Agile Testing: A Beginner's Guide

What is Agile Testing?

Unlike the WaterFall method, Agile Testing can begin at the start of the project with continuous integration between development and testing. Agile Testing is not sequential (in the sense its executed only after coding phase) but continuous.

Agile team works as a single team towards a common objective of achieving Quality. Agile Testing has shorter time frames called iterations (say from 1 to 4 weeks). This methodology is also called release, or delivery driven approach since it gives a better prediction on the workable products in short duration of time

In this article we will discuss about

Test Plan for Agile.

Agile Testing Strategies.

The Agile Testing Quadrant.

QA challenges with agile software development.

Risk of Automation in Agile Process.

Test Plan for Agile

Unlike waterfall model, in an agile model, test plan is written and updated for every release. The agile test plan includes types of testing done in that iteration like test data requirements, infrastructure, test environments and test results. Typical test plans in agile includes

1) Testing Scope

2) New functionalities which are being tested

3) Level or Types of testing based on the features complexity

4) Load and Performance Testing

5) Infrastructure Consideration

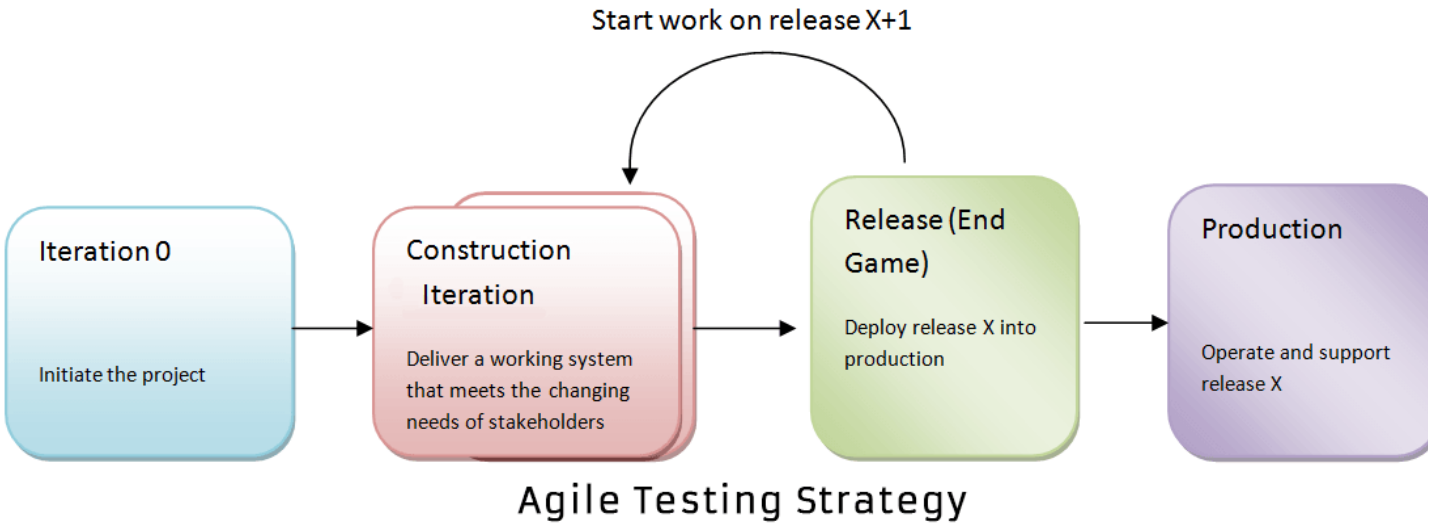
6)Mitigation or Risks Plan

7) Resourcing

8) Deliverables and Milestones

Agile Testing Strategies

Agile testing life cycle spans through four stages



Agile Testing Strategy

(a) Iteration 0

During first stage or iteration 0, you perform initial setup tasks. It includes identifying people for testing, installing testing tools, scheduling resources (usability testing lab), etc. The following steps are set to achieve in Iteration 0

a) Establishing a business case for the project

b) Establish the boundary conditions and the project scope

c) Outline the key requirements and use cases that will drive the design trade-offs

d) Outline one or more candidate architectures

e) Identifying the risk

f) Cost estimation and prepare a preliminary project

(b) Construction Iterations

The second phase of testing is Construction Iterations, the majority of the testing occurs during this phase. This phase is observed as a set of iterations to build an increment of the solution. In order to do that, within each iteration, **the team implements** a hybrid of practices from XP, Scrum, Agile modelling, and agile data and so on.

