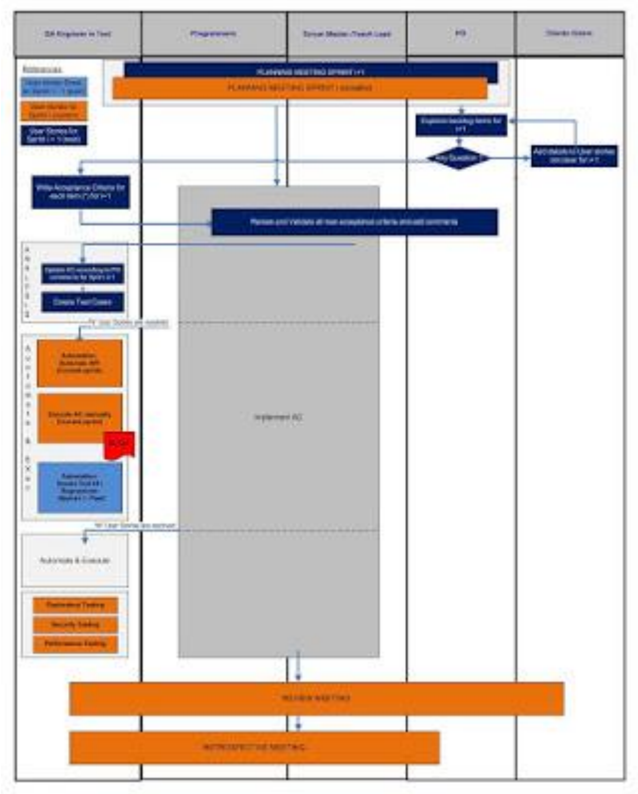
Testers Working in an Agile Team

Testers Working in an Agile Team

Today's tester role is more versatile and calls on a wide range of skills. Now testers have tasks throughout the sprint, and they also program — yes! — as regular programmers. (I don't say as developers, as all participants in the team are part of the development: the business analyst (BA), the designer, the tester, the tech writer and the programmers.) Testers don't need to be specialists in any one language; the more languages they know, the better the solutions they will find. A good engineer in testing should be able to adapt his or her knowledge according to the needs.

So let's examine in greater depth the testers working in a team and their task distribution along the sprint. (This process localizes tester tasks, though it could be missing tasks for other roles).



In an Agile method, we could have previous, current, and next sprints. Our tester will have tasks related to each of these stages.

Tester tasks

Be part of the planning meeting for the current sprint. The tester will be in charge of providing estimations about data creation, test cases/acceptance test design, test execution, framework design/improvements, scripting tasks, setup environments, etc.

Be part of the planning meeting for the next sprint. The tester needs to validate whether he will create any complex environments, or any major improvements for the automation framework, before starting the next sprint.

Write acceptance criteria for each item for the next sprint. Testers create acceptance criteria for the user stories for the next sprint, helping the BA (if the team includes BAs) by giving suggestions from a QA perspective regarding standards, user experience, possible performance issues, and future bugs. Depending how Agile we are, we will have acceptance criteria and/or some test cases (this will also depend on the company, since sometimes you need to show some evidence about execution as test cases). Creating acceptance criteria can be a very creative task, but we need to validate how possible they are and whether we are on the same page as the PO, so validation from the team could be necessary.

Update acceptance criteria according to PO comments and create test cases for the next sprint. Once we receive team feedback, we update the acceptance criteria and start creating test cases (or acceptance tests if we're working with application programming interfaces, or APIs).

Automate APIs (current sprint). If the application is divided in tiers, we will have APIs, or services. We could start scripting before programmers finish coding for the current sprint. At first, all our tests will fail, but gradually as programmers finish their work, our suite will pass. Automating them is easier than automating user interfaces, and they're easier to maintain as well. Automation can be achieved using tools such as SoapUI, Jmeter, etc.

Execute acceptance criteria manually (current sprint). Test cases should be executed manually at least once when each user story is done. Manual verification will give us a set of bugs, an idea about limitations and how critical a test case is. On the other hand, this practice will determine the automation candidates for the next sprint.

Automate smoke test UI/regressions (previous sprint). Automating the UI once the application is stable could help you avoid many headaches. In this stage, we will start automating the smoke test for the current features done or regressions for complex modules. In future iterations we will update the smoke test, adding the new features.

Exploratory testing. This kind of testing will find bugs that no automated test will ever find. There is nothing that compares with the tester's creativity, and we should apply this kind of testing once every sprint.

Be part of the review meeting for the current sprint. Testers or the BA could lead the internal/external demo. The goal is to show commitment to current sprint, and ideally we shouldn't show any issue live! So a tester could choose the safer path and point out the known issues as well — an extra slide could be enough. (Testers and the BA speak human language, while programmers don't, according to popular belief. . . .)

Be part of the retrospective meeting for the current sprint. In this meeting, everyone will participate, identifying things well done, things to improve, and actions to apply.