# Software Requirements Specification Report

for

Gambler Agent

Taylan Işıkdemir
Alper Güngör
Volkan Çetin
İlkcan Keleş

# Contents

# 1. Introduction

## 1.1 Purpose

Software Requirement Specification for Gambler Agent provides a complete description of all the functions and specifications of Intelligent and Learning System for Turn Based Games Project. Gambler Agent is an artificial intelligence agent for turn based games.

## 1.2 Scope

The intelligent system for turn based games are very common nowadays. Most turn based game developer companies design such a system. On the other hand there is not many instances of learning system for turn based games which are partially observable. Observable means all the game environment is open to each player. Tile and card  games are partially observable. Players only know the cards or tiles which are in their hands and thrown in previous turns.

Gambler Agent will be first implemented for a card game ‘King’. If the implemented agent does not reach to the desired level, then we will switch to ‘Okey’ game and implement a learning agent for that game. The game without learning system will only become a multiplayer game.

Gambler Agent will be available for internet users. All web browsers which support Flash is appropriate for game environment. Because of Gambler Agent is a learning a system, it is going to play better in the process of gaining experience by practising.

## 1.3 Definitions , Acronyms and Abbreviations

| | |
|---|---|
| DB: | Database |
| AI: | Artificial Intelligence |
| GUI: | Graphical User Interface |
| ILSTBG: | Intelligent and Learning System for Turn Based Games |
| API: | Application programming interface |
| Http: | Hypertext Transfer Protocol |
| UML: | Unified  Modelling Language |
| IEEE: | Institute of Electrical and Electronics Engineering |

## 1.4 Team Info

Our team name is ErikSoft. The members of the team are İlkcan Keleş, Volkan Çetin, Alper Güngör, Taylan Işıkdemir. All members are willing that Gambler Agent project will be used in masket and hope that the players enjoy the game against the agent.

### 1.4.1 Team Structure

We will have regular internal meetings once a week, every sunday. All four members of our team has to attend the meeting. In these regular meetings important decisions about Gambler Agent project will be made. All members will share their own ideas and the brightest ideas are going to be discussed.

We will also have external meetings with our teaching assistant Ümit Ruşen Aktaş and our project sponsor Özgür Alan. The topic of the meetings with the student assistant is mostly about the planning of project. The meetings' content with our sponsor is the details of the design and development process.

### 1.5 Overview

The rest of this document contains an overal description of the Intelligent and Learning System for Turn Based Games and specific constraints. There will be more specific details and information about the project content, general requirements and environment.

# 2.Overal Description

## 2 .1 Product Perspective

ILSTBG is aimed toward online game players who like challenges. The product is independent and mostly self-contained. Gambler Agent is like an AI engine, whose all internal designs are belong to our project group. However the project process contains development of interfaces to make the AI engine available for testing and publishing. This means at the end of the year we will have one complete game with AI players that can be played online and one game without AI players that can also be played online.

### 2.1.1 System Interfaces

Gambler Agent to be developed is a stand alone engine that is going to be integrated within a King game. All components must execute on Windows , Linux , MacOS.

### 2.1.2 User Interfaces

The user interface of the game is going to be achieved through the web browsers. Interface will be user friendly and very easy to get used to. Game screen will be embedded into web browser. There will be chat box, timer, information table and some buttons to add agents and start the game.

### 2.1.3 Hardware Interfaces

There is no constraint on which kind of hardware must be used. There are common hardware devices that are enough to interact with the game.
These are

- monitor screen:
  Screen provides visual information to user.
- keyboard:
  Keyboard provides user to communicate with other users
- mouse:
  Mouse is the main tool for playing King game
- speaker:
  Speaker provides audial information to user.

### 2.1.4 Software Interfaces

The required software products for Gambler Agent project are
Java SE 1.6, JDBC, ActionScript, Netbeans, Web Service.

## 2.2 Product Functions

User functions :

Register : New users can register to the system by filling the required form.

Login : Users have to login in order to join the game. After logging in, a user can access other functionalities.

Join Table: Users can join any existing tables which is not full.

Create Table : Users can create table and wait for other players, if they do not want to join an existing table.

Add Agent : If a user wants to play against the game bots, he can simply add an agent with a chosen level.

Chat : Users can easily chat with other users through our chatbox.

Check Game State : Users may want to see the status of the game during playing. It is enough to click a button in order to see the scores.

