Agile Testing with Test Studio

Agile software development in a nutshell

More and more teams are struggling to find a better solution to the dilemma "answer to changing business requirements or meet the project deadline". Even more, with the ever growing demand to shorten product release cycles, teams are pushed to adapt and customize not only their development patterns but their team structure as a whole.

For this reason, it's been quite trendy lately to talk about agile development methods. Promoting team collaboration, flexibility and adaptability to the quickly shifting environment, agile software development, and respectively agile testing, has been gaining momentum and popularity. Unlike traditional methods, the agile development model breaks tasks into small chunks, which reflect the short-term business goal for the period. Each short period (called iteration or sprint) involves a team working through a full software development cycle including planning, requirements analysis, design, coding, unit testing, acceptance testing, and final stage when the product is delivered to the business owners. Thus, overall risk is minimized, as the project adapts to changes quickly. Agile practices emphasize working software as the main way to assess progress. They also reduce the complete development cycle time.

The small chunks of business requirements/features that need to be first coded and tested are called user stories. Once a user story is defined by the business owners, all members of the team (developers, QAs, etc) can proceed with estimating the needed effort to develop and test it.

Sprints or iterations are the short period of time (1-4 weeks) in which a team can fully plan, develop and test a user story. The latter can either be a shippable feature or part of a bigger feature.

The main principles of agile software development include:

- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

Some of the popular agile methodologies include: SCRUM, Extreme Programming, Feature Driven Development, Agile Unified Process, Lean Development, Agile Modeling, and more.

Is my team agile?

How do you know if your team is agile? How does one quantify agility? You can evaluate your team's development efforts against several key performance indicators:

- Are all stakeholders involved in the project and actively participate?
- Are your team members collaborating?
- Does your team produce quality releases on a regular basis?
- Does your team analyze each release to improve process discipline?

All team members on an agile project belong to any of the below "sub-teams":

- Customer Team – this team defends the interests of the business owners. They are usually business and subject matter experts who define the user stories and ultimately the features of the product.
- Developer Team – these are software architects, programmers, database administrators who are to deliver the code for the product.

On an agile project, teams should work very closely with each other. While Customer Team defines the user stories, the

Developer Team helps with prioritization and then implementation of the stories.

One might ask himself – so, in which group does the QA professional fall? The right answer is: BOTH. A Tester is supposed to elicit requirements from the business stakeholders and transform them into tests, while at the same time advocate for continuous quality among the development team members throughout the project duration.

What's this agile testing all about?

Unlike a waterfall-organized team for example, testers on an agile project play an important role during all the phases of the project lifecycle:

During the planning phase of a project, agile testers do:

- Help the rest of the team understand stories by asking questions
- Only take on stories that can be fully tested by the end of the sprint
- Collaborate with customers to better understand their stories
- Collaborate with developers to make sure all have the same understanding for the sprint stories
- Start working on high level user acceptance tests for each story
- Think about any test data or testing dependencies needed for story testing

When in coding and testing phase of the project, testers do:

- Write detailed tests once coding starts
- Start with simple tests and add more complex ones once the latter are passing
- Complete tests for one story at a time
- Identify testing obstacles as early as possible and communicate to team
- Facilitate communication between the customers and developers for issues that arise from coding and testing
- Write some manual tests for scenarios that are too difficult to automate with the automated agile testing tool in hand
- Start automating tests once enough code is written for regression suite

During the final stage, and post-release testers take part in:

- Identifying testing related obstacles during sprint
- Estimating if there was enough time to test all completed stories
- Finding ways to improve next sprint
- Demoing product to business owners for feedback

The agile development model involves testing as early as possible, and as often as working chunks of the software are released. This is usually done by using automated agile testing tools to minimize the amount of manual labor involved. The Agile testing process does not entirely exclude manual testing; however, counting on manual tests only may result in either buggy software or slipping schedules, as it may not be possible to test the entire build manually before each release. On the other hand, automated testing promotes regression tests that stimulate enhanced project velocity. While automated tests are critical for an agile development project, they are also essential for any other development methodology.

