Module	Engineering1 (Eng1) - COM00019
Assessment Title	Assessment 2, Cohort 2
Team	Dragonite (Team 21)
Members	Omar Omar, Rhianna Edwards, Okan Deniz, Craig Smith, Omar Galvao Da Silva, Joel Wallis
Deliverable	Methods and Plans

Software Engineering Methods

In order to meet the constraints of the stakeholder's outlined requirements, a methodology that would assist this development in a beneficial way was needed. When deliberating on which methodology to use, the following points were considered: time constraint of the project and its size, the requirements that must be fulfilled and the need for flexibility. This narrowed our decision to three types of methods: Plan-driven, Soft Systems and Agile methods.

Plan-driven methods provide a structured way to create software by predicting and anticipating all future features that may be implemented. Planning of the project can be promoted by providing comprehensive explanations of tasks, workflows and responsibilities [1]. There is an emphasis on continued review and risk management analysis. This detailed level of planning requires the project to be developed in a stable environment and is mainly used in projects that have a long working period [4]. Plan-driven methods that were researched include Capability Maturity Model and Team Software Process (TSP). The focus of TSP is to create an environment that supports disciplined work and maintaining a self-directed team, that would help to establish an effective management and structure to the project. However, with TSP requiring the need for experience of managerial skills and Plan Driven Methodology not catering for the project's need for flexibility, the use of Plan-Driven methods was not chosen [7].

Soft System methods look to break down complex problems in order to truly understand the problem outlined by the stakeholder, and use this to develop multiple conceptual models that can be compared to the original real world problem [2]. The time taken to understand this problem and discuss potential solutions is a lengthy procedure that takes weeks . The model has limitations in comparing the conceptual model to the real world problem and it does not provide a framework for how the design should look, instead focusing on the search process for the best solution [8]. The time-constraint and potential need to integrate other Models meant this plan was not a rewarding model to use for this project [6].

Agile methodology focuses on the "concept of ongoing waves or sprints of project planning and execution" enabling the project team to adapt the design, scope and execution of the project deliverable [5]. Code is distributed in minor releases with short periods, where the project team works together with the Customer in close cooperation. The approach is easy to understand, and its core principles were aligned to the scope of the project and the stakeholders requirements. The focus on frequent software delivery allows for the development team to produce the product in a shorter time frame, when compared to other methodologies, and its allowability to iterate and innovate on functionality of the system would help to maintain a higher motivation within each individual of the team. The scope of the project and its delivery fell in-line with this methodology and was decided on.

With the **Agile methodology** chosen, **Scrum** was selected as the framework to follow as it reflected the continued benefits of the Agile methodology, such as the ability to cope with rapid changes, allowing for quick feedback from other team members, and usually low documentation overheads. The Scrum method outlined a flexible but concise managerial structure that would help the team to adapt to expected changing requirements (new customer requirements, time and resources). The structure involved defining a Product Owner, Scrum Master, and Development Team [3]. Weekly sprints are used to keep organisation and even the distribution of workload between the team members, which will be decided during each formal weekly meeting. This structure will help to ensure that there is a focus on delivering the fundamental features of the project [10].

Development and Collaboration Tools

Scrum management: Google Drive will allow us to create our own in house scrum management system using google docs. Members will self assign tasks which will be overviewed by the scrum master to ensure our time is being used most efficiently. At the end of each week, team members will update the document with the current status of their tasks, leaving markers for completed work or requesting feedback. Google drive is the ideal tool for our scenario as it ensures that all team members are up to date with the project and will allow for flexible reallocation of tasks where necessary. A viable alternative tool is Trello, however for our application, creating our own document will provide a more streamlined approach to keeping track of the progress of the project as this will be stored in the same location as the other projects deliverables for easy accessibility.

