Eryantis Protocol Documentation

Lorenzo Forgia, Luigia Falasca, Federico Andrea Luigi Ferreri

Gruppo 13

# Messages

## Connect

This message is sent from the client to the server to connect clients to server.

### Arguments

* URL: URL of the game.
* IP Address: IP of the game;

### Possible Responses

* Accept(): Server accepts connection.

## Login

This message is sent from the client to the server to create a player session.

### Arguments

* Nickname: Nickname of the player.

### Possible Responses

* LoginSuccess:Nickname unique.
* LoginFailure:Nickname already taken.

## GameUpdate

This message is sent from the server to the client to update the CLI/GUI.

### Arguments

* General Board;
* Player;

### Possible Responses

• Acknowledgment: the message has been received;

## NewGame

This message is sent from the client to the server when the player wants to

start a new game with another player.

### Arguments

* NumberofPlayer.
* modEXP:boolean.

### Possible Responses

* MessageReceived: the message has been received;

## MessageRecived

This message is sent from the server to the client when NewGame message has been acknowledged.

### Arguments

This message has no arguments.

Possible Responses

This message has no responses.

JoinGame

This message is sent from the server to the client to notify the client to join a game.

Arguments

* numberOfPlayer.
* modEXP: boolean;

Possible Responses

* JoinGame from the client.

## GameStarted

This message is sent from the server to the clients to notify the start of the game.

### Arguments

* Nickname: of the player.
* Id Player: id of the player.
* General Board.

### Possible Responses

* Acknowledgment: the message has been received;

## ItsYourTurn

This message is sent from the server to the clients to notify turn of the player.

### Arguments

This message has no arguments.

### Possible Responses

* Acknowledgment: the message has been received;

## PlayAssCard

This message is sent from the client to the server to notify the chosen Assistantcard.

### Arguments

* AssistantCard:chosen assistantcard of the player.

### Possible Responses

* GameUpdate: the message has been received;

## NotifyLastRound

This message is sent from the server to the client to notify the last round.

### Arguments

This message has not arguments.

### Possible Responses

* Acknowledgment: the message has been received.

## AskIfCharacterCard

This message is sent from the server to the client.

### Arguments

* This message has no argument.

### Possible Responses

* ReturnIfCharacterCard(boolean): reply if player wants to use a character card or not.

## ReturnIfCharacterCard

This message is sent from the client to the server.

### Arguments

* Boolean: true or false, if the player wants to use the character card.

### Possible Responses

* This message has no responses.

## AskForCharacterCard

This message is sent from the server to the client.

### Arguments

* ArrayList<CharacterCard>: player’s character card deck.

### Possible Responses

* ReturnCharacterCard(CharacterCard): Character card chosen by the player.

## ReturnCharacterCard

This message is sent from the client to the server.

### Arguments

* CharacterCard: Character card chosen by the player.

### Possible Responses

* This message has no responses.

## AskForStudentAndLocation

This message is sent from the server to the client.

### Arguments

* This message has no argument.

### Possible Responses

* GetStudentAndLocation(Color, Location): students and locations chosen by the player.

## GetStudentAndLocation

This message is sent from the client to the server.

### Arguments

* Color: chosen student.
* Location: chosen location (island tiles or dining room) where the player wants to put the student.

### Possible Responses

* RightChoice(): the choice is accepted.
* IllegalChoice(): the choice is not accepted.

## AskStepMotherNature

This message is sent from the server to the client when the player has to choice the steps of Mother Nature.

### Arguments

This message has no arguments.

### Possible Responses

* NumberOfStep(number).

## NumberOfStep

This message is sent from the client to the server when the player decides the steps of Mother Nature.

### Arguments

* Number: indicates the number of steps of Mother Nature.

### Possible Responses

* RightChoice(): the number is allowed.
* IllegaleChoice(): the number is not allowed.

## EndGame

This message is sent from the server to the client when the game ended because all the towers have been placed or because there are three or less groups of Islands.

### Arguments

This message has no arguments.

### Possible Responses

This message has no responses.

