
DEPARTMENT OF COMPUTING

---

COMP3211 ProjectGroup 1

# Jungle Game

Software Requirements Specification

---

*Name:*
CHEN
JIANG
ZHANG

*Student ID:*

Oct 2022

# Contents

**1 Preface** **1**

**2 Introduction** **2**
2.1 Purpose . . . 2
2.2 Scope . . . 2
2.3 References . . . 2
2.4 Overview . . . 2

**3 Glossary** **3**

**4 User Requirements Definition** **4**
4.1 User Requirements . . . 4
4.2 User Non-functional requirements . . . 6

**5 System Architecture** **7**
5.1 Architectural Patterns . . . 7
5.2 Reasons and Requirements . . . 7
5.3 Overall Structure . . . 7
5.4 Structure Components . . . 8
5.5 Interaction Relationship . . . 10

**6 System Requirements Specification** **11**
6.1 System Functional Requirements . . . 11
6.2 System Non-functional Requirements . . . 15

**7 Appendices** **18**
7.1 Properties of Pieces . . . 18
7.2 Rules Of The Jungle Game . . . 18

# 1 Preface

**Expected Readership:** Game Users, Software Managers, Investors.
**Version:** 1.0 beta
**Updates:** N/A
**Change Rationale:** N/A

# 2 Introduction

## 2.1 Purpose

To create the "Jungle Game" which could be operated under specified rules satisfaction perfectly with fun for users.

## 2.2 Scope

- Provide users with an efficient platform to play "Jungle Game", users could play the game correctly, intuitively, and smoothly.
- The users shall be able to gain happiness and satisfactions, as well as improve problem-solving skills and logical thinking abilities through playing this game.
- Our game is provided to the user group who are glad to play or enjoy playing the "Jungle Game"(no matter whether users understand the rules or not, they could get the specific rules in **User Requirements Definition Part**.) and we only provide Player-vs-Player mode for exactly two users in one game. All the round orders should be abiding by users consciously so that we `do not` provide any supervisory constraint.

## 2.3 References

The copy of the introduction of The Jungle Game.

## 2.4 Overview

This Software Requirements Specification is organised into five parts:

- The third section will defines the technical terms used in the document to help users understand the specification.
- The fourth section will describe the services provided for the user.
- The fifth section will provide an overview of the system architecture from a high level through illustrating system module components.
- The sixth section will provide a more detailed description of functional and non-functional requirements from a system's perspective.
- The seventh section will provide several appendices of the document, including properties of pieces table, and rules of the jungle game.

# 3 Glossary

This section mainly defines the technical terminology in this document and corresponds its abbreviation to its definition to help the users can better understand the document.

| Abbreviation | Meaning |
|---|---|
| CLI | Command Line Interface |
| GUI | Graphical User Interface |
| JG | Jungle Game |
| JVM | Java Virtual Machine |
| MVC | Model-View-Controller |
| NFR | Non-Functional Requirement |
| OOP | Object-Oriented Programming |
| REQ | Requirement |
| SR | System Requirement |
| TNFR | Types of Non-Functional Requirements |
| UR | User Requirement |

Table 1: Abbreviation Table

# 4 User Requirements Definition

## 4.1 User Requirements

- **UR-01**: The game board (scene) contains 7 columns and 9 rows of squares(7x9=63) for movement (See Figure 1). It has four parts, **Den**, **Trap**, **Water** area, and **General Land**;
- **UR-02**: Two **Den** squares lands in the center of the boundary rows of the playing board (Indicated in Red in Figure 1).
- **UR-03**: Six **Trap** squares are lands, and each three located to the two side and in front of the dens (Indicated in Yellow in Figure 1).
- **UR-04**: Two **Water** areas, each one formed by 3 rows and 2 columns squares as a 3x2 rectangle located in the center row, and two regions is separated by 1 column (Indicated in Blue in Figure 1).
- **UR-05**: Several **General Land** squares located in the rest of the areas as general squares (Indicated in White in Figure 1).

