2020

Method Selection and Planning

TEAM ZANETTA

HARRY KELLY, TIMOTHY KERR-CHIN, JOE SANDERS, SIYAAM MAHMOOD, EMILY TIERNEY & ALEXANDRE PINNEAU

Part a

Software Engineering Methods

As a team we opted for a plan-driven software engineering method over an agile method as we expected there to be little change in the requirements of the game throughout the software engineering process (i.e. <1% a month) and as we felt this type of method better enabled communication between the team and the client but also internally within a team with little team software engineering experience.

As a team we had two meetings most weeks in which we discussed decisions which we all needed to agree upon (i.e. big components of the game design), assigned tasks and fed back on our progress with our previously assigned tasks. We felt this method worked well for us as it allowed us to utilise each team members strengths in assigning them corresponding tasks and allowing other team members to advise where one person may be having difficulty with a task, as well as allowing open discussion for tasks in which we all wanted input e.g. initial game design for which we had two dedicated meetings, one where all team members were given an opportunity to share their game design ideas based on the requirements we had gathered and we discussed how some of these ideas may be combined as well as another meeting where we finalised our initial game design, coming to design compromises that all the team could agree on.

Assigning tasks at each meeting, rather than assigning tasks upfront at the beginning of the project, allowed us to deal with any inaccurate assumptions we may have made around the time each task would take whilst also allowing us to account for each team members personal circumstances throughout the project and lighten a team members workload when circumstances (i.e. illness or self isolation) might make tasks harder to achieve within a given time frame.

Tools

- Outside of the university mandated zoom meetings we held our team meetings on google meet as this allowed us to pre plan our meetings and send out invites via email that could easily be added into many of the teams personal planning tools, such as google calendar.
- We also used google drive to store project related documents as this allowed us to have an easily accessible team drive in which we could all store, access and edit documents, as well as giving multiple team members the ability to edit the same document at the same time.
- We decided to use GitHub pages for our project website along with an inbuilt GitHub pages theme (Cayman,) as this website would be easy for any team which later took over our project to replicate and also as github pages proved to be quite an intuitive and easy to learn tool.
- We used eclipse as our IDE as this offered a wide range of programming tools, as well as good integration with our GitHub repository
- We used the libgdx game-development framework/library in the implementation of our game. This framework has quite active forums which made it easier to find

answers to query's, however it is also quite low level which meant we had to write a lot of code to implement the game.

Other Tools Considered

- We considered using the ilntellij IDE instead of eclipse, however we found more resources to help us learn to work in eclipse, which is why we chose to work with eclipse in the end
- We considered unity as an alternative game engine, but decided against using it as many of its features were behind a paywall
- We considered using the lightweight java game library (lwjgl), however we expected libgdx to be a popular choice and therefore chose to sue this with future software expansion by other teams in mind

Part b

Team Structure

Our team's structure arose naturally, as we waited until our third meeting to officially assign roles. We also made sure to have deputies or shared responsibilities for most jobs to account for absences and to improve cooperation. In the end we assigned four roles as detailed below:

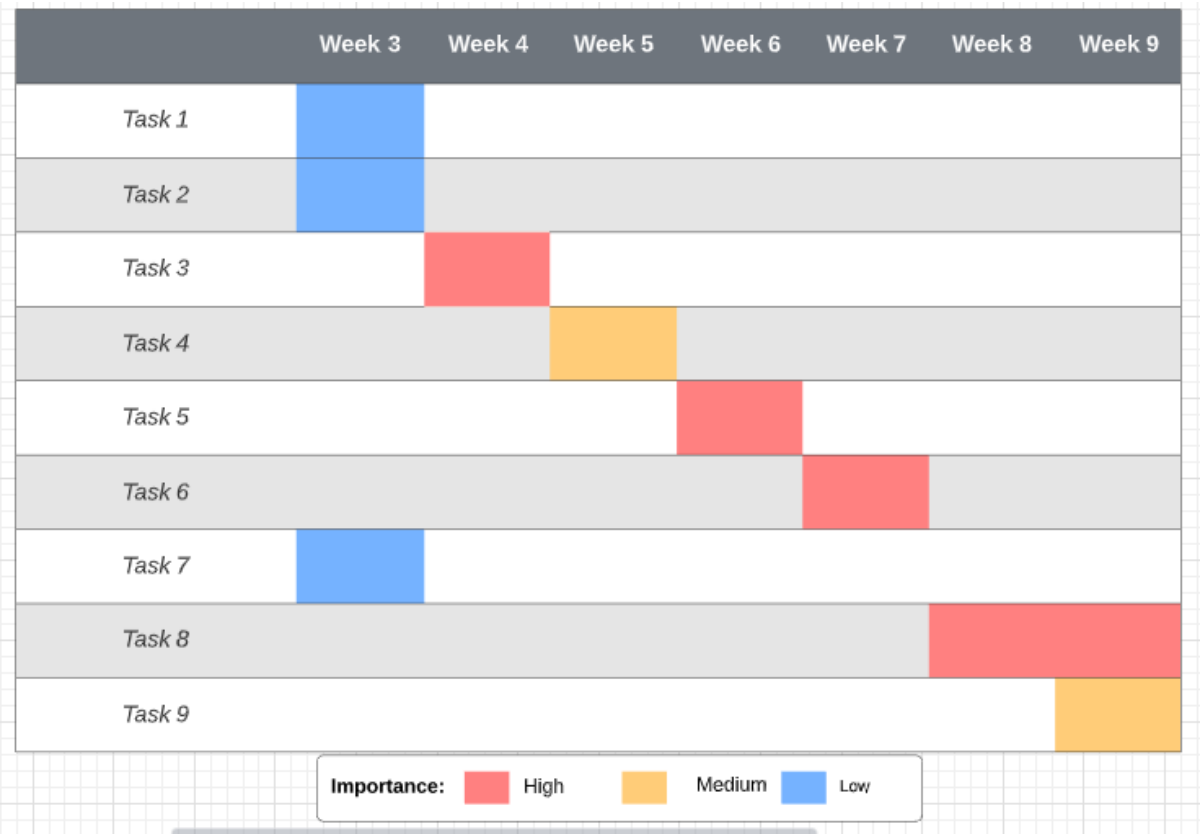
- Secretary: Responsible for taking minutes and logs of tasks discussed during meetings. This role had a deputy to account for if the main secretary wasn't at certain meetings
- Librarian: Responsible for version control and the organisation of documents. As this was an ongoing task that wasn't required to be done at specific times, the tole was divided between two people, rather than having a deputy.
- Chair: Responsible for leading the meetings and organising the order in which topics would be discussed, ensuring all business was seen to and everybody's voices were heard. Similarly, to secretary this role had a deputy to account for if the main chair wasn't at certain meetings.
- Report editors: Responsible for overseeing the production of reports. Similarly to librarian this role was shared between two people rather than having a deputy as it was an ongoing task

Part c

Table of Key Tasks

	Task	Importance	Dependencies	Anticipated Start Week (of term)	Anticipated Time in Weeks	Anticipated End Week (of term)
1.	**Establishment of a group name + logo + ethos. Familiarising within the group.**	Low	n/a	3	1	3
2.	**Assignation of roles within the group, i.e. Chairperson, Librarian, Secretary, etc…**	Low	n/a	3	1	3
3.	**Correspondence with the Client to establish requirements for game / requirement elicitation**	High	2	4	1	4
4.	**Planning of Game Design features + Establishment of which game engine to use influenced by game design**	Medium	3	5	1	5
5.	**Abstract Software Architecture**	High	4	6	1	6
6.	**Review of Software Architecture with Risk Assessment and Mitigation**	Low	5	7	1	7
7.	**Creation of a GitHub organisation for website and repository**	Low	n/a	3	1	3
8.	**Implementation**	High	6	8	2	9
9.	**Concrete Software Architecture**	Medium	8	10	1	10

This is also represented in the gantt chart below:



The above gantt chart shows a critical path of tasks 2,3,4,5,6 and 8
Tas9 is scheduled to start midway through task 8 as the concrete architecture can be completed once once some of the of the

We didn't stick to this initial plan fully and many tasks took longer than expected, as shown in our weekly snapshots. Some of the earlier tasks running over particularly meant that the amount of time left for implementation was greatly reduced.