Robo-Wars: Requirements Document

Team: D4

Nickolas Gough, Scott Hebert, Janelle Hindman, Yige Huang, and Tushita Patel

Project: Robo-Wars

Dr. Christopher Dutchyn

CMPT 370 Term I 2016/2017

Date: 10/02/2016

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.
0.99	2016/10/02 9:25 PM	Janelle	Editing Pass
0.995	2016/10/02 9:48 PM	Scott	Added section 7.5, edited Tables
1.000	2016/10/02 10:54 PM	Scott / Janelle	Finished Edits, added Index

Contents

Revision History ii
Executive Summary 1
1.0 Document Overview 2
1.1 Purpose of Document 2
1.2 Overview of Document 2
1.3 Terms and Definitions 2
1.3.1 UML-Related Terms 2
1.3.1.1 Actor 2
1.3.1.2 Action 2
1.3.1.3 Use case 2
1.3.1.4 Scenario 2
1.3.1.5 Graphical User Interface (GUI) 3
1.3.2 Game-Specific Terms 3
1.3.2.1 Match 3
1.3.2.2 Turn 3
1.3.2.2 Round 3
1.3.2.3 Hotseat 3
1.3.2.4 Program 3
1.3.2.5 Robot Record 3
1.3.2.4 Artificial Intelligence (or AI) 3
2.0 Background 3
2.1 Team Introductions 4
2.1.1 Tushita Patel 4
2.1.2 Yige Huang 4
2.1.3 Janelle Hindman 4
2.1.4 Scott Hebert 4
2.1.5 Nickolas Gough 4
2.2 Product Introduction 4
2.2.1 Overview of the RoboSport370 Board Game 5
2.2.2 Expected Benefits of the Computerized Interface 5
2.2.3 Product Implementation 5
2.3 Assumptions 5

2.3.1 Assumption of Knowledge of Game Rules 5
3.0 Actors and their Relationships 5
3.1 Overview 5
3.2 Actors 6
3.2.1 Host 6
3.2.2 Player 6
3.2.3 Spectator 6
3.2.4 AI 6
3.2.5 Robot Librarian 7
3.3 Relationships among Actors 7
3.4.1 System Diagram 8
4.0 Scenarios 8
4.1 Overview 8
4.2 Host Scenarios 9
4.2.1 Primary Scenario: Start New Game 9
4.2.2 Primary Scenario: View Match Results 10
4.2.3 Primary Scenario: View Robot Archive 11
4.2.4 Primary Scenario - Change Settings 13
4.2.5 Primary Scenario - Exit Application 14
4.3 Player Scenarios 14
4.3.1 Primary Scenario - Take Turn 14
4.3.2 Primary Scenario - Navigate Board 19
4.3.3 Primary Scenario - View Game Log 21
4.3.4 Primary Scenario - View Piece Statistics 22
4.3.5 Primary Scenario - End Game 23
4.4 Spectator Scenarios 24
4.4.1 Primary Scenario - Navigate Board (Spectator) 24
4.4.2 Primary Scenario - View Game Log (Spectator) 24
4.4.3 Primary Scenario - View Robot Statistics (Spectator) 24
4.4.4 Primary Scenario – End Game (Spectator) 24
4.5 AI Scenarios 24
4.5.1 Primary Scenario - Take Turn (AI) 24
4.6 Robot Librarian Scenarios 25

4.6.1 Primary Scenario - Enumerate 25
4.6.2 Primary Scenario - Download 26
4.6.3 Primary Scenario - Update Statistics 27
4.6.4 Primary Scenario: Register 27
4.6.5 Primary Scenario: Revise 28
4.6.6 Retire 29
5.0 Storyboards and User Interface 30
5.1 Overview 30
5.2 Interface Elements 30
5.2.1 Main Menu Screen 30
5.2.2 Player Select Screen 30
5.2.3 Game Options Screen 32
5.2.4 Turn Transition Screen 32
5.2.5 Game Screen 33
5.2.6 Match Results Screen 35
5.2.7 Robot Archive Screen 36
5.2.8 Game Settings Screen 38
6.0 Non-Functional Requirements 39
6.1 Overview 39
6.2 Data Requirements 39
6.3 Hardware Requirements 39
6.4 Software Requirements 39
7.0 Potential Extensions to System 40
7.1 Overview 40
7.2 Additional Game Mechanics 40
7.2.1 Use Case: Changing the Game Options 40
7.3 Sound and Music 40
7.3.1 Use Case: Adjusting Volume 40
7.4 Asynchronous Play System 41
7.4.1 Use Cases 41
7.5 Tutorial Mode 41
7.5.1 Advisor 41
7.5.2 Tutorial Minigames 42

Appendix A: RoboSport370 Rules ..... - 43 -
Appendix B: Git Tag ..... - 43 -
Index ..... - 1 -

Figure 1: System Diagram of Actors and Actions 8
Figure 2: Scenario Flow Diagram - Start New Game 10
Figure 3: Scenario Flow Diagram - View Match Results 11
Figure 4: Scenario Flow Diagram - View Robot Archive 12
Figure 5: Scenario Flow Diagram: Change Settings 14
Figure 6: Scenario Flow Diagram - Exit Application 14
Figure 7: Scenario Flow Diagram - Take Turn (Player) 15
Figure 8: Scenario Flow Diagram - Move Piece 16
Figure 9: Scenario Flow Diagram - Shoot 17
Figure 10: Scenario Flow Diagram - Inspect Space 18
Figure 11: Scenario Flow Diagram - End Turn 19
Figure 12: Scenario Flow Diagram - Navigate Board 20
Figure 13: Scenario Flow Diagram - Pan Board 20
Figure 14: Scenario Flow Diagram - Zoom Board 21
Figure 15: Scenario Flow Diagram - View Game Log 22
Figure 16: Scenario Flow Diagram - View Piece Statistics 23
Figure 17: Scenario Flow Diagram - End Game 24
Figure 18: Scenario Flow Diagram - Take Turn (AI) 25
Figure 19: Scenario Flow Diagram - Enumerate 26
Figure 20: Scenario Flow Diagram - Download 26
Figure 21: Scenario Flow Diagram - Update Statistics 27
Figure 22: Scenario Flow Diagram - Register 28
Figure 23: Scenario Flow Diagram - Revise 29
Figure 24: Scenario Flow Diagram - Retire 29
Figure 25: Robo-Wars Title Screen 30
Figure 26: Player Selection Screen 31
Figure 27: Robot Team Select Screen 32
Figure 28: Game Options Screen 32
Figure 29: Turn Transition Screen 33
Figure 30: Main Game Screen 34
Figure 31: Match Results Screen 36
Figure 32: Robot Archive Screen 37
Figure 33: Register Robot Dialogue 37
Figure 34: Revise Robot Dialogue 38
Figure 35: Retire Robot Dialogue 38
Figure 36: Game Settings Screen 39

Table 1: Use case Definition - Start New Game 9

Table 2: Use case Definition: Change Game Options 10
Table 3: Use case Definition - View Match Results 11
Table 4: Use case Definition - View Robot Archive 12
Table 5: Use case Definition - Change Robots 13
Table 6: Use case Definition - Change Settings 13
Table 7: Use case Definition - Exit Application 14
Table 8: Use case Definition - Take Turn (Player) 15
Table 9: Use case Definition - Move Piece 16
Table 10: Use case Definition - Shoot 17
Table 11: Use case Definition - Inspect Space 18
Table 12: Use case Definition - End Turn 18
Table 13: Use case Definition - Navigate Board 19
Table 14: Use case Definition - Pan Board 20
Table 15: Use case Definition - Zoom Board 21
Table 16: Use case Definition - View Game Log 21
Table 17: Use case Definition - View Piece Statistics 22
Table 18: Use case Definition - End Game 23
Table 19: Use case Definition - Take Turn (AI) 24
Table 20: Use case Definition: Enumerate 25
Table 21: Use case Definition - Download 26
Table 22: Use case Definition - Update Statistics 27
Table 23: Use case Definition - Register 27
Table 24: Use case Definition - Revise 28
Table 25: Use case Definition - Retire 29

Executive Summary

This is a requirements document for a computerized version of the Robo-Wars game. Intended for younger children, Robo-Wars is a simple and quick board game that can be greatly improved by a computer implementation. Among these improvements are the ability to play by yourself and increased scope for strategy.

The design team for Robo-Wars is made up of both veteran engineers and younger developers. Robo-Wars is implemented in Java, and, while designed for a Windows machine, is theoretically portable to any computer that can run a Java Virtual Machine.

The process of requirement analysis resulting in this document is primarily Unified Modeling Language (UML). Using this process, five (5) actors or roles were identified: the Host, the Player, the AI, the Robot Librarian, and the Spectator. These actors interface through the system through various use cases or scenarios, of which 20 Primary and eight (8) Secondary were identified.

Once identified, user interface elements were developed. Eight (8) separate screens were identified, and flow paths for the proper interaction among the screens were also developed. Certain flows, identified as typical use of the game, have storyboards associated with them.

If these requirements are accepted, a design architecture and comprehensive test plan will be constructed over the next month. Once these are accepted, the code construction will begin. The code should be fully constructed and deployed by the middle of December, 2016 if major delays are not encountered.

1.0 Document Overview

1.1 Purpose of Document

The purpose of this document is to set and inform the expectations of potential customers of the Robo-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:

- Section 1 explains the document itself, as well as its terms and definitions.
- Section 2 introduces the project team, the purpose of the Robo-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 proposes non-essential features 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 Modeling 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

An Actor is a specific role that interacts with the system. One entity, such as a specific human user, might take on several roles in the course of interacting with the system, and could be considered multiple Actors.

1.3.1.2 Action

An Action is a specific way of using the system.

1.3.1.3 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, and may represent an additional or alternate series of steps within a Primary Scenario.

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

A Match is a single game with two or more players, the flow of actions within that game, and its outcome.

1.3.2.2 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

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

Hotseat refers to a type of multiplayer structure where several users interact with the game through the same computer. Each user takes a turn interacting with the system, and turns are scheduled in a predetermined order. This is contrasted with a Network multiplayer structure where each user interacts with the game through a separate computer.

1.3.2.4 Program

A Program is a text file that contains instructions on how to operate a robot game piece using commands.

1.3.2.5 Robot Record

A Robot Record is a JSON-encoded text file that defines a robot in terms of the performance of a Program over time. It stores the name of a robot, the team that created it, and cumulative statistics on how well it has performed in matches, as well as its Program.

1.3.2.4 Artificial Intelligence (or AI)

Artificial Intelligence, or AI, refers to a computer-controlled entity within a Match that acts according to a Program stored in a Robot Record.

2.0 Background

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

2.1 Team Introductions

Team D4 is a five-person R&D team of students at the University of Saskatchewan. Below is a brief biography of each member of the R&D team.

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 worked under Dr. Regan Mandryk in the Human-Computer Interaction (HCI) Lab and under Dr. Chris Soteros in the Department of Mathematics and Statistics at the University of Saskatchewan. When she is not studying in school, Tushita likes to get involved in the community by doing volunteer work.