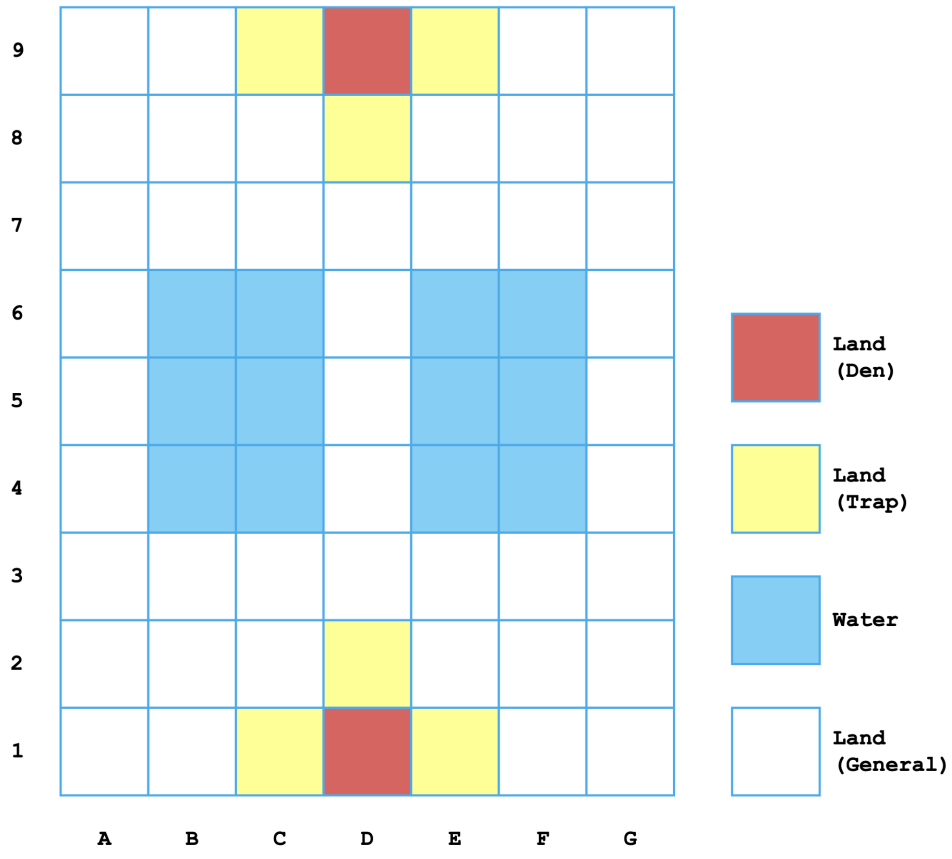


Figure 1: Game Board

- **UR-06**: The system provide two camps and two users to play against each other.
- **UR-07**: Users could mover freely on the playing board and can capture opponents' pieces according to piece priority and special REQs.
- **UR-08**: The user needs to enter the opponents dens and cannot enter his or her own dens.
- **UR-09**: There are altogether eight animals in the game, which exist certain ranks for capture among them:
  **Elephant**>**Lion**>**Tiger**>**Leopard**>**Wolf**>**Dog**>**Cat**>**Rat**(">" represent the capture relationship).
- **UR-10**: For capture between same rank, it depends **Capture-first** REQs which means only the mover in current round could capture another users' animal. There is also Exist a specific capture case that **Rat could capture the Elephant as well**.
- **UR-11**: Rat cannot capture the animals who are **different from its capture-before position**. It could only capture the animals when **they are both in the water or on the land**.
- **UR-12**: In each turn, one player **MUST** choose to move. Meanwhile, the user could only move **ONE** own piece in total of **ONE** square vertically or horizontally, no diagonal movement is provided.
- **UR-13**: For special area movement restrictions(water area), only Rats can enter the water area, and Lions and Tigers can jump across the river horizontally or vertically.
- **UR-14**: For each round, a player only could take one step on four positive directions($\uparrow, \downarrow, \leftarrow, \rightarrow$) with only one animal.
- **UR-15**: The system initializes **Three** traps for each users, and both users could enter their own traps or the enemy's traps. Pieces could capture any enemy piece in the trap squares, regardless of the rank REQs.
- **UR-16**: The user can win the game in two ways, by capturing all the opponent's pieces or by entering the opponent's dens.
- **UR-17**: The game board always starts with the **Initial State** (See Figure 2) in each first round.

