CMPT 370 Requirements Document for BattleBots

Mackenzie Power

Haotian Ma

Ryan Tetland

Will Revell

Mitchel Kovacs

Table of Contents

Premise

This document is the requirements contract between the student software engineers Mackenzie, Mitch, Will, Haotian, Ryan and our client Chris Dutchyn. We will be developing a robot battle game for our client hereby referred to as BattleBots. Everything contained within this document is our promise of what will be included in the final product. We will prioritize what is necessary in order for the game to run properly and discuss what features we could add to make the game better and more fun to play. Included in this document will be a detailed description of all actors, actions, scenarios, use case diagrams, sequence diagrams and prototype interfaces to help our client gain a better understanding of the game.

Product

The finished program (BattleBots) will allow users to play against other players, computer controlled robots, or run a simulation of a robotic battle between different robot teams. When the user chooses to play against other players or computers they will be able to visually see a board where they will take turns with their opponents. Each play will involve a series of rounds where each player is given the ability to move a specific robot and attack anything within that robots' range. All these movements are limited by the characteristics that specific robot. As the game goes on, players will be eliminated when they have no robot to make a move. The game ends when only one player has robots remaining. All robots will have their win and loss stats updated based on the outcome of the game. If the user chooses run a game with no human players then a simulation is run instead, showing the result of the simulation and an updated version of each robot team's stats. In order to run and complete this program, we will need to utilize several different resources, some of which are used to complete this document.

Platform and Software

BattleBots will be designed on a Linux based system and be executable using a remote access through tuxworld. All team collaboration will be done through a GIT repository accessible to the client. Most coding will be done in JAVA using eclipse and NetBeans. Coding for robot artificial intelligence will be written in Forth and will be utilized in JAVA using JSON. The use case diagrams are created in Microsoft PowerPoint and the sequence diagrams are created using LibreOffice Draw. All formal documents will be created using Kate and Microsoft Word. The next section of this document will outline the various actors (roles) that will be involved in the program.

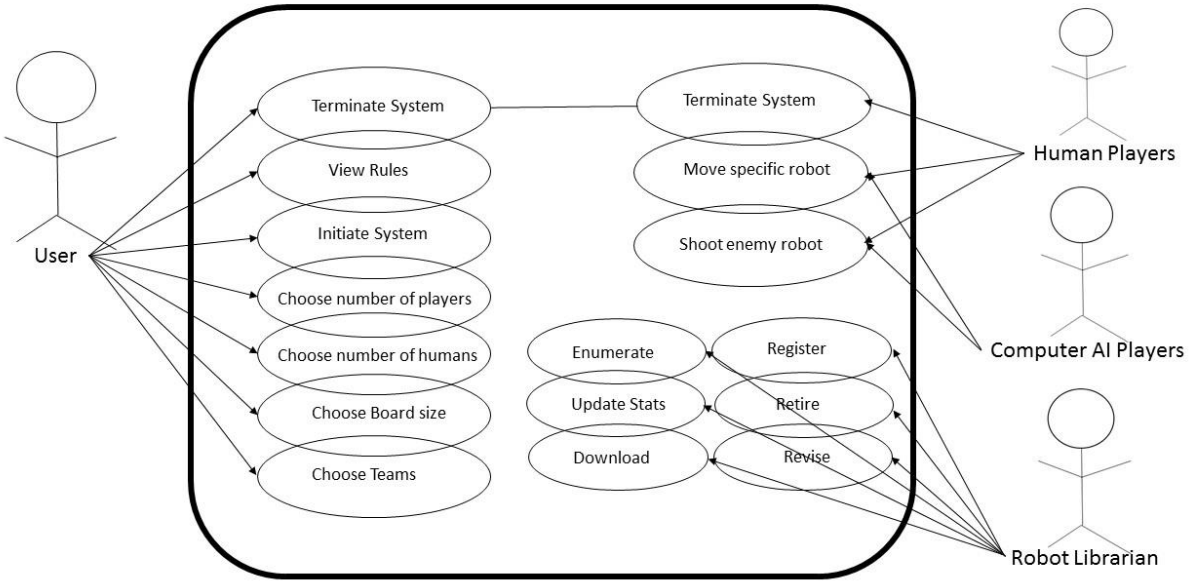

Figure 1. Use-Case Diagram for the program that we will be designing. It shows the four actors and their specific actions which will be described in detail down below.

Actors

For the project, there will be four main actors (or roles) that will be employed within this program. They are the user, the human players, the artificial intelligence robots that will play against human players and other robot players, and the robot librarian.

User

The first actor in the system will be the actual user, the individual that will be executing the program and dealing with the various user interfaces. This actor will be able view the rules of the game at any time; terminate the program at any time; choose the number of players; choose the number of human players; also choose the size of the board; view statistics after the game; and play the game again. The only limitations for these actions is the player will only be able to choose 2, 3, and 6 players for the game; and that the size of the board is only an option to choose from if there are 3 players.

Human Player(s)

The next actor in the system will be the human players involved in playing the game. Note: any individual who is fulfilling the user role can also assume the role of a human player. Human players have the ability to view the rules section, move a robot, attack a robot, and terminate the game.

Robot Player(s)

Like the human players, the robot players will have a similar role. They will have the option to move (if it is their turn) and attack another robot.