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.

2.1.3 Janelle Hindman

Janelle Hindman is a third-year Computer Science student at the University of Saskatchewan. She has an insatiable appetite for learning, and takes every opportunity to learn something new and improve herself. Janelle is Canadian, but has lived in Japan for the past nine years, where she contributes to the Japanese game development community. Her specialties are designing game mechanics and graphical user interfaces, and she enjoys developing and playing video games in her free time. With this project, Janelle hopes to learn about the software engineering process and improve her leadership abilities.

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. Scott is the co-owner of a simulation consulting firm, bringing years of experience and a mind full of ideas to this project. 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.

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 "Robo-Wars," a computerized interface and rules engine for the RoboSport370 board game. Robo-Wars is intended to be a colourful adaptation of RoboSport370, targeted at young children, that is simple to play. While full details about the RoboSport370 can be found in Appendix B,

below is a brief introduction to the game. After this introduction, an explanation of the computerized interface and its benefits will be given.

2.2.1 Overview of the RoboSport370 Board Game

RoboSport370 is a multiplayer board game where players attempt to use a team of robots to eliminate all opposing robots. The robots each have a unique combination of attributes, such as mobility and range, which influence how they move around the board and interact with other robots. The objective of RoboSport370 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: RoboSport370 Game Rules.

2.2.2 Expected Benefits of the Computerized Interface

There are four (4) main benefits to Robo-Wars, the Team D4 product, as compared to the paper board game RoboSport370. 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 assumptions made by Team D4.

2.3.1 Assumption of Knowledge of Game Rules

The 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 feature is not required for the product. Robo-Wars should have an interface that makes navigation as simple as possible for the user, but the game system will not explicitly teach the user the rules of the game.

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 AI, 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 matches 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, examining Robot Records stored in the system, 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 a match. 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 game piece if they are capable of doing so, shoot a space with a game piece 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 flow of the match. 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). They may view the statistics of their own game pieces, but not those of the other contestants.

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 game pieces 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 all of the robots in the match.

3.2.4 AI

An AI is a Program-controlled entity that takes actions as one of the three types of game pieces (Scout, Sniper or Tank) within a match. An AI is associated with a Robot Record, may act only according to instructions contained in its Program, and cannot be controlled by any human entity. Each computer-

controlled team may contain up to three AI-controlled 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. In taking its turn, 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 a match, its statistics are tallied and frozen. It can take no further action to change the game system. After a match has ended, the lifetime statistics for the Robot Record associated with the AI may be updated or discarded.

3.2.5 Robot Librarian

The Robot Librarian is a separate system that manages Robot Records. It interacts with the game system in order to provide Robot Records for use in game, and allows itself to be queried by the system. Its primary role is outside of matches; it provides Robot 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 Robot Records 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 Robot 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 Robot Record, revise the Program of an existing Robot Record, and retire a Robot Record in order to free up that Robot Record's name.

The records that the Robot Librarian handles will be JSON-encoded text, and the Programs will be plain text, as described in Section 6.2.

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

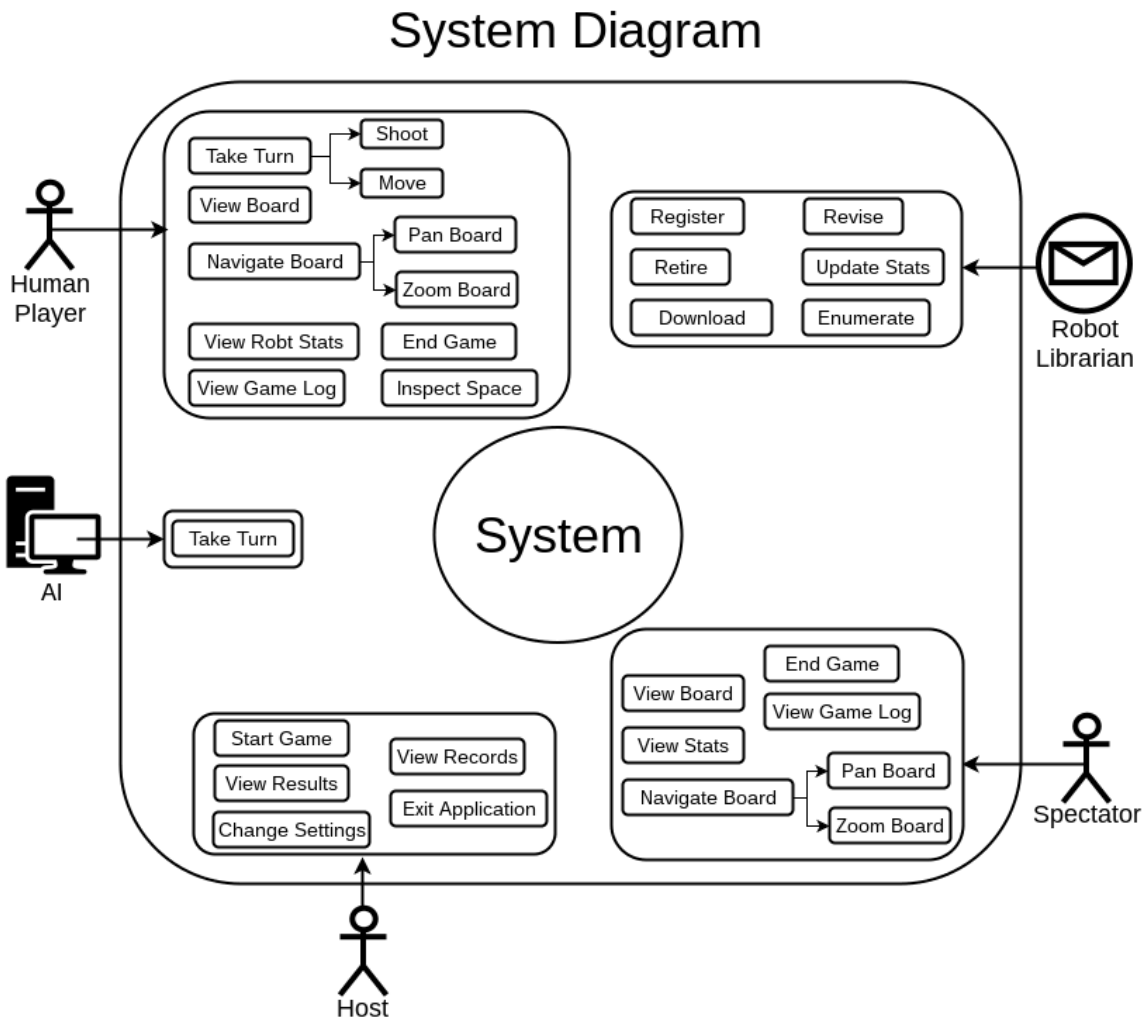

Figure 1: System Diagram of Actors and Actions

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 starting and ending matches.

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 Host clicks the "New Game" button from the main menu, the System displays the Player Selection screen, in which the Host selects the number of players, whether each player is human or AI and which Program the AI uses, enters the name of each player and clicks the "Start Game" button. The System then creates a new match with the specified players and displays the main game screen. From the Player Selection 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	

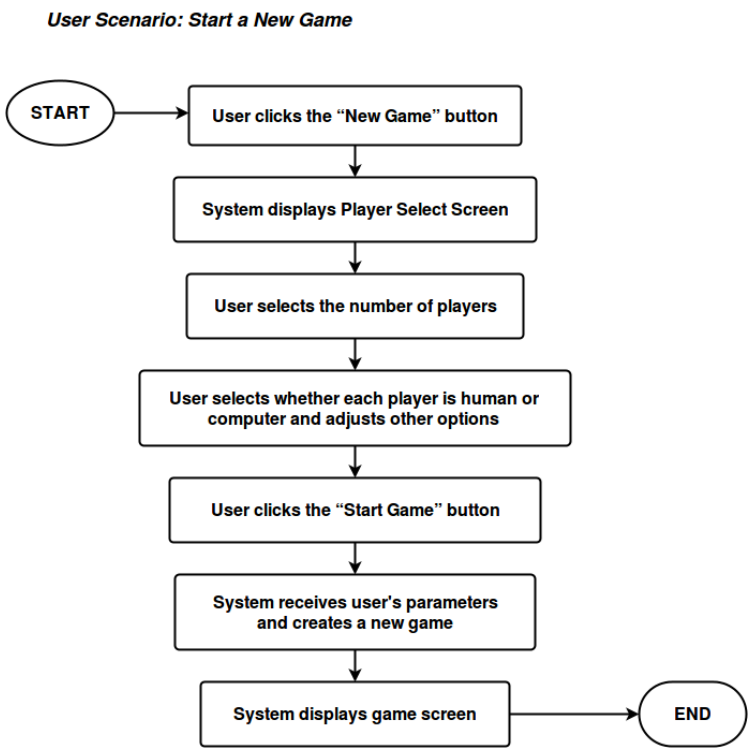

Figure 2: Scenario Flow Diagram - Start New Game

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 Host clicks the "Options" button in the Player Selection screen, the System displays the Game Options screen. The Host may then change any of the game options as desired. The changes will only be applied to the game if the Host clicks the "Save" button. Otherwise, if the user clicks the "Cancel" button, all changes will be discarded. Afterwards, the System will display the Player Selection screen.

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 match has finished. When the match ends, the System displays a small dialog window containing the winner of the match. The

Host clicks the “OK” button and the System displays the Match Result Screen. The Host may examine the statistics for the match by clicking the coloured tabs, and the System display the statistics for that player. The Host may also click the “View Log” button, and the System display the Game Log window, which the Host may close by clicking the “OK” button. The Host clicks the “End Game” button when finished, upon which the System will display the Main Menu. If the Host wishes to play another match with the same settings, the Host may click the “Rematch” button, and the System create a new match with the original settings, and will display the main game screen.

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.	

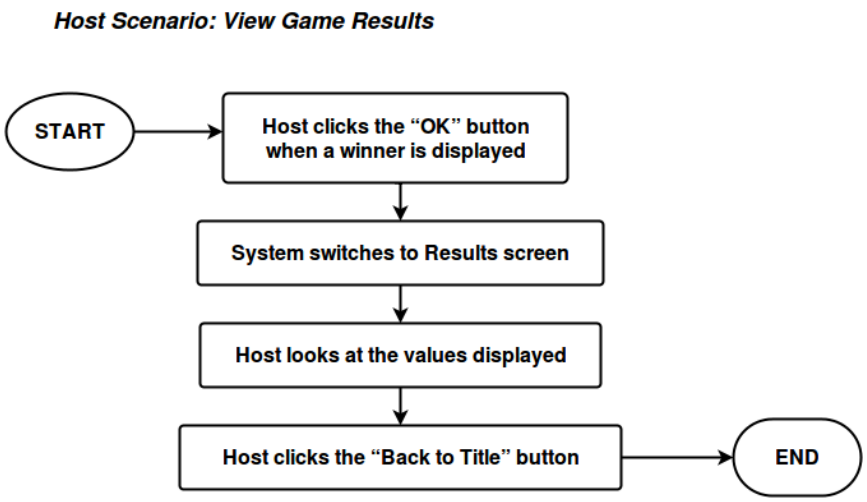

Figure 3: Scenario Flow Diagram - View Match Results

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 Program records stored in the application. From the Main Menu screen, the Host clicks on the “Robot Archive” button, and the System displays the Robot Archive screen. The Host may then make queries by selecting some criteria from the “Sort by…” menu, or by entering a search term into the text box and clicking the search icon. In response, the System will update the list of Programs. The Host may view statistics about a Program by clicking its name in the Program list, and the System will update

the display with the statistics of the selected Program. Once finished with queries, the Host clicks on the "Back" button and the System displays the Main Menu screen.

