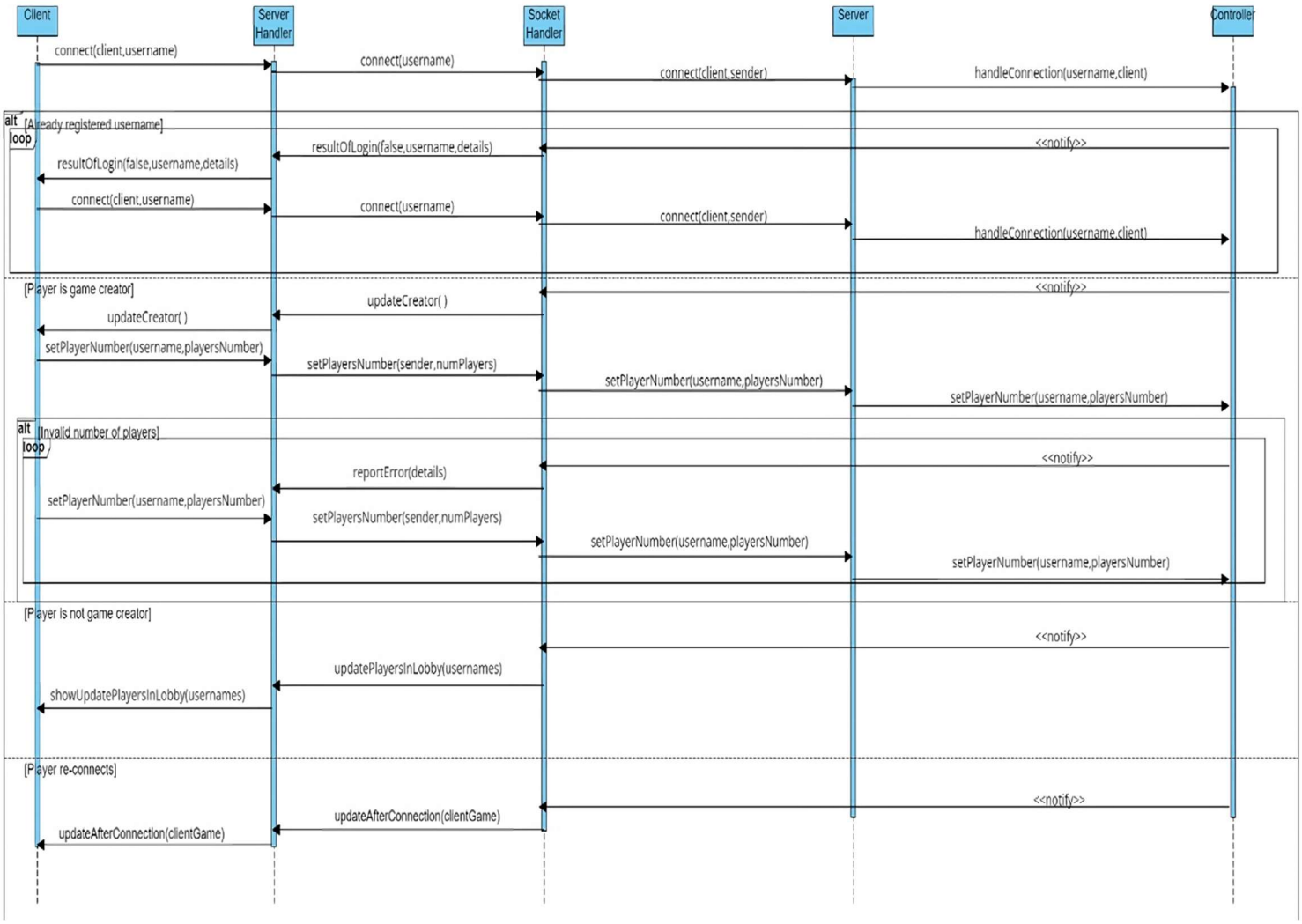
Communication protocol

The objective of his document is to establish a protocol for the interaction between client and server. RMI and Socket network protocol were implemented and messages have been written using the Json format, the phases were separated as shown below:

- Login Phase

The login can be carried out by:

- The first player, who will be considered as the creator of the game, and is allowed to enter the number of players in the game, a reportError() is sent if the number does not comply with the preset limits (min 2- max 4 players).
- A player who is not the first player, who will be added to the lobby
- A player who is reconnecting after a disconnection, who will directly enter the game.
- A player who wants to enter the game using a name that is already present in the game, to which a resultOfLogin with false accepted parameter will be sent.