Dismiss Agent : If a user wants to add a new human player, he can easily dissmiss the agent from table.

Agent functions:

Learning : The agent will be able to learn the game by training our collected data. There will be different kinds and levels of agents.

Playing : While playing, agent must give real time responses. For quick respond, we will use some learning methods and choose the appropriate one.

## 2.3 User Characteristics

All internet game players from all over the world will be a potential user for the product. There is no age limit. The only constraint for the users is being familiar with the rules of the games. The more skilled players we attract, the more qualified agents we will offer.

## 2.4 Constraints

The number of online users are restricted because of the server's capacity. Response time of web service is a performance constraint. While collecting training data, synchronization of agents- i.e threads- is an important constraint.

The same constraint holds for learning mechanism. In this phase, the available number of CPUs in department's supercomputer (NAR) is limited since there are many requests for CPUs. For this reason, data collecting process will be slow. For security reasons, data should be well protected and should be held in a secure server.

## 2.5 Assumptions and Dependencies

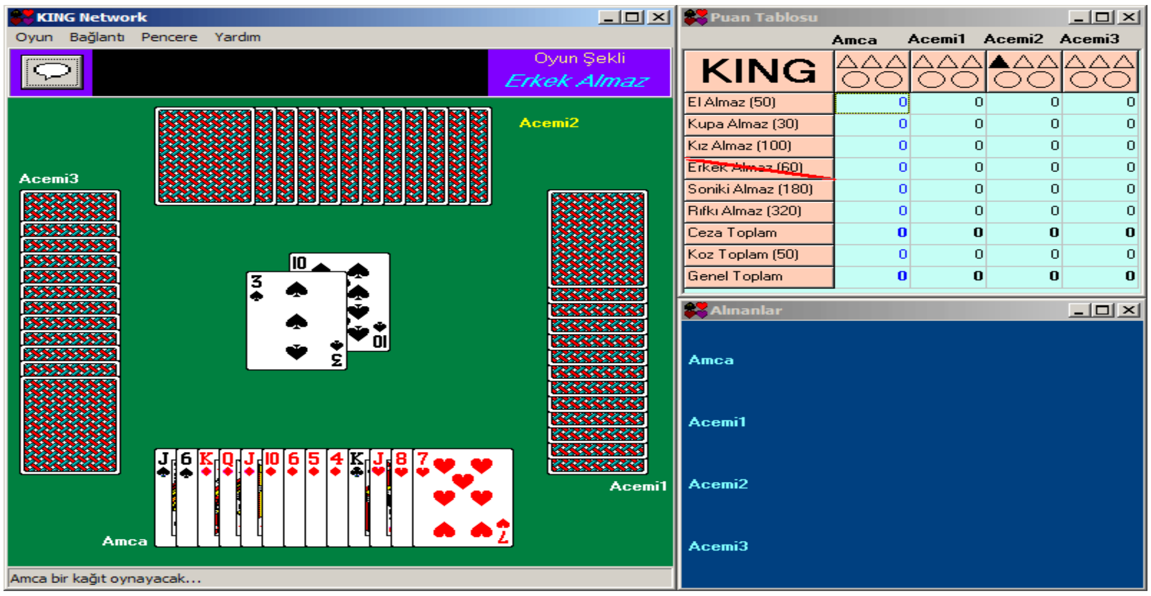
Learning phase for 'King' game may be unsuccesfull due to the properties of this game. Since there are seven different games in the 'King' game and the game is not fully observable, learning process in this limited time and processor number is a difficult task.

If this situation occurs,we will switch to the other game called 'Okey' which may be easier to implement a learning system for.

Since Gambler Agent application is a web-based Project, there will not be any system dependencies. It is enough to have the necessary plugins for the browser. We will make it available for all browsers.

# 3.Specific Requirements

## 3.1 External Interfaces



| KING | Amca | Acemi1 | Acemi2 | Acemi3 |
|---|---|---|---|---|
| El Almaz (50) | 0 | 0 | 0 | 0 |
| Kupa Almaz (30) | 0 | 0 | 0 | 0 |
| Kız Almaz (100) | 0 | 0 | 0 | 0 |
| ~~Erkek Almaz (60)~~ | 0 | 0 | 0 | 0 |
| Soniki Almaz (180) | 0 | 0 | 0 | 0 |
| Rıfkı Almaz (320) | 0 | 0 | 0 | 0 |
| Ceza Toplam | **0** | **0** | **0** | **0** |
| Koz Toplam (50) | 0 | 0 | 0 | 0 |
| Genel Toplam | **0** | **0** | **0** | **0** |

Figure-1: User Interface

**User Interface:**

**<u>Game window</u>** : It will be opened as 480X640 and it will have option for users to make window full screen.

**<u>Chat Box</u>**: Chat box will be available for the players who are on the related table.

**<u>Buttons</u>**:There will be a few buttons that have different functionalities like adding an agent ,opening information table,starting game.

