Change Report

Approach to change management

At the start of assessment 3, our team had an initial meeting in order to establish and distribute the changes that we were to make across each member of the team. We also used this initial meeting to strategize a general scheme in which we would review our approach, the tools we would utilise and separate the processes of implementation and documentation; creating two separate subproblems with a deadline for each. This approach allowed for more collaborative working and overall better management in dealing with the overall task.

Reviewing the requirements for Geese Lightning's game led to the judgement that our vision was analogous to theirs. Therefore, we produced our deliverables following their initial requirements, improving features and refactoring code after completion.

We continued with an agile approach to the assessment as it had worked well with our team in the assessment prior using the scrum methodology. Due to the similar nature and theme of the software that we would be working with, we justified that this approach would be suitable again. Hosting weekly scrum meetings to tackle any problems that had arisen, discuss what progress had been made and what tasks were left to do for each member, enabled us to rapidly produce functioning code and meet weekly deadlines for individual tasks.

After discovering outstanding benefits from using Taiga, we continued using this to create a formal structure to the incomplete tasks discussed in each scrum meeting. Each member had access to this taskboard along with an assigned task for each week. This proved useful for when members finished tasks early and could see which other tasks were still incomplete, offering help for these when needed.

GitHub was another tool that had previously brought our team great prosperity. By the time of assessment 3, our team had overcome any minor issues using GitHub and were confident using this service. Thus, we justified the use of this for managing our code again within this assessment.
The code itself was implemented by the whole of our team where any implementations made would be commented with a summary of what had been added and its function in regard to methods and classes.

Finally, Google Docs was reused for the organisation of our documentation. Our team was again strongly familiar with the software and due to its accessibility and reliability, we had no reason to look for an alternative tool.

Changes to Testing Report

We have made no change to the overall testing method of geese lightning as we believed that the change to the brief was small enough that the justifications for their methods hold up. While we have not changed the overall method, we have added in extra black box tests and peer reviews.

We continued to use test-driven development as we believe that it is the best way to ensure our implementation adhered to our requirements. We did not change anything from the original testing report[1] in this regard.

The way in which we used white box testing was the same as described in the original report[1] These were especially important for us as it helped us to understand how what we were changing affected the rest of the program and makings sure that these changes did not break any existing functionality. In the original report, it was mentioned that they used some IntelliJ features, we did not use these as most of our team has chosen to use eclipse.

We have added additional black box tests to the original document [2] to test new features that we have implemented. We made sure the original tests still passed to make sure we did not break previously implemented functionality. The new tests are highlighted in the updated document [3].

For our requirements testing we have just updated the pass/fail column from the original document[4] to reflect the new requirements we have satisfied with the current implementation, these changes are shown with highlighted text in the updated document[5] None have been added as the requirements have not changed.

One of the tools we did not use from the original testing report is the GitHub project board. While we did not use this board, we used an app called Taiga to check and review code. Similar to the GitHub board we used Taiga to signal when the code was ready to be reviewed before it was pushed on the main GitHub project. We used GitHub pull request to allow us to make comments if the code needed to be changed and merge it if not.

Changes to Methods and Planning

When we began working on the new project, we found the software engineering method used by the previous team to be adequate for a project and team of this size. We had also adopted the Scrum framework for Agile development in assessments 1 and 2, meaning our tools and workflow were already set up to continue utilising this method. Additionally, this allowed us to maintain our Scrum-based team roles, which we have found to be a very effective team structure so far.

For the most part, the tool selection of the last team was consistent with ours [7]. Most of the differences aren't crucial to this assessment or the way our team worked. We used different tools for team communication and the production of charts and game graphics. We chose to continue using the tools our team were using as we saw the changes as largely cosmetic and of little importance to the core game. For example, we produced some additional game assets using the Piskel tool rather than Gimp, which all the already designed assets were developed in. The result of using either tool is largely the same and our team already had experience with Piskel and this outweighed any potential inconsistency caused.

Notably, the previous team made use of GitHub Project Boards as their work management tool. As a team, we weren't familiar with this particular tool and although it seems very useful, we decided the rewards of adopting it didn't outweigh the time cost of learning and setting it up for the whole team. Also, we found our current work management tool to be sufficient and useful in maintaining consistency with the Scrum framework.

The game's code had so far been entirely produced in the IntelliJ development environment. Our team has worked consistently in Eclipse and decided to continue doing so for this assessment. Although both pieces of software are very accessible, we concluded that Eclipse was more appropriate for this project as all students have a good understanding of Eclipse development and, as a team, we have developed a very good understanding of the features of the platform. This decision was largely facilitated by the easy import of the project into an Eclipse environment without any apparent integration issues.

We have made no amendments to our existing Gantt chart, found in our updated method selection and planning, which includes slots allocated to assessment four [7]. In order to remain consistent with the software engineering method we have employed and to maintain our flexibility, we will use the plan as a loose guide rather than a concrete schedule. We believe that the decision to follow the Scrum methodology closely when planning for the next assessment is a good one as, at the time of writing, we do not know in detail what we will be required to implement in assessment four. Therefore, it is important to structure our team and our plans with adaptability in mind.

References

[1] Geese Lightning, ‘Testing report’ [Online], January 2019, Available: https://lloydbanner.github.io/SEPR-Team-7/Testingreport.pdf

[2] Geese Lightning, ‘White box testing evidence’[Online], January 2019, Available: https://lloydbanner.github.io/SEPR-Team-7/White-BoxTests.pdf

[3] Geese Lighting, ‘Black box testing evidence’ [Online], January 2019, Available: https://lloydbanner.github.io/SEPR-Team-7/Black-BoxTests.pdf

[4] Shaun of the devs, ‘Black box testing evidence update’ [Online], January 2019, Available: https://lloydbanner.github.io/SEPR-Team-7/Blackbox3.pdf

[5] Geese Lighting, ‘Requirement testings’ [Online], January 2019, Available: https://lloydbanner.github.io/SEPR-Team-7/RequirementTesting.pdf

[6] Shaun of the devs, ‘Requirement testing update’ [Online], January 2019, Available: https://lloydbanner.github.io/SEPR-Team-7/Req3.pdf

[7] Shaun of the Devs, ‘Method Selection and Planning’ [Online], January 2019, Available: https://lloydbanner.github.io/SEPR-Team-7/Plan2.pdf