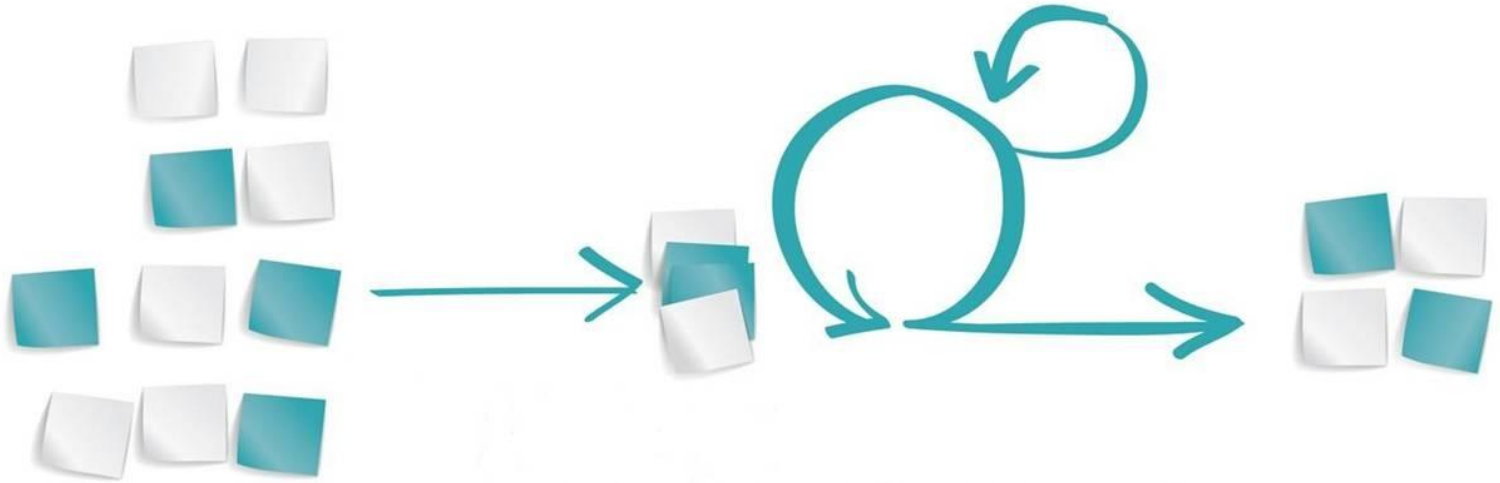
intelliware
software development

Agile



QA in an Agile Environment

QA in Agile – Introduction

Software quality assurance (SQA) is defined as *a planned and systematic approach to the evaluation of the quality of and adherence to software product standards, processes, and procedures.*[1] This systematic approach is actually quite different in Agile and non-Agile environments. There are several key differences in these approaches that we'll address in this presentation.

What You'll Learn in this Presentation:

- The role of QA on an Agile team
- The testing that QA should focus on

[1] https://jira.atlassian.com/secure/attachment/17146/sqa+activities.txt

Why is QA in Agile so different from Traditional QA?

Agile has two tenets in particular that muddy the waters around testing:

1. Teams are cross-functional and self-organizing
   - Everyone is expected to be able to help deliver any story (minimal specialization)
   - Everyone is responsible for ensuring high quality deliverables
2. Iterative development - every sprint's deliverable is a potential release candidate
   - Testing must be ongoing through the entire development lifecycle
   - Balance must be maintained between testing new functionality and system regression testing

Waterfall vs. Agile: The Role of QA

There are several other important differences between QA in a traditional (Waterfall) environment and QA in an Agile environment.[2]

Agile	Waterfall
Black box and white box testing, deep knowledge of internal work ings of internal workings of the application.	Black box testing, no need for deep knowledge of internal workings of the application.
Main function is to help produce killer applications.	Main function is to certify the quality of the product.
Work in parallel with development, testing as soon as new source code is produced.	Work in branches at the end of milestones.
Heavily based on automated testing.	Not much need for automated testing, if any. Only some UI automated testing is performed.
Integrated with the development team. There is only one team.	A completely separate team from development.
Key role interacting with the business. They make sure that the expectation from the customer (acceptance criteria) are met.	Not much interaction with the business. Their purpose is to make sure that the application meets whatever is specified in the requirements document.