**<u>Timer</u>**:The timer keeps the track of the time after game started.

**<u>Information Table</u>**:The information table show information about the players their scores and game status.

**The software interfaces:**

**Java SE 1.6** : is a widely used platform for programming in the Java language. We will use it for developing main features of the Agent

**JDBC** : is an API for the Java programming language that defines how a client may access a database. It provides methods for querying and updating data in a database.

**ActionScript** : It is a script language and it is used primarily for the development of websites and software targeting the Adobe Flash Player platform. We will use it to implement interface of game.

**Netbeans** : is an integrated development environment (IDE) for developing with Java.

**Web Service** : is typically an API that is accessed via http and executed on a remote system, hosting the requested service.

## 3.2 Functional Requirements

### 3.2.1 HumanPlayer

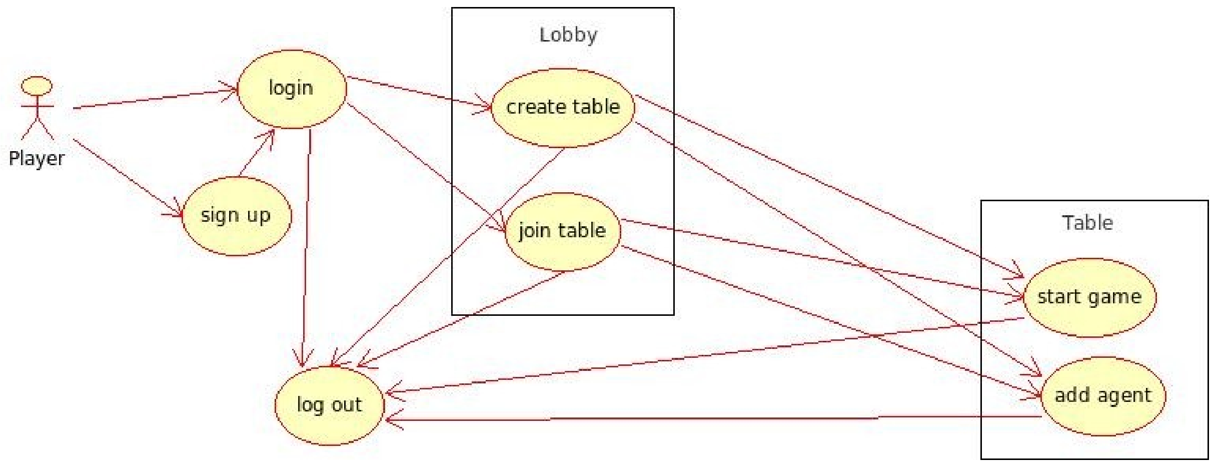


Figure -2:Use-case for human players

The human player can register the system via submitting a register form. This form includes the fields user name, e-mail and password as obligatory and birth date, place as optional.

A registered user will sign in to the system by entering user name and password. After the signing in, the game will automatically direct the user to the lobby. When a player in lobby, there are two options for this player. He can either create a new table or join an existing table. He can also look over the players in existing tables and their points and profiles for human players, their levels for agents.

If a player creates a new table, he will become the owner of the table. For this reason, he will some additional functionalities. One of these functionalities is when all players are ready, he will be able to start the game. He can add various levels of agents to the table to overcome the lack of enough players at that table. He can dismiss an agent if he wants to play with another human. He can also dismiss a human player from the table if he exceeds the time limit for a turn.

If a person quits during the game, he will penalized by decrementing his points. When the player logs out, the updated information of  the player will be transferred to the database.

### 3.2.2 Agent

#### 3.2.2.1 Design of  Agent

Learning process consists of four main steps.In the first step training data should be stored.This training data includes state-action pairs.This pair can be kept during the game.But  another way to keep this data set is to log the games and later on convert them to data sets.This approach provides that state-action definitions to be flexible.This game logs can either be obtained from virtual agents or human players.We are planning to obtain data sets from virtual agents.

