Agile & Scrum: What are these methodologies and how will they impact QA/testing roles?

By Marina Gil Santamaria
Summer, 2007

Introduction

Agile Software Development is a methodology for undertaking software development projects in which incremental functionality is released in smaller cycles, and work is performed in a highly collaborative manner by self-organizing teams that embrace and adapt changes to ensure that customer's needs are truly met. Agile Software Development is not new, in fact it was introduced in the 1990s as a way to reduce costs, minimize risks and ensure that the final product is truly what customers requested. The idea behind the Agile approach is that instead of building a release that is huge in functionality (and often late to market), an organization would adapt to dynamic changing conditions by breaking a release into smaller shorter cycles of 1 to 6 weeks. Each cycle is called an iteration, or **sprint**, and it's almost like a miniature software project of its own, because it includes all of the tasks necessary to release the incremental new functionality. In theory, at the end of each sprint, the product should be ready for a GA release. Agile methodology emphasizes real-time communication, preferably face-to-face, versus written documents and rigid processes. In addition, one of the most broadly applicable techniques introduced by the agile processes is to express product requirements in the form of user stories. Each user story has various fields including an "actor", a "goal" or task that they need to perform, an explanation of "why" it is needed and the associated value, and a corresponding "priority".

Most agile teams include all the people necessary to release software. At a minimum, this includes programmers and the group or team they are developing the application for, often referred to as their "customers" (customers are the people who define the product; they may be product managers, business analysts, or actual customers). Typically an agile team will also include a ScrumMaster, testers, interaction designers, technical writers, and managers.

What is **scrum**? Scrum is really a project management methodology to facilitate agile software development, and enable the creation of self-organizing agile teams. A **ScrumMaster** is like a traditional project manager in the sense that he/she oversees the centralization of team communication, requirements, schedules and progress. But it is also very different because his/her main responsibility is to facilitate team communications and provide guidance and coaching, while removing impediments to the ability of the team to deliver its goals. Unlike a traditional project manager, the ScrumMaster doesn't direct the team, because an agile team is based on the philosophy that a team member is committed to the other team members, not to a management authority.

Phases of an Agile Development Project using Scrum

Agile can be customized to fit each corporation in terms of size, iteration time, experience, etc, but typically an agile project will have these phases and milestones.

1) **Kickoff meeting**. Although this may seem routine for any project, with an agile development project this is a key element for getting the project launched. The goal of this meeting is to get everybody on the team together to review the **product backlog** (which is the master list of all requirements desired in the product that the product owner has drafted in the form of user stories), as well as the user personas (or the profile of each type of product user). In my opinion, this is a nicer and clearer way to introduce product requirements, because you really have more visibility into who is using the product, what are they trying to achieve and why, right from the beginning. A kickoff meeting usually lasts at least half a day with everybody together going over a "story writing workshop" -- in which stories are selected and then decomposed into programmable tasks and written in a white board together along with the time estimates for completion. If you have never seen a product backlog, you can check few samples here in QAZone . Sometime real customers are invited to the kickoff meeting as well to review and clarify the product backlog with the agile team.

2) The next step in the **sprint/iteration planning**, in which the team collectively decides the sprint goal and **sprint backlog** (list of prioritized work to be done for that particular sprint). While the team is collectively creating the sprint backlog, stories need to be broken into either sub-stories or smaller tasks. During this collective team exercise you can really see the differences in project management (at least if you have a more rigid and formal waterfall like type of background), because there is no management authority that assigns tasks to team members. On an agile team all the members jointly associate level of difficulty to specific tasks, they can remove or add additional stories and/or tasks, and tasks are distributed among the team on a per volunteers basis. Unlike a traditional project manager function, the ScrumMaster role in this meeting is to maintain the backlog list in the meeting based on team feedback and consensus, make sure that nobody is volunteering for too many tasks to the point of overload, and facilitate the process of building personal commitment to the team.

3) Now that the Sprint planning is ready in the form of sprint backlog –which is dynamic and not set in stone, in fact it is very likely that it will adapt and change based on new stories, new tasks and/or impediments found throughout the iteration --, **scrum meetings** will be set at the same time and in the same place on a daily basis. If you have never attended a scrum meeting, these meetings are very dynamic in nature and fast, never more than 30 minutes, and ideally 10-15 minutes. The objective is to go around so each team member can answer 3 questions: what have I accomplished since the last meeting (developed, tested, written, etc), what will I be working on next, and what are the problems, if any, preventing me from accomplishing my goals. These meetings are very important to make sure that the team moves towards achieving their sprint goal, or adapts/evolves and changes priorities and tasks as needed if new stories, impediments or new scenarios are encountered.

