A few years ago I was in California talking to the development people at Palm. They made some of the first of what were then called Personal Digital Assistants (PDAs), which we now call cell phones. They tracked everything they did automatically. One of the many things they measured was how long it took to fix a bug — that is, how much time it took a software developer to fix a problem he'd introduced into the system. The computer tracked this automatically, each and every time.

So let's say that one day, when the testers tried to integrate Matt's code into the rest of the system, they detected a bug. Matt, like most software developers, wouldn't want to go back and fix that code right away. Instead, he'd vow to get to it later. First, he'd write new code.

At most companies this kind of testing doesn't even happen on the same day. It could be weeks or months before all the code is tested, and only then are the problems discovered. But Palm performed daily, automated tests of all their code, so they knew right away when there was a problem.

They looked at the "Matts" across the entire company — hundreds of developers — and they decided to analyze how long it took to fix a bug if they did it right away versus if they tried to fix it a few weeks later. Now, remember, software can be a pretty complicated and involved thing, so what do you think was the difference?

It took twenty-four times longer. **If a bug was addressed on the day it was created, it would take an hour to fix; three weeks later, it would take twenty-four hours. It didn't even matter if the bug was big or small, complicated or simple — it always took twenty-four times longer three weeks later**. As you can imagine, every software developer in the company was soon required to test and fix their code on the same day.

The human mind has limits. We can only remember so many things; we can really only concentrate on one thing at a time. This tendency — for the process of fixing things to get harder as more time elapses — represents a similar limitation. When you're working on a project, there's a whole mind space that you create around it. You know all the different reasons why something is being done. You're holding a pretty complicated construct in your head. Re-creating that construct a week later is hard. You have to remember all the factors that you were considering when you made that choice. You have to re-create the thought process that led you to that decision. You have to become your past self again, put yourself back inside a mind that no longer exists. Doing that takes time. A long time. Twenty-four times as long as it would take if you had fixed the problem when you first discovered it.

Jeff Sutherland (September 30, 2014). Scrum: The Art of Doing Twice the Work in Half the Time. ISBN-13: 978-0385346450

Buy it on Amazon
Jeff Sutherland on Wikipedia

Referenced from http://www.hadermann.be/blog/219/the-cost-of-waiting-for-feedback