4. Method selection and planning

Group 11 - 11 Musketeers
Osama Azaz
Adam Dawtry
Tom Jackson
Brendan Liew
Holly Reed
Harry Ryan

Software engineering methods

- The team opted to use an agile methodology throughout the project in terms of how we planned to coordinate the process.
- As part of this agile technique we explored the Scrum technique which splits the tasks of the project into sprints which can then be followed up by a meeting to discuss the progress in the sprint and any concerns or problems found during the time period.
- The sprint size we used was a 2 week time frame, which we would begin and end with a meeting discussing the sprint that had passed and then the sprint we would begin.
- This approach method was chosen due to the short time frame involved and the smaller team size of 6 people meaning that work would have to be done in short bursts in order to hit the project deadline.
- Agile development allowed the team to focus more on the planning aspects to ensure the project ran smoothly with a high level of communication amongst the group which integrated especially well with the scrum approach.
- Although one strength of the Agile method is that the customer can be involved in the process and that the requirements of the project outcome can be changed throughout, this task had static requirements as set from the start. This allowed us to begin building the full functionality from the beginning and ask our customer for any questions or feedback during the weekly team sessions.
- This method also allowed us to update and evolve our plan throughout the project so that we could re-assess our approach to certain tasks given team feedback and current progress. This meant that whilst the plan was there to be followed it could be changed due to unforeseen circumstances or underestimation on the work load a task may require.

Development & Collaboration Tools

- For the implementation of the project requirements the following development tools were used:
    - Eclipse - Java IDE [1]
    - LibGDX - Java game development framework [2]
    - Piskel - Animated asset/sprite creation [3]
    - Gimp - Static asset/sprite creation [4]
    - Visual Studio Code - Website creation with the HTML extensions [5]
- For the development of models used in project planning the modelling tools used are shown below:
    - PlantUML - Universal modelling language tool [6]
- For collaboration and communication throughout the project the following software tools were utilised:
    - GitHub - Implementation sharing [7]
    - Google Drive - Document and file sharing [8]
    - Discord - Communication and meeting platform [9]

Fitness of the chosen Development & Collaboration Tools

- Eclipse was chosen as the main IDE for our project as it works especially well with our collaborative tool GitHub by allowing integration with our GitHub and allowing the pushing and pulling of the project to and from the repository.
- With eclipse integrating with GitHub this meant the rest of the team were able to feedback / integrate the code within their own tasks without the need of gathering the code from the

people working on the task, therefore speeding up the process in the sprints as constant communication at all times of day were not required.

- Initially we considered using another environment to develop our java code in like intellij however we opted to use eclipse due to experience using eclipse presented by team members and the easy integration of gitHub with this IDE meant it would integrate well with the agile development method as the project could be continuously improved over the course of multiple sprints.
- Piskel and Gimp were both chosen as the tools to create the game assets; both of these tools are free to use and easily available with Piskel being a free online editor and Gimp being a free downloadable piece of software. This meant it was easy to work on files and pass them around since the software required to work on them was readily available.
- Visual Studio Code was a choice made to implement the website due to its many extensions of which we used HTML extension and LiveServer. This meant the website could be changed regularly as our agile method demanded.
- PlantUML was the tool we decided to use in order to model the team's planning of the development process. It was particularly useful for creating the Gantt chart by which we modelled our teams plan of each task and their respective deadlines. We chose this due its open source nature and the ability to share links to models amongst the group and allow editing quickly and easily which is an aspect of the agile and scrum methods allowing change at the end of each sprint.
- An alternative to PlantUML was considered when it came to certain modelling such as the Gantt charts. Originally microsoft excel was thought of however the way of disturbing the chart via excel files which would need editing and not having something easily available for all was something that would have slowed down the process. Another alternative considered was GanntPro however without the paid version many aspects were limited and the free trial would not have covered the course of the project.
- GitHub was one of the collaborative tools mainly used alongside the website and code development sides of the project. It allowed multiple people to work on the same task at the same or different times without overwriting each other's work. Its integration across the IDE and the website meant updating our website with the latest snapshots of the development was done efficiently.
- Google Drive was another tool we found especially useful when it came to sharing files and documents that multiple team members may find useful or need to work on. It was useful when it came to the various meetings as it meant all team members could access and share feedback on the same files and any edits could be made and viewed by everyone working on that particular task.
- Google Drive allowed us to store everything in one place making collaboration a lot easier as files didn't have to be shared with other team members individually but instead stored for use throughout the entire project.
- Discord was the primary stage for communication amongst the group which is essential when using the scrum technique. The multiple chat rooms allowed meetings to be scheduled and all team members to be notified when they were scheduled and links to important references or information could be stored in another. All of our online team meetings were held in discord on top of this.
- Alternatives discussed for communication were zoom meetings and a whatsapp group chat however discord was chosen for its integration of both chat rooms and voice channels where screen share was available.