These virtual agents' levels will be different and game log will keep this information for later use.In the second step four types of agents will be defined. These are rule-based,search-based,random and hybrid(which uses both search and rules) agents.These are general types different kind of agents can be modelled by small changes.By the help of these agents,training data sets for supervised learning will be constructed.In the third step some machine learning algorithms will be tried.Some possible methods are Neural Network,support vector machines.The main purpose is defining a function F that it gets a state as input and returns an action as output.In the last step this F function will be transformed to a web service.These learning processes will be done on NAR machines.

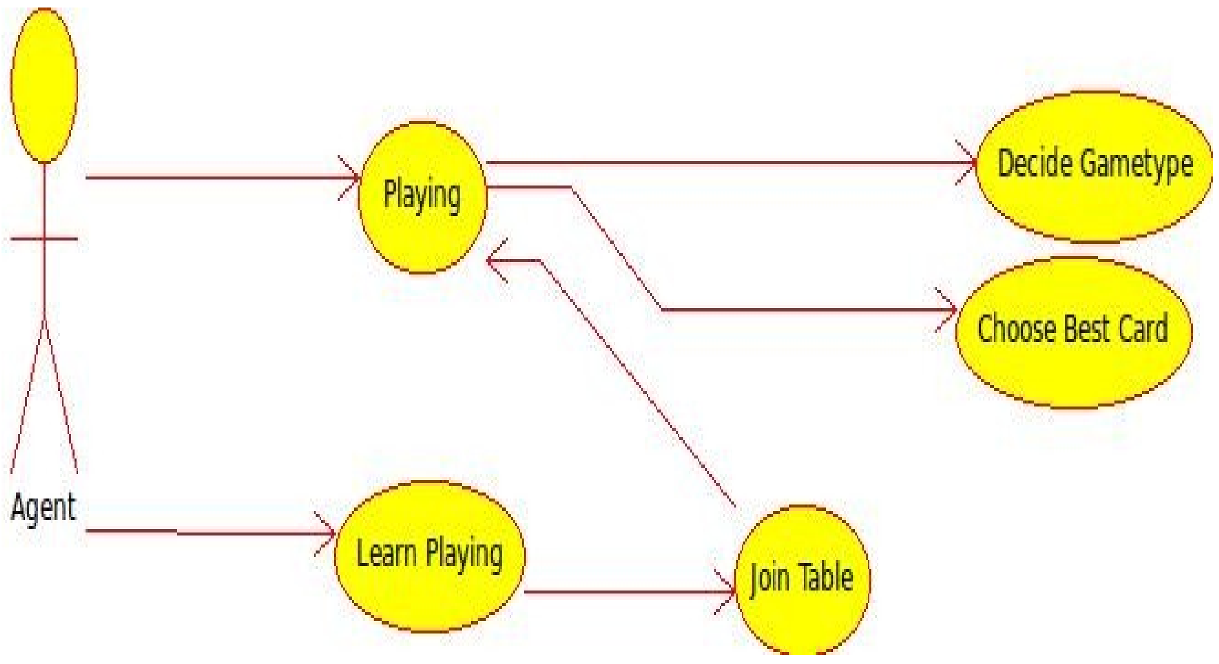


Figure-3:Use-case for agent players

#### 3.2.2.2 Final Gambler Agent

When agent player is called to the table, the game status will be given to the agent player. Then the agent will play the game according to the game status. It will become idle, when it is not the agent’s turn. When it is agent’s turn, if it is time to decide the game type, it will decide game type according to the cards in its hand and available game types. Otherwise, it will choose the best card and throw it, according to the learning, chosen game type and thrown cards. After the game finishes, the data which agents learn from will be updated.

