Group 21
Team 21

Method Selection and Planning

Doaa Doukh
Surbhi Lahoria
Sean Abong
Peter Beck
Isaac Ohara
James Cretney
Lloyd Newton

a)

During our treasure hunt activity, our group researched various engineering methodologies and software tools to help us before starting our project. We read the brief and looked through the project's requirements before starting this to get an idea of what we had to do.. Initially we chose Google Drive for document sharing and access as its user friendly interface and real time collaborative capabilities are extremely useful especially for simultaneous task management across the team- in our case being user requirements and risk management tables.

For our game engine we compared Libgdx and jMonkeyengine, however, we decided to use Libgdx due to its compatibility with Java. It also suits 2D games, whereas JMonkeyengine offers better tools when creating 3D games which isn't our goal for this project. There is also extensive documentation and tutorials regarding Libgdx which can be a useful guide for us as we aren't familiar with it.

When selecting an IDE we decided on IntelliJ IDEA over Eclipse due to the team's familiarity with it, moreover, it integrates LibGDX and Gradle easily through the build automation tool which should be more efficient and better to use.

We used a Gantt chart during the first two weeks to have a brief overview of when we wanted each section to be completed, and help us manage our project timeline better. However, as we progressed through the project we transitioned to a Kanban board as it offers more flexibility so we can dynamically assign and monitor each other's tasks and manage the workload.

To design our maps we used a software called Tiled, it allows us to use tilesets we found to create our own map to use for our game. It has a user friendly interface and is customisable which is very important in order to make it unique. As well as it being implementable with Libgdx. It is also free and there are many open source tilesets available online which we could find and use to our liking.

We used PlantUML to create our structural, behavioural and class diagrams as well as the Gantt chart. We can create different diagrams such as sequence and use case diagrams (reference them) and as you are editing the code you can see the diagram being changed which is an advantage. We can also integrate it with google docs easily and others can collaborate on a diagram. Before using this we used draw.io to create the work breakdown structure as we were familiar with this software previously and wanted to create it as soon as possible.

To share and access our code we used GitHub (web and desktop version) by creating an organisation called 'Team21Eng1' and it had multiple pages/repositories: HeslingtonHustle21 (the game), Behaviour diagrams and Team21Website. Our main repository had 5 branches: main, Altering-add-Event, GameplayLoop, TiledCollisions and map which we then merged into main. This is so the developers can each edit and work on the game without risking corrupting or deleting previous versions of the project.

b)

Our team's approach to team organisation and project planning was to split the work among different members of the team and collaborate together on different aspects of the project. This happened up until week 3 where we had assigned our roles which were:

- Peter as lead developer
- Surbhi as game designer
- Sean as developer
- Lloyd as technical writer
- Isaac as developer
- Doaa as project manager
- James as web developer and technical writer

Although we had specific roles, some of us worked on both the documentation and development when necessary like in week 5/6 in order to complete everything we needed in time.

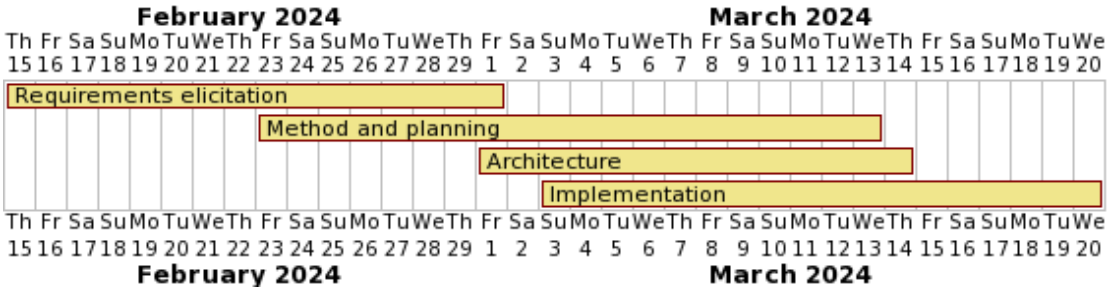
We also had a document to keep track of everything that happened whenever we met Meeting minutes. On our Friday practical sessions our group discussed what we needed to do before our next meeting (on Mondays) and started working during the lab. In our meetings on Monday we mainly showed what we all did at home and continued working on our parts.

We created a UML diagram on draw.io to show our work breakdown structure for the project as seen on the website in the diagrams section. This made it easier when planning everything and assigning work to each person in the group. Work breakdown also found in the diagram section on our website.

c)

We decided to create a Gantt chart to give an initial overview of when to complete the various parts of the project. We didn't assign roles at the start so we could see each other's strengths and assign tasks later on, and instead created a rough model showing how we would split the workload up throughout the work period.

Our group took a flexible approach to this project where we assigned tasks as we went along and helped each other as well as checking each other's work to make sure we checked off what was needed for assessment 1. We continued to meet regularly (in person) so the project manager could see that progress was being made to ensure everything would be done before the deadline.



Instead of creating a Gantt chart for each week, for more detailed parts of the project, we decided to use a Kanban board as our way of seeing who's doing what each week, as well as what has been completed and what's in progress. On Kanban you can see who was working on which task every week. Kanban boards or on our website in the diagrams section.

This is a summary of what was being completed each week:

Week 1:

This was the first week our group had met each other during our Friday lab session. We created a whatsapp group so we could communicate when to meet and started researching different software engineering methods, how to work as a team and the tools we needed to use for our project. We also read over the brief together as a group and made a shared Google drive.

Week 2:

At the start of the project we started working on the documentation (such as the obvious user requirements) together during the practicals and a few non-timetabled group meetings, and splitting the tasks in smaller groups. We did this as we waited for the customer meeting which was the following week. Peter, Isaac, Lloyd and James worked on the user and functional requirements while Doaa, Surbhi and Sean worked on the risk management.

Week 3:

This week we started working on the initial UML diagrams such as the work breakdown. We also booked the customer meeting for Tuesday 27th February at 2:30pm and all of the team were there to ask the customer our questions which we came up with this week. In the practical we also set up our Github repository and created a discord server for our group to communicate with each other more easily.

Week 4:

This week we completed our user & functional requirements after collecting more information from the customer, started on the main map and started the CRC cards. The website was also created and we started working on the structural and behavioural diagrams. The assets and map rendering had started to come together by our lead developer, Peter.

Week 5:

We began to work on the interior maps and completed the main map, implementing the main menu screen as well as implementing the back end of the code (score, time etc.),animations, events, scoring and started collisions. The development team were working on their code on separate branches and while this was happening the documentation team were completing the method and planning as well as updating the CRC cards and rest of the architecture. We updated each other on what part of the code still needed to be completed and made sure to review the brief for what we needed to include for Assessment 1.

Week 6:

For the final week, we merged the branches together for the code and added the code documentation as well as checking that everything worked together. We also worked on the implementation document and included what assets we used and the architecture document to complete everything once the code was finished. Then we implemented the diagrams onto the website and checked everything was in the correct format ready to be submitted before the deadline.