Approach to team organisation

- **Role Allocation**
    - Team organisation was initially approached by identifying the strengths and weaknesses of each team member as brought up by Ian Sommerville [10], noting areas to exploit their strengths and avoid any weak areas.
    - By exploring these strengths team members could then be allocated to certain tasks where they had significant experience in the role and an eagerness to complete that section.
    - Each task was made to have a minimum of two people allocated to it. One would be the lead on that section whilst the other was to be there as a backup or “shadow” team member and enable the lead to offload some work load should it be necessary.
- **Regular Scheduled Meetings & Sprints**
    - Meetings were scheduled on a weekly basis with an extra meeting held on a bi-weekly to end and initiate a new sprint process.
    - Sprints were set into stages of two weeks in which we could allocate work to be done from the previous sprint should it not be completed and add to the work already completed.

How this approach is helpful to the team and project

- By establishing roles amongst team members a group structure was created allowing a heightened focus on each task and making the project less daunting by distributing the workload fairly and equally.
- Furthermore the roles utilised peoples prior experience in other subjects and hobbies meaning the most effective people for the job were chosen and thus each task would be done to the best standard possible in the short time frame available.
- Having a “shadow” team member for each task ensured that the ‘Bus Factor’ was never one for any task which was especially important given the current circumstances of the coronavirus pandemic.
- Having this extra person per task also ensured that knowledge could be shared out so that should a member of the group be unable to carry out their task for some reason the task can be picked up by another team member and continued with minimal disruption as once again recommended in Software Engineering 9th edition [10].
- The regular meetings were also an essential part of the organisation process as they provided structure to the teams time management applying pressure on each task to have progressed since the last meeting.
- Meetings allowed team members to express any concerns and update the team on their progress so far outlining their estimates of completion time and if they would meet their deadline in the sprint.
- Meetings also allowed feedback and constructive criticism where team members on other tasks could express their opinions and views on how aspects of our project can be improved or add ideas on components to add / edit.
- Each sprint was helpful to the project progression especially in the implementation area as it allowed for us to build upon the already existing game by adding new features to reach the requirements and enhancing old features as suggested by feedback amongst team members to improve on the quality of the project.

