Methods and Planning

Overview

We plan to use an agile-scrum approach for our software engineering project, as this will allow our team to carry out work in a flexible manner, evaluating our progress throughout the project, and make any necessary changes to our plan as we go along.

Collaboration & Development Tools

Throughout the design and development stages, we as a team will use a wide variety of software and tools to complete our tasks in the most efficient and appropriate way.

Team Communication

We will be using Slack for general team communication, it allows us to have separate chat channels for different aspects of the project, letting us keep all necessary information separated. Since separate discussions are split into separate channels, we are able to find important information more easily for future reference. The team engages in daily communication on Slack[1], we use this to consistently report our progress, ask questions on ambiguous issues and organise future in-person meetings. We plan to use integration between development tools and Slack to help us keep track of progress in one clear feed, in particular the GitHub[2] integration.

Finally, for online meetings we will be using Join.me. Join.me[3] is a free tool with voice, text and video chat as well as a screen sharing feature which allows a presenter to share their screens with other team members. This also allows for control of your screen can be passed on to other team members, allowing them to easily demonstrate ideas and make changes if necessary. Join.me allows efficient and effective team meetings to take place even whilst all members are in separate locations, looking ahead we expect this to be a very useful feature during the Christmas holiday period. An often overlooked benefit of Join.me is that it does not require users who are viewing to install software. This allows us to meet in a virtual environment even whilst travelling since we can simply use a public computer with a modern web browser to take part in the meeting.

Software Development

Through a detailed and constructive debate, the team decided that the project will be constructed using the Java[4] programming language. One reason that heavily influenced our decision was that Java is a language accessible to all members of the team, as well as the entirety of our cohort. This is due to Java being taught in the TPOP module in our First year. This fact is likely be a boon to the team once implementation begins and it should help increase development speed. We did give serious consideration to other programming languages such as C#[5] with Unity[6] however few in our team have any experience in the language thus using it would involve learning a new language which may take longer and cause problems later in the development process. Furthermore, from our understanding the vast majority of our cohort has little to no experience with C# therefore we believe they would be less inclined to choose our project at the end of assessment 2.

Another important reason for choosing Java was that a wide variety of libraries and frameworks for the purpose of game development are already freely available for use – we expect this to further speed up our development process. As different team members prefer different IDEs, we plan to ensure our code base is compatible with both Eclipse[12] and IntelliJ[13].

After engaging in detailed research, we obtained 4 main options of libraries/frameworks we could use: Swing[7], Mini2Dx[8], LibGDX[9], LWJGL[10]. Although LWJGL is a very good library for game graphics development, it is far too low level for us to create a high quality for the game within the time constraints we have been given. Swing can be used for simple 2D games however LibGDX and Mini2Dx provide more flexibility and game specific tools, making development easier. In the end decided to use the LibGDX Java game development framework. We chose LibGDX over Mini2DX because, although Mini2DX is based upon LibGDX it lacks some of the more powerful features that LibGDX has and we didn’t want to limit ourselves so early on. LibGDX has support for all of the IDEs that we intend to use, it provides features that simplify several game development steps. One of the key feature is that it has support for tiled maps, we plan to use software called “Tiled”[11] that allows us to easily create a high quality 2D map and export it for use with the framework. This will save us massive amounts of time compared to alternative solutions, this can be used to improve gameplay and ensure that there are no bugs.

So as to ensure work is done in a balanced manner, we will engage in design process whilst the development stages are progressing. We will split the team up based on their skills and their preferences. The main tool we will be using to design the game assets is Piskel, it is a online tool that allows us to easily create game assets. We expect designs to start as paper prototypes that will be developed further as the project goes on. We will also be using Tiled to easily draw maps and rooms.

To ensure that several playability features we are considering are popular with the game's main audience, we conducted a survey using Google Forms asking questions regarding preferred input methods.

Code Management

Throughout the development process, we will be using GitHub as our code repository since it allows for easy collaboration. All members of the team will be contributing to the repository.

To manage development tasks, we plan to use the GitHub projects feature of our code repository. This lets us manage our tasks and code issues with a Kanban[14] board, and allows for items to be assigned to team members so everybody knows who’s doing what. This allows us to keep our project management and code in the same place, which should encourage us to effectively keep track of what’s happening as we’re working on the code. GitHub also allows us to link issues to the project, and these are useful because team members can collaborate on design decisions and keep track of what’s happening in the code.

Alongside GitHub projects, we plan to use git’s branch and pull request functionality to help us manage multiple developers on the same project. For every pull request, we will ensure that a different developer performs a code review, and that all tests pass before approving the branch. Tests will be added to our code as we write it, so that we maintain a high level of test coverage. We plan to use a continuous integration server to run all the tests every time a commit is made to GitHub, this will ensure that nobody accidentally breaks the codebase.

