CS-250: Assignment 7-1: Final Project
Prepared on: August 19, 2024
Prepared for: Prof. Robin Chataut
Prepared by: Alexander Ahmann

Contents

1 Background

In earlier journal entries where I discuss the *software development life cycle*, I explain software is a socially constructed entity.[1] By that, I mean that software is not the product of Nature, but rather the indirect collaboration of many scientists and engineers. While it is possible for an individual software engineer to write complicated software and even entire operating platforms "from the ground up," (Cecil, 2018) it is also important to know that software can only exist in the context of previously existing hardware created by other scientists and engineers. The same can be said about most other software: they are typically developed with *integrated development environments*, and

[1]See Ahmann (2024a) and Ahmann (2024b).

depend on libraries, *application programming interfaces* and other software—most of which were developed by third parties.

This kind of collective labour of software engineers needs to be organised, and the software development life cycle has different models for organising such efforts. In this paper, I will discuss implementing the *Agile* and *Scrum* models for software development applied to the making of a travel web application.

1.1 Problem Statement

ChadaTech is a hypothetical software development firm that is tasked with developing the *SNHU Travel* web and mobile applications, and has been using the *Waterfall* model for developing software (CS-250, n.d.-a). ChadaTech wants to transition to an *Agile* approach where the *Scrum* subset of management of developer labour is used to replace the current *Waterfall* model.

The task outlined in CS-250 (n.d.-a) puts me in the role of a scrum master for the hypothetical software company *ChadaTech.* Specifically, my job is to plan out *Sprint Reviews and Retrospectives.* I am tasked with applying roles for other developers and testers of the team, interpreting user stories, handling conflict and interruptions, faciliating communication, selecting which Agile/Scrum tools to use to organise the project, and of course evaluating the efficacy of the Agile process that I have constructed, and put into practice.

2 ChadaTech's Transition to Agile

2.1 Applying Roles

An *Agile* team with *Scrum* as the guideline for communication and management consists consists of three basic kinds of actors: the project owner, the *Scrum master*, and the developers. Software testers are thought of being as a subset of developers. A *product backlog*[2] is created to organise what features are to be added or changed, where anyone who contributes to the backlog is considered to be a developer.

The software being developed is a website and mobile application for a travel deals and discount service. I would need at least one of each of the subsets of developers: website developer, website tester, mobile develper developer, mobile application tester, application security engineer. The last occupation that I listed, *application security engineer*, is a special kind of

[2]See Karlsson (n.d.).

"tester" that is knowledgable on techniques to use hacking techniques to manipulate software applications into behaving in ways not intended by the website or mobile application developers. Such a role is important because of the impact of cyberattacks on organisations (Huang et al., 2023), and the quick identification of software vulnerabilities will reduce such risks.

2.2 Completing User Stories

The *Agile* and *Scrum* models of software development involves interviewing end-users of the software product, and then turning their user stories into concrete software engineering problems for the development and testing teams. The product owner interviews some (or all) end-users, and then collaborates with the Scrum master to make them into well-defined software engineering problems to be encoded into a backlog. The developers and testers will then do their best to implement what is on the backlog into solutions.

A case study for this involves some work that I did taking user stories from a recorded interview between the project owner and selected end-users (Ahmann, 2024c). Through an informal method of making needs and desires into concrete engineering problems, I read the transcripts of the interview, made them into solvable problems, and then transcribed them into a product backlog in the form of a *Microsoft Excel* spreadsheet.

2.3 Communication

I think that Agile and Scrum models of software development can greatly aide communication between individual developers, testers, project owners and even the end-users. This approach to software development encourages horizontal communication between the project owner, Scrum master and developers.

As an example, in the CS-250 (n.d.-b) I was able to demonstrate some basic ability of communicating with teammates in a hypothetical *Scrum* team. I assigned myself the role of a "software tester," and discussed how cybersecurity is an important feature in good software development. I also replied to teammates regarding how I feel they fit into the Scrum team, and what I feel the strengths and weaknesses are of the transition.

