CS2013 - Group 4 - Project 11 - The Turing Game

Requirements Document

1. Introduction

1.1. Overview - Purpose of system

The purpose of this system is to enable users to play a mobile app game that tests their ability to differentiate a bot(AI) from a human player. More specifically, the user enters a chat conversation paired randomly with either a bot or a human player. The user is given a specific topic to talk about for a specific time period, after which the user must guess whether they were talking to a human or a bot.

1.2. Scope

For this project we will need to create a small server to connect the human players, we will do so using the cloud messaging/matchmaking service. We will make use of IBM's watson servers and integrate them into the project to generate the bot's dialog to humans. We will be using Android Studio to develop the app using the Java language and the Android SDK.

1.3. Objectives and success criteria

Our success will be determined by how effectively we can create a server to match-make users, and implement a working bot on certain topics that could pass for a human. Another success indicator will be user interface, implementing a clear, interactive game interface with easy navigation of the app.

1.4. Definitions, abbreviations

SDK - A software development kit (SDK) is typically a set of software development tools that allows the creation of applications for a certain software package and system.

AI - Artificial Intelligence is intelligence demonstrated by machines, in contrast to the natural intelligence displayed by humans and other animals.

1.5. References

CCNA Data Center DCICT 640-916 Official Cert Guide; Shamsee, N.Klebenov, D. Fayed, H.;Cisco; p. 934.

https://en.wikipedia.org/wiki/Artificial_intelligence

https://www.ibm.com/watson/about/

https://en.wikipedia.org/wiki/Turing_test

2. Current system

The client doesn't have an existing system. However, given the brief, the project will somewhat resemble some existing implementations of the concept (for example, "ELIZA" which was designed to give users an illusion of understanding on the part of the program, https://en.wikipedia.org/wiki/ELIZA)

3. Proposed System

3.1. Overview

As mentioned before, the purpose of this project is to test users ability to differentiate an interaction with a bot(AI) to an interaction with a human player. We aim to do this by creating a server that will randomly connect users to other human players, and also match users to bots. We will assign topics to limit the conversation and train bots on certain topics. In the future this could be built upon by having many different bots for different topics, but as this is the first version we will be keeping the bot's conversation limited.

3.2. Functional Requirements

- The functional requirements specified by our client are that we create an app that lets users play against either a bot or a human, and guess which one they were chatting to.

- The client also specified that he would like us to program the app to ask users for comments and feedback on what gave the bot away if they guessed correctly, post-game to collect data for further analysis.

- To avoid players leaving the chat for long periods of time and disrupting the flow of the game, a timer will be implemented which limits the conversation time.

- The client had no preference towards iOS or Android platforms, so we will be building the app for the Android platform using Android Studio.

3.3. Non-functional requirements

- The system must be scalable to allow for the client to be able to build upon our work in the future if he wishes to implement further improvements to the back end server, and train the bot with more dialog responses.

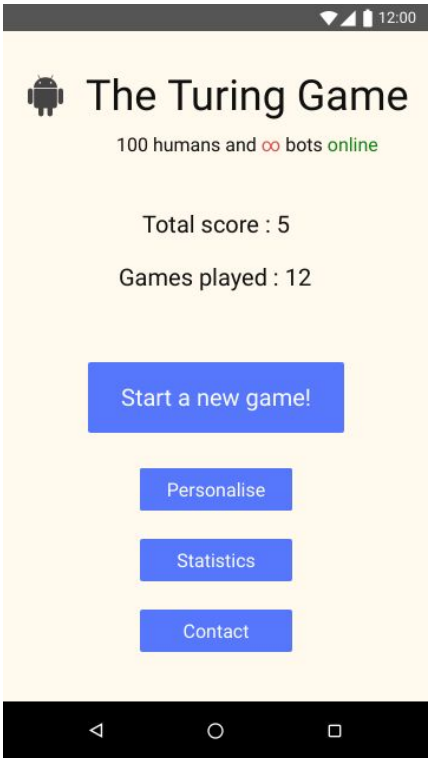
- The program should be easily accessible and easy to use by different software development teams that will be modifying the code, as usability is an important factor for this project.