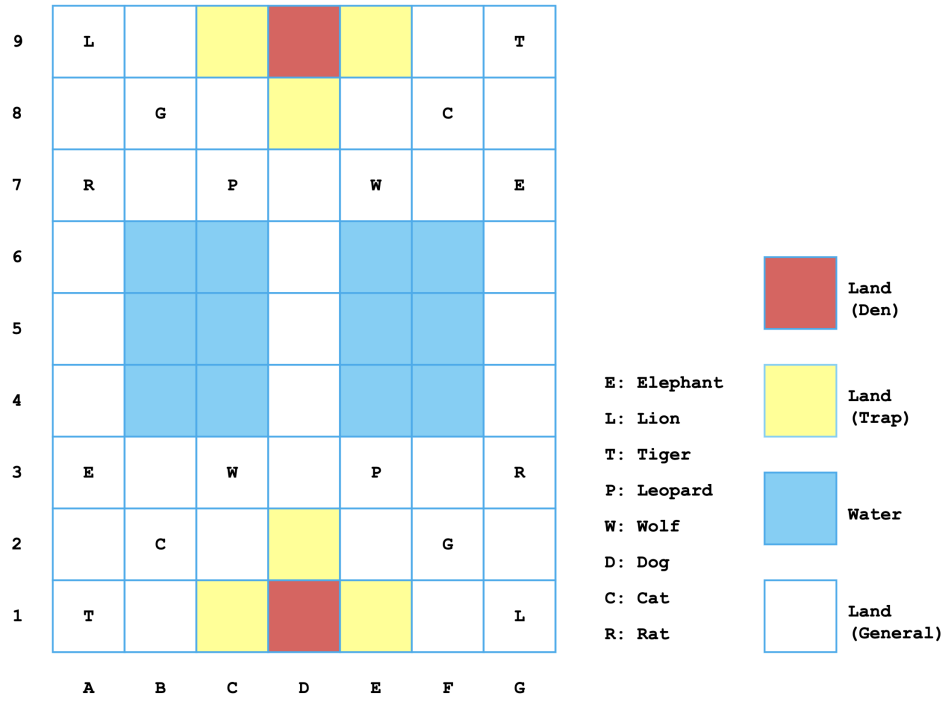


Figure 2: Initial State of the Game Board

## 4.2 User Non-functional requirements

- **UR-18**: The "Jungle game" runs as a command-line game.
- **UR-19**: Each movement must be played in the 7x9 board range.
- **UR-20**: It must and only have two actual players(computer player is not provided) for one game.
- **UR-21**: If a player leave the game, it will automatically judge the opponent's victory and reset the game.
- **UR-22**: Players couldn't skip the round without movement.
- **UR-23**: Players could capture their own animals as well.

# 5 System Architecture

## 5.1 Architectural Patterns

The **Model-View-Controller (MVC)** separates the presentation from the interaction from the system data. The entire game is divided into three mutually logical building blocks. The **Model** component building block game manages all data and accepts actions that transmit user commands. The **View** component defines the game board and the way the pieces are presented to the player. The **Controller** component manages the player interface, such as selecting the direction of movement, and these interactions are transmitted to the view and model.

## 5.2 Reasons and Requirements

1. The MVC pattern allows the system to be separated into 3 parts, in which each part is responsible for different and exclusive tasks with less overlap. It results in a good encapsulation of each functional part, and enhances their cohesion, which is also benefits the API design of the system.

2. The MVC pattern allows the system to increase reusability of the functions for further functional addition, which leads to low coupling for inheritance between superclasses and subclasses.

3. The MVC pattern allows the system to be used when multiple command data and ways of interacting with the data exist. It could also be used in the future when the need for interaction and presentation of player commands is finally known.

