Requirements Document

Project: Robot Wars

Team: D4

# Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Revision Number | Revision Date / Time | Revision Author | Revision Reason |
| 0.9 | 2016/10/01 5:58 AM | Scott | Adding Revision Table |
| 0.91 | 2016/10/01 6:10 AM | Scott | Renumbered Sections |
| 0.92 | 2016/10/01 10:45 AM | Scott | Edited Section 1. |
| 0.93 | 2016/10/01 12:32 PM | Scott | Edited Section 2. |
| 0.94 | 2016/10/01 1:21 PM | Scott | Edited Section 3. |
| 0.95 | 2016/10/01 2:55 PM | Scott | Edited Section 4. |
| 0.96 | 2016/10/01 6:26 PM | Scott | Finished Host Scenarios. Added pictures. Started Player Scenarios. |
| 0.97 | 2016/10/02 11:18 AM | Scott | Finished Player and Robot Scenarios. Added information in section 6 and 7. |
| 0.98 | 2016/10/02 12:46 PM | Scott | Finished Sections 4 and 5. |

Contents

[Revision History ii](#_Toc463175811)

[Executive Summary 1](#_Toc463175812)

[1.0 Document Overview 2](#_Toc463175813)

[1.1 Purpose of Document 2](#_Toc463175814)

[1.2 Overview of Document 2](#_Toc463175815)

[1.3 Terms and Definitions 2](#_Toc463175816)

[1.3.1 UML-Related Terms 2](#_Toc463175817)

[1.3.1.1 Actor 2](#_Toc463175818)

[1.3.1.2 Action 2](#_Toc463175819)

[1.3.1.3 Use-Case 2](#_Toc463175820)

[1.3.1.4 Scenario 2](#_Toc463175821)

[1.3.1.5 Graphical User Interface (GUI) 2](#_Toc463175822)

[1.3.2 Game-Specific Terms 3](#_Toc463175823)

[1.3.2.1 Match 3](#_Toc463175824)

[1.3.2.2 Turn 3](#_Toc463175825)

[1.3.2.2 Round 3](#_Toc463175826)

[1.3.2.3 Hotseat 3](#_Toc463175827)

[1.3.2.4 Artificial Intelligence (or AI) 3](#_Toc463175828)

[2.0 Background 3](#_Toc463175829)

[2.1 Team Introductions 3](#_Toc463175830)

[2.1.1 Tushita Patel 3](#_Toc463175831)

[2.1.2 Yige Huang 4](#_Toc463175832)

[2.1.3 Janelle Hindman 4](#_Toc463175833)

[2.1.4 Scott Hebert 4](#_Toc463175834)

[2.1.5 Nickolas Gough 4](#_Toc463175835)

[2.2 Product Introduction 4](#_Toc463175836)

[2.2.1 Overview of the Robot Wars Board Game 4](#_Toc463175837)

[2.2.2 Expected Benefits of the Computerized Interface 5](#_Toc463175838)

[2.2.3 Product Implementation 5](#_Toc463175839)

[2.3 Assumptions 5](#_Toc463175840)

[2.3.1 Assumption of Knowledge of Game Rules 5](#_Toc463175841)

[3.0 Actors and their Relationships 5](#_Toc463175842)

[3.1 Overview 5](#_Toc463175843)

[3.2 Actors 5](#_Toc463175844)

[3.2.1 Host 5](#_Toc463175845)

[3.2.2 Player 6](#_Toc463175846)

[3.2.3 Spectator 6](#_Toc463175847)

[3.2.4 Robot 6](#_Toc463175848)

[3.2.5 Robot Librarian 7](#_Toc463175849)

[3.3 Relationships among Actors 7](#_Toc463175850)

[3.4.1 System Diagram 7](#_Toc463175851)

[4.0 Scenarios 8](#_Toc463175852)

[4.1 Overview 8](#_Toc463175853)

[4.2 Host Scenarios 8](#_Toc463175854)

[4.2.1 Primary Scenario: Start New Game 8](#_Toc463175855)

[4.2.2 Primary Scenario: View Match Results 10](#_Toc463175856)

[4.2.3 Primary Scenario: View Robot Archive 10](#_Toc463175857)

[4.2.4 Primary Scenario - Change Settings 12](#_Toc463175858)

[4.2.5 Primary Scenario - Exit Application 12](#_Toc463175859)

[4.3 Player Scenarios 13](#_Toc463175860)

[4.3.1 Primary Scenario - Take Turn 13](#_Toc463175861)

[4.3.2 Primary Scenario - Navigate Board 17](#_Toc463175862)

[4.3.3 Primary Scenario - View Game Log 19](#_Toc463175863)

[4.3.4 Primary Scenario - View Robot Statistics 20](#_Toc463175864)

[4.3.5 Primary Scenario - End Game 21](#_Toc463175865)

[4.4 Spectator Scenarios 22](#_Toc463175866)

[4.4.1 Primary Scenario - Navigate Board (Spectator) 22](#_Toc463175867)

[4.4.2 Primary Scenario - View Game Log (Spectator) 22](#_Toc463175868)

[4.4.3 Primary Scenario - View Robot Statistics (Spectator) 22](#_Toc463175869)

[4.5 Robot Scenarios 22](#_Toc463175870)

[4.5.1 Primary Scenario - Take Turn (Robot Actor) 22](#_Toc463175871)

[4.6 Robot Librarian Scenarios 23](#_Toc463175872)

[4.6.1 Primary Scenario: Enumerate 23](#_Toc463175873)

[4.6.2 Primary Scenario - Download 24](#_Toc463175874)

[4.6.3 Primary Scenario - Update Statistics 25](#_Toc463175875)

[4.6.4 Primary Scenario: Register 25](#_Toc463175876)

[4.6.5 Primary Scenario: Revise 26](#_Toc463175877)

[4.6.6 Retire 27](#_Toc463175878)

[5.0 Storyboards and User Interface 28](#_Toc463175879)

[5.1 Overview 28](#_Toc463175880)

[5.2 Interface Elements 28](#_Toc463175881)

[5.2.1 Main Menu Screen 28](#_Toc463175882)

[5.2.2 Player Select Screen 28](#_Toc463175883)

[5.2.3 Game Options Screen 30](#_Toc463175884)

[5.2.4 Game Screen 31](#_Toc463175885)

[5.2.5 Match Results Screen 31](#_Toc463175886)

[5.2.6 Robot Archive Screen 32](#_Toc463175887)

[5.2.7 Game Settings Screen 32](#_Toc463175888)

[5.3 Storyboard Walkthroughs of Typical Use-Cases 33](#_Toc463175889)

[6.0 Non-Functional Requirements 34](#_Toc463175890)

[6.1 Overview 34](#_Toc463175891)

[6.2 Data Requirements 34](#_Toc463175892)

[6.3 Hardware Requirements 34](#_Toc463175893)

[6.4 Software Requirements 34](#_Toc463175894)

[7.0 Potential Extensions to System 34](#_Toc463175895)

[7.1 Overview 34](#_Toc463175896)

[7.2 Additional Game Mechanics 34](#_Toc463175897)

[7.2.1 Use Case: Changing the Game Options 34](#_Toc463175898)

[7.3 Sound and Music 35](#_Toc463175899)

[7.3.1 Use Case: Adjusting Volume 35](#_Toc463175900)

[7.4 Asynchronous Play System 35](#_Toc463175901)

[7.4.1 Use Cases 35](#_Toc463175902)

[Appendices: 37](#_Toc463175903)

[Appendix A: References 37](#_Toc463175904)

[-JSON reference 37](#_Toc463175905)

[Appendix B: Robot Wars Rules 37](#_Toc463175906)

[Index 38](#_Toc463175907)

[Figure 1: System Diagram of Actors and Actions 7](#_Toc463175908)

[Figure 2: Scenario Flow Diagram - Start New Game 9](#_Toc463175909)

[Figure 3: Scenario Flow Diagram - View Match Results 10](#_Toc463175910)

[Figure 4: Scenario Flow Diagram - View Robot Archive 11](#_Toc463175911)

[Figure 5: Scenario Flow Diagram: Change Settings 12](#_Toc463175912)

[Figure 6: Scenario Flow Diagram - Exit Application 13](#_Toc463175913)