Robot Librarian

The last actor in the system will be the robot librarian whose role will be to query various robot teams and their stats for the user to select from; download the robot AIs for use; register new robots; retire current robots (clears their stats); revise current robots; and finally keep and update the statistics of each robot it has access to after a game.

Actions

Each of the previous actors listed above will have a set of actions they will be able to do. The importance of these actions will be stated below. In addition, a description of each action will be provided along with any conditions that apply to them. If there are any alternative scenarios associated with a particular action they will be included as well. The actions that will be first described will be those deemed as "Must Have" for this program.

Must Have

The "Must Have" section will list actions that are integral to the functionality of the game. It will not include all of the features that we would like to implement, which is outlined in the following section, but rather the features that will be required to allow the game to be playable.

Initiate System

In order for the game to be played, the user must have a way to actually initialize the game. Though this is a relatively trivial matter, it is important to take note since without being

able to initialize the game the entire program is useless. The program must be executable through tuxworld.

View Rules

The user must be able to click on a button labelled "Help" which would display a screen that explains all of the rules of the game and comments for troubleshooting. From there, the user should be able to click a “Back” button to return to the previous screen. It is important for any user to be able to view the rules and troubleshooting page at key points when it is convenient. In order to access the rules page, the user must either be in the initial main menu screen or be playing the actual game. Access to it at any other time would be unnecessary. As mentioned, being able to return to the previous screen after viewing the rules is required to keep flow of the program streamline.

Terminate System

Another aspect of which keeps the program streamline is for the user to have the option to exit the game at any point in time while the program is running. There will be a button labelled "Quit" which, when pressed, will terminate the program. There are no conditions set on being able to terminate the program. An important scenario to take note of would be termination of the program during the actual game play. While no errors should occur, any actions committed by robots prior to termination will not be saved. This means that the stats of each robot will not be changed regardless if they committed an action during the game play that would change any of their stats.

Choose Number of Players

Another important action the user will have is the option to select to select the number of players they would like to participate in their game. They will have three options: two players, three players, or six players. Depending on how many players are chosen will influence the next action on how large the board size will be.

Choose Board Size

The user should be able to select how big the board is based on how many players are present. If three players are chosen, the user will then have to decide whether to play on a board with a length of five hexagon spaces on a side or seven hexagon spaces on a side. If two or six players are chosen, then the game will played on a board of length five and seven (respectively).

Choose Number of Humans

Once the number of players is chosen, the user then must choose how many of the players in the game will be humans, which will determine how many will be controlled by robots. Number of players the user can select to be human is limited by how many players the user initially stated. Through the use of radial buttons, the maximum number of human players the user can select will be the number of players previously selected to play.

Select Teams to Fight

Once all the previously stated actions are complete the user must be able to select specific robot teams to participate in the game to fight. The robot librarian must be able to enumerate and show the list of robots and their statistics in which the user can choose from. Using boxes, the user will be able to select the specific number of robot teams (based on how many players they said were participating). There will be no way for the user to proceed to the next part of the program until the right number of robot teams have been selected.

Enumerate

In order for the user to select which robot teams they want to play with, the robot librarian must be able to generate the statistics for the user to be able to view. These statistics must be put into a readable table. In order for this to be done, the code file downloaded by the librarian must be readable. If the file is not readable for whatever reason, then the system will have an error when trying to display the statistics. To fix this, the only current solution would be for the user to terminate the program and have someone examine the file. Once done, the process should be attempted again.

Download Robots

In order for the robot librarian to enumerate the statistics for the user, they will first have to download the information. Issues can arise when the file containing the robots is not present, in which case the program would terminate and wait for an openable file to be present.

Register New Robots

Once the file is downloaded and enumerated, the librarian must have the ability to add new robot teams for the user to use. It will give new team names and individual robot names. This action will only be present if there is a file new information can be saved and so long as the system has enough memory to store the new team. Any team will start with all statistics at zero.

Revise Existing Robot Teams

Newly created teams and current teams should also be editable. By editing, the team name and individual robot names can be altered. The result in editing any team will result in the statistics for that team being brought back to zero.

Retire Existing Robot Teams

In addition to being edited, robot teams must also be able to be deleted from the library. This would result in that team being no longer playable or allow the user to select it.

Update Robot Stats

The last action needed for the robot librarian to perform is the ability to update the statistics of all robots and their teams after every game. In order for this to occur, a full game must be completed. If the game is terminated before a game is completed, then any adjustments to stats that occurred during the game will not be recorded or updated.

Display

Now that all properties have been selected, the game-play screen is ready to be displayed. The program must be able to display the correct size of game board. There will be five spaces on each side for two players, and seven on each side with six players. If the user has selected three players, they have also chosen the board size to be either five or seven spaces on each side. The program must also randomly assign each player in the game to a colour and robot team.

Simulation

If there are no human players in the game, the program will simulate the game behind the scenes and display the end of game screen. If this is taking too long the user has the option to terminate the game. The user must be able to end the game because the game may take a long time to simulate, and may potentially never end.

Move and Attack Robots

Each player, human and computer, must be able to move the appropriate robot and be able to attack any robots within their range. They must also be able to move again if they still have any moves left. In the case of humans these options will be determined by the player supplying input to the keyboard and mouse. No other player should be able to move until the previous player has completed their moves. For teams that are being operated by the computer

the program must have an interpreter to take the moves determined by a certain robots' code and translate that into the computers actions in the game.

