AGILE TESTING

Practical guide to Software Quality

THE AGILE MANIFESTO

We are uncovering better ways of developing software by doing it and helping others do it.

CUSTOMER COLLABORATION
over contract negotiation

INDIVIDUALS AND INTERACTIONS
over processes and tools

RESPONDING TO CHANGE
over following a plan

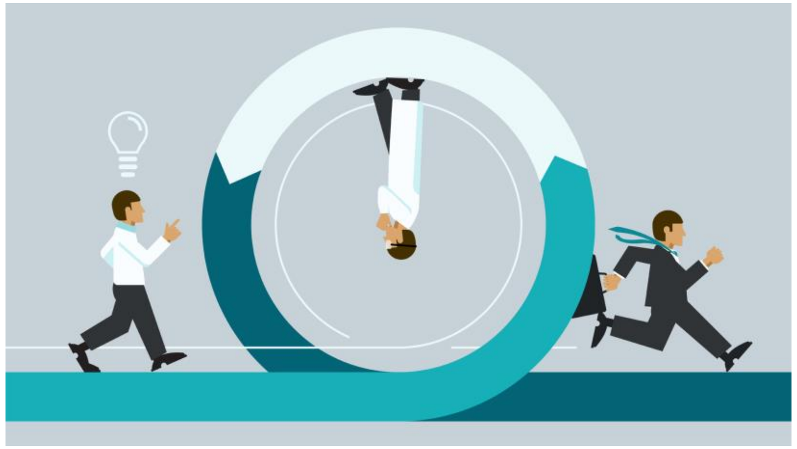
WORKING SOFTWARE
over full documentation

AGILE TESTING ?

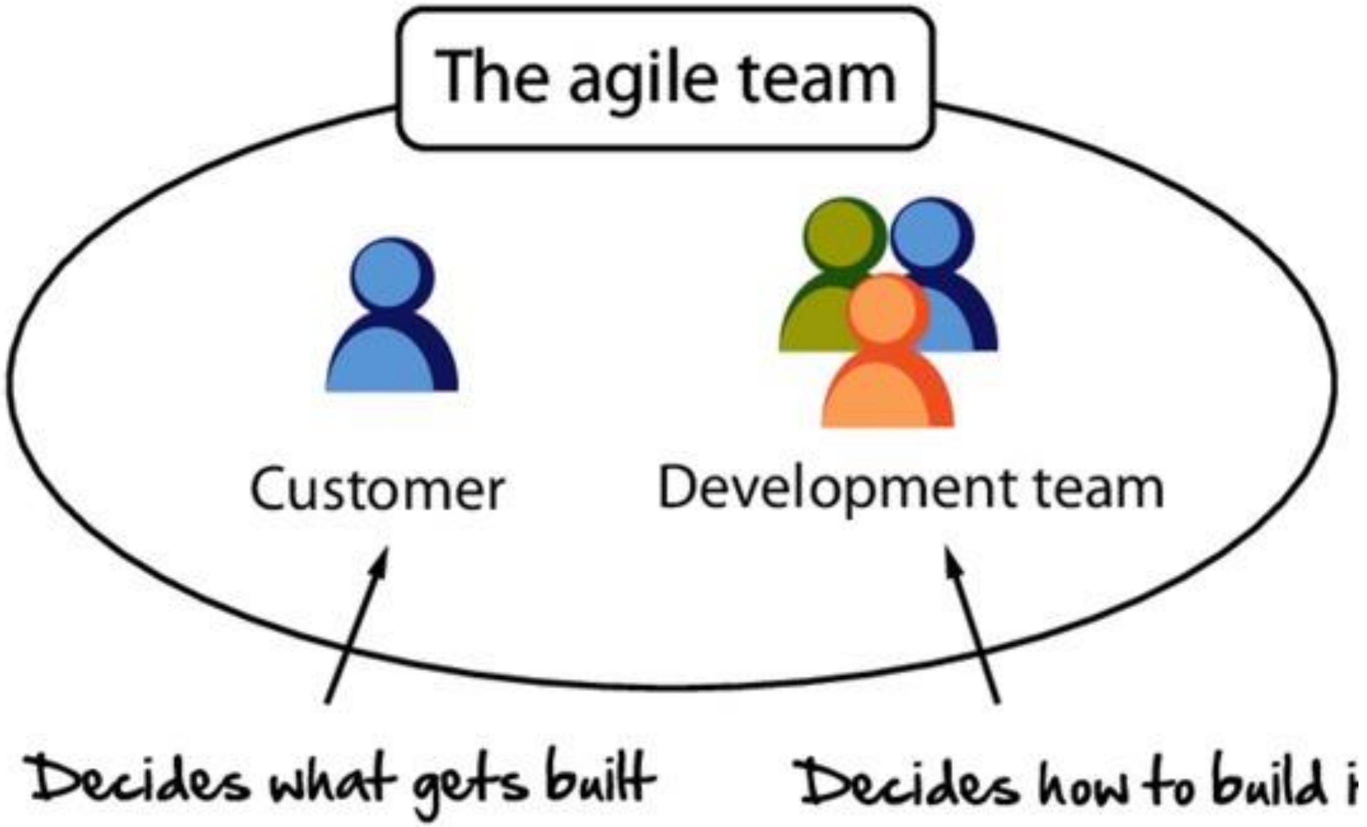
- **Agile testing** is a software testing practice that follows the principles of agile software development.
- Agile testing involves all members of a cross-functional agile team, with special expertise contributed by testers, to ensure delivering the business value desired by the customer at frequent intervals, working at a sustainable pace
- Specification by example is used to capture examples of desired and undesired behavior and guide coding.

