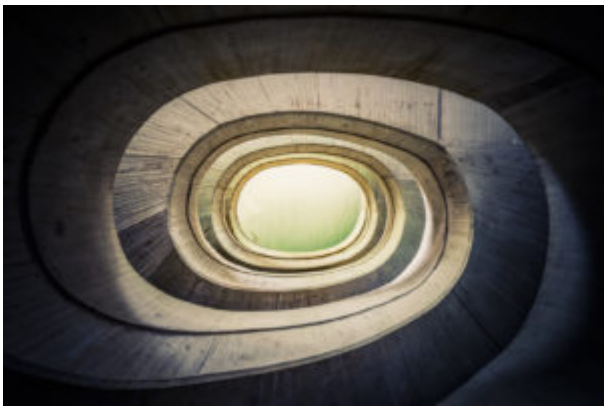
Role of Software Architect in Agile Projects

written by Manoj Khanna | January 2, 2017



One of the biggest misconceptions about Agile is that architecture is not required in the Agile development. ‘We‘re Agile; we don’t need architecture’—is something that everybody involved in Agile has heard at least once.

Let’s start by establishing a common understanding of what the software architecture is, to which everyone can agree. The definition of architecture is quite broad, and the roles and responsibilities of software architects vary dramatically from company to company.

Here is how Martin Fowler identifies architecture in Patterns of Enterprise Application Architecture:

“Architecture” is a term that lots of people try to define, with little agreement. There are two common elements: one is the highest-level breakdown of a system into its parts; the other—decisions that are hard to change.”

I find that we all can agree on those two common elements. Do we need the highest-level system break down? Absolutely. Do we need huge documents and long design stages? No. Agile is not against the architecture—it’s against useless, bulky documentation that nobody reads anyway.

From the perspective of change, the role of architecture in Agile development becomes quite clear – A good architecture is responsive and will support agility; a poor architecture will resist it and reduce it. And, since one of the benefits of adopting Agile is a better response to changes in the requirements, it's obvious that flexible and extendable architecture is a key to this.

The biggest issue that I've noticed is the very thin line between architectural design and software design. I've seen companies where the different implementations of the following practice were used: Architects created design documentation and developers were responsible for writing the code. This introduced a myriad of problems, starting with developers feeling that they were not fully trusted. This also gave developers an excuse not to really think about the design. 'We're just coders, not responsible for the design and we do only what we are told to do.' is a common attitude that I have witnessed. In Agile, the developer is responsible for the code he writes (and unit tests) as well as the design since nobody else will provide him with it.

Ideally, the high-level software architecture is completed before coding starts. And I really have to be careful here – completed doesn't mean written in stone; it can change, but with an understanding "this is the best of what we know right now." This doesn't necessarily include a database design or class diagram, and the level of details really depends on the approach you will be taking moving forward. I found that for certain systems the Domain Drive Design (DDD) is extremely useful and has made my life far easier. Therefore, I like to have a domain model and a basic set of domain classes and their relations defined, but not to the level of methods and attributes.

Personally, I prefer projects to have a design stage; this is when the high-level business domain model and user stories are created. At this stage the main architectural decisions are

made – the technology that will be used, the database server, the application type (for example, Mobile, rich client, or service), the architecture style (client server, layered architecture, SOA) is selected, and the architectural frame that will be applied is selected as well. The document created during this phase is not solely architectural effort, it is a collaborative effort of business analysts, developers, and network administrators. The output of this design stage is not only a high-level architectural document. (This is not an attempt to make fixed predictions of the future or create a detailed software design upfront as this approach places all the significant decisions at the point of least knowledge in a project's lifecycle). This is simply a way of getting and sharing the common understanding of the system we are all about to develop.

This is an excerpt from the forthcoming book, The Art of Being Agile.

[Image Courtesy: Flickr/Helix/Philip Gunkel]