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 Robot Records.	
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.	

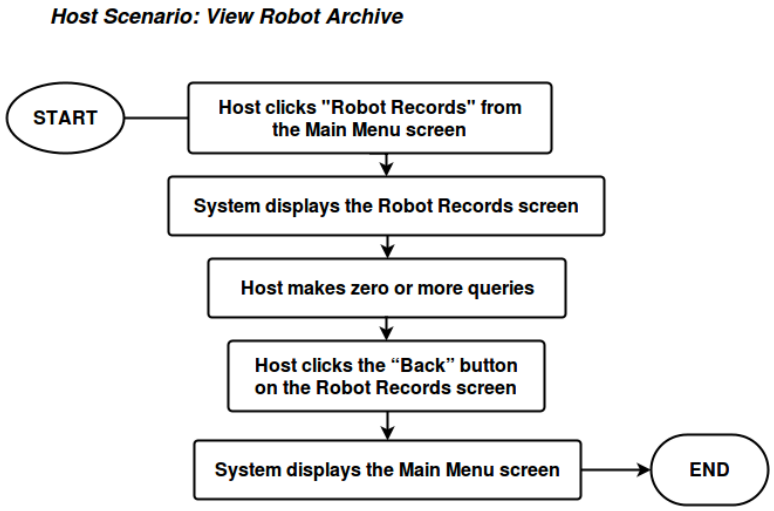

Figure 4: Scenario Flow Diagram - View Robot Archive

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 Programs registered in the system while in the View Robot Archive Scenario. There are three possibilities for changing Programs: adding a new Program, changing a current Program, or removing an old Program. To add a new Program, the Host clicks the "Register" button, and the System displays a dialogue box prompting the Host to enter a name and select a file from their file system. The Host locates and selects their desired file, and clicks the "Register" button, and the System processes the query and closes the dialogue box, displaying the Robot Archive screen. To change a Program, the Host clicks the name of the Program to change, followed by the "Revise" button, and the System displays a dialogue box prompting the Host select a file from their file system. The Host locates and selects their desired file, and clicks the "Revise" button, and the System processes the query and closes the dialogue box, displaying the Robot Archive screen. To remove a Program, the Host clicks the name of the Program

to remove, followed by the “Retire” button. The System then processes the query and updates the list of Programs.

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 Robot Records.	
System	Presents Host with file structure to find Robot Record.	
Host	Locates and selects Robot Record.	Cancels without finding a Robot Record.
System	Transfers request to Robot Librarian	
Postcondition	Robot Record has been Registered.	
Error Conditions	Host cannot select a Robot Record.	
	Robot Record is not correctly Registered.	

4.2.4 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. The Host clicks the “Settings” button in the Main Menu screen, and the System displays the Settings screen. The Host may then change any of the game options as desired. The changes will only be applied to the game if the Host clicks the “Save” button. Otherwise, if the user clicks the “Cancel” button, all changes will be discarded. Afterwards, the System will display the Player Selection screen.

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	

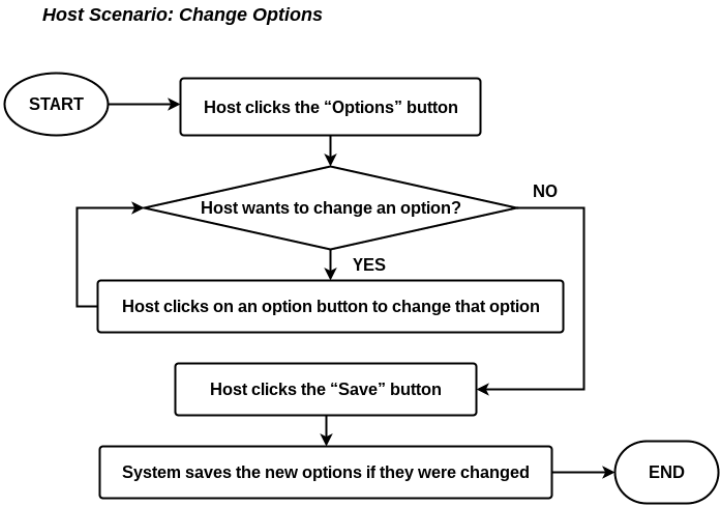

Figure 5: Scenario Flow Diagram: Change Settings

4.2.5 Primary Scenario - Exit Application

The Exit Application Scenario is a Primary Scenario that applies when the Host wants to exit the program and the Main Menu screen is being displayed. The Host clicks the "Exit" button, and the System closes the application.

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.	

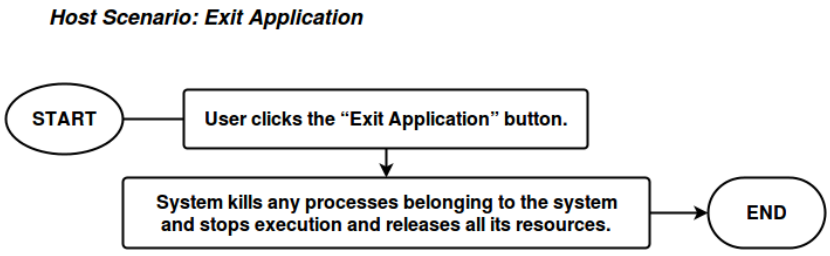

Figure 6: Scenario Flow Diagram - Exit Application

4.3 Player Scenarios

The following are all Scenarios where the primary Actor is the Player of the game. Players interact with the system primarily to perform game actions or to gain more information about the game state.

4.3.1 Primary Scenario - Take Turn

The Take Turn Scenario is a Primary Scenario that applies when a Player's game piece becomes active. First, the System displays a dialogue box on the Turn Transition screen to confirm the Player's turn has

begun. The Player clicks the “OK” button, and the System then displays the main game screen. The Player then has the option to perform a limited number of Move Piece, and Shoot actions to change the game state. The Player can also perform an unlimited number of Inspect Space, Navigate Board, View Game Log and View Statistics actions in order to view information about the game state. At any time, the Player may perform the End Turn action.

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

Flow Controller	Main Flow Step	Alternate Flow Step
Precondition	A Player’s Game piece is currently the Active Game piece.	
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.	

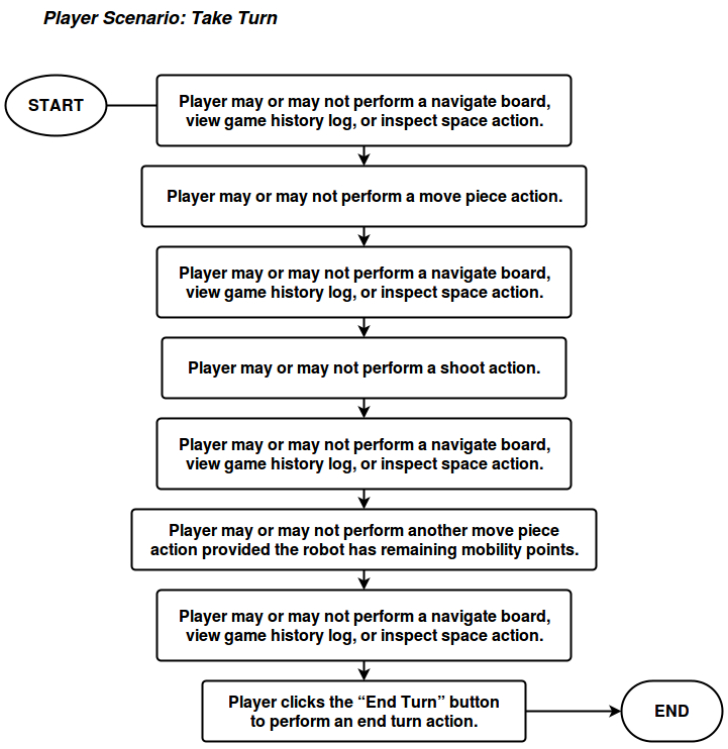

Figure 7: Scenario Flow Diagram - Take Turn (Player)

4.3.1.1 Secondary Scenario - Move Piece

The Move Piece Scenario is a Secondary Scenario that can apply while the Player is in the Take Turn Scenario. The Player selects a space within range of its piece's remaining mobility points, and the System displays the Context Menu. The Player selects "Move" from the Context Menu, and the System moves the piece to the selected space, reduces the piece's mobility points by the cost of the movement, and updates the game log.

Table 9: Use case Definition - Move Piece

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

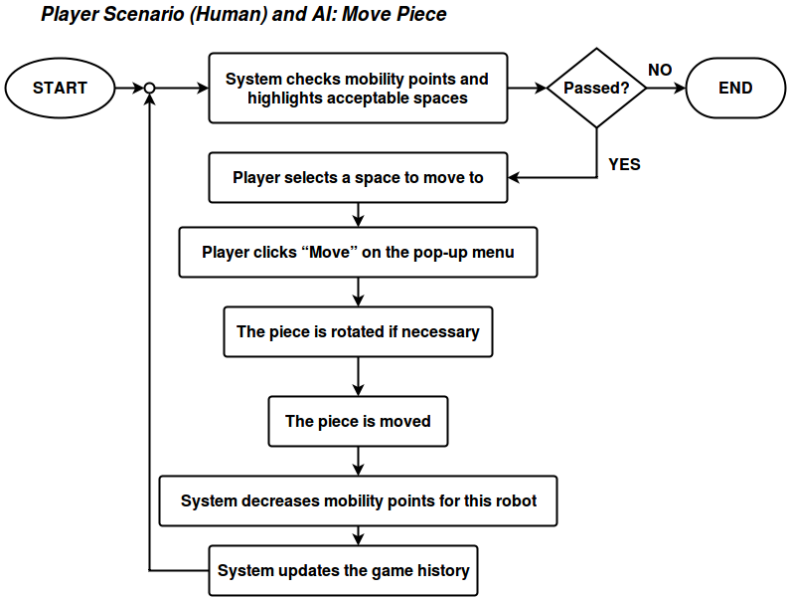

Figure 8: Scenario Flow Diagram - Move Piece

4.3.1.2 Secondary Scenario - Shoot

The Shoot Scenario is a Secondary Scenario that can apply while the Player is in the Take Turn Scenario, only if the Player has not taken the Shoot action already during this turn. The Player selects a space within range of its robot's range points and the System displays the Context Menu. The Player selects "Shoot" from the Context Menu, and the System applies damage to the pieces occupying the space, applies death effects if necessary, and updates the game log.