[Figure 7: Scenario Flow Diagram - Take Turn (Player) 14](#_Toc463175914)

[Figure 8: Scenario Flow Diagram - Move Robot 15](#_Toc463175915)

[Figure 9: Scenario Flow Diagram - Shoot 16](#_Toc463175916)

[Figure 10: Scenario Flow Diagram - Inspect Space 16](#_Toc463175917)

[Figure 11: Scenario Flow Diagram - End Turn 17](#_Toc463175918)

[Figure 12: Scenario Flow Diagram - Navigate Board 18](#_Toc463175919)

[Figure 13: Scenario Flow Diagram - Pan Screen 19](#_Toc463175920)

[Figure 14: Scenario Flow Diagram - Zoom Board 19](#_Toc463175921)

[Figure 15: Scenario Flow Diagram - View Game Log 20](#_Toc463175922)

[Figure 16: Scenario Flow Diagram - View Robot Statistics 21](#_Toc463175923)

[Figure 17: Scenario Flow Diagam - End Game 22](#_Toc463175924)

[Figure 18: Scenario Flow Diagram - Take Turn (Robot) 23](#_Toc463175925)

[Figure 19: Scenario Flow Diagram - Enumerate 24](#_Toc463175926)

[Figure 20: Scenario Flow Diagram - Download 24](#_Toc463175927)

[Figure 21: Scenario Flow Diagram - Update Statistics 25](#_Toc463175928)

[Figure 22: Scenario Flow Diagram - Register 26](#_Toc463175929)

[Figure 23: Scenario Flow Diagram - Revise 27](#_Toc463175930)

[Figure 24: Scenario Flow Diagram - Retire 27](#_Toc463175931)

[Figure 25: Robo-Wars Title Screen 28](#_Toc463175932)

[Figure 26: Player Selection Screen 29](#_Toc463175933)

[Figure 27: Robot Team Select Screen 30](#_Toc463175934)

[Figure 28: Game Options Screen 30](#_Toc463175935)

[Figure 29: Main Game Screen 31](#_Toc463175936)

[Figure 30: Match Results Screen 32](#_Toc463175937)

[Figure 31: Robot Archive Screen 32](#_Toc463175938)

[Figure 32: Game Settings Screen 33](#_Toc463175939)

[Table 1: Use-Case Definition - Start New Game 8](#_Toc463175940)

[Table 2: Use-Case Definition: Change Game Options 9](#_Toc463175941)

[Table 3: Use-Case Definition - View Match Results 10](#_Toc463175942)

[Table 4: Use-Case Definition - View Robot Archive 11](#_Toc463175943)

[Table 5: Use-Case Definition: Change Robots 11](#_Toc463175944)

[Table 6: Use-Case Definition - Change Settings 12](#_Toc463175945)

[Table 7: Use-Case Definition - Exit Application 12](#_Toc463175946)

[Table 8: Use-Case Definition - Take Turn (Player) 13](#_Toc463175947)

[Table 9: Use-Case Definition - Move Robot 14](#_Toc463175948)

[Table 10: Use-Case Definition - Shoot 15](#_Toc463175949)

[Table 11: Use-Case Definition - Inspect Space 16](#_Toc463175950)

[Table 12: Use-Case Definition - End Turn 17](#_Toc463175951)

[Table 13: Use-Case Definition - Navigate Board 17](#_Toc463175952)

[Table 14: Use-Case Definition - Pan Board 18](#_Toc463175953)

[Table 15: Use-Case Definition - Zoom Board 19](#_Toc463175954)

[Table 16: Use-Case Definition - View Game Log 20](#_Toc463175955)

[Table 17: Use-Case Definition - View Robot Statistics 20](#_Toc463175956)

[Table 18: Use-Case Definition - End Game 21](#_Toc463175957)

[Table 19: Use-Case Definition - Take Turn (Robot Actor) 23](#_Toc463175958)

[Table 20: Use-Case Definition: Enumerate 23](#_Toc463175959)

[Table 21: Use-Case Definition - Download 24](#_Toc463175960)

[Table 22: Use-Case Definition - Update Statistics 25](#_Toc463175961)

[Table 23: Use-Case Definition - Register 25](#_Toc463175962)

[Table 24: Use-Case Definition - Revise 26](#_Toc463175963)

[Table 25: Use-Case Definition - Retire 27](#_Toc463175964)

# Executive Summary

WRITE ME LAST

# Document Overview

## 1.1 Purpose of Document

The purpose of this document is to set and inform the expectations of potential customers of the Robot Wars application. After reading this document, a potential customer should understand

* what the application can be expected to do,
* what interactions with the application are possible, and
* what will be necessary to properly use the application.

## 1.2 Overview of Document

To this end, the document is broken up into a number of sections. This section, Section 1, explains the document itself, as well as its terms and definitions. Section 2 details the project team, the purpose of the Robot Wars application, and the assumptions dealing with the team’s understanding of the system. Section 3 lists the various Actors or roles that interact with the application. Section 4 details the use-cases, or ways, that the Actors can interact with the application. Section 5 contains notional user interface elements that will operationalize the use-cases from Section 4. Section 6 gives the non-functional requirements identified by the team. Section 7 details non-essential elements that may or may not be incorporated into the final product. After this, there are appendices holding supplementary information, as well as an index.

## 1.3 Terms and Definitions

Below can be found specific terms used in this document that may have reserved meanings; specifically, their meaning is not the common everyday meaning of the term. They are divided into two sections: Unified Markup Language (UML) related terms and Game-specific terms.

## 1.3.1 UML-Related Terms

The UML-related Terms used in this document are as follows:

### 1.3.1.1 Actor

One term is Actor. An Actor is a specific role that interacts with the system.

### 1.3.1.2 Action

Another term is Action. An Action is a specific way of using the system.

### 1.3.1.3 Use-Case

A third term is Use-Case. A Use-Case can be thought of as an Actor-Action pair, and is a specific way a specific Actor interacts with the system.

### 1.3.1.4 Scenario

Next, we have the term Scenario. For the purposes of this document, the terms Scenario and Use-Case are identical in meaning. Additionally, Scenarios can be either Primary or Secondary in nature. A Primary Scenario is a typical usage pattern. A Secondary Scenario is an atypical usage pattern.

### 1.3.1.5 Graphical User Interface (GUI)

The final UML-related term is Graphical User Interface, abbreviated as GUI. This term refers to the screens that the users will navigate to use the system. For this document, GUI and Interface (by itself) are identical in meaning.

## 1.3.2 Game-Specific Terms

The Game-Specific Terms used in this document are as follows:

### 1.3.2.1 Match

The first Game-Specific term is Match. This is a single game and its outcome. For this document, Match and Game are identical in meaning.

### 1.3.2.2 Turn

A second Game-Specific term is Turn. A Turn is a single robot’s choices regarding what to do in the game. It begins when the next available robot is selected, and ends when the robot either runs out of available choices, or chooses to end its Turn.

### 1.3.2.2 Round

The next Game-Specific term is Round. A Round is a sequence of Turns where every robot is given the opportunity to play a Turn. A new Round begins after the last robot with an opportunity to play takes its turn.

### 1.3.2.3 Hotseat

Another Game-Specific term is Hotseat. Hotseat refers to a type of multiplayer structure where each user interacts with the game through the same computer. This is contrasted with a Network multiplayer structure where each user interacts with the game through a separate computer.

### 1.3.2.4 Artificial Intelligence (or AI)

A final Game-Specific term is Artificial Intelligence, or AI. In this document, this refers to the Computer Players that interact with the game system. This has identical meaning to Computer Player.

# Background

Having finished with terms, this next section provides information regarding the Research and Development (R&D) team and the application itself.

## 2.1 Team Introductions

Below is a brief biography of each member of the R&D team.

**<Possibly insert picture here>**

### 2.1.1 Tushita Patel

Tushita Patel is a third year undergraduate student majoring in Computer Science at the University of Saskatchewan. Having a strong background in fine arts and mathematics, Tushita's interests lie in Human-Computer Interaction and Artificial Intelligence. Tushita is currently working as a Teacher Assistant and has spent time working under Dr. Reagan Mandryk in the HCI Lab and under Dr. Chris Soteros in the Department of Mathematics and Statistics in the past years at the University of Saskatchewan. When she is not studying in school, Tushita likes to get involved in the community by doing volunteer work.

