October 10, 2019

Requirements Engineering

CS-C3150 Software Engineering

Requirements Engineering is a process of understanding why we are developing a system or software. It's about knowing the current situation and having an understanding of what a customer or user needs.(Kauppinen, 2013) Requirement engineering involves three main activities which are **requirement elicitation**, **requirement specification**, and **requirement validation**. The requirements for a large software system are always changing. One reason for this is the undefined nature of some requirements, also called as "wicked" problems. Due to this, the software requirements are bound to be incomplete which makes documentation difficult.(Sommerville, 2016) Therefore requirement engineering is one of the most important as well as one of the most difficult tasks in software engineering.

Traditional Requirement Engineering

In the traditional model, requirement engineering is the first step in software development. All the functional and non-functionals requirements are documented upfront at the beginning itself. This is not a single step, but an iterative process. Early during the iterations, most of the effort goes into understanding the high-level business, user and non-functional requirements. In later iterations, most of the time is spent on understanding the non-functional and the system requirements. All of this is essentially done before the actual development of the systems starts which helps to give an estimate of whether the system could be implemented under the budget and other constraints. This kind of RE model is suitable in situations where the requirements don't change too frequently. The structured workflow of this model makes RE very stiff and doesn't provide the user with any room for changing the requirements. This model ignores the volatility of requirements toward change and its effect on the later phases of development. (Sommerville, 2016)

Agile Requirement Engineering

In an agile model, it is assumed that RE continues throughout the life-cycle of the system. In agile, requirements engineering is implemented with continual collaboration with the customer. Based on feedback from various stakeholders, the requirements are constantly added, deleted or updated throughout the development process. Most organizations use the concept of user-stories to specify the requirements. Unlike traditional RE, where prioritization of requirements happens once, in agile RE the requirements are prioritized in every development cycle. In contrast to traditional RE where the prioritization depends on multiple factors, in agile RE it depends predominantly upon the business value defined by the customer. The developers then discuss these requirements in detail with the customers before or during the development. This kind of RE model is useful where it is impossible to get the complete requirement specification at once. (Cao & Ramesh, 2008)

Agile RE solves many major problems in the traditional RE like communication gaps, validation and customer involvement. The customers steer the project according to their understanding of the product solution. Informal communications discard the need for formal documentation and approval procedure. Regular and frequent communication with customers helps developers get a clearer picture of the requirements and it also creates a satisfactory relationship between them. All of this makes the process much more flexible from a customer's point of view.(Cao & Ramesh, 2008)

This makes it very clear that agile RE overcomes many challenges faced by the traditional RE. However, in safety-critical scenarios, the traditional RE turns out to be more effective.

References

Cao, L. & Ramesh, B. 2008. *Agile Requirements Engineering Practices: An Empirical Study*[online]. [viewed 12 October 2019]. Available from: https://mycourses.aalto.fi/pluginfile.php/1088028/mod_resource/content/12/AgileRE.pdf

Kauppinen, Marjo. 2013. *Requirements Engineering* [podcast]. 6 February 2013. [viewed 12 October 2019]. Available from: https://mycourses.aalto.fi/pluginfile.php/1088029/mod_resource/content/12/Kauppinen.mp3

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