[2] https://dzone.com/articles/waterfall-vs-agile-qa-management

Waterfall vs. Agile: Management's View of QA

Management also sees QA differently in an Agile environment.[3]

Agile	Waterfall
With regards to development and QA, focus on empowerment.	With Regards to development and QA, focus on controlling.
With regards to business, focus on producing the best application for the customer.	With regards to business focus on producing contracts for business requirements, detailed architecture and planning.
Manage basic agile processes: Scrum, Lean, XP…	Manage complex processes: RUP, CMMI

[3] https://dzone.com/articles/waterfall-vs-agile-qa-management

QA ON AGILE TEAMS

Where do QA team members fit in?

In short, through the entire process!

1. Requirements generation
2. Estimations
3. Planning
4. Documentation
5. Day-to-day sprint execution
6. Defining “done”
7. And of course, testing!

Quality Assurance

Quality Control

1. Requirements Generation

- QA can assist the BA / Product Owner with writing story cards
  - Identify missing user stories
  - Identify what is out of scope
  - Identify dependencies between user stories
- QA can assist the BA / Product Owner with detailed story documents
  - Identify edge cases
  - Generate Acceptance Criteria
  - Identify gaps in details on the detailed story documents
- Typically, QA team members know the ins and outs of the whole system better than anyone else on the team!

2. Estimations

- Epic Estimations
  - Identify functionality the developers may not have considered
  - Provide overall system knowledge, particularly around interdependencies
- Story Estimations
  - Identify edge cases
    - Developers often focus on “Happy Path”
  - Identify potential impact on other parts of the system
  - Identify potential automated regression tests

3. Planning

- Release Planning
  - Identify sprints with significant regression test time
  - Plan releases such that expected QA workload is sustainable
- Sprint Planning
  - Identify user stories with excessive QA requirements
  - Plan sprints such that expected QA workload is sustainable
- Remember – Story point estimates are based on development effort
  - Some user stories have QA requirements which exceed the norm; the QA team is responsible for pointing this out to avoid QA overload during the sprint

4. Documentation

- Test Cases
  - Based off of acceptance criteria
  - Contains specific details and/or test data
- Detailed Story Document
  - Works with BA / Product Owner to ensure that any issues or gaps are captured
  - Sometimes responsible for writing acceptance criteria
- User Guides
  - Sometimes responsible for writing user guides (due to their overall system knowledge)

5. Day-to-Day Sprint Execution

- Communication
  - Regular interaction with developers and BA / Product Owner
  - Comments on tracking system items (epics, stories, bugs, etc.) for historical reference
- Participation in core Agile meetings
  - Kickoff, sprint planning, stand-up, demos, retrospectives, estimations
- Parallel Testing
  - On days when user stories are completed, emphasis is on those
  - On other days, emphasis is on system regression testing
  - On some projects, writing of automated regression tests is a QA responsibility

6. Defining Done

- Acceptance Criteria
  - Ensuring the acceptance criteria is comprehensive
  - Ensuring edge cases are identified and covered in the user story
- Regression
  - Verifying new functionality has not impacted existing functionality
- Story Workflow Tracking
  - Moving user stories to COMPLETE after successful test execution
  - Assigning user stories back to developers after failed test execution

7. Testing

- Work in parallel with development
  - Completion of testing of most user stories by end of sprint
  - Some stories will complete testing in following sprint
- Identify and report issues
  - Issues should be documented and communicated to the BA
  - BA works with the customer or product owner to determine if they need to be resolved in the current sprint
  - Re-test issues that have been fixed
- Balance their time between new functionality and regression testing
  - Details to follow…

Parallel QA

- QA should take place in parallel with development
  - Stories should be tested as they are completed during the sprint
  - Prevents overload of the QA team at the end of the sprint (which is typically what happens on traditional projects)
  - Reduces the risk of surprises
- QA may not be completed by end of sprint – that's okay
  - QA for stories completed near the end of a sprint may spill into the next sprint
  - This is not unusual and ought to be accommodated

