4. Method Selection and Planning

Cohort Number: 1
Group Name: NeveSix (Team 6)
Group Member Names:

- Sam Savery
- Ewan Hutcheson
- Igor Smolinski
- Sanjna Srinivasan
- **Zack Tyler-Kyle**
- Phoebe Russell
- Pranshu Dhungana

Overview:

After much deliberation we had narrowed the field down to two possible methodological approaches: the agile methodology, and the waterfall methodology.

Primarily, the waterfall approach is one we collectively are most familiar with, as this is the methodology employed in almost every individual project- the requirements of the project are established prior to commencement, the approach is planned and designed as opposed to immediately creating versions of the product, this plan is adhered to in implementation and once implemented the product is checked against the requirements to assess validity. This is a logical and structured path from idea to realisation.

However, due to the nature of this product and the two distinct phases already pre-established (Assessments 1 and 2), we ultimately realised that the agile methodology- particularly the scrum variant- best matches the pacing and stages required for the successful development and deliverance of the product. Hence:

A comprehensive initial plan is created, similar to the first stages of the waterfall approach, arising from the requirements gathered and presented in ***2. Requirements*** (above). Consequently creating a single concise statement of such requirements for clarity across all stages of development. This plan is the backbone for all that follows, even after the expected future revisions of the requirements and project status.

From this initial plan follows the beginnings of implementation: object relation maps, function planning and general abstraction of the product. A workable- albeit constrained- build of the product arises from this and is tested and presented/reviewed as of Assessment 1's conclusion. This finale to Assessment 1 provides us the perfect natural marker for the next iteration of the project, as well as allowing us a further chance to refine or add to the requirements, hence satisfying the customer collaboration aspect of the agile approach.

Even outside of the customer-facing elements of agile, the principles of self-organising teams and communication as a priority over extensive and bulky documentation is extremely appealing to our seven-strong team- seeming the best way to ensure all of our members see their share of the work and hence their potential in this project realised.

Team Composition:

Being a team of seven affords us the luxury of numbers when it comes to delegating workload and roles, however it was still most logical and most in-keeping with the principles of the agile methodology to assign roles based on the core strengths of each of our team members. The final structure agreed upon divides the team accordingly: Phoebe and Sanjna find their strengths in documentation and planning, and as such have taken on a proportionately larger role in the earlier parts of this assessment- focussing on sections like ***Requirements*** and overall allocating their skills to the written side of things. Igor, Sam and Ewan find themselves more comfortable thinking in terms of implementation, and as such

take on implementation-relevant sections such as creating the object relation diagram for all objects contained in our product and generally handling the ***Architecture*** of this assessment. The final group are a mix of the two prior- reasonably comfortable with implementation as well as project planning and management. Hence we have Pranshu and Zack as free agents working together to take on a mix of project sections; ***Risk Assessment and mitigation, Method selection and planning*** as well as maintaining and managing the record of each team meeting falls under their purview.

While our team composition matches our relative strengths, the implementation of the product itself is a section we have chosen to tackle as a unit, meaning everyone will have made significant contributions to the writing of this assessment's submission as well as the product to deliver at its conclusion.

To summarise, we have broken down the assessment into its constituent parts, divided our team according to proficiency in writing, implementation and project management, and distributed the workload according to these divisions. All of this culminating in a team unit best structured to deliver the highest quality version of each section in this assessment.

Tools:

To aid our agile approach, we reasoned that the most structured and efficient form of communication is paramount, so Discord was selected as our primary channel and WhatsApp as our secondary channel. Between them both we have covered the most comfortable and frequently used methods of communication of each member of the team, with Discord in particular facilitating the partitioning of text/voice channels for specialisation: multiple text channels such as the "#Important-Resources" and "#Meeting-Times" provide easy-to-access information about critical parts of the project, whereas having three distinct voice channels allow for team members to branch off into groups and work on separate sections without the raucous distraction of general group chatter.

Naturally, a project requires documentation and the team requires access to said documents, hence the logical choice of storage location is a team Google Drive where all members have equal access and permissions. This coupled with folders organising sections of project (such as the "Assessment Written Sections" folder, where much of the past few pages reside!) and our team has a convenient and simple method of organising the project as it progresses and grows in size, particularly as the requirements evolve- all in-line with the agile methodology. Truthfully, Google Drive was immediately settled upon for its convenience, however Onedrive was proposed as an alternative as it provides almost all of the same functions as Google Drive, plus a few of our members have familiarity with the software already. The primary distinguisher between the two is simply the value of the extra 10GB of storage given for free through Google Drive, as well as the integration with the Microsoft suite of resources- Google Drive is the obvious choice.

