Software Architecture in Agile Development

CS-E3150 Software Engineering

According to Sommerville (2016), a software architecture is a "description of how a software system is designed." All the properties of a software like security, performance, availability, etc depend upon the architecture.

Role and Importance

According to Kruchten, (2015), software architecture is something invisible to external users. But it is important in dealing with complexity that comes with the software size, limited resources, and quality requirements. A plan is needed on how the developer team is going to use which resource because it gets difficult to change it after writing a lot of code.

Software architecture helps in easy communication among various stakeholders as it acts as a high-level presentation of how the system is going to work (Sommerville 2016). Architecture requires some analysis of the system which helps in finding possible flaws in the design of the system at an early stage. Thus with the increasing size and complexity of the system, planning the architecture becomes more important.

Emergent versus Planned property

In the traditional waterfall model, architecture design is usually done right after the requirements have been specified (Sommerville 2016). But in agile development, software architecture is something that is not explicitly planned, instead, it emerges as the development proceeds (Kruchten, 2015). But it still needs some planning because it's difficult to make changes to the architecture later on. For example, the architecture might work in the beginning for a small number of users, but it might not work when the system scales up (Kruchten, 2015).

If we look at software architecture as a planned property like in the waterfall model, then we need to be very precise in terms of what we expect the system to do, because incorporating changes in a waterfall model at a later stage is very expensive. But it is difficult because the requirements are usually vague. Considering the architecture as emergent property allows us to make architectural decisions gradually. But fundamental decisions must be taken in the beginning, for example, the programming language to code in.

So in agile development, software architecture is more like a mixture of a planned and emergent property. Some basic architecture is planned at the beginning of the project just to get started, and the rest emerges later on as the project proceeds and things start getting clearer. (Kruchten, 2015; Abrahamsson et al., 2010)

When, how and by whom?

According to Kruchten (2015), architecture designing should be done in the first one-third or one-half of the project. Abrahamsson (2010) also says that early planning of architecture is useful because these decisions are hardest to undo, change and refactor. Before taking a

decision, it’s important to think about its range of impact in terms of the team, components/qualities and other properties of the system. Decisions that seem good for a near-future might not be effective in the long term. Kruchten advises being adaptive about architectural decisions. That is, we don’t have to make a decision and instantly freeze it. Instead we should make a decision so that it can help us get started, and then later change it if need be.
Software architecture designing can be done by a single software architect or by a team. As Kruchten (2015) says, it is a role, not a job. But having a software architect is helpful so that the development team can be reminded at regular intervals to take certain architectural decisions.
A software architect is expected to have a good understanding of what an architectural significant decision might be. S/he should be familiar with technologies available and is preferred to have good contact with the product owner so that the requirements can be understood easily (Kruchten, 2015). The role of a software architecture includes making architecture significant decisions, mentoring, troubleshooting, focusing on external and internal coordination, etc. (Abrahamsson et al., 2010)

References

ABRAHAMSSON, P., BABAR, M., KRUCHTEN, P., 2010. Agility and Architecture: Can They Coexist?. IEEE Software [online]. [viewed 01 March 2020]. Available from: https://mycourses.aalto.fi/pluginfile.php/1177211/mod_resource/content/6/AgilityArchitecture.pdf

KRUCHTEN, P., LASSENIUS, C. 2015. An interview with Professor Philippe Kruchten about Software Architecture in Iterative and Incremental and Agile Development [podcast]. [viewed 01 March 2020]. Available from: https://mycourses.aalto.fi/pluginfile.php/1177213/mod_resource/content/15/PKruchten.mp3

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