AGILE ALL THE WAY

- Business people, programmers, testers, analysts decides together how best to improve the software
- Working together with all stakeholders
- Business Facing Test ~ Test that define the business experts' desired features and functionality
- Exploratory testing



TEAM


The agile team
Customer
Development team
Decides what gets built
Decides how to build i

CUSTOMER TEAM

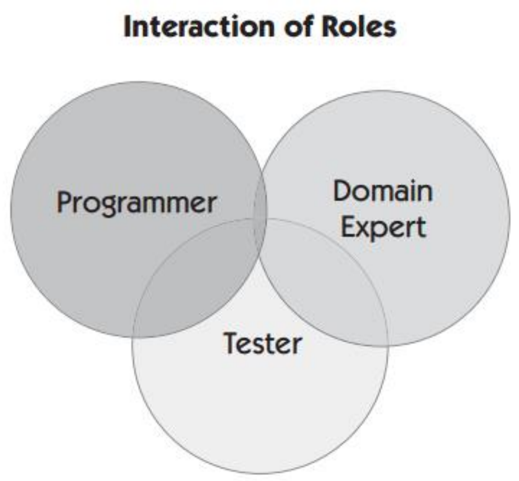
- Customer Team
  - Business Expert
  - Product Owners
  - Domain Experts
  - Product manager
  - Business analyst
  - SME
- Write stories and feature sets
- Write examples that drive coding in the form business facing test
- Tester help elicit requirement and examples and helping customer express their requirement as test

DEVELOPER TEAM

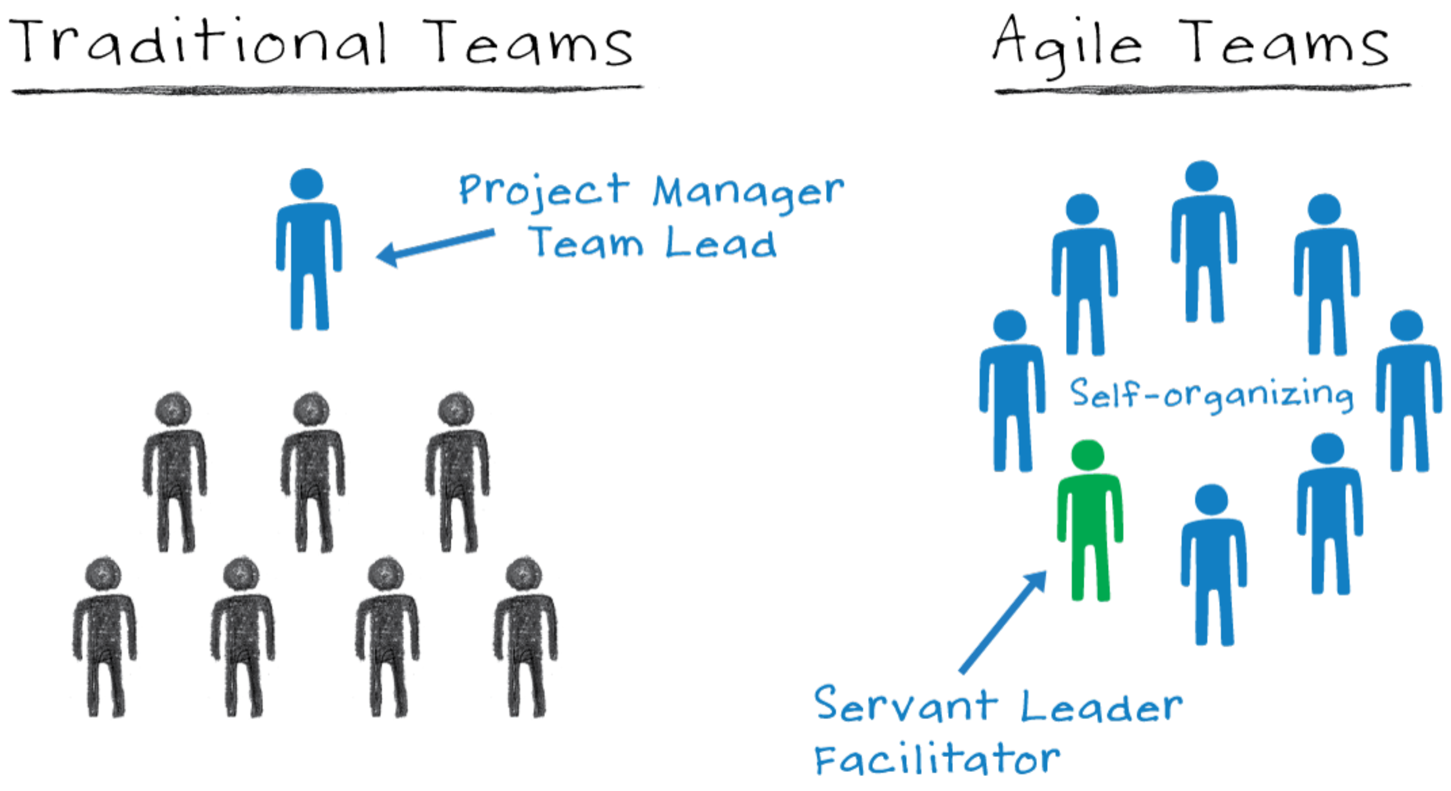
- Everyone that deliver codes
- Programmers, Sys Admin, Architects, DB Admin, Technical Writers, Security Specialist
- Full Stack Developers
- Testers also …

