5. Project Review Report

5.1 Method/Tool	**Reasoning**	**Changes**
Development method: Scrum	**1.** The incremental nature of scrum suited our team size and organisation [2] **2.** The style of a scrum meeting, whereby work done in the previous sprint is discussed and further work in the next sprint is decided upon [2], suited our situation well; not every team member's timetable was the same, and the SEPR practicals in term 1 and 2 acted as our face-to-face scrum meetings **3.** Each sprint left us with a working codebase to fall back on in case of error, lost files and/or backup issues **4.** The roles of scrum (product owner, scrum master) provided clear points of contact for team members **5.** Provided a clear vision of development requirements and each member's tasks throughout the project	**1.** This was used successfully throughout most of the project and made sure that the project was being continuously developed **2.** During Assessment 2 as the team was separated over the Christmas period and everyone was available at different times, instead of week-by-week sprints, everyone did their work in their own time. The team still worked toward incrementally improving the product and documentation in a Scrum-like fashion. This allowed the team to be even more flexible in terms of working time, which was necessary to accommodate the change in circumstance **3.** During Assessment 3 and 4 we went back to week-by-week sprints, as the incremental improvement approach was suited to the task of adding new functionality to an existing project.Since we were implementing a different team's requirements and extending code written by another team; we set aside some time at the beginning to become familiar with the current implementation.
Development lifecycle: Scrum + evolutionary approach (exploratory development style, as described by Sommerville [1])	**1.** This allowed for time to re-evaluate our requirements and promoted involvement with the customer. Our increments produced by Scrum could be analysed to see what features fit with the customer's wishes, and could then be added to with new features. The scrum method of software development recognises that customer requirements, and therefore development processes, are often unpredictable. It is a highly flexible process and is also well-suited to teams of our small size [2].	**1.** No changes were needed since any changes were accommodated by the lifecycle anyway.
Version Control: Github [3]	**1.** Github is fairly simple to use after an initial learning curve. **2.** Allows for development along branches, which merged with ease. Branches were good for experimenting with code especially during the swaps to help understand the code. Also people coded at the same time so branching allowed changes to be committed without affecting other people's code. It was also chosen as it is truly distributed, meaning that we always have at least seven copies of our code (one for each team member) which can be accessed without access to the GitHub server.	**1.** We did not change this method throughout the project; github proved to be a reliable, useful asset from the start **2.** It was also well-suited to the assessment swaps by nature, providing inbuilt version control, traceability, and collaborative tools. It was widely used by the other groups in our cohort and so we saw no reason to change our use of it.
Website: Github hosting	**1.** We used GitHub pages as our web hosting, as it is free and is connected with our GitHub project. There is also a simple editor that can quickly generate a good looking website with minimal effort.	**1.** Though the website contents changed throughout the project, we did not change this method as it produced good-looking, simple web pages and provided all the functionality we needed, as well as being built into github, a tool we continually used throughout the project
Communication: Slack messaging service [4]	**1.** We used this website as our primary messaging service because it allowed us to communicate instantaneously with team members. Messages could either be posted to	**1.** We successfully used slack throughout the project to communicate outside of face-to-face meetings **2.** We found that our message organisation would occasionally be less rigorous; 'meeting' discussion would