Should Have

There are a few functions that will be included in the final product that were not explicitly stated as being required. We feel as though these functions should be added enhance the gameplay and provide the user with a better experience.

Current Robot Stats

During the game, there will be a section on the side that will display the current statistics of the game. This will include each teams robots that remain, along with how much health they have left. For this to remain accurate, these statistics must be updated every time a robot shoots.

End Statistics screen

Once the game is over, a screen will be displayed that shows the final statistics of the game. This means that statistics must be kept track of for each team. This will include how much damage they dealt out, how much damage they received, and how many spaces they moved. After this it will update these statistics to the Robot Librarian.

Could Have

Due to time constraints we will not be able to include all of the components that we would like to. We have come up with a few ideas that we believe would improve the game, but they are not detrimental to the functionality of the game. We are not promising that these features will be added, just ideas that could be added in future version of this program. The following is a few aspects that could be added in future versions.

Animation

When a robot is moving to a new position on the map it looks as though it is driving there, instead of just disappearing and reappearing from space to space. Also when a robot is firing at a space on the map, it would show a missile flying through the air to the spot, and an explosion would be shown on collision.

Sound Effects

It would be nice to add sound effects to our game. This would possibly include sounds for firing missiles, explosions, and driving robots. We could also have some sort of intense background war music playing.

More advanced AI

Based on testing game-play, we would observe the actions of the CPU's strategy, and devise new strategies that would make the game more difficult for humans to play against. Perhaps, as an example, the CPU's could detect if another player is doing well, and teams up with other CPU's in the game to even things out. We would make an option in the game setup interface to choose a desired difficulty.

Networking

As of now, our game is designed to be played on a single computer. In a future version, the game would be able to be played on different computers for different human players. Each players screen would only display the area that they are allowed see, and would not be allowed to do anything until it is their turn to move. This would be more desirable for multi-player use.

Choice of team

It would be nice to add a feature that allows the user to choose their team that they would be playing with, rather than being assigned

Feedback based on user testing

Once we have a working version of our game, we would test it with some people and use their feedback to make improvements based on their opinions.

Product Functionality

It is important that this program be user friendly for anyone that wishes to play it. The goal is that it should be easy to handle regardless of any one users' background. This section will provide a detailed walkthrough of the various functions and actions that can be taken for one complete play through of the game.

Upon execution of the program, the user will be brought to the main menu which will have three options on the screen to choose from. The "Help" button will bring the user to a screen where they will be able to view the rules of the game in detail and provide information on troubleshooting any technical difficulties by providing contact information. From this screen, the user will be able to return to the main menu with a "Back" button and also be able to terminate the program with a "Quit" button. It is important to note that user will always be given the ability to immediately terminate the program through the use of a "Quit" button on every screen, including the main menu. Once the user is ready to play the game, they will select the "Begin" button which will take the user to the game properties screen.

In the game properties screen, the user will set all the parameters needed to initialize the game. In addition, the user will be able to quit the game as previously stated or go back to the main menu. The first property the user will select will be the number of players that will be participating in the game. The choices the user will have are 2, 3 or 6. After that, the user will select how many of those players will be controlled by humans. Once this is done the user will choose the size of the game board which will have sides of either five or seven hexagons in length. This choice is only available to the user if they have selected the number of players to be three. Choosing two or six players will automatically have the board size as five and seven hexagons in length, respectively. Once these properties are fulfilled, the ability to continue will be available through a "Continue" button which will being the user to the team selection screen.

At the team selection screen, the user will have multiple options to choose from. There will be table listing the statistics of each robot team and their robots available for view. From this, the user can decide which teams they want to play in their game. If a new team is desired, they can create a new team by selecting the "New Team" option from the menu, where they will be able to select the teams name and the individual robots for that team. There will also be the ability to edit any existing teams as well, and retire (delete) teams as well. Any teams that are edited or created will have their stats reset to zero. By checking off the team boxes, the user can decide which ones to use and then proceed to the actual game.

The game screen will be where the actual game will be played. At all times there will be two buttons in the bottom left corner to either quit or be able to view the rules again. The teams are displayed on each side, with the health of their robots visible. The bottom right corner will show which robot is currently playing and how many moves they have left to make. The hexagon shaped board made up of smaller hexagons will be in the center of the screen. Each player controls their team of three robots on the board: a scout, sniper and a tank. Each robot has different properties. The tank has the highest attack and health at 3 points, meaning that it can do three health points of damage to any robot it attacks. The tank also has the lowest movement and attack range at 1 point for each which means the tank can only move one space per turn and can only attack robots one space away. The sniper has the highest attack range at 3 points. Its attack,

health and movement are all at 2 points. The scout has the highest movement at 3 points, but has the lowest health and attack at 1 point and attack range at 2 points. A single turn consists of a series of rounds where every robot from each team will move once. The Turn starts with the red team, who will move their highest movement robot and then procced to orange, yellow, green, purple, and then blue. Once this round is complete, each team will move their next highest movement robot and proceed in the same order. The turn is over once all robots have moved. If a team does not have another robot to play in their round, then it moves to the next team. Human players will take turns switching off the keyboard as this version of the program will only allow players to play on the same computer. Not all robots are visible to the player, only the ones within the range will show up on the board. Robots can only see other robots that are in their range, and will utilize a series of functions in the Forth programming language to be able to communicate with each other. The primary function of which will be a mailbox system where they will communicate visibility and health to other robots in range. If a robot is damaged by another robot then their health will decrease by the attackers' damage. The health of that robot will be updated on the screen for human players watching the game. Once a robot's health reaches zero, they will no longer be able to play. When only one team has robots remaining, the game is over and the user is brought to a summary screen.