INTERACTION

- Customer Team will prioritize
- Developer team will determine how much work they can take on
- Work together in requirement as a tests and examples



TRADITIONAL VS AGILE



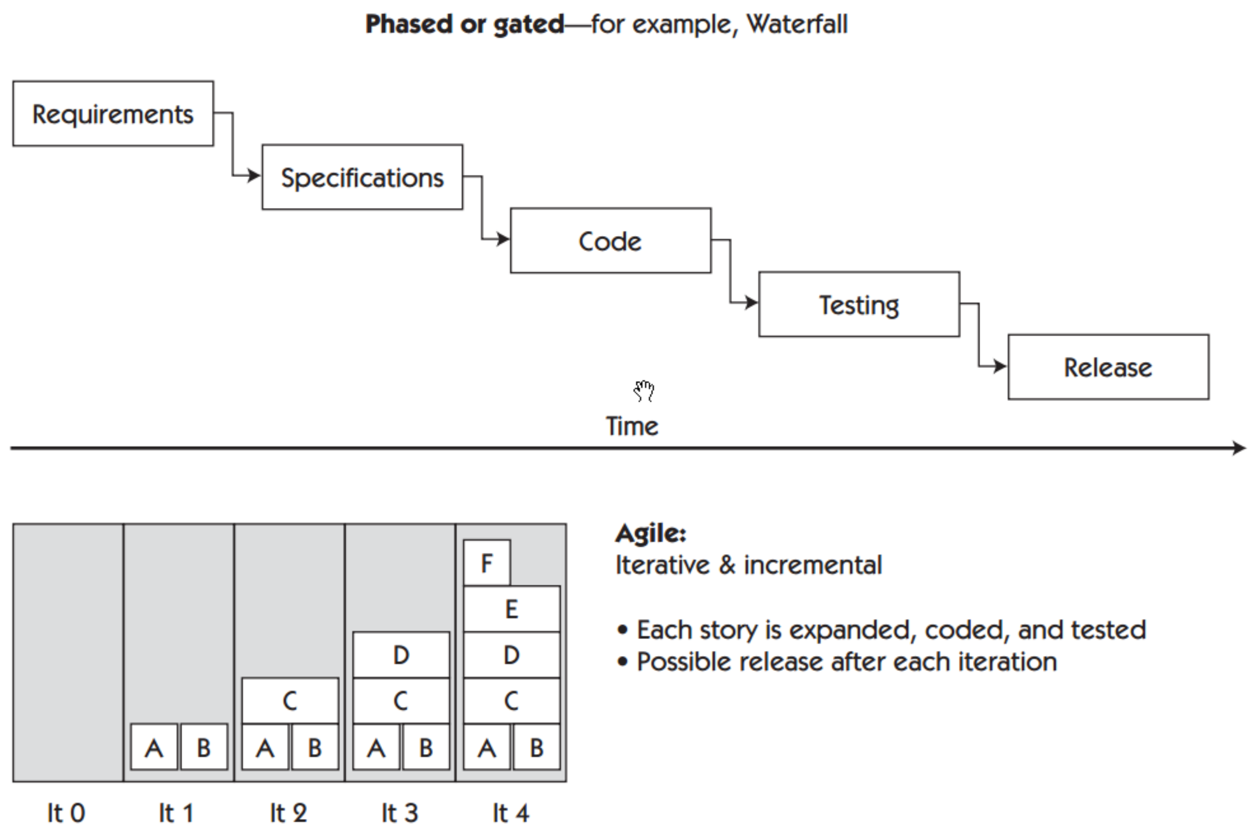
IN TRADITIONAL TEAM

- Rushed testing phase and a delayed release
- Development cycle is long
- Focused on making sure all the specified requirement are delivered in the final product
- Release can be postponed if the requirement is not met
- Tester use requirement documents to write test plan

IN AGILE TEAM

- Short iteration
- Hard for big organization
- Work closely with business and have detailed understanding of requirements
- Input on prioritizing features
- Don’t sit and wait for works
- Get up and look for ways to contribute

TRADITIONAL VS AGILE TESTING


Phased or gated—for example, Waterfall
Requirements
Specifications
Code
Testing
Release
Time
F
E
D
C
A
B
It 0
It 1
It 2
It 3
It 4
Agile:
Iterative & incremental
• Each story is expanded, coded, and tested
• Possible release after each iteration

AGILE TESTER ?

- a professional tester who embraces change, collaborates well with both technical and business people, and understands the concept of using tests to document requirements and drive development
- Developer that become test-infected
- Exploratory tester

AGILE TESTING MINDSET

- Continuously looking for the way to produce good high quality software
- Gather and share information. Don’t limit themselves only on testing issues
- Work with customer and product owner
- Work together with developer to create business facing test or specification by example. Turn specification to executable test

