Method Selection & Planning

Mario

GROUP 28

Joseph Frankland

Anna Singleton

Saj Hoque

Leif Kemp

Shi (Lucy) Li

Hugo Kwok

4(a) Method Selection and Planning:

To develop this project we're going to be using the Agile development methodology and work in sprints of varying lengths depending on the complexity of what is going to be implemented, focussing on specific features of the project.

We'll be working on the sprint using 4 key stages: Design, Implementation, Documentation and Testing. Then iterating on these stages if necessary.
The design stage is predominantly about the conception of the ideas, not necessarily how they are going to be added, but more so about the core fundamentals of the functionality and how well we believe it'll integrate into the existing system. We may also decide to build this function in a test-driven manner, so we may design the tests and outcomes we want from this implementation in this stage before we start building it, which provides a stronger framework with regards to what we want our feature to do and not do.

The implementation stage is taking what we've agreed on for said functionality and bringing it into our game and figuring out the best and most optimal way to do so. To develop this project we are going to be using LibGDX, which is a game development framework that uses Java and allows us to develop our game for multiple different platforms without requiring modifications of lots of code in order to support them. jMonkeyEngine was also considered as an engine of choice, but because the majority of us have not worked with Java based engines before, we've gone with LibGDX as it provides more extensive and clear documentation which is better for starters.

The documentation stage will happen alongside the implementation stage, which will allow us to easily refer back to it if necessary and also provide clear instructions for other developers going into this code. Working on it as we develop the functionality as well will reduce the risk of missing something out when documenting it.

The testing stage is where we determine whether or not the addition works, then noting down how to improve it should it need improving or finding and fixing any bugs we find. If we find that this functionality hasn't been implemented in a satisfactory way the sprint will be repeated from the necessary stage until we deem it satisfactory or decide to scrap it for whatever reason.

Using a waterfall approach to developing this game wouldn't be preferable as we would have to figure out how every single feature would work at each stage, which runs the risk of poor implementation in practice, missing out documentation or creating tests that are too complex or not complex enough (i.e we don't end up testing a certain feature), whereas using the Agile methodology means that when we do a sprint on one feature, it will be rigorously designed, tested and will meet a high standard alongside the rest of the system.

The main collaboration tool we will be using for this project is GitHub/Git, which allows us to maintain a repository for the game that we can push, pull, branch etc. This allows each of us to work on the game independently of each other, without having to wait for one particular feature to be finished from someone else's end thanks to branching, and runs no risk of overwriting another person's work.
Other software that has been considered is Google Drive, which is convenient and quick to set up but is clunky to update and pull whenever a change is needed.
GitHub also synergises the best with the Agile methodology as each sprint can be given its own branch as we work on it, which is incredibly powerful for developing our project as it will keep it contained from our main, functional branch, allowing us to develop and test it without running the risk of tampering and breaking the main game.

4(b) Team Approach

For our team’s organisation strategy, we decided to split into smaller sub-teams, but with no defined leader. We chose to split into smaller teams as we believe that larger teams can be too unwieldy if not managed very carefully. We feel that we are not experienced enough to work in a single large team and not get in each other’s way, and we feel that with two smaller teams working on separate parts of the project at the same time, we can organise ourselves much more efficiently. For example, one sub team worked on question 2, whilst another team worked on question 3. This is good because it allows us to work at our peak efficiency since we are not redoing work that other team members have already done.

In terms of the project, the two sub teams method is a good fit, because it's a small project that has a small time frame. The two teams approach is good for this because it minimizes the amount of bureaucracy that organising a single larger team can bring, which is especially important with the limited time we have for this project. Since it's a small project, a section of the project can easily be covered by a subset of the team as a whole, meaning it's not necessary to have everyone work on it at once.

This is the method we have been using for the pre-implementation work, and we plan to continue with a similar method for the implementation, whilst also using an agile methodology as described in part (a). We will split into 2 teams of 3, each working on a different section of the project, with each team having a programming lead, and simultaneously working on documentation and code. This will work well because it will allow us to split the workload, and each team member can play to their own strengths, and get a good amount done.

4(c) Systematic Approach

Link to progress of group over the course of the project
https://annabeths.github.io/ENG1-Project/snapshots

Key tasks (starting and finishing dates, task priorities and dependencies)

Start of project

- 2021-11-19

Interview Planning

- Started on 26-11-2021
- Finished on 1-12-2021
- Lasted 6 days

Interview

- Starts when interview planning finished
- Started and finished on 2-12-2021
- Lasted 1 day

Requirements

- Starts after the interview had taken place
- Started on 3-12-2021
- Planned to last 14 days
- Planned to finish on 16-12-2021
- Final formatting and details accomplished on 28-1-2022

Architecture

- Starts after the interview had taken place
- Started on 3-12-2021
- Planned to last 14 days
- Planned to finish on 16-12-2021

Abstract Architecture

- During 17-12-2021, we split the architecture down into 2 parts: abstract and concrete.
- We have rescheduled the latter half of architecture (the concrete part) to during implementation as it could not be done yet at that moment.

Concrete Architecture

- Concrete architecture requires more implementation details, which had only been decided later on during the project. Hence postponed.
- Planned to start on the 7-1-2022
- Planned to last 17 days
- Planned to finish on 24-1-2022
- Finished on 1-2-2022

Method Selection & Planning

- Starts when requirements finished
- Planned to start on 17-12-2021
- Method Selection & Planning was given an earlier start date as 4a is now complete, while we were waiting on the completion of 3a to begin work on 3b
- Actual start date: 10-12-2021
- Planned to finish on 23-1-2022

Risk Assessment & Management

- Starts when architecture finished
- Planned and started on 17-12-2021
- Planned and finished on 23-12-2021
- Final formatting and details accomplished on 28-1-2022

Implementation-Section

- Planned to start 1-1-2022, delayed to 3rd then 10th due to the January Common Assessment Period
- Planned to last 27 days
- Planned to end 29-1-2022
- Finished on 1-2-2022

End of Project

- 5-2-2022.

Changes to our approach during project

The largest change to our original plan was the splitting of the abstract and concrete architecture into different time periods. We decided on this as we realised that in order to complete this we would need to have a sufficient amount of the implementation completed before we could start on the concrete architecture. Our original plan also deviated from its course as we did not account for the busy schedule during the Common Assessment Period, leading to a week's delay of starting the implementation stage. Requirements and Risk assessment finished a lot later than expected due to confirmation of specific details needed from the customer.