Method Selection and Planning

Method

For every software engineering project, choosing the right development methodology is essential for success. With this in mind, our team carefully approach the selection process by first considering the primary constraints imposed onto our project:

- Project Size: Short project of about 7 months.
- Cost of Delay: High as we have concrete deadlines.
- Team Size: Small team of 6 members.
- Requirement Enhancement: Requirements subject to changes.
- Client Feedback: Client can be contacted as often as needed.
- Experience on Project: Team members have limited experience with a project of similar type.

Given the high possibility of changes in the requirements as development proceeds, we immediately ruled out the Waterfall model; this is because the linear structure with discrete phases means that once the application is in the Testing stage, it would be very difficult to go back and make changes for the enhanced requirement. This constraint also drew our attention to the change-friendly Agile approach and after careful consideration of the Manifesto for Agile Software Development[1] we all agreed that the adaptive and cooperative characteristics of the Agile paradigm, along with the efficiency it brought to small development teams, make it most appropriate for the development process of “York Pirates!”.

Having chosen Agile as the main development principle, our team then looked at methodologies that follow its philosophy and decided Scrum[2] would best fit our constraints and preferences. This approach greatly emphasizes teamwork by having the team “try to go the distance as a unit, passing the ball back and forth” to address complex problems [3]. Scrum starts with a wish list of features in order of priority called a product backlog. Tasks will be picked from here and moved to a sprint backlog for development during the 1-4 week long sprints. Each sprint ends with a review and the team then chooses another backlog to develop in a next sprint. Brief team meetings called scrums are organised regularly to discuss progress and make plans.

Having the project split into Sprints enables our team to work with whatever requirements we have and build upon them with frequent client’s feedbacks. In addition, the Sprint review, where the client checks and provides feedback on the completed Sprint, accommodates for diverse changes that can be triaged in the following Sprint. This approach makes Scrum very flexible with requirement enhancements and allows us to capitalise on our ability to contact the client as often as possible. Also, by using Sprint, we can make sure the project is completed on time based on estimation from data like the velocity of previous Sprints. This is particularly useful given the high cost of delay of our project. Finally, a small team like ours will have an easier time self-organising (a core nature of Scrum) and requires less overhead for meetings and discussions compared to bigger ones.

Despite the usefulness of Scrum, there are aspects that are not suitable for us thus our project cannot solely follow its traditional implementation and adjustments must be made. The original Scrum requires daily Scrum with every members presented, which is difficult considering our members’ schedules. The client, though can be contacted as often as possible, is also very busy and requires a decent amount of time beforehand to set up meeting with. To accommodate for these problems, our team decided to change the frequency of meetings from daily to three times a week (Monday, Thursday, Friday), with the stakeholder attending at least the Friday meeting. Our project is also relatively short compared to similar ones in the industry, so the Sprint duration will be reduced to one week, from one Friday to the next, and Sprint review can be conducted during the Friday meeting of every week.

Tools

After careful analysis we concluded that our team requires tools for: Development, Communication, Document sharing, Version Control, Task Management and UML Diagram/Gantt Chart production. The tools we use in the development process of our project should be accessible by all members of our team, the client as well as others in case we pass on our project to another team.

Development

One of the main constraint requirements of the project is that it has to be in Java, thus we need a Java-based game engine or framework to help us develop our game. After careful consideration our team chose to go with LibGDX as it is one of the most powerful framework in Java, especially for 2D games. Libgdx is also extremely popular and used widely in the industry so there are a lot of existing tutorials and guides to ease us into the development process.

Communication

We will use primarily Facebook Messenger for communication outside of in-person meetings. We chose Messenger as it allows swift responses between members through notifications and can be accessed from a wide variety of platforms (smartphones, laptops, PCs, etc).

The team already uses it on a day-to-day basis and it has some handy built-in features such as image/video-sharing and polls. It is also fairly easy to create smaller group chats, allowing us to still communicate effectively if we split into smaller sub-teams to work on different aspects of the project.

Document Sharing and Production

We chose Google Drive as the hub for sharing our documents, logs and research with each other. Due to the fact that Google Drive is based on the cloud and has automatic backup, the risk of losing documents (Risk 6) is extremely small. All team members are experienced with it and through the university get unlimited storage and the ability to create team drives, a feature we will be utilising.

For document production we will use Google Docs as it is built into Google Drive and has the classic Word-style interface that everyone is familiar with. Furthermore, it allows for simultaneous editing and has a live commenting system making it much easier to collaborate on files as a team.

