**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 Package

#### 3.1.1.1 3DCoordinate Class

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

#### 3.1.1.2 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.

**Fields:**

playerList: LinkedList<Pair<int, Player>>

playerNum: int

currentPlayer: Player

gameMap: Map

**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 set the next player as current player

**Precondition:** nothing

**Input:** nothing

**Return:** void

**Postcondition:** the current player is set to the next player

**runPlay(): void**

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

**Precondition:** nothing

**Input:** nothing

**Return:** void

**Postcondition:** a new play starts

**updateGame(): void**

**Summary:** this method is to update the game after an action executed by the current player

**Precondition:** nothing

**Input:** nothing

**Return:** void

**Postcondition:** the game is updated

#### 3.1.1.3 Map

The map class

setMapSize(int mapSize) : void

* + - Summary:
    - Precondition:
    - Postcondition:
  + updateMist() : void
    - Summary:
    - Precondition:
    - Postcondition:

#### 3.1.1.4 Player

The player class

* + isDead() : bool
  + getCurrentRobot() : Robot
  + nextRobot() : void

HumanPlayer extends Player

* + move() : void
  + turn() : void
  + shoot() : 3DCoordinate

#### 3.1.1.5 AIPlayer extends Player

#### 3.1.1.6 Robot

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

### 3.1.2 View package

#### 3.1.2.1 GameStartView Class

The game start view welcome players and give the choice that start new game, manage the robots and exit the game. This class is used to create the game start frame including game name label, start game button, garage button, and exit button. When players click or press these button, the controller will respond these events, and send the result to set game mode view or garage view.

**gameStartView():void**

**Summary:** this method is to the game start frame, with game name, a button to start game, a button to go to garage and a button to quit the game.

**Precondition:** nothing

**Input:** nothing

**Return:** void

**Postcondition:**

The game start view constructed.

#### 3.1.2.2 SetGameModeView Class

The set game mode view displays the various mode, give player choices to set game mode. This class is used to create the view frame including a list of radio button of game mode, a list of radio button of player’s number, a list of combo box of player type, and a start button. After a player chooses the start game button, the controller will create the set game mode view. And any choice in this view will send to the controller.

**setGameModeView(): void**

**Summary:** this method is to construct the set game mode frame, a list with three radio button to set game mode, a list of three radio button to choose player’s number, a list of combo box to set player type, and a button to start game.

**Precondition:** nothing

**Input:** nothing

**Return:** void

**Postcondition:**

The set game mode view constructed.

#### 3.1.2.3 GameBoardView Class

#### 3.1.2.4 PlayerStatusView Class

#### 3.1.2.5 GarageView Class

### 3.1.3 Controller

#### 3.1.3.1 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