Group 8

Author: HeZi’an

Software Requirements

HuaRongDao

Table of Contents

[System Objective 2](#_Toc39096639)

[Domain Analysis 2](#_Toc39096640)

[System Architecture 4](#_Toc39096641)

[Use Cases 4](#_Toc39096642)

[Software Requirements 5](#_Toc39096643)

[R1: GameUI 5](#_Toc39096644)

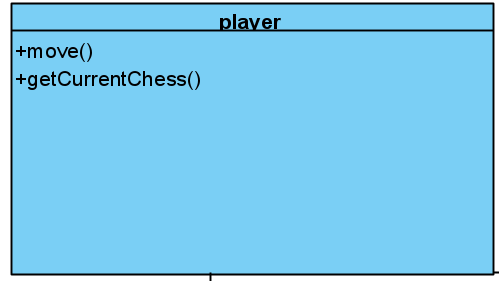
[R2: GameProcess 5](#_Toc39096645)

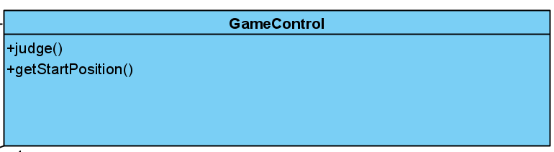
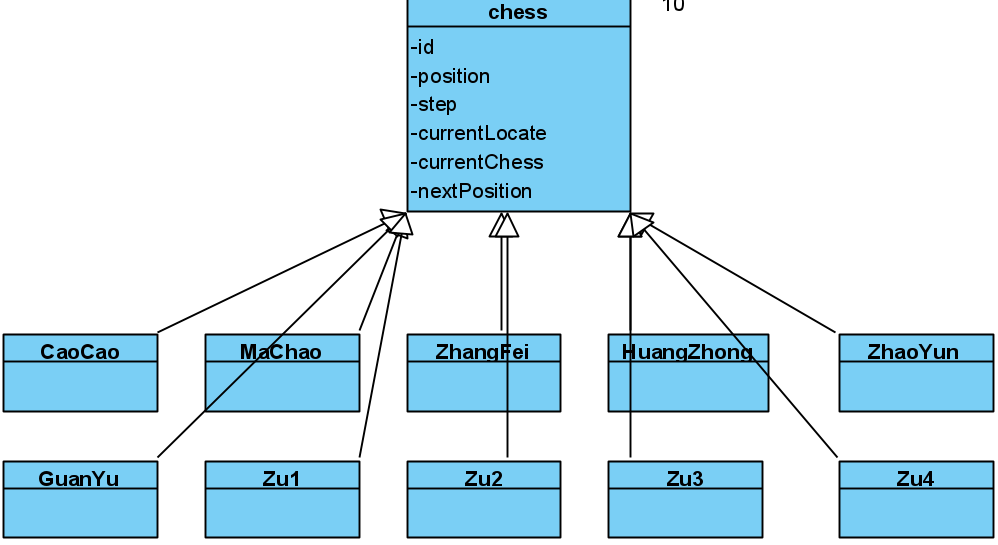
[R3: ChessDB 5](#_Toc39096646)