Table 10: Use case Definition - Shoot

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

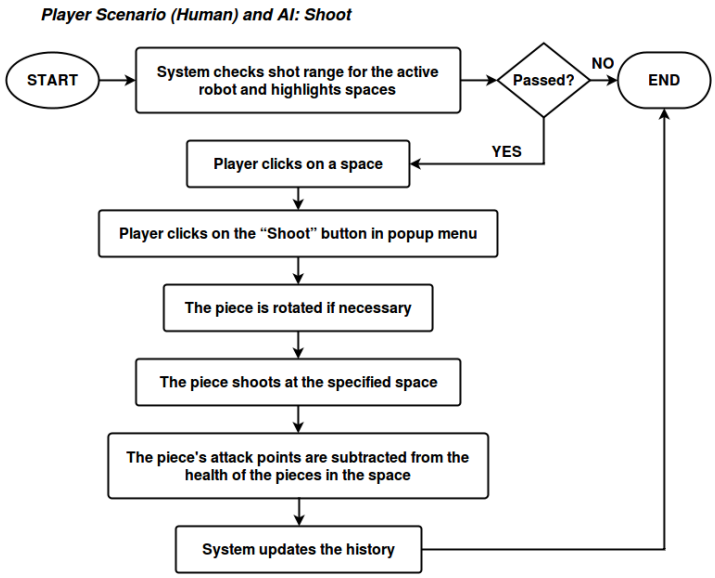

Figure 9: Scenario Flow Diagram - Shoot

4.3.1.3 Secondary Scenario - Inspect Space

The Inspect Space Scenario is a Secondary Scenario that can apply while the Player is in the Take Turn Scenario. The player selects any space on the board and the System displays the Context Menu. The Player selects "Inspect" from the Context Menu, and the System displays information about the pieces in the selected space.

Table 11: Use case Definition - Inspect Space

Flow Controller	Main Flow Step	Alternate Flow Step
Precondition	A Player's Game piece is currently the Active Game piece.	
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.	

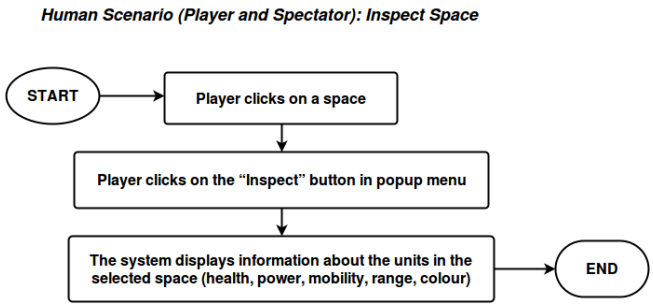

Figure 10: Scenario Flow Diagram - Inspect Space

4.3.1.4 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. The Player clicks the "End Turn" button, and the System displays a pop-up confirmation. The Player may click the "No" button to cancel the action, or may click the "Yes" button, upon which the System ends the player's turn, selects the next player and piece, and displays the Turn Transition screen

Table 12: Use case Definition - End Turn

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

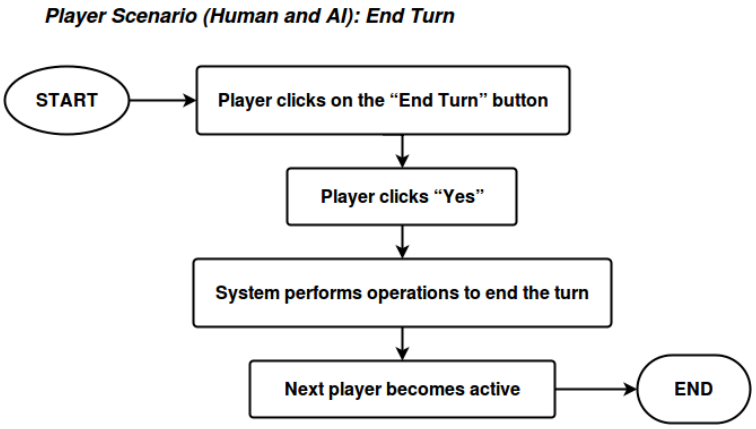

Figure 11: Scenario Flow Diagram - End Turn

4.3.2 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 executes 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.	

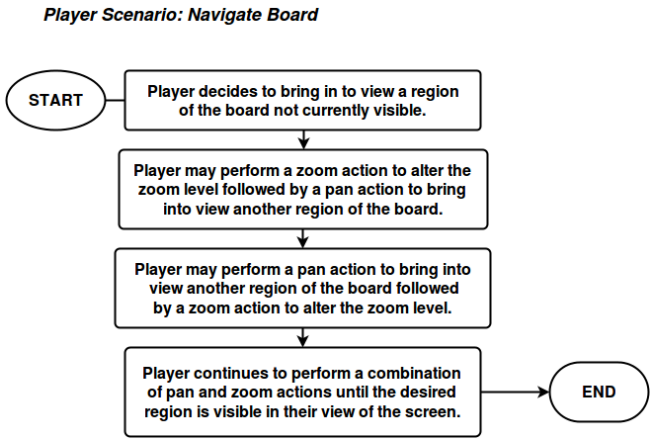

Figure 12: Scenario Flow Diagram - Navigate Board

4.3.2.1 Secondary Scenario - Pan Board

The Pan Board Scenario is a Secondary Scenario that can apply while the Player is in the Navigate Board Scenario. The Player clicks the "Pan Board" button, and the System enables Pan Board mode. In Pan Board mode, the Player presses the mouse up against one of the edges of the screen or presses one of the arrow keys, and the System moves the Player's view in the indicated direction. The Player then clicks the "Pan Board" button again, and the System disables Pan Board mode.

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.	

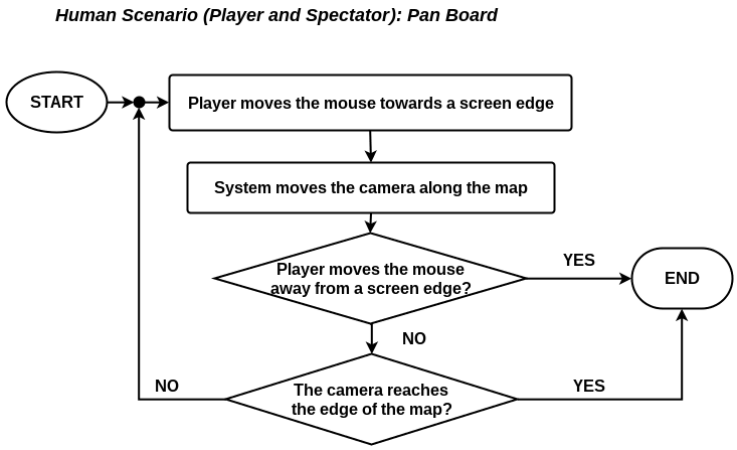

Figure 13: Scenario Flow Diagram - Pan Board

4.3.2.2 Secondary Scenario - Zoom Board

The Zoom Board Scenario is a Secondary Scenario that can apply while the Player is in the Navigate Board Scenario. The Player clicks either the "+" button or the "-" button, and the System will expand or shrink the board display.

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.	

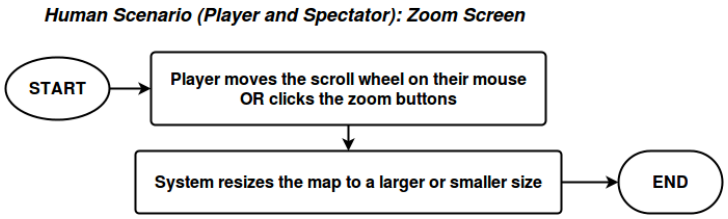

Figure 14: Scenario Flow Diagram - Zoom Board

4.3.3 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. The Player clicks on the "Game Log" bar, and the System displays the Game Log window in the corner of the main game screen. The Player views the list of actions in the Game Log window, and may scroll through the list with the scroll bar, while the System updates the list of actions displayed in the Game Log window. Finally, the Player clicks on the "Game Log" bar again, and the System hides the Game Log window.

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.	

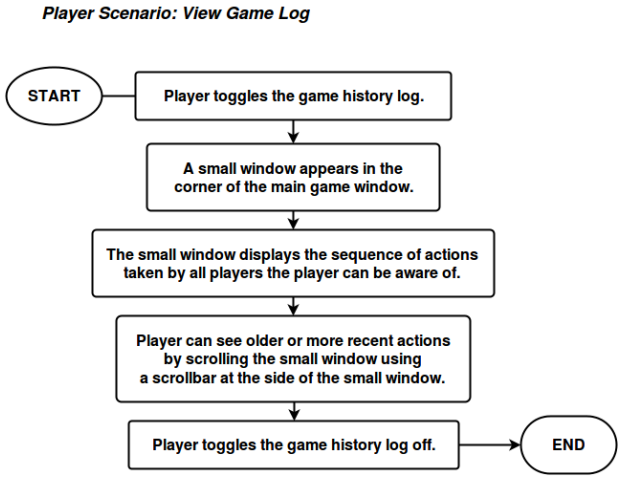

Figure 15: Scenario Flow Diagram - View Game Log

4.3.4 Primary Scenario - View Piece Statistics

The View Robot Statistics Scenario is a Primary Scenario that can apply when the Player has access to the game board. The Player clicks the "S" button, and the System displays the Statistics window. The Player views the statistics, and then clicks the "S" button again. The System then hides the Statistics window.

Table 17: Use case Definition - View Piece 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 Game piece Statistics.	
System	Shows Statistics for all known Game pieces.	
Player	Chooses to stop viewing Game pieces.	
System	Hides Statistics for Game pieces.	
Postcondition	The game board is displayed.	
Error Conditions	The Game piece Statistics do not toggle correctly.	
	The Game piece Statistics shows incorrect information.	

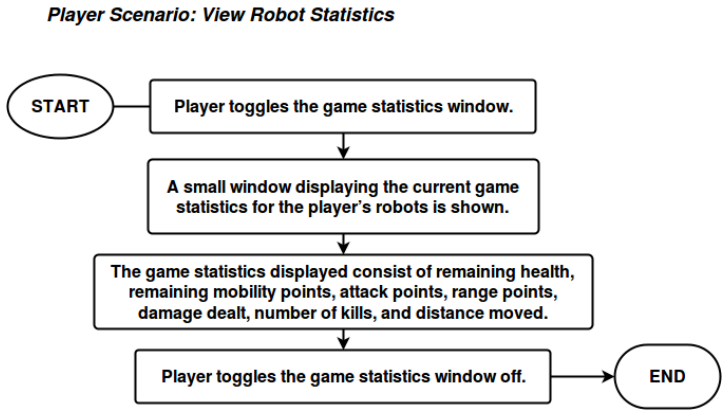

Figure 16: Scenario Flow Diagram - View Piece Statistics

4.3.5 Primary Scenario - End Game

The End Game Scenario is a Primary Scenario that can apply when the Player has access to the game board. The Player clicks the "End Game" button in the top-left corner of the main game window, and the System displays a confirmation dialogue prompting the player to confirm their decision to end the game. The Player may click the "Yes" button in the confirmation dialogue, and the System will display the Main Menu screen. The Player may instead click the "No" button, and the System will close the confirmation dialogue.

