Change Report

Cohort 1 Group 2

Yusuf Almahmeed
Jane Boag
Charmaine Chamapiwa
Charlie Coyne
Anupam Cunden
Bertie Kerry
Leo Xu-Latti

This page does not count toward the total page number

Processes, tools and conventions for managing changes

Once we had inherited the code and documentation from the other team, we first reviewed the new design brief and discussed the necessary changes. These changes were discussed in group meetings where ideas were brought up and the best ones selected which both satisfied the brief while still remaining realistic to complete and implement within the available timeframe.

We used an agile software engineering approach, mainly using the Scrum and Extreme Programming (XP) methods. We structured the work needing to be done into short sprints to keep the workload manageable and demonstrate regular progress being made. Each sprint was 1-2 weeks long and started with a group meeting where we discussed what tasks needed to be done and reviewed what had already been completed. These tasks were then allocated to team members, based on their skill set and preference. Early sprints focused on split focus and responsibilities between coding, and documentation, while later sprints required joint input, where the full group collaborated on documentation together. We also used XP practices as it allowed for test driven development, pair programming and continuous integration, all needed for implementing code changes. It also made sure that game features were fully complete before another was started, so issues were not found very late into the project. We did this by using unit and manual tests. Combining these two frameworks allowed our code to be easily adapted, while still maintaining traceability of all changes made.

We used GitHub for all code and documentation, following on from the previous group who also used GitHub. This allowed for easy collaboration as all members could work on separate aspects of the code and then share it to the main repository for review. GitHub's version history made it easy to track changes and updates, while ensuring that all team members had access to the most recent version of both the game and the website. We also used Google Docs and Google Drive for writing and storing all deliverables, as did the other team. Google Docs saves progress automatically and has version history, allowing us to look back over previous drafts and contribution history to review and keep track of changes made.

Weekly group meetings were held to review progress, reprioritise and reallocate tasks. This supported the continuous replanning approach of the Scrum framework. Communication was maintained between group meetings by WhatsApp to make problems known to all team members and quickly solved.

i. Requirements

Previous URL: https://eng1-group9.github.io/website/Req1.pdf
New URL: https://github.com/BertieKerry/ENG1-Website2/Req2.pdf

The updated requirements for this project were elicited from the new design brief. Unlike in the initial phase where details needed to be clarified though a client interview, the added requirements, leaderboard and achievements, were more specifically detailed in the new brief. In this phase, no client interview was conducted as the new design brief was sufficient enough to elicit the requirements from. The new design brief was discussed and analysed through group discussions, where team members shared their thoughts and came to a final decision that aligned with the new product brief but also considered accessibility and time constraints. The presentation of the requirements remained unchanged as it was already done clearly and well structured.

User Requirements

The user requirements were updated to reflect the changes in product brief and the new gameplay features.

The number of each of the game events were updated to reflect the new product brief. Instead of just one positive, negative and hidden event, the new brief calls for at least 3 positive, at least 5 negative and at least 3 hidden events in the game. This change was updated in the Req1 document and also in the User Requirements Referencing System under the UR_EVENTS row.

There were also 2 new user requirements added in line with the brief. They were:

- The game shall include a leaderboard displaying the top 5 scores.
- The game shall feature achievements visible during gameplay and after completion that affect the player's score positively or negatively.

The primary goal of the game was refined to highlight that now the player must escape the maze as quickly as possible to achieve the highest score possible. This change was made in line with the new product brief, where speed and efficiency now directly affect the player's success and if reflected in the score.

System Functional Requirements

There were 2 changes made to this section of the document which are the added features in the product brief, leaderboard and achievements. These additions reflect the updated gameplay elements. Aside from these, the rest of the functional system requirements remained unchanged as they reflected gameplay elements which are all still relevant and needed in the game, such as displaying the player's score and the pause functionality. No fundamental changes were necessary as all the previously established requirements were still required.

System Non-Functional Requirements

There were 3 main changes made to this section of the document. These changes were made as the original Req1 document did not address these features. These are:

- Added requirement for player movement. This guarantees that the user experience remains fully responsive and fluid and that the user goes through no unnecessary frustration caused by unresponsive controls for player movement.
- Updated how the game can be executed from just a Windows device to other OS systems and screens, such as Mac. This allows for diversity in end user environments as many users would have different operating systems. It also makes our game more widely accessible and will now appeal to a wider target audience.
- Updated the non-functional priorities to include maintainability which will help facilitate ongoing development and bug fixes. This will ensure efficient project progress and support the sustainability of the game.

