Block Minigame Plugin

COSC412 Individual project

John Mulholland

Professor Broadwater

03/31/2021

Revision Sheet

03/16/2021: First write up.

03/31/2021: Revisions for Professor Broadwaters comments.

Preface

The purpose of this plugin is to create a fun minigame for users to participate in. The plugin revolves around the idea of the players in the minecraft server chasing around a randomly assigned block within a five minute window. If a player fails to reach the block in the timeframe, the player will die, the last player left alive is the winner.

The plugin will complete the purpose above, it will be tested for bugs and if bugs are found they will be dealt with swiftly. There should not be anything else that is not related to this plugin added in.

Table of Contents

1.1 Project Overview

The purpose of this plugin is to create a fun minigame for users to participate in. The plugin revolves around the idea of the players in the minecraft server chasing around a randomly assigned block within a five minute window. If a player fails to reach the block in the timeframe, the player will die, the last player left alive is the winner.

The plugin will complete the purpose above, it will be tested for bugs and if bugs are found they will be dealt with swiftly. There should not be anything else that is not related to this plugin added in.

The plugin will be coded through the use of Eclipse, Bukkit, and Minecraft.

1.2 Project Deliverables

The following deliverables will be provided.

1. Software Project Management Plan (this document)
2. Software Test Plan (STP)
3. Software Quality Assurance Plan (SQAP)
4. Technical Documents and Software
   - Software Requirements Specification (SRS)
   - Software User Documentation
   - Source Code (including installation and configuration instructions).
5. Block Minigame Plugin.
6. Server Installation Instructions

1.3 Evolution of the SPMP

The plugin is scheduled to have a working prototype completed by Mid-May. Unscheduled changes due to a change in clientele/play testers needs will be documented via the SPMP and changes in Use Cases/Sequence Diagrams. Once documented changes will be made by the developer.

1.4 Reference Materials

1.5 Definitions and Acronyms

Bukkit: The plugin software that builds the plugin for the server.

Plugin: A file that changes the games original files.

Server: Locally ran server that houses the plugin and will be used for testing the plugin.

Eclipse: Java IDE used to code the plugin.

Minecraft: Video game where the base code will come from.

Developer: John Mulholland, both a client and developer.

2.1 Process Model

This project was started on March 15th, 2021 and will be completed by the end of the semester. Some major milestones are completing the Use cases and Sequence diagrams on Mar 16, 2021, Design Documents on Mar 17, 2021, and SPMP document by Mar 18, 2021.

The project will use object-oriented code and outside resources for development. The project will be completed by an individual programmer. There will be a certain number of goals to complete which will be outlined in the SPMP. Updates will be given to the play testers when major patches are completed. The group files and documents will all be uploaded using GitHub and coded in Java. There will be properly named repositories for each corresponding file and document.

2.2 Organizational Structure:

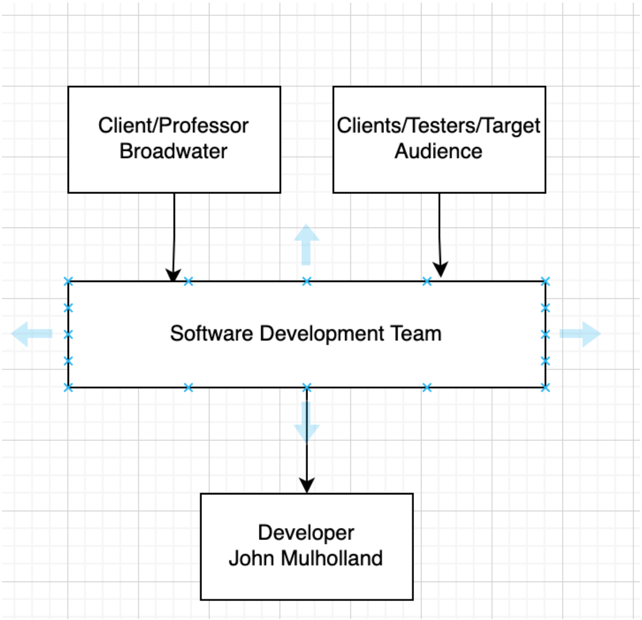

Figure 1: Organizational Structure

2.3 Organizational Interfaces

There will be minimal communication between any other entity besides the play testers and the developers in this current assignment. This is due to the lack of foreseeable need for a client base. However, the team will be attempting to connect the databases to other forms of frontend and backend software.

2.4 Project Responsibilities

John Mulholland(Developer) will be responsible for all events/actions that occur during the testing and creation process of this plugin. Once the game has begun working, bug hunting will be the biggest priority, followed by adding additional functionality if possible.The project will be ready for play testing and presentation before May 7th. All funding will come from John Mulholland and will be evaluated when funding is deemed necessary.

