**REQUIREMENTS**

**GROUP D1, CMPT370**

Nico Dimaano, ned948

Niklaas Neijmeijer, nkn565

Kyle Seidenthal, kts135

Brendon Sterma, bws948

Jiawei Zang, jiz457

**INTRODUCTION**

This document outlines the requirements for creating a system for the “Robot Arena” game. The “Robot Arena” game is a simple game, played on a board. There can be 2, 3 or 6 players at a time. Each player begins with three robots, each with different statistics. The players take turns moving their robots, starting with the robot with the highest range. On their turn, a player may move, shoot or do nothing. The last player with a robot on the board wins the game. The system we are creating will implement this game and its rules on a computer system that allows the game to be played by multiple players at once. This system will make the game less tedious to keep track of, and allow a different gameplay experience than playing the game on a traditional tabletop board. The most noticeable of these differences will be the ability to hide enemy players from the view of a player. We can use the system to hide enemy robots when they are out of range, which brings out a new strategy and difficulty to the game. Also, the ability to play the game over a network will make gameplay more convenient, giving each player their own machine to play on. The rest of this document will outline what such a system will require to be successful.

GIT TAG GOES HERE WHEN COMPLETE --------------------------------------------------------

**SYSTEM DIAGRAM**

Below is a high level diagram of the system:

**ACTIONS AND THEIR SCENARIOS**

Below is an overview of actions and scenarios compatible with the system.

**User: Navigate Menu Scenario**

When the user starts the program they are presented with the main menu screen. The user may navigate the menu which presents options for the game such as choosing to join or create a match, or exit the program.

NEED MORE INFO

**User: Create Game Scenario**

When the user has chosen to create a game from the menu, they are presented with option regarding the number of human and non-human players are to be part of the match. A new game is initiated and the user becomes a player in the system. Should the player decide to cancel, they are sent back to the main menu.

**Preconditions: -**The player has successfully started the program and can navigate the menu screen.

**Post-conditions:** -The user is a player

**-**A new instance of the game is created

**Error-conditions:** -If the player does not have a network connection and chooses two or more human players, the game cannot be created and is not joinable by other human players.

**FIG x.x:**  *The above figure shows the sequence of events taken by a user who has chosen to create a new game. Note that at any time, should the player decide to cancel this action, they are sent back to the main meu screen.*

**User: Join as player**

The user may also choose to join a game as a player. In this scenario, the user choses the “Join Game” option from the menu screen. The user is then prompted to enter a form of match identification for the match they would like to join. If the user has a valid network connection, they are prompted to choose to be a player or an observer. connected to the match as a player.

**Preconditions:** -A valid game must exist

-There must be an empty player slot in the game

- The user and game must both have valid network connections

**Post-conditions:** -The user becomes a player

-The user has joined a match

**Error-conditions:** -The user has no network connection

**FIG x.x:** *The above figure shows the sequence of events when the user chooses to join a match as a player.*

**Secondary scenario: User – Join as observer**

Should the player choose to join the game as an observer instead of as a player (as in step 5 of the diagram above) they are connected to the game as an observer. This allows them only to watch a game in progress.

**PLATFORM**

**SUMMARY**