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	

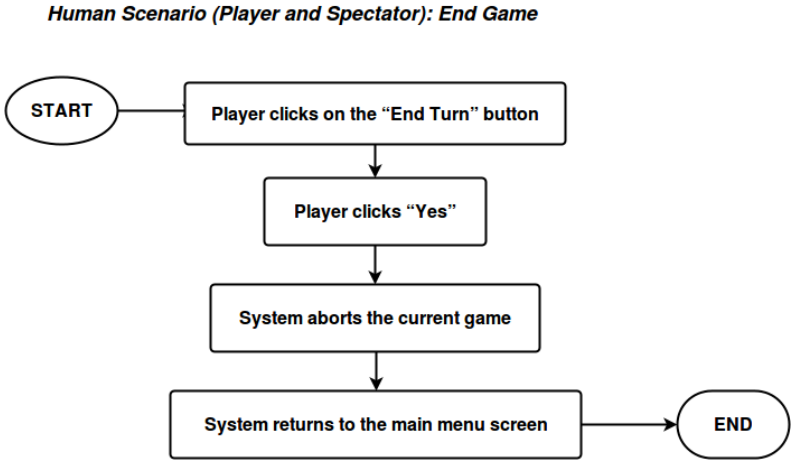

Figure 17: Scenario Flow Diagram - End Game

4.4 Spectator Scenarios

The Spectator Scenarios are comprised of many of the same Scenarios as the Player's Scenarios. Specifically, the Spectator can initiate the Navigate Board, View Game Log, View Robot Statistics, and End Game Scenarios. The difference between a Spectator and a Player in these Scenarios is that the Spectator can see information for all the contestants, 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 Section 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 Section 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 Section 4.3.4, but can see all details.

4.4.4 Primary Scenario – End Game (Spectator)

The Spectator's End Game Scenario is functionally identical to the Player's End Game Scenario, as listed in Section 4.3.5.

4.5 AI Scenarios

The AI Actor has only a single Scenario: the Take Turn Primary Scenario. It is different from the Player's Take Turn Primary Scenario in that it consists of interpreting the commands of the AI's associated Program and performing them.

4.5.1 Primary Scenario - Take Turn (AI)

The AI's Take Turn Scenario is a Primary Scenario that applies when an AI's piece is the active piece. The AI reads instructions from its Program and determine which actions to perform. It will then query the System, and the System will reply with information or the new game state. This process continues until the AI reaches the end of its Program, at which point the AI signals the System that it is ending its turn. The System will then determine the next player and piece, and display the Turn Transition screen for the next contestant.

Table 19: Use case Definition - Take Turn (AI)

Flow Controller	Main Flow Step	Alternate Flow Step
Precondition	An AI's Game piece is currently the Active Game piece.	
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.	

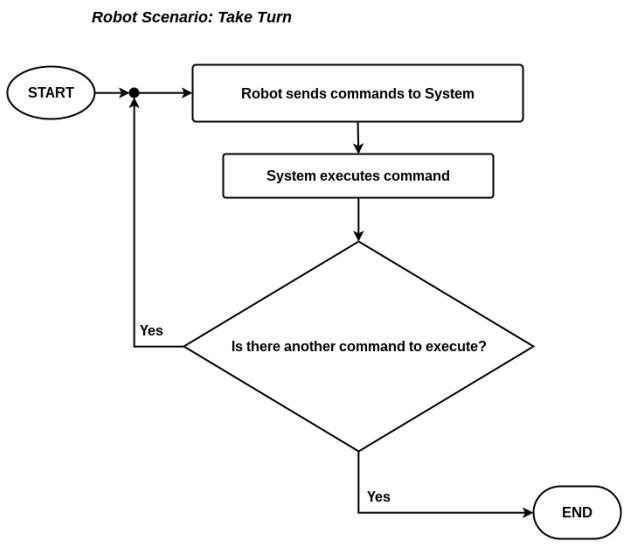

Figure 18: Scenario Flow Diagram - Take Turn (AI)

4.6 Robot Librarian Scenarios

The Robot Librarian Actor has six (6) Scenarios associated with it. It stores and manipulates the Robot Records available to the Host for use with AIs.

4.6.1 Primary Scenario - Enumerate

The Enumerate Scenario is a Primary Scenario that applies when the System asks the Robot Librarian to list the available Robot Records that have certain features. The System queries the Robot Librarian, and the Robot Librarian finds all the Robot Records with that feature, Then the Robot Librarian replies to the System with a list of Robot Records.

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.	

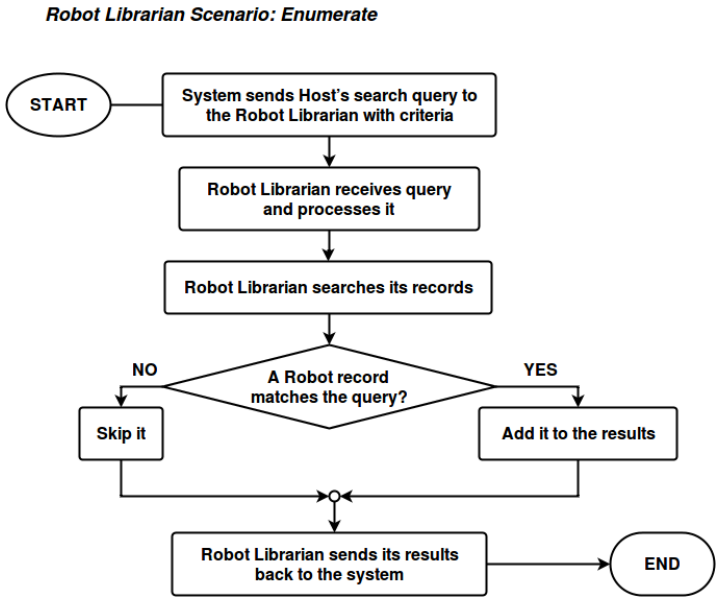

Figure 19: Scenario Flow Diagram - Enumerate

4.6.2 Primary Scenario - Download

The Download Scenario is a Primary Scenario that applies when the System asks the Robot Librarian to load the Robot Records. The System queries the Robot Librarian. The Robot Librarian then accesses the directory where Robot Records are stored. It downloads the records and stores them in its catalogue while the application is open.

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 Robot Records	
Robot Librarian	Searches file structure for Robot Records.	
Robot Librarian	Stores any Robot Records found in library.	
Postcondition	Robot Records that are found are stored.	
Error Conditions	Found Robot Records not stored.	

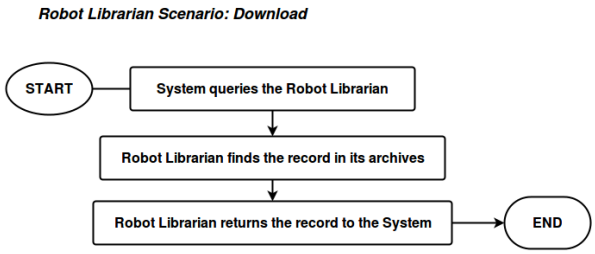

Figure 20: Scenario Flow Diagram - Download

4.6.3 Primary Scenario - Update Statistics

The Update Statistics Scenario is a Primary Scenario that applies when the System asks the Robot Librarian to update the Statistics for a given Robot Record. The System sends the name of the Robot Record and the new statistics to the Robot Librarian. The Robot Librarian selects the correct Robot Record and updates the Robot Record with the new statistics in its catalogue. Then it accesses the directory where Robot Records are stored and writes a new copy of the record to the directory.

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 Robot Record.	
Robot Librarian	Adds new data about Robot Record.	
Robot Librarian	Updates statistics about Robot Record and confirms update.	
Postcondition	Robot Record Statistics updated.	
Error Conditions	Statistics not Updated.	
	Confirmation not returned.	

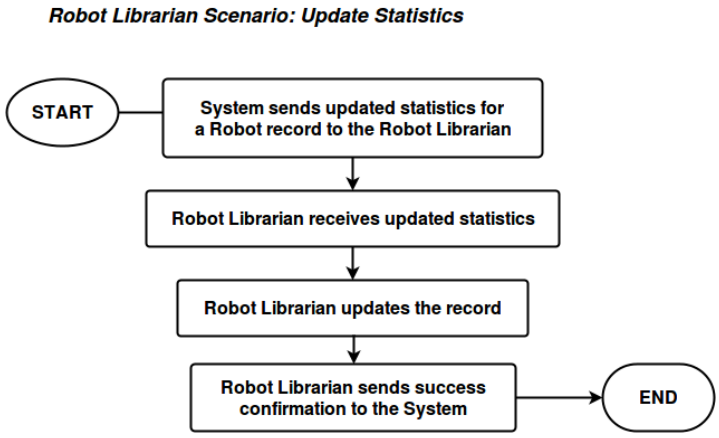

Figure 21: Scenario Flow Diagram - Update Statistics

4.6.4 Primary Scenario: Register

The Register Scenario is a Primary Scenario that applies when the System asks the Robot Librarian to create a new Robot Record. The System sends the Robot Librarian the name of the new Robot Record to create and a Program file. The Robot Librarian creates a new Robot Record with the given name and associates it with the Program. Finally, it sends confirmation of the new Robot Record to the System.

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 Robot Record	
Robot Librarian	Creates Record	

Robot Librarian	Confirms creation	
Postcondition	Record created.	
Error Conditions	Record not Created.	
	Record created for wrong Robot Record.	

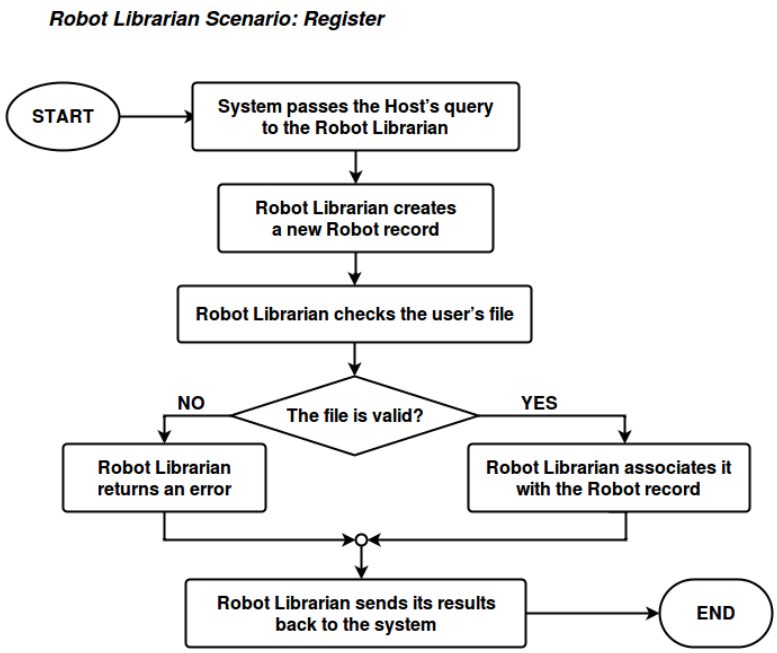

Figure 22: Scenario Flow Diagram - Register

4.6.5 Primary Scenario: Revise

The Revise Scenario is a Primary Scenario that applies when the System asks the Robot Librarian to change which Program is associated with a given Record. The System sends the Robot Librarian the name of the Robot Record and a Program. The Robot Librarian finds the Robot Record associated with the name, and overwrites its Program with the Program provided by the system. Finally, it sends confirmation of the updated Robot Record to the System.