Documentation tool and Deliverable Repository: Google Drive acts as the central location to store all the documents related to the project. It allows for simultaneous work on single or multiple documents. With support for different document types such as Docs, Sheets, Slides and third-party tools, such as UML diagrams, it stood out over alternatives. These alternatives included Microsoft Word which was not chosen as it only allowed for asynchronous collaboration and the transferring of this document type required additional services. Dropbox and OneDrive (with its integration of further Office tools) were noted but their feature sets are not as vast as those offered by Google Drive.

Communication tool: Zoom and **Discord** are used to communicate with other team members through audio calls and text channels. Zoom allows for scheduled sessions that can be set up weekly to hold formal meetings between the client and team members to clarify any queries or issues. Discord uses a system that allows multiple text channels to be set up referring to different areas of the project. This will be useful for organisation and will provide an easy way for the team to send across ideas through text or images. Discord also has features that allow users to join voice calls which can be used for hosting flexible meetings during sprints.

Version Control tool: GitHub will be used for the version control tool for the project. It works as an online repository and allows for easy collaboration of code development. The branching feature will prevent unnecessary confusion and damage to the project, when team members are coding at the same time. As GitHub is a widely used VC tool and offered large amounts of integration with other software, it was chosen.

UML diagram tool: Lucidchart will be used to create the architecture diagrams for the game. It allows for early discussions of basic entities to be conceptualised and the relationships between them, before the implementation process is started. It allows for flexibility when adding or removing functionality in later iterations of the diagram. The tool is easy to learn, free and allows us to work on the UML diagram synchronously with integration with Google Docs.

Asset Creation Tools: Photoshop and **Piskel** will be used to create custom assets for the game. Piskel is an online sprite creator that will allow us to create these sprites on a pixelated grid, ensuring a consistent design across assets. Photoshop will be used to edit existing assets that may need modification with size or cropping.

Testing tools: JUnit and Mockito are frameworking used for testing the functionality of an application. The frameworks are used for writing and running tests, which can be automatically run to check the output against the expected output. These easy to use frameworks allows for tests to be run quickly and easily, which is ideal for our project. Mockito allows you to create and configure mock objects, which in turn allows you to test specific functionality in isolation. JMockit was an alternative we could have used however it involved a steeper learning curve than mockito which would not be ideal for the limited project timeframe.

Continuous integration tool: Travis CI is a tool used for building code to see if the current github commit will build successfully. It has automatic github integration which makes it perfect for our project, and also supports many different languages, but we will only be using it for java as this is a customer requirement. It is also very quick to set up, making it an excellent choice for our project given the limited time frame. An alternative we considered is Jenkis. However, Jenkins is highly configurable and would take too long to set up for our project.

Team Organisation

Organisation - Our google drive scrum document is used for the management of tasks. An organised table allows the project tasks to be stored along with the name of the assigned member, the date, and the current status of that task. These tasks will be assigned and updated during our weekly scheduled sprint planning sessions. Each sprint will last one week with set goals keeping team members on track, with the exception of some larger tasks being spread over numerous week sprints. The sprint planning sessions will involve assigning tasks for the following week's sprint, along with discussing potential questions that may need to be asked in customer meetings. To monitor progress of a sprint, each task will have a progress column with the parameters: “Assigned”, “In Progress”, “Completed”, “Needs Reviewing”. This will allow all members in the team to maintain an understanding of each individual's progress with their allocated tasks.

To further this, flexible date meetings will be held during the week to ensure any issues with tasks can be resolved before the end of the sprint. These sessions are very flexible and will be determined using Discords voting system to make sure all team members are available. Another Google Docs document should be used to record meeting notes from these sessions and record attendance. This will come in useful for those who cannot attend and allows for reflection on issues or points discussed to keep everyone up to speed on the current project situation.

Roles and Tasks Allocation - To ensure that all members are playing to their strengths, tasks will be discussed among members and will be assigned collaboratively. At least one other person will be assigned to each task to help mitigate any potential backlogs that may build up from incomplete tasks. This will help meet deadlines, and allow the project to run smoothly. In addition to this, regular reviews of completed tasks will be done to ensure that all points have been met to the correct requirement.