In construction iteration, agile team follows the prioritized requirement practice: With each iteration they

take the most essential requirements remaining from the work item stack and implement them.

Construction iteration is classified into two, confirmatory testing and investigative testing. **Confirmatory testing concentrates** on verifying that the system fulfills the intent of the stakeholders as described to the team to date, and is performed by the team. While the investigative testing detects the problem that confirmatory team have skipped or ignored. In Investigative testing, tester determines the potential problems in the form of defect stories. Investigative testing deals with common issues like integration testing, load/stress testing and security testing.

Again for, confirmatory testing there are two aspects**developer testing** and **agile acceptance testing. Both of them** are automated to enable continuous regression testing throughout the lifecycle. Confirmatory testing is the agile equivalent of testing to the specification.

Agile acceptance testing is a combination of traditional functional testing and traditional acceptance testingas the development team, and stakeholders are doing it together. While developer testing is a mix of traditional unit testing and traditional service integration testing. Developer testing verifies both the application code and the database schema.

(c) Release End Game Or Transition Phase

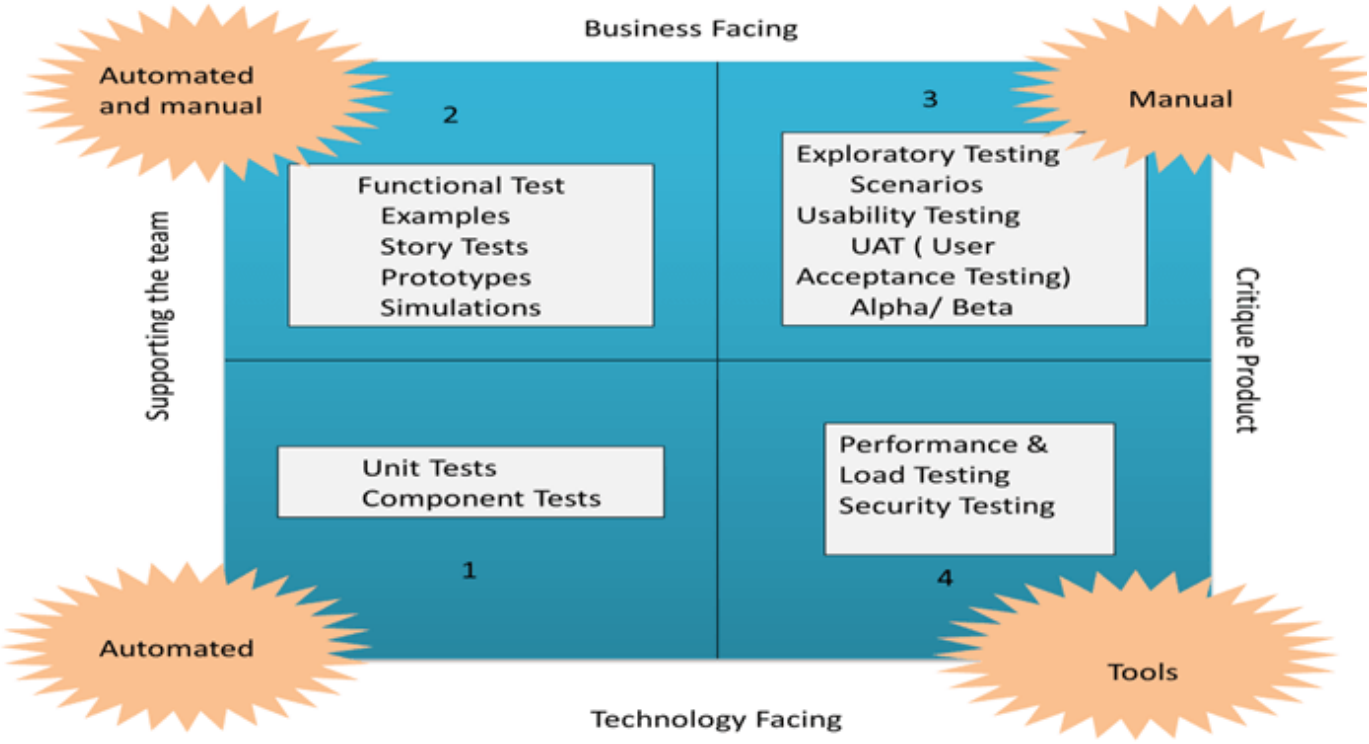
The goal of "Release, End Game" is to deploy your system successfully into production. The activities include in this phase are training of end users, support people and operational people. Also, it includes marketing of the product release, back-up & restoration, finalization of system and user documentation.

