Method Selection & Planning

Cohort 3 Group 6 - Carbon Goose

Members: Bailey Horsley, Owen Jones, Rory Ingram, Ken Jacob, Abishek Kingslin Raj, Louis Polwarth, Adam Johnson

We are going to be using IBM's Rational Unified Process (RUP) for our engineering method, as this method is plan-driven, and used for large projects in a stable environment. Starting with storytelling, we will outline the requirements needed for the completed project to create a plan of our Use Cases. From here, we will convert these Use Cases into a more Object-Oriented design, with Class Diagrams and modelling using UML. Model-driven development is perfect for this project, as we have a fixed product brief to follow. Therefore we are able to expand on the product brief and client meetings, and refine accurately into models, without any changes being made to the requirements.

In order to create models accurately and efficiently, we will be using PlantUML to create various UML diagrams using text-based Unified Modelling Language. This fits with our method selection well, as it enables us to quickly edit and improve our diagrams without worrying about re-drawing them manually for each iteration.

Once these models have been completed, we will be using LibGDX as a game development tool for the Java project. We considered multiple other game engines, namely JMonkeyEngine (more geared towards 3D games), FXGL (an out-of-the-box but more basic engine), and Litiengine (limited to tile maps, runs slower), which seemed to be the primary contenders. However, given that we are unfamiliar with Java game development, and there are more resources available for LibGDX as it is the more popular library, we have decided to move forward with it.

In terms of collaboration, we are using three different software tools. For communication, we are using Discord, which allows us to group our correspondence to different "channels", which keeps topics related. For collaborative working we are using a repository on github, meaning that we can all collaborate to develop the game using git. This has many advantages such as being able to track who has made specific changes and being able to create versions to track the overall progress of the project as it's being developed. This also means that the project can be worked on at the same time independently and merged together with ease at completion, allowing for the work to be split more manageably between different group members. To keep track of which tasks need to be done, we have created a Trello board, which contains cards for each task in the work breakdown. These cards can be moved between the lists "To-Do", "Doing", and "Done". Trello fits very well with our method selection, as specific tasks can be managed independently by group members.

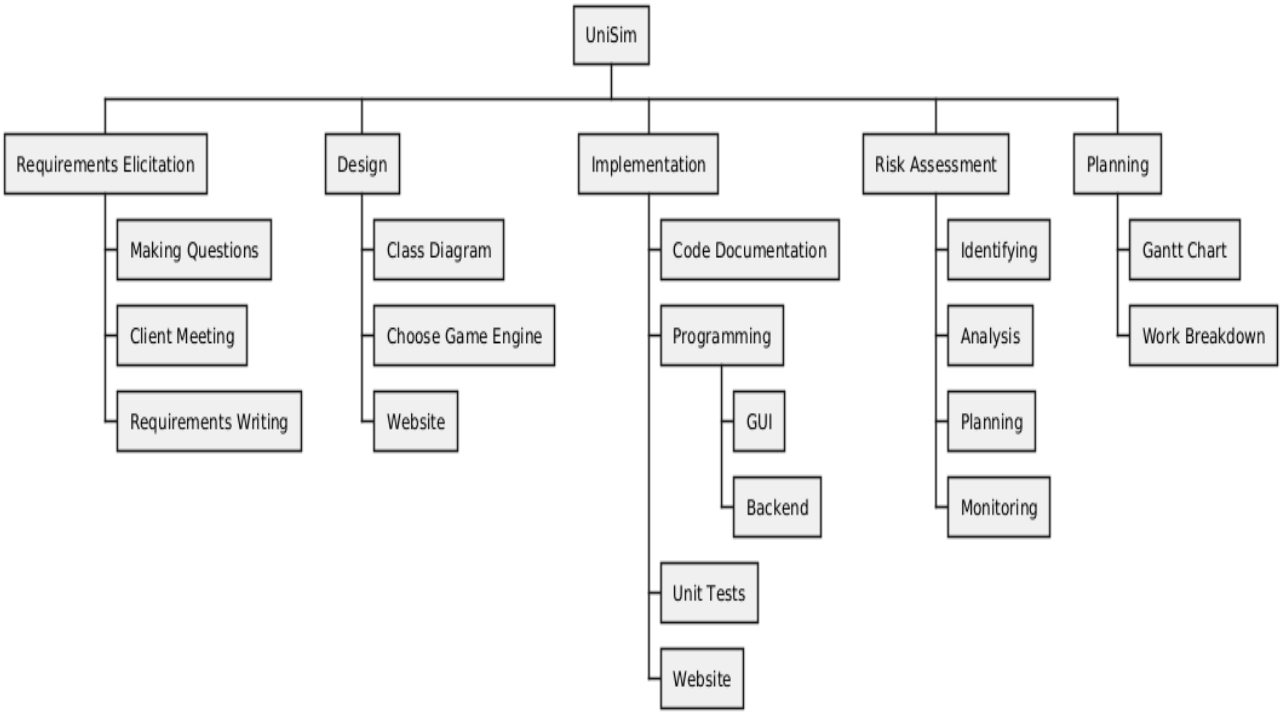
To effectively manage our project, we established a structured approach to team organisation, focusing on at least one customer meeting, regular group meetings among members, clear communication, and use of collaborative tools. We held two in-person group meetings per week, during which we researched and worked on project-related tasks, and assigned tasks to each team member. We kept a meeting log that recorded the previous week's meeting agenda to ensure that the work was completed and the next week's meeting objectives, which we reviewed at the start of each meeting.

