UNIVERSITY OF YORK

DEPARTMENT OF COMPUTER SCIENCE

ENG1 Group Assessment 1
Team 1

Auber

Plan1.pdf - Method Selection and Planning

Group Members:

Jonathan Davies

Jamie Hewison

Harry Smith

Annie Sweeney

Zee Thompson

Mark Varnaliy

Software Engineering Methods

We have chosen SCRUM as the methodology of choice for our group project due to the fast pace nature of the assessment, and the small team we are working in. Given that we only have a few weeks to have the project and other deliverables ready, we do not have time to focus on other excessive and unnecessary documentation. We are able to have weekly meetings in which we discuss what we are working on, and the goals we aim to have achieved by the next meeting. We are able to have a few short sprints over the course of the project, allowing us to flesh out each deliverable sufficiently [1].

Development & Collaboration Tools we used:

- GitHub Pages
    - We have used GitHub Pages in order to allow us to create our team's website, as it allows us to make one with the functionality of a github repository. This allows all team members to easily access and contribute to the site. This will also make it easier to hand the website over to the next team.
    - As an alternative to this we considered using basic web hosting tools to make our website, but decided that GitHub Pages was the best method since we already had a group repository in place.
- Google Drive
    - Google Drive was chosen to host our group project files as it allows for simultaneous access and editing of the project's documentation. Again, this will make it easy to hand over the project to any groups that may choose it.
    - As an alternative to Google Drive, we considered other, similar solutions such as OneDrive. However, since the University provides us with our Google Drive accounts with near unlimited storage, we figured this'd be the best option as everyone would definitely have access to it.
- libGDX
    - As a team we searched the internet for game engines that support Java, libGDX was a great choice as it is widely adopted [2]. There is lots of helpful information to learn how to use the engine [3]. We had also heard of other groups choosing to use this engine, and figured that it would make it easier for our transition if we had used the same engine as the group we swap with.
    - As alternatives we considered other Java game engines, such as LITIENGINE and jMonkeyEngine. However, libGDX was eventually chosen as it seems the easiest to work with and is most widely adopted in industry, and we knew other teams were working with libGDX, so it'd be easier to swap projects if we already knew how to use this engine.
- Git (via GitHub)
    - We decided that a distributed version control system would suit our project best, as we are unable to group together to work. We chose Git (using GitHib) for this as it is an industry standard tool [4], and some group members already knew how to use it. This distributes the project and allows us to work asynchronously, merging our code and resolving conflicts to make a functioning game.

    - The alternative to using a Git VCS was to use one based on Apache Subversion (SVN). However, SVN's features are quite limited compared to Git and it discourages branching - making it harder to try different things separately in the code. For this reason, we decided to stick with Git for our project.
- IntelliJ IDEA
    - We decided to use IntelliJ IDEA as our IDE to create our project. We chose this as it was easy to install and begin using, and a lot of the getting started documentation for LibGDX is based on IntelliJ - making it very easy for us to get started.
    - As an alternative, we considered using the Eclipse IDE. However, some of us tried that and found it difficult to get it to create our first LibGDX project. For this reason, we chose to stick with IntelliJ IDEA as that meant we could closely follow the getting started guides and we could get on with the project.

Approach to Team Organisation

As a group, we opted against the traditional rigid team structure of having a leader, followed by the rest of the group having concrete roles. Instead, we decided that we would all be equal, discussing our tasks in our weekly meetings and updating the plan accordingly. This way, every group member holds each other accountable, and the responsibility of making sure everyone is on top of their work doesn't fall to one person. This also means that there is no risk of a team leader falling ill and letting the whole project fall behind, and if a team member does fall ill, the tasks that they were doing can easily be re-distributed equally between the rest of the group.