Test Automation

- Developers are responsible for writing automated unit tests with every story
- Developers should also write automated integration tests
- On automated integration tests:
  - Helps regularly regression test the system
  - Harder to write (and slower to run) than unit tests
  - Should not use mocking (or stubs, or fakes) in most cases
  - Often requires a test-specific database configuration
  - Many tools exist ,such as Selenium or Protractor, to assist

Examples of Developer Automated Testing versus QA Manual Testing

- Developers focus on “System Logic”
  - Does known input produce expected output?
  - Are logical pathways followed at the correct time?
  - Are calculations producing expected results?
  - Does the system handle zero, negative, null, and extreme values correctly?
  - Are exceptions thrown at appropriate times? And handled correctly?
- QA team members focus on “System Behaviour”
  - If a user does ‘A’, ‘B’, and ‘C’, does the system respond with ‘D’ as expected?
  - Does the system respond as expected in different states? Is it repeatable?
  - Is the user experience satisfactory? Even under load?
  - Can unexpected user interaction “break” the system?
  - If a user interacts with system ‘A’, does system ‘B’ act appropriately?
  - Are there inconsistencies between different parts of the system?

Testing Example #2: Compatible with Manual Testing

- Getting around system security
  - User logs into the system,
  - then navigates to a secure area,
  - then logs out of the system, and
  - then clicks the back button to return to the secure area.
- An automated test can work, but faces the following challenges:
  - Time consuming to write and to execute on every build
  - Can fail due to network or database instability, causing the build to fail
  - Numerous potential scenarios to be considered
    - Different browsers, different secure areas, different cookie settings, etc.
- QA team members can try many combinations quickly and easily!

Balancing New Functionality & System Regression Testing

- In Agile, QA must balance testing new functionality versus system regression testing
  - When stories are completed the focus is on testing new functionality; confirming that acceptance tests work as expected
  - At other times, the focus shifts towards regression testing
    - Manual testing will be focused on boundary and behavioural testing that can't be easily automated
    - Automated regression tests can be written either by developers or QA (depends on team composition and workload); they should run on every build

QA CHALLENGES IN AGILE

1. Development time issues

- A story takes much longer than expected
  - Reduces available QA time
- Developers focus on too many concurrent stories
  - Results in many days of no new story completions
  - Followed by multiple stories completing development concurrently

2. Environment Complications

- Test server(s) having issues
  - Time needs to be spent fixing the server(s)
  - QA needs to wait for someone else to fix them
- Test data corruption
  - Time needs to be spent tracking down data problems
  - Time needs to be spent fixing data problems
- External systems being down
  - While developers can mock these scenarios in unit tests, QA needs the actual systems working for their testing
  - Often requires waiting for systems to be restored

3. Unexpected Workload Increase

- Sprint scope changes
  - New test cases need to be written
  - May have to regression test based on story removal
- Large or high priority bugs
  - Reproducing the issue
  - Re-testing after development fix

Potential Remedies to QA Challenges in Agile

- 1. Organize development work
  - Schedule higher risk stories early in the release (and sprint)
  - Have developers focus on fewer concurrent stories
  - Plan themed sprints to help reduce regression testing requirements
  - Avoid including too many stories (in sprint planning) with high QA requirements
- 2. Share responsibilities
  - BA and/or developers can perform some QA duties if needed
  - Regression testing (or production issue testing) can be done by another team
  - Developers can write additional automated tests

About Intelliware Development Inc.

Intelliware is a custom software, mobile solutions and product development company headquartered in Toronto, Canada. Intelliware is a leader in Agile software development practices which ensure the delivery of timely high quality solutions for clients. Intelliware is engaged as a technical partner by a wide range of organizations in sectors that span Financial Services, Healthcare, ICT, Retail, Manufacturing and Government.

- /company/intelliware-development-inc-
- /intelliware.inc
- /intelliware_inc
- /GooglePlusIntelliware
- /www.intelliware.com

Learn more about Agile in our *Agile Methodology Series*