Reflection Report on Software Eng

Team #11, Mac-AR
Student 1 Matthew Collard
Student 2 Sam Gorman
Student 3 Ethan Kannampuzha
Student 4 Kieran Gara

This document reflects on the work done by the MacAR team. It outlines changes made due to feedback form external stakeholders, how the product developed over time, and how the team would have approached the project differently with the knowledge that we had at the end.

1 Changes in Response to Feedback

Weekly meetings were held with the project supervisor to assess the progress of the project. As such, constant feedback was received which was implemented in increments. Conversely, feedback from the TA and instructor were implemented in large chunks at the end of each deliverable.

1.1 SRS and Hazard Analysis

In terms of supervisor feedback, the requirements that defined the project underwent the most changes. Initially, the scope of the project was very vague and it was up to the team to further define the project. The only hard requirements were that the product has to utilize AR and be collaborative. As a result, the team outlined a set of basic networking requirements, along with very vague requirements for the puzzles that the users would be interacting with. As a response to this revision 0, the supervisor provided a points of feedback. In addition to general points like increasing usability and performance, the main point of feedback was that each of the puzzles should be testing a different aspect of the users skill set (memory, kinetic motion, etc). This feedback is what led to the development of the majority of new requirements for the Rev 1 SRS, primarily the requirements revolving around concrete puzzles.

TA and instructor feedback informed us less on details for the content of the product, and more on technical changes that we would have to apply. For the SRS, it was clear that our inexperience with this type of app caused us to be too vague. Feedback pointed out that we were missing crucial details, especially on the networking side (like specific requirements for room sizes and device

specifications). This feedback allowed us to form a more concrete definition of the requirements of how multiple users would interact with the system, which greatly helped our development for both Rev 0 and Rev 1. The Hazard Analysis had less concrete feedback, with most of it being related to syntax errors. Overall the combined feedback from the supervisor, TA, and instructor also allowed us to narrow our focus, getting rid of irrelevant requirements, such as the save game requirements, in favour of focusing on what actually mattered for the project.

1.2 Design and Design Documentation

The core structure of the design remained remarkably similar throughout the various waves of feedback, with the major changes coming in the form of the responsibilities of each module and how they presented themselves. The one major change to the structure of the module hierarchy, similar to the SRS and Hazard Analysis, came in the form of narrowing down the scope. Originally more puzzles were planned to be added. However, both the supervisor and the TA/ instructor pointed out that the existing content after Rev 0 was enough, and it would be a better use of our time to make sure that content was refined rather than adding more. Due to this, our design ended with the 5 puzzles at the end of Rev 1.

For the actual design, after the Rev 0 supervisor feedback it was decided that the game should have a more natural flow. This caused the Multiplayer Puzzle Module to expand in scope, which allowed for multiple puzzles to be spawned around the users in different locations. Overall the design was considerably more complex than Rev 0, but allowed for a much better user experience.

For the design documents, it was pointed out by the TA how the design was inefficient. Generalizing functions, turning local members into state members, and generally cleaning up the design of the modules allowed for a project that was both much easier to work with, and able to be expanded on in the future.

1.3 VnV Plan and Report

The VnV documents had a unique set of challenges that caused our Rev 0 version to be inadequate, and TA and instructor feedback was very helpful with getting it back on track. One of the main pieces of feedback was a lack of specificity, with no specific figures referenced and only the average results being listed. This happened due to a lack of extensive testing. For Rev 1, we went back and redid all of the performance, usability, and similar test metrics to allow us the ability to provide more concrete results.

2 Design Iteration (LO11)

From the beginning of the project, our team knew we were going to make a AR Escape Room that involved a series of puzzles that groups of players would solve. These puzzles would require communication amongst players in order

to facilitate social interaction. Initially, we had planned to have 7 puzzles, as well as features for saving/loading game sessions so that individuals could start a game and finish it later. As we progressed through the term, however, our design became more focused on making a game with less puzzles so that users could complete the game in one session. Since the application was going to be used in Dr. Yuan's study for analyzing social interaction, this made more sense. 2 puzzles ended up being scrapped, resulting in our application having 5 puzzles, and the save/load game feature was scrapped. Furthermore, in the beginning we did not have a theme for our escape room but as time progressed the group decided on making the theme related to escaping prison. The aesthetics/graphics of the game were made darker to mimic the imagery of a prison. Additionally, game mechanics such as joining lobbies were updated based on usability feedback in order to make them more intuitive.

3 Design Decisions (LO12)

The biggest constraint on our project was the time constraint of finishing this project in 8 months while balancing work given by other courses. This time constraint resulted in certain features being scrapped as well as having less time to spend polishing graphics and game mechanics. Things such as having aesthetically pleasing UI elements were not a priority as the most important things to have completed in the short time were the fundamental game mechanics. Additionally, one large constraint given to us by our supervisor was that the game had involve AR elements. This was a very big constraint as it made our designs for puzzle focus on being able to be used in an AR environment, and positioned properly in the users environment. Furthermore, our supervisor also gave us the constraint that the application must support social interaction. Because of this, our group made design decisions to have voice and text communication available through the app in order to allow for users to communicate with one another.