## 5.3 Overall Structure

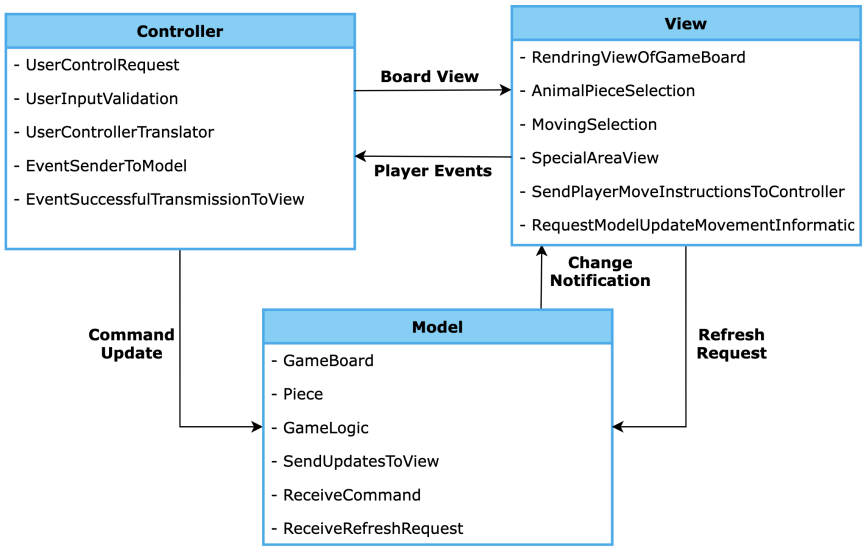


Figure 3: MVC Architectural Patterns

The overall structure is divided into three parts includes `model, view and controller` (See Figure 3). For model part, it contains game board, piece(animal) and game logic for playing. For view part, it mainly contains users' operation and visualisation contents with CLI interface to control the animals movement. For controller part, it mainly receives the users' operation with translator to turn it into model language and update to model part for further logical process, meanwhile returns notification to view part for users' interaction.

## 5.4 Structure Components

- **Model**

- `Game Board`: Provide a GameBoard class storing different squares based on specific properties and functionalities to support Pieces' movements and captures.

- `Piece`: Provide a Piece class to represent each animal piece acting on the Game Board.

- `Game Logic`: Provide a GameLogic class containing game rule and provide logic judgement on the game activities.

- `Send Updates To View`: Provide a SendUpdatesToView class to locate all updates on the Game Board, Piece, as well as other classes, and send them to the View part as Notifications.

- `Receive Command`: Provide a ReceiveCommand class to receive commands from the Controller part to check any actions on the Model mapped with events from View part.

- `Receive Refresh Request`: Provide a ReceiveRefreshRequest class to receive refresh request from View part, and send latest updates to View part.

- **View**

- `Rendering View Of GameBoard:` Provide the ability to render the board to the player, presenting the received commands to the player via command line mode.

- `Animal Piece Selection`: Let players to choose different pieces and presenting the pieces to the player to be able to distinguish the differences.

- `Moving Selection`: Provide the player with the ability to choose the direction and show the player to make the choice to move up, down, left and right.

- `Special Area View`: Special areas are displayed on the board, and special areas have special movement rules and predation rules as a way for player to differentiate from normal areas.

- `Send Player Move Instructions To Controller`: Transmit the player's forward movement command to the controller for the next step.

- `Request Model Update Movement Information`: The layout of the entire board is updated by requesting moves from parts of the model, as well as the survival of individual pieces, or win and loss situations.

- **Controller**

- `User Controller Translator`: Functionally provides translator to convert the users' request into machine language that model part could useful for logical implementation.

- `User Input Validation`: Functionally provides the inspection mechanism for revising of users' request translation.

- `User Control Request`: Functionally receives the control request from users to continue controller's afterwards operations dealing with the command.

- `Event Sender To Model`: Functionally provides dedicated tunnels for users' requests transmission to model part.

- `Event Successful Transmission To View`: Functionally provides feedback notification back to users while successfully sends the translated commands to model part.

## 5.5 Interaction Relationship

The section would show the application architecture of with the detailed connections and interactions of each components.