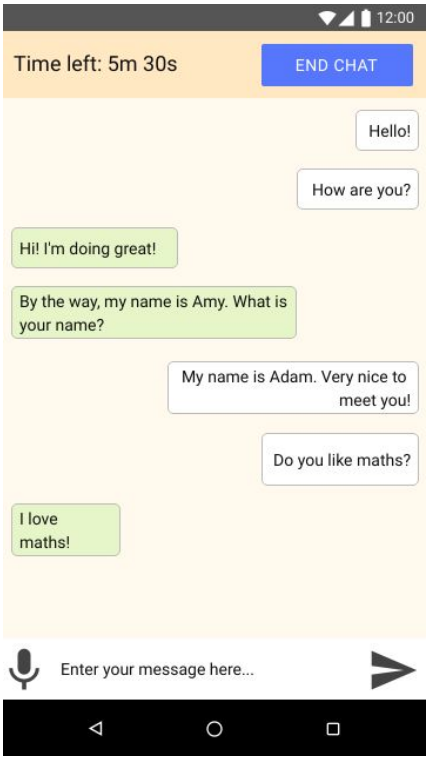
- A desirable non functional requirement would be that the app itself is user friendly and aesthetically pleasing.

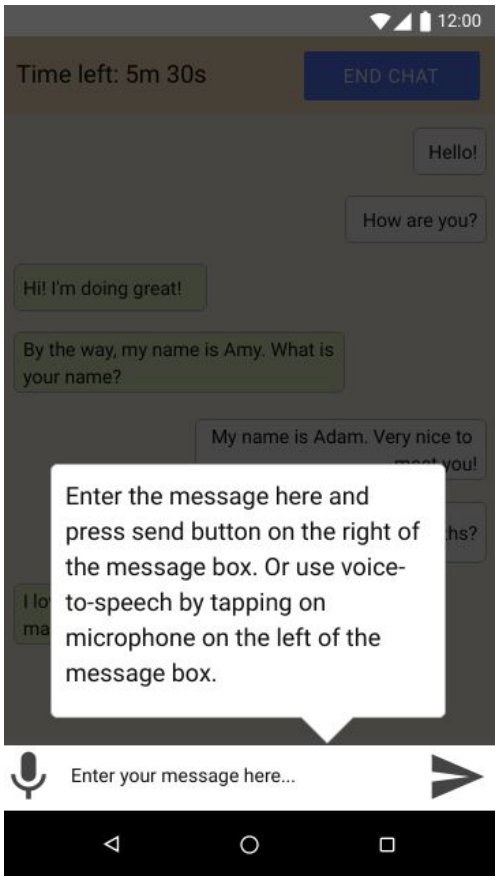
- Server latency should be kept to minimum to facilitate a free flowing conversation.