Roles assigned will be a Product Owner, who will ensure that the most value is delivered to the business by the development team, and a Scrum Master, that coaches every team member and ensures they are adhering to the Scrum method [7]. The Development team will not be specified as it will allow the team to respond quicker to volatile situations throughout the project.

Development Architecture - Using the principles outlined in research of the Scrum method, fixed meetings are held on Thursday and Saturday mornings. These times are decided on to ensure tasks will be completed by these deadlines and to maintain a strict routine. Daily stand-up meetings will not be possible, so the sprint goals will be assigned during these scheduled sessions with discord text channels available throughout the week to allow the team to communicate with one another to help out with issues and/or provide new ideas. Flexible meetings throughout the week may be set up using the voice call feature in Discord which will allow a subset of members to work through task problems in sessions to help reach the sprint goals before the deadline. If work cannot be completed/solved in the given timeframe, other team members can be assigned to the task or members may reassign tasks that they are working on to maintain the constant delivery principle of Scrum.

Project Plan

The priority of each task was decided based on:

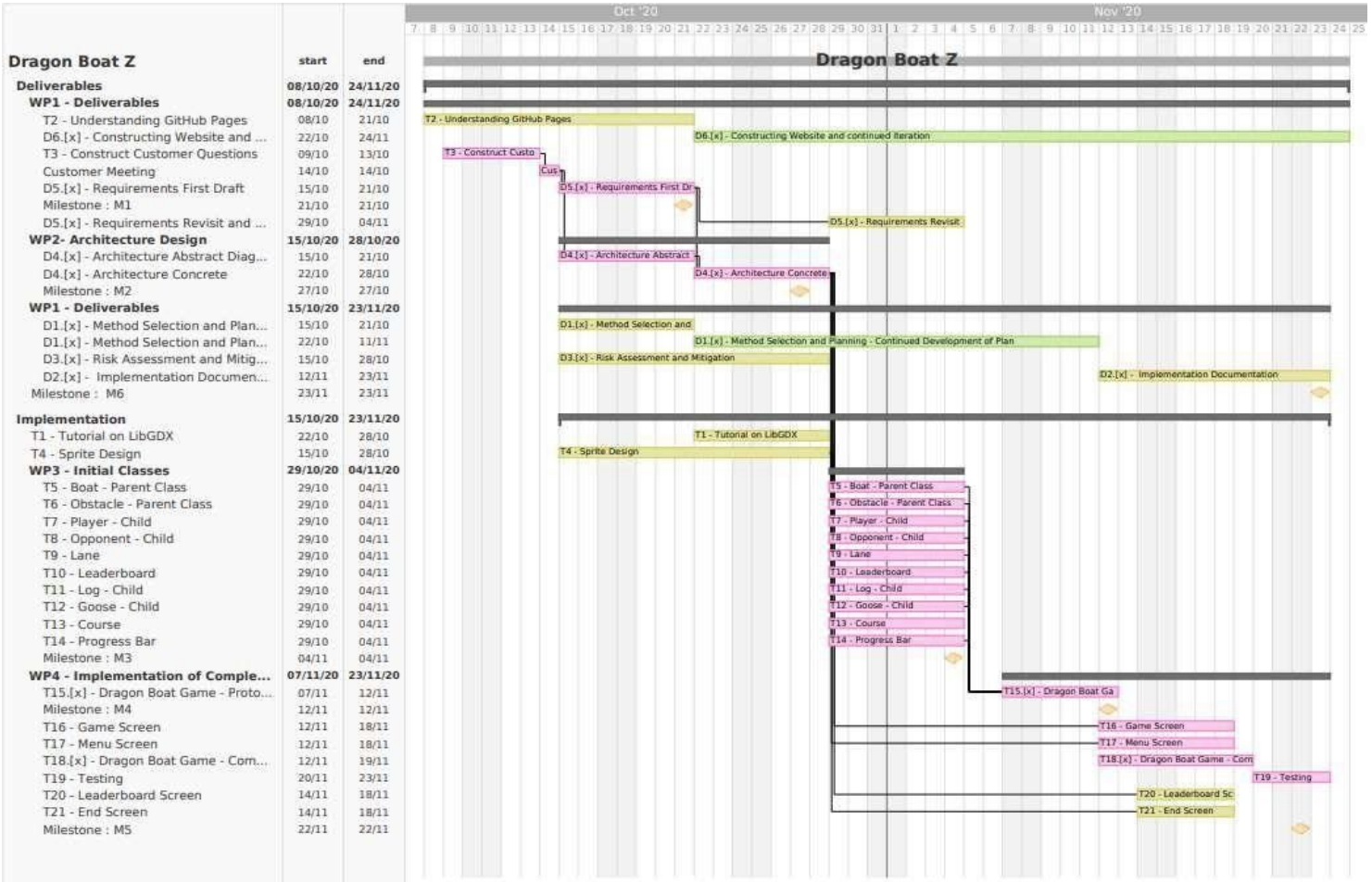
- The time needed to complete the task.
- The number of marks it was worth, if related to **documentation.**
- The impact of the task not being completed and the priority value assigned to it in the requirements deliverable, if related to **coding.**