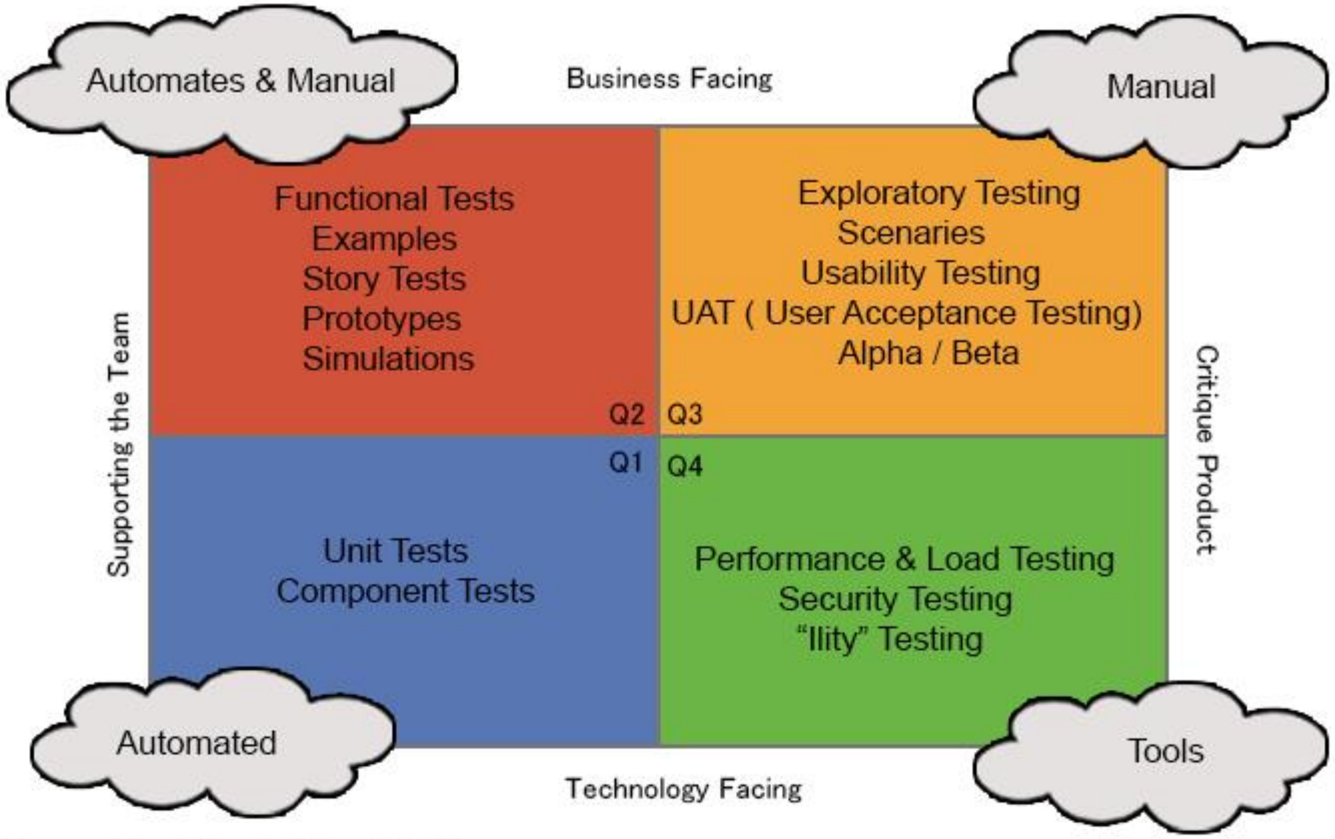
10 PRINCIPLES AGILE TESTER

- Provide continuous feedback
- Deliver value to customer
- Enable face to face communication
- Have courage
- Keep it simple
- Practice continuous improvement
- Respond to change
- Self-organize
- Focus on people
- Enjoy

DAILY LIFE

- Completed User story => Test
- Committed user story, work on acceptance criteria and specification by example
- Don’t wait for something to happen. Find something to work on.

The Agile Testing Quadrants



Source: Lisa Crispin, Brian Marick

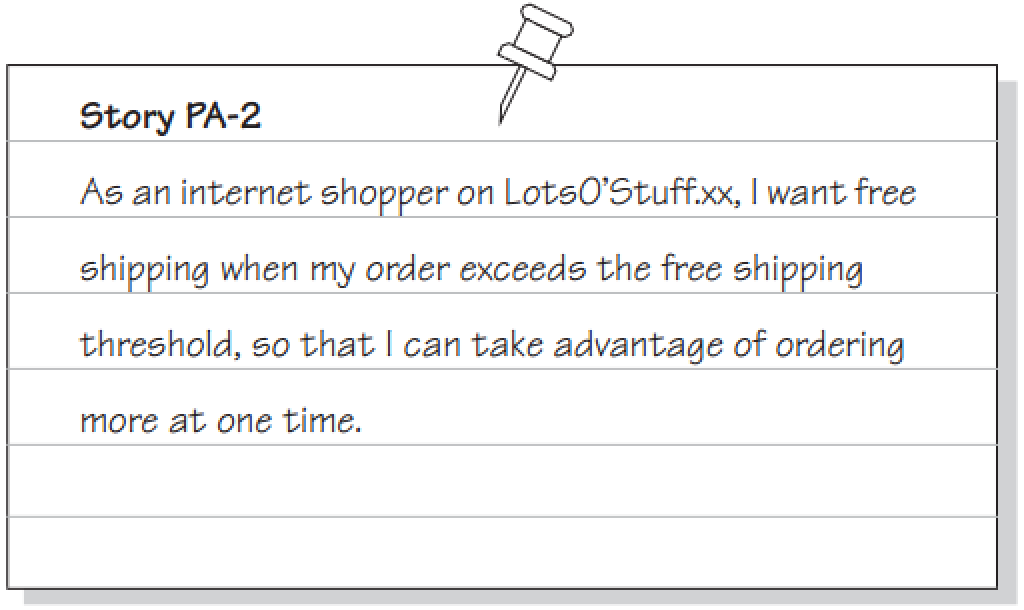
CHECKLIST

- Are we using unit and component tests to help us find the right design for our application?
- Do we have an automated build process that runs our automated unit tests for quick feedback?
- Do our business-facing tests help us deliver a product that matches customers' expectations?
- Are we capturing the right examples of desired system behavior? Do we need more? Are we basing our tests on these examples?
