*Photo Credit: David Goehring (https://creativecommons.org/licenses/by/2.0/)
photo cropped, resized*

Agile or frAgile: Agile Development Needs Integrated Test

By Bob Bretall
Enterprise Agile Coach

Sometimes companies doing Agile development are really just doing iterative code and fix. Not maliciously; they don't seem to have a solid understanding of what doing Agile development really entails. The result is not realizing the full benefits of Agile development which often means low-quality frAgile code!

A common root cause of frAgile code is waterfalling test into the end of an Agile iteration. It's not meant to be something where a lot of programming is done for the first 8 or 9 days of a two week iteration and then thrown over the wall to be tested in the last day or two. Integrating test with day-to-day programming tasks is essential for delivering (relatively) bug-free software on a regular/accelerated basis.

Integrating Test into the Development Team

Eliminate the split between development & test, if it exists. Remove the thought that test is a separate 'phase' at the end of an iteration performed by people on a different team or in a different organizational hierarchy. Cross-functional integrated teams is a core concept in Agile. Agile development is all about teamwork, so let's get the entire team working together throughout the iteration. It would seem this goes without saying, but integrated test does not seem to be universally implemented by organizations "doing Agile".

THE DESARA GROUP

Integrating test with day-to-day programming tasks is essential for delivering (relatively) bug-free software on a regular/ accelerated basis.

Some Agile teams don't have a role called "tester", but there is always a need to write test cases for the user stories, make sure they pass, and make sure that some reasonable amount of regression tests (hopefully automated) are run on a regular basis.

My personal experience is that I have more commonly seen the "split-off testing team" in organizations new to Agile that are still holding onto their traditional organizational structure. Sometimes that structure is dictated by Human Resources and sometimes it is just inertia. Whatever the reason, a champion is needed to help guide the team to implementing integrated quality practices supported by the right individuals on the Agile team.

Ideally, working side-by-side, programming and test will happen throughout the iteration. As each piece of code is finished it is tested to make sure it works correctly before moving onto the next piece of code. A user story is never considered finished until it has been tested and shown to work; this should prevent lots of untested code being worked on in a compressed 'code and fix' cycle all bunched up at the end of the iteration.

Early Test Emphasis Improves Code Quality

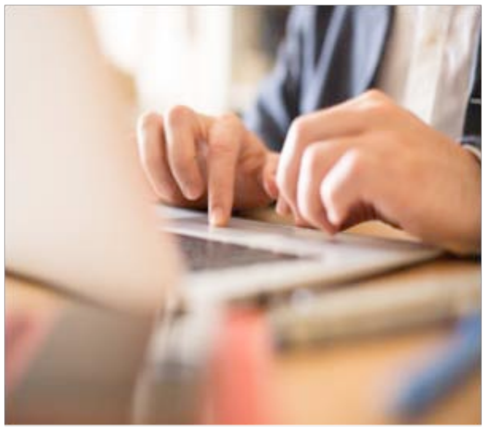
Have you experienced last minute test scrambles with 'code & fix' programming hacks, last minute builds, and skipped tests? If those sound familiar and they happen fairly often in your iterations, it's a good sign that something needs to change.

Testing at the last minute when there is insufficient time for a proper amount of testing just so you can be ready for a delivery is very stressful for everyone involved. Finding and fixing bugs earlier in the iteration reduces late night "fire drills" requiring overtime work to get working code completed for a deadline.

Test early and often; code quality improves, morale improves, delays and late discovery of problems that slip through the cracks are reduced.

A lot of people like to chant the 1st item on the Agile Manifesto when they want to dismiss anything that smacks of process:

Individuals and Interactions *over processes and tools*



If programmers are able to see the acceptance test cases as they write the code and ensure the code they write satisfies them, we're one step ahead of the game from the get-go.

Something that is missed in the "Individuals and Interactions" part is the implication that one of the most important ***interactions*** between ***individuals*** on the team is the regular interaction between people writing the code and people testing the code. We need to make sure the code is doing what it needs to do to satisfy the requirements, which is usually embodied in the user stories among other places.

This whole topic may seem obvious to people who are already doing this best practice, but hopefully it will turn on a light bulb for others who are still working with a siloed test organization.

Acceptance Test Driven Development (ATDD)

ATDD requires a lot of good communication between business customers, developers, and testers. Perfect for the Agile cross-functional team environment! The highlight here is on the people filling the test role on the team writing the acceptance tests early on, preferably before the developers begin coding.

Acceptance tests capture an external "user's eye view" of the system and are perfect for validating the code is doing what the user expects the system to do. If programmers are able to see the acceptance test cases as they write the code and ensure the code they write satisfies them, we're one step ahead of the game from the get-go.

ATTD is related to Test-Driven Development (TDD), but TDD is a much bigger step and not one I have personally seen implemented very often (which does not mean it's not happening a lot, just that there are also a lot of places where it's just not happening). Proponents of TDD swear by the results it delivers. In TDD, developers write initially failing automated test cases that define how the code being developed should work, then write the minimum amount of code to pass that test, and then refactor the code. Since it relies on developers writing the test cases, it frequently experiences serious push-back from the developers on the team. In my experience, and what I have heard from others, ATDD is a much easier strategy to implement than TDD and ATDD delivers increased quality at the end of an iteration roughly equivalent to TDD.

Lessons Learned

When converting an organization with a “test silo” that is separate from development into an integrated Agile cross-functional team, there are a couple of things to take into account:

- **Communication is Key:** We need to spend time making sure everyone on the team understands the problem we’re trying to solve and buys into the proposed solution (making a blended cross-functional team). When team members arrive at the conclusion themselves instead of having it mandated on them from on high we’re more likely to be successful.

  It is essential that everyone understands and buys into the value they are bringing to the team. Everyone needs to understand the core Agile concepts and practices used by the team, experienced coaches giving hands-on mentoring during day-to-day work is preferred. A couple of classes or a webinar is no substitute for hands-on guidance over the course of several iterations.

- **“Developers Don’t Do Test”:** This was a definite attitude that I have seen drive a stake through the heart of TDD. That said, having developers use acceptance tests as guidance went over much better, which made ATDD an easier upgrade to sell that still delivered good results in increased end-of-iteration quality.

Summary

The role of test is the one thing I have seen that is often under-served on Agile teams. It is an area that can be addressed to achieve really solid gains in quality and reduce time lost due to churning through builds and last minute fixes at the end of an iteration.

What I discuss here only scratches the surface of an extremely involved topic but hopefully provides food for thought and will cause some readers to look into upgrading their testing capability. For much more on this topic, including some really excellent stories from real practitioners sprinkled throughout, check out the book: Agile Testing – A Practical Guide For Testers And Agile Teams by Lisa Crispin and Janet Gregory. I found this book an invaluable resource.

THE DESARA GROUP

Bob Bretall

Enterprise Agile Coach

Bob Bretall has spent over 30 years in the software development field with hands-on experience in all aspects of the software development lifecycle. He has been a developer, designer, team lead, manager, trainer, mentor and consultant.

Read More **Bob.Bretall@DesaraGroup.com**