I. Introduction

This project will be a simulator that will simulate battles between robots created by other players all on a virtual map with a set of rules that must be followed. The project will contain multiple parts that will all have different requirements and restrictions. The simulator created will be able to simulate battles with other players robots. The battle will then be displayed graphically on a hexagonal map which spectators will be able to watch and possibly manipulate.

The robots will be designed with a specific standard that will be decided in a standards meeting, this project will include designing a set of robots to participate in other teams simulations. The robots will be able to run on any simulation provided it adheres to the standards laid out in the standards meeting.

II. Background

In September, 2015, our group was commissioned by Professor Christopher Dutchyn to design a program, called RobotSport370, which would simulate battles between player's robots created on a virtual map. We were given certain requirements, important features, 'could-have's, and extras, which we will outline in our scope.

This project will require a design method to be followed to the successful completion of the project. We have begun this project to meet these requirements and practice this design process.

III. Scope

A. Must Have:
    1. The software must be able to run on the Universities' Linux machines using TuxWorld.
    2. The software must be able to simulate matches between other players and be able to run simulations with robots designed by other teams. This involves writing an interpreter to interface between the robot's language, and the simulation's.
    3. Standard set of robot actions must be implemented, as decided in the robot language specification. these include "move", "shoot", "scan", "identify", "hex", and possibly others if the standard evolves over time
    4. A User Interface will be implemented that will allow users to view the hex map, and see important information about a match. Some sort of graphics will be required, at the very least ASCII graphics will be used, but if time permits more detailed graphics will be made.

5. A set of robots must be created using the the robot language specified in class.

B. Should Have:
    1. The software should be able to have an option to run a simulation without displaying graphics, this would allow many simulations to be run in a short amount of time.

    2. In the user interface having useful information about the simulation to show the spectators would be useful, players could see information such as the amount of time the simulation has been running or information about the robot's current status and team information.

C. Nice to Have
    1. The spectator should be able to play and pause the simulation at will

    2. The software could have a debug/testing simulation in which we can view a match at different speeds, rewind, fast forward, and view stats about each robot.

    3. The testing mode would benefit from having a feature that allows the user to move robots around the map on the fly, so they can experiment with different scenarios.

D. Would Like to Have
    1. If time permits, advanced graphics and sounds would be implemented. These may include animations, or detailed sprites for each robot. Complex graphics are considered a low priority, and will only be implemented after the rest of the system is working.

    2. More complex debugging options would be a desired feature if time is permitting. Ideally, we would like to be able to step through and edit code as a match plays, so we can experiment with different robot behaviours.

IV. Limitations

A. Issues
B. Questions - Difference Between spectators and testers?
C. Number of teams on map?
D. Clarification on tournament mode?

