Eryantis Protocol Documentation

Elis Kina, Davide Osimo, Alessandro Fusè

Gruppo 38

Messages

LoginRequest(nickname)

Client sends a LoginRequest when he wants to create a new match.

Arguments

- nickname: client specifies his player nickname;

Possible Responses

- LoginResult(nicknameAccepted, connectionSuccesfull);
- GenericMessage(message);
- DisconnectionMessage(nicknameDisconnected, messageStr)

LoginResult(nicknameAccepted, connectionSuccesfull)

Server sends a LoginResult with the result of the login.

Arguments

- nicknameAccepted: boolean;
- connectionSuccesfull: boolean;

Possible Responses

- LoginRequest(nickname).

ModeMessage(nickname, mode)

Client chooses Normal Mode or Expert Mode of the match.

Arguments

- nickname: player nickname;
- mode: string (n/e).

PlayerNumberRequest()

Server asks the first client connected the number of players of this match.

Possible Responses

- PlayerNumberResult(nickname, playerNumber).

PlayerNumberResult(nickname, playerNumber)

Client sends a PlayerNumberResult when he chooses the number of players of this match.

Arguments

- nickname: player nickname;
- playerNumber: number of players.

BoardMessage(nickname, game)

Server sends BoardMessage each time the model is updated.

Arguments

- nickname: server nickname;
- game: the game instance updated.

AssistantCardRequest(nickname, deck)

Server asks the client to choose an Assistant card.

Arguments

- nickname: server nickname;
- deck: list of player's Assistant cards left.

Possible Responses

- AssistanCardResult(nickname, card);
- GenericMessage(message).

AssistantCardResult(nickname, card)

Client sends a AssistantCardResult when he has decided the Assistant card to play.

Arguments

- nickname: player nickname;
- card: is an integer from 1 to 10 that represents the value of the card chosen.

MoveMessage(nickname, studentColor, numIsland)

Client sends MoveMessage during his action phase each time he moves a student from his entrance.

Arguments

- nickname: player nickname;
- studentColor: player specifies which student he wants to move choosing a color;
- numIsland: if the player chose to move the student to an island, he has to specify its number, otherwise numIsland will be set to 0 if the student is going to the Dining Room.

Possible Responses

- GenericMessage(message).

MotherNatureRequest(nickname, maxMoves)

Server sends this request when a player has to move Mother Nature

Arguments

- nickname: server nickname.
- maxMoves: max number of steps that MotherNature can make.

Possible Responses

- MoveMotherNatureResult(nickname, numMoves).

MotherNatureResult(nickname, numMoves)

Client sends MotherNatureResult when he chose the number of steps.

Arguments

- nickname: player nickname.
- numMoves: number of steps that MotherNature is going to make.

Possible Responses

- GenericMessage(message);

- MoveMotherNatureRequest(nickname, maxMoves).

CharacterCardDescriptionRequest(nickname, askInterrupted)

Client sends CharacterCardDescriptionRequest when he wants to read the character card effects.

Arguments

- nickname: player nickname;
- askInterrupted:

Possible Responses

- CharacterCardDescriptionResult(nickname, text).

CharacterCardDescriptionResult(nickname, text)

Server sends the three Character Card effects chosen for the current match.

Arguments

- nickname: server nickname;
- text: contains the three card descriptions.

Possible Responses

- CharacterCardMessage(nickname, card, studentColor, numIsland).

CharacterCardMessage(nickname, card, studentColor, numIsland)

Client sends CharacterCardMessage when he wants to apply one of the three effects available.

Arguments

- nickname: player nickname;
- card: card name;
- studentColor: this argument will be the color of the student chosen by the client if the card needs to know a student to apply his effect, otherwise it will be null;
- numIsland: this argument will be the number of the island chosen by the client if the card needs to know an island to apply his effect, otherwise it will be null.

Possible Responses

- GenericMessage(message).

CloudMessage(nickname, numCloud)

Client chooses from which cloud he's getting the students.

Arguments

- nickname: player nickname;
- numCloud: number of the cloud chosen.

Possible Responses

- GenericMessage(message).

EndMessage(nickname, quit)

Client types "quit" to leave the game.

Arguments

- nickname: player nickname;
- quit: string.

Possible Responses

- DisconnectionMessage(nicknameDisconnected, messageStr).