- Do we show prototypes of UIs and reports to the users before we start coding them? Can the users relate them to how the finished software will work?
- Do we budget enough time for exploratory testing? How do we tackle usability testing? Are we involving our customers enough?
- Do we consider technological requirements such as performance and security early enough in the development cycle? Do we have the right tools to do "ility" testing?

QUADRANT 1

- The term *test-driven development* misleads practitioners who don't understand that it's more about design than testing.
- Code developed test-first is naturally designed for testability.
- Quadrant 1 activities are all aimed at producing software with the highest possible internal quality.

QUADRANT 2

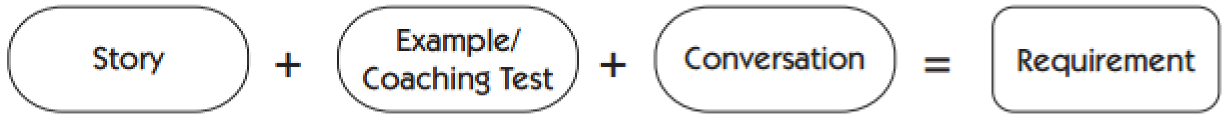


Story PA-2

As an internet shopper on LotsO'Stuff.xx, I want free shipping when my order exceeds the free shipping threshold, so that I can take advantage of ordering more at one time.