Team Organisation

Team organisation model

As mentioned above, we plan on utilising the traditional Agile-SCRUM[16] model when organising our team. In-person meetings are held multiple times a week, here each team member goes through the progress they have made since the previous meeting. This allows all members to have a clear understanding of how the project as a whole is progressing. The meetings also allow us to sort out any queries or worries that members may have. The team will be split into 3 sub teams, and we then discuss what each sub team should accomplish and what tasks each member or sub team should meet by the next team meeting. These constant updates allow us to account for any unforeseen events and alter our development process based on these events.

Team Roles

Through unanimous voting, we chose Brooke Hatton as our group leader. We determined that his prior experience in tasks similar to those we were undertaking and his general teamwork skills made him a natural leader for our group. We may potentially change our leadership during future assessments if the group feels it is necessary.

<u>The reports in assessment 1 are assigned in the following manner:</u>

- Ben Jarvis was assigned the requirements document
- Benjamin Grahamslaw was assigned the risks and mitigations report
- Brooke Hatton and Jason Mashinchi were assigned the Architecture report
- Joseph Shufflebotham and Vishal Soomaney were assigned the Methods report
- Vishal Soomaney was assigned the user stories and scenarios

Each member proofread all of the documents and contributed to all of the documents through comments and small edits.

Team Collaboration

We plan to use the tools above to help us with our collaboration on this project, in particular Slack and GitHub. During our meetings, the team will split up tasks evenly and we will be able to keep track of progress with the Kanban board in GitHub Projects and the Slack integration. As we have been using Github throughout the first assessment, there will be no confusion due to work platform transition when we use Github and Git regularly in the second assessment for game implementation.

During the project, we will determine specific and realistic goals that we expect to meet within certain timeframes. We have chosen to use SMART[16] goals (specific, measurable, achievable, realistic, time-bound) to help us meet our targets. We chose this as it will help us produce useful goals that we can use to guide our work, and meet our deadlines.

