Project Review Report

Team Management & Structure

During the first phase of the project we split into two groups of three. Each group were given a different set of documents to complete. We felt at this stage it would be best to minimize formally defined roles and instead distribute workload in a way where it suites the individual's strengths. Through this process we could see the group members that were the most proficient in writing up documentation.

In the early phases of project two, we felt it would be good to highlight our successes and failures of the previous assessment. Using this we could fuel our understanding about how our team organisation worked. Particularly in the first assessment, which may have otherwise failed in the second. This self-evaluation tactic allowed us to further our research into software team management structures, giving us a better understanding about them and why they may be potentially suitable to use in our case.

At the start of phase two, it was decided it was be no longer feasible to follow the team organization strategy used previously as workload had increased significantly. Due to our rigorous research we are able to compare team structures available to us and understand why one of them would be ideal for our team and the project. We chose to adapt to the Scrum team structure [1] in the second phase of the project because it was suitable for a smaller team like ours. The team management did not change in the third phase of the project as the Scrum team management was already proven to be working productively, It was unnecessary at this point to make any changes.

The most prominent change in the team structure was the allocation of work over the course of the project. As the team gradually got better at understanding each other, work was distributed effectively to suit each members strength and weaknesses. For example, throughout the first phases of the project Scott and Matthew demonstrated that they were skilled programmers and so assumed the informal role of lead software programmers who were head of development.

The constant improvement and self-analysis in our team management followed the capability maturity model (CMM) [2]. We used the five levels of maturity model to examine the progress of our team management throughout the project. The initial level was the starting point for our project - where we were put into an ad-hoc team. The repeatable level was the first phase of our project where the team was initially organised. The defined level was the second phase where the team management was changed to follow the Scrum process. The managed level was the third phase where team management was kept the same. By the end of these first four levels, we were able to learn skills and abilities form one another enabling our team to grow and become self-organized. The optimization level was the last phase of our project where the team was naturally working efficiently.

Development Methods & Tools

After analysing and comparing various software development methods available to us we decided to use Scrum framework [3] - an agile method. It was initially chosen because it was suitable for a smaller team like ours, flexible for assignment of work and to encourage regular meetings. This was ideal for our team as we lacked software engineering experience, it aided us by allowing us to learn as we go while working effectively. It was ideal for the project as it allowed us to make progress at all times, without falling behind on schedules, keeping us motivated. It also granted us more adjustability and allowed us to swiftly adapt to any requirements that changed. This framework has remained the same through the completion of the project because it has proven to be effective and reliable. It was effective because our team was able to efficiently handle issues and risks that became evident in a given sprint,

which were quickly resolved before the next one preceded. It was reliable because it suited to the evolution of our team management and structure.

Tool	Use	Evolution
Google Drive[4]/ Office Package[5]	Collaborative work and file sharing. Useful in meetings and creating docs/charts/tables.	No, remained throughout the project and no issues were experienced.
Google Calendar [6]	Arranging team meetings. Useful for keeping tracking of meetings.	Yes, no longer used after the second phase as Slack interfaced creating events. No events were added after.
Slack [7]	Formal communication method between team members. Useful for coordinating with other tools.	Yes, no longer used after the second phase because members used Facebook as primary communication method.
Trello [8]	Members can see tasks available/assigned/completed Useful for product backlog.	No, remained throughout the project and no issues were experienced.
Github [9]	Version control. Useful as only members to access code.	No, remained throughout the project and no issues were experienced.
Gmail [10]	Coordination of meetings with customer. Useful when face-to-face meeting inessential.	No, remained throughout the project and no issues were experienced.
IntelliJ IDEA [11]	Implementation of code. Useful as all members have used it previously.	No, remained throughout the project and no issues were experienced.
Lucidchart [12]	Create UML class diagrams. Useful as it easy to use.	No, remained throughout the project and no issues were experienced.
GitKraken [13]	Means of user interface for Git. Useful as it eases use of Git.	No, remained throughout the project and no issues were experienced.

Project Review Report

References

[1] Scrum.org, “Scrum,” [Online]. Available: https://www.scrum.org/. [Accessed 24th April 2018].

[2] Software Engineering Institute, Carnegie Mellon University. “Capability Maturity Model for Software, Version 1.1”. Available:http://www.sei.cmu.edu/reports/93tr024.pdf. [Accessed 24th April 2018].

[3] I.S. Sommerville. Software Engineering, Ninth Edition. Available: https://edisciplinas.usp.br/pluginfile.php/2150022/mod_resource/content/1/1429431793.203Software%20Engineering%20by%20Somerville.pdf. [Accessed 24th April 2018]. 2017].

[4] Google Drive. “A safe place for all your files”. Google Drive. [Online]. Available: https://www.google.com/drive/. [Accessed 24th April 2018].

[5] G Suite. “Get Gmail, Docs, Drive and Calendar for business” G suite. [Online]. Available: https://gsuite.google.co.uk/intl/en_uk/. [Accessed 24th April 2018].

[6] Google Calendar. “With Google's free online calendar, it's easy to keep track of life's important events all in one place”. Google Calendar. [Online]. Available: https://www.google.com/calendar. [Accessed 24th April 2018].

[7] Slack. "Slack: Where work happens". Slack. [Online]. Available: https://slack.com/. [Accessed 24th April 2018].

[8] Trello. “Trello”. Trello. [Online]. Available: https://trello.com/ . [Accessed 24th April 2018].

[9] Github. “The world’s leading software development platform . Github”. Github. [Online]. Available: https://github.com/. [Accessed 24th April 2018].

[10] Gmail. “Gmail is email that’s intuitive, efficient, and useful”. Gmail. [Online]. Available: https://mail.google.com/. [Accessed 24th April 2018].

[11] IntelliJ IDEA. “Capable and Ergonomic IDE for JVM”. IntelliJ IDEA. [Online]. Available: https://www.jetbrains.com/idea/. [Accessed 24th April 2018].

[12] Lucidchart. “Flowchart Maker & Online Diagram Software”. Lucidchart. [Online]. Available: https://www.lucidchart.com/. [Accessed 24th April 2018].

[13] GitKraken. “The Legendary GIT GUI client for Windows, Mac and Linux”. GitKraken. [Online]. Available: https://www.gitkraken.com/. [Accessed 24th April 2018]