

Requirements Document

CMPT 370 Group A3

Platform

Robosport must have the capability to run on the Java Virtual Machine, and therefore will run on any platform that has the capability to run the Java Virtual Machine. Robosport will not require a powerful computer to run on, it will be able to run on any common hardware like a laptop.

Goals

There are three main goals for the project:

1. Write a **simulator** that executes the game.
2. Create a **test platform** to debug the robots.
3. Write a team of robots in RobotLanguage to compete in matches.

Scope

Must have

Simulator

The program must have the ability to import robots via plain-text files containing programs written using the robot language as defined in the RobotSport370 Language specification. During a Robosport match each robot must behave according to their program and must follow the commands defined in their program as defined in the specification with the limitation that the robot must not violate the **rules of a RoboSport Match**.

There must exist a user interface for the program such that a user may select “RoboSport370 Language” program files for the robots participating in any given match. This interface must also allow the user to define a match to take place, choosing among the various match types available as are specified below, including at minimum **Watch Match** and **Instant Results**.

Should have

The program should if possible contain the **Test-Bench** mode specified below. There should be a rate control to set how fast a match will execute.

Could have

The program could have sound effects and music to improve the experience.

Would Like to have

If development time permits, it would be desirable to have the ability to import entire robot ‘teams’ as a single file. Higher quality graphics are also dependent on time available, as is the depth of the RoboSport terminal optionally provided as part of the **Test-Bench**.

Robot Limitations

Every Robot has the following characteristics:

- Health Points
- Movement Range
- Weapon power
- Weapon range
- Team number
- Robot number

Health: If health is 0 or below the robot disappears from the board

Weapon power: The robot will deal this many points of damage when they deal a successful hit to another robot.

Weapon range: The robot can successfully hit a robot which is within this many tiles of the robot.

Team number: Each team is assigned a number which will allow the robots to identify who is an ally and who is a foe.

Robot number: Each robot is assigned a number which will allow the robots and user to identify each robot on a team.

Movement Range: Each time a robot moves one square its “Movement Range” is decreased by one. It replenishes to its specified value at the beginning of each turn.

Time Limit

Each robot match will terminate after a set amount of turns, this is to prevent stalemates. If a match times out, then whichever team has the most surviving robots will be declared the winner. In the case where both teams have the same

number of robots, the team with the most cumulative health points is the winner. If the cumulative health points are equivalent, then the match is declared to be a tie.

Robosport Rules

All robots must perform under their **limitations**.

Number of teams should not be more than 6 in a single match.

All the teams must start with 4 robots on their team.

Modes

Watch Mode

This mode will allow the user to select 2 or more robots to fight one another and must allow the user to select program files from which to load each of these robots. The user can either watch the match or step through the match one action at a time and observe the results. After the match, it will provide a print out of relevant statistics from the match, including at minimum the winner of the match.

Instant Results Mode

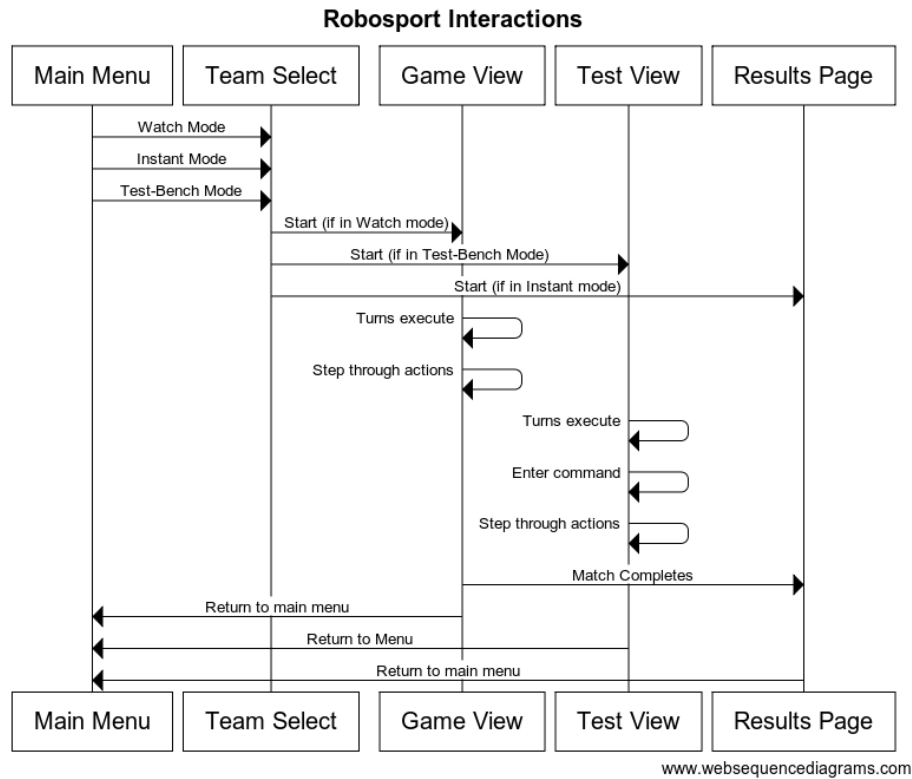
This mode will allow the user to select 2 or more robots to fight one another and must allow the user to select program files from which to load each of these robots. The match will be simulated in full without any visual output to the user. After the match, it will provide a print out of relevant statistics from the match, including at minimum the winner of the match.