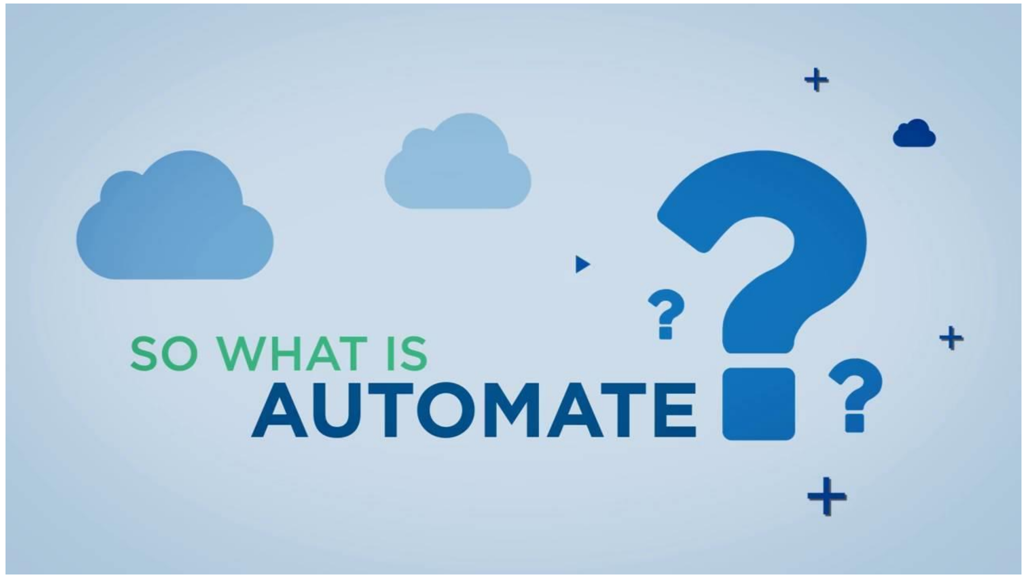
QUADRANT 2

- Start with story
- Cover business requirement by example
- Customer test drive coding in term of executable specification or specification by example.



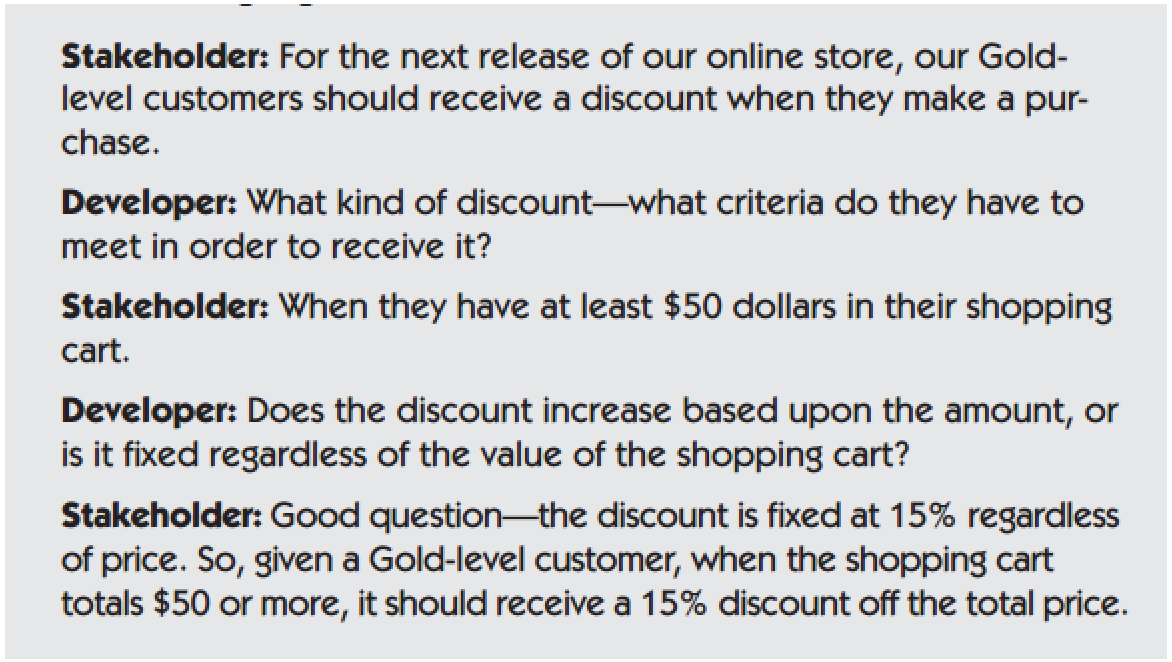
SPECS BY EXAMPLES

- *There are 5 items on a page. I want to select item 1 for $20.25 and put it in the shopping cart. I click to the next page, which has 5 more items. I select a second item on that page for $5.38 and put it in my shopping cart. When I say I'm done shopping, it will show both the item from the first page and the item from the second page in my shopping cart, with the total of $25.63*