3.4. System prototype (models)

3.4.1.User interface mockups



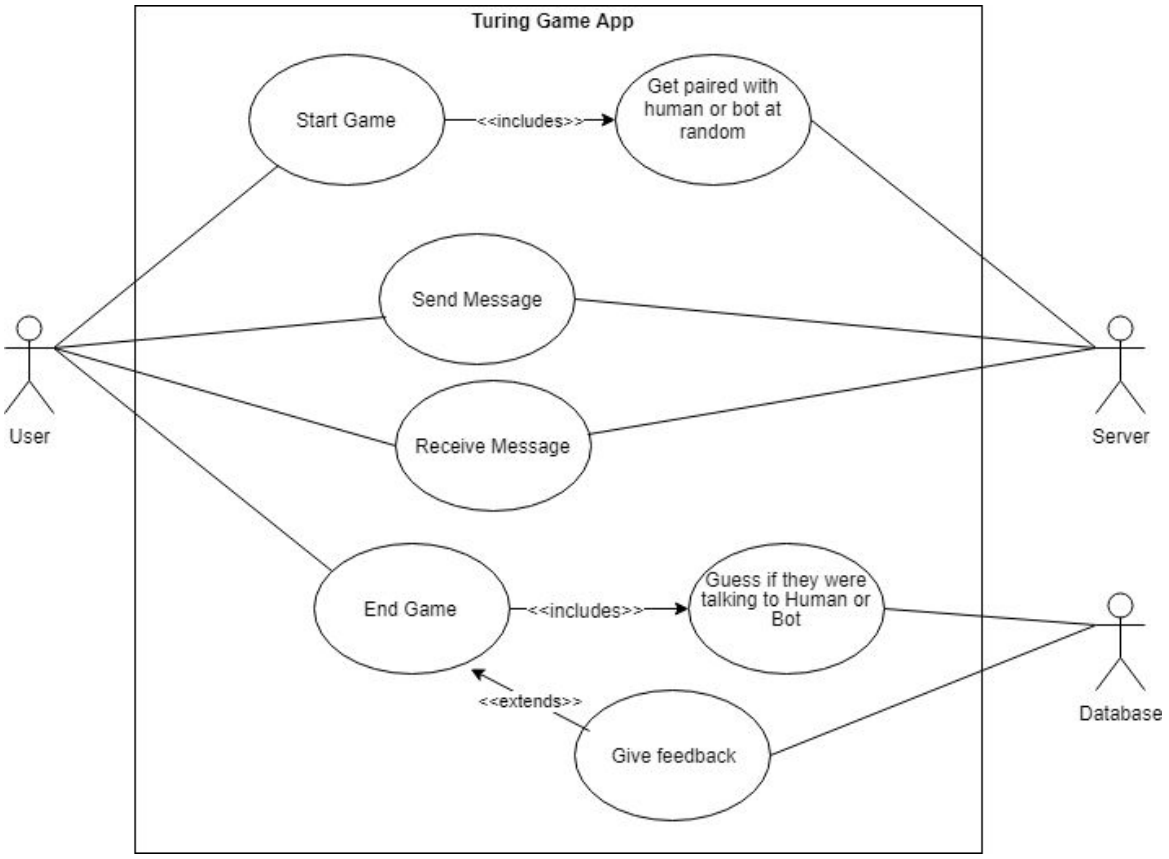




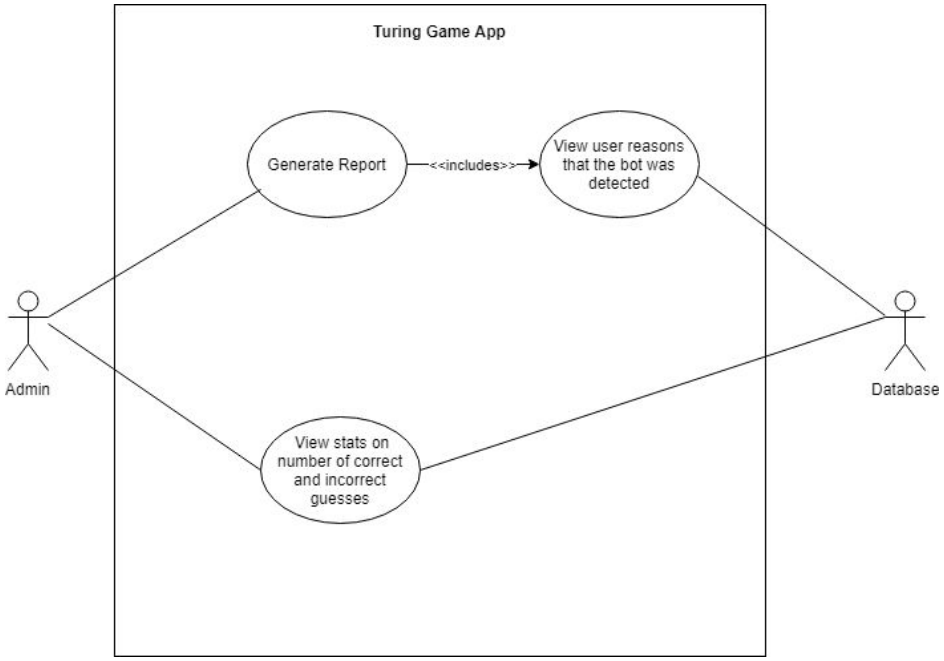
3.4.2.Use cases (including text narratives)