Our primary communication tool was discord, we set up dedicated channels for different aspects of our project management, including notes and resources sharing, meeting room bookings, and general discussion channel. Additionally, we conducted any emergency group meetings if required through discord. This setup was particularly beneficial as it allowed us to resolve issues quickly and provided a central hub for information and resources that each team member could access at any time.

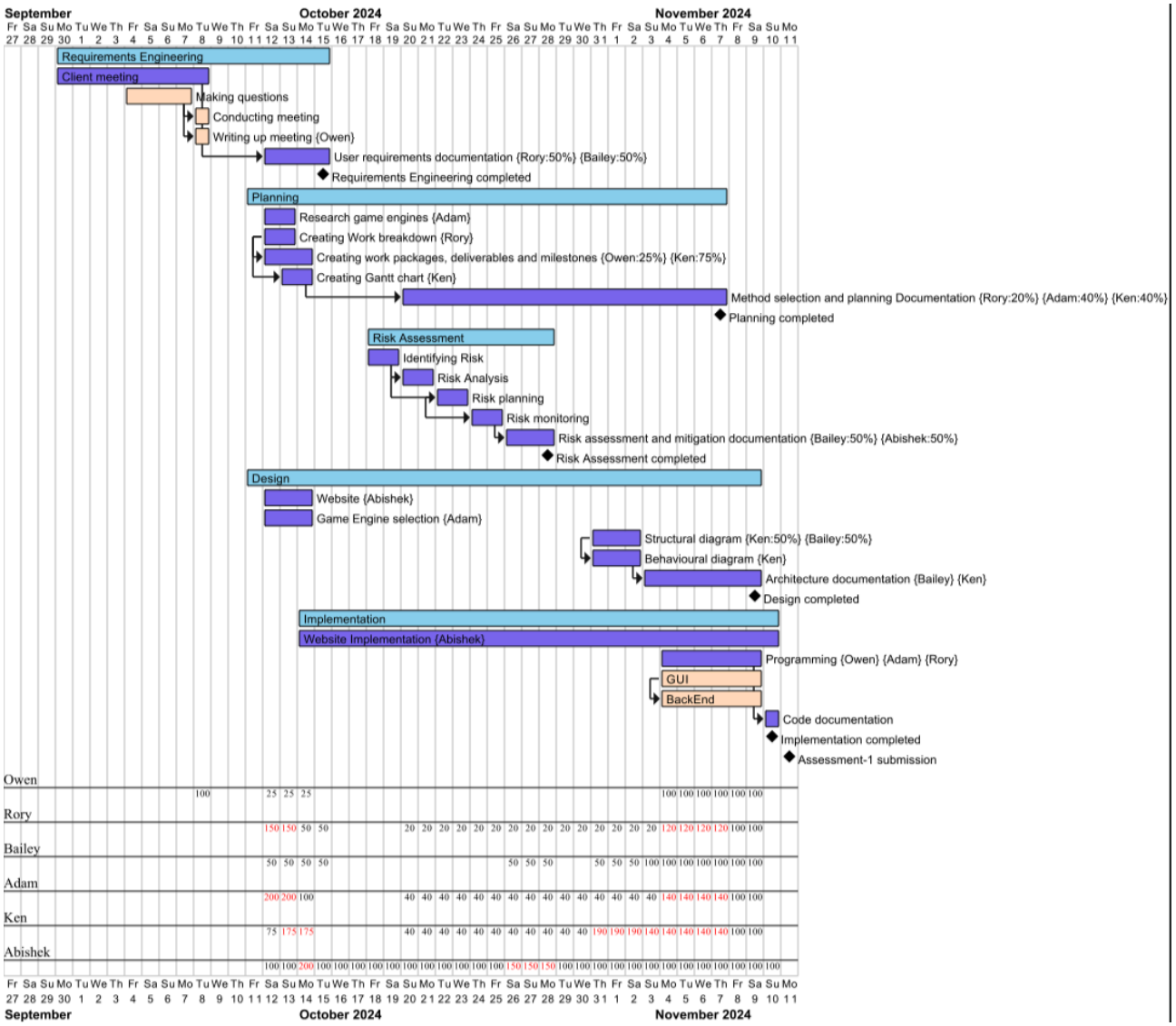
We used GitHub for version control and collaboration on coding and implementation of the game. GitHub's version control features were especially valuable for tracking changes and preventing conflicts, making it easier for us to integrate code from multiple members without issues.