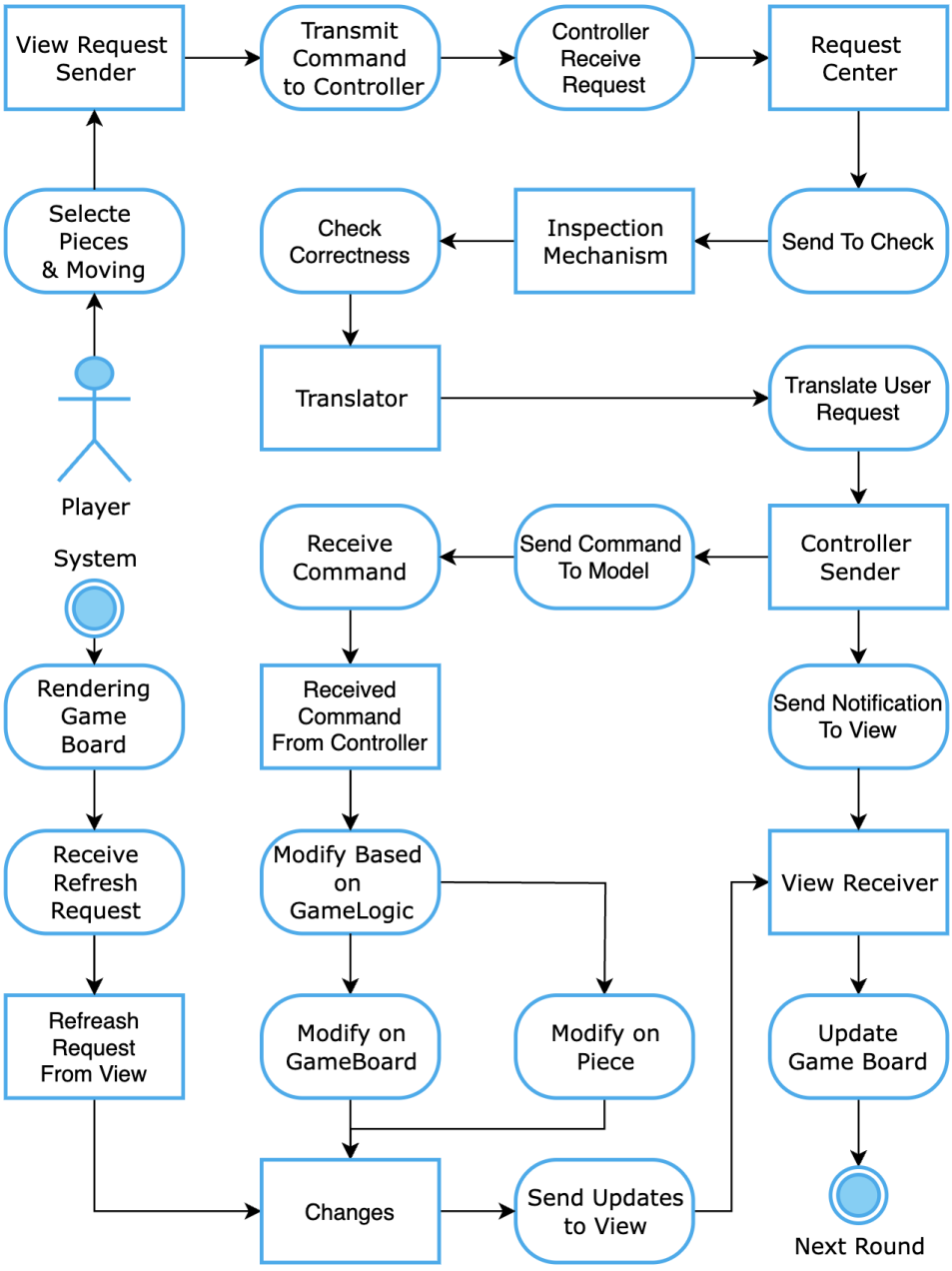


Figure 4: Activity Diagram of MVC Architectural Patterns

# 6 System Requirements Specification

This section describes functional and non-functional requirements in more details and futher details will added to the non-functional requirements.

## 6.1 System Functional Requirements

| ID | SR-01 |
|---|---|
| Title | Gameboard Properties |
| Requirement | System shall provide the functionality that each area will be clearly distinguished by the special symbol for the player, such as land(.)river area( ), trap(x) and den(*). These symbols will be used as the border of the area. |
| Rationale | Giving better understanding for players to distinguish the different parts for board. |
| Reference(SR,UR) | UR-01, UR-02, UR-03, UR-04, UR-05 |
| Priority | 1 |

| ID | SR-02 |
|---|---|
| Title | Game board Functionality |
| Requirement | System shall provide the functionality to the player with an area to move in and the player must complete all actions in this area. The system shall divide the board into different area for players, and different areas have different operation rules. |
| Rationale | Allow players to move around in defined areas. |
| Reference(SR,UR) | UR-06, UR-07, UR-08 |
| Priority | 1 |

| ID | SR-03 |
|---|---|
| Title | Winning condition |
| Requirement | System shall provide the functionality that players only two ways to win the game: one way is that system shall detect animal numbers for each player in each round, if one player's animal number return to 0 then the opponent would win. Another way is that system shall detect den's attribute for each players to find that whether there is animal in one player's den. If so, the opponent would win. |
| Rationale | Directly provide two ways for players to win the game. |
| Reference(SR,UR) | UR-16 |
| Priority | 1 |