V. Actors

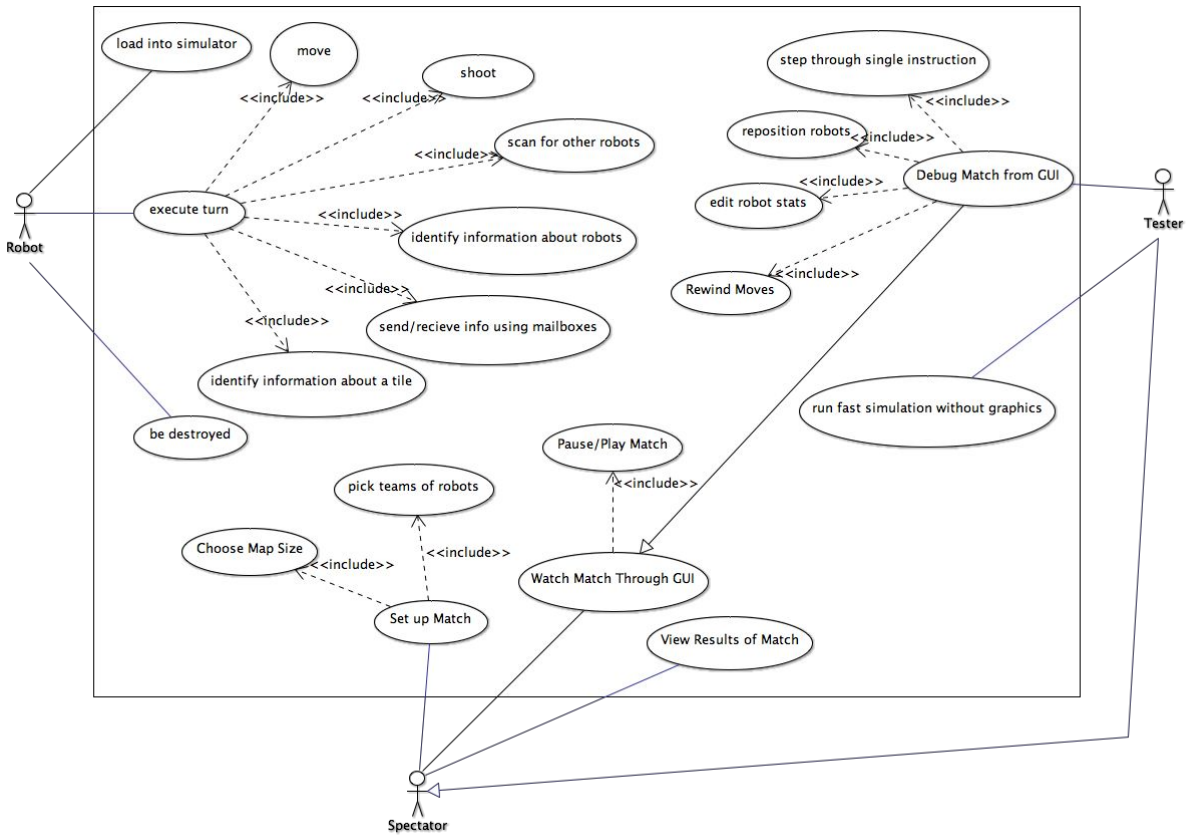
A. External Robots
B. Spectators

C. Testers

VI. Use Cases

Text and then reference an attached diagrams

A. As a spectator I want to watch a match through a GUI
B. As a spectator, I want to be able to pause and resume a match as it runs
C. As a spectator, I want to pick teams of robots for a match
D. As a spectator, I want to choose the map size
E. As a robot I want to be loaded into the simulator
F. As a robot I want to be able to shoot
G. As a robot, I want to be able to move
H. As a robot, I want to be able to scan for other robots
I. As a robot, I want to identify information other robots
J. As a robot, I want to be able to identify details of a given tile
K. As a robot, I want to be able to send/recieve information to my teammate through mailboxes
L. As a robot, I want to be removed from play when my health reaches 0
M. As a robot, I want to be able to execute my script on my turn
N. As a Tester I want to be able to rewind
O. As a Tester I want to be able to step through instructions
P. As a tester, I want to be able to edit robot's stats on the fly
Q. As a tester, I want to be able to reposition robots on the fly
R. As a Tester I want to simulate many matches simultaneously without a UI

