**Assessment:** 1

**Deliverable:** Method selection and planning

**Team Name:** Team 8

**Team members:** Charlie Hayes, Matilda Garcia, Joshua Stafford, David Kayode, Ionut Manasia, Matthew Tomlinson

# Method Selection and Tools (a)

As a team, we considered numerous software engineering methods which we could use as a fundamental basis for building our project. We came to a decision to use an agile methodology approach as opposed to utilising plan-driven methods, and have specifically chosen the Scrum framework to work with [1]. Before reaching a decision on which methods we would be using we considered the different aspects of each method, the benefits and drawbacks of each, how successful they would be in terms of the final project being built, and how they would help our team work effectively and collaboratively.

The main alternative methodology we considered was the plan-driven approach, which had the main focus of producing extensive documentation for every part of project planning and building. In terms of how this would support our project progress, this set of methods would have the benefit that they are highly effective when a lot of communication is needed with the customer and within the team. With lots of documentation being produced it is easy to see how progress is being made over time, and feedback can be provided to the customer by sending over documentation for them to view the progress of the project. Under the circumstances at the moment where communication with the customer and the team is online and through email/conference calls, having documentation to prove progress is being made and that can be shared between everyone in the team, including the customer, is a large benefit to ensure the final product is a success and deadlines are met. However, with plan-driven software development, to produce the necessary documents it would be incredibly time consuming, and for the given project with a deadline in a number of weeks, it would not be manageable to have everything documented in time, as well as working code build afterwards. In addition, plan-driven methods would require all planning, requirements and risk assessments etc. to be strictly set out at the beginning of project development, which would not be useful in terms of long term maintenance. If at any point requirements need to be changed or introduced, then the specification, plan, risk assessment, etc. would all need to be readjusted which would take time to start over. Since we are faced with new requirements being introduced by the customer at a later date, we came to the decision that this method would involve too much modification of the original plan and specification and would therefore not allow us to effectively produce a complete project within the deadline.

We chose to continue with an agile method approach, using the Scrum framework to structure our software development project. We decided this was the best methodology for us to use going forward with planning and developing our product as we have a short time frame to produce a working version of the project as well as meeting all of the customer's requirements, whether introduced at the beginning of development or during the process. Agile methods are all about coping with change, and since we are inevitably going to have to cope with new requirements being introduced, which would possibly change the structure of our project, this is something we need to take into consideration. These methods involve employing short iteration cycles, where software is continuously delivered to the customer. The benefits of this are that this provides rapid feedback for project management, so this allows all members of the team to see which requirements have been met and what changes need to be made to meet more of them in the next iteration. Also, this is of value to the customer as they are able to see the product at different stages of development, and they

are included as part of the team, so they are able to request any change in requirements or specification which can be considered in the next iteration. Scrum is the framework we chose to implement agile methods into our software development, and this includes dividing the development into sprints of 1-2 weeks. At the beginning of each sprint, the team will meet to discuss the plan for the next portion of development. There is a daily scrum where each person reports what they have done, what they are going to do, and the problems they anticipate will come along when they are completing their tasks. This method is highly beneficial in terms of teamwork as it involves each member of the team working together, discussing their plans, and sharing project progress and documentation so that everyone is kept up to date. Since we are working remotely, keeping each member of the team up to date with project progress and making sure each person is working on their tasks for the week is crucial in delivering successful implementations of the project throughout the development cycle.

In terms of tools our team has chosen to support building our project, we have chosen a few different tools for different parts of the project. We have created a Github repository for the team, which has in-built tools for version control which is important to the success of the project, as the code being worked on needs to be safely backed up and kept up to date so that there are no errors that arise from multiple people working on alternate parts of the program. Github also allows us to track progress of the project and our documentation, and we have used the Kanban boards tool to create to-do lists of tasks for each week, and assign specific tasks to members of the team to complete. This allows us to monitor project and documentation progress as we can see which tasks are still left to do, which are in process, which have been completed, and which have been assigned to us. We also have used a team google drive, with documents for separate sections of the project brief, which enables us to work collaboratively on documentation and have it kept up to date. We have also been using zoom and discord for team communication and conference calls, to discuss progress and set up meetings.

# Team Organisation (b)

Organisation is an important part of the 8-Team as it aids communication between the team members and enables us to set specific goals and track our progress effectively. Organisation also makes our team capable of adapting to changes.