**<Possibly insert picture here>**

### 2.1.2 Yige Huang

Yige Huang is a third year undergraduate student majoring in Computer Science at the University of Saskatchewan. Completing her entire second year in a single summer, Yige is a determined and competent student who specializes in team projects. When she is not completing assignments or studying for an exam, Yige enjoys working on solving problems and writing code.

**<Possibly insert picture here>**

### 2.1.3 Janelle Hindman

Janelle Hindman is in her third year of uinversity pursuing a B.Sc. Four-year with a major in Computer Science. Having an insatiable thirst for learning, Janelle takes every opportunity to learn and improve herself. Janelle is Canadian, but has lived in Japan for the past ten years, where she joined the Japanese gaming scene. Specializing in designing game mechanics and developing graphical user interfaces, Janelle enjoys developing and playing video games in her free time. Janelle hopes to learn a lot about the software engineering process and improve her leadership abilities.

**<Possibly insert picture here>**

### 2.1.4 Scott Hebert

Scott Hebert is a non-traditional graduate student working in the Aries Lab at the University of Saskatchewan. Originating from the United States of America, Scott already has a Master’s of Science in Industrial Engineering, a Bachelor’s of Science in Industrial Engineering, a Bachelor’s of Arts in Humanities, and co-owning a simulation consulting firm, Scott brings years of experience and a mind full of ideas. When not furthering his education, Scott enjoys playing strategy and role-playing games as well as reading fiction and fantasy novels. Scott is looking forward to building a great product with a great team.

**<Possibly insert picture here>**

### 2.1.5 Nickolas Gough

Nickolas Gough is a third year undergraduate student majoring in Computer Science at the University of Saskatchewan. Nickolas is working as a Teacher Assistant and has spent time working under Dr. Carl Gutwin in the Human-Computer Interaction Lab at the University of Saskatchewan. He enjoys studying algorithms and data structures, but Nickolas also enjoys studying biology and mathematics.

## 2.2 Product Introduction

Team D4’s product is a computerized interface and rules engine for the board game Robot Wars. While full details about the Robot Wars can be found in Appendix B, below is a brief introduction to the board game. After this introduction, an explanation of the computerized interface and its benefits will be given.

### 2.2.1 Overview of the Robot Wars Board Game

Robot Wars is a multiplayer board game where players attempt to use a team of robots to eliminate all opposing robots. The robots are differentiated a unique combination of attributes. The objective of Robot Wars is to be the only remaining player possessing a live robot. Eliminating other players’ robots is accomplished by moving around the hexagonal board and shooting at other robots.

A full description of the board game elements and rules can be found in Appendix B: Robot Wars Game Rules.

### 2.2.2 Expected Benefits of the Computerized Interface

There are four (4) main benefits to the Team D4 product as compared to the paper board game. These benefits are

* reducing cognitive load regarding the state of the game so that players can focus on strategy,
* increasing the strategic element of the game by limiting the visibility of the game board,
* allowing a single player to play against AI opponents, and
* allowing a user to create, store, and evaluate the effectiveness of different AI strategies.

### 2.2.3 Product Implementation

The project will be implemented using Java, an object-oriented programming language. While the full product specifications are given in sections 6.3 and 6.4 below, it is expected to be playable on any computer running Windows 7 and Java 1.8.0\_101 or higher.

## 2.3 Assumptions

With any implementation, there are assumptions made by the R&D Team. Below are listed some of the ones made by Team D4.

### 2.3.1 Assumption of Knowledge of Game Rules

One major assumption made by Team D4 is that the users of the product which are playing the game know how to play the game. While various tutorials may be created (as detailed in section 7), this is not required and there will be minimal assistance by the product in knowing how to play the game Robot Wars.

# 3.0 Actors and their Relationships

## 3.1 Overview

In order to completely determine the requirements of the Team D4 product, the actors have been identified and detailed below. There are five (5) actors identified and discussed in this document: the Host, the Player, the Spectator, the Robot, and the Robot Librarian.

## 3.2 Actors

### 3.2.1 Host

The Host is the entity that starts the application and configures matches. It primarily interacts with main menus and takes no action within the game itself. The Host role must be taken by a human who is using the system, and this role will not be taken by any AI or program. There can only be one Host interacting with the system at any one time. The Host's primary methods of interacting with the system are starting new matches, adjusting options for the program, and exiting the program. It is also responsible for confirming the end of a game after a match has completed.

### 3.2.2 Player

A Player is a human entity that takes turns within the game system. Once a match has begun, zero or more Players may be present and may act on the system. Since the game is designed as a Hotseat application, only one Player may be interacting with the system at any one time. Players have a limited view of the board and are limited in the actions they can perform on the board by their current game state.

Players can take actions related to their turn when the game system determines that it is their turn. On a turn, they may move a robot if they are capable of doing so, shoot a space with a robot if they are capable of doing so, examine a selected space for additional information, and may signal the system that they are finished by ending their turn. Players are also capable of aborting a match without completing it.

In addition to these actions which have an effect on the game state, Players may take action that allows them to view information without affecting the game. Players are capable of panning and zooming the map display, in order to view larger maps. They are also able to view or hide a game log, which displays a history of moves played to this point (from the Player's perspective).

If all human Players in a match are eliminated, the last Player to be eliminated will be converted to a Spectator role by the game system.

### 3.2.3 Spectator

A Spectator is a human entity that is able to view the game state during a match, but may take no actions that affect the game state. A Spectator may only exist during a match where the only extant contestants are AI-controlled teams of Robots. Since the game is designed as a Hotseat application, there may only ever be one Spectator interacting with the system at one time.

Unlike Players, Spectators have an unlimited view of the board and game state. They are able to observe the turns and actions of all Robots and view the map unobstructed, but may not affect the game state with this information. Spectators may abort a match without it running to completion. They also can take all Player actions that have no effect on the game: zooming and panning the map and viewing and hiding the game log. In addition, Spectators may click a button to view statistics about the game.

### 3.2.4 AI

An AI is a Program-controlled entity that takes actions as one of the three types of game pieces within a match. An AI may act only according to instructions contained in its Program that has been downloaded before the game and cannot be controlled by any human entity. Each AI team may contain up to three Game Pieces at one time, and each of these may act according to the same or different Programs.

An AI takes action on the game system in a manner similar to a Player, but accomplishes this interaction differently. It sends commands and makes queries of the system, but does not need to view the board in a graphical sense, and so it cannot perform actions like panning and zooming the map. An AI can take action to move if it is able, and shoot if it is able, as well as query the state of spaces within its vision. An AI also uses the game system to pass messages to its teammates.

When an AI is eliminated from the game, its statistics are tallied and saved. It can take no further action to change the game system.

### 3.2.5 Robot Librarian

The Robot Librarian is a separate system that manages Program records, including the statistics of that Program. It interacts with the game system in order to provide Programs for use in game, and allows itself to be queried by the game. Its primary role is outside of matches; it provides Program records to the system just prior to a match, and then receives the result of a completed match in order to save the statistics and update the records of any Programs used in that match. If a match is aborted without completing, the statistics are not updated and the Robot Librarian does not interface with the system.

The Robot Librarian is able to take actions that read and write to Program records. It is able to enumerate all of the records stored and sort them according to various parameters. It is able to download records for the game system to use, and upload updated versions of those records after a match. It is also able to register a new Program record, revise the record of an existing Program, and retire a Program record in order to free up that Program's name.

The records that the Robot Librarian handles will be JSON-encoded text.

## 3.3 Relationships among Actors

While there are no direct relationships among the various Actors in the application, they can and will interact through the system. As a summary of the above Actors, Figure 1 below is a diagram of the Actors, the System, and the Actions which cross the System boundary.