Test-Bench Mode

Test-Bench mode contains the capability for the user to load robots from files. It must then allow the user to step through the program and observe the results. As the match progresses the user must be able to view the statistics of the robots involved (e.g. their health, defense, etc.) and change the state of the robots (stats, position) via a command terminal. Robots may be loaded in via command.

Scenarios

Sequence Diagram



Watch Mode

- When the user selects **watch mode** from the main screen, an interface to select and configure robot teams will appear.
- When the user confirms their robot team in watch mode, the game window will open up and simulation will begin
 - The user can press the step button to advance a single step, or let the match play
 - When the match ends, a screen showing the match results is shown.

Instant Results Mode

- When the user selects **instant results** mode from the main screen, an interface to select and configure individual robots will appear.

- When the user confirms their robot team in instant results mode, the match will be invisibly simulated and a screen showing the match results is shown.

Test-Bench Mode

- When the user selects **test-bench** mode from the main screen, an interface to select and configure individual robots will appear.
- When the user confirms their robot team in test-bench mode, the test-bench interface will open
 - The user can press the step button to advance a single step, or let the match play
 - The user can enter commands to alter the state of the test

Robots must be able to participate in matches and perform according to **rules and limitations**.

Interfaces

The following is a description of the various user interface screens that will be present in the RoboSport application. Whether or not they are displayed is dependent on the selected game mode, as seen in the **Scenarios** section. Illustrations of the interfaces can be found in the **Storyboards** section.

Main Interface

The main interface contains the options to select from the available modes as specified in **Modes**.

Team Selection Screen

The Team Selection screen provides an interface to load robots into up to 6 different teams. It is a simple interface that shows 6 different teams, each with their own “Load” button to allow the user to select a robot to load. The robots are loaded one at a time. Once the user is happy with their teams, the “Confirm” button takes them to the next screen.

Game Screen

This screen displays the hex-grid on which the position of the robots will be displayed as the match is simulated. It also contains controls including a Play, Stop, and Step button, as well as a playback speed controller to control the simulation. Finally, it has an event log to show what the robots are doing.

Test-Bench Screen

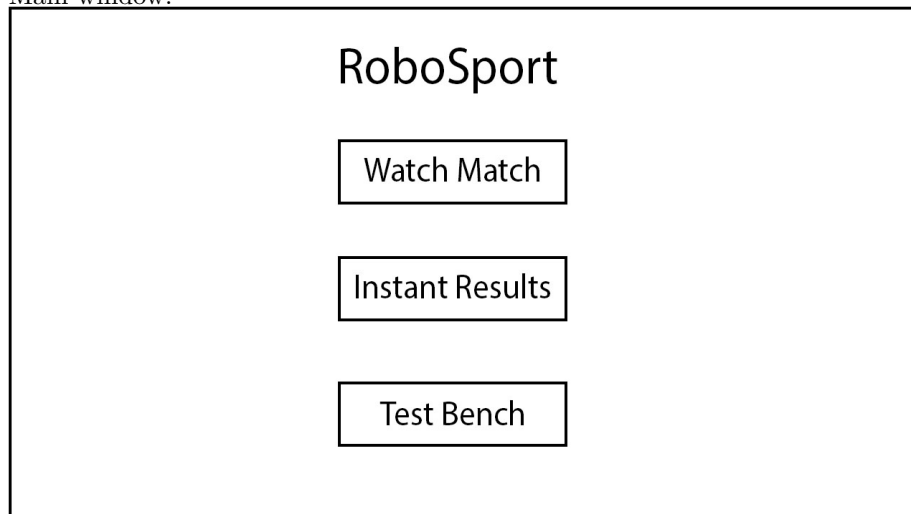
The Test-Bench Screen is identical to the Game Screen, except for the inclusion of a Command Terminal in which the user can enter commands to debug the robots.

Results Screen

The Results Screen displays the winner of the match, as well as stats for each team such as number of turns survived, damage dealt, and damage taken.

