University of York Department of Computer Science
Engineering 1

Methods Selection & Planning

Cohort 3 Team 5 - alltheeb5t

Aaron Heald
Alex Gu
Arun Hill
Jade Stokes
Maksim Soshchin
Meg Tierney
Will Hall

Methodology

When working on a large project it is essential to use an appropriate programming methodology to keep the project organised and delivered on time. We researched 3 different programming methodologies and compared their strengths and weaknesses.

We first looked into the waterfall methodology, this is where you follow a linear development path going from plan to final product. It is divided into different phases of development such as Requirements and Implementation which needs to be reviewed and signed off before moving onto the next phase. Although this is the simplest to understand and use, it doesn't promote change because once you have signed off on a phase you can't go back and change it. For this reason we didn't choose to use the waterfall methodology as we will constantly be suggesting changes and new ideas to improve the project.

Following this, we researched the spiral methodology. In this methodology you repeatedly work through development cycles which consist of 4 stages; Planning, Risk analysis, Engineering and Evaluation. This allows for constant iteration and improvement of the software since flaws and new ideas are always being evaluated and implemented. However, this methodology requires a lot of planning and management which is intended for larger projects alongside a focus on risk analysis which isn't needed for this project.

Our sprints followed an outline set by the Scrum methodology. We held meetings on Monday afternoons to discuss what is needed from each member of the team for the week and then work on our tasks throughout the week and review how the tasks were progressing during our Friday practical. We also kept track of which tasks were ongoing with a short tracker at the end of each meeting, stating who was in charge of a certain task and whether it was open or closed. As such, we ended up following the recommended Scrum pattern for our meetings:

- **Sprint Planning:** On a Monday, we would meet and decide what parts of the project needed to be worked on for that week, and what was needed from each member of the team. It was important that everyone understood their role so we could all move forward with a key goal in mind for that week.
- **Scrum:** The team would keep in communication with each other throughout the week using our chosen messaging apps to ensure work was going smoothly and if any problems had been run into.
- **Sprint Review:** On a Friday, we would showcase what work had been completed through the week, and see how much work needed to still be done or changed over the weekend. This gave us an opportunity to find out what wasn't working during the week and reassign any tasks or deadlines for the following sprint.
- **Sprint Retrospective:** Before our next bout of planning on a Monday, we would try to find what went well during the last sprint and keep that energy for the following one, whilst trying to find solutions to any problems that had come up so that we could continue development smoothly.

Collaboration & Development

As a team we decided to go with the Google Suite as our main set of tools due to their great collaboration features, ease of use, and native version control features. We explored other sets of tools such as the 'Microsoft 365' set of tools which the university also provides, however we found that it was easier to use Google Drive on our personal computers as we were already logged into the university email on Gmail rather than having to log into our university accounts on Microsoft Office just to access Word.

For the implementation, we decided to go with Github as our code version control system and a mix on the development team of IntelliJ and VSCode as our IDE so that the development team could choose whichever they felt most comfortable with, as by this point everyone was working on completely different aspects of the project. To create our structural, behavioural and Gantt Chart diagrams we used PlantUML as it was online and had fully written out tutorials to make any kind of diagram. For the artwork we used a mixture of Aseprite, Microsoft Paint and Photoshop to create the UI and sprites owing to their versatility and plethora of features. When making the music and sound effects we used Ableton Live 12 since it is a fully featured digital audio workstation with industry standard tools and effects.

As for communication, we chose WhatsApp to be our main point of contact. We thought about mainly using Discord due to its ability of being able to share your screen whilst in a team call, but decided that this would be better used for the development team to troubleshoot and ask questions, as well as separating out the chats for each part of the project if need be. We decided that whatsapp would be better for general chat about meeting times and if anyone wasn't going to make it to a meeting rather than specific questions about the project. This was good because it meant people didn't have to scroll back through a conversation about a part of the project they weren't working on if they needed information about the next meeting.

Organisation

We chose to organise our project by sticking to the scrum method of agile development. To make sure that everything in the project ran smoothly, we decided to assign areas of focus for each person, ensuring that everyone was a 'Product Owner' in at least one area. For the bigger sections, like the Change Report and the Implementation, we ensured that multiple people were made the Product Owners so that the workload wasn't too heavy for one person.