Project Systematic Plan

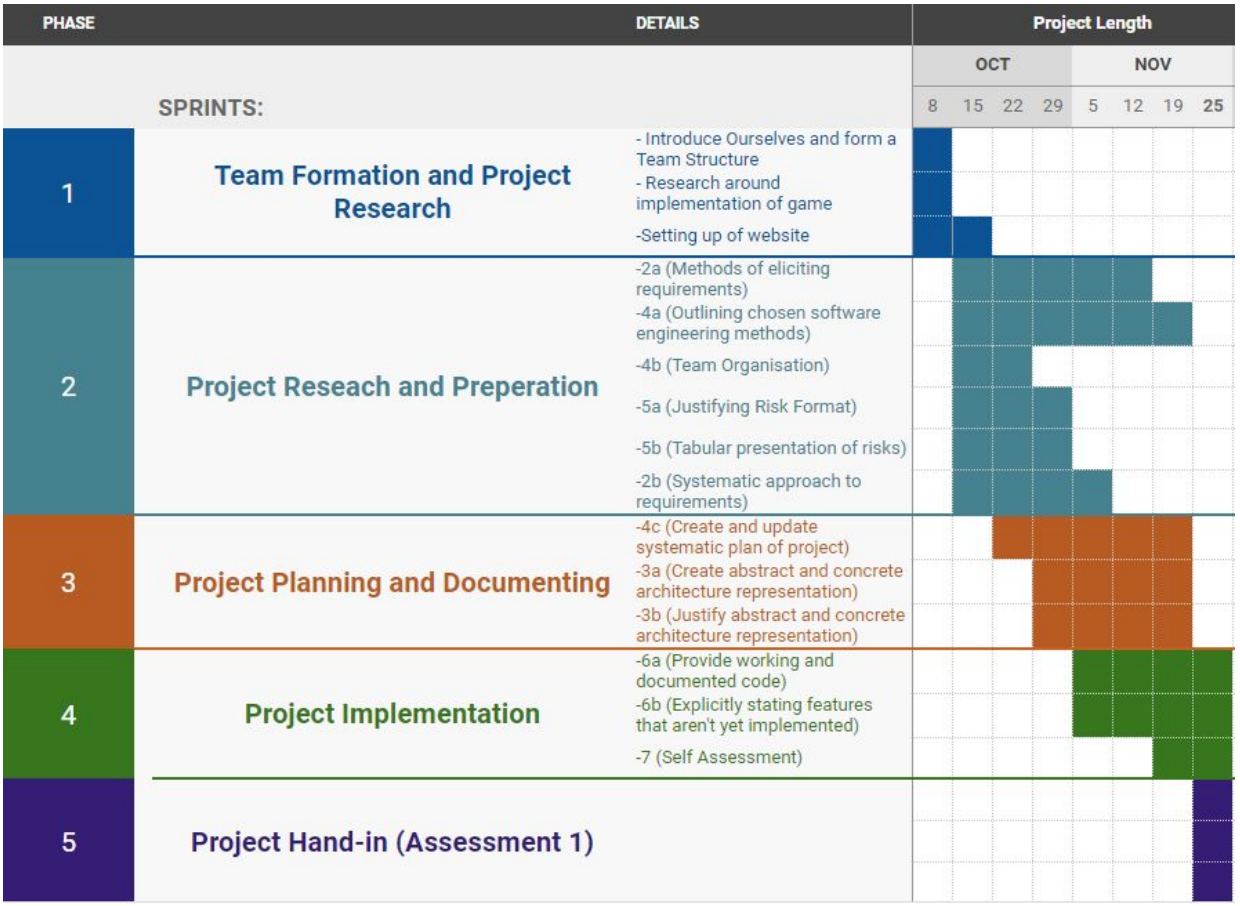
In order to create a systematic plan for our project, we created a Gantt chart so that we could easily see what each week's tasks are, and whether we are on, behind or ahead of schedule. The Gantt chart makes this easy as it lays out each task along with a description, and shows a visual representation of each task's start and end dates. We decided that the best method of prioritising tasks was simply to work through the assessment document from start to end, as the majority of later tasks are dependent on earlier ones. However, there are a few instances where we decided later tasks were important - such as if they might take slightly longer - so we decided to move them towards the top of the Gantt chart. We decided that an explicit critical path wasn't necessary, but it can be identified by the starting dates of each task. For example, tasks which start later can't be done until at least a proportion of the tasks above it have been completed. This proportion can be identified by the delay in the starting dates - if tasks A and B were both 4 weeks long and B is listed as starting 1 week later than A, then this implies that ¼ of task A must be complete before task B can begin.

This Gantt chart was updated weekly during the project to reflect any changes in how we were approaching each task. If we became delayed on one part of the project, the Gantt chart would be updated to show this task taking longer than expected, as well as any delays

on parts of the project relying on this part. This method evolved somewhat during the course of the first assessment, as we approached the hand in date. At the beginning, the chart was just used as a rough outline of what we expected to do over the course of the term, however, by the end of the term, it evolved into a more rigid plan, showing what we still needed to get done in time for the hand in date.

Weekly snapshots of our Gantt chart can be found on the project website in order to show how the plan progressed over the course of the project.

An example snapshot of the Gantt chart is shown below. This is how the Gantt chart looked during our weekly team meeting in Week 7 (Thursday 12th November 2020).



References

[1] "What is Scrum?", *Scrum.org*. [Online]. Available: https://www.scrum.org/resources/what-is-scrum. [Accessed: 12- Nov- 2020].

[2] "Best Java game engines as of 2020", *Slant*, 2020. [Online]. Available: https://www.slant.co/improve/topics/21245/~java-game-engines. [Accessed: 12- Nov- 2020].

[3] "Introduction", *Libgdx.badlogicgames.com*. [Online]. Available: https://libgdx.badlogicgames.com/documentation/. [Accessed: 12- Nov- 2020].

[4] A. Chhatpar, "GitHub has become the industry standard for good reasons", *TrustRadius*, 2017. [Online]. Available: https://www.trustradius.com/reviews/github-2017-09-01-04-18-55. [Accessed: 12- Nov- 2020].