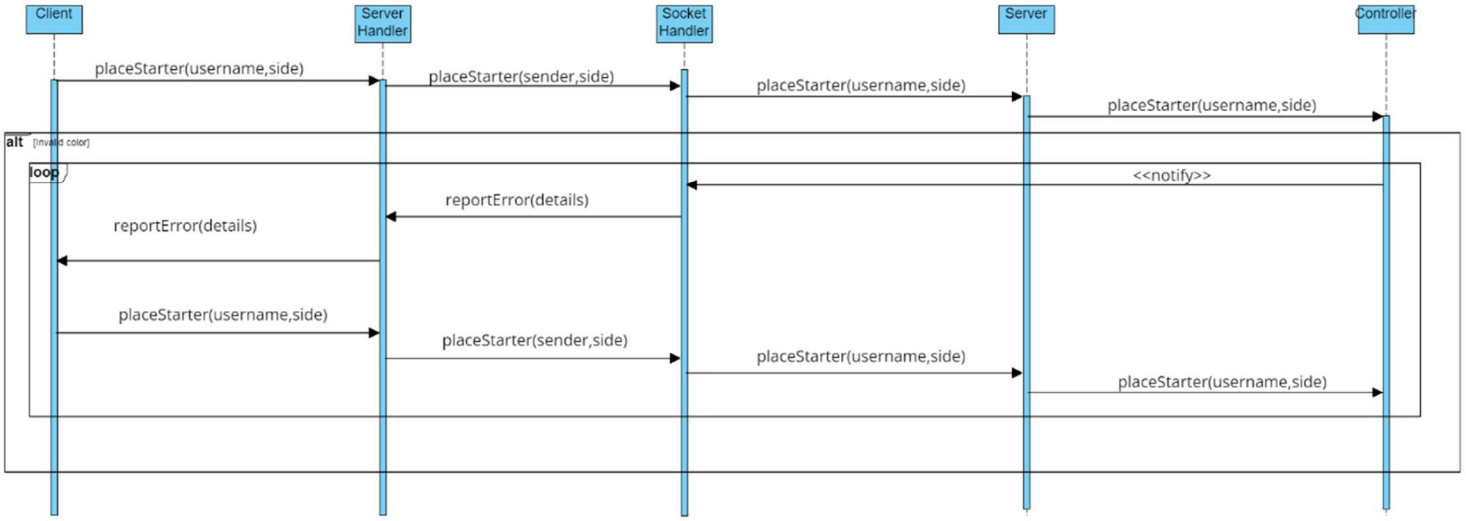


- Set Up Phase

In this phase, the game is configured, and the next activities are carried out: placeStarter, chooseColor and placeObjectiveCard.

