# Method selection and planning

## Software Engineering Methods

For project development to be successful, it is essential for the methodology to be the best suited for the project. In order to select the best-suited methodology, the following factors were considered: time constraints, user requirements, the client, group size, individual responsibilities, project size and other factors. These factors were chosen due to being impactful on how the development of the project and its result were to progress, while the scale of the project would strongly impact the choice of methodology.

During the research into the different options, a common methodology was Plan-driven Software Development. This method, described as “heavy-weight” and “traditional”, depended on clearly defined increments. An example of this method is RUP, a method that divides the development process into four distinct phases.

A benefit of this methodology is that the steps of development would be repeatable and comparable, making the document development much easier, due to simply building upon the previously iterated version. Ultimately, we decided against the Plan-Driven methodology. This was due to the requirements being completed during the first iteration and not developed, which would not be suited to rapid development as requirements would need to be changed often, a situation the group would have to face further in the project timeline.

Similarly, the tight constraints of the procedure and segmented development would give the developers less freedom and flexibility; an important factor in the choice of methodology due to the timetabled schedule and requirements of the project team as university students, alongside the factor that the team may not have experience in project development [1].

Another methodology we looked into was Agile Software Development. Unlike Plan-driven Methods, the method prioritises “individuals and interactions over processes and tools” and “respond to change over following a plan” [3]. As we wanted to consider methodologies that were used in projects similar to ours, we found Agile as the most common in industry software development teams. Additionally, Agile Development contained an idea based around scheduled 1-4 week sprints; this would fit especially well around the project deadlines while allowing for fast and frequent feature development and allow for constant refining and prioritising the overall product backlog [4].

After deciding on an Agile Methodology, the particular framework to follow was still under discussion. The SCRUM framework was an obvious choice due to its increasing popularity within large organisations, as well as being best used with a small team size. The SCRUM framework splits the project into multiple smaller projects, creating a sprint backlog which priorities certain requirements over others [5]. After the sprint, the team reviews the work produced and makes any changes prior to moving onto another chunk of the backlog. While these sprints were typically one month long, the group opted for one to two week long sprints, to account for other academic responsibilities. The daily scrum meetings were also reduced to one or two meetings a week because of this, with Asana emails to remind members of their assigned tasks and Facebook Messenger to raise any queries prior to the next meeting. With the team agreeing on these adaptations to the framework, our decision to use SCRUM was confirmed. Our feedback post-deadlines would be our basis for client feedback, allowing for iteration of documentation.

## Development & Collaboration Tools

### Version Control: GitHub & GitKraken

To store the game in its development phase, the team had experience with two platforms: GitHub and Google Drive. Since Google Drive often stores programs as .txt files and has limited features for commenting on changes, GitHub was far superior displaying the code and allowing editing comments, while also allowing us to write our website code on GitHub. It was also believed that it would suit the project well, due to GitHub being widely used for open-source projects and allowing multiple developers to contribute to the project [6]. Each team member also downloaded GitKraken, a cross-platform Git Client which allows each member to view changes to the master program and access the GitHub Repository easily.

### UML Tool: StarUML

To create the UML diagrams we have chosen to use StarUML. We chose this software as it was free and several team members had considerable experience using it, reducing the learning time needed.

### Communication:

To encourage frequent communication during the project timeline, a Facebook Messenger Group Chat was set up to help arrange meetings, in addition to raising questions while working individually and discuss deadlines. Alongside this, the application ASANA will be used to keep the team updated and informed through emails as to their assigned tasks and its details. With an overall group opinion being on meeting up in person, we decided a VOIP group was unnecessary and organised a weekly meeting every Monday in order to keep ahead of the project deadlines. Other communication tools, such as Slack, were briefly considered as a communication method; however due to the lack of ability to screen share, in-person meetings were much preferred. Communication with the client will be through email or Skype, depending on the client's preference and situation.

### Documentation & File Sharing: Google Docs

In addition to GitHub, a platform was needed to store additional documentation or documentation still in development. Internet-based cloud storage methods, such as Microsoft's OneDrive and Google Drive, were decided to be the best system due to the team members wanting to work on their personal computers in addition to the university computers, providing a large amount of flexibility in where the team can access their files. Due to the majority of team members already using Google Drive for their personal documents, we agreed on Google Drive. As a platform, Google Drive is extremely useful, due to fact multiple users can work on a document simultaneously, with changes being saved in a list, and users can add comments on the document to raise problems during development.