## ShowWhoWins

This message is sent from the server to the client to show who wins the game when the game ends.

### Arguments

* Player: it’s the winner.

### Possible Responses

This message has no responses.

## RightChoice

This message is sent from the server to the client when the player makes an action that is allowed.

### Arguments

This message has no arguments.

### Possible Responses

This message has no responses.

## IllegaleChoice

This message is sent from the server to the client when the player makes an action that is not allowed.

### Arguments

This message has no arguments.

### Possible Responses

This message has no responses.

## AskForChoosenCloud

This message is sent from the server to the client when the player has to pick a cloud .

### Arguments

### List Cloud: it represents the list of clouds on the general board that are available.

### Possible Responses

* ChooseCloud.

## ChooseCloud

This message is sent from the client to the server when the player picks the cloud.

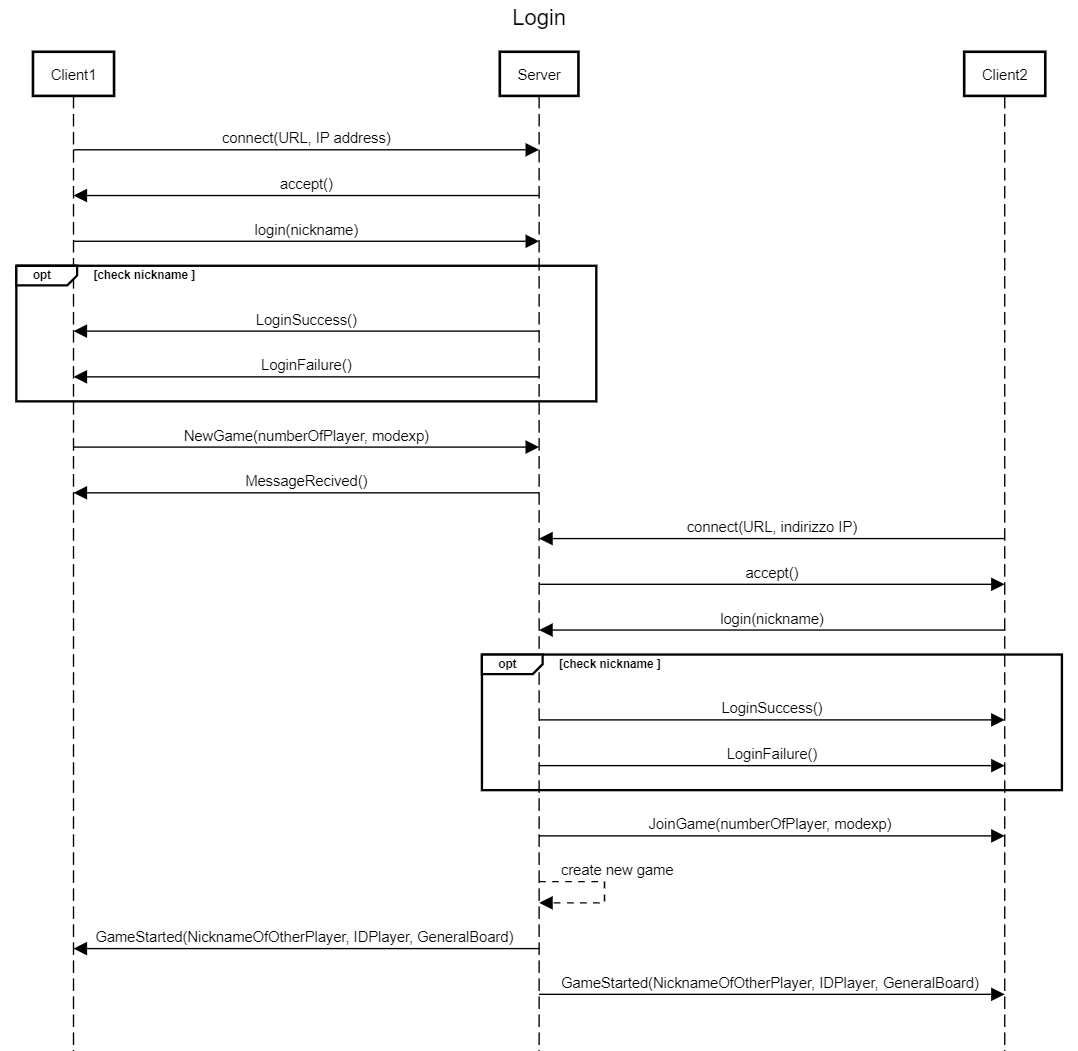
### Arguments

### Cloud: it represents the cloud chosen by the player.

### Possible Responses

This message has no responses.