USER STORY

- *As a (role), I want (function) so that (business value).*



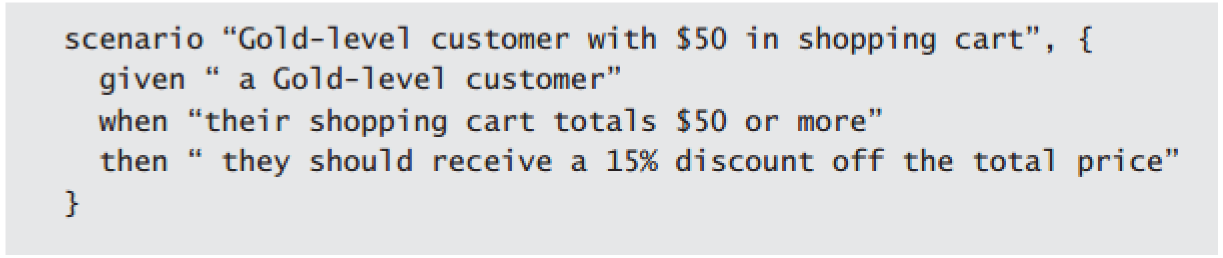
Stakeholder: For the next release of our online store, our Gold-level customers should receive a discount when they make a purchase.

Developer: What kind of discount—what criteria do they have to meet in order to receive it?

Stakeholder: When they have at least $50 dollars in their shopping cart.

Developer: Does the discount increase based upon the amount, or is it fixed regardless of the value of the shopping cart?

Stakeholder: Good question—the discount is fixed at 15% regardless of price. So, given a Gold-level customer, when the shopping cart totals $50 or more, it should receive a 15% discount off the total price.



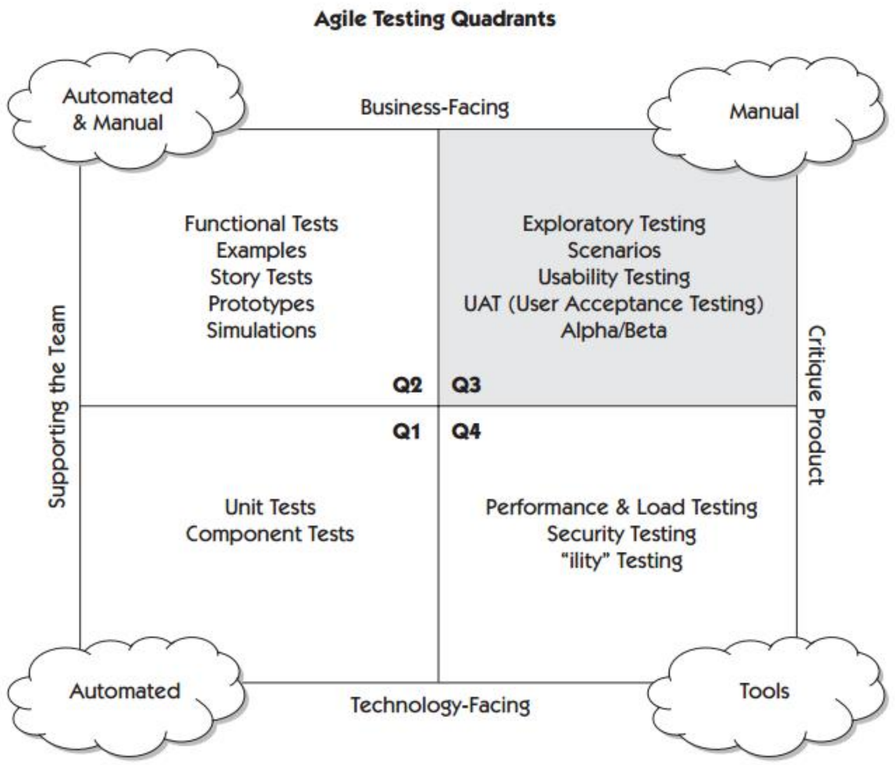
```
scenario “Gold-level customer with $50 in shopping cart”, {
  given “ a Gold-level customer”
  when “their shopping cart totals $50 or more”
  then “ they should receive a 15% discount off the total price”
}
```

