Battle of the Century: Technical Debt vs. Fea...

By Sean Still on *March 27, 2018*

Our teams are asked quite a bit about performance testing in an Agile environment and how that might differ from the traditional waterfall approach. If you're asking the same question, you've hit on one of the greatest heavyweight grudge matches in world history! Technical debt and feature development have battled for the spotlight since the dawn of time, or at least the dawn of Agile.

So, who do we want the winner of this epic battle to be? The answer is neither. If either of these heavyweights comes out on top, the likelihood of a successful project is slim. To put it simply, features are what customers are willing to pay for and technical debt is what they expect.

For example, imagine you are at your favorite restaurant and you've ordered your favorite meal. In this example, the meal is no different than a feature, that's what you're paying for. Cleaning the tables, bussing the dirty dishes, etc. is no different than technical debt. It is not something you're willing to pay extra for, it is something you expect the restaurant to take care of. What if your favorite restaurant focused solely on the food and didn't bus the dirty dishes or clean the tables? On the flip side, what if they focused only on cleaning and paid no attention to the food. In either case, the likelihood of your favorite restaurant staying in business is low. Thus, any project having these out of balance is unlikely to succeed.

Now that we conceptually understand the difference between features development and technical debt and how important it is to keep them in balance, let's talk about how Agile deals with these. Agile treats technical debt like credit card debt, keep the balance as low as possible, constantly pay it down. A team could carry the debt for a little while but the longer they do, the greater it builds and the harder and more expensive it is to pay back. The best plan is to pay as you go. With this approach, unit, regression, performance and every other type of testing you can think of should be factored into each sprint. It's not uncommon to devote 20-30% of a team's sprint capacity to paying down technical debt.

One benefit of this approach is if there is a problem detected, it's detected very early, and it's easily identified. Automated feature tests that are built as a part of the story help to save time as the project progresses, eliminating costly manual testing. Waterfall, on the other hand, pushes any testing towards the end of a project making even a single problem very hard to identify and even more difficult to resolve. If you have a choice, it's much better to fail early.

When you're on an Agile project it is easy to succumb to pressure for feature stories and ignore technical debt. Our job is to do everything in our power to ensure this doesn't happen and that technical debt is properly handled. In the long term you will be better off, and your team will ultimately deliver better value.