### 3.4.1 System Diagram

C:\Users\Scott-Sterling\Desktop\systemDiagram.png

Figure 1: System Diagram of Actors and Actions

# 4.0 Scenarios

## 4.1 Overview

Once the Actors are defined, it is necessary to define the scenarios or use-cases under which the Actors interact with the System. The format used for each scenario is as follows:

* There is a text description for each Scenario.
* Then a table appears that has the following components.
  + First is listed the Preconditions. These are conditions that must be true in order for the scenario to apply or execute.
  + Next comes the Flow of Events. This is a high-level overview of the actions that the Actor and System take to execute the scenario.
  + After the Flow the Postconditions are listed. These are the conditions that are true after the Scenario is finished executing. Often this can be thought of as the change in the system state caused by the Scenario.
  + Finally, any Error Conditions are listed. These are conditions that can cause the Scenario to fail to execute.
* After the table, a figure will appear that shows pictorially a sample flow path for the use-case.

The scenarios below will be listed according to the Actor who initiates them.

## 4.2 Host Scenarios

The first Scenarios to be described are the Host Scenarios. These Scenarios primarily deal with creating games.

### 4.2.1 Primary Scenario: Start New Game

The Start New Game Scenario is a Primary Scenario that applies after the application has finished loading. When the application starts up and the user clicks the “Start Game” button from the main menu, the user is redirected to the “Player Select” window, in which the user selects the number of players, whether each player is human or AI, enters the name of each player and clicks the “Start” button, upon which a new game begins with the specified players. From the “Player Select” screen, the user has the option of clicking the “Options” button and adjusting the game options before beginning the game.

Table 1: Use-Case Definition - Start New Game

|  |  |  |
| --- | --- | --- |
| Flow Controller | Main Flow Step | Alternate Flow Step |
| Precondition | Host is at Title Screen. |  |
| Host | Choose to Start Game |  |
| System | Allows Host to Select Players |  |
| Host | Host chooses # and Type of Players | May choose other options |
| Host | Sends System Game Creation Info |  |
| System | Creates game with selected Players and Options |  |
| Postcondition | New game created |  |
| Error Conditions | Incorrect Parameter Values | Incorrect Options |
|  | Game does not Start |  |

C:\Users\Scott-Sterling\Desktop\startGame.png

Figure 2: Scenario Flow Diagram - Start New Game

#### 4.2.1.1 Secondary Scenario: Change Game Options

The Change Game Options Scenario is a Secondary Scenario that applies if the Host decides to change underlying game options while creating a game. When the user clicks the “Options” button in the “Player Select” window, the user is redirected to the “Options” window. The “Options” window contains all the game options the user is allowed to change. Changing the options is as simple as selecting an alternative to the current option setting. The changes will only be applied to the game if the user clicks the “Save Options” button. Otherwise, if the user clicks the “Cancel” button, all changes will be discarded. This is summarized in Table 1, below.

Table 2: Use-Case Definition: Change Game Options

|  |  |  |
| --- | --- | --- |
| Flow Controller | Main Flow Step | Alternate Flow Step |
| Precondition | Host is Selecting Players. |  |
| Host | Choose to Change Options |  |
| System | Displays Options |  |
| Host | Changes Options as Desired and Saves. | Cancels Changes. |
| System | Updates Options and returns Host to Selecting Players | Returns Host to Selecting Players |
| Postcondition | Game options are set as Host desires. | Game options are not changed. |
| Error Conditions | Host not redirected correctly. | Changes saved. |
|  | Changes not saved. |  |

### 4.2.2 Primary Scenario: View Match Results

The View Match Results Scenario is a Primary Scenario that applies after a game has finished. When the match ends, the winner of the match is displayed in a small window. The player clicks the “Okay” button and is redirected to another screen in which the match results are displayed. Results are displayed for each robot of each player.

Table 3: Use-Case Definition - View Match Results

|  |  |  |
| --- | --- | --- |
| Flow Controller | Main Flow Step | Alternate Flow Step |
| Precondition | A Match has ended. |  |
| System | Displays the results of the Match. |  |
| Host | Observes the results |  |
| Host | Chooses to return to Title Screen. | Chooses to view full log. |
| System | Returns Host to Title Screen. | Shows full log of match. |
| Host |  | Chooses to return to Title Screen. |
| System |  | Returns Host to Title Screen. |
| Postcondition | Host is on Title Screen. |  |
| Error Conditions | Match results not shown. | Full log not shown. |
|  | Incorrect results shown. |  |
|  | Host not redirected. |  |

C:\Users\Scott-Sterling\Desktop\viewGameResults.png

Figure 3: Scenario Flow Diagram - View Match Results

### 4.2.3 Primary Scenario: View Robot Archive

The View Robot Archive Scenario is a Primary Scenario that applies when the Host wants to view or manipulate the Robot Archive stored in the application. From the Title Screen, an option to view the archive of robot programs available for use will be present. If selected, the System will display the records of the robots. The Host can then query for more information on robots, if desired. Once finished with queries, the Host can choose to return to the Title Screen to select other options.

Table 4: Use-Case Definition - View Robot Archive

|  |  |  |
| --- | --- | --- |
| Flow Controller | Main Flow Step | Alternate Flow Step |
| Precondition | Host is on Title Screen |  |
| Host | Chooses to view Robot records |  |
| System | Displays the Robot records screen. |  |
| Host | Chooses to query Robots. |  |
| System | Sends query to Robot Librarian and displays result. |  |
| Host | Chooses to return to Title Screen. | Chooses to make more queries. |
| System |  |  |
| Postcondition | Host is on Title Screen. |  |
| Error Conditions | Robot Records do not display correctly. |  |
|  | Queries not Processed properly. |  |

C:\Users\Scott-Sterling\Desktop\viewRobotArchive.png

Figure 4: Scenario Flow Diagram - View Robot Archive

#### 4.2.3.2 Secondary Scenario : Change Robots

The Change Robots Scenario is a secondary scenario that applies when the Host wants to change the Robots registered in the system while in the View Robot Archive scenario. After the Host clicks the “Register” button in the Robot Archive, a small window appears prompting the user to select a robot file from their file system. The Host locates and selects their desired file, clicks the “Register” button, and the system registers the robot into the game.

Table 5: Use-Case Definition: Change Robots

|  |  |  |
| --- | --- | --- |
| Flow Controller | Main Flow Step | Alternate Flow Step |
| Precondition | Host is on Robot Archive window. |  |
| Host | Chooses to register Robots. |  |
| System | Presents Host with file structure to find Robot. |  |
| Host | Locates and selects Robot file. | Cancels without Finding a Robot file. |
| System | Transfers request to Robot Librarian |  |
| Postcondition | Robot File has been Registered. |  |
| Error Conditions | Host cannot select a Robot File. |  |
|  | Robot file is not correctly Registered. |  |

### Primary Scenario - Change Settings

The Change Settings Scenario is a Primary Scenario that applies when the Host wants to change application settings such as graphics, sound, and similar functionality. When the user clicks the “Settings” button in the “Title Screen” window, the user is redirected to the “Settings” window. The “Settings” window contains all the application settings the user is allowed to change. Changing the settings is as simple as selecting an alternative to the current option setting. The changes will only apply to the system if the user clicks the “Save Options” button. Otherwise, if the user clicks the “Cancel” button, all changes will be discarded.

Table 6: Use-Case Definition - Change Settings

|  |  |  |
| --- | --- | --- |
| Flow Controller | Main Flow Step | Alternate Flow Step |
| Precondition | Host is in Title Window. |  |
| Host | Chooses to change Settings. |  |
| System | Displays Settings to Change. |  |
| Host | Changes Settings and Confirms. | Cancels without Confirming |
| System | Updates Settings and Returns Host to Title Window |  |
| Postcondition | Settings updated. |  |
| Error Conditions | Host does not reach Settings. | Setting changes occur without Confirming. |
|  | Setting changes do not occur when Confirmed. |  |
|  | Host not Returned to Title Window |  |

C:\Users\Scott-Sterling\Desktop\changeSettings.png

Figure 5: Scenario Flow Diagram: Change Settings

### Primary Scenario - Exit Application

The Exit Application Scenario is a Primary Scenario that applies when the Host wants to exit the program. While the system is running and after the Host has been returned to the Title window by navigating back, the user can click the “Exit Application” button to exit and kill the system.