| ID | SR-04 |
|---|---|
| Title | Capture |
| Requirement | System shall provide the functionality to determine capture situation according to the rank comparison between animals which follow the rules that mentioned in player requirement. After each capture, system will automatically detect the animals number for each players to detect the winning condition. |
| Rationale | Realise the capture function for game process and detect one of the winning condition correctly for players. |
| Reference(SR,UR) | UR-09, UR-10, UR-11, SR-03 |
| Priority | 1 |

| ID | SR-05 |
|---|---|
| Title | Special Areas |
| Requirement | System shall provide the functionality to set the water area, this area has special rules, different animals here have different movement rules and capture rules. |
| Rationale | Giving better understanding for players to know the special rules of the water area |
| Reference(SR,UR) | SR-02, UR-04, UR-11, UR-13 |
| Priority | 1 |

| ID | SR-06 |
|---|---|
| Title | General Movement |
| Requirement | System shall provide the functionality that the player MUST move one and only one of own pieces horizontally or vertically on the Game board which includes up, down, left, and right, excluding diagonal movement, and all movements shall fellow the Game board Functionality and Properties.<br>System shall also provide the functionality that the player is able to do the above mentioned movements by selecting four positive directions($\uparrow$, $\downarrow$, $\leftarrow$, $\rightarrow$) with a target piece. |
| Rationale | To satisfy the rules and enable players to move pieces intuitively. |
| Reference(SR,UR) | SR-02, UR-12, UR-14 |
| Priority | 1 |

| ID | SR-07 |
|---|---|
| Title | Rat Movement in Water Area |
| Requirement | System shall provide functionality that Rat could get into water and move one square in each round, however, if the rat is on land previously, it couldn't move to the water if the water square exist another Rat. |
| Rationale | Satisfy the rules |
| Reference(SR,UR) | SR-02, UR-04, UR-11, UR-13 |
| Priority | 1 |

| ID | SR-08 |
|---|---|
| Title | Other Animals Movement about the Water Area |
| Requirement | System shall provide functionality that Tiger and Leopard could straddle the water area to the next land square, however, it couldn't move if there exist Rat on the route to opposite of water. Besides, other animals couldn't have any movement about water area. |
| Rationale | Satisfy the rules |
| Reference(SR,UR) | SR-02, UR-04, UR-11, UR-13 |
| Priority | 1 |

| ID | SR-09 |
|---|---|
| Title | Capture in the Water Area |
| Requirement | System shall provide the functionality that pieces in the water area cannot capture pieces in other areas. |
| Rationale | Satisfy the rules |
| Reference(SR,UR) | SR-04, SR-07, UR-11 |
| Priority | 1 |

| ID | SR-10 |
|---|---|
| Title | Movement in the Den |
| Requirement | System shall provide the functionality that pieces in the den cannot enter their own den, but could enter the enemy's den. |
| Rationale | Giving better understanding for players to know the special movement rules when they enter the den. |
| Reference(SR,UR) | SR-06, UR-08, UR-16 |
| Priority | 1 |

| ID | SR-11 |
|---|---|
| Title | Movement in the Trap |
| Requirement | System shall provide the functionality that all animal could move in trap with rank falls to the lowest that could be capture by any animal. |
| Rationale | Satisfy the rules |
| Reference(SR,UR) | SR-04, SR-07, UR-11 |
| Priority | 1 |

| ID | SR-12 |
|---|---|
| Title | Results of Abandonment |
| Requirement | System shall provide the functionality that players who give up the game are judged to have failed directly. |
| Rationale | Giving better understanding for players to know the results of giving up the game. |
| Reference(SR,UR) | - |
| Priority | 1 |

| ID | SR-13 |
|---|---|
| Title | Movement Boundary |
| Requirement | System shall provide the functionality that players cannot put or move own pieces outside the gameboard (See Figure **??**), and system shall not provide the palyer the operation to violate abovementioned movement boundary. |
| Rationale | Satisfy the rules, and set the movement range area. |
| Reference(SR,UR) | SR-01, UR-01 |
| Priority | 1 |