Version Control

We plan to use Github for the version management of our software as creating private repositories is free for students like us. In addition, the ability to setup different branches of a repository enables changes to be applied and checked gradually without breaking the code from the master branch should problems arise, which fits perfectly with our Scrum approach. GitHub is also decentralised which allows for restoration of code from local copies should the server failed (Risk 6).

An added benefit of using GitHub is that it also provides a service called Github Pages, which are websites that describe our repositories. We can use this for making our project webpage as it is easy to pick up and we do not have to deal with server-side problems.

Task Management

We chose Trello as it is accessible on multiple platforms and the Trello-board style with its drag-drop functionality is extremely easy to pick up and satisfying to use. This style also greatly promotes our Scrum approach as the sprint backlog can be laid out as cards and specifically assigned to members at the beginning of a sprint, making it easy to keep track of the progress of different cards throughout the sprint alongside the ability to set deadlines and an in-build comment system for individual cards.

UML Diagram and Gantt Chart Producer

We used StarUML to create our UML diagram and ProjectLibre for our Gantt Charts. The reason being both of them are free and easy to use with intuitive interface, which is important as none of us have experience making these documents before.

Team Organisation

Our group dedicated time to research software engineering roles that would comply with the project brief. From this, we decided on: Team Leader, Secretary, Client Interface, Risk Manager, Web Developer, File Management, Head Developer, Scrum Master and Audio Designer. Despite everyone having their own specialist roles, these are not fixed and the team's structure is subject to change if performance is lacking. It is also important to note that we will all be developing the code as a group and performing all of the tests together.

Team Leader - Alex Hodder-Williams
The group elected Alex as team leader during a meeting session as he showed constant commitment to the group and expressed an interest in furthering his leadership skills. Alex is a very vocal member of the group and would thus be suitable for sorting team conflicts and ensuring everyone knows what they need to do.

Secretary and Client Interface - Mehti Kendrick
The role of the Secretary is to ensure all meetings are planned and recorded. This involves keeping note of the location, attendance, time, what was discussed and what everyone needs to do before the next meeting. Mehti is also the Client Interface so he must be the group's direct link to the client, knowing what the client needs and wants from the project. He will know this by keeping in constant contact with the client in order to ensure the final product meets the clients expectations.

Risk Manager - Bradley Gee
As the Risk Manager, Bradley will responsible for recording all risks the group have or may encounter. He will try and help the group avoid risks throughout the project. Bradley was perfect for this role as he helped compile both documents concerning the risks of the project and therefore knows them better than any other group member.

Web Developer and File Management - Taylor Willmott
Taylor holds the most web development experience in the group and will therefore be the groups dedicated Web Developer. This job consists of formatting all group documents from the 'Group Google Drive', converting them into PDF's and then finally uploading them onto the website he has made. As a group, we felt this went hand in hand with the role of File Manager as if he was in charge of all files, it is easier for him to upload them to the website.

Head Developer and Scrum Master - Duc Vu
Duc has already co-developed a game before, making him the perfect candidate for Head Developer. Drawing from his previous knowledge, he aims to enforce code standards upon the rest of the group and check all code has been documented to the highest degree of accuracy.

Audio Designer - Matthew Barnes
Matthew was chosen as our Audio Designer as he has previous audio experience through his GCSE's and A-Levels. In this role, Matthew will be carefully designing sound effects and linking them to in-game cues.

Systematic Plan

Whilst we are using Trello for short-term planning and task-allocation, this isn't really suitable for a long-term plan of the entire project. As such, we decided to create some Gantt Charts to plan out each of the assessments. However, they are far too big for this document and so we are hosting them on our project's website. They can be found at the links below:

Assessment 2 - https://therandomnessguy.github.io/SEPR/Images/Ass2Gantt.png

Assessment 3 - https://therandomnessguy.github.io/SEPR/Images/Ass3Gantt.png

Assessment 4 - https://therandomnessguy.github.io/SEPR/Images/Ass4Gantt.png

A few things to note: we based the priority on the number of marks each aspect of the project was worth and the critical path for each assessment is highlighted in red.

References:

[1]: The Agile Manifesto. Available at:
http://agilemanifesto.org/

[2]: The Scrum Guide. Available at:
https://www.scrumguides.org/docs/scrumguide/v2017/2017-Scrum-Guide-US.pdf

[3]: The New New Product Development Game. Available at:
https://ullizee.files.wordpress.com/2013/01/takeuchi-and-onaka-the-new-new-product-development-game.pdf