Table 7: Use-Case Definition - Exit Application

|  |  |  |
| --- | --- | --- |
| Flow Controller | Main Flow Step | Alternate Flow Step |
| Precondition | Host is currently in Title window. |  |
| Host | Chooses to leave game. |  |
| System | Stops execution and releases all its resources. |  |
| Postcondition | Application has stopped |  |
| Error Conditions | Application does not stop. |  |
|  | Application does not release all resources. |  |

C:\Users\Scott-Sterling\Desktop\exitApplicationFlow.png

Figure 6: Scenario Flow Diagram - Exit Application

## Player Scenarios

The following are all Scenarios where the primary Actor is the Player of the game. They involve interacting with the system primarily to perform game actions or to inform the Player so that better strategies can occur.

### Primary Scenario - Take Turn

The Take Turn Scenario is a Primary Scenario that applies when a Player’s Robot becomes active. Once the Player’s turn has begun, the Player has the option to perform a certain combination of move piece, shoot, and inspect space actions. While it is the player’s turn, the Player can perform the navigate board action, view game history log action, and monitor their own statistics. At any time, the player may choose to not perform any more actions and end the turn.

Table 8: Use-Case Definition - Take Turn (Player)

|  |  |  |
| --- | --- | --- |
| Flow Controller | Main Flow Step | Alternate Flow Step |
| Precondition | A Player’s Robot is currently the Active Robot. |  |
| Player | May perform any of the alternate actions listed to the right. These can be done in any order. The only limitations are the preconditions of the alternate actions. | Move Action  Shoot Action  Inspect Space Action  Navigate Board Action  View Game History Log Action |
| Player | Chooses to End Turn. | Time runs out for the turn. |
| System |  | System ends the turn. |
| Postcondition | The Player’s turn has ended. |  |
| Error Conditions | The Player’s turn does not end. |  |
|  | The Player cannot end the turn. |  |

C:\Users\Scott-Sterling\Desktop\takeTurn.png

Figure 7: Scenario Flow Diagram - Take Turn (Player)

#### Secondary Scenario - Move Robot

The Move Robot Scenario is a Secondary Scenario that can apply when the Player is in the Take Turn Scenario. When a player selects a space within range of its robot’s remaining mobility points and selects “Move” from the context menu that appears as a result of selecting the space, the robot is moved to the selected space, the robot’s remaining mobility points are reduced by the cost of the movement, and the history log and what can be seen by the robot are updated as necessary.

Table 9: Use-Case Definition - Move Robot

|  |  |  |
| --- | --- | --- |
| Flow Controller | Main Flow Step | Alternate Flow Step |
| Precondition | A Player’s Robot is currently the Active Robot.  The Robot can still move. |  |
| System | Displays the movable range of the Active Robot. |  |
| Player | Chooses where to move the Active Robot. |  |
| System | Asks whether the Player wants to Move, Shoot, or Inspect. |  |
| Player | Chooses to Move. | Chooses another Option. |
| System | Moves the Active Robot and updates the Log and Stats. |  |
| Postcondition | Active Robot has Moved |  |
| Error Conditions | Active Robot does not Move. |  |
|  | Active Robot moves incorrectly. |  |

C:\Users\Scott-Sterling\Desktop\moveRobot.png

Figure 8: Scenario Flow Diagram - Move Robot

#### Secondary Scenario - Shoot

The Shoot Scenario is a Secondary Scenario which applies when the Player is in the Take Turn Scenario. When a player selects a space within range of its robot’s range points and selects “Shoot” from the context menu that appears as a result of selecting the space, the selected space is shot at, the remaining health of any robots occupying the space is reduced by the attack points of the player’s robot, and the game history log and the live robots are updated as necessary.

Table 10: Use-Case Definition - Shoot

|  |  |  |
| --- | --- | --- |
| Flow Controller | Main Flow Step | Alternate Flow Step |
| Precondition | A Player’s Robot is currently the Active Robot.  The Robot can still shoot. |  |
| Player | Chooses where to shoot. |  |
| System | Checks whether the Active Robot can shoot there. |  |
| System | If can, damages all Robots in that location and updates log and stats. | If not, displays that it cannot. |
| Postcondition | Active Robot has shot. |  |
| Error Conditions | Active Robot does not shoot when it can. | Active Robot shoots when it cannot. |
|  | Log/stats not updated properly. |  |

C:\Users\Scott-Sterling\Desktop\shoot.png

Figure 9: Scenario Flow Diagram - Shoot

#### Secondary Scenario - Inspect Space

The Inspect Space Scenario is a Secondary Scenario which can apply while the Player is in the Take Turn scenario. When the player selects a space and selects “Inspect” from the context menu that appears as a result of selecting the space, the system will display information and statistics regarding units occupying the selected space in another window.

Table 11: Use-Case Definition - Inspect Space

|  |  |  |
| --- | --- | --- |
| Flow Controller | Main Flow Step | Alternate Flow Step |
| Precondition | A Player’s Robot is currently the Active Robot. |  |
| Player | Chooses where to inspect. |  |
| System | Checks whether the Player has visibility. |  |
| System | If so, shows all stats and data related to the space. | If not, displays that it cannot. |
| Postcondition | Information is displayed. |  |
| Error Conditions | Information is not shown. | Information shown. |
|  | Incorrect information shown. |  |

C:\Users\Scott-Sterling\Desktop\inspect.png

Figure 10: Scenario Flow Diagram - Inspect Space

#### Secondary Scenario - End Turn

The End Turn Scenario is a Secondary Scenario that can apply when a Player does not want or is not able to perform more actions in a Turn. When a player clicks the “End Turn” button after some combination of moving, shooting, and viewing the board or the player’s time limit has expired, the system ends the player’s turn and selects the next player and robot.

Table 12: Use-Case Definition - End Turn

|  |  |  |
| --- | --- | --- |
| Flow Controller | Main Flow Step | Alternate Flow Step |
| Precondition | A Player’s Robot is currently the Active Robot. |  |
| Player | Chooses to end turn and Confirms. | Time Limit Expires. |
| System | Ends the player’s turn, updates log, and activates the next Robot. |  |
| Postcondition | The next Robot is activated. |  |
| Error Conditions | Turn does not end. |  |
|  | No Robot is activated. |  |
|  | Incorrect Robot is activated. |  |

C:\Users\Scott-Sterling\Desktop\endTurn.png

Figure 11: Scenario Flow Diagram - End Turn

### Primary Scenario - Navigate Board

The Navigate Board Scenario is a Primary Scenario that applies at any time if a Player has access to the game board. This can be done between other Scenarios or even in the middle of another Scenario. When the Player wishes to see a part of the game board that is not currently visible in their main game window, the Player will execute a combination of pan and zoom actions to bring the desired region of the game board into their view.

Table 13: Use-Case Definition - Navigate Board

|  |  |  |
| --- | --- | --- |
| Flow Controller | Main Flow Step | Alternate Flow Step |
| Precondition | A Game is currently being played. |  |
|  | A Player has access to the game board. |  |
| Player | May perform any of the alternate actions listed to the right. These can be done in any order. The only limitations are the preconditions of the alternate actions. | Pan Board  Zoom Board |
| Player | Chooses to stop navigating. | Time runs out for the turn. |
| Postcondition | The Player has the selected view of the board. |  |
| Error Conditions | The Player cannot pan. |  |
|  | The Player cannot zoom. |  |

C:\Users\Scott-Sterling\Desktop\navigateBoard.png

Figure 12: Scenario Flow Diagram - Navigate Board

#### Secondary Scenario - Pan Board

The Pan Board Scenario is a Secondary Scenario that can apply when the Player is in the Navigate Board Scenario. After a player clicks the “Pan Board” button, the player enters the “Pan Board” mode, presses the mouse up against one of the edges of the screen to indicate the direction to move their view, and the system moves the player’s view in the indicated direction. The arrow keys may also be used to indicate the direction of movement.