On the implementation and coding side of things, we entertained the idea of using Google Drive ubiquitously, however the vast benefits of the Git-Github duo far outweighed the slight convenience of holding all project elements in one place. Firstly, Git is a version control platform that provides support for merging work from multiple sources: an invaluable tool when navigating a multi-programmer project, especially when up to seven of our team are

working simultaneously on different parts of the same code. Further, considering that this is the first collaborative project for some of our members as well as standard human error, the ability to rollback to a previous version of the project at the press of a button is exceptionally useful, alongside the tracking of commits to ensure the workload is equitably distributed.

Github and GitKraken are effectively our Google Drive for Git repositories, and provides us ease of access to the work conducted through Git, organised chronologically in a handy timeline.

IntelliJ IDE finds itself the coding environment for this project, simply for its Java-oriented nature over other more general IDEs like Visual Studio, it keeps our code consistent (since we all will be using the IDE) with fewer bugs and exceptions due to its helpful tooltips and warnings when code is written/formatted incorrectly- making refactoring and code revision easier, which plays into the Scrum methodology we are following. Plus it plays well with LibGDX:

Our game engine of choice is LibGDX- a Java-based product- almost purely for its easy integration with IntelliJ IDEA and the host of useful classes it provides for staging scenes (Stage class), controlling the camera (OrthographicCamera class) and manipulating actors (Actor class). However it does also provide cross-platform functionality, giving our project a wider reach in the devices it can run on.

And of course, it goes without saying: each of these products and services all have a free student licence available- a godsend for a group with a small budget!

The Plan:

The initial plan was to have all written sections of this assessment being worked on concurrently, as opposed to following the order provided in the assessment brief. Meaning we could have almost all of the deliverables, planning and designing done before we began implementation thereabouts a month before submission. Hence, our handy starting Gantt chart looked like this: (https://neves6.github.io/gantt/week1.jpg).

Of course this led to a few dependencies arising that necessitated certain tasks taking a higher priority than others- such as the requirements of the project needing to be acquired before almost everything else, which is reflected in the above chart by the earlier start of requirements gathering than the rest of the deliverables. As a general rule, the earlier the starting date of the activity, the higher the priority we assigned it, hence:

-Technically, icebreaking and meeting the team is our initial top priority- with all other activities hinging on it. But given that this happened before we even saw the assessment brief collectively, it is not included in the Gantt chart.

-The website essentially being the hub for our entire project was set as top priority and as such to be created immediately in Autumn Week 8 (2nd week from start as shown in the above chart). Its maintenance follows logically as needed throughout the project duration.

-Requirements are necessary to allow us to plan and begin designing the game in the first place, so they are set to be gathered before the rest of the sections are started (including this one!).

-Once requirements are gathered, a table containing them and their attributes can be created as well as the rest of the sections of this assessment commencing creation- now with the information necessary to complete them.

-System architecture, planning of the project, and the risk assessment all have equal priority and begin at the same time (ordered on the Gantt chart as they appear in the assessment brief), keeping all seven of us working on different sections concurrently.

-Implementation and the library/asset explanations follow once the architecture sections are complete and we have a structure to go from. Thus this is set as one of the final things to be done and with the lowest priorities. That is to say, creating the game is very important but there are barriers to starting its creation in the prior written sections.

Plan Evolution:

Naturally, plans are idealised and provide a useful guideline for us to follow, however any number of factors can shape and change these throughout development. The most obvious and clear example of this in our project is the beginning of implementation of the game: we had initially intended to begin implementation only after the prior written sections were sufficiently completed, as indicated in the starting Gantt chart.

However, we found that sufficient progress was made even by week 4 of the project to begin the early stages of project development- IntelliJ IDE setup, LibGDX setup, Github repository created and structured as well as the team being briefed on how to use these new tools. And so implementation was underway even as early as week 4. This is reflected in the final update of the Gantt chart: (https://neves6.github.io/gantt/week14.jpg).

As seen, “Code Implementation” and “Code Documentation” began in Week 4- way ahead of schedule from the initial plan. Equally, this early start on implementation afforded us more time to work on the written sections; in particular the subparts of these that involved long-term updating and documentation on the project. This very section comes under that, with the website containing the Gantt chart after each week of progress.

By the penultimate week to submission, almost the entire code was written with very minor additions left (sound effects and updating the title screen to say “Piazza Panic 1” instead of “Titlescreen”)- ahead of project schedule.

All further aberrations and differences from the original plan are relatively minor, however are clearly illustrated in the week-on-week updates to the Gantt chart seen at (https://neves6.github.io/gantt.html).