October 16, 2019

Software Architecture in Agile Development

CS-E3150 Software Engineering

Software architecture refers to the fundamental structure of the software system and looking at it at the highest level of abstraction. It is concerned with how a software system should be organized and involves designing the overall structure of the system. (Sommerville, 2016)

Role and Importance

Software architecture is needed in dealing with complexity. Size related factors, strict quality requirements, limitations on physical resources may further contribute to the complexity of developing software. When developing a large and complex software system things need to be planned and the role of software architecture is this thinking before doing something and what you are about to do. (Männistö, no date)

Agile developers find software architecture important and helpful in terms of communication between team members, evaluating design alternatives, inputs to subsequent design, etc. User stories used in agile development can often ignore quality requirements because they focus on functional requirements. Unfulfilled quality requirements can render the whole software useless and risky. The main objective of software architecture in agile development is to remind the developers of both functional and quality requirements. Software architecture is also important as it serves as a vehicle for communicating with the stakeholders. Architecture reviews and analysis are an effective way to find potential problems and risks in a system design in early stages of development cycle of the software. As the complexity of a software increases, so does the relevance and importance of software architecture. (Abrahamsson et al., 2010)

Emergent v/s Planned property

When used in the agile model, software architecture can't be termed as an emergent or a planned property because it is a mixture of both. Software architecture should be such which can easily accommodate both anticipated and non-anticipated changes.(Kruchten, no date) If we consider the software architecture as a planned property, we can then define the system very efficiently in terms of the functional and non-functional requirements. However, in doing so we must be very specific about the architecture which can make the design process time-consuming and frustrating. This also constrains the development teams and it may lead in taking wrong architectural decisions. The proposed architecture may work effectively at the beginning of the development cycle, but it may not when the product implementation is scaled up. The range of impact of a wrong decision in such situations becomes very large and often leads to creating the whole software from the beginning which increases the overall cost of the project. However, if we think of software architecture as an emergent property, we could incorporate architecture modifications very easily which makes the organization resilient. But it would also lead us to make the architectural significant decisions incrementally which can put the whole project in chaos.

It is thus clear that architecture design is both a planned and emergent process. Generally, at the beginning of an agile software development process, some fundamental decisions are made which help the developers start with the project. After some iterations, the architecturally significant things are looked upon and further decisions and plans are organized over time. In this way we can build the architectural design over time. (Abrahamsson et al., 2010)

When, How and by Whom?

Architectural designing and planning should be done right before starting the development of the software system. In this phase, it is very important to have a good understanding of the context of the system. This helps you to make architecturally significant decisions. Before taking any decision it is important to think about the range

of its impact on the rest of the components if it turns out to be ineffective and needs to be changed in the future. Taking decisions hastily based on short-sighted requirements may seem effective, but can prove to be a nightmare when the system is scaled up later. After taking the decisions, we can start with the development of the software incrementally. In this way the make the important decisions first which help us layout the overall structure of the system and push back those decisions which can be easily reversed. This also enables the developers to have an initial base from which they can start and improve the architecture with every passing iteration. (Kruchten, no date)

Architecture doesn't emerge in a day. You need people who understand it and have experience. Architectural designing and planning can be done by a single architect or a team of architects. A software architect is expected to facilitate the software development process as well as represent the system's quality attributes. The architect makes high-level decisions and communicates it to the developers who then devise a way to implement it. A good architect should have to understand the importance of software architecture and architecturally important decisions.(Männistö, no date) The architect should be good with people and understand both the problem and the solution domain. He/she should have good technical knowledge and should be familiar with the application domain of the software. He/she should decide the right time to freeze the architecture to provide the developers the stability to finish a project. (Abrahamsson et al., 2010)

References

ABRAHAMSSON, P., BABAR, M., KRUCHTEN, P., 2010. *Agility and Architecture: Can They Coexist?. IEEE Software* [online]. [viewed 17 October 2019]. Available from: https://mycourses.aalto.fi/pluginfile.php/1088042/mod_resource/content/7/AgilityArchitecture.pdf

KRUCHTEN, P., LASSENIUS, C. no date a. *An interview with Professor Philippe Kruchten about Software Architecture in Iterative and Incremental and Agile*

Development [podcast]. [viewed 16 October 2019]. Available from: https://mycourses.aalto.fi/pluginfile.php/1088044/mod_resource/content/15/PKruchten.mp3

MÄNNISTÖ, T., LASSENIUS, C. no date a. *An interview with Professor Tomi Männistö about the basics of software architecture* [podcast]. [viewed 16 October 2019]. Available from:https://mycourses.aalto.fi/pluginfile.php/1088043/mod_resource/content/12/Mannisto.mp3

SOMMERVILLE, I., 2016. *Software Engineering*. 10th ed. Boston: Pearson.