3307 Group Project - Final Documentation

Project R.E.M.

December 7, 2018

Group 15

Claudiu Cretu

Damien Dorobek

John Fischer

Brayden Horth

Jordan Nesbitt

**Project Summary**

Initially, we had set out to create a single player top-down wave-based shooting game in C++ to run on the Raspberry Pi. We titled our game R.E.M., it revolves around a young boy, Simon, who fights monsters from hell in his nightmares. R.E.M. features ten waves for the player to traverse through, with each wave spawning more enemies than the last. At the tenth wave, the player must defeat a boss in order to beat the game.

At the beginning stages of development, our main goal was to create a map and give the player the ability to move. Once we coded this functionality, the next step was to implement player shooting and along with this, multiple gun types for the player to use. We then implemented three various enemy types. The first enemy type was a bat that was slightly faster than the player with low health, the second was a goat that was a bit slower than the player with moderate health, and the third was a dragon that was even slower than the goat with high health. Once the enemy movement was coded, we had to ensure that colliding with the player caused the player’s health to decrease and that the enemy’s health decreased when the player’s bullet collided with them. In addition, we wanted the player to gain health and currency upon defeating enemies so they can purchase items in the shop.

The next step was to implement all the menus, including the main menu, the shop menu, and the game over menus. The main menu was implemented first, which gives the player the option to start the game, quit the game, and read the instructions. Then, we created the shop menu which gives the player the option to purchase health, three guns, or to leave the shop. We made it so the three guns are randomized each time the player accesses the shop and made the shop accessible only between waves. Finally, we implemented the game over screen which would be displayed if the player died before completing the ten waves.

To round out our game, we added a final boss at the beginning of the tenth wave that shoots at the player similarly to how the shotgun works. If the player defeats the boss, another screen is displayed informing them that they have won. Once all of these tasks were completed, we added a story to add some depth to our game and balanced out our game so it was not too easy or too hard to beat.

**Key Accomplishments**

Throughout the development of our project we achieved many accomplishments that worked to our benefit. Teamwork was a key aspect of this entire project, we were constantly communicating as a group online and we would consistently schedule meetups to work together. From the very beginning we started connecting as a cohesive team, this is evident in our initial proposal and brainstorming which was received well. Through our communication and teamwork we built strong relationships that helped us work together more efficiently. Because of the strong relationships that we formed, we were able to modularize the group into different roles based of our individual strengths.

All of our members contributed to the project to some extent, each of these contributions are a testament to the previous experiences that we all brought going into this project. These experiences, whether personal projects, work from other courses, or online tutorials, proved to be extremely useful in the development of this project. For example , we used modularization to separate large components of code into functions and we used inheritance to classify game objects under an Entity class to make processes like collision easier. These were both topics that we learnt from previous course work.

**Key Problem Areas**

The programming of our project got off to a tough start. We tried as a group to use two different game engines, both of these failed to meet our needs of useability. We then decided to work without a game engine and instead use a graphics library. The first graphics library we tried was not successful and used much of our time, but finally after deciding on SFML we were ready to being coding. This process was extremely taxing on our limited time, if we were to be presented with the situation again we would individually carry out “spikes”. Data protection of objects was not a high priority in our project, this did not turn out to be a problem since our program is so small and does not rely on other software. However, this does not mean it was not an issues. To resolve this we should plan our classes better and include “setters” and “getters” for each data member that needs to be accessed.

Bitbucket is a useful way to ensure that projects with multiple developers remain in sync; but due to the learning curve of using bitbucket we shied away from this and relied on google docs to share code. This lead to some technical challenges while developing the project and therefore in the future it would be beneficial to learn how to properly use the bitbucket interface.

We initially approached the project using the Big Documentation Up Front (BDUF) approach. We then transitioned into using a dynamic planning style of stories and were able to make more effective use of our time and resources in developing our software. We initially thought like BDUF since it was easier to conceptualize the project as we already have experience thinking like this. In the future adopting a more dynamic approach from the beginning would save time and resources on development.

**Lessons Learned**

The key takeaways from the project were learning how to develop software as a team and accomplish our goals. As a team, we learned how to understand each group members way of designing and implementing code, and bringing all of our code together for the program to work. A big takeaway from this project was time management. Given the size of our project, we learned how to manage our time effectively during each stage of our project. Given the amount of time we had, certain aspects and ideas from our project have to be removed or changed in order to save time. Lastly, our group learned that when building a large project, many unexpected problems and errors arise that will inevitably slow development.

Some of the things our group would do differently is effectively use “spikes”. This would allow each member of our group to do research on their own. This would save us time when trying to discover what technologies will work with our project. Some of the things our group would keep the same is our groups brainstorming and creativity. Our group was able to implement many of our ideas into our project, and overall motivated us to make our project even better than we thought.

Practices our group developed as a part of our project was assigning different aspects of our project to each member. Group members were assigned to things such as game menu implementation, enemy and player movement, character and map design, and weapons. This was beneficial because when a problem or error occurred in any of those functions, we knew which group member would be able to solve it in the shortest time possible.

Areas that could have been improved was our groups time management. It always seemed that during the final few hours of a stage project, we were rushing and solving our biggest problems at the last moment. To improve this, our group should have created more group meetings and set aside more time for this project. More time ultimately helps identify any new problems that could occur in the future and that extra time helps solve these problems before we reach the final hours of developing our project.

As a group, we would work on a similar project like this in the future. This project gave us experience on how to function within a software team and taught us the many problems that can happen when developing a large project. If we did a similar project again in the future, our group would be more capable of creating the processes required when building a large project and being able to overcome the many obstacles that are encountered when building a large project.