## 6.2 System Non-functional Requirements

| ID | SR-14 |
|---|---|
| Title | Fairness |
| NFR | System shall guarantee the correctness of rules operation and no movement retract for each players to reallize the fairness. |
| TNFR | Product Requirements |
| Rationale | For better playing experience with fairness. |
| Reference(SR,UR) | - |
| Priority | - |

| ID | SR-15 |
|---|---|
| Title | Configuration |
| NFR | System shall provide one of each eight animals for each players and the initial state of gameboard. |
| TNFR | Product Requirements |
| Rationale | For better playing experience with same configuration. |
| Reference(SR,UR) | - |
| Priority | - |

| ID | SR-16 |
|---|---|
| Title | Reliability |
| NFR | The system can respond correctly to the user's choices and corresponds to the priority of each rule, and the system needs to run smoothly and without errors. |
| TNFR | Product Requirements |
| Rationale | - |
| Reference(SR,UR) | - |
| Priority | - |

| ID | SR-17 |
|---|---|
| Title | Maintainability |
| NFR | The code is written in a way that facilitates the implementation of new features. In order that new requirements can be updated into the system even in the future. |
| TNFR | Product Requirements |
| Rationale | For better playing experience with same configuration. |
| Reference(SR,UR) | - |
| Priority | - |

| ID | SR-18 |
|---|---|
| Title | External elements |
| NFR | System shall only provide the rules and operation of the JG, instead of providing any external elements includes players' conflict, concious and so on. |
| TNFR | External Requirements |
| Rationale | For better playing experience with same configuration. |
| Reference(SR,UR) | - |
| Priority | - |

| ID | SR-19 |
|---|---|
| Title | Availability |
| NFR | System do not consider the impact of the network, the game runs on a single machine with high availability and no connection lag. |
| TNFR | Product Requirements |
| Rationale | Identify the factors required to ensure the specified level of availability. |
| Reference(SR,UR) | - |
| Priority | - |

| ID | SR-20 |
|---|---|
| Title | Linguistic elements |
| NFR | System shall only provide English version for JG playing. |
| TNFR | External Requirements |
| Rationale | For better playing experience with same configuration. |
| Reference(SR,UR) | - |
| Priority | - |

| ID | SR-21 |
|---|---|
| Title | Dependability elements |
| NFR | System shall dependent on and run in the envirnment with JVM. |
| TNFR | Product Requirements |
| Rationale | For supporting the system basic running envirnement. |
| Reference(SR,UR) | - |
| Priority | - |

| ID | SR-21 |
|---|---|
| Title | Portability |
| NFR | JG is very convenient, small files can be copied and transferred to another computer at any time. |
| TNFR | Product Requirements |
| Rationale | Provide the ease of porting software to other hosts and/or operating systems. |
| Reference(SR,UR) | - |
| Priority | - |

# 7 Appendices

## 7.1 Properties of Pieces

| Rank | Piece (en) | Piece (Abbreviations) |
|---|---|---|
| 8 | Elephant | E |
| 7 | Lion | L |
| 6 | Tiger | T |
| 5 | Leopard | P |
| 4 | Wolf | W |
| 3 | Dog | D |
| 2 | Cat | C |
| 1 | Rat | R |

## 7.2 Rules Of The Jungle Game

- The board of Jungle game consists of seven columns and nine rows of squares.
- The pieces move within the squares like in chess, not on the lines like in Chinese chess.
- Each side of the board has pictures of the eight animals and their names to indicate the initial position of the pieces. After the initial setup, these animal spaces have no special meaning in the game.
- There are several special squares and areas on the jungle board. Lairs are located in the center of the border rows of the board and are marked in Chinese.
- Traps are located on the sides and in front of the nests and are also labelled in Chinese.
- There are two water areas or rivers in the center of the board: each water area consists of six 2X3 rectangular squares and is labelled "River".
- In the middle between the edge of the board and the river, there is a single column of common land squares.
- Each side has eight pieces representing different animals, each with a different rank.
- Higher-level pieces can capture all pieces of the same or lower level. However, there is one exception.
- Rats can catch elephants, while elephants can't catch mice. The ranking of the animals, from high to low.
- Pieces are placed on the board with corresponding animal pictures, which are always displayed on the board.
- The rat is the only animal allowed to go up to the water square.

- Rats must not catch an elephant or another mouse on land directly from the water. Similarly, rats on land cannot attack rats in the water.
- If both pieces are in the water or on land, the rat can attack the opposing mouse.
- Lion and tiger pieces can jump across the river by moving horizontally or vertically. They move from one square by the river to the next non-water grid on the other.
- If there is a rat (whether friendly or enemy) on any of the water levels in between, this movement is not allowed. Lions and tigers can use this jumping movement to capture enemy pieces.
- Rats can eat elephants (but not from the water level).
- A piece can eat any enemy piece in a player's trap square, regardless of its level.