Our assigned focus areas were as follows:

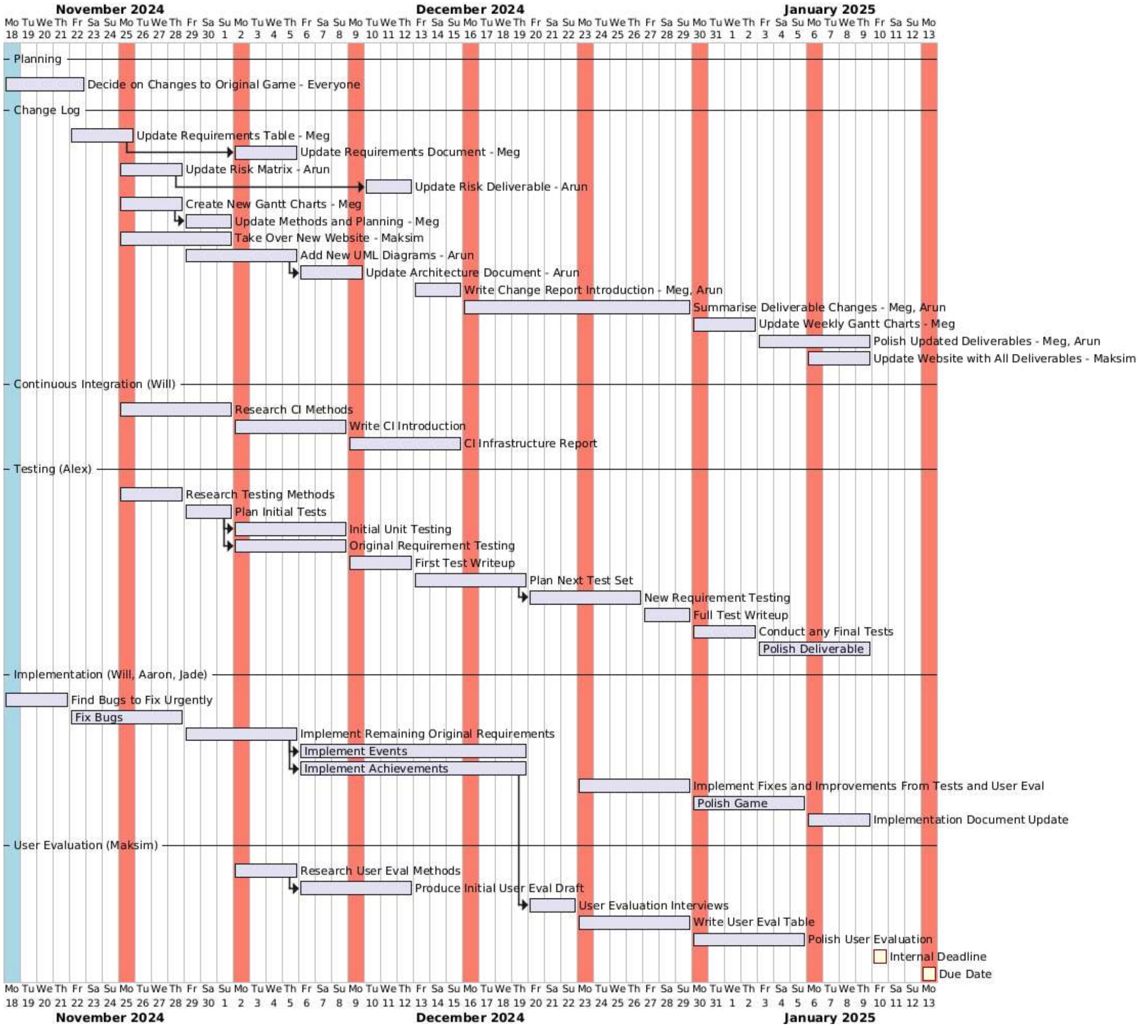
- **Website: Maksim**
- **Change Report: Meg** (Requirements, Methods) **& Arun** (Architecture, Risk Assessment)
- **Implementation: Will, Jade, Aaron**
- **Testing: Alex**
- **User Evaluation: Maksim**
- **Continuous Integration: Will**

Giving each other specific areas of focus for the project meant that we could easily define what each person was meant to be doing for the week based on what part of their deliverable was needed for later parts of the project. This ensured that no member of the team was unclear on what their role was for this project and that nobody felt like they weren't making a contribution.

We also decided on Arun to make a record of our meetings and keep track of any ongoing tasks at the end of each meeting to ensure that we all knew what was expected of us.

Each week we had 2 meetings, one on a Monday and the other on a Friday. On Mondays we discussed how the project was going and outlined the tasks for the week alongside creating that week's gantt chart. On Fridays we reviewed that week's work and suggested any changes which could be made over the weekend before the next meeting. In each meeting we updated our logbook which kept track of attendance and what was discussed in the meeting. Because of the winter break in the middle of the project, we weren't able to keep to this meeting schedule all the time as everyone was busy at different times, so we decided to keep each other updated through our Whatsapp group and the development team kept in contact through discord with any reported bugs and issues they were having. When we came back to York, we made sure to have a meeting to see what progress everyone had made in person and to begin the process of wrapping up the project.

