| | |
|---|---|
| **Module** | SEPR |
| **Year** | 2019/20 |
| **Assessment** | 2 |
| **Team** | Salt N Sepr |
| **Members** | Alberto Lee, Archie Godfrey, Jacqueline Eilertsen, Jake Schoonbrood, Joshua Levett, Matteo Barberis |
| **Deliverable** | Method Selection and Planning (Plan1_update) |

# Method selection and planning

## Development and collaboration tools

The team chose to use the agile method SCRUM; this method fits into the framework of the software engineering project that will be running over the course of 2nd year. By using the SCRUM method, the team benefits from its advantages such as lowering the risks by keeping a transparency for product owner and stakeholders and the team. There is a flexibility that allows adaptability for customer changes and requirements which other methods do not take into consideration as they use a higher amount of documentation. With less documentation that comes with SCRUM changes are easier to handle as clarification through documentation.

**The assessment is split into 4 smaller assessments each with different requirements; SCRUM fits well with the software engineering project with this in consideration as the flexibility of SCRUM is beneficial to maintain the adaptability of the team.**

Agile methods are based on a manifesto with four principles: individuals and interactions over processes and tools, working software over comprehensive documentation, customer collaboration over contract negotiation and responding to change over following a plan. With these four principles, the characteristics for agile methods are incremental, cooperative, straightforward and adaptive.

By doing weekly sprints we develop an incremental approach, that keeps the team and project open to new requirements from the customer and therefore we also remain adaptive in our approach. By using this method we create a straightforward approach with clear documentation of how the project will proceed.

By using the SCRUM method, the team will perform weekly sprints and therefore maintain a deadline one week before hand-in; with daily stand-ups over Facebook Messenger; **over the holidays the weekly review sessions will also take place over Messenger as a meet-up is not possible**.

For development tools we will be writing the application in Java whilst using the libGDX open-source framework [1]. Using libGDX as our framework will allow us to reuse code between platforms, significantly decreasing the amount of new code that has to be written, increasing the speed at which it can be ported.

LibGDX is specifically good for 2D games, which is the format our game will be. It functions with programming languages Java, C and C++, with the requirement that we have to use Java for our game [2]. Using libGDX allows for Java tools to be reused as the framework is Java based. Furthermore, on its own Java has a slow performance[3], however libGDX uses Java Native Interface (JNI) which uses a native platform that enables Java to run on a virtual machine. This means that Java can be used in combination with languages as C and C++, which can handle different methods better than Java and which are closer to assembly code. As another tool

within libGDX we will use the 2D particle editor which allows us to make 2D effects for our game, such as the water effect from the fire engines; with the 2D editor it allows for velocity, duration changes and in which direction the water is going[4].

**There are two big build systems: Gradle and Maven. Between these two we chose Gradle as it has a faster build time and more modular customisation. With the incremental builds that Gradle use, it checks whether tasks are updated, if it is the task is not executed which gives a shorter build time. One of the downsides to Maven a goal can only be attached to another goal, this restriction is on Gradle. As we chose to work with LibGDX it is easier to use Gradle as it requires less plugins to make it compile [5].**

**Tiled[6] is an open-source general purpose editor we are using to make a map of York. We chose it as it produces a Tiled Map in a format that is accepted by Libgdx. Tiled provides an easy-to-learn interface and allows the map to be separated into layers. Layers can later be used to define which areas of the map should be collided with, this is useful as it eases the implementation for the game.**

**For the creation of assets, we are using Aseprite [10]. Aseprite is a pixel art and animation creation software specifically built for game development. It has many features that allow for the quick development of assets which will be useful for creating new art as well as modifying open-source art for the game. We also used MagicaVoxel [11] for the creation of the firetruck assets. MagicaVoxel is a open-source 3d Voxel editor that allows for the creation of 3D models. As a team we decided we wanted to have a fake 3D fire truck, this is accomplished by creating a Voxel model which is then sliced into multiple layers. In the game it is reconstructed with a slight offset between layers to create the effect of having a 3D model in a 2D environment.**

**The codebase has been setup so that it can be written in Eclipse, Intellij and Visual Studio Code. We chose to make it compatible with all 3 programs in order to make it as accessible as possible for our team and teams that inherit our code later on.**

One of the tools used for collaboration between the team is Github. Github is a code sharing platform that uses the git version control system, improving the ease of multiple people working on the same codebase between weekly sprints, **furthermore we are also using Github to manage the tasks that need to be done. This also provides a history that we can use to check project progress against the Gantt chart to stay within the schedule**. We will also be using Google Drive for storing and collaborating on documentation as well as report taking. Lastly, Facebook Messenger will be our main means of communication for daily updates on progress and will allow us to solve simple problems with no meet-up face-to-face necessary. Using several platforms for different aspects of the project will allow more organisation and structure between code, documentation and daily updates. **We decided on these tools because team member was familiar with them before the start of the project.**

## Team organisation and why

The team members have been allocated different roles in the group based on Belbin team roles which will help create a high-performing team. The different roles are based on the strengths and weaknesses of each team member, which will result in a more efficient group [7]. We use the Belbin team role assignments as they can be adapted to the team and roles can be allocated to more than one person, so it fits more directly to the person and the teams working ethics. The following list is how the team roles have been allocated for the first assessment.