The systematic plan

- Risk Assessment
    - Start Date - 25/11/2021
    - End Date - 1/12/2021
    - Dependencies - None.
    - Priorities - Ensure the risks have been identified and a mitigation plan is in place for each of them. The deliverable should be created and on the way to completion.
- Design & Architecture
    - Start Date - 28/11/2021
    - End Date - 8/12/2021
    - Dependencies - Customer Meeting to determine the requirements must have been had.
    - Priorities - The requirements must be tabulated in a way that they are easily readable, the game design and plan will be underway with particular attention to objectives and gameplay mechanics, the architecture of the teams software and development process also being noted. All 3 of these processes should have deliverables started on them during this time.
- Implementation
    - Start Date - 9/12/2021
    - End Date - 25/01/2022
    - Dependencies - Game Design and requirements have been decided and finished so that a full game model can be created.
    - Priorities - By the end of this time period there should be a fully functional game with most if not all of the elicited requirements listed in the requirements documentation.
- Art Work
    - Start Date - 9/12/2021
    - End Date - 25/01/2022
    - Dependencies - Game Design and requirements have been decided and finished so that it is known what sprites are needed.
    - Priorities - Boat and colleges are the main priorities as they should be large enough to see on the screen from afar and accessibility to those with colour vision deficiency is essential.
- Testing
    - Start Date - 26/01/2022
    - End Date - 1/02/2022
    - Dependencies - Implementation of code has been finished
    - Priorities - Identify any problems with gameplay or difficulty and adjust the code as necessary, make note of any requirements that have not been met either adding them into the game accordingly or taking note of them in the Implementation deliverable page.
- Documentation of Code
    - Start Date - 26/01/2022
    - End Date - 1/02/2022
    - Dependencies - Implementation of code has been finished
    - Priorities - Ensure the full code is fully documented with java docs
- Presentation
    - Start Date - 2/02/2022

    - End Date - 4/02/2022
    - Dependencies - Full project completion
    - Priorities - Have a 5 minute presentation prepared detailing the structure of the game code, the software used to develop the game and a demonstration of the game should be ready.

Evolution of the plan

- One of the main changes from the initial plan was that we wanted to do two versions of the implementation, presenting the first version to our customer and gathering feedback to implement in our second version. However this was quickly changed due to the time constraints over the Christmas holidays meaning that the implementation would not be started until spring term started on the 10th of Jan. This also meant that there would only be one testing period.
- Another change to our plan was that despite our efforts to ensure no task was done by only one person it happened that both team members assigned to producing the sprites became ill and couldn't work on the project for the beginning of spring term. This meant that although the implementation would have functionality by the deadline the sprite designs would be pushed back and need to be implemented past the deadline and during the testing phase. However, this was not essential when testing the requirements of the game's functionality.
- Initially the team also aspired to begin the design and architecture planning phase earlier than the aforementioned start date. However, due to clashes in the team members schedules and the customers availability we could not perform an initial meeting with the customer to find out requirements and present our ideas for them to discuss and feedback on. As this was a dependency for the beginning of our design phase this meant that the sprint in which that was included was slightly smaller than the others.

References

[1] Eclipse Foundation, Inc, “The community for open innovation and collaboration,” *Eclipse.org*. [Online]. Available: https://www.eclipse.org/. [Accessed: 31-Jan-2022].

[2] “LibGDX,” *libGDX*. [Online]. Available: https://libgdx.com/. [Accessed: 31-Jan-2022].

[3] “Piskel - Free online sprite editor,” *Piskelapp.com*. [Online]. Available: https://www.piskelapp.com/. [Accessed: 31-Jan-2022].

[4] “GIMP,” *GIMP*. [Online]. Available: https://www.gimp.org/. [Accessed: 31-Jan-2022].

[5] “Visual Studio Code - code editing. Redefined,” *Visualstudio.com*. [Online]. Available: https://code.visualstudio.com/. [Accessed: 31-Jan-2022].

[6] “Open-source tool that uses simple textual descriptions to draw beautiful UML diagrams,” *PlantUML.com*. [Online]. Available: https://plantuml.com/. [Accessed: 31-Jan-2022].

[7] GitHub. 2022. *GitHub: Where the world builds software*. [online] Available at: <https://github.com/> [Accessed 31 January 2022].

[8] “Meet Google Drive – One place for all your files,” *Google.com*. [Online]. Available: https://drive.google.com. [Accessed: 31-Jan-2022].

[9] “Discord,” *Discord*. [Online]. Available: https://discord.com/. [Accessed: 31-Jan-2022].

[10] I.S. Sommerville. *Software Engineering*. 9th ed. Pearson Education Limited 2011