Storyboards

Main window:



Team selection window:

Team Selection

Confirm

1

Load

Loaded Robots:

2

Load

Loaded Robots:

3

Load

Loaded Robots:

4

Load

Loaded Robots:

5

Load

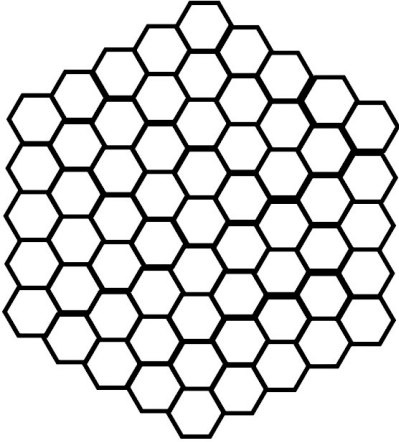
Loaded Robots:

6

Load

Loaded Robots:

Game screen:



▶

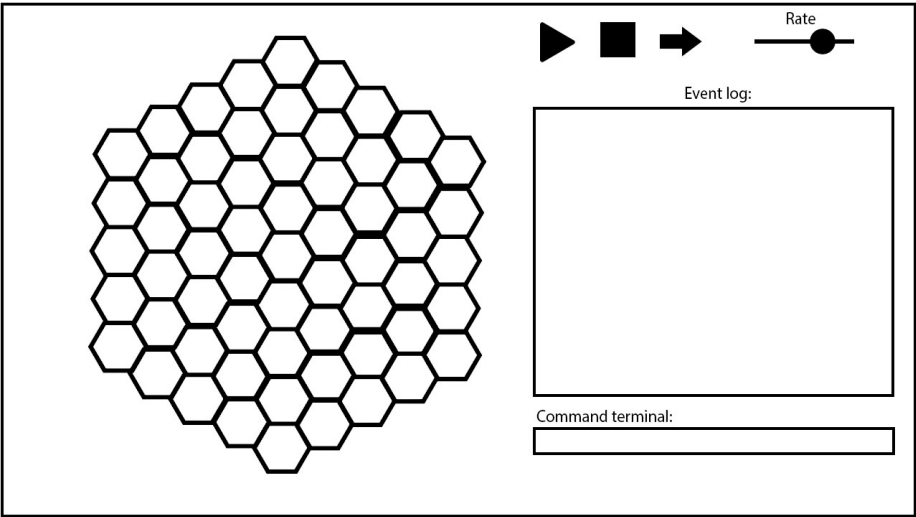
■

➡

Rate

Event log:

Test-Bench screen:



Results screen:

Winner: Team {#}

1

{rank} Turns survived:
Damage dealt:
Damage taken:

2

{rank} Turns survived:
Damage dealt:
Damage taken:

3

{rank} Turns survived:
Damage dealt:
Damage taken:

4

{rank} Turns survived:
Damage dealt:
Damage taken:

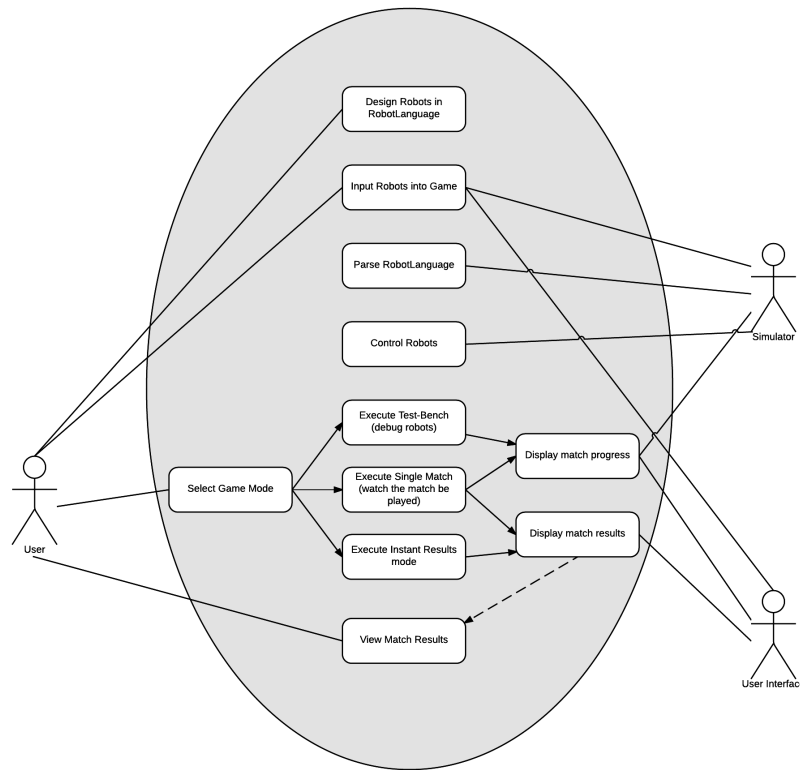
5

{rank} Turns survived:
Damage dealt:
Damage taken:

6

{rank} Turns survived:
Damage dealt:
Damage taken:

Use Case Diagram



The above use-case diagram outlines the different actions that can be performed by the 3 actors in the program:

- The User
- The Simulator
- The User Interface

The connections between actors and actions show which actors are involved in performing that action. Arrows between actions show the possible paths of execution an action could take. For example, the user selecting a game mode has 3 different possible outcomes depending on which game mode he or she selects.