We have at least **2 systems: a server and an android app.**

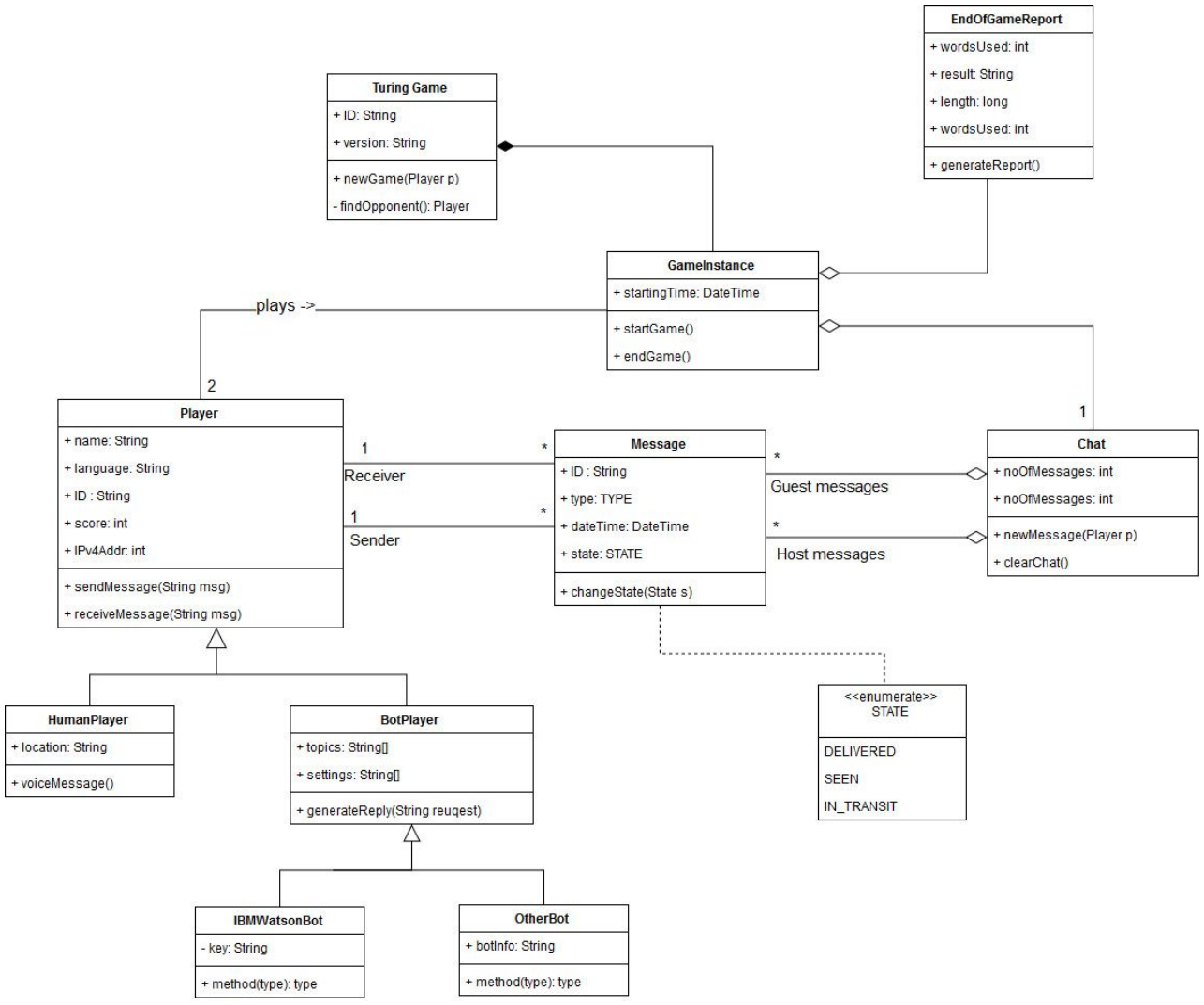
1. For **user** use to play a game



2. For **app developer/admin** use to generate reports and statistics



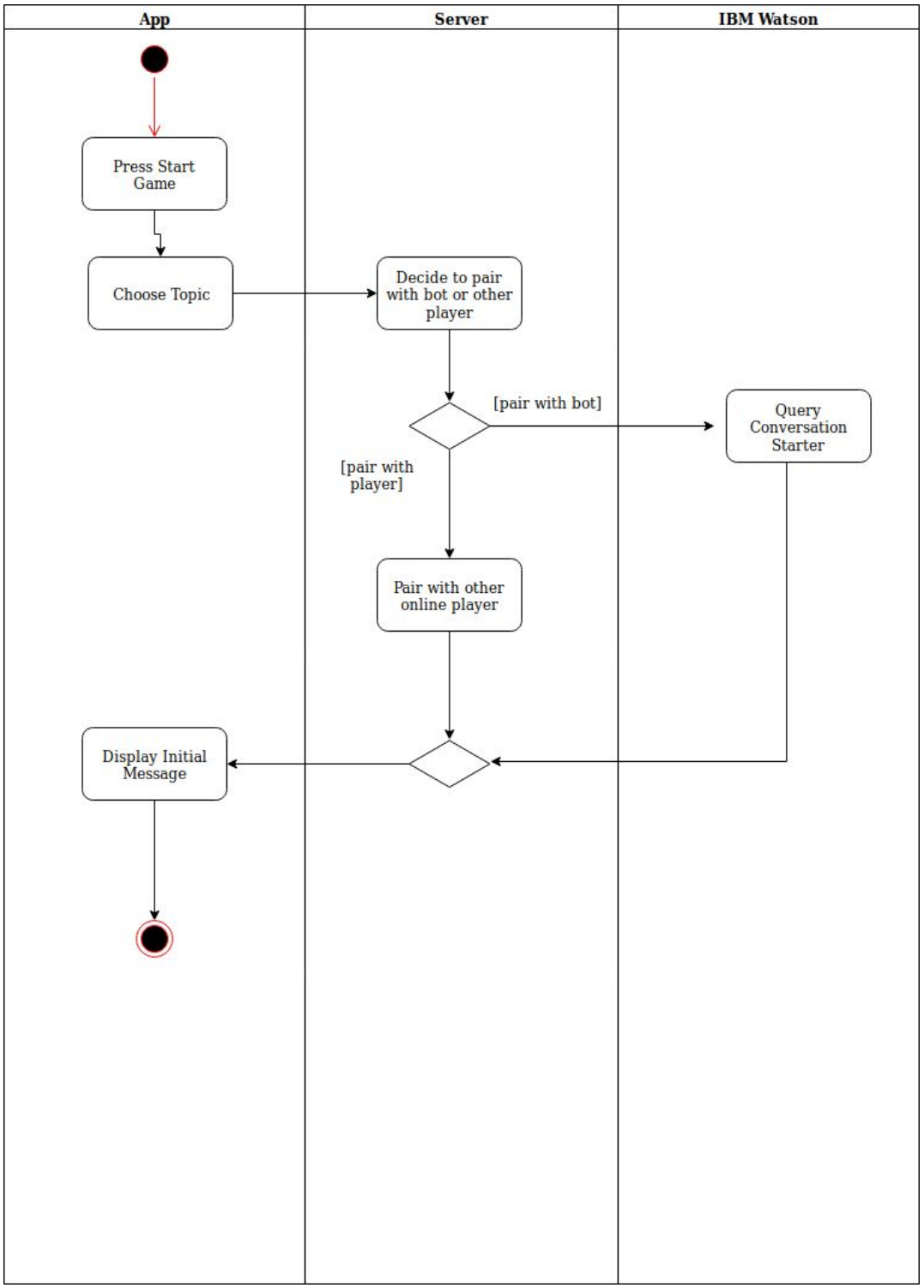
3.4.3.Object model


EndOfGameReport
+ wordsUsed: int
+ result: String
+ length: long
+ wordsUsed: int
+ generateReport()
Turing Game
+ ID: String
+ version: String
+ newGame(Player p)
- findOpponent(): Player
GameInstance
+ startingTime: DateTime
+ startGame()
+ endGame()
plays ->
2
1
Player
+ name: String
+ language: String
+ ID : String
+ score: int
+ IPv4Addr: int
+ sendMessage(String msg)
+ receiveMessage(String msg)
1
*
Receiver
1
*
Sender
Message
+ ID : String
+ type: TYPE
+ dateTime: DateTime
+ state: STATE
+ changeState(State s)
*
Guest messages
*
Host messages
Chat
+ noOfMessages: int
+ noOfMessages: int
+ newMessage(Player p)
+ clearChat()
HumanPlayer
+ location: String
+ voiceMessage()
BotPlayer
+ topics: String[]
+ settings: String[]
+ generateReply(String reuqest)
<<enumerate>>
STATE
DELIVERED
SEEN
IN_TRANSIT
IBMWatsonBot
- key: String
+ method(type): type
OtherBot
+ botInfo: String
+ method(type): type