### Task Organisation: Asana

To distribute tasks within the team we have been using an application called ASANA. ASANA allows the team members to assign themselves or others to a specific task and assigning it a deadline. The assigned members are emailed the task details in addition to this. Once completed, the task can be marked as complete, which also emails the other team members to notify them of the completed task, providing transparency for all team members as to the project's current state.

## Team Organisation & Structure

### Organisation

When deciding on how best to organise our team, we looked at different types of software that could be of use. After conducting some further research, we decided to use an application called ASANA which can be used to manage team plans, projects and processes [7]. We chose this application as it was easily accessible to all team members, as it is available as both a web and mobile application, and is ‘one of the most popular task management apps available’ [8]. This allows us to organise who will be in charge of managing and completing each task while allowing everyone's progress to be tracked by the SCRUM master.  Once a task has been completed, it can be ticked off our linked progress list and all team members will be notified of this development. Completed tasks will then be added to a review list and later will then be reviewed by the rest of the team. Normally this will be completed during team members or discussed in the Facebook messenger group chat.

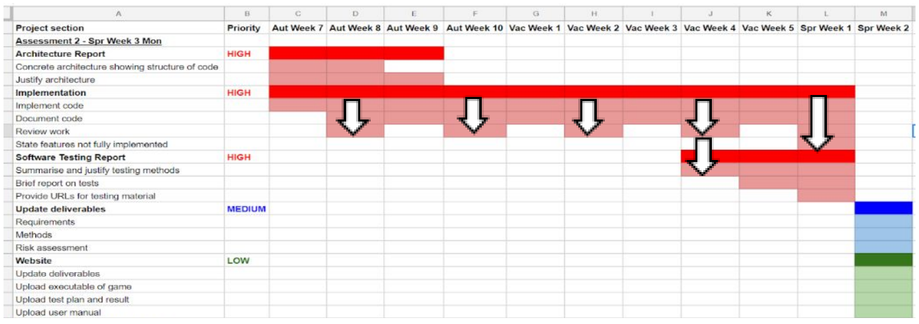
### Structure

With our choice to follow a SCRUM framework, we also had a team organisational structure to follow, forming a SCRUM team. So to follow to agile method principles, the team needed to allocate the role of “SCRUM Master” to a team member which would require them to guide the team and ensure the SCRUM process runs smoothly. This role is essential to keep the team focused on its goal [5]. As some team members may have more or less experience in a software engineering project, we discussed each other’s skills and experience and who would be willing to take up the role. Ultimately, the decision was to split the role between Eleanor and Merry; who would monitor the documentation and programming respectively. This option was chosen to make the role of SCRUM Master easier for those taking it, as well as hopefully improve the quality of their performance. Additional roles were decided to not be fixed, allowing for flexibility in the team’s experience. As SCRUM requires a daily meeting, the team discussed this and decided it would be highly implausible due to the team’s academic and individual commitments, settling on one to two weekly meetings.

In addition to this, we have created a sprint backlog to act as a ‘prioritised wish list’ [5] which holds our project requirements. Our sprints will last around a week to ensure that tasks are completed at a constant and fast rate.

**Project Plan**

We have chosen to prioritise each task depending on its amount of work, amount of marks and its vitality to the overall project. Each task has been broken down into smaller sections, which will help to ensure we complete all aspects of the task efficiently and effectively. To organise the development of our project, we have used a Gantt chart, as shown below. This representation clearly shows how long each task should take and the progress that is required before completing further tasks.