Table 14: Use-Case Definition - Pan Board

|  |  |  |
| --- | --- | --- |
| Flow Controller | Main Flow Step | Alternate Flow Step |
| Precondition | Player is in Navigate Board Scenario. |  |
| Player | Chooses to Pan. |  |
| Player | Moves Mouse out of the viewable board area. |  |
| System | Move viewable board area in the direction of Mouse. |  |
| Player | Chooses to stop panning. | Time runs out for the turn. |
| Postcondition | The Player has the selected view of the board. |  |
| Error Conditions | The Player cannot pan. |  |
|  | Pans incorrectly. |  |

C:\Users\Scott-Sterling\Desktop\panScreen.png

Figure 13: Scenario Flow Diagram - Pan Screen

#### Secondary Scenario - Zoom Board

The Zoom Board Scenario is a Secondary Scenario that can apply when the Player is in the Navigate Board Scenario. After the player clicks the “Zoom View” button, the player enters the “Zoom View” mode, the player adjusts the zoom of their view to their liking using the “+” button to indicate the view is to be zoomed in and “-” button to indicate the view is to be zoomed out. The system saves the player’s zoom level after the player clicks the “Exit Zoom Mode” button.

Table 15: Use-Case Definition - Zoom Board

|  |  |  |
| --- | --- | --- |
| Flow Controller | Main Flow Step | Alternate Flow Step |
| Precondition | Player is in Navigate Board Scenario. |  |
| Player | Chooses to Zoom. |  |
| Player | Selects Zoom Level. |  |
| System | Zooms to selected level. |  |
| Player | Chooses to stop zooming. |  |
| Postcondition | The Player has the selected view of the board. |  |
| Error Conditions | The Player cannot zoom. |  |
|  | Zooms incorrectly. |  |

C:\Users\Scott-Sterling\Desktop\zoomScreen.png

Figure 14: Scenario Flow Diagram - Zoom Board

### Primary Scenario - View Game Log

The View Game Log Scenario is a Primary Scenario that can apply when the Player has access to the game board. Once the player toggles the game history log on, a small window in the corner of the main game screen is displayed containing all actions for all player throughout the current game. The window can be scrolled to view older or new actions.

Table 16: Use-Case Definition - View Game Log

|  |  |  |
| --- | --- | --- |
| Flow Controller | Main Flow Step | Alternate Flow Step |
| Precondition | A Game is currently being played. |  |
|  | A Player has access to the game board. |  |
| Player | Chooses to view log. |  |
| System | Shows log. |  |
| Player | Chooses to stop viewing log |  |
| System | Closes log. |  |
| Postcondition | The game board is displayed. |  |
| Error Conditions | The game log does not toggle correctly. |  |
|  | The game log shows incorrect information. |  |

C:\Users\Scott-Sterling\Desktop\viewGameLog.png

Figure 15: Scenario Flow Diagram - View Game Log

### Primary Scenario - View Robot Statistics

The View Robot Statistics Scenario is a Primary Scenario that can apply when the Player has access to the game board. When the player clicks the “View Stats” button, another window appears displaying the current statistics of the player’s robots. The statistics consist of a robot’s remaining health, remaining mobility points, range points, attack points, damage dealt, number of kills, and distance moved.

Table 17: Use-Case Definition - View Robot Statistics

|  |  |  |
| --- | --- | --- |
| Flow Controller | Main Flow Step | Alternate Flow Step |
| Precondition | A Game is currently being played. |  |
|  | A Player has access to the game board. |  |
| Player | Chooses to view Robot Statistics. |  |
| System | Shows Statistics for all known Robots. |  |
| Player | Chooses to stop viewing Robots. |  |
| System | Hides Statistics for Robots. |  |
| Postcondition | The game board is displayed. |  |
| Error Conditions | The Robot Statistics do not toggle correctly. |  |
|  | The Robot Statistics shows incorrect information. |  |

C:\Users\Scott-Sterling\Desktop\viewRobotStatistics.png

Figure 16: Scenario Flow Diagram - View Robot Statistics

### Primary Scenario - End Game

The End Game Scenario is a Primary Scenario that can apply when the Player has access to the game board. When a Player clicks the “End Game” button in the top-left corner of the main game window, a confirmation window appears prompting the player to confirm their decision to end the game. The game is ended when the user clicks the “Yes” button in the confirmation window. The game continues if the player clicks the “Cancel” button in the confirmation window.

Table 18: Use-Case Definition - End Game

|  |  |  |
| --- | --- | --- |
| Flow Controller | Main Flow Step | Alternate Flow Step |
| Precondition | A Game is currently being played. |  |
|  | A Player has access to the game board. |  |
| Player | Chooses to End Game. |  |
| System | Asks for confirmation |  |
| Player | Confirms | Cancels |
| System | Ends Game for All Players. |  |
| Postcondition | The game is over and the Title Screen appears. |  |
| Error Conditions | No Confirmation Screen | Game Ends. |
|  | Game Does not End |  |

C:\Users\Scott-Sterling\Desktop\endGame.png

Figure 17: Scenario Flow Diagam - End Game

## Spectator Scenarios

The Spectator Scenarios are comprised of many of the same Scenarios as the Player’s Scenarios. Specifically, the Spectator can initiate that Navigate Board, View Game Log, and View Robot Statistics Scenarios. The difference between a Spectator and a Player in these Scenarios is that the Spectator can see everything, whereas the Player can only see a subsection of the game board.

### 4.4.1 Primary Scenario - Navigate Board (Spectator)

The Spectator’s Navigate Board Primary Scenario is identical to the Player’s Navigate Board Primary Scenario as listed in 4.3.2, but can see all details.

### 4.4.2 Primary Scenario - View Game Log (Spectator)

The Spectator’s View Game Log Primary Scenario is identical to the Player’s View Game Log Primary Scenario as listed in 4.3.3, but can see all details.

### 4.4.3 Primary Scenario - View Robot Statistics (Spectator)

The Spectator’s View Robot Statistics Primary Scenario is identical to the Player’s View Robot Statistics Primary Scenario as listed in 4.3.4, but can see all details.

## Robot Scenarios

The Robot Actor has only a single Scenario: the Take Turn Primary Scenario. It is different from the Player Actor’s Take Turn Primary Scenario in that it consists of interpreting the commands of the Robot Actor and performing them instead of waiting for decisions to be made.

### Primary Scenario - Take Turn (Robot Actor)

The Robot Actor’s Take Turn Scenario is a Primary Scenario that applies when an AI’s Robot is the Active Robot. Once the AI’s turn has begun, the system will read the AI’s program and determine which actions to perform based on the program provided to the system by the robot file. The AI can perform the move piece, shoot, and inspect space actions.

Table 19: Use-Case Definition - Take Turn (Robot Actor)

|  |  |  |
| --- | --- | --- |
| Flow Controller | Main Flow Step | Alternate Flow Step |
| Precondition | An AI’s Robot is currently the Active Robot. |  |
| AI | Sends commands to the System one at a time for the System to Execute. |  |
| System | Executes command in the order they are received. |  |
| Postcondition | The AI’s turn has ended. |  |
| Error Conditions | System cannot understand AI commands. |  |
|  | System incorrectly applies AI commands. |  |

C:\Users\Scott-Sterling\Desktop\robotTakeTurn.png

Figure 18: Scenario Flow Diagram - Take Turn (Robot)

## Robot Librarian Scenarios

The Robot Librarian Actor has six (6) Scenarios associated with it. It handles the different lists of commands (programs) available to the Host for use with AIs.

### Primary Scenario: Enumerate

The Enumerate Scenario is a Primary Scenario that applies when the Robot Librarian is asked to list the available programs that have certain features. When this happens, the Robot Librarian will find all programs with that feature, and send them back for display.

Table 20: Use-Case Definition: Enumerate

|  |  |  |
| --- | --- | --- |
| Flow Controller | Main Flow Step | Alternate Flow Step |
| Precondition | Robot Librarian is initialized. |  |
| System | Sends query to Robot Librarian |  |
| Robot Librarian | Processes and applies query. |  |
| Robot Librarian | Gives records |  |
| Postcondition | Records are displayed. |  |
| Error Conditions | Incorrect records found. |  |
|  | Query not understood. |  |

C:\Users\Scott-Sterling\Desktop\enumerate.png

Figure 19: Scenario Flow Diagram - Enumerate

### Primary Scenario - Download

The Download Scenario is a Primary Scenario that applies when the Robot Librarian needs to find more programs to include in its library. It searches through a file structure until it finds a program to add, and then adds it to its library.