The final testing stage includes full system testing and acceptance testing. In accordance to finish your final testing stage without any obstacles, you should have to test the product more rigorously while it is in construction iterations. During the end game, testers will be working on its defect stories.

(d) Production

After release stage, the product will move to the production stage.

The Agile Testing Quadrant



The agile testing quadrant separates the whole process in four Quadrants and helps to understand how agile testing is performed.

a) **Agile Quadrant I** The internal code quality is the main focus in this quadrant, and it consists of test cases which are technology driven and are implemented to support the team, it includes

1. Unit Tests

2.Component Tests

b) **Agile Quadrant II** It **contains** test cases that are b**usiness driven and are implemented** to support the team. This Quadrant focuses on the requirements. The kind of test performed in this phase is

1. Testing of examples of possible scenarios and workflows

2. Testing of User experience such as prototypes

3. Pair testing

c) **Agile Quadrant III**– This quadrant provide feedback to quadrants one and two. The test cases can be used as the basis to perform automation testing. In this quadrant, many rounds of iteration reviews are carried out which builds confidence in the product. The kind of testing done in this quadrant is

1. Usability Testing

2. Exploratory Testing

3. Pair testing with customers

4. Collaborative testing

5. User acceptance testing

d) **Agile Quadrant IV– This quadrant concentrates** on the non-functional requirements such as performance, security, stability, etc. With the help of this quadrant, the application is made to deliver the non-functional qualities and expected value.

1. Non-functional tests such as stress and performance testing

2. Security testing with respect to**authentication and**hacking

3. Infrastructure testing

4. Data migration testing

5. Scalability testing

6. Load testing

QA challenges with agile software development

a) Chances of error are more in agile, as documentation is given less priority, eventually puts more pressure on QA team

b) New features are introduced quickly, which reduces the available time for test teams to identify whether the latest features are according to the requirement and does it truly address the business suits

c) Testers are often required to play a semi-developer role

d) Test execution cycles are highly compressed

e) Very less time to prepare test plan

f) For regression testing, they will have minimal timing

g) Change in their role from being a gate-keeper of quality to being a partner in Quality

h) Requirement changes and updates are inherent in an agile method, becoming the biggest challenge for QA

Risk of Automation in Agile Process

- Automated UI provides a high level of confidence, but they are slow to execute, fragile to maintain and expensive to build. Automation may not significantly improve test productivity unless the testers know how to test
- Unreliable tests are a major concern in automated testing. Fixing failing tests and resolving issues related

to brittle tests should be a top priority in order to avoid false positives

- If the automated test are initiated manually rather than through CI (Continuous Integration) then there is a risk that they are not regularly running and therefore may cause failing of tests
- Automated tests are not a replacement for an exploratory manual testing. To obtain the expected quality of the product, a mixture of testing types and levels is required
- Many commercially available automation tools provide simple features like automating the capture and replay of manual test cases. Such tool encourages testing through the UI and leads to an inherently brittle and difficult to maintain tests. Also, storing test cases outside the version control system creates unnecessary complexity
- In order to save time, many times automation test plan is poorly planned or un-planned which results in the test fail
- Test set up and tear down procedures are usually missed out during test automation, while Performing manual testing, test set up and tear down procedures sounds seamless
- Productivity metrics such as number of test cases created or executed per day can be terribly misleading, and could lead to making a large investment in running useless tests
- Members of the agile automation team must be effective consultants: approachable, cooperative, and resourceful, or this system will quickly fail
- Automation may propose and deliver testing solutions that require too much ongoing maintenance relative to the value provided
- Automated testing may lack the expertise to conceive and deliver effective solutions
- Automated testing may be so successful that they run out of important problems to solve, and thus turn to unimportant problems.

Conclusion

Agile testing involves testing as early as possible in software development life cycle. It demands high customer involvement and testing code as soon as it becomes available. The code should be stable enough to take it to system testing. Extensive regression testing can be done to make sure that the bugs are fixed and tested. Mainly, Communication between the teams makes agile testing success!!!

Do you have any tips or experiences to share for Agile Testing? Do leave a comment below-