4) At the end of the sprint or iteration, usually a **final acceptance meeting** takes place, which is typically done by presenting what the team has accomplished, and by delivering a demo to the customer or to a larger audience.

5) At the end of an iteration there is also a s**print retro meeting**, similar to a postmortem meeting at the end of other traditional projects, so the team gets together to evaluate what worked well, and what needs to be improved during their next iteration.

Top 3 things a QA professional should expect when an organization adopts Agile/Scrum development techniques

Agile and Scrum are really changing the way testing is perceived throughout the project. Testing is not a phase at the end; it really is integrated throughout the entire iteration cycle, and it goes hand in hand with programming tasks. In my experience, when comparing a testing role performed within an agile project, or when using more rigid and formal approaches, I have found that with agile methodologies there is:

1) Better communication and more collaboration among QA & development folks

Gone are the days of "give me requirements" and "I will give you bugs and reports back"…QA folks are involved in the project from the start – along with their development counterparts--, and they have access to the same information about product requirements and customer needs at the same time. This participation from the onset, combined with the fact that development and QA are part now of the same agile team, that they get together on a daily basis, and that they have full visibility into the tasks that each other is performing towards the overall success of the sprint, means better and more frequent communication among themselves. In addition, because the entire team meets everyday (development, QA, product management, etc) there are more opportunities for collaboration and more view points towards performing a particular task. Also the traditional "rivalry" that you may find among QA and development is eliminated because there is a single agile team now working to achieve a common goal.

2) A new "peer to peer" relationship between development and QA personnel

You should be prepared to "speak up" much more. Agile methodologies are all about building self-organized teams, and the voice of a QA engineer/tester carries the same weight than a developer. Think about it. In the daily scrum meeting each team member gets asked about their accomplishments (testing, developing, writing product documentation, etc), future plans, and obstacles, treating all of the members as equal partners. On an agile team the question of "how are we going to test it", is as important as “how are going to build it”. In addition, because testers tend to be exceptionally good at clarifying requirements and identifying alternative scenarios, (especially when they have full visibility into product requirements and customer needs), they provide valuable input on design and architectural decisions throughout the project, right from the beginning. And these contributions translate into more respect and appreciation from their development counterparts.

3) Looking for ways to optimize testing efforts will be a “must”

You really need to think about automation, and planning and performing your testing efforts very efficiently. With shorter development cycles of typically no more than 6 weeks, and with builds being released all the time, testing efforts really need to be optimized as much as possible, because there is not separate test phase as such. One of the ways to achieve this is by leveraging both Exploratory Testing and Automated Testing throughout the project. Exploratory testing will come very handy when looking for bugs, opportunities to improve, and missing features. So you should plan on “exploring” the product at the beginning of each new sprint, or any time that there is a change done to a product feature within the sprint cycle. Similarly, you will need to plan and build your scripts to perform automated functional and regression testing within the sprint, because there is not enough time for performing thorough manual testing. One of the things to remember is that there are no really lengthy requirement document or specifications –other that the stories encapsulated on the backlog files --, so the only way to make sure that each feature is fully developed, tested, and accepted by the product owner before counting it as "DONE!", is by using the sprint backlog as your own test plan (or writing a test case or script for every feature). Some teams are treating test case scenarios as entries that need to be added to the product/sprint backlog files for planning and tracking purposes. Another factor to consider is that development is much more heavily engaged in testing, so you should leverage this, and work very closely with them to plan and build more automated scripts that cover realistic scopes.

Summing up:

If you enjoy being involved in product decision making, helping to shape how a product looks and works, and working in a collaborative environment that encourages team work and peer to peer relationships with your development counterparts, you will enjoy working on an agile project. On the down side, agile software development can be a little bit intimating at the beginning. Agile is all about embracing and rapidly adapting to changes –which might be hard to accept at the beginning-- plus there are new processes, and new communication styles in place, so you might feel a little reluctant about it. However, once you get into the dynamics of agile software development, it can be a very fun and empowering experience!

You can visit us at Empirix's QAZone for additional information about other QA topics.