3.1 Management Objectives and Priorities

Our main goal is to create fully functional software that will accurately meet our play tester's needs.. If the need of third party software arises, we will want to focus on keeping it cheap, and using it sparingly.

3.2 Assumptions, Dependencies and Constraints

Assumptions: We assume that the developer has very little experience in development, therefore, patience is key for each member of the team.

Dependencies: We are dependent on the play testers's wants; making sure that the developer updates the team as needed. In addition, our team is dependent on the guidance of online information in order to finish this plugin.

Constraints: I am constrained by my classloads, jobs, and the lack of budget.

3.3 Risk Management

In the event of

- Lack of funding: The project is free to complete and will be completed
- The project is too large: In the event the scope is too difficult to complete, I will complete the project to the best of my ability and attempt to complete it past the presentation time.
- Requirements creep: If the requirements become too much, I will decide which requirements are considered vital to project completion and complete them in the order.
- Staff: This is an individual project and as such no staff will be hired, and there will be no danger of individuals leaving the project.

- Client Disapproval: If the client/play testers do not like this product their opinion will be noted.

3.4 Staffing Plan

There will be greater than 2 play testers and one developer. In the event the developer falls behind, an incomplete submission will be considered. In the event of a lack of play testers the developer will have to playtest using empty accounts.

3.5 Monitoring and Controlling Mechanism

The developer will be incharge of keeping track of time. In the event he falls behind the schedule implemented in the SPMP the developer will need to consider submitting an incomplete plugin.

4.1 Methods, Tools and Techniques

Playtesters will be used to check if the plugin is working properly. However, the scope of the plugin will be mostly at the discretion of the developer.

4.2 Software Documentation

There will be inline comments and any necessary changes will be committed in the SPMP. The files will be stored locally during development and will be stored in github for production.

4.3 Project Support Functions

The majority of the education in plugin design will come from wikis, youtube videos, and other internet sources. The use of minecraft and locally run servers will also be used.

5.1 Work Breakdown Structure

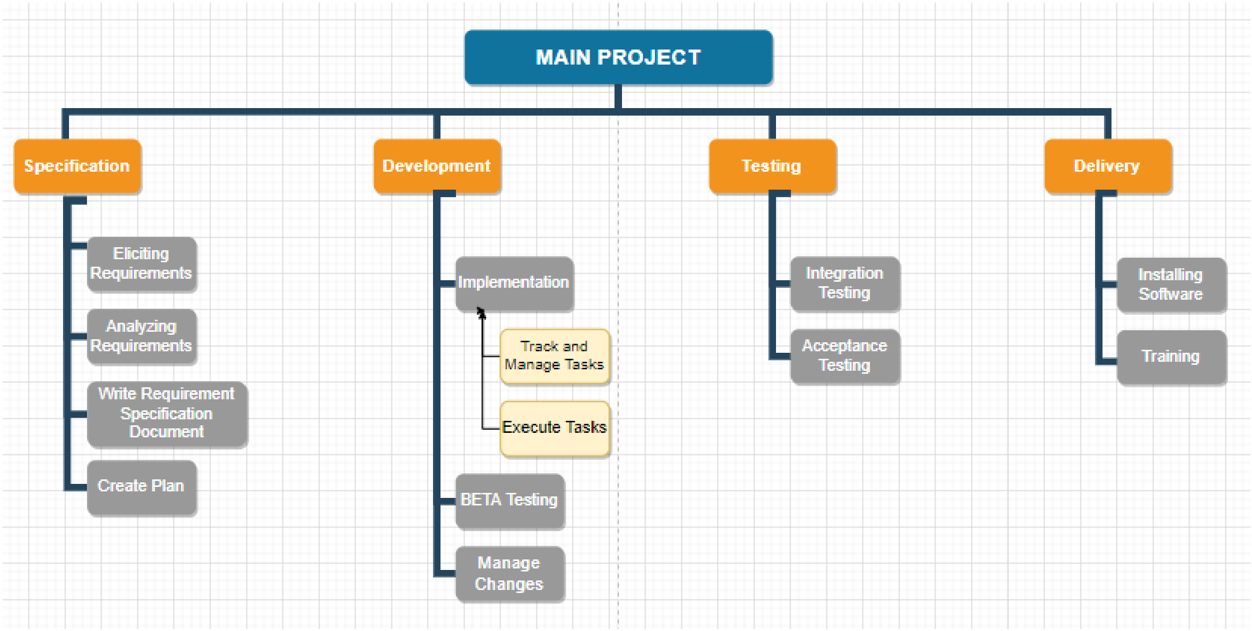

Figure X Work Breakdown Structure

5.b Dependencies Between Task

There will be very few dependencies between tasks due to the solo developer in the project. However, if the need arises, pushing to github will be used to suit the need.