# Login



Connect message is sent by the Client1 to the server which responses with Accept message.

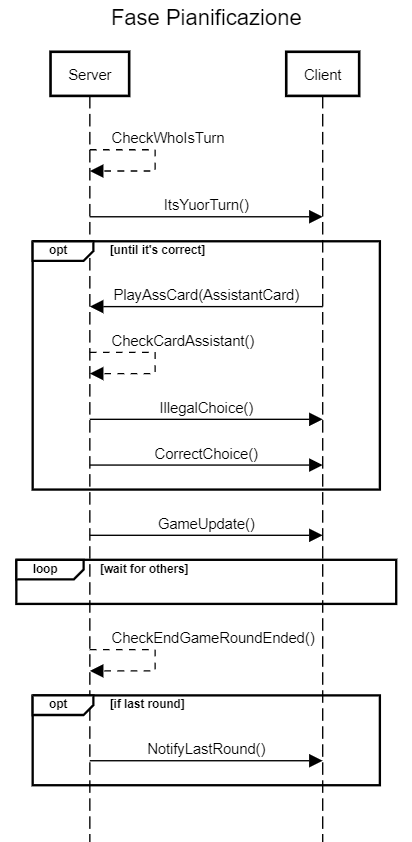
Then the Login message is sent with the nickname inserted by the client1. The server responds either with LoginSuccess or LoginFailure. If the server responds with LoginSuccess, the connection is ready to accept game-related messages.

Then Client1 sends NewGame message to server to create a new game.

Also the Client2 sends the same messages to server to Login but Client2 receives the JoinGame message to join the game created before.

After all players joined the game,server notify all clients the start of the game with GameStarted message.

# Planning Phase



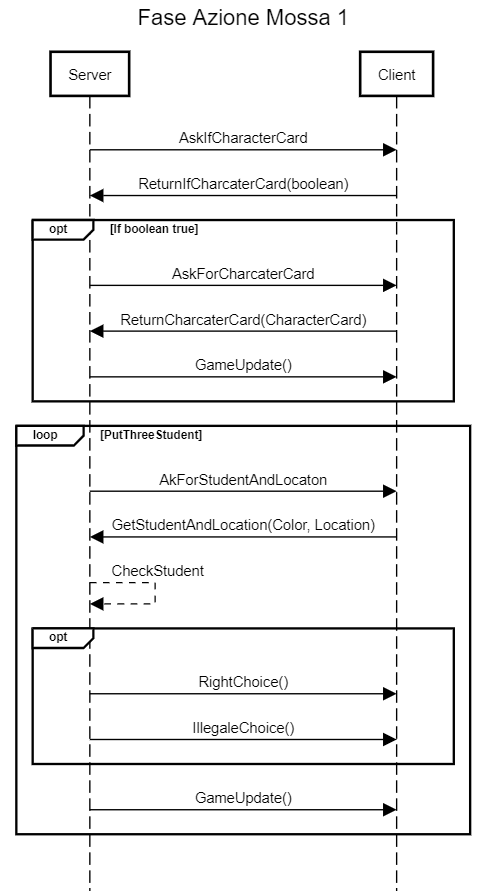
Server checks which player must play his turn ,then it sends the ItsYourTurn message to the client to notify his turn,so the Client sends the chosen AssistantCard to server which updates the CLI/GUI.

Server sends the previous messages to all clients when it’s their turn.

After the Server checks if Game must finish.