There are a few things a tester should consider in order to decide what to automate for each sprint:

- Start with stories that do not have dependencies than can break your automated tests
- Write the "happy path" test first and then expand to complex or edge scenarios
- Estimate the complexity of a test case scenario and the potential regression value of its automation

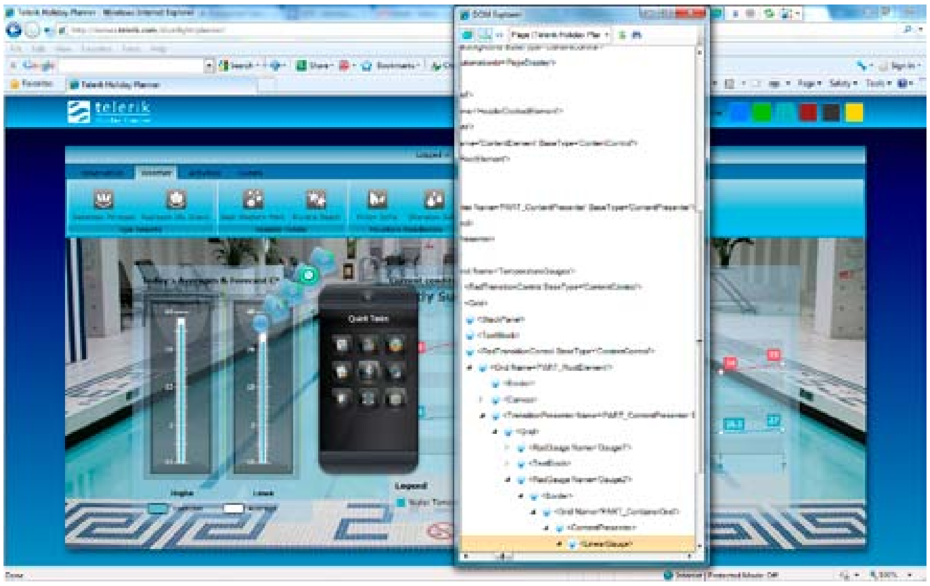
Riding the agile testing wave with Test Studio

Following the agile testing or not, testers nowadays have to adapt to rapid deployment cycles and disruptive changes in testing patterns. When on the lookout for an automated agile testing tool, testers need to consider the below features that will help him/her be more productive:

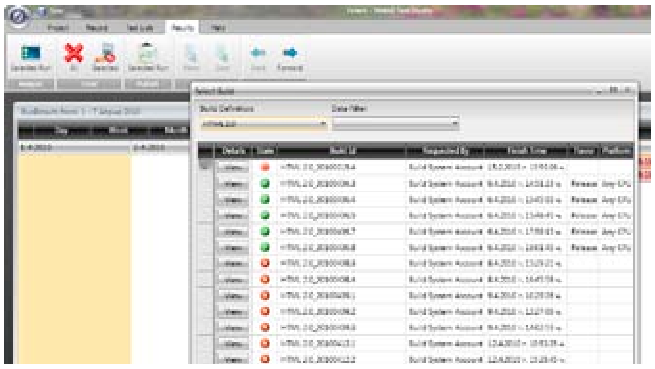
- The solution needs to serve multiple roles (QAs AND Developers) in order to facilitate communication among team members
- It should integrate within source control environment
- The agile testing tool needs to offer effortless test case maintenance
- It should support a test-first approach
- The solution has to allow writing test automation code using real languages, with real IDEs

While applicable to all types of development methodologies and team practices, Test Studio is an especially good fit for both developers and QAs on an agile development team. Telerik's set of automated testing tools feature editions tailored to the needs of programmers and testers in order to facilitate their collaboration and provide a common working platform. What's more, to further aid team cooperation, both editions of Test Studio introduce some common functionality:

- Common powerful recorder for testing (HTML, AJAX, Silverlight, ASP.NET MVC)
- The ability to create, open and edit the same tests in any edition – allowing developers on the team to take a certain test and further extend it and customize it in Visual Studio



- Source Repository Integration (TFS) to allow testers check-in test results next to the production code. In addition, this feature also facilitates the work of all testers on the team who work on the same project: now they can submit their results independently and at the same time.



- Common test execution model

Telerik Test Studio helps even further speed up team productivity and accelerates agile testing processes by introducing the below functionality:

- Rich Highlighting Surface
- DOM Explorer for HTML & Silverlight (Visual Tree)
- Element Menu for rich on-the-spot recordings
- Automatic Synchronization Steps
- Command line execution support
- Build Server Integration (CruiseControl, TeamBuildServer, TeamCity)
- Results Management using a results calendar
- Test List Management that supports ordering

Additional information and trial download

- *Learn more about Test Studio*
- *Download trial*