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.	

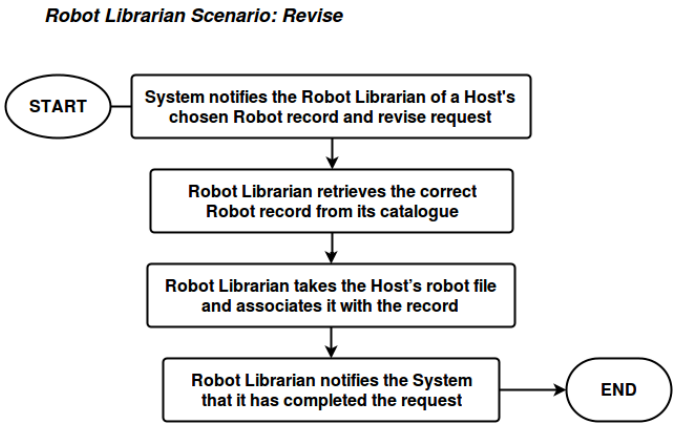

Figure 23: Scenario Flow Diagram - Revise

4.6.6 Retire

The Retire Scenario is a Primary Scenario that applies when the System asks the Robot Librarian to retire a Robot Record, freeing the name for use with another Robot Record. The System sends the Robot Librarian the name of the Robot Record to retire. Robot Librarian finds the Robot Record associated with the name, changes its name, and marks it as retired within the system. Finally, it sends confirmation of the retired Robot Record to the System.

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 Robot Record.	
Robot Librarian	Finds Record and removes it.	
Robot Librarian	Confirms removal.	
Postcondition	Record removed.	
Error Conditions	Record not removed.	

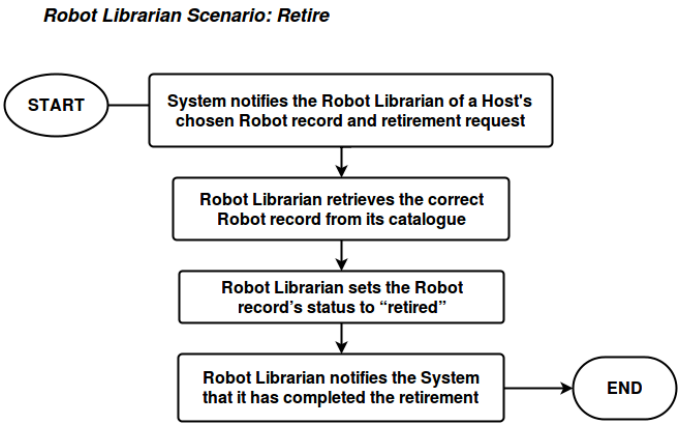

Figure 24: Scenario Flow Diagram - Retire

5.0 Storyboards and User Interface

5.1 Overview

The scenarios listed in Section 4 inform how the program should look and behave. The storyboards and user interface (UI) elements for this application represent the expected method of interacting with the application over time. The interface elements will be discussed first, and then storyboards for interactions among the elements will be provided.

5.2 Interface Elements

The interface elements for this application describe the GUI elements determined to be necessary from the scenarios in Section 4. The figures shown below are nonfunctional, but should provide a reasonable example of how interaction with the application should occur.

5.2.1 Main Menu Screen

The Host sees the Main Menu screen after opening the application. The Main Menu screen welcomes the Host to the game, displays the logo and offers four options in the form of clickable buttons; "New Game", "Robot Archive", “Settings” and "Exit Game."

The "Play Game" button takes the Host to the Player Selection screen, which is discussed in Section 5.2.2.

The "Robot Archive" button takes the Host to the Robot Archive screen, which is discussed in Section 5.2.6.

The function of the "Exit Game" button is the same as the "Close [X]" button provided in the top-right corner of most operating system windows. It will cause the application to close.

Figure 25: Robo-Wars Title Screen

5.2.2 Player Select Screen

The Player Selection screen allows the Host to set up a new match. The Host uses the radio buttons to select either

- a two-player game,

- a three-player game, or
- a six-player game.

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

Next, the Host 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. The System will provide a selection of default player names. The Host may check a box to choose between a human or computer player for each individual player. At the bottom of the screen, there are three buttons: the “Back” button, the “Options” button, and the “Start Game” button.

The “Back” button takes the Host back to the Main Menu screen, without saving any of the changes made to the Player Selection screen.

The “Options” button takes the Host to the Game Options screen, which is discussed in Section 5.2.3.

The “Start Game” button will send the current preferences to the System, and the System will create a new match and display the Turn Transition screen, which is discussed in Section 5.2.4.

Figure 26: Player Selection Screen

5.2.2.1 Robot Team Select Dialog

Clicking on the “Select AI” button next to a player will bring up the Robot Team Select dialogue. This dialogue allows the user to choose a Robot Record to assign to each piece for the computer player. If no Robot Record is selected, the System will assign each piece a default Robot Record. Clicking the “OK” button will close the dialogue and save the current selections the Host made for each piece.

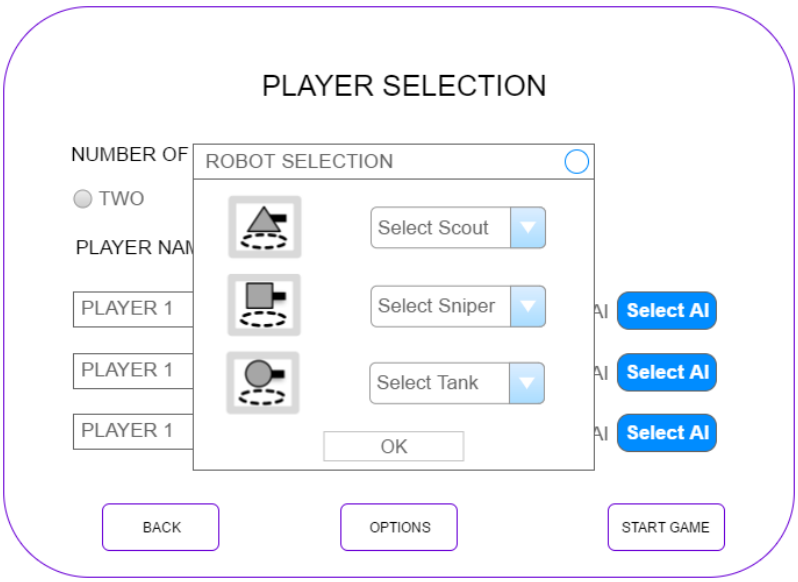

Figure 27: Robot Team Select Screen

5.2.3 Game Options Screen

The Game Options screen offers the user to choose between various options for the match. “Save Robot Statistics” will enable or disable the updating of Robot Records after the completion of a match. Optional features discussed in Section 7 could allow alternate rules for a match, such as a different map. These would be selected from the “Rules” dropdown menu.

The Host can click the “Cancel” button to discard any changes and go back to the Player Selection Screen, or click the “Save” button to save any changes and go back to the Player Selection screen.

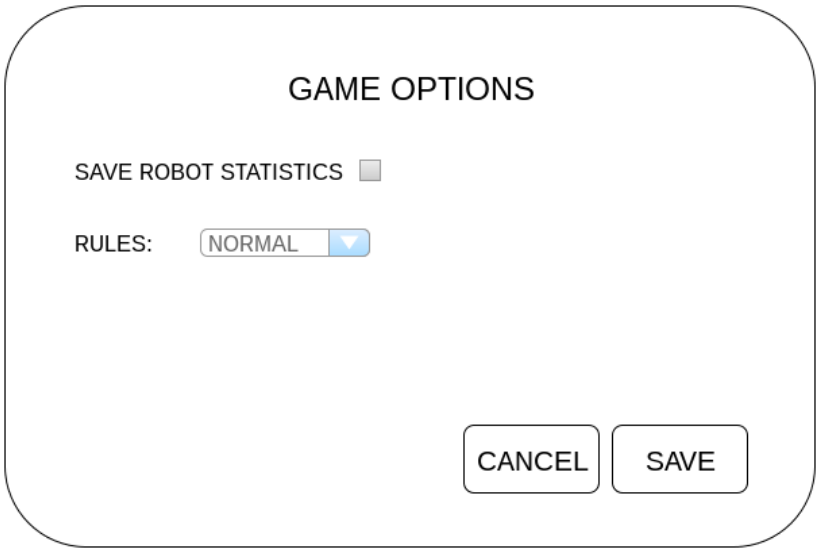

Figure 28: Game Options Screen

5.2.4 Turn Transition Screen

The Turn Transition screen is displayed by the System in between turns in a match. Since the game uses a Hotseat multiplayer structure, the Turn Transition screen is used to shield the game board of one

human Player from another. The Player whose turn it is must click the “OK” button to continue, and the System will display the main game screen.

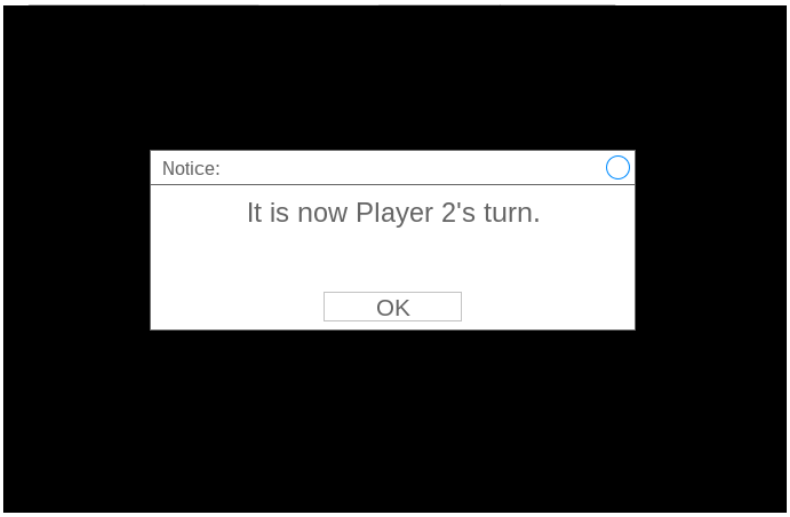

Figure 29: Turn Transition Screen

5.2.5 Game Screen

Clicking the “OK” button from the Turn Transition screen leads the Player to the main Game screen.