If the user had previously chosen zero human players during the properties screen, what will happen is that a simulation will run between the selected robot teams and bring the user to the summary screen after the simulation has run.

The summary page states who won and some brief statistics like how many moves that player made and how many robots they killed. The user will be given the ability to view an updated stats of all teams with a "Stats" button; be able to terminate the program; or play another game which will bring them to the properties screen. By clicking the stats button, a table of all teams stats will be displayed, and the user will again be able to select to play again or quit the program.

Sequence Diagrams

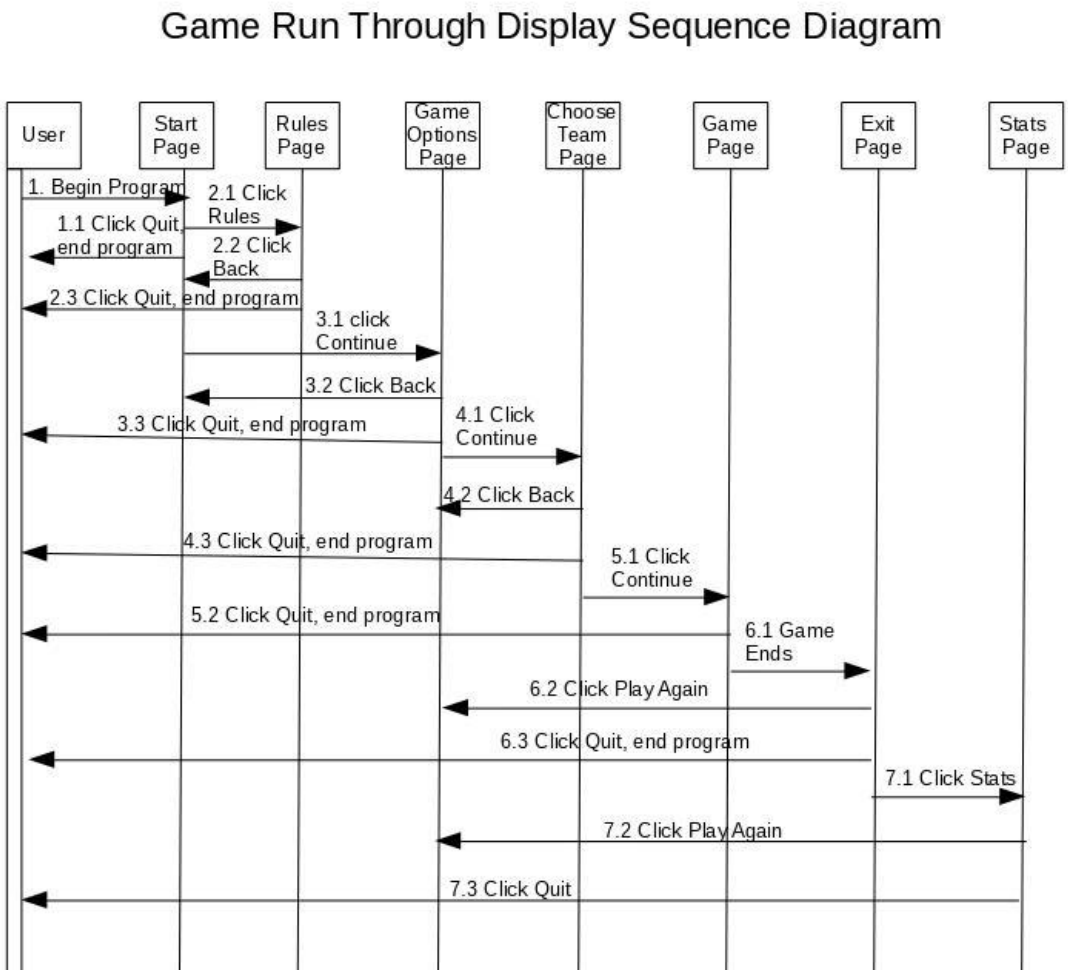

Figure 2: The Game Run Through Display Sequence Diagram shows in detail the different paths a user can take through the games GUI interfaces. Beginning at the Start page and ending at the post-game stats page.