| A | B | C | D | E | F | G | H | I | J | K | L | M |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Project section | Priority | Aut Week 7 | Aut Week 8 | Aut Week 9 | Aut Week 10 | Vac Week 1 | Vac Week 2 | Vac Week 3 | Vac Week 4 | Vac Week 5 | Spr Week 1 | Spr Week 2 |
| Assessment 2 - Spr Week 3 Mon | | | | | | | | | | | | |
| **Architecture Report** | HIGH | 🟥 | 🟥 | 🟥 | | | | | | | | |
| Concrete architecture showing structure of code | | ▨ | ▨ | | | | | | | | | |
| Justify architecture | | ▨ | ▨ | ▨ | | | | | | | | |
| **Implementation** | HIGH | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | |
| Implement code | | ▨ | ▨ | ▨ | ▨ | ▨ | ▨ | ▨ | ▨ | ▨ | ▨ | |
| Document code | | | ▨ | ▨ | ▨ | ▨ | ▨ | ▨ | ▨ | ▨ | ▨ | |
| Review work | | | ▨ | | ▨ | | ▨ | | ▨ | | ▨ | |
| State features not fully implemented | | | | | | | | | | | ▨ | |
| **Software Testing Report** | HIGH | | | | | | | | 🟥 | 🟥 | 🟥 | |
| Summarise and justify testing methods | | | | | | | | | ▨ | ▨ | ▨ | |
| Brief report on tests | | | | | | | | | | ▨ | ▨ | |
| Provide URLs for testing material | | | | | | | | | | | ▨ | |
| **Update deliverables** | MEDIUM | | | | | | | | | | | 🟦 |
| Requirements | | | | | | | | | | | | ▨ |
| Methods | | | | | | | | | | | | ▨ |
| Risk assessment | | | | | | | | | | | | ▨ |
| **Website** | LOW | | | | | | | | | | | 🟩 |
| Update deliverables | | | | | | | | | | | | ▨ |
| Upload executable of game | | | | | | | | | | | | ▨ |
| Upload test plan and result | | | | | | | | | | | | ▨ |
| Upload user manual | | | | | | | | | | | | ▨ |

The critical path for each task is shown in block colour; we will have multiple teams working on different tasks concurrently. Task dependency is show by the arrows, highlighting which areas must be completed before starting other tasks.

We have updated this the gantt chart for assessment 1 to include the user manual, as we had previously overlooked this task. [9]

**Assessment 3, Spr week 3 - Spr week 7**

| **Task** | |
|---:|---|
| **Change report** | Summarise approach to change management. |
| | Justify any changes to documents. |
| **Implementation** | Implement documented code. |
| | Implementation of architecture and requirements. |
| **Website** | Update deliverables. |
| | Upload executable of game. |
| | Upload test plan and result. |
| | Upload user manual. |

This plan has been updated to further break down the tasks for assessment 3 and clearly show when we plan to complete each one. [9]

**Assessment 4, Spr week 7 - Sum week 3**

| **Task** | |
|---|---|
| **Evaluation and testing report** | Explain and justify teams approach. |
| | Comment on met/unmet requirements. |
| **Implementation** | Implement documented code. |
| | Summarise software modification. |
| **Project review report** | Summarise approach to team management. |
| | Summarise software engineering methods. |

This plan has been updated to further break down the tasks for assessment 4 and clearly show when we plan to complete each one.

## References

[1] Wikiversity, "Plan-Driven Software Development," [Online]. Available: https://en.wikiversity.org/wiki/Plan-driven_software_development#RUP_Artefacts_[Artifacts_in_American-English].

[2] WeblineIndia, "Top 15 software development methodologies with their Advantages and Disadvantages," 9 Feb 2018. [Online]. Available: https://www.weblineindia.com/blog/top-15-software-development-methodologies-with-advantages-and-disadvantages/#sdm12.

[3] E. Carmichael, "Benefits and Pitfalls of using scrum software development methodology,"QASymphony, 22 Aug 2018. [Online]. Available: https://www.qasymphony.com/blog/agile-methodology-guide-agile-testing/ .

[4] Segue Technologies, "8 benefits of Agile software development," Segue Technologies, 2016 June 2016. [Online]. Available: https://www.seguetech.com/8-benefits-of-agile-software-development/.

[5] Belatrix Software development blog, "Benefits and Pitfalls of using scrum software development methodology," Belatrix Software development blog, 26 Dec 2013. [Online]. Available: https://www.belatrixsf.com/blog/benefits-pitfalls-of-using-scrum-software-development-methodology/.

[6] Unleashed Technologies, "What is GitHub and how can it benefit your development team?," Unleashed Technologies, 9 Sept 2015. [Online]. Available: https://www.unleashed-technologies.com/blog/2014/08/01/what-github-and-how-can-it-benefit-your-development-team.

[7] Online-Asana, "Use Asana to manage your team's work, projects and tasks," Asana, [Online]. Available: https://asana.com/.

[8] Process Street, "Trello vs Asana: The best project management app in 2017," Process Street, 2 Aug 2018. [Online]. Available: https://www.process.st/trello-vs-asana/.

[9] SEPR “Updated Gantt chart” Abstract Delete [ONLINE] Available:
https://mh1753.github.io/AbstractDelete/Documentation/Assessment%202/Plan2Gantt.pdf