changing the name from QA to test

• Provide information about the product under test, not its goodness or badness
The Agile/Lean Product Development Team Owns Quality

• PO defines quality

• Team implements it as a team

• QA/Tester provides information about the product as the team develops and tests it
Questions for You to Ponder

• Do you think of yourself as a “product development team”?
  • If not, what would it take for you to do so?

• Does someone have a vested interest in you/your group owning quality?
Let’s Stay in Touch

• Please link with me on LinkedIn:
  • www.linkedin.com/in/johannarothman

• Subscribe to the Pragmatic Manager newsletter:
  • http://www.jrothman.com/pragmaticmanager/