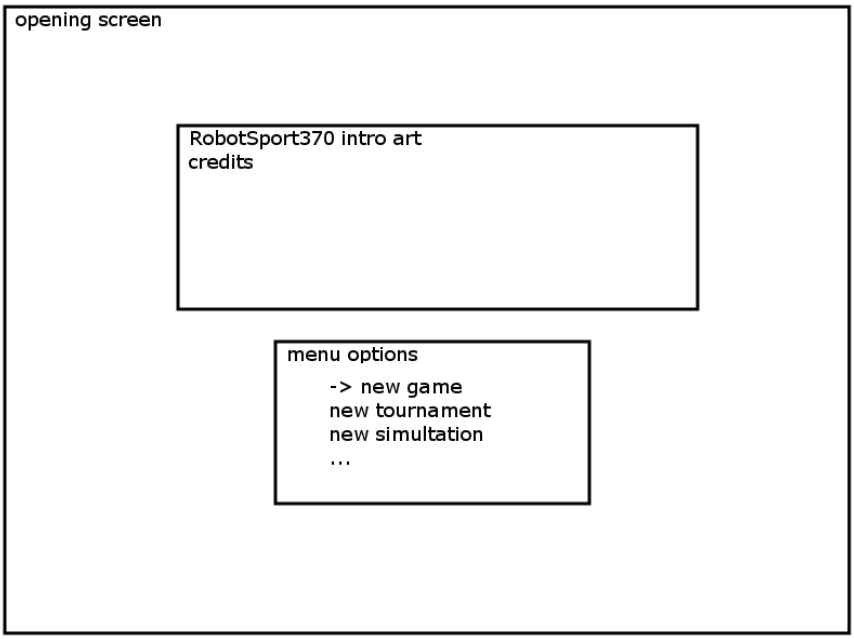


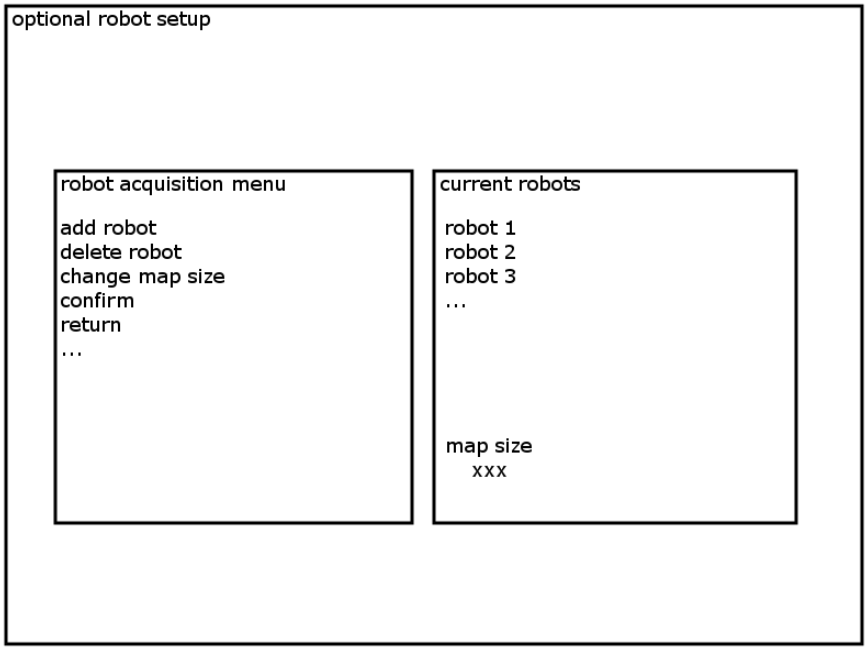
VII. Activity Diagrams

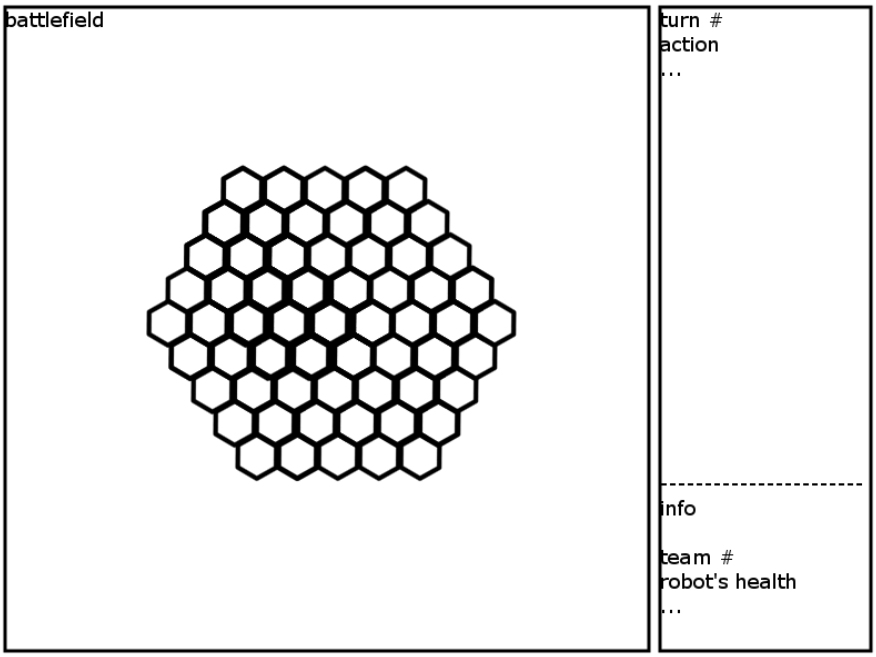
A. Load robots into simulator
    a. Preconditions:
    b. Postconditions:
    c. Errorconditions:
B. Select map size
C. Select number of teams
D. Select testing mode
E. Perform tests using testing gui (pause/play, rewind?)
F. Display Results
G. Return to beginning

VIII. Storyboards

For our storyboards, we came up with three primary screens to create, as below. These storyboards act as a guide for our graphical user interface (GUI).


opening screen
RobotSport370 intro art
credits
menu options
-> new game
new tournament
new simultation
...


optional robot setup
robot acquisition menu
add robot
delete robot
change map size
confirm
return
...
current robots
robot 1
robot 2
robot 3
...
map size
xxx



-> debugging version and a end of game version

IX. Conclusion

Notes

Requirements 1. play,pause, rewind, skip
2. (5,7,9,11) size
3. UI
4. Graphics
5. Testing Interface

Diagrams 1. Use Cases/scenario
2. activity - fancy flow-chart
3. story board

Actors -Robots
-Spectators/UI Users

Use Cases -?

Standards meeting October 2nd
---
5 steps of design

1) Requirements -we are here
2) Architecture 1
3) Architecture 2
4) Coding
5) Documentation

Recipe cards to make class diagrams
-