

# Eriantys Protocol Documentation

Filippo Gandini, Federico Mazzucato, Nazzareno Messinò  
Group 44

June 21, 2022

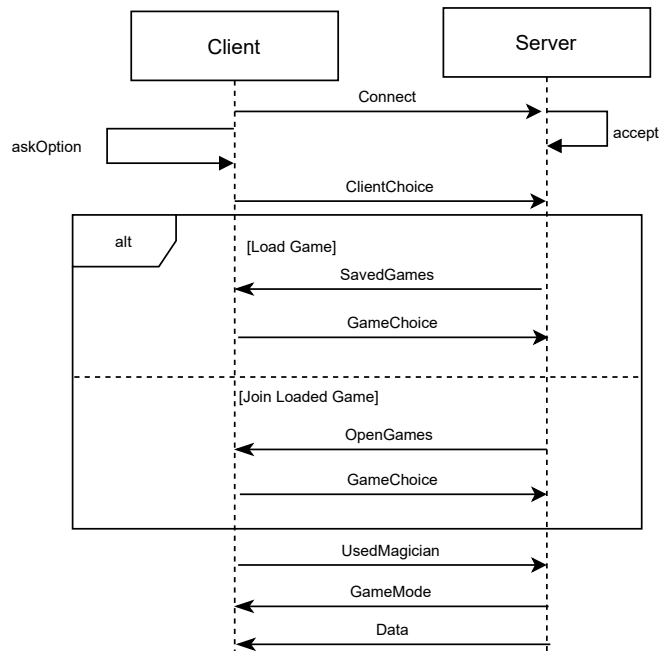
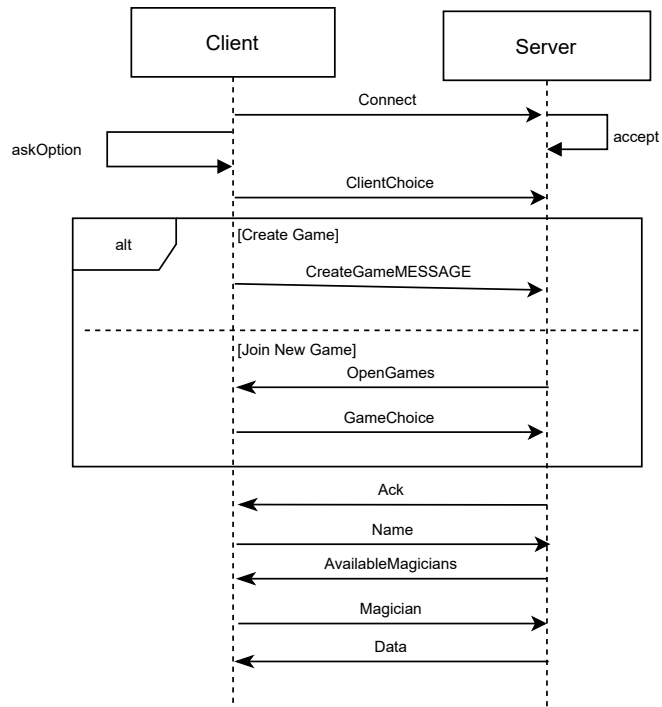
# 1 Scenarios

## 1.1 Access

The client connects to the server, communicates whether he wants to create, join or load a game.

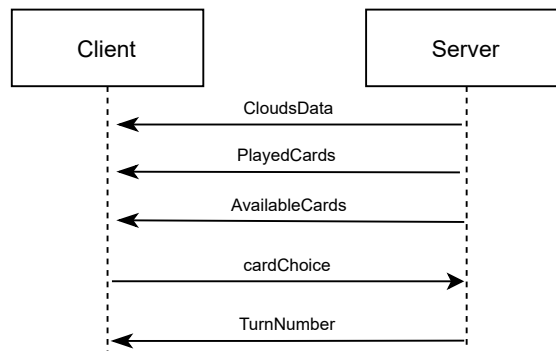
1. CreateGame: creates the game defining the game name and the game mode.
2. LoadGame: loads a game among the saved games on the server.
3. JoinGame: joins a new or a loaded game.