On the main Game screen, a horizontal panel on the top shows thumbnails of all the players, with their names and color to identify them. If there is a player who has no remaining pieces, their thumbnail on the panel is greyed out and has a red cross across it. The thumbnail of the current player is displayed in the top right corner, with additional information displayed such as the attributes of the active robot.

A “Quit Game” button is in the top left corner of the screen. On the left side, there are buttons for zooming in, zooming out, toggling Pan Board mode, and displaying the Statistics window. The bottom left corner has an expandable window that contains the Game Log, which the Player may use to refer to the game history. The Player clicks on the Game Log bar to expand the Game Log window.

The game board is oriented around the active piece of the Player by default. When the Player wants to perform the Move, Shoot or Inspect actions, they click on a target space, and a context menu appears with buttons for the player to click to perform that action.

When the player wants to end their turn, they click on the “End Turn” button in the bottom right corner of the screen. A pop-up dialogue asks the player to confirm their decision, warning them if they have moves left. Once the player confirms the decision to end their turn by clicking “Yes,” the System displays the Turn Transition, 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.

Once the match is over, a dialog box pops up declaring the winner. The Player is then lead to the Results screen.

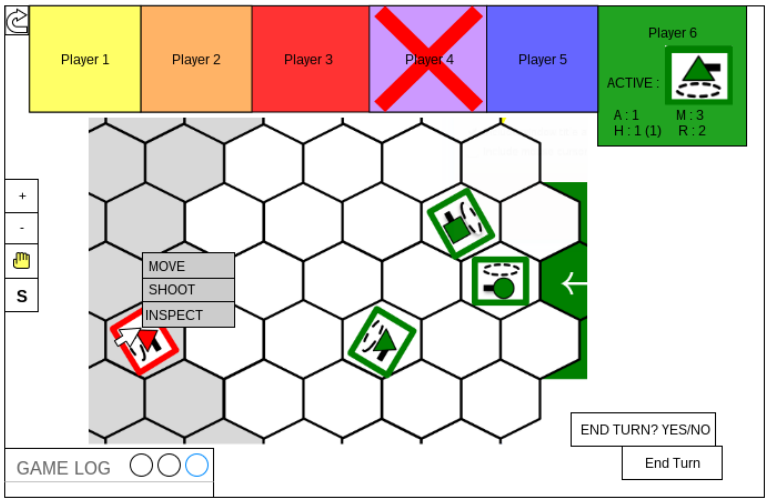

Figure 30: Main Game Screen

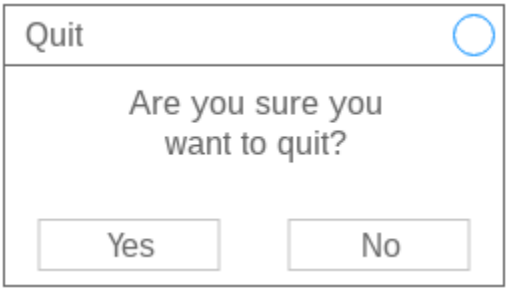

Figure 30: End Game Dialog

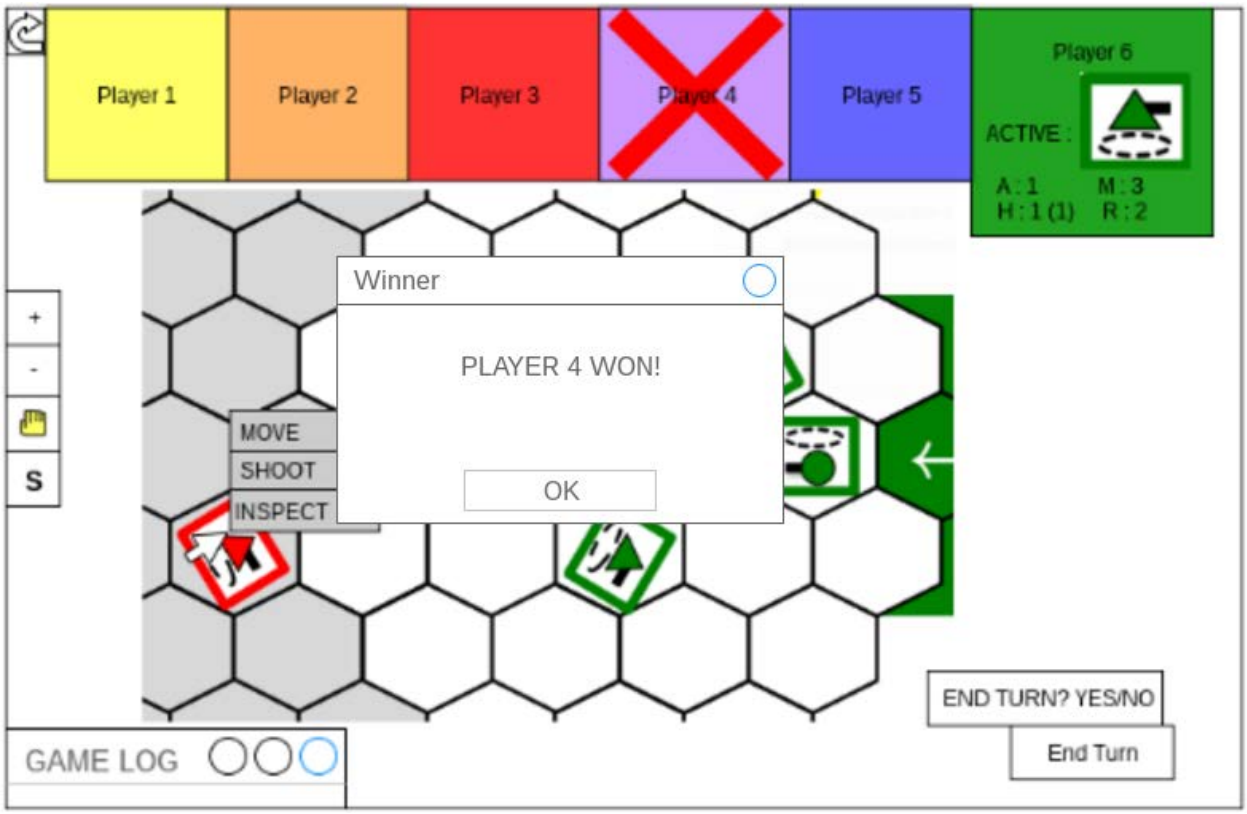

Figure 30: Match End Dialog

5.2.6 Match Results Screen

The Results screen displays the winner of the match, and the statistics of each player's pieces in a very compact, yet well-organized manner. The Host can select a player based on their color, and view the statistics of each piece for that player. There are three option buttons at the bottom: the "End Game" button, the "View Log" button, and the "Rematch" button.

The "Rematch" button will start a new match with the same players and settings as the match that just ended.

The "View Log" button will display the Game Log Window, with information about the entire game. It may be closed by clicking the [X] button.

The "End Game" button will take the Host back to the Main Menu screen.

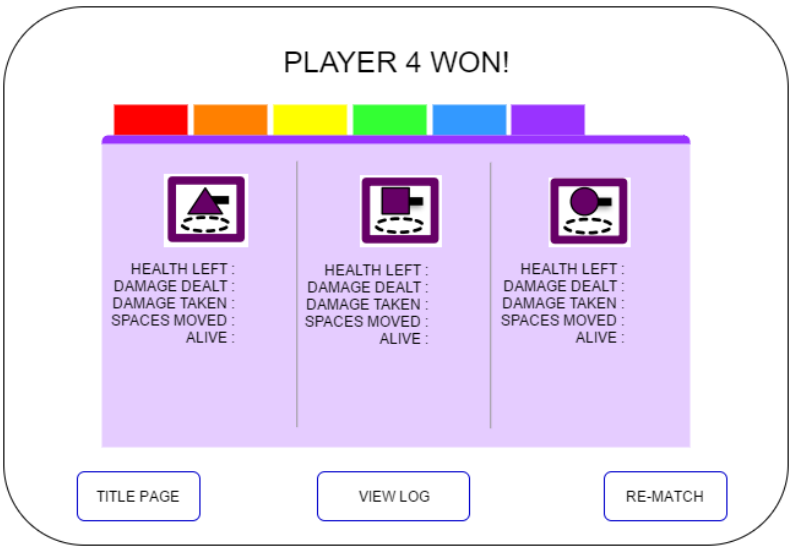

Figure 31: Match Results Screen

5.2.7 Robot Archive Screen

The Robot Archive screen allows the Host to search for a particular Robot Record, and sort the list of records by some criteria, such as name or team. By default, the list module displays every Robot Record by name. The Host may type some text into the search bar and click the magnifying glass icon in order to search for a particular Robot Record by name. In this case, the list module will update to show only the matches for the search. The Host can click on a record in the list module to select it. Information about a particular robot is shown on the right side of the screen when a record is selected.

The “Register”, “Revise”, and “Retire” buttons are displayed below the statistics.

Clicking “Register” will bring up the Register dialogue, which prompts the Host to enter the new Robot’s name and upload a Program file. Clicking “Cancel” will close the dialogue without saving, and clicking “Register” will create the new Robot Record and take the Host back to the Robot Archive Screen.

The Host can select a record from the list module, and then click the “Revise” button to bring up the Revise dialogue, which prompts the Host to upload a Program file. Clicking “Cancel” will close the dialogue without saving, and clicking “Revise” will update the Robot Record with the selected Program and take the Host back to the Robot Archive Screen.

The Host can select a record from the list module, and then click the “Retire” button to bring up the Retire dialogue, which asks for confirmation that the Host wishes to retire the record. Clicking “Cancel” will close the dialogue without saving, and clicking “Revise” will retire the Robot Record and take the Host back to the Robot Archive Screen.