4 Economic Considerations (LO23)

The current iteration of the product purely exists for the study being performed by the teams supervisor, and as such only contains the essentials for that purpose. That being said, through constant feedback from within the team and from the TA/ instructor, the product was designed to be easily expandable. As such it wouldn't be a large leap with a bit more time to flesh out and market as a procedurally generated AR puzzle game that could be played for fun rather than just for the one study. The main consideration for a widely distributed version is the fact that currently, all of the networking technology used in the app is free, and as such has a low allowed bandwidth. For the app to be able to be used by more than 1000 users/month, the various technological considerations are as follows:

- Relay (Client/ Server Communication): Roughly $0.25 per average concurrent user
- Lobbies: $0.09 per GiB
- Vivox (Voice/ Text Communication): Roughly $0.4 per user

These values can vary depending on how peak concurrent users there are in a month, but at face value we can assume that the cost to run the app would be roughly $0.74 per user per month. One option would be to charge users $1 per month to use the app, but more than likely the only feasible way to market a simple product like this would be the more standard model for apps of a one time purchase. In this case we could charge $10 for the product, which would ensure each user pays for over a year of networking costs up front. This is a fairly steep price for a phone app, and barring implementing micro-transactions, it would be more feasible to look into what other networking technology exists. The primary economic consideration up to this point was ensuring that everything the team used was free, so not much research was done into the most cost effective paid technologies. A more in depth study would need to be conducted, taking into account the expected number of users every month.

5 Reflection on Project Management (LO24)

Project management was a very important part of coordinating completion of deliverables and features of the application. The main project managements tools used were through Git. Git issue tracker was used to keep track of upcoming deliverables as well as TA and peer review feedback. The Git project boards were used to break down the aforementioned tickets into clear groups so that work could be easily split between team members and tracked. Tags were similarly used, with workflows set up to automatically assign issues with certain tags to specific boards.

5.1 How Does Your Project Management Compare to Your Development Plan

The development plan was, predictably, slightly more ideal that the development in reality. This was most apparent in the project scheduling section of the plan. The outline was to have deliverables done well in advance of their due date, with a google calendar being used to maintain a strict schedule. This wasn't feasible. The combination of technical problems, other classwork, and just general expectations from our everyday lives meant that deliverables often couldn't be fully completed until the day they needed to be handed in. Similarly, maintaining a strict schedule would have only served to hamper the group, and as such members were allowed to work on their own time as long as they met their deadlines. Similarly, the team member roles ended up being largely inaccurate. Rather than having a clearly defined "NFR Lead", "Front-End

Developer", etc, everyone instead found a niche in a certain part of the codebase and took on all of the responsibilities that the part required. As such, every member of the team was more or less responsible for the same general things, just for their own section.

Some parts of the development plan were accurate though. The technologies used remained relatively unchanged throughout the project. The coding standards, barring a few cleanup tasks, were followed to keep the codebase consistent. As the team got more and more comfortable with Git issues and workflows, we were able to follow the workflow plan very closely. The team communicated well following the team communication plan. Overall, although a few parts of the development plan were overambitious, the team followed it well where it mattered.

5.2 What Went Well?

A lot went well, but the core reason why everything was able to go so smoothly was because of teamwork. The team was in near constant communication throughout the project, using discord to coordinate and discuss work. As a lot of the AR and networking technologies were new to everyone on the team, no one hesitated to ask for help which allowed us to tackle big problems as a group. The team cleanly coordinated different modules for everyone to work on, ensuring that no work would conflict where possible. Git issues and boards were updated so that everyone could see the flow of work being done and adjust accordingly if any parts got held up.

Team dynamics aside, the technological side of things also went surprisingly well despite how new the technology used was to the team. We were able to combine what we've learned up to this point with tutorials and documentation online to teach ourselves how to best leverage Unity and its built in libraries. As a result, we were able to create a good, stable product.

5.3 What Went Wrong?

Although we were able to teach ourselves about the new technologies, there was a significant ramp up time to reach the point the team was at the end of Rev 1. Due to this, time and scope were the largest issues the team ran into. The original idea for the app was much more ambitious than the final product. In the end, certain features like the save/ load game, randomized puzzle generation, more complex assets, etc, had to be scrapped in favour of the core functionality. This, along with the time it took to perfect our project management system through git and get used to working together, meant that not having enough time was the largest restricting factor.

5.4 What Would you Do Differently Next Time?

If we were to do this, or a similar project again, then the main change would probably be to have a more focused scope. The fact that the direction was

a bit unclear at the beginning of the project, coupled with our expectations being too high, led to the team focusing on aspects of the project that were less important. This fed into the time management issues mentioned in section 5.3. During the SRS stage, more work would be put into defining which requirements were high priority and which were just nice to have, then expanding the project from there.