When the correct amount of clients are in the game, the client receives an ack. Finally, the server sends the game's data to start the match.



## 1.2 Plan

The client receives the information needed to play a card and then sends his choice. Finally when all the players have sent their card, each client receives his game turn.



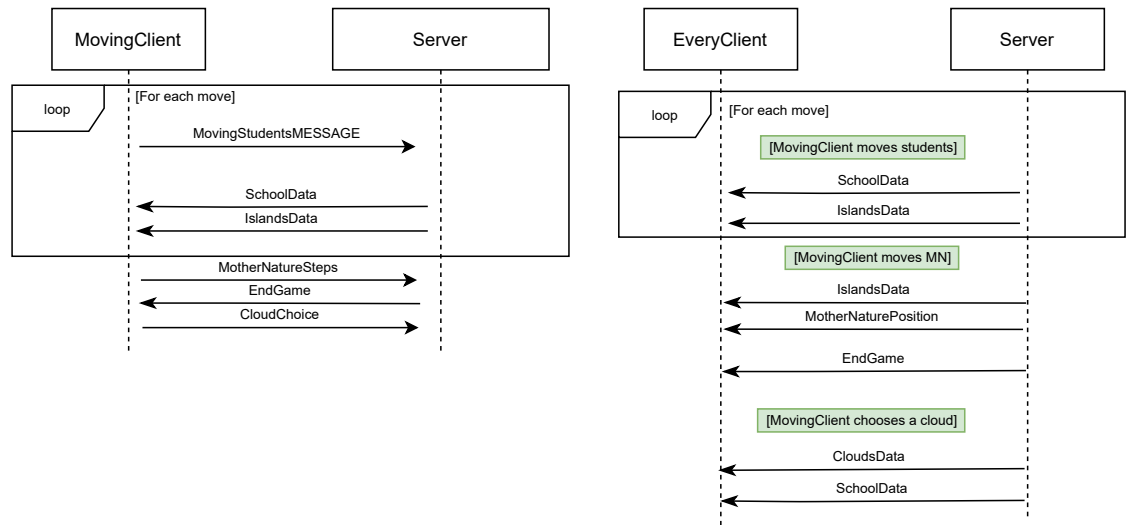
## 1.3 Move

In this phase the protocol is different whether the client is moving or is watching another player moving (because he is waiting for his turn to move or he has already moved).

### 1.3.1 Standard Mode

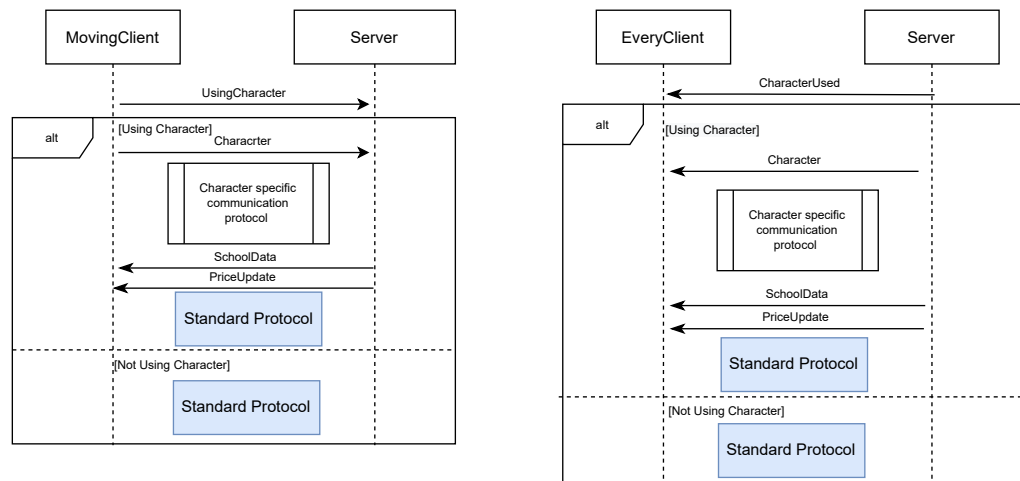
The moving client sends the information about his moves and receives a message communicating if the game has ended.

