Team iPatch

Assessment 4: Project Review Report

Christian Pardillo Laursen, Filip Makosza, Joseph Leigh, Mingxuan Weng & Oliver Relph

Team management and structure

Throughout the project, our team had a strong concept of the roles necessary for a competent and stable approach in order to achieve the best possible result.

We believed having a meeting chair was necessary for the productivity of meetings as it makes it easier for the group discussion to be kept relevant and not to divert from the original meeting reasons, which is vital in a short development window, such as the one we had. We initially decided that Christian Pardillo Laursen was best fitted for this role and throughout the project we felt this role suited Christian strengths and required no change. The role of secretary was also of vital necessity in order to arrange meeting times and keep a note of the minutes. We initially placed Filip Makosza in this role and, in a similar fashion to before, felt that Filip was well prepared for this role and felt no need to change. In addition, the roles of librarian and editor were given to Mingxuan Weng and Josh Wakefield respectively. These roles were imperative to the collection and construction of vital documents which played a large part in project progression as well as project tracking. As a team we felt Mingxuan executed this role well and believed that there was no change necessary, however, due to an unforeseen complication we lost a team member and had to reassign the editor role to be shared between the rest of the members, as so was stated in the risk assessment. This was a high impact change and so productivity was hindered slightly for the first week, but after that period we maintained our original efficiency and managed to meet all deadlines.

After our initial planning stages we felt it was also necessary to have a main developer team which consisted of Christian Pardillo Laursen, Filip Makosza and Joseph Leigh due to their previous experience and confidence in their coding strengths.

In terms of team management we had plans in place if any of the risks, such as our loss of a team member highlighted previously, took place and affected any of our team or our roles. For example, in the case a team member fell unwell to the point of not being able to participate in a meeting, then any of the other members, without a role, would be on hand to fill in and make sure we didn't fall behind as a group.

These team roles were put in place during planning and before implementation started as we believed they would be best suited to our team, based on the fact that we were all fairly new to each other, we didn’t know each others strengths and weaknesses. We discussed this beforehand and decided that, should our current team roles not be satisfactory, then we would change them or maybe the whole structure of the roles entirely. We had a back up plan where if the roles where to be insufficient then, as a team, we would have a vote on the new assignment of roles or, if a certain role needed more attention, we would shuffle around the roles in order to have multiple people helping on certain ones. For example, during the implementation of our project, at some points we felt that having 3 main developers was not enough for the amount of coding required and so other members helped in those roles whilst the (at the time) less time critical roles were placed onto 1 or 2 people rather than 3.

Development methods and tools

During the course of the project we didn't use too many varying software development methods and tools because we saw the ones we originally selected to be viable for the task at hand. From the start of the project we used Github[1] for version control. It allowed us to create branches to test out potential features for the game, it also allows us to comment on each others pull requests to discuss code integrity and feature necessity. The use of Git hub also meant that if we added a feature that caused errors or accidentally committed code which overwrote other important features we could change our master branch back to a previous version. For assessment 3 and 4 we used Travis CI[4] this checked for errors in the code we submitted to the repository. For assessments 2-4 we used the jMonkeyEngine[5] Its inbuilt functions and objects made constructing a three-dimensional game quite manageable leaving us with minimal work to do in regards to physics and collisions, also with Github integration it made it very easy to update our repository versions. We used Discord[2] as our method for communication throughout the whole project as it was the easiest way for us all to communicate.

Two of the main software development methods we used throughout this project were SCRUM[3] and blackbox testing. Because the majority of the software work was done by a small portion of our group we communicated feature plans and progress verbally or over Discord. Even though we drafted an early Gantt chart we didn't end up following it perfectly due to conflicting schedules and problems during the software development process. We used SCRUMs because they were quite flexible and fit into our weekly meetings. We used blackbox testing as it seemed to be the most suitable method of testing our code due to the vast variety of situations which could occur during runtime, so it felt more suitable to test them during runtime with specific test cases rather than using software based tests. There wasn't much change in our methods throughout the project because what we had worked so we felt that there was little need to change.

One of the main risks that changed during our project was risk 4 losing a group member, we lost one of our group members within 2 weeks of the start of assessment 1 meaning that if the risk were to occur again the impact would have been even more severe that when it first happened, in order to manage this risk certain members of the team took on extra work and roles in order to pick up the slack left by the missing member. Other than this our original risk evaluation turned out to be rather accurate so we worked hard to avoid risk cases and if they did occur we dealt with them appropriately.

References

[1]"Build software better, together", *GitHub*, 2019. [Online]. Available: https://github.com/. [Accessed: 30- Apr- 2019].

[2]"Discord - Free Voice and Text Chat", *Discord*, 2019. [Online]. Available: https://discordapp.com/. [Accessed: 30- Apr- 2019].

[3]"What is Scrum?", *Scrum.org*, 2019. [Online]. Available: https://www.scrum.org/resources/what-is-scrum. [Accessed: 30- Apr- 2019].

[4]"Travis CI - Test and Deploy Your Code with Confidence", *Travis-ci.org*, 2019. [Online]. Available: https://travis-ci.org/. [Accessed: 30- Apr- 2019].

[5]"jMonkeyEngine", *Jmonkeyengine.org*, 2019. [Online]. Available: http://jmonkeyengine.org/. [Accessed: 30- Apr- 2019].