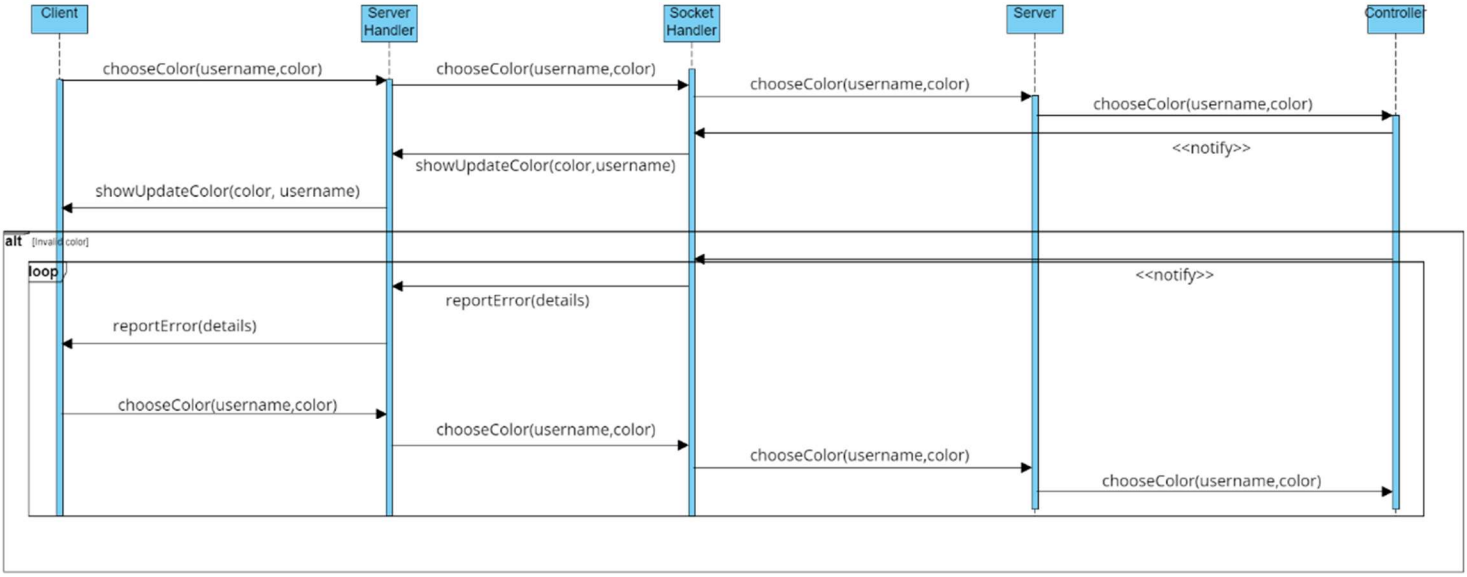
Place Starter

In this phase, the client sends the username and the side of the starter card, once the card has been positioned, the server sends a reportError() if the this action isn't available for the player.



Choose Color

The client sends to the server its username and the color that it has chosen (Color), later, when the server has checked that the chosen color is available, it assigns the color, if not, the server sends a reportError().



Place Objective Card

In this phase, the client sends the username and the index of the chosenObjective, once the card has been positioned, the server sends a hit message, the colors to choose and the username of the person who has chosen the color, if the card is not positioned correctly, the server sends an error message.