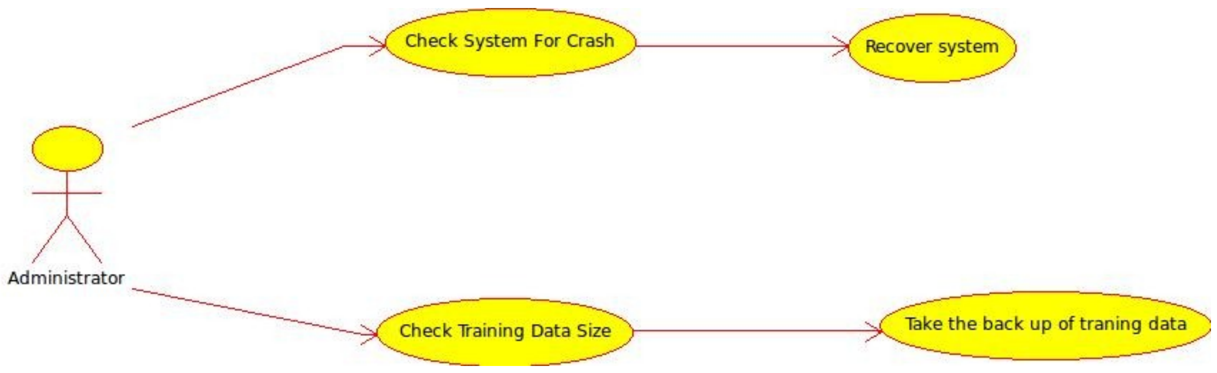


Figure -4:Use case for administrator

The administrator of the system will check the system  whether  the system is crashed or not in some fixed intervals.If system crashes the administrator will recover the system.Another mission of the administrator is checking  whether traning data size exceed the treshold or not.If the training data size exceeds the treshold then the administrator will take the back up of the data.

## 3.3 Nonfunctional Requirements

### 3.3.1 Performance Requirements

In a game the agents must respond  in a fixed time period between turns because  Gambler Agent is an online game application so it requires real-time responses.The number of  game tables in the system will be limited since every table is a new thread.The  number of  logged users will be limited,too.

### 3.3.2 Design Constraints

Reports during development phase of the project will be prepared according to IEEEStd830. UML design and flow charts are going to take part in that phase. JavaDoc style comments will be used for documentation of the source code. In the implementation phase SVN is going to be used for accordance in development team. Any changes of DB schema during implementation phase will be recorded and initial design will be updated accordingly.

### 3.3.3 Software System Attributes

#### 3.3.3.1 Usability

Gambler Agent application's client side will be very easy to use and enjoyable. Users of the application should not waste time to get used to play the game. Agent usage will be very simple for newcomers since the only thing they have to do is just pressing a buton and choosing the level of the agent.

#### 3.3.3.2 Reliability

User accounts will be hold in the DB, so if an error occurs user will not suffer from this event. User accounts keep information about their previous games and some personal information. For this reason it  is important to keep the data consistent.

#### <u>3.3.3.3 Availability</u>

Online game applications have the risk of server failure because of excessive number of users. Important precautions will be taken to avoid this circumstance such as setting up checkpoints and some regular interval backups. So application will always have a stable state which we offer to users.

#### <u>3.3.3.4 Security</u>

Gambler Agent application’s security will be guaranteed to all users. No personal information will be shared with or sold to any other third party companies.

#### <u>3.3.3.5 Maintainability</u>

Our design will be flexible. Whenever a new functionality is needed for application, it will be easy to integrate because our design is going to have a layered structure.

#### <u>3.3.3.6 Portability</u>

Gambler Agent is an online game application so it will be able to acces it from any platform.

# 4.Data Model and Description

## 4.1 Data Description

The project will include Lobby, Table, Game, Player ,Card,Training data,Action and State objects.

### 4.1.1 Data Objects and Relationships

Lobby  object will be the topmost layer  through  other data objects.Lobby will include the logged players and  created tables.When a player joins a table then it will be removed from Lobby object and transferred to Table object.

Table object  will have  a Game object and  max four  Player object.The players can be either type of human or agent.

Game object will do the main job.Game object will include  Card object. This object will deliver the cards to players when game is started.This object also will keep the game chart and point chart.The information of played game types are kept in the game chart.

Player object  will keep the  array of  delivered cards  and his/her available game type chart.In this object also players thrown cards and points wil be kept.

Training data object will keep the  data of the all game which is played according to a game type.At the end of  a specific game an instance of training data object will be created and send to the database.This data will be formed from State-Action  pairs.

State data object will keep the information of the  instant game data.The actions will be decided according to tihs state object.

Action object will keep the  move according to given state.