GenericMessage(message)

This message is sent by the server when he has to:

- ask the client one of the following questions:
    - Do you want to play in Normal or Expert mode? [n/e];
    - Do you want to move a student to your plank or island? [p/i];
    - Which cloud do you choose? Insert the cloud number;
    - Type "quit" to leave the game.
- communicate an invalid client choice;
- communicate anything else.

Arguments

- message: is a generic string

Possible Responses

- ModeMessage(nickname, mode);
- MoveMessage(nickname, studentColor, numIsland);
- CloudMessage(nickname, numCloud);
- EndMessage(nickname, quit).

PingMessage()

Each client sends this message to ping the server.

Possible Responses

- ErrorMessage(nickname, error).

DisconnectionMessage(nicknameDisconnected, messageStr)

Server sends DisconnectionMessage to the client that must be disconnected.

Arguments

- nicknameDisconnected: nickname of the player that is going to be disconnected;
- messageStr: the string “disconnected from the server”.

ErrorMessage(nickname, error)

Client sends ErrorMessage to the server when it got issues with communication.

Arguments

- nickname: is set to null;
- error: one of the following strings:
    - Connection lost with the server;
    - Could not send message;
    - Could not disconnect.

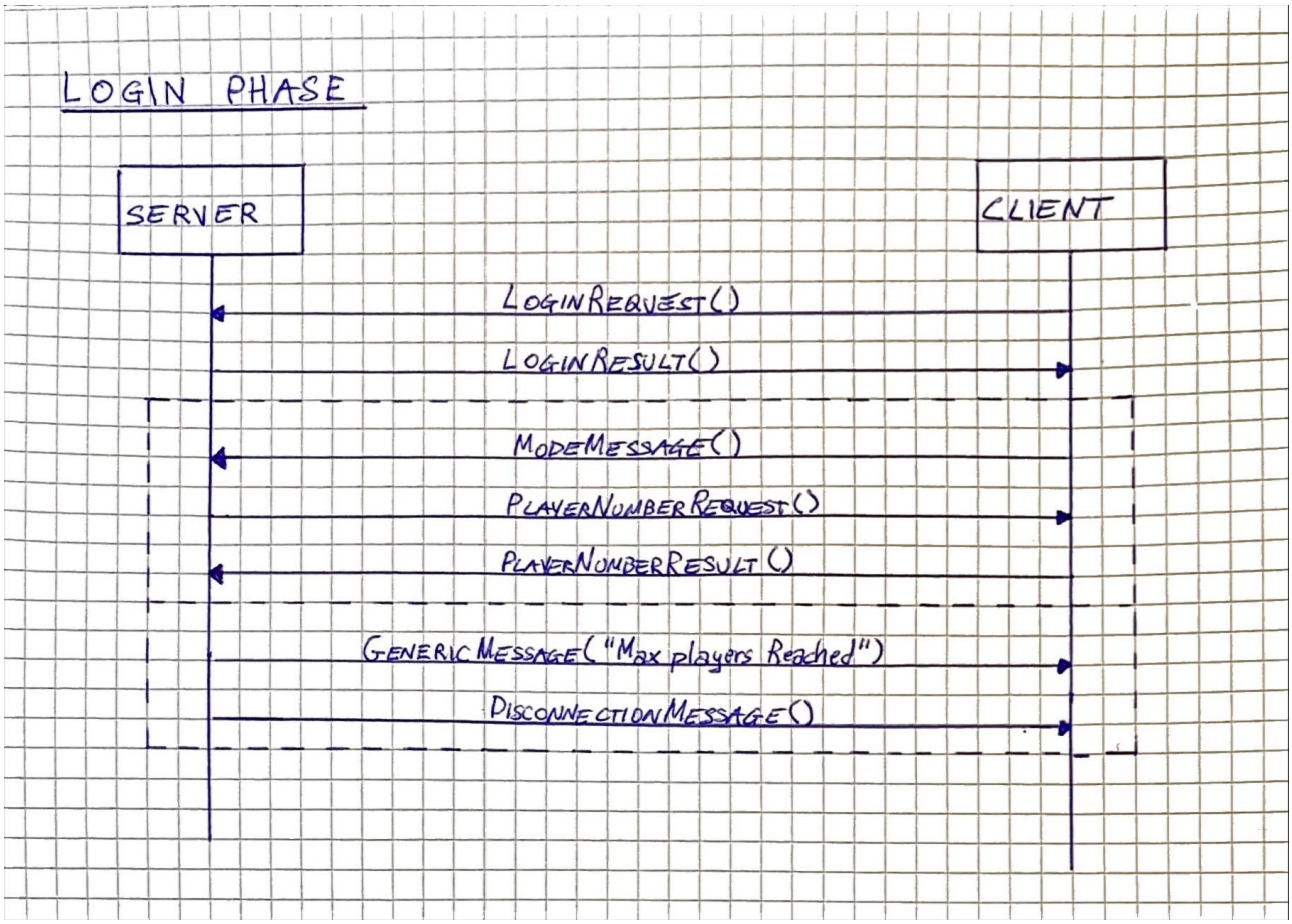
Possible Responses

- DisconnectionMessage(nicknameDisconnected, messageStr).

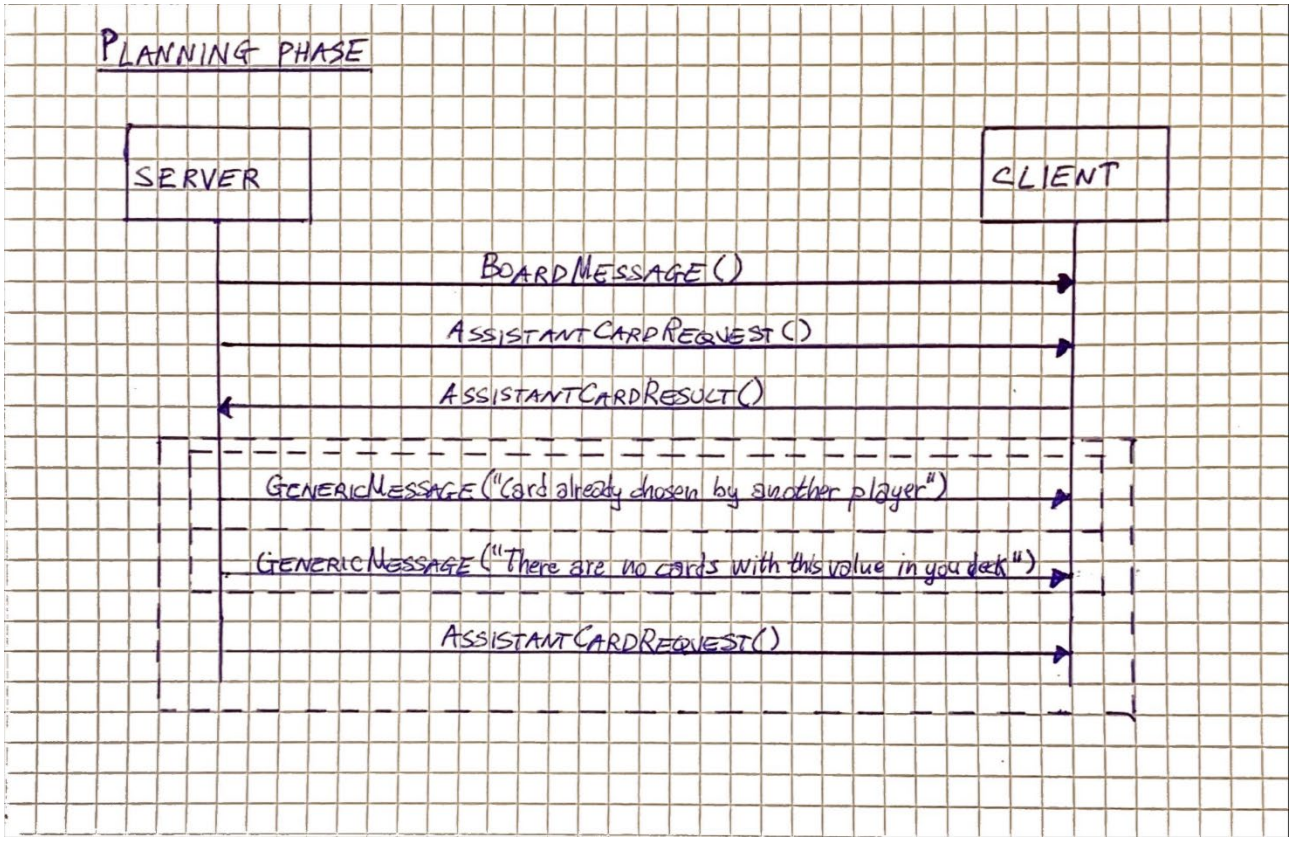
Sequence Diagrams

In the following diagrams we haven't inserted some GenericMessage(message) due to readability of the phases.