Table 21: Use-Case Definition - Download

|  |  |  |
| --- | --- | --- |
| Flow Controller | Main Flow Step | Alternate Flow Step |
| Precondition | Robot Librarian is initialized. |  |
| System | Sends request to Robot Librarian for more programs |  |
| Robot Librarian | Searches file structure for programs. |  |
| Robot Librarian | Stores any programs found in library. |  |
| Postcondition | Programs that are found are stored. |  |
| Error Conditions | Found programs not stored. |  |

C:\Users\Scott-Sterling\Desktop\download.png

Figure 20: Scenario Flow Diagram - Download

### Primary Scenario - Update Statistics

The Update Statistics Scenario is a Primary Scenario that applies when the Robot Librarian is asked to update the Statistics for a given program. When asked, the Robot Librarian takes the new information about the program and adds it to the current data about the Program. It then updates the statistics using all of the data.

Table 22: Use-Case Definition - Update Statistics

|  |  |  |
| --- | --- | --- |
| Flow Controller | Main Flow Step | Alternate Flow Step |
| Precondition | Robot Librarian is initialized. |  |
| System | Sends request to Robot Librarian to update statistics for a Program. |  |
| Robot Librarian | Adds new data about Program. |  |
| Robot Librarian | Updates statistics about Program and confirms update. |  |
| Postcondition | Program Statistics updated. |  |
| Error Conditions | Statistics not Updated. |  |
|  | Confirmation not returned. |  |

C:\Users\Scott-Sterling\Desktop\updateStatistics.png

Figure 21: Scenario Flow Diagram - Update Statistics

### Primary Scenario: Register

The Register Scenario is a Primary Scenario that applies when the Robot Librarian is asked to create a new Statistics record for a given program. When asked, the Robot Librarian takes the request and creates a new Statistics record and associates it with the Program. It then confirms the creation of the Record.

Table 23: Use-Case Definition - Register

|  |  |  |
| --- | --- | --- |
| Flow Controller | Main Flow Step | Alternate Flow Step |
| Precondition | Robot Librarian is initialized. |  |
| System | Sends request to Robot Librarian to create a new Record for a Program |  |
| Robot Librarian | Creates Record |  |
| Robot Librarian | Confirms creation |  |
| Postcondition | Record created. |  |
| Error Conditions | Record not Created. |  |
|  | Record created for wrong Program. |  |

C:\Users\Scott-Sterling\Desktop\register.png

Figure 22: Scenario Flow Diagram - Register

### Primary Scenario: Revise

The Register Scenario is a Primary Scenario that applies when the Robot Librarian is asked to change which Program is associated with a given Record. When asked, the Robot Librarian takes the request and changes the owner of the Record to the given Program. It then confirms the revision of the Record.

Table 24: Use-Case Definition - Revise

|  |  |  |
| --- | --- | --- |
| Flow Controller | Main Flow Step | Alternate Flow Step |
| Precondition | Robot Librarian is initialized. |  |
| System | Sends request to Robot Librarian to change which Program is associated with a Record. |  |
| Robot Librarian | Finds Record and changes Program ownership. |  |
| Robot Librarian | Confirms revision. |  |
| Postcondition | Record revised. |  |
| Error Conditions | Record not revised. |  |
|  | Record revised incorrectly. |  |

C:\Users\Scott-Sterling\Desktop\revise.png

Figure 23: Scenario Flow Diagram - Revise

### Retire

The Register Scenario is a Primary Scenario that applies when the Robot Librarian is asked to disassociate a Record from the Program with which it is associated. When asked, the Robot Librarian takes the request and removes the link between the Record and the Program. It then confirms the removal of the Record.

Table 25: Use-Case Definition - Retire

|  |  |  |
| --- | --- | --- |
| Flow Controller | Main Flow Step | Alternate Flow Step |
| Precondition | Robot Librarian is initialized. |  |
| System | Sends request to Robot Librarian to remove a Program associated with a given Record. |  |
| Robot Librarian | Finds Record and removes Program ownership. |  |
| Robot Librarian | Confirms removal. |  |
| Postcondition | Record removed. |  |
| Error Conditions | Record not removed. |  |

C:\Users\Scott-Sterling\Desktop\retire.png

Figure 24: Scenario Flow Diagram - Retire

# 5.0 Storyboards and User Interface

## 5.1 Overview

The storyboards and user interface elements for this application represent the expected method of interacting with the application over time. The interface elements will be discussed first, and then a couple of storyboards for interactions among the elements will be shown.

## 5.2 Interface Elements

The interface elements for this application describe the notional GUI elements determined to be necessary from the scenarios in section 4. The ones shown below are currently nonfunctional, but should provide a reasonable degree of comfort in how interaction with the application should occur.

### 5.2.1 Main Menu Screen

The user sees the title screen when he/she opens the game. The title screen welcomes the user to the game, displays the name and logo and offers three options; namely, "PLAY GAME", "ROBOT ARCHIVE", and "EXIT GAME".

These options are presented in the form of buttons. The "PLAY GAME" button leads the user to the "Player Selection" screen, which is discussed below.

The "ROBOT ARCHIVES" button allows the user to view past records of the games, including information such as \_\_\_\_\_.

The function of the "EXIT" button is the same as the "Close [X]" button on the top right corner of the window. Typically, the user would click on the "PLAY GAME" button to start a new game. The system will proceed to the "Player Selection" screen.

C:\Users\Scott-Sterling\Desktop\titleScreen.png

Figure 25: Robo-Wars Title Screen

### 5.2.2 Player Select Screen

The "Player Selection" screen allows the user to set up the game. The user would select either

a) a two-player game,

b) a three-player game, or

c) a six-player game.

The user is restricted to select exactly one of these options.

Next, the user inputs the names of the players for the provided number of players. A selection of two or three players disables four or three of the input fields respectively. Moreover, the user is also presented with options to choose between human player and AI player for each individual player, besides their name. At the bottom of the screen, there are three options, namely

1. “Back” button, which the user can pick if he or she chooses to not go ahead with a new game,
2. “Options” button, allowing users to select options specific to the game, and

“Play” button, leading the user to the Game screen.

C:\Users\Scott-Sterling\Desktop\selectPlayers.png

Figure 26: Player Selection Screen

#### 5.2.2.1 Robot Team Select Dialog

C:\Users\Scott-Sterling\Desktop\robotSelection.png

Figure 27: Robot Team Select Screen

### 5.2.3 Game Options Screen

If the user wants to explore options offered for the game, he or she may choose to click on the “Options” button, which would lead him or her to the Options screen.

The Options screen offers the user to choose between various options for the game, such as \_\_\_\_\_.

Once the user is finished making changes to the screen, he or she will exit back into the “Player Selection” screen, if he or she wishes to make more changes to the player selections. This shall be done by clicking on an “OK” button at the bottom right of the screen.

C:\Users\Scott-Sterling\Desktop\gameOptions.png

Figure 28: Game Options Screen

### 5.2.4 Game Screen

Pressing on the “START GAME” button in the “Player Selection” screen leads the user to the game. (Now, we shall refer to the player as a player, instead of a User. )

The game screen has various noticeable features. First, a long, horizontal panel on the top shows thumbnails of all the players, with their names and color to indicate the specific player. If there is a player who has quit the game or his robots are all dead, their thumbnail on the panel is greyed out and has a red cross across it. The thumbnail of the current player is always going to be on the top right corner, with additional information displayed such as the powers, and next-active robot. The game board also has a quit game button on the top left corner of the screen. On the left side, there are optional buttons for zooming in, zooming out, and toggle between pan and click. The bottom right corner has an expandable window of the log generated in the game, which the user may use to refer to the game history.