### 4.1.2 Complete data model

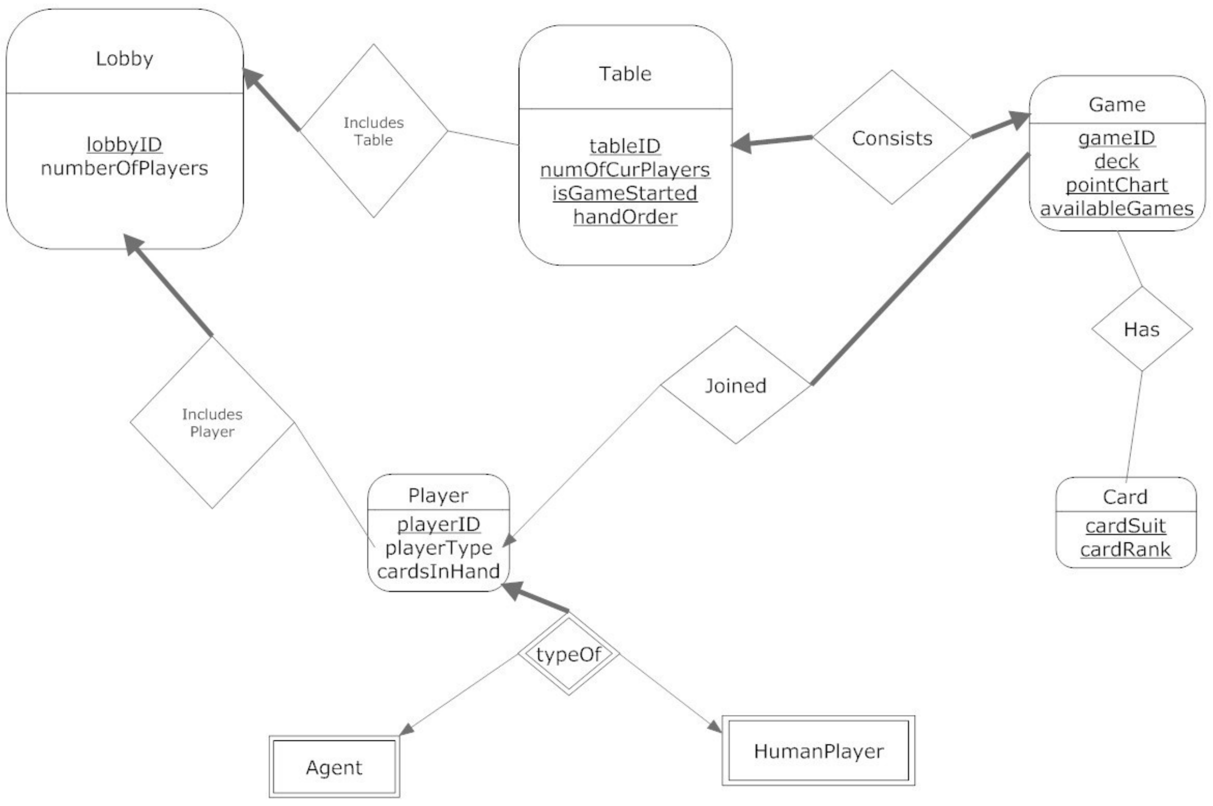


Figure-5 ER Diagram of System

# 5.Behavioral Model and Description

The project will be an online gaming system in which human players and agents is included. The system behavior will be different according to the player's type. While developing agents, there will be a training stage.

## 5.1 Description for software behavior

Initially, we will develop rule-based agents. These agents will play the game amongst them in order to collect training data. After collecting enough data to train our agents, agents will learn how to play the game better gradually.

When our agents learn how to play, we will begin to allow users to add agents to the table. While playing the game with human players, our agents will continue learning via saving the game information to the database at the end of the game.

System will give the game status to the agent if players add an agent during the game. After getting the game status, agent will know whether it must choose a game type or it must choose the best card to throw. Then our agent will call required function to handle this.