Requirements Referencing System

There were 2 added new rows in the User Requirements Referencing System under the user requirements UR_ACHIEVEMENTS and UR_LEADERBOARD. Both are under the priority ‘shall’ as both are directly specified in the design brief.
2 of the existing requirements were refined to be more precise. These were UR_SCORE and UR_EVENTS. UR_SCORE now specifically includes the requirement for escaping ‘as fast as possible’ to highlight the aim of the game better. Similarly, UR_EVENTS was updated to specify that there should now be at least 5 negative events, at least 3 positive events and at least 3 hidden events.
No further changes were made to this system as it showed traceability between user and system requirements. Each user requirement has a clear ID and is linked with system requirements through the descriptions.This linkage allows for effective progress tracking and helps make sure all user requirements are met. As this traceability system was already well-established and sufficient, no changes were made to the presentation.

ii. Architecture

Previous URL: https://eng1-group9.github.io/website/Arch1.pdf
New URL: https://github.com/BertieKerry/ENG1-Website2/Arch2

System Structure:

Once we inherited the project, we reviewed the existing architecture to decide whether changing the architectural model was necessary. The inherited system was built using a closed-layered structure consisting of Presentation, Business, Persistence and Data layers. The system also had an ECS embedded in the business layer for game logic. This existing design already supported extensibility and maintainability. Due to the already supported extensibility, changing the architecture to our initial CMV model would have created a substantial amount of additional work without providing any clear benefit.

Design Process and Evolution:

For our design process, we focused on having a stable architectural model. In the early stages of our extension, we assessed whether new features such as scoreboard functionality and new events required structural changes. Ultimately we concluded that the current system works fine, however we just had to adapt to the new structure. As a result, the architecture remained unchanged, with developments occurring at the ECS level to introduce new entities and systems. We also considered reintroducing elements from our earlier CMV design that could improve performance, but then the group agreed that it was unnecessary since the current system complemented extensibility well enough.

Closed-Layered Architecture:

Since there were no issues with the closed-layered structure, the original responsibilities remained the same. Presentation remained focused on UI and rendering, Business handled ECS logic, Persistence managed interactions with system services and configuration, and Data stored assets. This preserved the original team's separation-of-concerns principles without any complex changes. Maintaining this structure allowed the game to still meet the NFR_RESILIENCE, NFR_RELIABILITY requirements, which the original project satisfied.

Design Decisions:

In terms of the code itself, we added new entities, systems and features while still adhering to the organisational layout of the inherited code structure. This allowed us to efficiently extend the game while also maintaining consistency throughout the code and documentation sections.These extensions included scoreboard, hindering events, and expanded map areas. Such events link back to the FR_SCORE_CALC, FR_EVENTS, and FR_MAP system requirements.

Languages and Tools

As for the development, the game was extended in Java using the LibGDX game engine, satisfying the same CR_ENGINE requirement. The decision to remain with LibGDX was made because no new requirements introduced functionality that would justify changing the engine. In addition, the team was already familiar with LibGDX from our original project, which allowed development to continue smoothly without needing to learn and integrate a new engine. Moreover, we continued using Gradle for project management to maintain consistency with the project toolchain.

Changes to Assessment 1 Deliverables

Overall, the architectural diagram inherited from the original team required minimal changes. The layered architecture diagram remained unchanged as the system boundaries and interactions were still valid. Updates were limited to extending the ECS diagram to include new entities and systems introduced for Assessment 2.

No further changes were necessary, as the initial architectural structure was sufficient for the additional product requirements.

iii. Method selection and Planning

Previous URL: https://eng1-group9.github.io/website/Plan1.pdf
New URL: https://github.com/BertieKerry/ENG1-Website2/Plan2

The agile methodology adopted by the previous team, using both the Scrum and XP methodologies was continued by our group. There was no change to the method selection as it was appropriate for the project and was already embedded in the inherited code and documentation. This ensured continuity across both code and documentation. It also was the same methodologies our group had chosen for our initial project, meaning all team members were already familiar with the processes needing to be done. It would have been more time consuming and less efficient if we had decided to learn and adopt a new methodology at this stage with no added benefit to the project.

The tools used mainly remained the same. Our group continued to use Google Docs and Google Drive as the chosen documentation tools for the same reasons the previous group had chosen them. These tools were already effective and supported collaboration and traceability of changes through version history. We had also used these tools for the first half of the project so all team members were very familiar with the systems.