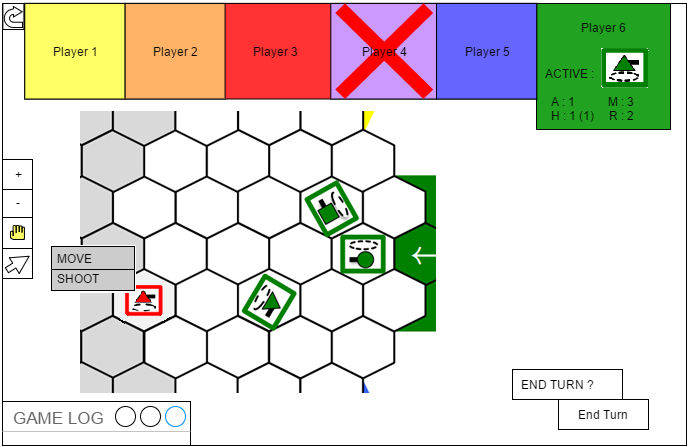
The game board is oriented around the active piece of the player by default. When the player wants to move or shoot, they click on the particular hexagon, and the interface offers the player to choose between move and shoot. When the player wants to end their turn, they simply click on the “End Turn” button on the bottom right corner of the screen. The interface then asks the player to confirm their decision, additionally informing them of the number of moves they have left. Once the player confirms the decision to end their turn, the screen blanks out and displays an end of turn message, and asks for the next player (identifying the player by their name and colour) to take the Hotseat. In case of an AI player, the human player would have to wait until the AI player makes its move, at the end of which, the next human player is notified about their move.

Figure 29: Main Game Screen

### 5.2.5 Match Results Screen

Once the game is over, a dialog box pops up declaring the winner. The player is then lead to see the Results screen.

The Results screen displays the statistics of each robot of each player in a very compact yet well-organized manner. The User can pick the player based on their color, and view the statistics of each robot. There are three option buttons at the bottom: the ‘Go Back to Title Screen’, ‘View Log’, and the ‘Rematch’ button.

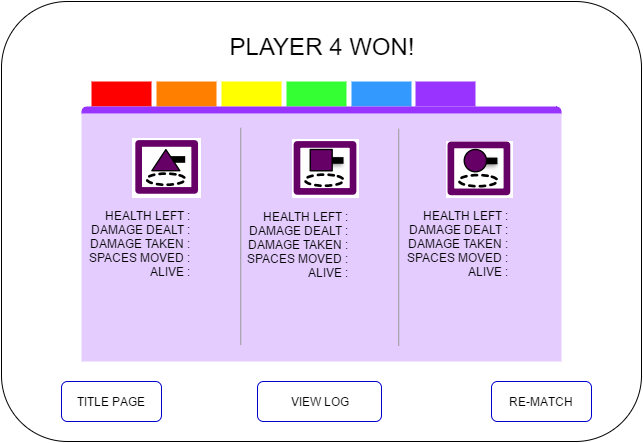


Figure 30: Match Results Screen

### 5.2.6 Robot Archive Screen

The Robot Archive screen offers the user to search for a particular robot, and sort the list of robots by order. The information about a particular robot is shown on the right side when a particular robot is selected. The “Register”, “Revise”, and “Retire” buttons are also displayed below the statistics.

C:\Users\Scott-Sterling\Desktop\robotArchive.png

Figure 31: Robot Archive Screen

### 5.2.7 Game Settings Screen

The settings screen on the title page primarily offers the user with two options. The first option is for the sound, which is a slider. The other option is for key mapping, for advanced players who prefer to pick keys. There is a ‘Save’ and a ‘Back’ button at the bottom of the screen.

C:\Users\Scott-Sterling\Desktop\settings.png

Figure 32: Game Settings Screen

## 5.3 Storyboard Walkthroughs of Typical Use-Cases

# 6.0 Non-Functional Requirements

## 6.1 Overview

In addition to the functional requirements above, this project has several data, hardware and software requirements that must be considered.

## 6.2 Data Requirements

Robot records must conform to a standardized format, so that users may download external Robot records from other sources and integrate them into the program if desired. The Robot records will be stored as JSON-encoded text files (.json). Each will contain a Robot's statistics and identifier (name and team), as well as the AI program that the Robot will run during its matches. The software must be able to load these files from disk and parse them so that they may be used within the game.

## 6.3 Hardware Requirements

This project is designed to run on Windows machines which are running the Windows 7 64-bit operating system. This means that the target machines should be equipped with at least a 1 gigahertz (GHz) processor, 2 gigabytes (GB) of RAM and graphics hardware that supports DirectX 9.

This project also requires that the user has mouse and keyboard peripherals connected to the machine, and that the keyboard is using a QWERTY configuration.

## 6.4 Software Requirements

This project will be constructed using the Java language, version 1.8.0\_101. The user must have Java version 1.8.0\_101 installed on the machine in order to run the application.

# 7.0 Potential Extensions to System

## 7.1 Overview

This section covers game features and mechanics that are requirements of the system, but may be added to the application given sufficient resources and time.

## 7.2 Additional Game Mechanics

Robo-Wars has a standard set of rules, but adding methods of customizing matches would add variety to the play experience. For example, a Host could alter the rules of the game to change the number or type of units given to each team, or change the map that the game is played on.

New game maps or different numbers of units could be implemented in the game system without impacting the flow of Player actions such as taking a turn. Options would need to be added to the Game Options screen in order to allow the Host to specify the new rules.

Other types of game mechanics, such as adding new types of units altogether, are possible but might require changes to the flow of Player actions.

### 7.2.1 Use Case: Changing the Game Options

From the Player Select Screen, the Host would click the “Options” button, and the System would display the Game Options Screen. The Host would then change the “Rules” parameter from “Normal” to “Special” by selecting the option from the dropdown menu, and the System would display controls for any new options implemented, such as new maps or units. The Host could change these, and then would click the “Save” button to apply them to the current match. The System would save these preferences and then display the Player Select Screen.

## 7.3 Sound and Music

Sound effects and music can add aesthetic value to a game and make user interfaces easy to understand by providing aural feedback. Adding sound and music to the program would make the application more appealing to users.

The cost of implementing sound into the project would be primarily test-related. Each action would need to be tested carefully to make sure that the correct sound would be played at the correct volume, without overlapping or other artifacts. In order to not upset the user, general options would need to be added to the Options Screen in order to mute or adjust the volume of sound effects and music within the application.

### 7.3.1 Use Case: Adjusting Volume

In order to adjust the volume, the Host would click “Options” from the Main Menu Screen. The system would respond by displaying the Options Screen, where the Sound Effect Volume and Music Volume options would be displayed. The Host would then click on the sliders associated with the volume and drag it left or right to adjust. When satisfied, the Host would click on the “Save” button and the System would save the new values, and finally display the Main Menu Screen.

## 7.4 Asynchronous Play System

Robo-Wars is designed to be played in Hotseat mode, but its turn-based nature makes it a good candidate for an asynchronous play system. In an asynchronous system, a Player would connect with other Players over a network to play a match, but instead of maintaining a constant network connection, each Player would take his turn and upload the result to a central system. He would then be free to exit the game, and could return later to download his opponents’ moves and take his next turn.

Unlike in the Hotseat mode, an asynchronous system would allow a Player to be participating in several games simultaneously. The Player would need to log in to his profile, and then would be able to enter any of the games he is currently registered in.

An asynchronous system would add significant development time and complexity to the project. A new Server actor would be necessary in order to fulfill the network functions. The project would require a central server, an authentication protocol and an enhanced testing plan. A “Continue Game” button would need to be added to the Main Menu Screen, and two new views would be required: a Login Screen and a Lobby Screen.

### 7.4.1 Use Cases

#### 7.4.1.1 Starting a Match

In order to begin a match, the Host would click the “Continue Game” button on the Main Menu Screen. The System would display the Login Screen, and prompt the Host to enter a username and password. The System would send the details to the Server for authentication, and on receiving a reply, would display the Lobby Screen for the Host.

The Host would click the “New Game” button on the Lobby Screen, and would be prompted to enter the names of users he would like to invite to a new game. The System would transmit these requests to the Server, and the Server would transmit these to the game clients of the other Hosts. Once all the Players had agreed to the invitation, the System would display the Game Screen.

#### 7.4.1.2 Taking a Turn and Ending a Turn

If it is the current Player’s turn, the act of taking the turn is identical to the use case described in Section 3.3.1. Once the Player takes action to end his turn, a waiting screen is displayed. The Player may click a button to go back to the Lobby Screen. The player may click “Update” to update the status of all matches in progress in the lobby. Once the Player is able to take a turn again, a “Ready” notification will appear next to the game on the Lobby Screen.

# Appendices:

## Appendix A: References

### -JSON reference

## Appendix B: Robot Wars Rules

# Index

Match, iii, iv, v, 3, 8, 23, 27