## System Objective

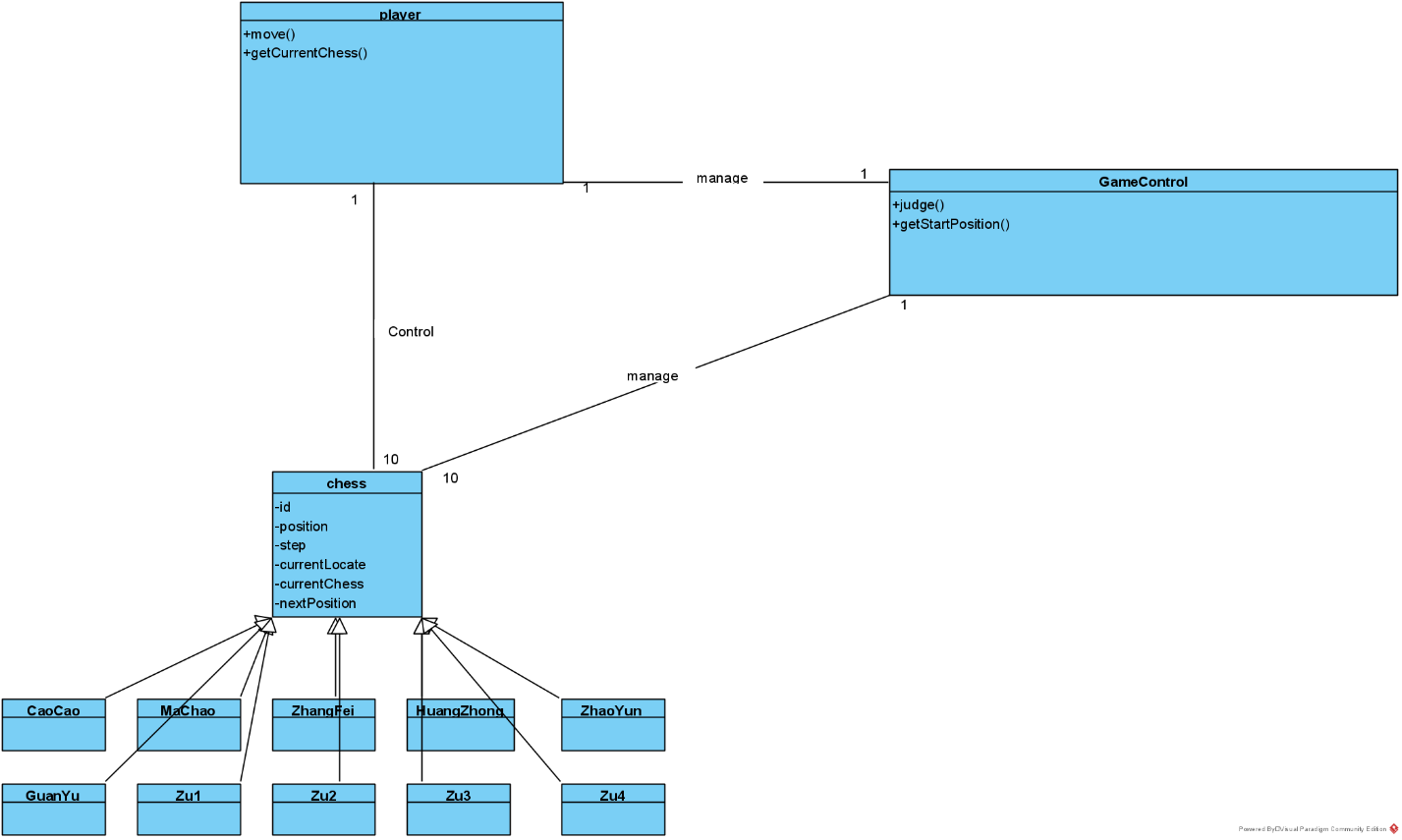
In this project, we are developing a game called Huarong Road. The main gameplay is that the player moves the position of each piece on the board to make the designated piece (Cao Cao) leave the board through the exit.