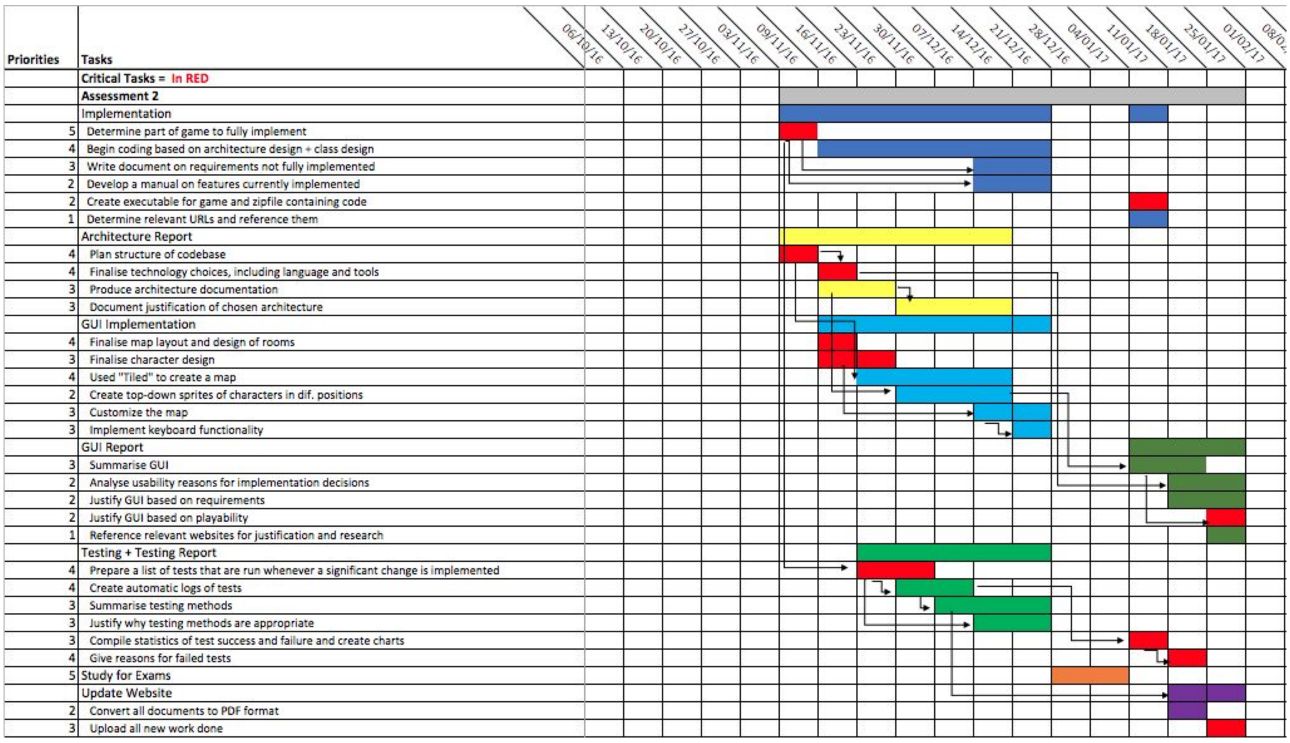
Systematic Plan

The focus of the team will fully switch to the second assessment on Wednesday 9th November, once the first assessment has been submitted. We plan on completing the second assessment by Tuesday Spring Week 2 giving us a week to account for any unexpected developments or fixing issues that happen to arise. It also gives us time to analyse, criticise and improve our own work thus improving the quality of our code and enhancing the functionality and efficiency. This will also ensure that we have very high quality documentation, once all group members are happy with the readability of our code we plan on requesting that students from other teams look at sections of our code to test readability. There will also be a 2 week rest from SEPR to account for time spent studying for exams and also the exam week itself.

A Gantt chart [18] containing the schedule for key tasks from all the assessments can be found in the appendix. This chart includes priorities, task dependencies and a critical path.

Assessment 2 Gantt Chart

Below is an extract of our Gantt chart related to Assessment 2. The full version can be seen on our website.



Priorities	Tasks
	Critical Tasks = In RED
	Assessment 2
	Implementation
5	Determine part of game to fully implement
4	Begin coding based on architecture design + class design
3	Write document on requirements not fully implemented
2	Develop a manual on features currently implemented
2	Create executable for game and zipfile containing code
1	Determine relevant URLs and reference them
	Architecture Report
4	Plan structure of codebase
4	Finalise technology choices, including language and tools
3	Produce architecture documentation
3	Document justification of chosen architecture
	GUI Implementation
4	Finalise map layout and design of rooms
3	Finalise character design
4	Used "Tiled" to create a map
2	Create top-down sprites of characters in dif. positions
3	Customize the map
3	Implement keyboard functionality
	GUI Report
3	Summarise GUI
2	Analyse usability reasons for implementation decisions
2	Justify GUI based on requirements
2	Justify GUI based on playability
1	Reference relevant websites for justification and research
	Testing + Testing Report
4	Prepare a list of tests that are run whenever a significant change is implemented
4	Create automatic logs of tests
3	Summarise testing methods
3	Justify why testing methods are appropriate
3	Compile statistics of test success and failure and create charts
4	Give reasons for failed tests
5	Study for Exams
	Update Website
2	Convert all documents to PDF format
3	Upload all new work done

Bibliography

[1] Slack [Online] www.slack.com [Accessed 8/11/2016]

[2] Github [Online] www.github.com [Accessed 8/11/2016]

[3] Join.me [Online] www.join.me [Accessed 8/11/2016]

[4] Java [Online] www.oracle.com [Accessed 8/11/2016]

[5] C# [Online] www.msdn.microsoft.com [Accessed 8/11/2016]

[6] Unity [Online] www.unity.com [Accessed 8/11/2016]

[7] Swing [Online] www.oracle.com [Accessed 8/11/2016]

[8] Mini2DX [Online] www.mini2dx.org [Accessed 8/11/2016]

[9] libGDX [Online] www.libgdx.badlogicgames.com [Accessed 8/11/2016]

[10] LWJGL [Online] www.lwjgl.org [Accessed 8/11/2016]

[11] Tiled [Online] www.mapeditor.org [Accessed 8/11/2016]

[12] Eclipse [Online] www.eclipse.org [Accessed 8/11/2016]

[13] IntelliJ [Online] www.jetbrains.com [Accessed 8/11/2016]

[14] Kanban board [Online] www.github.com [Accessed 8/11/2016]

[15] Piskel [Online] www.piskelapp.com [Acessed 8/11/2016]

[16] Waterfall to Agile: Flipping the Switch - Bhushan Gupta [Online] Available: http://www.uploads.pnsqc.org/2012/papers/t-21_Gupta_paper.pdf [Accessed 25/10/2016]

[17] Goal Oriented Requirements Engineering - Axel van Lamsweerde [Online] Available: http://dl.acm.org/citation.cfm?id=776930 [Accessed 04/11/2016]

[18] Gantt Chart [Online]