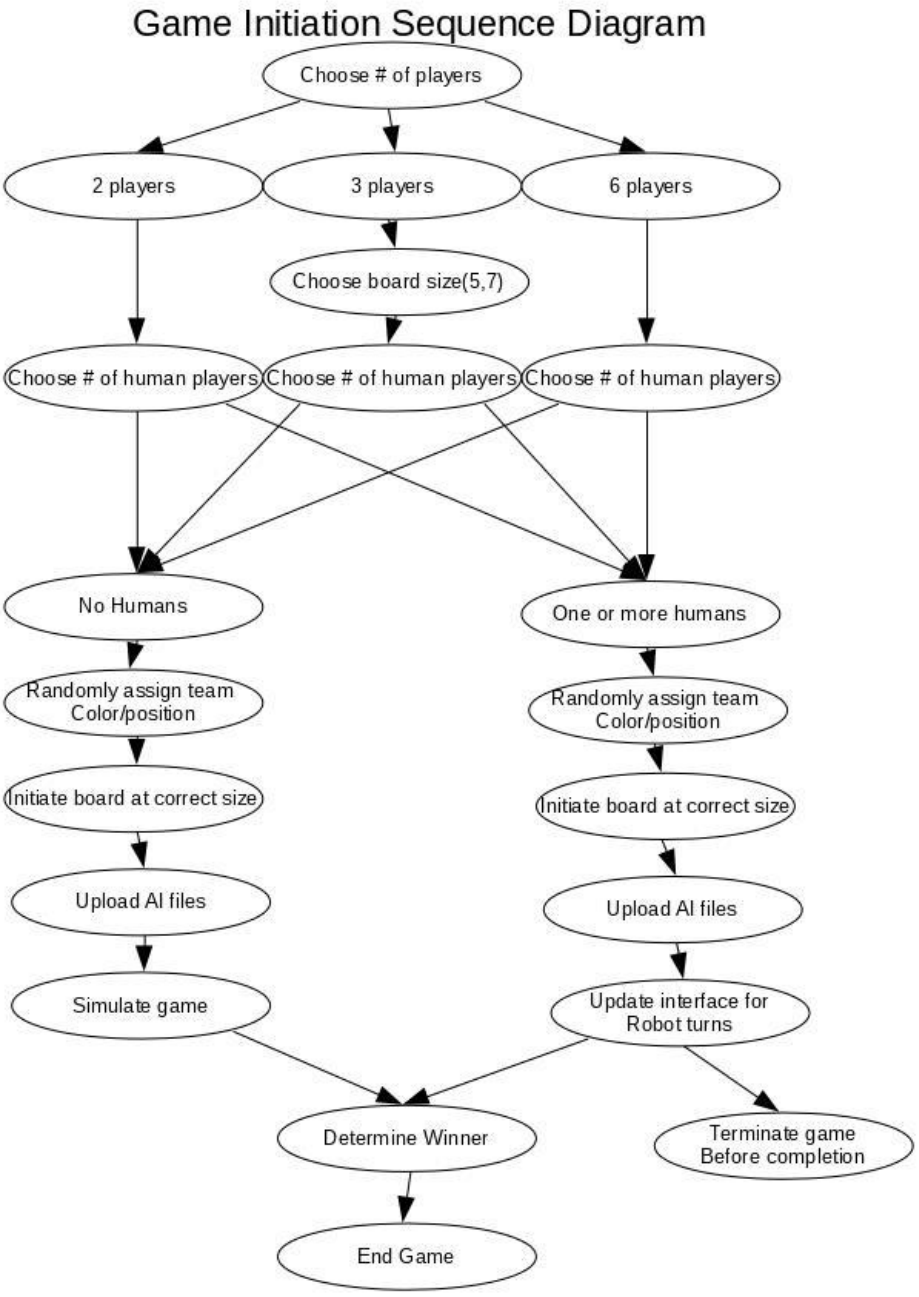

Figure 3: The game initiation sequence diagram describes a play through of a single match of Battlebots, from the various game options, through board initiation until a winner is decided and the end game stats shown. It does not include the numerous possible robot actions during a match.