EXECUTABLE SPECS

```
scenario "Gold-level customer with $50 in shopping cart", {
  given "a Gold-level customer", {
        customer = new GoldCustomer()
        }
  when "their shopping cart totals $50 or more", {
        customer.shoppingCart << new Item("widget", 50.00)
  }
  then "they should receive a 15% discount off the total price" , {
        customer.orderPrice.shouldBe 42.50
        }
}
```

QUADRANT 3

- It’s difficult to automate business-facing tests that critique the product, because such testing relies on human intellect, experience, and instinct



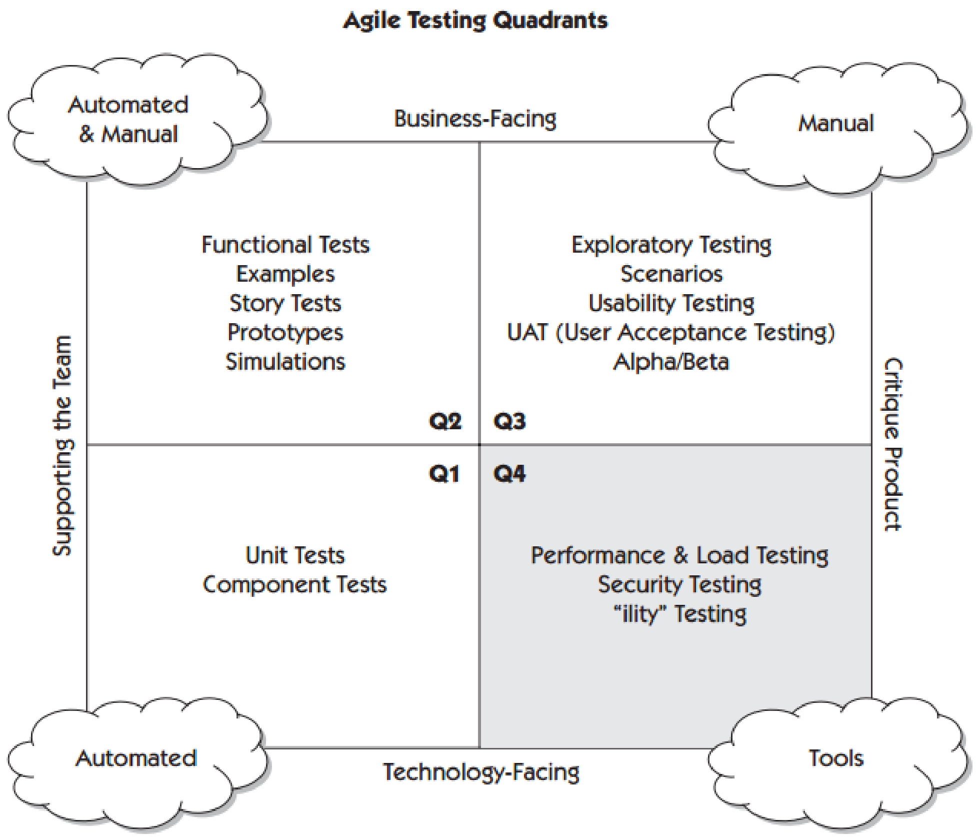
EXPLORATORY TESTING

- It is a sophisticated, thoughtful approach to testing without a script, and it enables you to go beyond the obvious variations that have already been tested
- Combines learning, test design, and test execution into one test approach.
- As you test, you learn more about the system under test and can use that information to help design new tests
- Exploratory testing starts with a charter of what aspects of the functionality will be explored
- Is systematic, but pursues "smells" (anomalies, pieces that aren't consistent)
- Learns to recognize problems through the use of Oracles (principle or mechanism by which we recognize a problem)

EXPLORATORY TESTING

- Chooses a theme or role or mission statement to focus testing
- Time-boxes sessions and side trips
- Thinks about what the expert or novice user would do
- Explores together with domain experts
- Checks out similar or competitive applications

QUADRANT 4


Agile Testing Quadrants
Automated & Manual
Business-Facing
Manual
Functional Tests
Examples
Story Tests
Prototypes
Simulations
Exploratory Testing
Scenarios
Usability Testing
UAT (User Acceptance Testing)
Alpha/Beta
Supporting the Team
Critique Product
Q2
Q3
Q1
Q4
Unit Tests
Component Tests
Performance & Load Testing
Security Testing
"ility" Testing
Automated
Technology-Facing
Tools

WHY AUTOMATE ?

- Manual testing takes too long.
- Manual processes are error prone
- Automation frees people to do their best work.
- Automated regression tests provide a safety net.
- Automated tests give feedback early and often
- Tests and examples that drive coding can do more.
- Tests provide documentation.
- Automation can be a good return on investment

BE AGILE !