Using these priorities, a Gantt chart was constructed to display the various parts of the project and highlight any anticipated dependencies between tasks. An online tool [9] was used as this allowed for synchronous collaboration between team members and allowed for adjustments to be made in response to volatile requirements. This allows for a clear understanding of the implication of any delays in the project.

Each task of the project was assigned a colour, representing its priority. Green for low priority, Yellow for medium priority, and Red for tasks critical to the completion of the project. These critical tasks were used to define a critical path (the longest sequence of dependent tasks that must be completed to produce a game that is defined by the SSON). The dependencies (tasks that require prior tasks to be completed) were demonstrated using links between them. To compensate for a potential backlog between sprints building up, the Gantt Chart allows for a period of three days at the end of the project to compensate for delays in task completion.

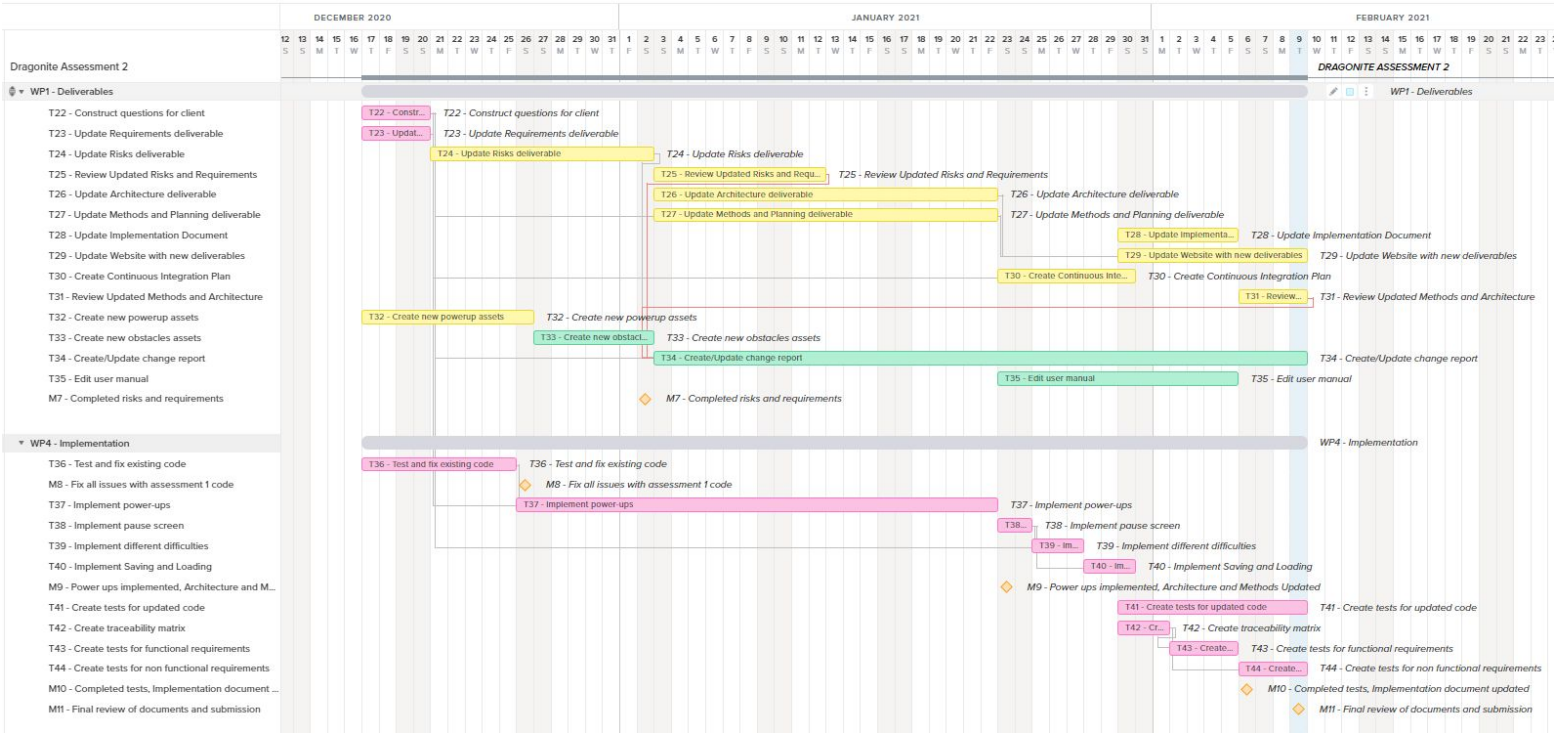
The Gantt Chart is split into separate sections for Deliverables and Implementation, to allow for readability. During our weekly Sprint Planning sessions, the Gantt Chart will be used as an indicator of tasks that should be completed and prioritized during that time frame. When designing the chart, Work Package IDs, Task IDs, Deliverable IDs and Milestone IDs are used to maintain a clear structure to the project and to stick to the limited time frame given