## 5.2 Activity Diagrams

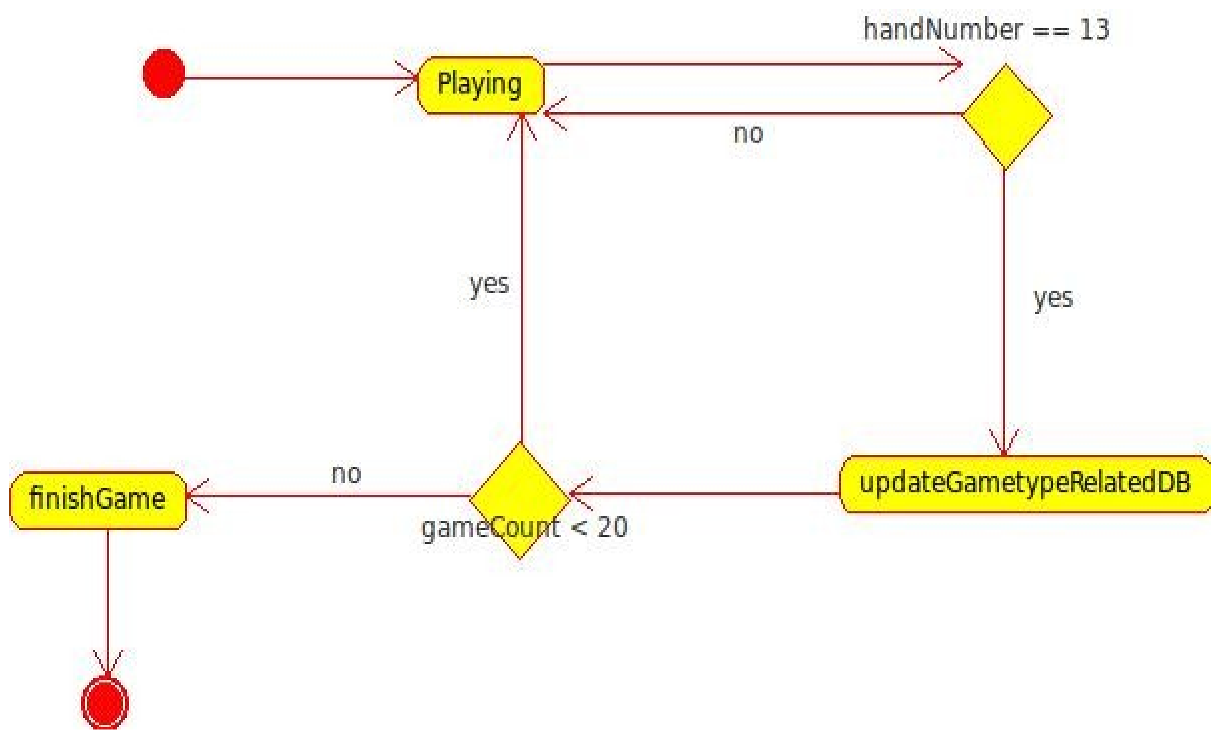


Figure-5 : Collection of Training Data Activity Diagram

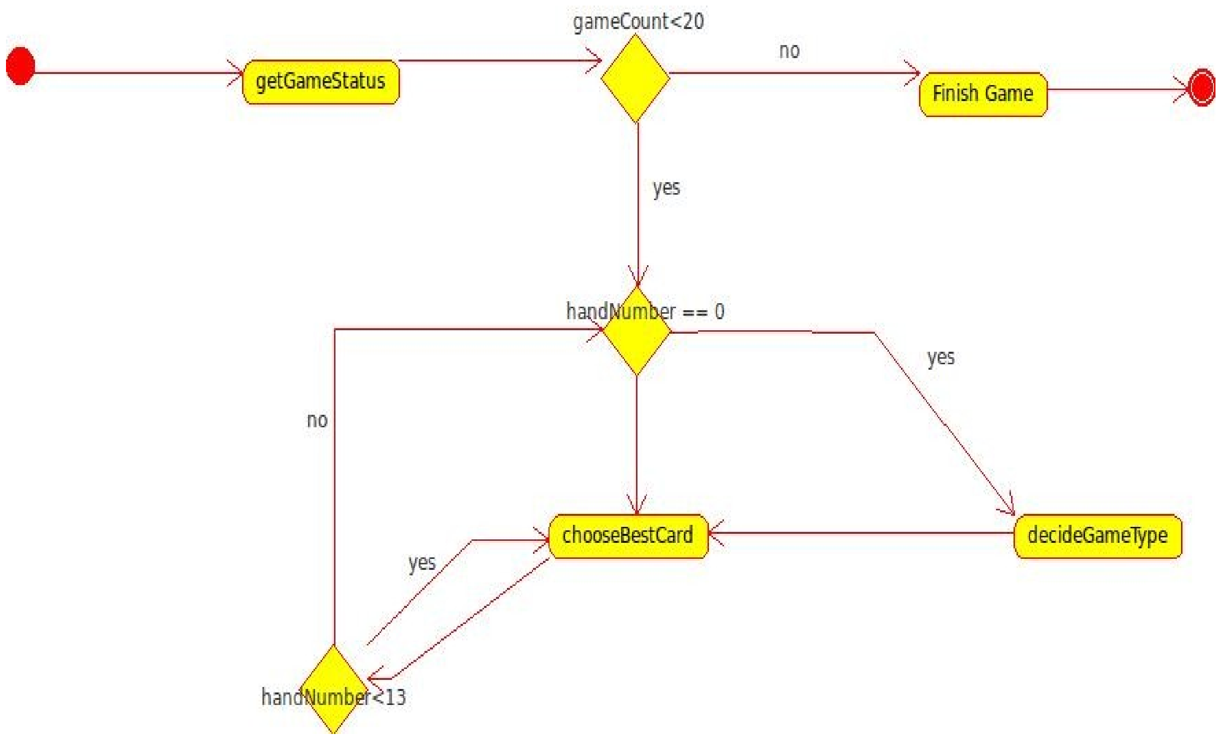


Figure-5: Agent Playing Activity Diagram

# 6.Planning

## 6.1 Team Structure

Our team consists of Alper Güngör, Taylan Işıkdemir, Volkan Çetin and İlkcan Keleş. In our team, there is not any project leader. We decide on important issues all together. Every stage of the work will be decided by group, and accomplished by whom it is assigned to.

We will also have weekly meetings with our assistant Ruşen Aktaş and our sponsor Özgür Alan to take some advice and to decide the next stages.

## 6.2 Estimation

Until the end of this semester, it is planned that we will write rule based agents and start to collect training data. There will be many agents