	the entire team in one of several channels or privately to an individual team member **2.** As new messages on a different topic could go into a different channel, conversations were easier to hold without discussions being disrupted **3.** All messages are archived and searchable, so we were able to access all previous communication at any time.	sometimes be held in the ‘general’ channel. We made an effort to organise our communication properly after this was noticed **3.** Slack became particularly useful during assessment 2, due to the team being split up over the Christmas period. Meetings were held via Slack rather than in person and work was distributed during these meetings, along with progress reports **4.** In Assessment 3 and 4 we went back to face to face meetings, but Slack was still extensively used for progress reporting and meeting organisation.
Collaboration: Google Drive [5]	**1.** Free, easy-to-use service with integration with our university accounts/emails **2.** Allows team members to immediately view the latest iteration of the document and post comments for others to consider **3.** Edits to documentation can be seen by other group members in real time.	**1.** No changes were needed as it provided a reliable service.
Development language: Java with LibGDX on Eclipse IDE	**1.** The team was familiar with the language due to it being covered in first-year modules. **2.** Therefore starting the project, and continuing other team’s software, was easier **3.** We also had more choice when swapping with other teams as many others used Java/LibGDX for implementation, therefore cutting down on time required to familiarise ourselves with unfamiliar languages and libraries.	**1.** Since Java LibGdx projects were always chosen for swaps, no changes were needed. **2.** This makes it easier to build on previous experience on using this method.
IDE: Eclipse	**1.** The team was familiar with using eclipse . **2.** Has useful tools for refactoring code which speed up development. **3.** JUnit is built in so easier for testing. **4.** Already installed on university computers.	**1.** No changes were needed as Java was used for all projects and it provided all the functionality that was needed.
UML software: JsUML2	**1.** It is free online and can be exported to different file formats. **2.** Was the easiest to use when tested by various members of the team as it uses a drag and drop UI.	**1.** No changes were needed as it could produce the graphs that were needed..
Gantt chart: Gantter	**1.** It is free with full access to features unlike other similar products. **2.** Integrates with google drive which we were using. **3.** Can export to other formats. **4.** Can be accessed online and is cloud based so can be used for collaboration.	**1.** No changes needed. Easy to use.

5.2 Team Management

We had one Scrum Master, one Product Owner, and everyone else was a Scrum Team member as we are following the Scrum principles. They are defined as [1]:

1. Scrum Master - The Scrum Master was responsible for ensuring the team could always do the best work they possibly could, and that the principles of Scrum were followed. She helped by organising meetings, made sure the backlog was always in a workable state by working alongside the Product Owner, and made sure the team didn't take on too much (or perhaps too little) work during a sprint. If the team deviated from the Scrum principles, for instance, by not finishing a sprint with a potentially shippable result, it was the Scrum Master's job to address this.
2. Product Owner - The Product Owner was responsible for having an overall vision of what the Scrum was trying to build and prioritised the product backlog accordingly. However, the Product Owner (normally) didn't have direct control over the sprints, such as how much work the team did in a sprint or how many sprints were run. This was down to the team and Scrum Master, as they knew best how much they could do in a sprint, and how best they work. The Product Owner's job was to motivate the team by providing clear goals for them to achieve.
3. Scrum Team Members - Team Members are responsible for building the product. Scrum differs from more traditional software engineering project models in that team members are not split into different roles such as tester, architect, and programmer, but rather everyone works together to complete work that was agreed upon for each sprint. This requires all team members to have an understanding of the project requirements and actively participate in all areas of the development, rather than waiting to be allocated something to program. Each member may work on a variety of different tasks within the project to further the goals of each sprint.

5.2.1 Changes in management

We retained the Scrum Master model throughout most of our project. It worked well to keep the team organised. However, due to the team size, roles were less rigidly defined. All of us being students, there was no particular hierarchy other than what we defined for ourselves. It was occasionally natural for others to step in and fulfil certain duties such as organisation and work allocation. While this was mostly the job of the Scrum Master, the close-knit nature of our team meant that, in the case of any absences, other members could fill in at their discretion.

The Product Manager was less adhered to than the Scrum Master. As more of a managerial role, we found that, due to the scale of work required and the dynamic of a university group project in which everyone is required to contribute their share, a position like Product Owner wasn't ideally suited to our project. However, they did prove themselves valuable when doing interviews and question sessions with clients due to their work on a vision of the final product, so this aspect of the role was not changed. The position differed from the scrum standard in that, due to also being involved in sprints, the Product Owner did offer some contribution to the meeting organisation.