Gantt Chart for Assessment 1:



Dragon Boat Z	start	end
Deliverables	**08/10/20**	**24/11/20**
WP1 - Deliverables	**08/10/20**	**24/11/20**
T2 - Understanding GitHub Pages	08/10	21/10
D6.[x] - Constructing Website and ...	22/10	24/11
T3 - Construct Customer Questions	09/10	13/10
Customer Meeting	14/10	14/10
D5.[x] - Requirements First Draft	15/10	21/10
Milestone : M1	21/10	21/10
D5.[x] - Requirements Revisit and ...	29/10	04/11
WP2- Architecture Design	**15/10/20**	**28/10/20**
D4.[x] - Architecture Abstract Diag...	15/10	21/10
D4.[x] - Architecture Concrete	22/10	28/10
Milestone : M2	27/10	27/10
WP1 - Deliverables	**15/10/20**	**23/11/20**
D1.[x] - Method Selection and Plan...	15/10	21/10
D1.[x] - Method Selection and Plan...	22/10	11/11
D3.[x] - Risk Assessment and Mitig...	15/10	28/10
D2.[x] - Implementation Documen...	12/11	23/11
Milestone : M6	23/11	23/11
Implementation	**15/10/20**	**23/11/20**
T1 - Tutorial on LibGDX	22/10	28/10
T4 - Sprite Design	15/10	28/10
WP3 - Initial Classes	**29/10/20**	**04/11/20**
T5 - Boat - Parent Class	29/10	04/11
T6 - Obstacle - Parent Class	29/10	04/11
T7 - Player - Child	29/10	04/11
T8 - Opponent - Child	29/10	04/11
T9 - Lane	29/10	04/11
T10 - Leaderboard	29/10	04/11
T11 - Log - Child	29/10	04/11
T12 - Goose - Child	29/10	04/11
T13 - Course	29/10	04/11
T14 - Progress Bar	29/10	04/11
Milestone : M3	04/11	04/11
WP4 - Implementation of Comple...	**07/11/20**	**23/11/20**
T15.[x] - Dragon Boat Game - Proto...	07/11	12/11
Milestone : M4	12/11	12/11
T16 - Game Screen	12/11	18/11
T17 - Menu Screen	12/11	18/11
T18.[x] - Dragon Boat Game - Com...	12/11	19/11
T19 - Testing	20/11	23/11
T20 - Leaderboard Screen	14/11	18/11
T21 - End Screen	14/11	18/11
Milestone : M5	22/11	22/11

