Upfront Thinking Is Like Insurance

by Mike Cohn • 31 Comments

Insurance is great for all sorts of things. I have health insurance in case I become ill or injured. I have auto insurance that will repair or replace my vehicle if it's damaged. It also protects me in case I am involved in an accident that harms someone else. I have life insurance so that if I die, my wife and daughters are taken care of.

These are types of insurance I've chosen to have. There are other types of insurance I've also chosen not to have. For example, I've personally chosen not to have dental insurance. Most years that works out great for me. But there are some years when I have a big bill and wish I'd paid for the insurance. But, overall, I've been happy with that decision.

Similarly, every time I buy a plane ticket online there's a little checkbox asking if I want to buy travel insurance. I think it covers me in case I get sick right before a flight or something like that. I've never bought that and I've never regretted it. I've also flown over 2 million miles. So forgoing that type of insurance has also worked out well for me.

So insurance is great. I can't imagine going without my health insurance. But other types of insurance can be thought of as calculated gambles, which is exactly what they are. And sometimes those gambles pay off, sometimes they don't.

We can think of upfront thinking on development projects in the same way we think about insurance. Like insurance, upfront thinking is a way of paying a little today to avoid paying a lot tomorrow.

Upfront thinking on software projects can take a number of forms. It can be someone thinking about the architecture of the system. It could be a UI designer sketching wireframes. It could be an analyst building detailed scenarios and confirming all manner of edge cases conditions through workflows. Or it could be a database designer thinking about the structure of the database. I'm not saying any of these are bad (or that they're good). They are merely examples

of upfront thinking.

Some projects will benefit from some amount of upfront thinking (although perhaps not all forms at once that I've just listed), just like many of us benefit from having some types of insurance.

How Much Upfront Thinking Is Enough?

An important consideration is how much upfront thinking is enough. The best way to determine this is, unfortunately, in hindsight. But use the level of upfront thinking you do on current projects to help you decide how much to do on future projects. Here's how ...

Suppose your team finishes a project and they never had to reverse a decision. Every decision was made perfectly the first time. I'd say that team overinvested in upfront thinking. They over insured.

Consider instead a team that finishes their project and only had to reverse one decision. But it was a major decision and caused a total re-architecting of the system. That team underinvested in upfront thinking. They underinsured.

Finally, consider a team that finishes the project and had to reverse a lot of little decisions. None caused big re-architecting, but there were a lot of little decisions that each caused rework. On their own, each is small. But added together, it was a lot. Again, here's a team that underinvested in in upfront thinking. They were under insured.

So, as projects progress, evaluate whether the team is having to occasionally backtrack on decision and architectural decisions. They should occasionally have to do so. If they never do so, they've over invested in insurance by doing too much upfront thinking. But, if they're backing up too much or in big ways, they have underinvested and should do more upfront thinking.

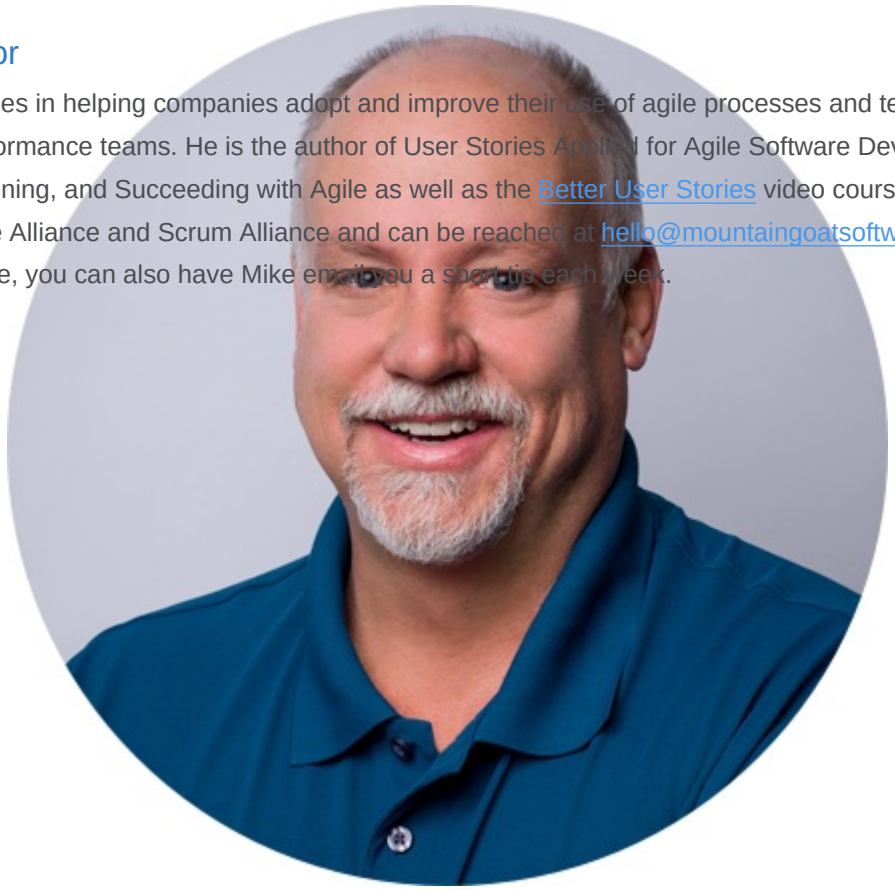
What Do You Think?

Use the Comments Section below to let me (and everyone else!) know how your projects are doing. Are you over- or under-investing in upfront thinking? Are there strategies you've found helpful for doing just enough upfront thinking?

Posted: September 22, 2015

Tagged: user experience, software, architecture, ui design, projects, development, upfront thinking, wireframe

About the Author



Mike Cohn specializes in helping companies adopt and improve their use of agile processes and techniques to build extremely high-performance teams. He is the author of User Stories Applied for Agile Software Development, Agile Estimating and Planning, and Succeeding with Agile as well as the Better User Stories video course. Mike is a founding member of the Agile Alliance and Scrum Alliance and can be reached at hello@mountaingoatsoftware.com. If you want to succeed with agile, you can also have Mike email you a short tip each week.