Sea of Geese

Project Review Report

Team Management

Throughout this project, our team structure has been based on the assignment of specific tasks to groups. During Assessment 1, we generally assigned tasks to pairs, with website maintenance and record keeping being assigned to individual members. When a task assigned to a pair was completed, the pair would then be assigned to a different task until all the work was completed.

In Assessment 2, we decided we wanted to adopt an approach that could allow for more coordination, and individuals were each assigned a responsibility, including that of a meeting chair. The exception to this was development, which was the responsibility of two people. Flexibility was still a feature of our approach, and team members would help out in different areas as required. At the beginning of the project, more members worked on development, but once the main portion of that was completed, more would work on documentation.

In Assessment 3, we identified two main aspects of game development to be completed. Accordingly, the group was divided into two sub-groups of three, each working on one of those areas. Within each team were two developers and one change manager, who would keep track of the changes and begin the documentation. Due to the limitations of people's skill in coding, we decided to that each pair should have someone who was more proficient at coding and someone who was less so, so that they weren't left to struggle. Once both of these large aspects were completed, some individual members were assigned smaller development tasks, and others were assigned to work on documentation. Finally, all members were assigned to documentation.

We judged that our team management approach had worked very well in Assessment 3, so it was decided to repeat it as much as possible in Assessment 4. Therefore, we identified two main areas of development, with a sub-team of three then being assigned to each. When it came to changing focus to documentation, we made a slight adjustment to our approach and assigned a pair to each different part of documentation. The logic behind this decision was to reduce the risk of documentation being uncompleted due to a team member becoming unavailable. Overall, we consider that this approach has continued to work well.

Development Methods and Tools

For this project, we decided early on to use the Agile methodology. This was because it is suited to customer-centred development involving teamwork and self-organisation [1]. We also felt that it would offer the team more flexibility than an alternative such as the Waterfall methodology [2]. For the specific implementation of Agile, we decided on Scrum [3] as we judged it to be simple to get used to and suited to our team size. The length of Scrum "sprints" varied according to what was required during different stages of development. For example, in Assessments 3 and 4, the initial sprints were week-long as the aim was to complete a fairly large amount of software development. However, by the end, with sprints focussed on completing smaller aspects of documentation, the sprint length had decreased to two days.

At the very beginning of our project, the plan was to use Eclipse as our Java IDE, because it was the one that most of the group was already familiar with. However, we also looked into other IDEs and soon decided that we should switch to IntelliJ, for a variety of reasons, including the integrated support for Version Control systems. It also has very accurate code completion and code analysis that makes writing code so much easier. For example, the IDE will tell the user whether or not their code is reachable, or if a variable has been defined and then never referenced, allowing redundant code to easily be located and removed.

The version control system we used was git, in part because some of the members of our team were already familiar with it and partially because IntelliJ supported it. This made it much easier to push and pull code from the repository. Some members used Git Kraken, a Git GUI, because it's interface was a lot more user friendly.

For the creation of documentation, we decided to use Google Docs as it allowed real-time collaboration and sharing of documents. In addition, Google Drive has served as storage for miscellaneous files that don't belong in the main Git repository.

We initially used Facebook messenger to communicate with each other. While this worked well for informal discussions and planning meetings, we also started using Discord for more formal meetings using the voice call function. This was mostly used during the holidays when the group were not all in the same place, so that we could still have meetings between sprints. We also found that Discord's ability to organise messages into different channels was useful as it allowed us to keep conversations about different aspects of work separate.

Bibliography

[1] Cprime. What is Agile? What is Scrum?, [Online] Available: https://www.cprime.com/resources/what-is-agile-what-is-scrum/ [Accessed: November 6th, 2018]

[2] TPX Manifesto. Agile vs Waterfall: Comparing project management methodologies, [Online] Available: https://manifesto.co.uk/agile-vs-waterfall-comparing-project-management-methodologies/ [Accessed: November 6th, 2018]

[3] Scrum Alliance. Learn about Scrum, [Online] Available: https://www.scrumalliance.org/learn-about-scrum [Accessed: November 6th, 2018]