Gantt Chart for Assessment 2:



Tasks are assigned a unique identifier, a description and a reference to its Work Package.

Work Packages are used to define combinations of Project's tasks that are related to each other.

Milestones are used to define specific points in the project where a defined goal has been reached (based on completion of tasks belonging to a Work Package).

- M1 - Customer Meeting and Requirements Completion — Due Date: 23/10/2020
- M2 - Architecture Abstract and Concrete Completion — Due Date: 27/10/2020
- M3 - Initial classes Completion — Due Date: 04/11/2020
- M4 - First Prototype iterated — Due Date: 11/11/2020
- M5 - Completion of Testing and Game — Due Date: 22/11/2020
- M6 - Final Deliverables Completed — Due Date: 23/11/2020
- M7 - Have all deliverables updated for assessment 2 — Due Date: 02/01/2021
- M8 - Fix issues with original assessment 1 code — Due Date: 26/12/2020
- M9 - Power ups implemented, Architecture and Methods Update — Due Date: 23/01/2021
- M10 - Completed tests, Implementation document updated — Due Date: 06/02/2021
- M11 - Final review of documents and submission — Due Date: 09/02/2021

Project Evolution

By developing a concise and well laid out project plan in the early stages of the project, only a few changes had to be made over the course of the project. Weekly sprints did not occur around the christmas week, as well as the exam period in january due to lack of time available between all group members. To keep on track with the project timeline, we extended some sprints to longer periods of time to allow for more flexibility between the team members. Flexible, unscheduled meetings still took place however, between subsets of the team when additional support was needed.

References

[1] H. Svensson, *Developing support for agile and plan-driven methods,* PhD dissertation, KTH, Stockholm, 2005, p.26. [Accessed: 23/10/2020]

[2] S. Burge, "An Overview of the Soft Systems Methodology", *System Thinking: Approaches and Methodologies,* 2015, pp.1-14. [Accessed: 23/10/2020]

[3] *What is Scrum?,* Scrum.org. [Online]. Available at: https://www.scrum.org/resources/what-is-scrum [Accessed: 15/10/2020].

[4] *[No title].* [Online] . Available at: http://www.se.rit.edu/~se456/slides/PlanDrivenMethodologies.pdf. [Accessed: 18/10/2020]

[5] *[No title].* [Online]. Available at: https://www.wrike.com/project-management-guide/agile-methodology-basics/. [Accessed: 05 Nov. 2020].

[6] *Soft Systems Methodology.* [Online] . Available at: https://www.umsl.edu/~sauterv/analysis/F2015/Soft%20Systems%20Methodology .html.htm. [Accessed: 02 Nov. 2019] .

[7] *Scrum- what it is, how it works, and why it's awesome,* Atlassian.com [Online]. Available at: https://www.atlassian.com/agile/scrum [Accessed: 26/10/2020]

[8] (2018, September.11), *Soft Systems Methodology (SSM) .* [Online]. Available: https://www.toolshero.com/problem-solving/soft-systems-methodology-ssm/. [Accessed: 24 Oct. 2020].

[9] *Online Gantt Chart Software & Project Planning Tool.* [Online] . Available: https://www.teamgantt.com/ . [Accessed: 15 Oct. 2020].

[10] U. Eriksson, (2015, July.15), *Agile Software Development and Requirements.* [Online]. Available: https://reqtest.com/agile-blog/requirements-in-agile-software-development/ . [Accessed: 1 Nov. 2020] .