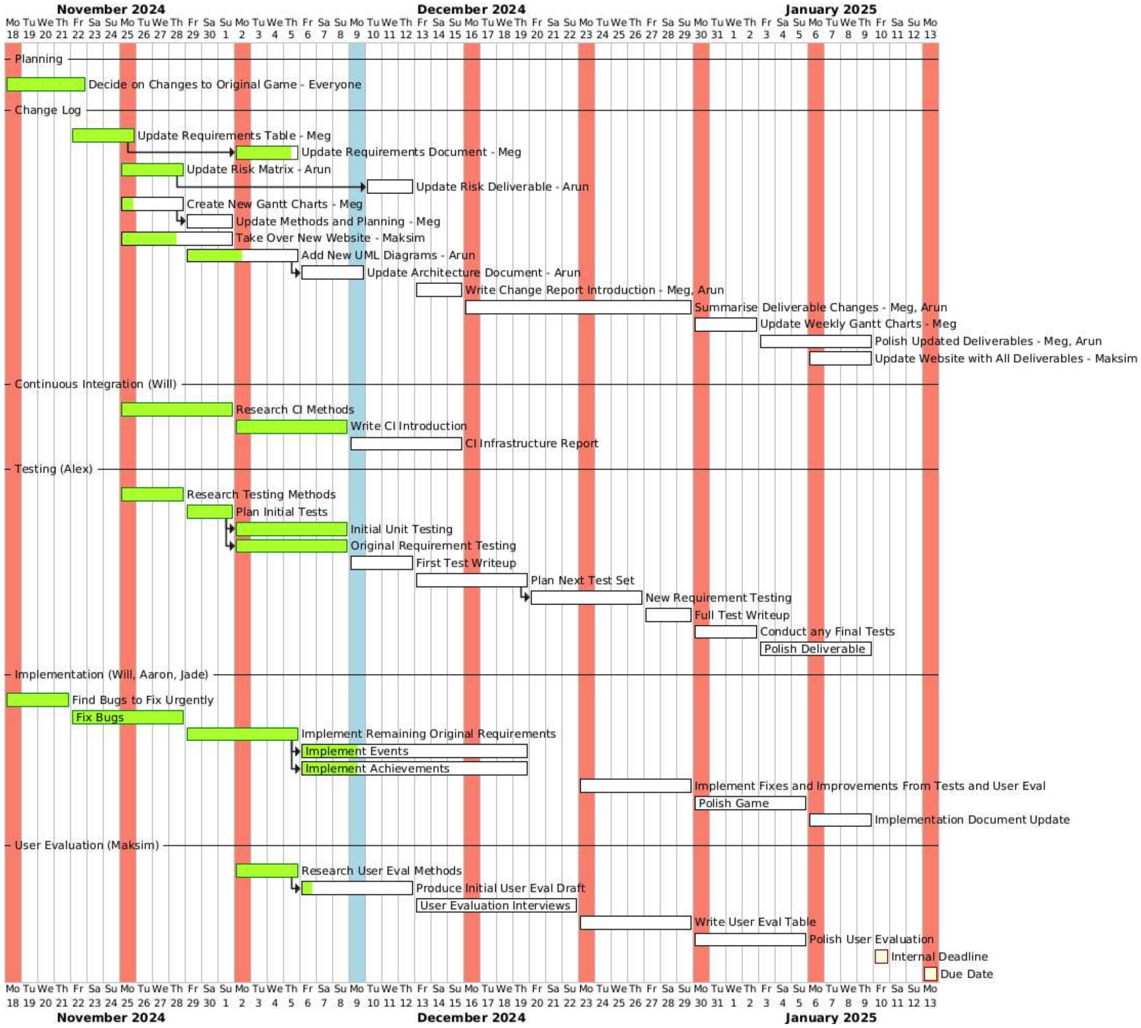
Project Plan



In the first week of this part of the project, we had to examine the game to make sure that we knew what state the deliverables and code were in, as well as decide on what new features we could feasibly add to complete the project brief. This also meant we could begin updating the system requirements whilst the development team fixed any bugs that they found. This was a big collaborative effort as everyone had ideas for which direction we should take the game in.

The Gantt Chart was updated with progress of each part of the project each week so that we could keep track of where everyone was. We talked about the progress we'd made compared to our plan at the beginning of each sprint.

The main change made to the Gantt chart over the course of the project was that the user evaluations began earlier than anticipated, as there was a designated practical on the 13th of December to begin completing the User Evaluation Interviews. We decided that it was alright to only have a few events and achievements implemented to start these, as it still gave users a good idea of what the final game would likely look like. The chart below shows this change as well as our progress at the time that we changed it (beginning of Week 11)



Since everyone was busy over the winter vacation, not much progress was made which meant the progress bars didn’t increase much during this time and the project was delayed slightly. This didn’t change the plan as we had accounted for people being busy over the festive season, so we overestimated some of the times that certain tasks would take.

Following one of our risk assessment protocols (R2), we also ended up changing our internal deadline to Saturday the 11th of January because of other university projects that the team had due around the same time as our project that we wanted to work on through the week instead. We decided to have a big final meeting on the 11th so that we could review and complete any missing aspects of the project before submitting our finalised work on Sunday the 12th.