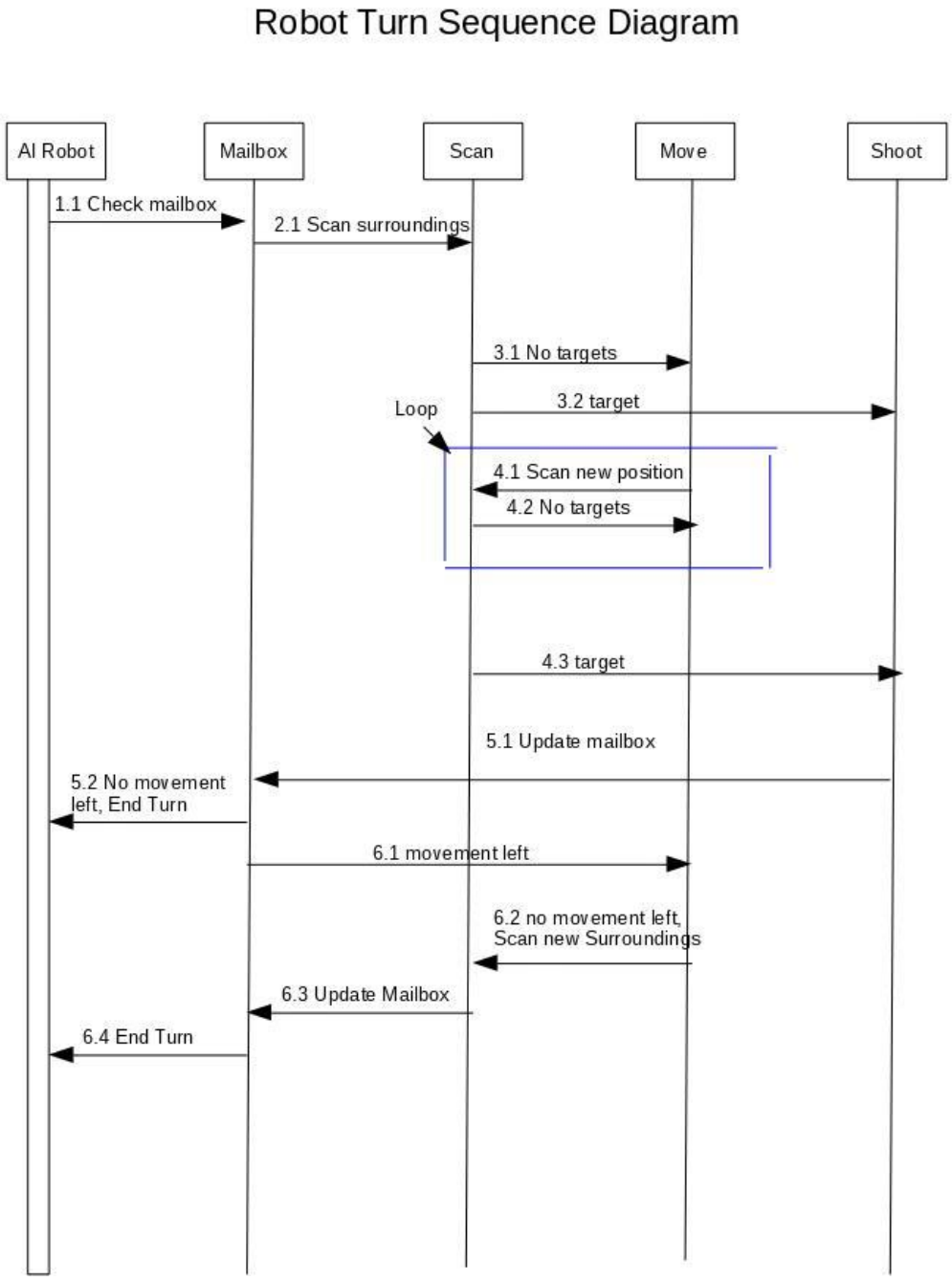

Figure 4. The Robot Turn Sequence Diagram describes in detail the different action paths an AI controlled robot can take from the beginning of their turn until the end.

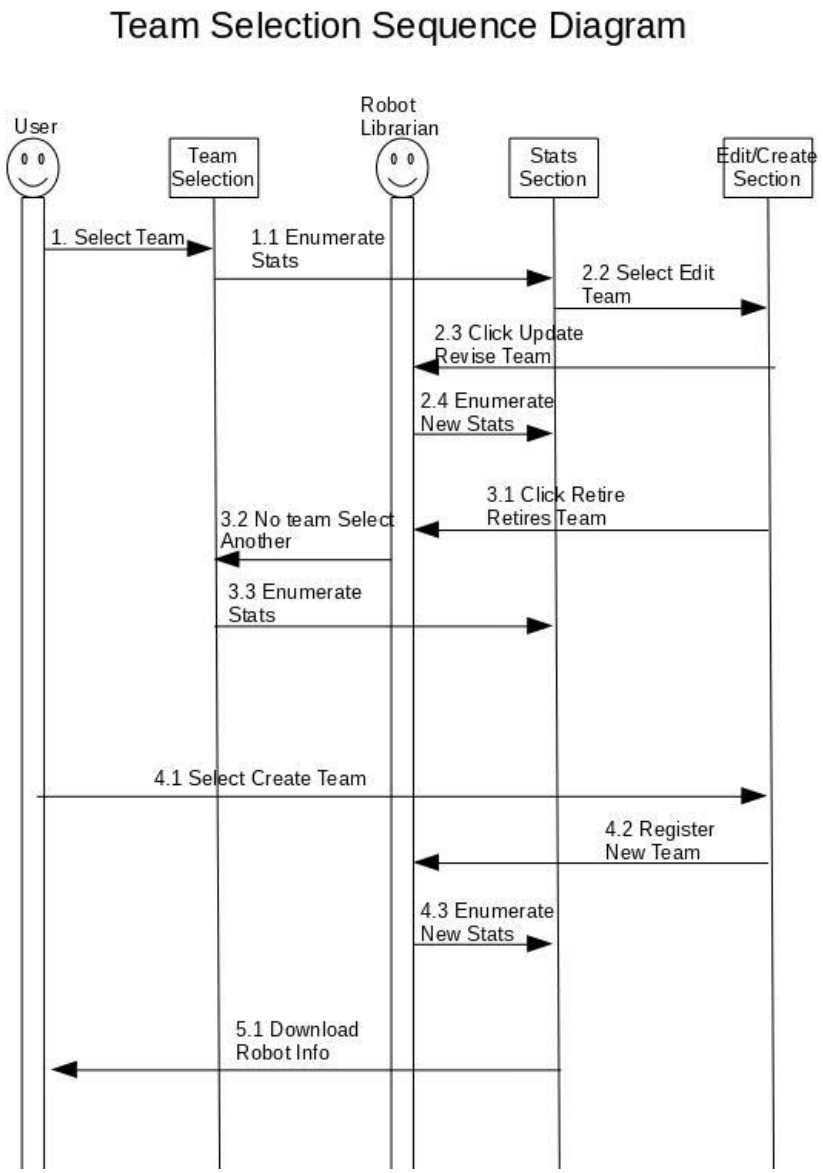

Figure 5. The Team Selection Sequence Diagram describes the process involved in choosing, creating, editing, or deleting a robot team.