All the clients receive the updates of the game and the EndGame message.



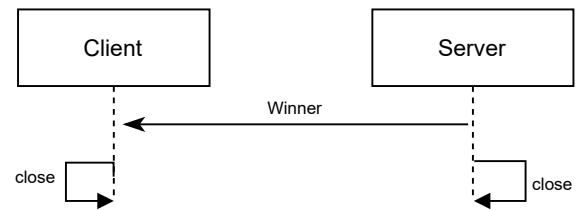
### 1.3.2 Expert Mode

The moving client declares if he wants to use a character during his turn. If he chooses to not use it, the protocol is the same of the Standard Mode. If he chooses to use a character he communicates the character he chose. Then, there are some character specific protocol messages before the standard communication (character-specific protocols aren't defined in this general document).



## 1.4 End

The server sends to the client who are the magicians who won the game.



## **2 Messages**

### **2.1 From Server to Client**

#### **2.1.1 SavedGames**

A list of strings which communicates the saved games loaded.

#### **2.1.2 OpenGames**

A map of strings and game modes which communicates the open games waiting for players to join.

#### **2.1.3 Ack**

A boolean value which indicates if the game was joined correctly.

#### **2.1.4 AvailableMagicians**

A list of the magicians that haven't already been chosen.

#### **2.1.5 Data**

An object which contains a numerical description of the game enabling to create/update the GUI.

1. BoardData: an object containing the position of mother nature and the information of the characters used in the game.
2. List<SchoolData>: a list of objects containing a numerical description of a school and of the related player money and available cards.
3. IslandsData: an object containing a numerical description of the islands.
4. CloudsData: an object containing a numerical description of the clouds.

#### **2.1.6 PlayedCards**

A map of the cards and magicians representing the cards played from the other players in the current planning phase.



### **2.1.7 AvailableCards**

A list of the player's cards not already played in the previous planning phases.

### **2.1.8 CharacterUsed**

A boolean value which communicates if the client who is moving choose to use a character.

### **2.1.9 Character**

An integer which communicates the id of the character choose form the client who is moving.

### **2.1.10 PriceUpdate**

A Map which communicates the update of the Characters prices

### **2.1.11 MotherNaturePosition**

An integer which communicates the id of the island on which there is mother nature.

### **2.1.12 EndGame**

A boolean value which communicates if the game has ended.

### **2.1.13 Winner**

A string which communicates the magicians who won.

## **2.2 From Client to Server**

### **2.2.1 ClientChoice**

An instance of the enum ClientChoice which communicates whether the client wants to create, join or load a game.

### **2.2.2 CreateGameMESSAGE**

An object with the necessary information to create a game.

1. GameMode: instance of the enum GameMode which defines the type of game to create (standard,expert - players,teams..).
2. Name: string defining the name of the game which must be unique since it is the identifier.

### **2.2.3 GameChoice**

A string which communicates the game the client wants to join/load.

### **2.2.4 Name**

A string setting the nickname of the client.

### **2.2.5 Magician**

An instance of the enum Magician which communicates the magician chosen from the player.

### **2.2.6 UsedMagician**

An instance of the enum Magician which communicates the magician used from the player in the game before it was interrupted.

### **2.2.7 MovingStudnetsMESSAGE**

An instance of a map communicating where the player has moved a student from his entrance.

### **2.2.8 MotherNatureSteps**

An integer which communicates the number of steps for mother nature chosen by the client.

### **2.2.9 CloudChoice**

An integer which communicates the id of the cloud chosen by the client.

### **2.2.10 UsingCharacter**

A boolean value which communicates if the client wants to use a character during the action phase.

### **2.2.11 Character**

An integer which communicates the id of the character the client wants to use.