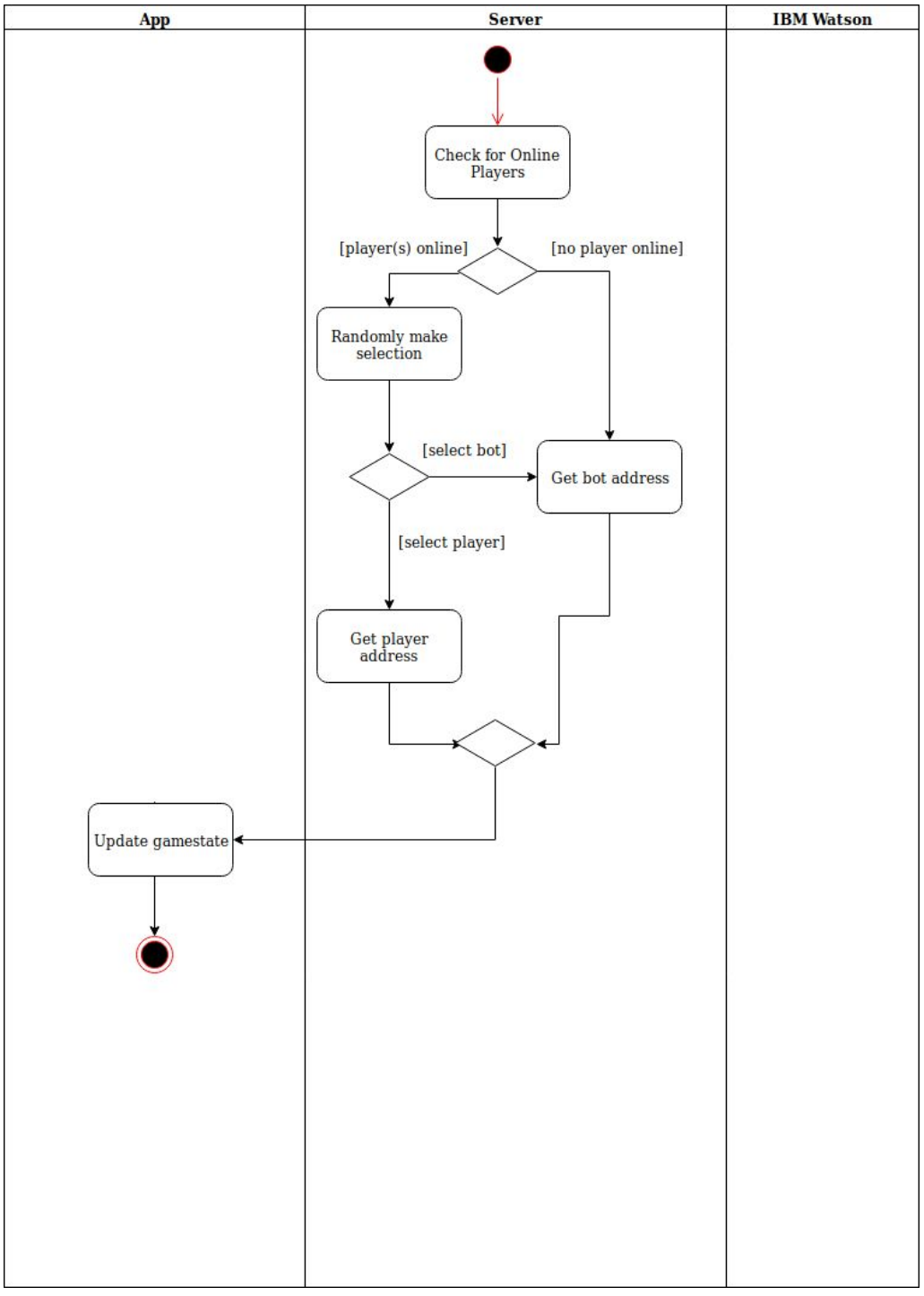
3.4.4.Dynamic model

High level activity diagram. Such as

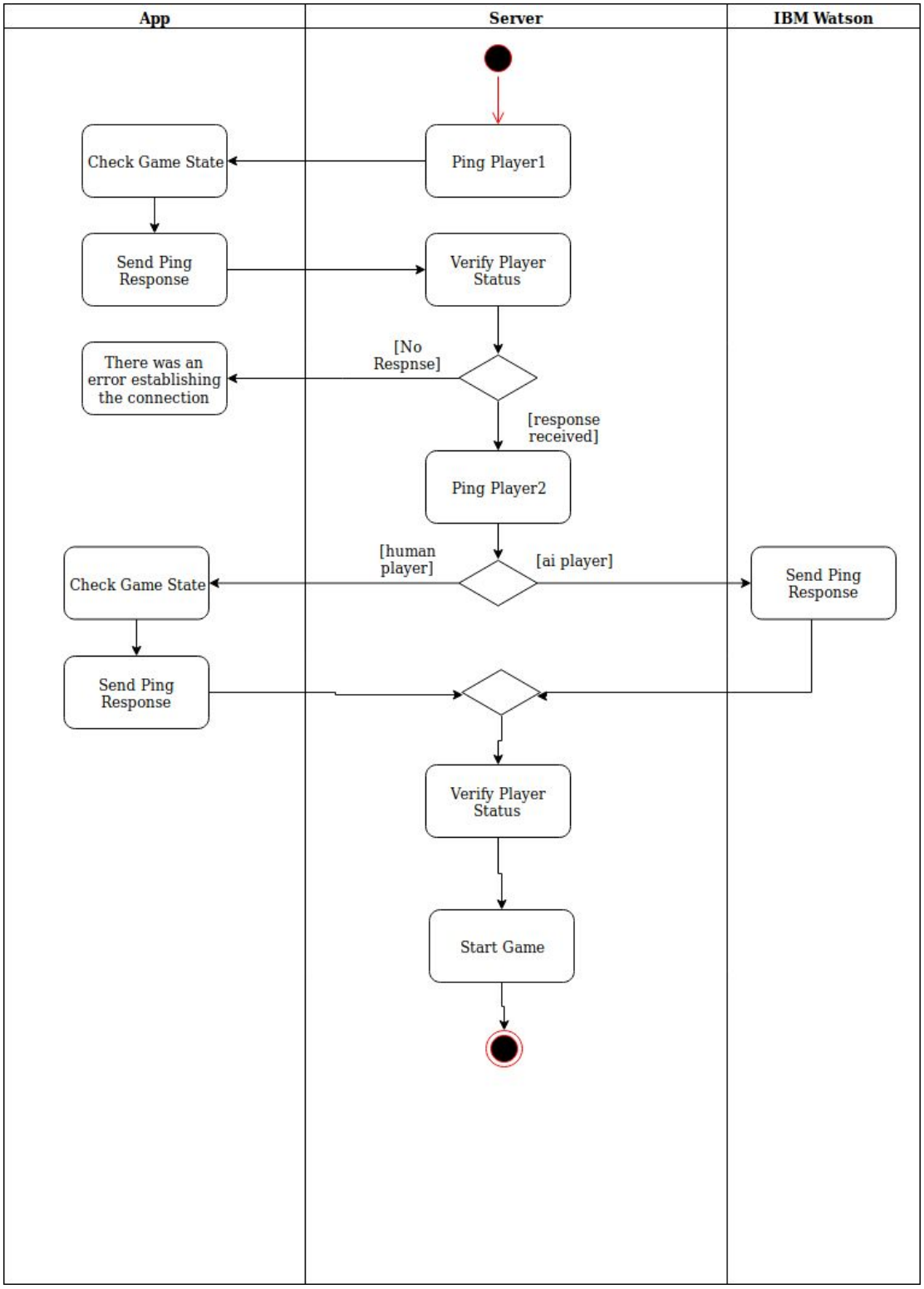
1. Start game



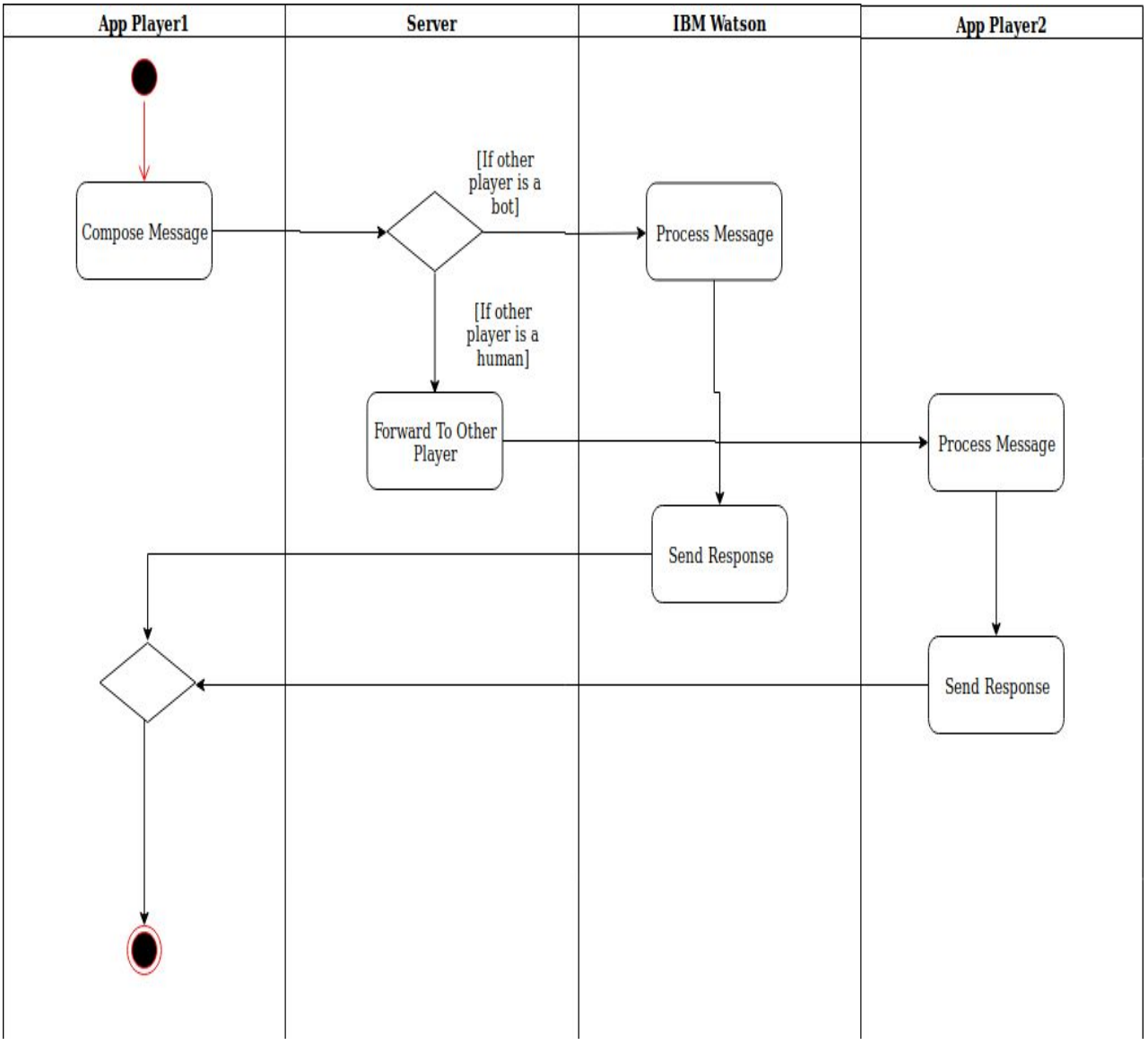
