PROJECT REVIEW REPORT

NP Studios – Team 8

Team Members

Lucy Ivatt, Jordan Spooner, Alasdair Pilmore-Bedford, Matthew Gilmore,
Bruno Davies, Cassandra Lillystone

Approach to Team Management

Our final approach to team management and structure consists of having set distinct roles within the group. The roles included a project manager (Cassie), a technical lead (Lucy), a designer (Alasdair), a tester (Bruno) and several developers (Jordan and Matthew). These roles are common in industry especially when using agile [1].

These roles differ slightly to the ones decided on at the very start of the project. Initially Matthew and Jordan were secretary and librarian, respectively. However, as time progressed and we became more comfortable with the project, it became clear that we did not need dedicated people for these roles. As we gained understanding of each other as team members it became fitting that Matthew and Jordan took on developer roles to spread the workload more effectively, with secretary and librarian being their secondary roles. Cassie remained as the project manager as this role was extremely important and fit her personality well. Lucy remained the technical lead as it was clear she had a good understanding of the development side and project structures. Bruno, who was originally report editor, changed his role to be the tester, and we instead allocated write up sections to each team member. These sections would mostly be about work they had completed personally, meaning they would be the most equipped to include high level of detail and accuracy. Additionally, we believed not having a dedicated tester would create a significant risk e.g. bugs or unimplemented requirements may be missed. Bruno was the fitting individual for the role as he had previous experience writing unit tests and he felt comfortable taking on this responsibility. It was especially important to have someone separate to the development team to write the tests as they are more likely to pick up on mistakes. Alasdair remained as the designer as he also had previous experience with this.

For the final assessment, we have been meeting roughly 2 to 3 times a week to ensure we are all making suitable progress and to keep each other fully informed. This is a decision which has not evolved much throughout the process. We have always recognised the large amount of work to be done and we found 2 meetings a week plus plenty of individual time yielded successful results.

To make team management easier we have primarily been using Discord for communicating with each other as well as for calls for when face-to-face meetings are not possible. The tools we use for communication have definitely evolved over time. At the beginning of the project we were only using Facebook Messenger to communicate with each other. It was mainly used to organise face-to-face meetings, which we relied heavily upon, and alert each other of any important updates. Towards the end of the project we began to face more difficult issues with the code during individual work. We decided to move to Discord where we could organise our messages more efficiently (use code block formatting with syntax highlighting, pin important information, etc.) and also join voice calls to communicate more effectively when not together. Due to COVID-19 we are limited to meetings via these calls. We communicate more frequently, updating each other on everything we do in the suitable channel, as due to the climate there is a higher chance of risks materialising.

Another result of the current climate is possibly mitigating risks differently. For example, where we said that we would use resources provided by the university, if a risk materialises, we instead will have to share the workload to unaffected team members. We are also going to rely on virtual meetings or emails with the customer if we need to as we can no longer meet him in person.

The current climate means that team management is slightly harder, but we have managed by slightly changing our methods to adapt to the situation.

Methods and Tools

The software development method we chose to use was agile development with SCRUM, allowing requirements to be changed throughout the process without a detrimental effect to the product. This is a decision which we have not changed because it is the most suitable methodology for this type of software engineering project. It is highly iterative [2], which suits the structure of assessments being separated into three sections (which included coding) with new requirements to be implemented each time. We have also communicated with the client consistently throughout the whole process to get our ideas / changes approved, which is a key feature of agile development. SCRUM is also quite common with agile game development [2], and we have found it to be highly effective, allowing us to produce successful product iterations. Therefore, there was no need to change our methods.

For development we used a range of different tools. We used git and GitHub for version control and our IDE of choice was IntelliJ. One thing that changed slightly with our chosen tools is how we each used Git. At the beginning we all decided to use Git Bash, however towards the end some members of the team preferred to use other methods, for example Git Desktop or the git integration within IntelliJ. This was just a personal choice for some people based on what they felt comfortable with. We did not see a reason to use anything other than Git for version control as none of us had any major experience with anything else, and we had no issues with it. We continued to use IntelliJ as our main IDE because we found it worked the best with the game engine, as LibGDX is especially designed for it [3].

For all of our reports and documentation we used Google Drive to store them, as well as using the inbuilt text editor. This allowed us to easily collaborate on documents, viewing changes in real time and being able to access documents remotely very easily. At the beginning, we decided to use Trello to keep track of how documentation writing was allocated between members of the team, due to its simplicity and array of project management tools such as setting deadlines, assigning members etc. We then introduced the use of GitKraken Glo Boards specifically for our backlog and programming tasks. We believed this would work well as it had direct integration with GitHub as well as similar project management tools as mentioned previously.

However, as we had documentation tasks in Trello, documentation written in Google Drive and programming tasks in GitKraken Glo, we had everything split between 3 different services which we believe overcomplicated the project slightly. Therefore, in the final assessment we only used Google Drive. In assessments 3 and 4, the project became primarily about change management and after research [4] we decided that a change tracker was an extremely important document to maintain. We created this in the form of a table and found it would only be duplicating what we had on GitKraken Glo (description of change, change builder, progress etc.), therefore we stopped using this service.

We also realised in the later assessments that it was easier to begin writing documentation together once the code was complete. This made it easier to keep track of progress, which team member was allocated to a section, and ensured anything we were writing about (such as how something was implemented) would not change and therefore saving us time. These reasons made Trello less necessary and instead we took advantage of the comment system on Google Docs to communicate with each other, as well as in meetings (online or in person).

Overall, we found the change tracker much easier to manage than GitKraken, saving us time for more important work. As we already used Google Drive for documentation, it also allowed us to have everything in one central space. In summary, the reason for the change of documentation tools is due to the changing needs of the project and our understanding of team and project management; we found it was easier to manage the team and the tasks using fewer tools.

References

[1] W. Cole, “Roles and responsibilities in a software development team,” fluent, 09 08 2018. [Online]. Available: https://this.isfluent.com/blog/2018/roles-and-responsibilities-in-a-software-development-team. [Accessed 20 04 2020].

[2] S. Aleem, L. F. Capretz and F. Ahmed, “Game development software engineering process life cycle: a systematic review,” *Journal of Software Engineering Research and Development,* 2016.

[3] Cornell University, “Java Game Development Resources,” [Online]. Available: http://www.cs.cornell.edu/courses/cs3152/2019sp/materials/programming.php. [Accessed 2020 04 22].

[4] Wikipedia, “Change management (engineering),” 27 03 2018. [Online]. Available: https://en.wikipedia.org/wiki/Change_management_(engineering). [Accessed 2020 04 24].