2.4 Handling Interruptions

The Agile and Scrum models of software development offer powerful tools for handling interruptions from the external environment. Such interruptions come in the form of an unexpected event, like a sudden need to change a core feature of the project. While I do not have direct experience with such interruptions, I can say that Agile/Scrum offers powerful tools to be robust in the face of drastic shocks from the external environment because of my understanding of Agile's design.

"Humanizing Work" (2023) discusses some ways that the Agile/Scrum model can help to reduce any costs that are associated with interruptions. Specifically I want to discuss that, through *Scrum Retrospectives*, *Daily Scrum* and other social events, the project owner, Scrum master, and developers and testers can all work together to discuss an interruption, and modify the product backlog to make room for addressing it.

2.5 Organisational Tools

Regarding the kinds of tools that ChadaTech should employ: I recommend using the ones developed Microsoft (n.d.) for implementing a backlog. A cloud office suite like *Microsoft Office 365* can also be employed. I think that we should also use *Skype* when doing remote based Scrum meetings.

3 Discussion

3.1 Evaluation: Waterfall and Agile

Both the *Waterfall* and *Agile* models of software development have their strengths and weaknesses. The Waterfall model is a linear approach of software development where much planning goes to executing the steps in the development phase, whereas the Agile model is a more flexible approach, hence its name, which can adapt to constantly changing demands and needs. The Waterfall model has the main limitation of being too rigid in execution, not allowing for going back to previous steps in the development phase, whereas the Agile model can be seen as too flexible, which may make it difficult to measure progress.

In correspondance with another software developer[3] I learnt that the "good stuff" of both the Waterfall and Agile models can be integrated into a "homemade" model of software development.

[3]This was another student in the Discussion forums who had experience in video game developmment studios implementing both Waterfall and Agile models of software development.

4 Conclusion

ChadaTech should embrace the new Agile and Scrum models of software development. It may be hard for workers who are accustomed to other methodologies to adapt to this new change, but I think that if Agile shows itself to be benefitial to both the developers, managers and end-users, then it would have served its purpose. We can even borrow from other software development models, like Waterfall, if the situation demands it.

References

Ahmann, A. (2024a). *CS-250: Assignment 6-2: Module 6 Journal.*

Ahmann, A. (2024b). *CS-250: Assignment 5-3: Module 5 Journal.*

Ahmann, A. (2024c). *CS-250: Assignment 3-2 User Stories Spreadsheet.*

CS-250 (n.d.-a). *Final Project Guidelines and Rubric.* Southern New Hampshire University.

CS-250 (n.d.-b) *6-1 Student Discussion: Vision Quest Software Case Study.* Southern New Hampshire University.

Cecil, N. (Sept. 7, 2018). *Man killed by train had tech following.* Columbia Gorge News. Retrieved on Aug. 14, 2024 from: https://www.columbiagorgenews.com/thedalleschronicle/news/man-killed-by-train-had-tech-following/article_1f03fc0d-c223-5f20-a915-460fce4299f1.html

Huang, K., Wang, X. Wei, W. & Madnick, S. (2023). *The Devastating Business Impacts of a Cyber Breach.* Harvard Business Review. Retrieved on Aug. 16, 2024 from: https://hbr.org/2023/05/the-devastating-business-impacts-of-a-cyber-breach

Karlsson, J. (n.d.). *A Guide to Agile Product Backlogs.* Perforce. Retrieved on Aug. 16, 2024 from: https://www.perforce.com/resources/hns/agile-product-backlog-basics

“Humanizing Work” (Mar. 6, 2023). *Dealing with Interruptions on a Scrum Team.* Retrieved on Aug. 18, 2024 from: https://www.humanizingwork.com/dealing-with-interruptions-on-a-scrum-team/

Microsoft (n.d.). *Azure DevOps Services.* https://azure.microsoft.com/en-us/products/devops