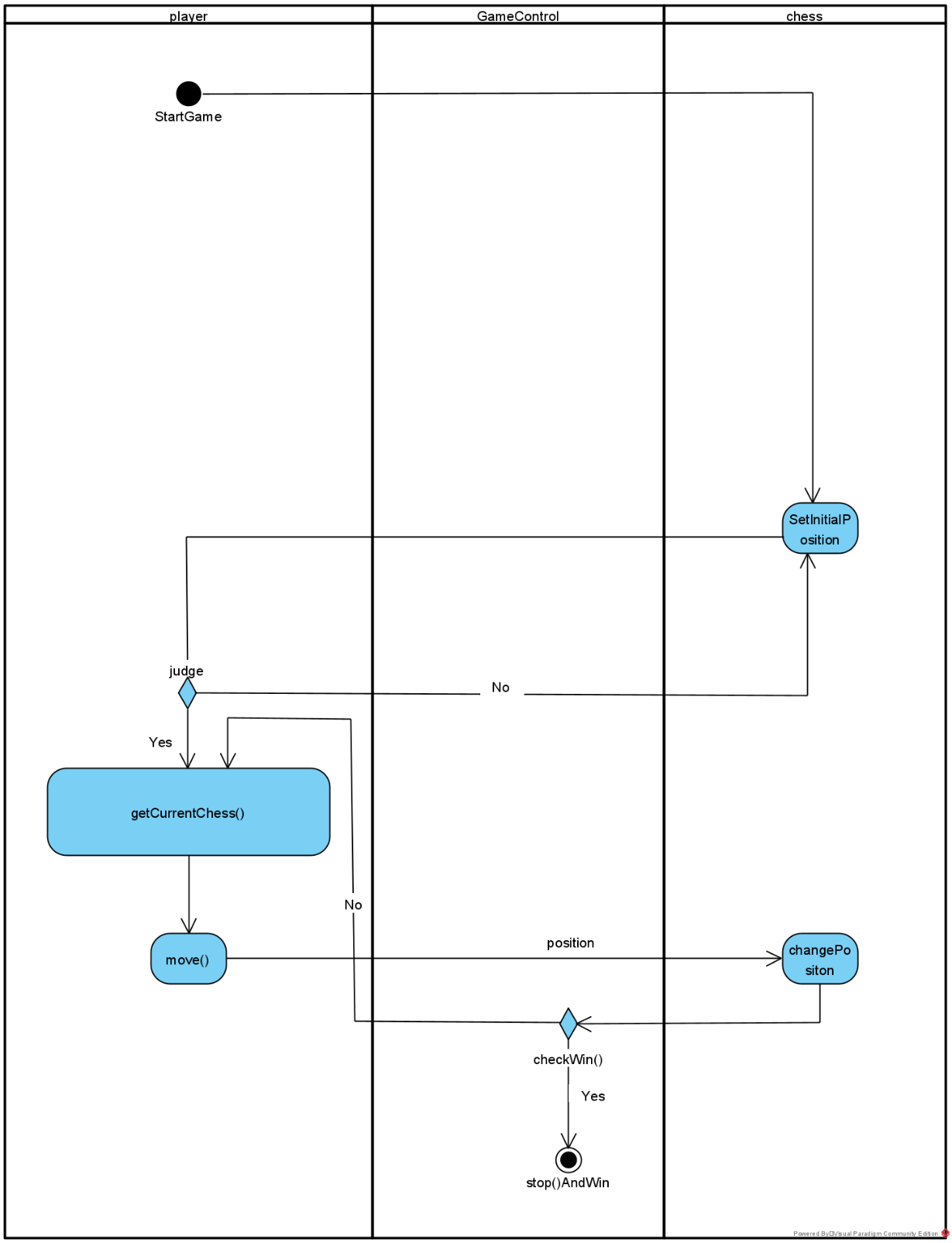
## Domain Analysis

 The components of this game can be categorized into Player, GameControl and Chess.