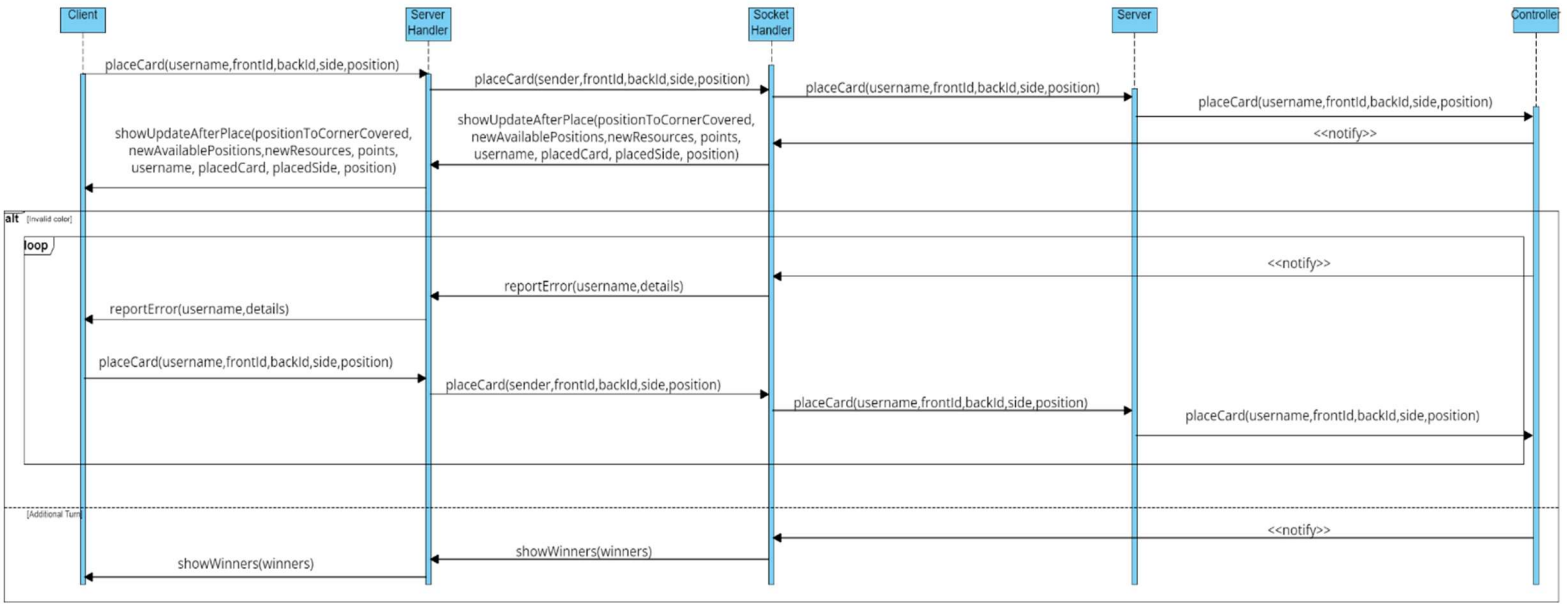
- placeCard

In this phase, we can differentiate two possible actions, which are: the placement of the card in a regular turn and the placement of the card when the Additional Turn has arrived.

In both cases, the client sends to the server the username, frontId, backId, side and position in the playground of the card.

If the card has been correctly positioned, in the regular turn, the server sends the updated playground, resources, available positions, covered corners, placed card, placed side, client's username and the points earned in the placement.

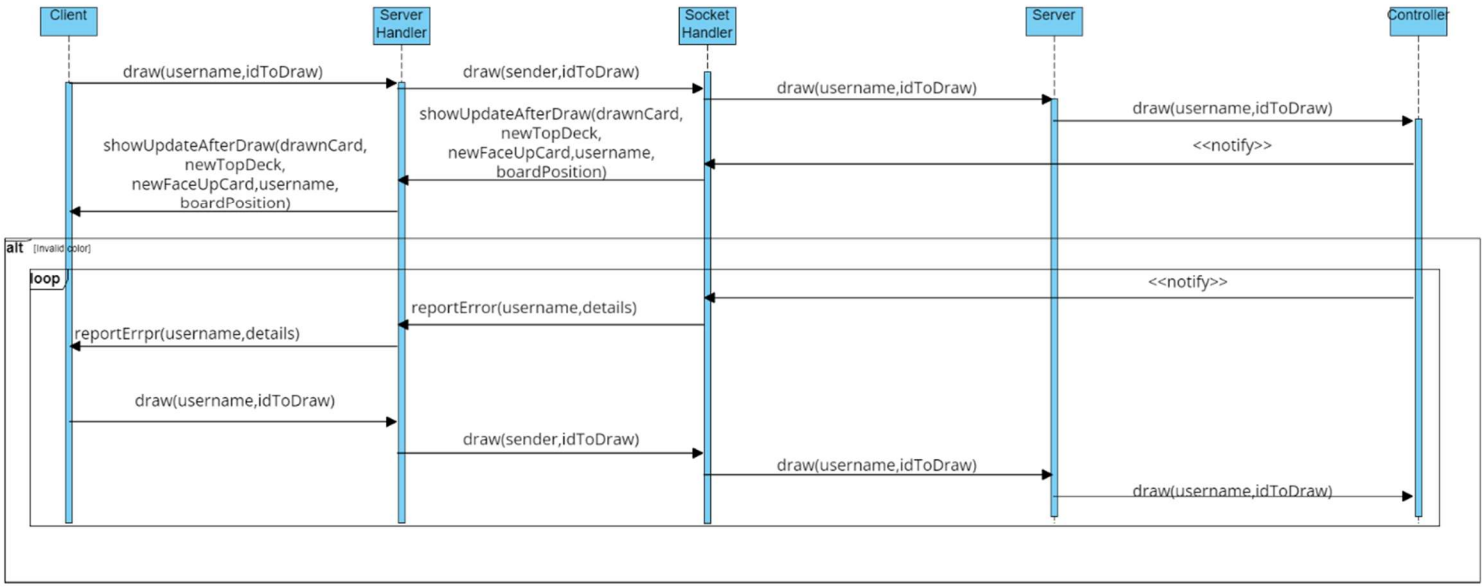
In the additional turn, if the card is positioned correctly, the server sends the winners of the game.