Find an opponent



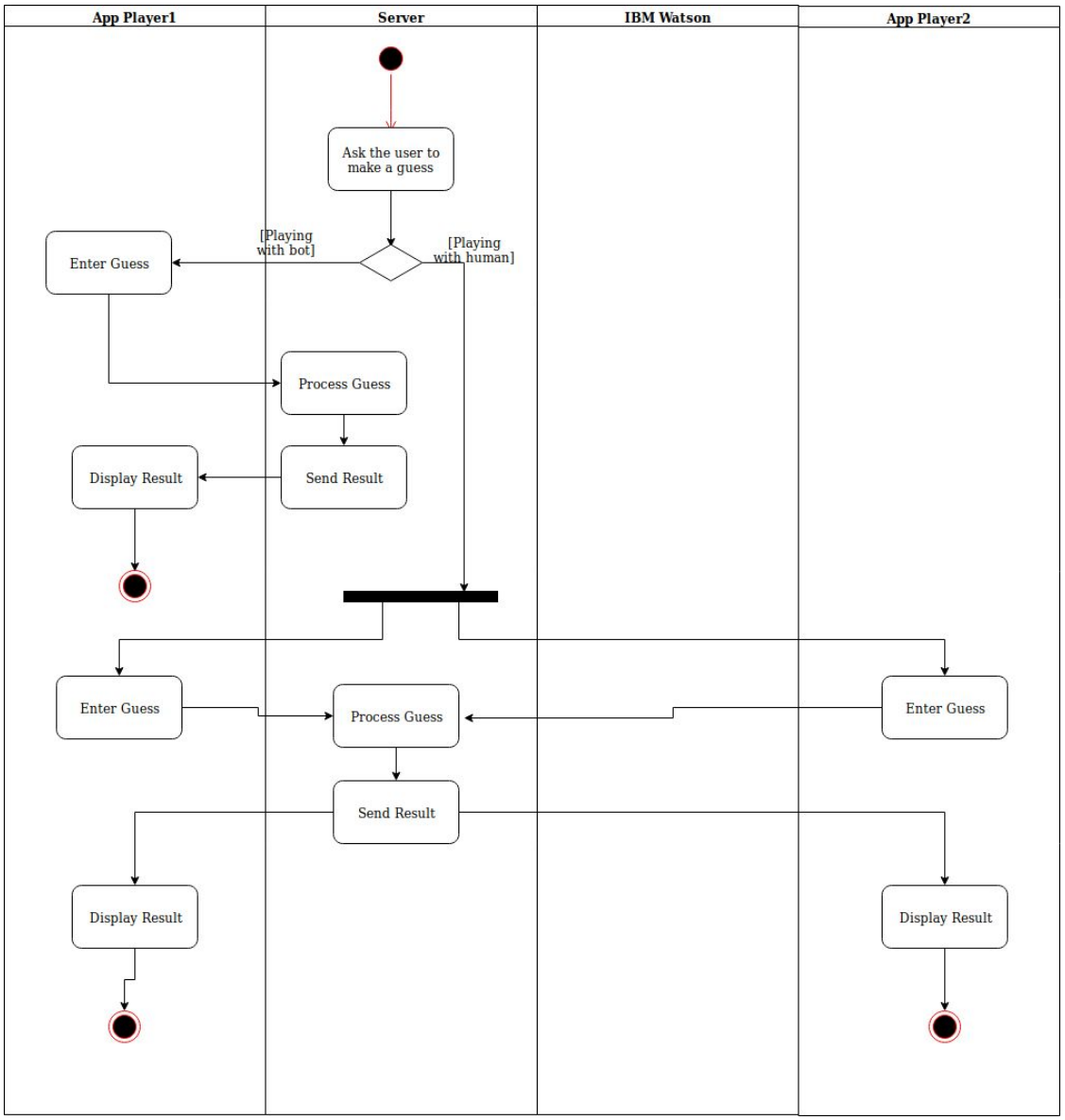
Establish connection



Exchange messages



Make a guess


App Player1
Server
IBM Watson
App Player2
Ask the user to make a guess
[Playing with bot]
[Playing with human]
Enter Guess
Process Guess
Send Result
Display Result
Enter Guess
Process Guess
Send Result
Enter Guess
Display Result
Display Result