Can a business be too agile?

Agile development has been embraced by businesses around the world. It improves speed to market by cutting down on the development cycle. If done properly, it improves the quality of output and mitigates risks.

Image: www.freedigitalphotos.net

But risk increases when agile methodologies are not applied correctly. And, unfortunately, it is quite easy to get it wrong.

According to Cornel Masson, Senior Software Engineer of Stone Three Venture Technology: "The best thing about agile development isn't necessarily its speed, it's the fact that it is iterative, which makes it highly reactive. You start with just the essentials and add new functionality every couple of weeks.

"Critically, at the end of each development phase (or 'sprint'), the product should be of working quality - ready to go straight into the hands of your customers for early market validation. If an element doesn't work, it's not the end of the world; it gets adjusted in the next sprint."

Unfortunately, this is where corners are often cut and the basic principles of agile development are forgotten, or discarded, in the quest for speed.

A huge factor

"Time to market is a huge factor in product and software development, and agile is definitely the most successful approach for getting a product out quickly. But when good design principles are overlooked in the interest of saving time, you almost always end up with a result that's unstable and poor in quality," said Masson, who has been in the software development business for over 25 years.

"I have seen cases where developers 'hacked' something together quickly, without making sure the foundations were solid or carrying out proper quality assurance processes. This is not Agile development, just hacking. Agile starts with a small 'feature set' that is iteratively added to, but quality should never be sacrificed. Hack jobs usually have to be rebuilt completely, at great expense.

The US health insurance initiative healthcare.gov (also known as ObamaCare) is an often-cited example of software development gone wrong. What should have been straightforward e-commerce site turned into an embarrassing failure of huge proportions.

There were a number of failings, but one of the biggest was that the second principle of the Agile Manifesto - 'working software' - was not adhered to. The front-end of healthcare.com was an Agile project (completed within a few months), but the back end never actually worked. What should have been an amazing legacy project from the world's most powerful man became a laughing stock.

Rock solid and scalable foundations

When it comes to developing programs and systems that need to handle vast amounts of information, its vital that the foundations are rock solid and scalable, otherwise they'll crumble as more complex functionality and extra users are added to the platform.

"The first few weeks and months of development are vital," said Masson, "These sprints typically take a bit longer, because we're doing the groundwork. We need to build a solid base and test it thoroughly. After each sprint we run our clients through a physical demonstration of what we have built to make sure everyone knows how it works and is satisfied with it. Ideally, real end-users should test the product."

Some green-field (entirely new) projects, however, require a lot of exploration to test the concept.

Masson explained: "A prototype, or proof of concept, is a 'skeleton' of the final project that helps ensure everyone is on the same page and uncovers any glaring gaps or errors, or just tests the market. It's a great guide for developers to work to.

"The danger, however, lies in stakeholders not understanding (and developers not communicating) that such a prototype is not the real thing. It might have to be discarded or substantially 'hardened' before it can be 'productionised'.

Not worth cutting corners

Francois Swanepoel, CTO of Stone Three expanded: "It's like comparing a real castle to a temporary film set designed to look like one. The fake one gives you an idea of what the real thing looks like; how much space there is and where the gaps are, etc. - and it gives you a much better idea than a map or small model would - but it's flimsy, you couldn't necessarily live in it. If your requirements change you still may need to build the real thing from scratch."

So, while it's easy to get caught up in the time-to-market rush, it's important to remember that it's not worth cutting corners and throwing out good development and design principles.

And anyway, time to market doesn't guarantee success - just ask Amazon, Facebook or Google. None of them were the first to market. Instead, they focused on introducing a quality, workable product that could scale, making small but regular improvements on an ongoing basis until each cemented itself as the leader in its particular industry.