- draw

Starts with the client sending an integer represented by the argument idToDraw and its username. This field (idTodraw) can take six values related to the position of the card selected. The first four values (1, 2, 3, 4) indicate the position on the board of the selected face-up card, while the other two values (5, 6) represent which deck to draw from, either the resource deck or the golden deck.

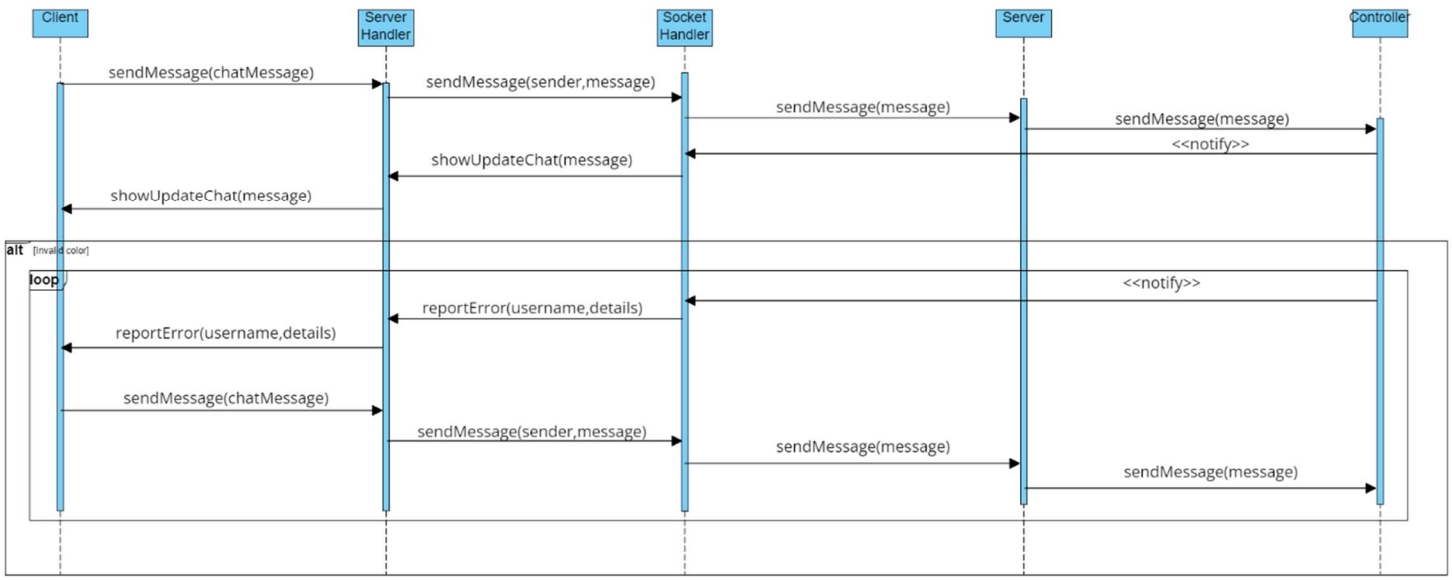
If the first client request is successful, the server responses updating the game after the draw. Otherwise, it sends a reportError(username,details).



- Chat

During the whole match, since it starts, until it ends, users can send messages to other players.

Every message is represented with a message object, containing message details (sender, receiver and content). After the message reception, the server updates the chat if there are no errors, otherwise it sends a reportError().



Message description

The messages sent between client and server are NetworkMessages, each of which has as parameters: the type and the sender. The different types of messages are listed below with a brief description, parameters and responses.

Note that: the type is an enumeration of messages and in the following charts, they have been categorized by sender.