- Joshua Levett - Specialist
- Jake Schoonbrood - Completer finisher
- Archie Godfrey - Monitor evaluator and implementer
- Matteo Barberis - Resource investigator
- Alberto Lee – Plant
- Jacqueline Eilertsen - Team worker and co-ordinator

These roles have been allocated to each individual person on mutual agreement of the team. The roles were assigned to each individual by using a team role test [8] that shows the most prominent traits for each individual; the results were then applied to the Belbin role strengths and weaknesses and what the member felt most comfortable with. Joshua Levett showed equal strengths as executive, analyst and expert from the team role test, based on this and what he was comfortable with regarding the roles in Belbin he was allocated the specialist; who shows in-depth knowledge of key areas.

Jake Schoonbrood was allocated the role as completer finisher who is a perfectionist and will correct all errors; in the test Jake Schoonbrood biggest trait was as a driver which means he is ambitious and energetic: needed for the end of an assessment to make corrections and a finishing touch. With this considered and what he felt most comfortable with he got the role as completer finisher.

In the test Archie Godfrey had most percentage as an expert, which means he showed great skill with a specific task at hand and have therefore got the roles implementer and monitor evaluator as the definition of these roles is providing a logical eye and able to carry out strategic plans.

The resource investigator is an optimistic and outgoing person who explores many opportunities and bring ideas to the team; with this in mind and the test results being majority innovator, Matteo Barberis was allocated the role as resource investigator.

Alberto Lee got the role as plant; this means he is the creative and imaginative mind of the team which also suits the test where he scored most as an innovator. Furthermore he felt confident in being able to contribute a lot to the game design and implementation as he has experience making games.

As a split between team worker and co-operative Jacqueline Eilertsen was allocated she showed most skills as an executive and chairperson during the test. By definition of the team

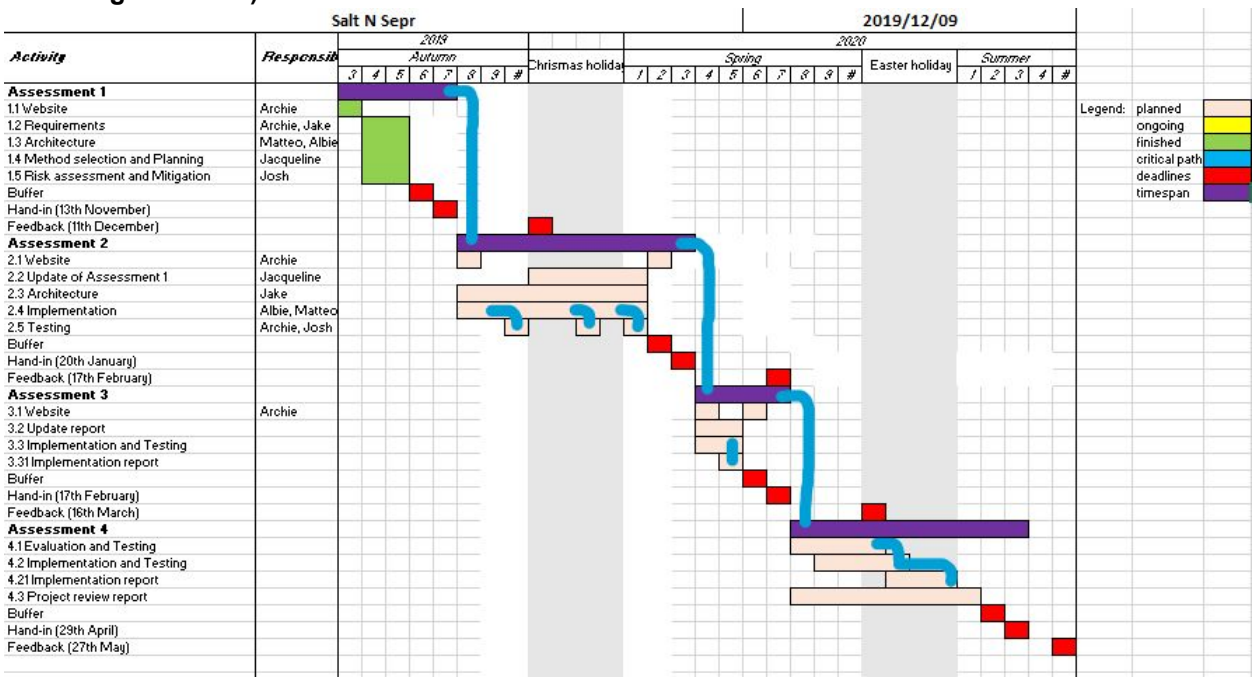
roles the team worker and co-operator shows focus on the team and team member strengths by being diplomatic and perceptive.