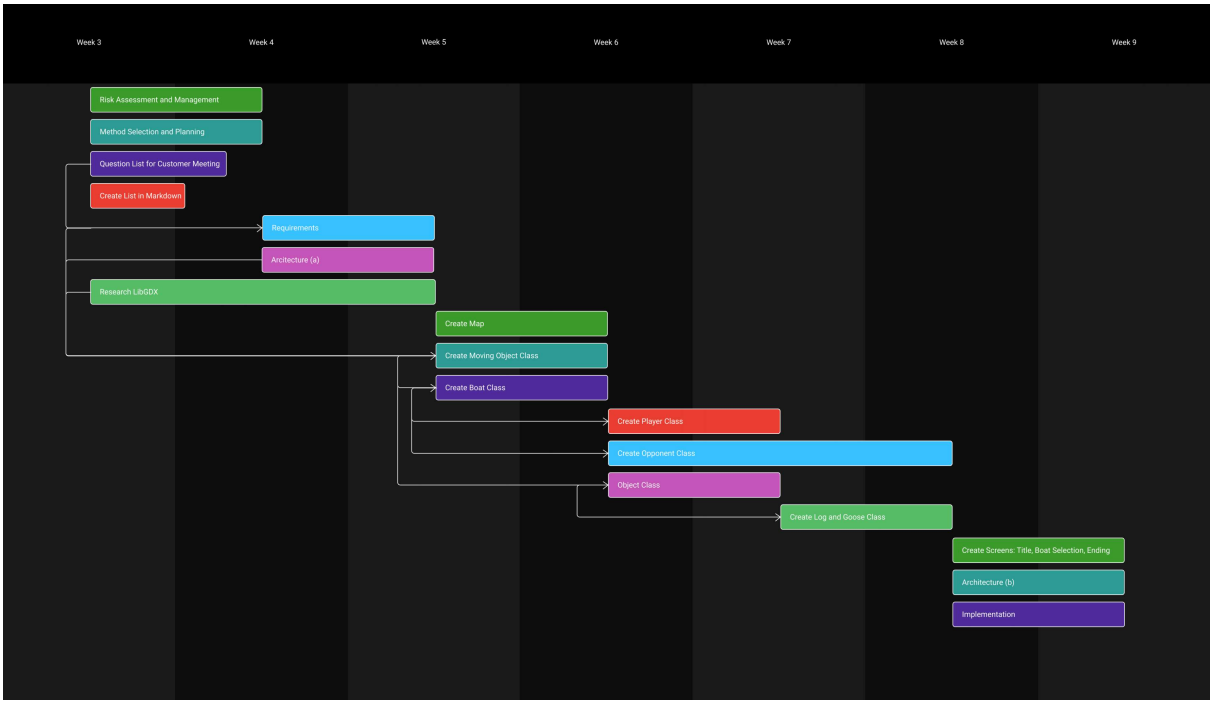
Firstly, the team sets weekly meetings along with general and specific tasks to be done before the next meeting. The frequency of our meetings vary as sometimes we might need to have an impromptu meeting or organise a pre-team customer meeting as well as a post meeting. The majority of our calls are done via a discord call or zoom call if it's scheduled on our timetables.

Discord is one of our effective modes of communication. With the discord channel features we've created different channels for different aspects of our project. For example the tutorial channel is used to post useful tutorials and the general channel is for general updates and discussions. Using the discord platform is an appropriate way of organising our communications as opposed to a general group chat. This is due to the fact that accessing relevant information is easy and time efficient using discord compared to scrolling through a normal group chat in order to find specific information. The 'to-do' channel also helps set general weekly tasks.

In order to set specific tasks, we make use of the GitHub project feature . This feature enables us to divide our deliverables into various set tasks, assign those tasks to specific team members and also mark them as done or in progress. This approach is suitable for our team and project because it reduces the possibility of any tasks being left out or the possibility of unnecessary duplication of tasks.

Additionally, Google Docs and Drive also contribute to our organisation as a team. This is because Google Docs enables our team to collaborate in real time from any remote location and saves time on sending edited emails to one another. The shared folders allows us to organise our documentation as well as track the progress of each team member's tasks.

# Systematic Plan (C)




In our first week of project development we produced our initial plan of progress (See fig. Week3 gantt chart on team website). In the initial planning stages we decided the best way to begin the process would be to initially focus on early documentation and planning. We devised a gantt chart which listed the key tasks to begin working on in this first week of development (w.c 15/10), with Method selection and planning and Risk Assessment being the deliverables of focus. Other key tasks included creating a list of questions to ask in our customer meeting arranged for week 4, as well as all team members being assigned the task of researching our chosen game library LibGDX. The deadline for method selection and planning and risk assessment to be completed was the following week, and our research task was set to be ongoing over a couple of weeks to ensure we had a start point for implementation and programming.

Each week we revisited our plan and updated our gantt chart with new tasks to start, that were to be completed in following weeks. Following our customer meeting we were able to work on the Requirements and Architecture documentation with the aim to have a plan of the main components of the game created by the following week (deadline being Thursday 29/10). The libGDX research task was also set to continue with no current deadline.

In our third development week we revised our plan again, switching our initial focus from documentation to working on code implementation. Key tasks of the week were creating the base map, the moving object class, and boat class, using the information from the completed section (a) of Architecture. The deadline for these key tasks was the following Thursday.

In the fourth week (see Gantt chart Week6) we introduced further code related key tasks that needed to be implemented by the next Thursday meeting, which extended from the work

completed in the previous week. These key tasks were creating the player and opponent classes from Boat, and also creating the class for MovingObject.

Our plan was again revisited on schedule weekly, and for this next week, with the focus still being on creating the code for the game, the key task was creating the log and goose classes as the obstacles of the game. This task was given a deadline of the following week, and we also had to extend the deadline of creating the opponent class so that it could be completed fully by the following week.

In our final week of development we had to revise our plan as we still had to finish some parts of code implementation in order to meet the set requirements, however there remained some sections of documentation that we needed to focus on. The key tasks were creating the game screens in terms of programming, and outstanding documentation tasks were section B of architecture, and the implementation section. At this point the focus was also to ensure the documentation was completed fully, and all formatted correctly before submission.

Revision of our plan on a weekly basis allowed us to keep up to date with which tasks had been completed, and which still needed to be done and given a suitable time portion to complete them. This way we could ensure that all tasks met their deadlines, and were given an extension if necessary.

# References

[1] Atlassian, *Scrum - what it is, how it works, and why it's awesome*. [Online]. Available: https://www.atlassian.com/agile/scrum