User Interfaces

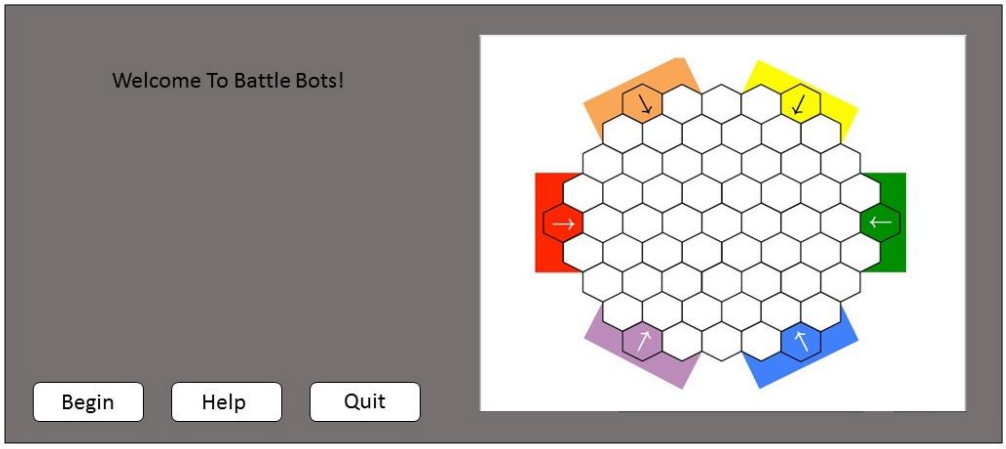

Figure 6. This is the main menu interface. The “Begin” button take the user to the Game Properties screen, “Help” brings up the game rules screen, and “Quit” exits the game.

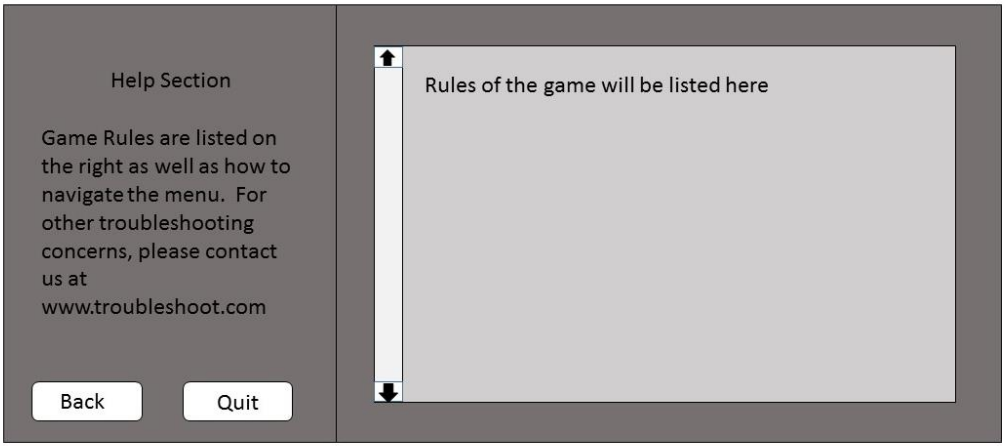

Figure 7. The Help Section screen displays the rules of the game and a link to a website for troubleshooting. The “Back” button returns to the main menu, and “Quit” exits the game.

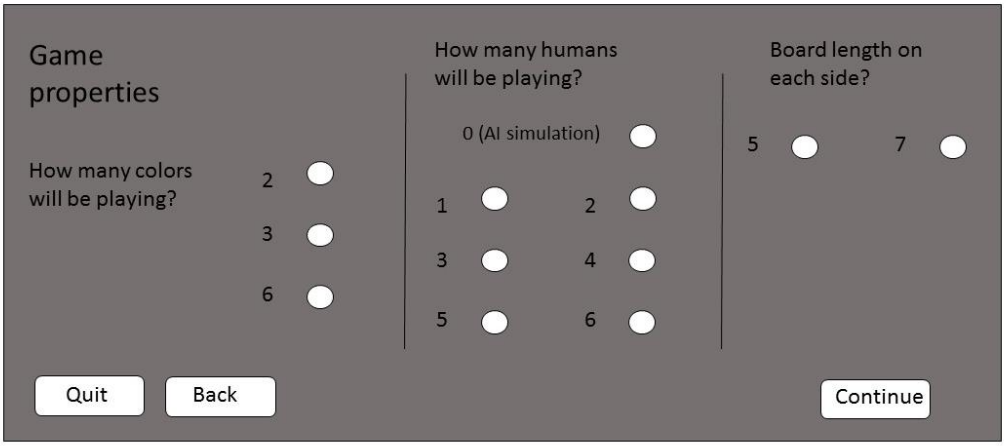

Figure 8. In the Game properties interface, the user will be able to choose how many team will be participating. They will then choose how many of those teams will be played by humans. If the user chose three players, they will then get to choose if the board size will have five or 7 spaces on each side.