which is only rule based according to all game types. Games will be played amongst these agents and will provide us enough data to start AI stages.

In the second semester, we will focus on mainly learning stages and finally when our agents are ready to join the game, we will supply user interface. We will have to make a choice between some learning algorithms which are reinforcement learning and supervised learning.

## 6.3 Process Model

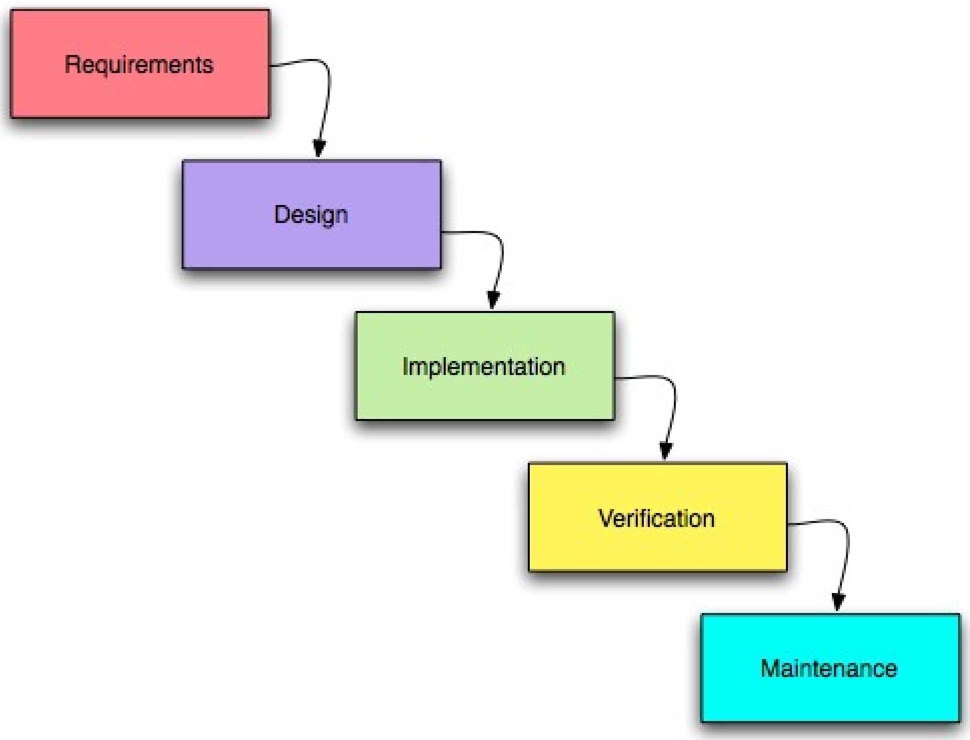


Figure-6: Waterfall Process Model

We have seperated the development stages to different parts which are requirements, design, implementation, verification and maintenance. For this reason, Waterfall process model is chosen. At each phase, required documentation will be done. The document for the requirement phase is this document.