There was a slight change in the communication tools used. The previous team used WhatsApp and Discord for communication, whereas our group only used WhatsApp. This is because some team members were unfamiliar with the Discord platform, but all were familiar with and comfortable to use WhatsApp. This decision was made to avoid making members use a new app for communication. It removed confusion while still allowing for effective communication.

No changes were made to version control as GitHub continued to be used. Team members had already familiarised themselves with this platform having used it for the first half of this project, or before in previous projects. GitHub was continued to be used for tracking code and documentation changes and for its complete history of updates, preserving traceability.

The approach to organisation remained largely unchanged. We still had in person weekly meetings to review progress and reassess and reallocate tasks where necessary, as the previous group had done. However, there was a slight change, as unlike the previous team, we did not create a new Gantt chart each week. Instead progress and planning updates were recorded by updated planning tables. At the start of each week, the tasks in the planning table were re-evaluated, and a new table for that current week was created. This took into account the backlog from incompleted tasks from the previous weeks and included new tasks.

The main section of this document updated was the project plan. The plan received by the previous team covered weeks 1-6 only, corresponding to their time working on the project. As our group took over this project from week 8, the existing plan was of no use to us. To fix this issue, we created our own plan at the start of week 8 to cover our time working on the project. Rather than extending the existing Gantt chart, we introduced a structured planning table that included:

- Week to complete task
- Sprint goals

- Planned tasks
- Responsible team members.

This table based approach was chosen as it aligned with the Scrum sprint planning where goals and backlog are reviewed weekly. The table format was also easier to update or create a new version of when the plan changes, showing the adaptive nature of the agile methodology. The sprint goal was removed as a column in the weekly updated planning tables, and instead, placed as a short statement above each table. This was done to improve clarity as the tasks in the initial planning table were further split into more rows in the table. As the weekly sprint goal would've been repeated for each task, it was placed outside the table to remove unnecessary repetition and increase clarity of the planning tables. In addition to supporting sprint planning, the table format allowed tasks to be split up into smaller chunks, allowing for different team members to be responsible for smaller aspects of the project. This helped keep the workload manageable and ensured all team members contributed fairly to the project.

iv. Risk assessment and mitigation

Previous URL: https://eng1-group9.github.io/website/Risk1.pdf
New URL: https://github.com/BertieKerry/ENG1-Website2/Risk2

Risk Management Process

No changes were made to the risk identification process carried out by the previous team. Risks continued to be identified collectively, through discussions during group meetings, using the assessment brief and client preferences again, as a baseline. This approach remained the same as it was effective in finding a thorough set of risks that could arise during our time working on this project. It also allowed for consistency and continuity throughout documentation. The original set of risks previously identified remained relevant after our group took over the project, so there was no need to make any significant changes to the risk management process. Altering this process would have been unnecessary and not a good use of our time within the project's limited timeframe.

No changes were made to the overall risk analysis or planning processes. Although as a team, we decided that the risks would all remain the same as all were still relevant, we re-evaluated the likelihood and severity ratings to ensure they were still accurate. We came to the decision that these should remain unchanged, as the brief and context hadn't changed substantially and the risks were all still as likely to happen and impactful as before. The only adjustment in this area was the reassignment of the risk ownership, reflecting the change in team members responsible for monitoring and mitigating these risks.

Risk Register

Changes were made to the risk register in the ID structure, likelihood, severity and ownership columns, relating to 5b in the deliverables. However, no new risks were added and no existing risks were removed from the document as previously identified risks were still relevant to the continuing development of the project. The original risk register only used Project (P) and Technical (T) identifiers. This was expanded to include People&Project (P&P), Product (PR) and Technology (T). This change was done to make the risk set more thorough, and cover broader aspects of the project. It also allowed for risks to be more accurately categorised to reflect their impact and helped traceability and understanding. For example, the risk previously identified under TP1 relating to team members getting sick or falling behind is now identified under P&P1 as it concerns people related issues that directly impact the project.

The ownership of risks column was updated to reflect the new team members who have taken over responsibility for these risks, after project handover. This was done to ensure accountability and make sure if these risks were to occur, it was clear who was responsible for implementing mitigation strategies and monitoring the situation.

Although the likelihood and severity ratings of the risks remained unchanged, the appearance of these columns in the risk register was updated. Colour coding was used to show whether the risk was low (green), moderate (orange) or high (red) likelihood/severity. This change improves the readability of the risk register, as it is easier to see the importance

of the risks, and also is consistent with the format of a standard risk register. The colour coding also helped us as a team focus attention towards the higher risk items during reviews.