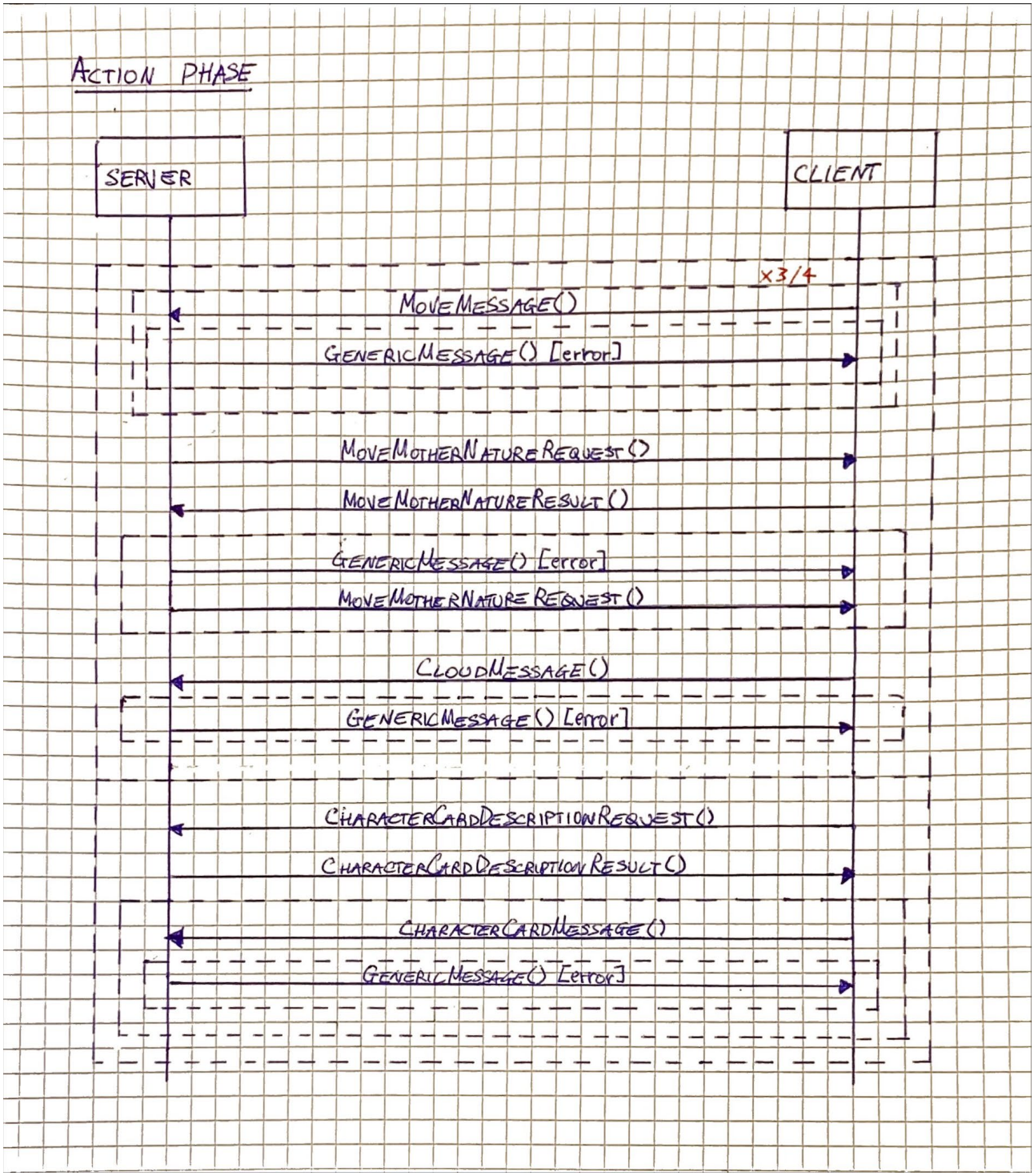
Login phase



Planning phase



Action phase



Endgame phase