This Scrum Team Members role was mostly adhered to. The equally-spread work nature of scrum is why we had initially picked it as our method, and Team Members ensured that principle was followed. Initially it was planned that everyone, including the product owner and scrum master, would be writing code, which is atypical of Scrum. Instead some of the group were more inclined only worked on the documentation. As documentation is a large part of the assessments, we found that there was a need for some members to focus only on documentation. This allowed for those more confident in their coding capabilities to work on the game with fewer merge issues, as less overlaps occur when less people work on the code.

For each Assessment the roles for documentation and coding were assessed based on the amount of work to be done and the time. For example Assessment 3 needed to be done on a small time limit and a lot of requirements needed to be fulfilled, so there were more team members responsible for software code during this part. Whereas for Assessment 4 the changes to the code that needed to be made was much smaller so less coders were needed. This could only be done because the flexibility of Scrum allows for team roles to be fluid. The Scrum Master and Product Owner remained the same throughout the project however.

Overall the roles that Scrum dictates were suited our team as we are a small team and iteratively adding functionality to a potentially shippable game is the most safe and efficient way to develop it, and allowed us to follow the requirements in a systematic way.

Bibliography

[1] I. Sommerville, Software engineering. Harlow, England: AddisonWesley, 2007.

[2] Mundra, A.; Misra, S.; Dhawale, C.A., "Practical Scrum Team: Way to Produce Successful and Quality Software," in Computational Science and Its Applications (ICCSA), 2013 13th International Conference on , vol., no., pp.119123, 2427 June 2013

[3] T. Preston-Warner, C. Wanstrath and P. Hyett, "GitHub Pages", GitHub Pages, 2008. [Online]. Available: http://github.io. [Accessed: 20- Apr- 2016].

[4] E. Costello, S. Butterfield, C. Henderson and S. Mourachov, "Slack: Be less busy", Slack, 2013. [Online]. Available: http://www.slack.com. [Accessed: 20- Apr- 2016].

[5] "Google Drive - Cloud Storage & File Backup for Photos, Docs & More", Google.com, 2012. [Online]. Available: http://www.google.com/drive/. [Accessed: 20- Apr- 2016].

Appendix

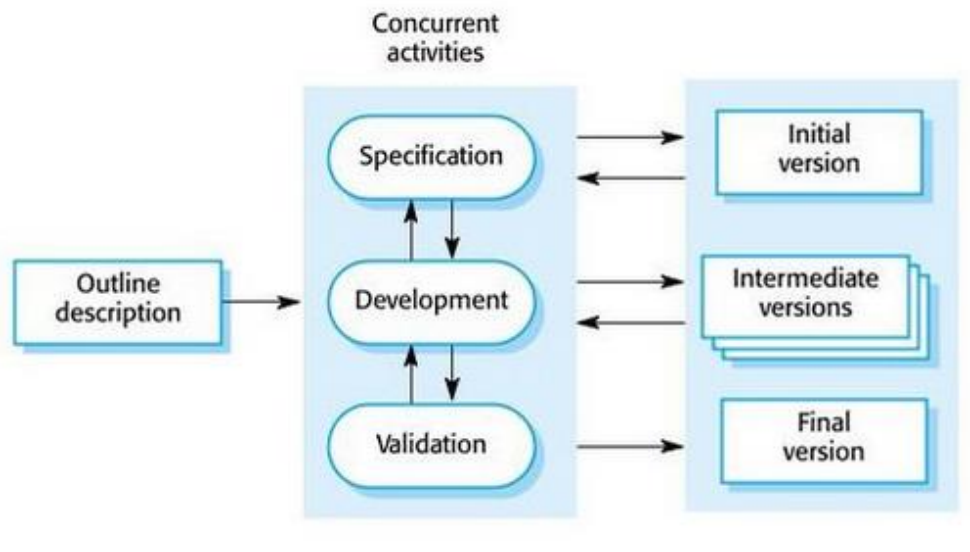

Fig 1. Diagram showing evolutionary approach. [1]