**War of Robotcraft**

**Design Document**

Team: A3

Team Members:

|  |  |  |
| --- | --- | --- |
| **Name** | **NSID** | **Student ID** |
| Fu, Chen | chf354 | 11183491 |
| He, Jiahuan | jih889 | 11183346 |
| Wang, Shisong | shw940 | 11157916 |
| Xie, Ruida | rux793 | 11194258 |
| Yang, Chen | chy202 | 11183550 |

Date: Oct 21, 2016

|  |  |  |  |
| --- | --- | --- | --- |
| **Version Number** | **Change Request Number** | **Description of Changes** | **Approved Date** |
| 0.1 | 0 | First Draft | 2016-10-21 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Document History Log:

# Introduction

# Architecture

# Detailed Design

## 3.1 Packages and Classes

### 3.1.1 Model

### 3DCoordinate

The 3d coordinate class contains three integer values x, y, and z.

### Game

The game class makes the whole game run functionally.

* + setPlayerPosition() : void
    - Summary:
    - Precondition:
    - Postcondition:
  + nextPlayer() : void
    - Summary:
    - Precondition:
    - Postcondition:
  + runPlay() : void
    - Summary:
    - Precondition:
    - Postcondition:

### Map

The map class

* + setMapSize(int mapSize) : void
    - Summary:
    - Precondition:
    - Postcondition:
  + updateMist() : void
    - Summary:
    - Precondition:
    - Postcondition:

### Player

The player class

* + isDead() : bool
  + getCurrentRobot() : Robot
  + nextRobot() : void
* HumanPlayer extends Player
  + move() : void
  + turn() : void
  + shoot() : 3DCoordinate

### AIPlayer extends Player

### Robot

* + turn(int direction) : void
  + move() : void
  + shoot(int distance) : 3DCoordinate
  + damaged(int attackPoint) : void
  + isDestroyed() : bool

### 3.1.2 View package

### GameStartView Class

The game start view

### SetGameModeView

### GameBoardView

### PlayerStatusView

### GarageView

### 3.1.3 Controller

### WarOfRobotcraft

* + game : Game
  + ActionPerformed(ActionEvent e) : void
    - Summary: handle all the button clicked event
    - Precondition: button is clicked in views
    - Input: e is an ActionEvent from clicked button
    - Output: none
    - Postcondition:
  + keyTyped(KeyEvent e) : void
    - Summary: handle all the key typed event
    - Precondition: key is pressed in views
    - Input: e is a KeyEvent
    - Output: none
    - Postcondition:
  + startTimer() : void
    - Summary: start a new timer to be displayed in GameBoardView
    - Precondition: none
    - Input: none
    - Output: none
    - Postcondition: a new timer is started and is showed in GameBoardView
  + GameStartView\_ButtonClicked(JButton clickedButton)
    - Summary: handle the button event of buttons in GameStartView
    - Precondition:
    - Input:
    - Output: none
    - Postcondition: a new SetGameModeView instance created if pressed new game button; a new GarageView instance created if pressed garage button; game quit if pressed quit button
  + SetGameModeView\_ButtonClicked()
    - Summary: handle the button event of buttons in SetGameModeView
    - Precondition:
    - Input:
    - Output:
    - Postcondition:

### 3.1.4 WarOfRobotcraft

### Entry

## 3.2 UML

# Different

**Game Class**

This class is used to create a new game including a list of players, the number of all players, and the current player who takes control of a play. Whenever a player executes an action that is moving a robot or shooting at a specific direction, the game will be updated the status. Then controller will send these updates to corresponding view objects, and the view object displays them.

**setPlayerPosition(): void**

**Summary:** this method is to set all player’s position to corresponding side of the game board.

**Precondition:** the game is created.

**Input:** nothing

**Return:** void

**Postcondition:**

Set players to position Red and Green if the number of players is 2;

Set players to position Red, Yellow and Blue if the number of players is 3;

Set players to all positions if the number of players is 6.

**goNextPlayer(): void**

**Summary:** this method is to let the next player take control of the game

**Precondition:**

**Input:** nothing

**Return:** void

**Postcondition:**

**runPlay(): void**

**Summary:** this method is to run a play

**Precondition:** the game is created

**Input:** nothing

**Return:** void

**Postcondition:**