Module	Engineering1 (Eng1) - COM00019
Assessment Title	Assessment 2, Cohort 2
Team	Quackthulu (Team 16)
Members	Aaron Price, Alec Coates, Charlie Curedale-Rayner, Eleanor Griffin-Smith
Deliverable	Method Selection and Planning

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 should be chosen, 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 [9]. 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.

For our project, our group used the Scrum[7] agile engineering method. We chose this method because we are a small team of 4 developers meaning that we can all easily communicate together as a team and effectively utilise every member's skills and capabilities. Assignment of task and progress can be discussed during regular team meetings.

Development and Collaboration Tools

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: Facebook messenger and **Zoom** are used to communicate with other team members. Facebook messenger is mainly used for clarification of roles or to ask for a Zoom meeting. Zoom is used so that we can speak to each other as due to Covid-19 we cannot meet up and discuss issues in person. It also facilitates screen sharing so we can more easily collaborate.

Version Control tool: GitHub will be used for the version control tool for the project. It works as an online repository and allows us to collaborate on code development. The feature of branching 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: Draw.io will be used to create the architecture diagrams for the game. It can be used to create flowcharts and UML diagrams. The tool is easy to learn, free and we used it in Assessment 1.

Implementation Tool: Java using the **LibGDX Module** will be used as Java is a customer requirement stated in the Product Brief. LibGDX will be used as the game library, as the documentation and online resources about it are vast (including tutorials). Additionally, there was a consensus within Cohort 1 and 2 over its use, allowing for easy transfer of projects in Assessment 2. Alternatives considered were Slick2d and Joge (Java OpenGL Game Engine).

Team Organisation

Organisation - Being a smaller team and having the experience of assessment 1 we decided to follow a similar way of organisation in assessment 2. Our team holds bi-weekly SCRUM meetings to discuss the developments we have made, provide support for any team members who are struggling with any of their tasks and plan what needs to be done for the next meeting. All our meetings are held using internet video conferencing because the current COVID-19 pandemic limits us from meeting together in person due to health risks.

Roles and Tasks Allocation - Using our previous experience from assessment 1, tasks were assigned depending on individual strengths. Everyone in the team will be assigned tasks throughout the project. Selected team member should be able to complete the task well and in a timely fashion. When assigned tasks over a zoom call we discussed and made sure everyone was happy with their task assignments.

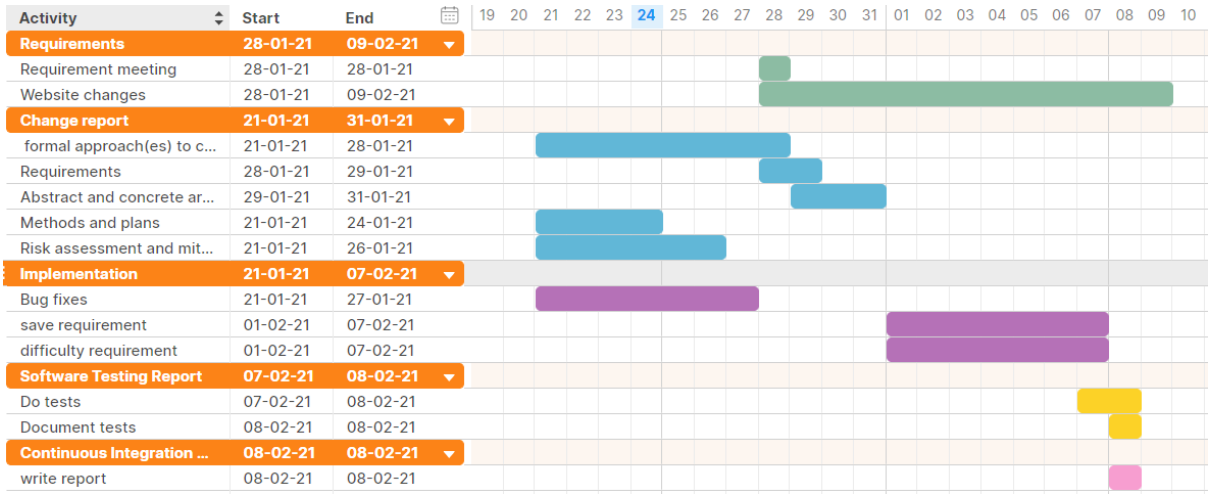
Development Architecture - Using the principles outlined in research of the SCRUM method, meetings are fixed on Sunday and Thursday afternoons each week using Zoom. Fixed times were decided on to further ensure tasks will be completed on time and to maintain a routine. Time constraints of the team mean that daily stand-ups will not be possible. If work cannot be completed/solved in a given timeframe, additional team members can be assigned to the task or members may switch the task that they are working on. This is to maintain the constant delivery principle of SCRUM.

Project Plan

Task priority was decided based on:

- The time taken to compete.
- The number of marks it was worth and the number of people it requires, 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 this, an initial Gantt chart was constructed to display the various parts of the project and highlight any anticipated dependencies between tasks. An online tool [10] 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.



The Gantt Chart was split into separate sections for reflecting the deliverables for Assessment 2 to allow for readability. Progress is assessed during meetings and tasks are reassigned accordingly. Below you can see the task dependencies we used to create our schedule.

Task dependencies:

Requirements Meeting -> Requirements -> Architecture -> Implementation

Implementation -> Software testing Report

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] A. Özerten, (2016, December.26), *Benefits and Challenges of Self-directed Teams in Software Projects*, Commencis. [Online]. Available: https://medium.com/commencis/benefits-and-challenges-of-self-directed-teams-in-software-projects-fe8df2d842e8. [Accessed: 28 Oct. 2020].

[9] (2018, September.11), *Soft Systems Methodology (SSM)*. [Online]. Available: https://www.toolshero.com/problem-solving/soft-systems-methodology-ssm/. [Accessed: 24 Oct. 2020].

[10] *Online Gantt Chart Software*| *Gantt Chart*| *Tom's Planner*. [Online]. Available: https://www.tomsplanner.com/ . [Accessed: 18 Jan. 2021].

[11] 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].