This approach was well-suited for our project and team. The in-person meetings encouraged active participation, facilitating team discussion and research on the requirements and working of our game, while discord provided flexibility for remote communication if necessary. Using GitHub allowed us to collaborate effectively on the technical aspects of the project, enabling all team members to contribute if required regardless of their specific coding experience

Work Breakdown:



Gantt Chart:



Project Plan Evolution:

Week 3: Gantt chart creation and plan outline

We created an initial Gantt chart based on our work breakdown structure created on week 2, which can be seen in "week 2" of the "Project Plan Evolution" part of our website. After meeting with the customer and finalising requirements, we outlined our work packages, tasks, subtasks, and milestones. By the end of week 3, we had completed researching game engines, creating formal work breakdown structures using PlantUML and developed the first version of our Gantt chart using PlantUML.

Week 4: Website setup and Requirements engineering completion

By the end of week 4, we had completed User Requirements documentation, selected a game engine based on prior research, set up our website, worked on rough sketches of structural and behavioural diagrams during our meeting sessions. Additionally, we started working on the risk assessment part of the project.

Week 5: Adjustments for delays and Risk assessment completion

On week 5 we started working on our method selection and planning document, due to some delays in this task, we extended its deadline on the Gantt chart and also rescheduled the formal structural and behavioural diagrams. We completed the risk assessment and finished writing our risk assessment and mitigation documentation. We also pushed back the programming task's start date to learn more about our game engine since we were not familiar with the game engine and to ensure we had adequate planning completed before beginning to develop our game.

Consolidation week: Further adjustments due to delays and completion of Architecture

During our consolidation week we started working on the structural and behavioural diagrams as planned and were able to complete it a few days ahead of schedule, this allowed us to start working on the architecture documentation earlier than expected and was able to complete it on time. Further, we postponed the programming task by a few more as we were trying to get more familiar with the game engine.

Week 6: Finish implementing the game and all deliverables

Week 6 we started programming and finishing up our deliverables. By the end of the week we completed our method selection and planning documentation, our architecture document and the implementation of the game.