These roles have been delegated by the team as a whole and who is most confident having which role in the team; the roles will be maintained throughout the project unless it is seen unfit or the team no longer has the need for that role.

## Plan for rest of SEPR (dependencies and critical path)

The most common task dependency is finish-to-start which means the predecessor task needs to be finished before the next task, the successor, can begin [9]. Using this approach assessment, one needs to be complete before assessment two can begin. This includes the updating of the assessment briefs and documentation. In regard to dependency for each individual assessment tasks will run alongside each other as to maintain the plan for finishing a week before the hand-in deadline; however, some tasks do depend on the previous task such as implementation and the implementation report are dependent on each other, as the report cannot be done before the implementation is done.

The future for the project is shown in figure below **(updated with assessment 1 shown in green indicating it is done)**.




The figure shows the plan for the future of the SEPR project with all the tasks required for each assessment and the duration of each task within the assessment. We aim to finish all our assessments a week before each hand-in date, which is what the buffer represents. By having a buffer, we leave ourselves extra time to redo and go through all of our tasks as a group, as we split up tasks to individual or smaller groups, so we are more efficient in our workload. The key tasks for all of the assessment is laid out on the figure, along which the

earliest starting dates and latest finishing dates.

The critical path for this plan can also be seen on the figure (**is now shown in a bold blue line)**. When a task finishes and another start there is a critical path; however, we will aim to be able to run parallel in regards to for example implementation and implementation report, as not to waste time. Under assessment two testing would be reliant on implementation although it is going to be done in intervals throughout implementation.

For assessment two, the workload is going to be split between smaller groups where each individual is working on parts that suits their strength. The key tasks for assessment two within the main tasks are implementing two fire engines, three ET fortresses which should not be able to improve over time, ETs should not be able to destroy the fire station. For the architecture there needs to be a structure of code with a specific programming language and the tools used. These are the general tasks for assessment two including the implementation tasks and other tasks such as the updating of assessment one and architecture.

## References


[1] "libgdx", Libgdx.badlogicgames.com, 2019. [Online]. Available: https://libgdx.badlogicgames.com/. [Accessed: 22- Oct- 2019].

[2]"How to choose an Android game engine: libGDX vs Unity - Dario Penic", Dario Penic, 2019. [Online]. Available: http://dariopenic.com/how-to-choose-an-android-game-engine-libgdx-vs-unity. [Accessed: 24- Oct- 2019].

[3]J. Cook, "LibGDX Game Development By Example", Google Books, 2019. [Online]. Available: https://books.google.co.uk/books?hl=en&lr=&id=uxt1CgAAQBAJ&oi=fnd&pg=PP1&dq=why+use+libgdx+&ots=6L0SGsk3U-&sig=Uf3O5xC6LlbXzkkjvz10ORIKZj0&redir_esc=y#v=onepage&q&f=false. [Accessed: 25- Oct- 2019].

[4]"libgdx/libgdx", GitHub, 2019. [Online]. Available: https://github.com/libgdx/libgdx/wiki/2D-Particle-Editor. [Accessed: 27- Oct- 2019].

**[5] A. Springfellow, “Gradle vs Maven”, DZone, 2017. [Online]. Available: https://dzone.com/articles/gradle-vs-maven [Accessed: 13- Dec- 2019].**

**[6]” tiled”, Tiled, 2019. [Online]. Available: https://doc.mapeditor.org/en/stable/ [Accessed: 13- Dec- 2019].**

[7]"The Nine Belbin Team Roles", Belbin.com, 2019. [Online]. Available: https://www.belbin.com/about/belbin-team-roles/. [Accessed: 29- Oct- 2019].

[8]"Team roles test | take this free team roles test online at 123test.com", 123test.com, 2019. [Online]. Available: https://www.123test.com/team-roles-test/. [Accessed: 30- Oct- 2019].

[9]"Understanding Task Dependencies in Project Management", Projectinsight.net, 2019. [Online]. Available: https://www.projectinsight.net/project-management-basics/task-dependencies. [Accessed: 01- Nov- 2019].

**[10] ” Aseprite”, Aseprite, 2019. [Online]. Available: https://www.aseprite.org/ [Accessed: 13- Dec- 2019].**

**[11] ” MagicaVoxel”, MagicaVoxel, 2019. [Online]. Available: https://ephtracy.github.io/ [Accessed: 13- Dec- 2019].**