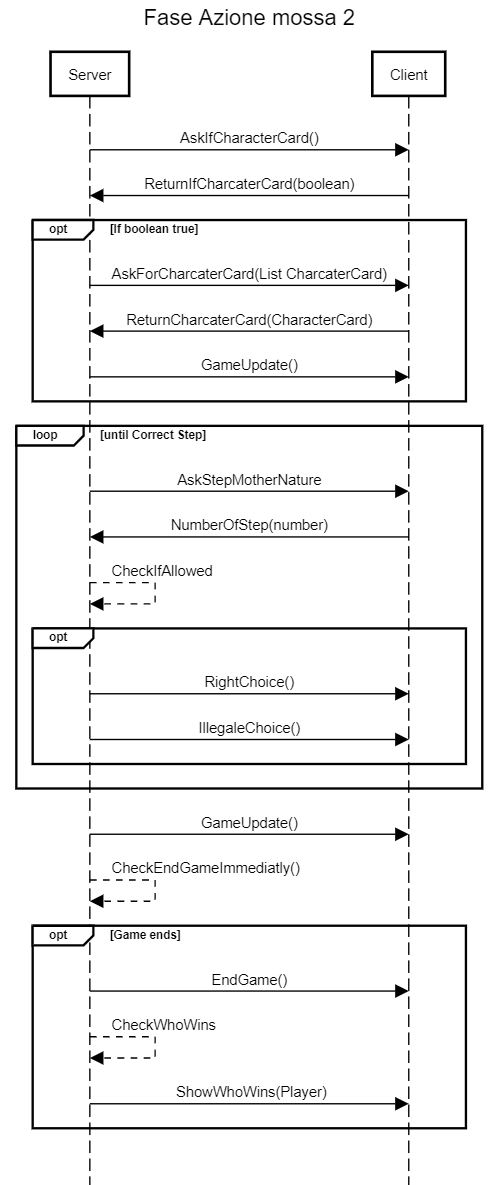
If it’s the last Round the server notifies all clients with NotifyLastRound message.

# Action Phase 1



The server sends the message AskIfCharacterCard. The client responds if wants to use it or not throw the message ReturnIfCharacterCard which contains a boolean. If the client responds true, the server sends AskForCharacterCard message. The client chooses the character card and sends it to the server throw ReturnCharacterCard message then the GameUpdate message is sent by the server. At this point for three times the server sends AskForStudentAndLocation message to the client who chooses a student and the location which could be an island tiles or the dining room and sends it to server throw GetStudentAndLocation message. Then the server checks if the player can make that movement and if the choice is possible, it sends RightChoice message to the client, if it’s not possible it sends IllegalChoice message and so the server sends a new AskForStudentAndLocation message to the client. Then the GameUpdate message is sent.

# Action Phase 2



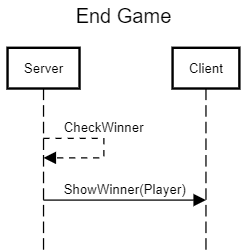
The server sends the message AskIfCharacterCard. The client responds if wants to use it or not throw the message ReturnIfCharacterCard which contains a boolean. If the client responds true, the server sends AskForCharacterCard message. The client chooses the character card and sends it to the server throw ReturnCharacterCard message then the GameUpdate message is sent by the server. At this point the server sends the message AskStepMotherNature. The client responds sending the message NumberOfStep, in which there is the steps that Mother Nature has to do. The server controls if the steps are allowed: if they are allowed it sends the message RightChoice, else it sends the message IllegaleChoice and it asks again for the steps, until the choice is correct. Then the server sends the message GameUpdate. At this stage the game can end because all the tower are placed or because there are three or less than three groups of Island. So the server will check if the game is in this condition and if it is, the server notifies the client with the message EndGame. At the end the server shows who win through the message ShoWhoWins.

# Action Phase 3

# 

The server sends the message AskForChoosenCloud in which asks the client to pick a cloud. Then the client picks the cloud and sends it through the message ChooseCloud. At the end the server sends the message GameUpdate.

# End Game



The server sends the message ShowWinner to the client to notify who won the game.