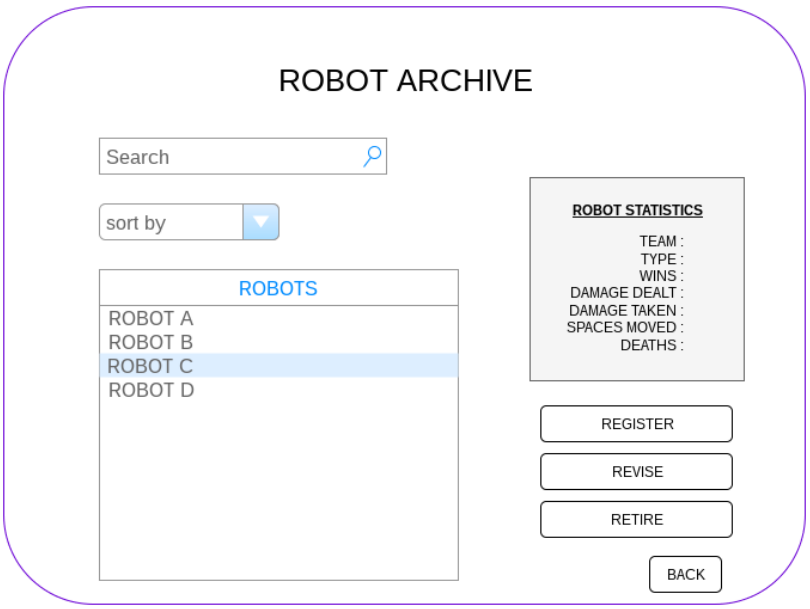

Figure 32: Robot Archive Screen

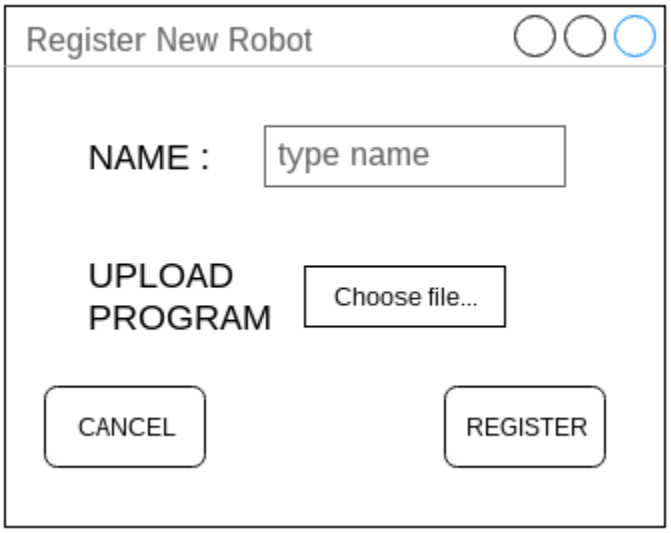

Figure 33: Register Robot Dialogue

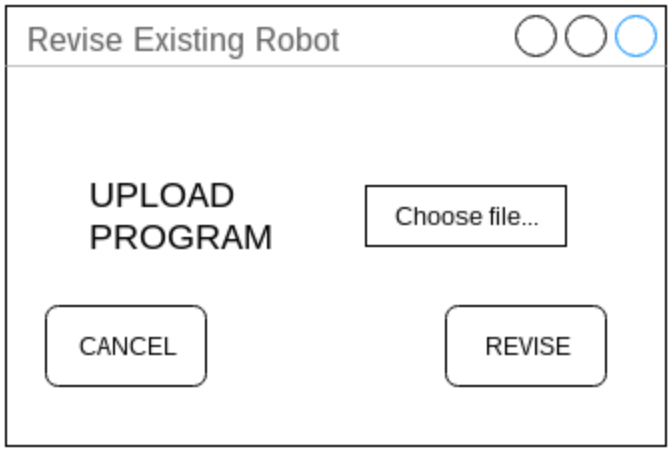

Figure 34: Revise Robot Dialogue

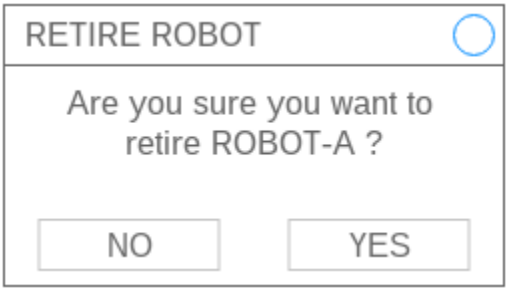

Figure 35: Retire Robot Dialogue

5.2.8 Game Settings Screen

The Settings screen offers the Host two options. The first is a slider, which will control the volume of game sound and music. This is an optional feature, described in Section 7.3. The other options are dropdown boxes allowing key mapping, for advanced players who prefer to highlight game board spaces with their keyboard. ‘Save’ and a ‘Back’ buttons at the bottom of the screen allow the Host to save any changes to the settings, or discard changes made. Clicking either will take the Host back to the Main Menu screen.

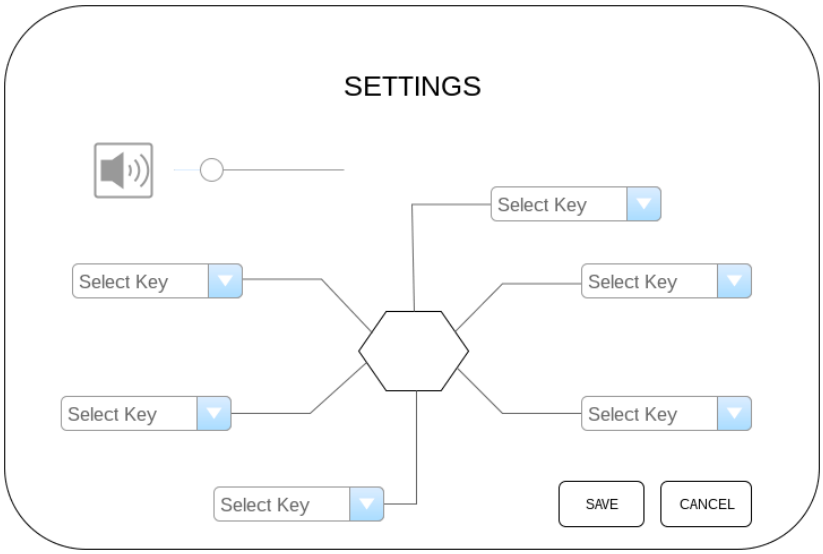

Figure 36: Game Settings Screen

6.0 Non-Functional Requirements

6.1 Overview

In addition to the functional requirements described in the sections 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 text of the Program that an AI will run during its matches.

Each Program is a separate plain-text file (.txt) that contains the instructions for an AI to run during a match. Each line will contain an instruction that will be parsed by the AI.

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

Robo-Wars 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.

Because Robo-Wars will be built using Java, there is a possibility, but not a guarantee, that the application will run on systems other than the Windows 7 64-bit operating system.

7.0 Potential Extensions to System

7.1 Overview

There are many facets of Robo-Wars and RoboSport370 which may be improved. This section describes features and mechanics that are not 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.

7.5 Tutorial Mode

Another enhancement that could be made to the game is to include a mode for learning how to play the game. There are several ways to do this. Currently, two approaches are being considered.

7.5.1 Advisor

One approach is to have an Advisor that can sense when a Player has not performed a specific type of action. When this occurs, the Advisor would appear an offer advice to the Player. The Player could also ask the Advisor for advice at any time.

The benefit of this approach would be that the Player could learn the game while in the actual play environment. However, Advisor approaches in other games can get repetitive and require careful handling.

7.5.2 Tutorial Minigames

Another approach is the Tutorial Minigame approach. This entails a separate game mode, selected from the Title Screen, where specific mechanics related to the game are taught in controlled environments. As an example, there would be one tutorial on how to move a game piece, while another tutorial handles how to shoot a game piece.

This approach can greatly assist in learning the individual actions that comprise a game. A reductionist approach like this has the drawback of potentially reducing the speed at which a Player learns the game as a whole.

In order to begin the tutorial, the Host would click the “Tutorial” button on the Main Menu screen. The System would display the Main Game screen, and the Player would be prompted to perform specific actions during a turn. Instead of displaying the Turn Transition screen at the end of each Player turn, the System would instead allow the Player to view the board while the AI makes a moves.

Appendix A: RoboSport370 Rules

(Copied from “game.pdf” on the Course Moodle)

• With two players, red and green are used, and the board has 5 spaces on a side. With three players, red and yellow and blue are used, and the board can have 5 or 7 spaces on a side. With six players, all colours are used, and the board has 7 spaces on a side. The colours are assigned randomly.

• All robots enter in their home space, facing along the arrow.

• Any number of robots, including ones from opposing teams, can occupy the same space.

• A turn begins with Red team playing it robot with greatest movement; then orange plays its robot with greatest movement, ..., finally blue plays its greatest-movement robot — this completes one round. Then a second round is played where Red plays its next-highest movement robot, then orange, ..., up to blue. Then a third round where Red’s lowest-movement robot plays, ..., finally blue’s lowest-movement robot plays. Thus, every robot gets one play per turn, and fastest robots move first.

• Note that, Red’s highest-movement robot might not be its scout, because that robot might be dead.

• If a team does not have a robot to play during a round, skip to the next colour.

• A play consists of moving, shooting, and moving again. It costs no movement point to turn to face any direction (0—5), and one movement point to enter a space. If a robot runs out of movement points, it cannot move any further during that turn. Different robots have different movement amounts. At the end of every turn, robots regenerate their movement points—i.e. they can move again next turn.

• Robots can only see and shoot at robots in range. As they move, new spaces become visible.

• Shooting at a space means selecting a range and direction causes the shooting-robot’s attack-rating to be subtracted from the health of all robots in that targeted space. If a robot’s health drops to (or below) zero, the robot is dead and removed from play. Until a robot is dead, its movement and damage ratings are unimpaired. Note that range=0, direction=0 is valid.

• If more than one team’s robots are still alive, another turn starts with Red’s highest-movement robot.

• If only one team’s robots are still alive, that team wins the game.

Appendix B: Git Tag

The Git Tag for the Team D4 repository is “REQUIREMENTS.”

Index

Action, iii, 2, 15
  definition, 2
Actor, iii, vii, 2, 8, 15, 25, 26
  definition, 2
AI, iii, iv, 1, 3, 5, 6, 7, 9, 25, 26, 34, 36, 41, 44
  definition, 7
Artificial Intelligence. *See* AI
Benefits, iii, 5
Graphical User Interface. *See* GUI
GUI, iii, 3, 32
Host, ii, iv, 1, 6, 9, 10, 11, 12, 13, 14, 26, 32, 33, 34, 35, 37, 38, 40, 42, 43, 44
  definition, 6
Hotseat, iii, 3, 6, 35, 36, 43
  definition, 3
Match, 3, 11, 37, 43
  definition, 3
Player, ii, iv, v, vi, vii, 1, 6, 7, 9, 10, 13, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 32, 33, 34, 35, 36, 42, 43, 44
  definition, 6
Program, iii, 3, 7, 9, 11, 12, 25, 29, 30, 38, 41
  definition, 3
requirements, 1, 2, 6, 41, 42
Robot Archive, iv, v, vi, vii, 11, 12, 13, 32, 38, 39
Robot Librarian, iv, 1, 6, 7, 12, 13, 26, 27, 28, 29, 30, 31
  definition, 7
Robot Record, iii, 3, 7, 13, 28, 29, 30, 31, 34, 38
  definition, 3
Robo-Wars, i, vi, 1, 2, 5, 33, 41, 42, 43, 44
Round, iii, 3
  definition, 3
Scenario, iii, iv, v, vi, 3, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32
  definition, 3
Spectator, iv, 1, 6, 25
  definition, 6
Team, i, iii, vi, 4, 5, 6, 34
Turn
  definition, 3
Use case, iii, vii, 2, 3, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 26, 27, 28, 29, 30, 31
  definition, 2
User Interface. *See* GUI