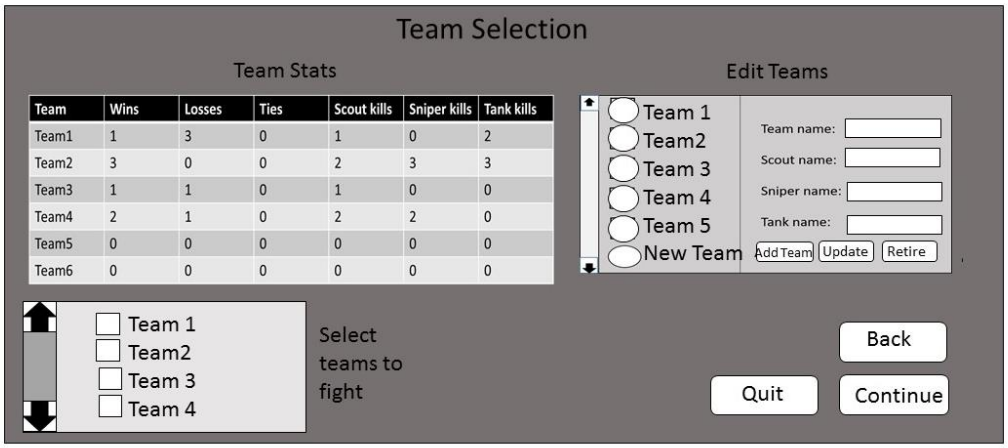

Figure 9. The user is able to choose which teams will play in the game. They will also be able to add a new team, and retire an existing team. The "Back" button takes the user back to the game properties; the "Quit" button exits the game; and the "Continue" button takes the user to the game

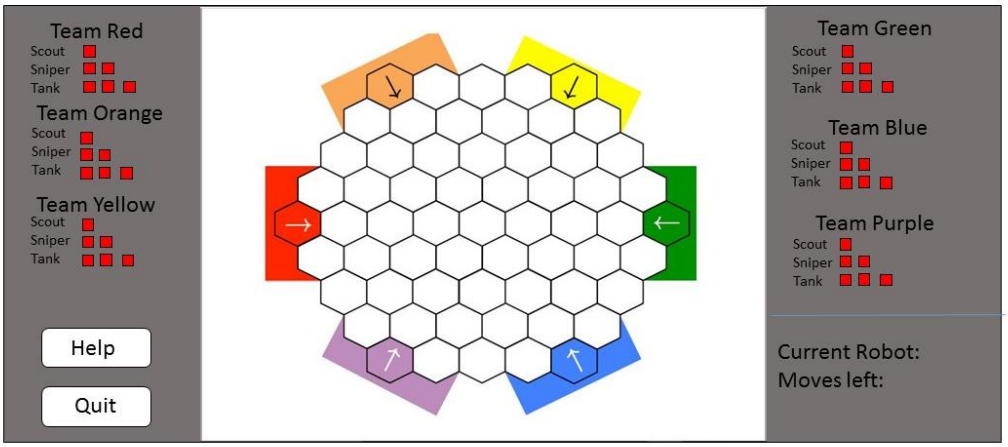

Figure 10. This is the game screen. It displays each teams robots remaining and each robots health remaining. When it is a human teams turn, it shows the robot they can move, along with how many moves it has left. It also has a "Help" button to display the rules screen, and a "Quit" button to exit the game.

Figure 11. The winner of the game is displayed as well as 3 buttons. The "Play Again" button starts up a new game with the same amount of human teams, and AI teams. The "Stats" button takes the user to the stats screen and the "Quit" button exits the game

Figure 12. This is the end game stats screen. It displays each teams' wins, losses, ties, scout kills, sniper kills, and tank kills. The "Play Again" button starts up a new game that will bring the user back to game properties screen. The "Quit" button exits the game.

Executive Summary

Battle Bots is a program written by Makenzie Power, Haotian Ma, Ryan Tetland, William Revell and Mitchel Kovacs as the final project for the computer science class Intermediate Software Engineering. BattleBots is an arena based robot fighting game where humans and AI face off with teams of three robots with variable player amounts and board sizes to determine a victor. The game will be designed using JAVA and FORTH to run on linux based systems utilizing Eclipse, NetBeans, JSON, LibreOffice, Microsoft PowerPoint, Kate and Microsoft Word. For the game to function there are several important features that must be included. The user must have access to a help section, a program termination option, and must be able to navigate through the games interfaces. When setting up a game the user will be able to select the number of players in the game, the number of AI and humans, the board size in a 3-player game, and the team they want to play as. During gameplay the board will be initialized according to the previously chosen game options, randomly assign colours, and will correctly determine the turn order. Each robot will be able to scan, move and shoot using either AI code or input from the mouse and keyboard. There will be statistics shown during the game and game end screen will be shown upon completion as well as a stats screen at the end of the match. In the future versions of BattleBots we hope to include animation, sound effects, more advanced AI, networking, the ability to choose specific robots for their team and a customer feedback section. In the report we give a step by step run through of the game from the start-up screen through the game initiation and game play until the final score screen. There will be four actors in the game including the user, human players, robot players and robot librarian. A use case diagram has been included for each of the before mentioned actors. All actions are described in detail including their pre and post conditions along with the various scenarios associated with each of them. Lastly, there are 4 provided sequence diagrams and 6 GUI interfaces. The sequence diagrams show the step by step process through several important primary actions: team selection, a robot's turn, game initiation

and game run through display. The 6 GUI interfaces give the reader a visual representation of the start page, rules page, game options page, team selection page, game interface, result page and the statistics page.