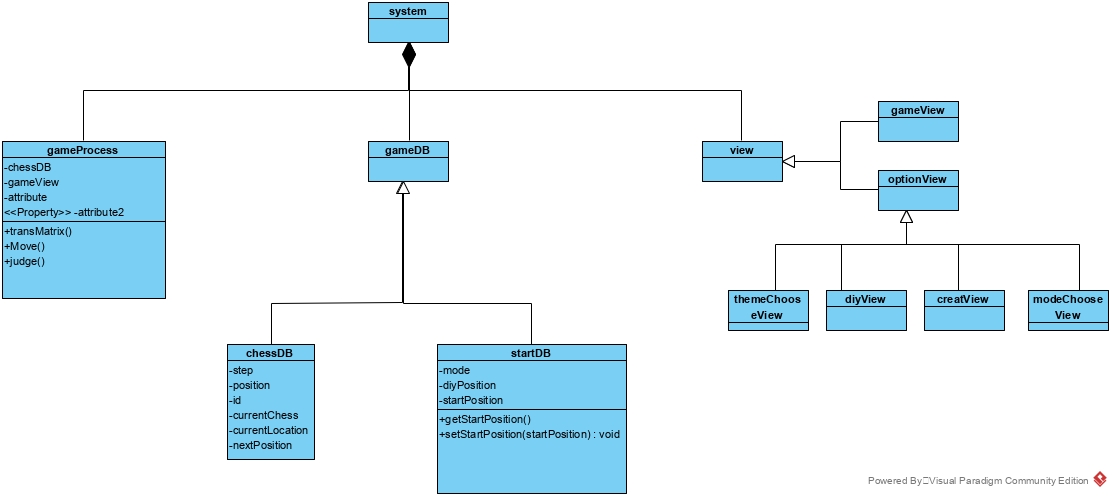


The relationships among different components are shown as follows:

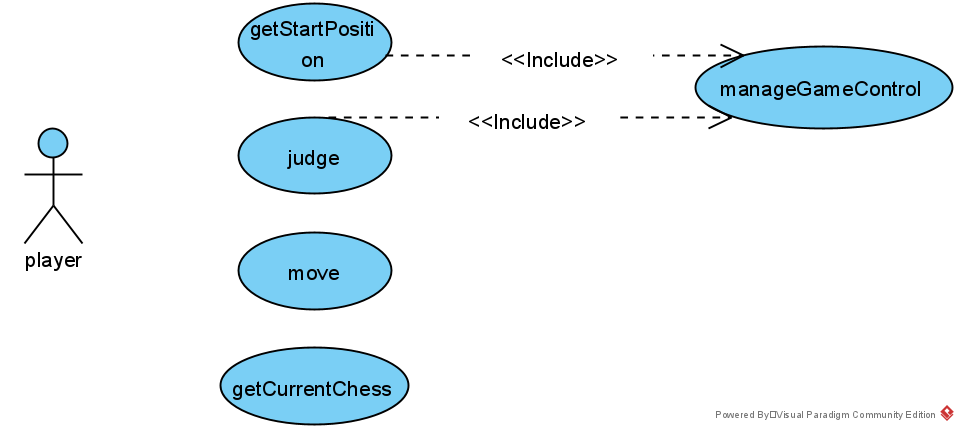


Here is the sequence of the game:

## System Architecture

From the information above, we will design a software system shown as below:

## Use Cases

The system can achieve the following use cases from the player’s perspectives:

## Software Requirements

### R1: Unit

* R1.1: The StartDB can display some pre-set position of chess
  + R1.1.1: getStartPosition: get the position of the pre-set chess
* R1.2: GameProcess help to make the game work
  + R1.2.1: transMatrix: trans the position of the chess in to a matrix
  + R1.2.2: getCurrentChess: select the chess by id and position
  + R1.2.3: move: move the selected chess
  + R1.2.4: judge: check if there is a solution

### R2: Integration

* R2.1: GameProcess+StartDB
  + R2.1.1: getStartPosition () with transMatrix()
  + R2.1.2: getStartPosition () with getCurrentChess()
* R2.2: Can check if the randomly generated game can be solved.

### R3: Functional

* R3.1: Change Theme
  + R3.1.1: theme
* R3.2: Change Mode
  + R3.2.1: mode
* R3.3: DIY Chesses
  + R3.3.1: diy
* R3.4: Play Game
  + R3.3.1: run