Method Selection and Planning

Assessment 2 - Updated from Team 21

Hannah Thompson
Kyla Kirilov
Ben Hayter-Dalgliesh
Matthew Graham
Callum MacDonald
Chak Chiu Tsang

Assessment 1 (Team 21)

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.

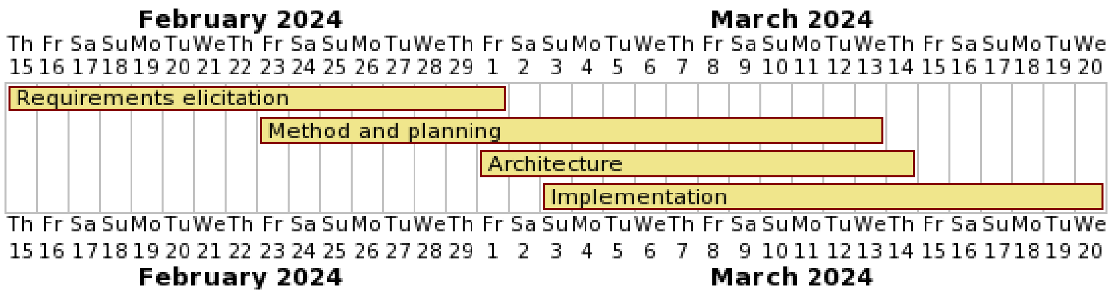
We also had a document to keep track of everything that happened whenever we met Meeting minutes . On our Friday practical sessions our group discussed what we needed to do before our next meeting (on Mondays) and started working during the lab. In our meetings on Monday we mainly showed what we all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

Assessment 2 (Team 25)

Engineering Method:

Our team adopted an agile approach to software development [1] in which we divided the semester into one-week sprints. This allowed us to define a set of requirements, implement the required code, review the outcome, and then consider a new set of requirements. In each sprint, we completed this cycle and used the Friday practicals sessions to discuss our progress across the week.

Taking an agile approach helped us to adapt to changing requirements and break development up into a series of steps, each of which was planned and documented. Similarly, it promoted development of both the documentation and code simultaneously, which encouraged good practice (keeping documentation up to date) and organisation.

Team Structure:

When team 25 took over this project, we assigned corresponding roles to our team members to mirror the original team's planning and structure:

- Kyla as lead developer
- Ben as game designer
- Matthew as developer
- Hannah as technical writer
- Callum as technical writer

- Fergus as developer and technical writer

These roles were equivalent to those assigned a few weeks into our weekly meetings, once each team member had decided on their strengths and weaknesses. These designations play into our individual talents and experience, placing those with stronger coding skills into implementation roles, and those with stronger written literacy into documentation development.

We review these roles weekly, ensuring that everyone is comfortable with the tasks they are assigned and is able to advance without dependencies on other team members. Where these dependencies arise, team members work together to eliminate the bottleneck, allowing everyone to progress.

The Gantt charts showing snapshots of each week can be found on our website, under 'Method Selection and Planning'.

Note: As some of these diagrams are very large, it may be easier to Right Click → Open in New Tab to view all of the contents.

Week 7: ***Fig. 1***
We prepared our project for the presentation, ensuring that all documents were publicly available and that our powerpoint was informative. Hannah, Kyla and Ben delivered the presentation, while the other members were available to answer questions.

After every group's presentation, we met to look at all the games and select a project to take over. We decided that Group 21's game was the most well-structured, and their documentation was accurate and up-to-date, so we chose this project to continue.

Week 8: ***Fig. 2***
We spent a week preparing and planning for the next phase of the project. The documentation team gathered the necessary documents and began updating the deliverables and writing the change report, while the development team worked through the existing code and started implementing new features. Kyla designed the assets needed for this phase of the assessment, and Chak Chiu began writing automated testing for new and existing classes.

Week 9: ***Fig. 3***
Work continued on the elements mentioned in week 8. Hannah updated the website to accommodate the new requirements, and Ben began implementing the new 'achievements' class required for assessment 2. At this point, Hannah also took over the planning of the scoring algorithm, since work on the documentation side was lighter than the development side. The documentation team continued to update the original deliverables and upload content to the website.

Week 10: ***Fig. 4***
We moved onto implementing the scoring and achievement algorithms, as well as new settings and controls screens. The documentation team completed the change report while the development team added the new classes to the project. Chak Chiu continued to focus

on implementing tests for each new class. At this point, we were hoping that the achievements functions would be complete, but since we had issues adding them to the incomplete end screen, we pushed implementation back to Week 11.

Week 11: ***Fig. 5***
At this point, we realised that we had fallen behind on implementing all the tests, and were running into issues with some classes being more difficult to mock than others. Chak Chiu re-designed the testing system to be compatible with all classes, and created a checklist to keep track of the testing progress. Ben paused his work on the scoring and achievements algorithms to assist in fixing the testing, while Chak Chiu set up GitHub Actions to automate the build and test process on each push.

During this week, Hannah and Callum focused on completing the Change Report and writing the Testing document. Meanwhile, the whole team planned the User Evaluation process and prepared the necessary documents.

Week 12: ***Fig. 6***
We intended to complete user evaluation this week, but were forced to delay it in order to finish some key aspects of the game, including the controls and settings screens and the finalised scoring leaderboard. The user evaluation therefore ran into the start of Week 13, delaying our completion of the user evaluation document by a week.

The entire team focused on implementing the final code changes, finalising the testing and fixing any small bugs or errors that we encountered in the game. We also added screen and GUI elements to make the game more understandable for new players, as we thought that at this point it would not be particularly beginner-friendly which breached a number of our usability requirements.

Week 13: ***Fig. 7***
In the final week of the project, the main focus was finishing any documentation that had been delayed due to waiting on implementation, and to prepare for the assessed presentation. We also decided to make the leaderboard global, so set up the website and a server to handle this. The final versions of each screen were combined to make the final game, and the last user evaluation sessions took place.

As of the morning of 22/05/2024, the final things to be completed for this assessment are adding the documents to the website, building the final executable and generating the final architecture diagrams.

References:

[1] I. Sommerville, “3.1 Agile Methods ”, in *Software engineering*, 10th ed. Boston, Mass. Amsterdam Cape Town Pearson Education Limited, 2016, pp. 57-62