From Client:

Type	Parameters	Parameters Description	Response
UPDATE_CREATOR	No parameters		* No response
UPDATE_AFTER_LOBBY_CRASH	No parameters		* No response
UPDATE_AFTER_CONNECTION	game	game: representation of the game	* No response
SHOW_UPDATE_PLAYERS_IN_LOBBY	usernames	usernames: usernames of the players	* No response
SHOW_UPDATE_PLAYER_STATUS	isConnected, username	isConnected: player status, username: the username of the player	* No response
SHOW_UPDATE_COLOR	username, colorSelected	Username: the username of the player, colorSelected: the color	* No response
SHOW_UPDATE_OBJECTIVE_CARD	chosenObjective, username	chosenObjective: the objective card, username: the username of the player	* No response

SHOW_UPDATE _AFTER_PLACE	positionToCornerCovered, newAvailablePosition, newResources, points, username, placedCard, placedSide, position	positionToCornerCovered: new covered corners, newAvailablePositions: new available tiles, newResources: new resources, points: updated points, username: the username of the player, placedCard: the placed card, placedSide: the chosen side, position: the position in the playground	* No response
SHOW_UPDATE_AFTER_DRAW	drawnCard, newTopDeck, newFaceUpCard, username, boardPosition	drawnCard: the drawn card, newTopDeck: the new top back deck card, newFaceUpCard: the new face up card, boardPosition: the position from which the card was drawn	* No response
SHOW_UPDATE_CHAT	message	message: the new message	* No response
SHOW_UPDATE_CURRENT_PLAYER	currentPlayerIdx, currentPhase	currentPlayerIdx: current player index, currentPhase: the currentphase	* No response
SHOW_UPDATE_SUSPENDED_GAME	No parameters		* No response
SHOW_WINNERS	winners	winners: game winners	* No response
ERROR	details	details: error details	* No response

From Server:

Type	Parameters	Parameters Description	Response
CONNECT	sender	sender: the username of the player	* RESULT_OF_LOGIN
PLACE_STARTER	sender, side	sender: the username of the player, side: the side of the starter card	* ERROR, if the an error occurs during the placement * No response, if the card is placed correctly
CHOOSE_COLOR	sender, color	sender: the username of the player, color: the color	* ERROR, if an error occurs * No response, if the card has been correctly assigned
PLACE_OBJECTIVE	sender, chosenObjective	sender: the username of the player, chosenObjective: the index of the chosen objective card	* ERROR, if the index isn't valid or the playerAction/gamePhase isn't valid * No response, if the chosen objective card has been correctly placed
PLACE_CARD	sender, frontId, backId, side, position	sender: the username of the player, side: the chosen side, backId: the back identification, frontId: the front identification, position: the selected position	* ERROR, if the card isn't valid , there aren't enough resources to place the card, the game is suspended or the playerAction/gamePhase isn't valid * No response, if the card has been placed correctly
DRAW	sender, idToDraw	sender: the username of the player, idToDraw: the id representing the deck or any of the face up cards positions	* ERROR, if the identifier isn't valid, the deck is empty, the face up card isn't valid or the playerAction/gamePhase isn't valid * No response, if the card has been drawn correctly

SEND_CHAT_MESSAGE	sender, message	sender: the username of the player, message: the new message	* ERROR, if the author doesn't match the sender or the recipient isn't valid * No response, if the message has been correctly sent
SET_PLAYER_NUMBER	sender, numPlayers	sender: the username of the player, numPlayers: the number of player of the game	* ERROR, if the number of players isn't valid * No response, if the number of players has been correctly set
DISCONNECT	sender	sender: the username of the player to disconnect	* "Request of disconnection from unknown connectionId", if the sender's handler is null * "User left the server", if the sender has been correctly disconnected
RESULT_OF_LOGIN	accepted, username, details	accepted: boolean indicating whether the player is accepted or not, username: the username of the player, details: error details	* "I'm already connected", if the username is already connected * INVALID_LOGIN, if the username isn't correct *No response if the player has been accepted
FULL_LOBBY	No parameters		* No response
EXCEEDING_PLAYER	No parameters		* No response

From Client and Server

Type	Parameters	Parameters Description	Response
HEARTBEAT	sender, id	sender: the sender of the message, id: message identification	* Unknown user error, if the sender is null * No response, if the heartbeat (ping) has been sent correctly