Eryantis Protocol Documentation

Alessio Buda, Dario Mazzola, Gabriele Munafo'

Group 21

1. Messages

1.1 NewGame

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

Arguments

- This messsage has no arguments.

Possible responses

This messsage has no possible responses.

1.2 Nack

This message is sent from the server to the client when a generic message has not been acknowledged. If the client receives this message, it notifies the player of the error and asks him/her to make his/her choice again.

Arguments

- TypeOfError: specifies the possible errors

Possible responses

This messsage has no possible responses.

1.3 SelectNumPlayers

This message is sent from the server to the client to communicate that the player has to choose the number of players for this match

Arguments

- This messsage has no arguments.

Possible responses

- ChosenNumPlayers

1.4 ChosenNumPlayers

This message is sent from the client to the server, when the client represents the first player connected, to communicate the number of players for this game.

Arguments

- NumPlayers: the number of players for this game.

Possible responses

- Nack: when the chosen numPlayer is not valid.

1.5 SelectExpertMode

This message is sent from the server to the client to communicate that the player has to choose whether the match will be played in expert mode.

Arguments

- This messsage has no arguments.

Possible responses

- ChosenExpertMode

1.6 ChosenExpertMode

This message is sent from the client to the server, when the client represents the first player connected, to communicate whether the current game will be played in expert mode.

Arguments

- ExpertMode: true if the game mode chosen is "expert mode".

Possible responses

- This messsage has no possible responses.

1.7 SelectNickname

This message is sent by the server to the client to ask the player to choose the nickname again because the one he had chosen is not available.

Arguments

- This messsage has no arguments.

Possible responses

- Nickname

1.8 Nickname

This message is sent from the client to the server to communicate the player's chosen nickname. Input validity must be checked by the server.

Arguments

- Nickname: the player's nickname.

Possible responses

- This messsage has no possible responses.

1.9 SelectWizard

This message is sent from the server to the client to communicate that the player has to choose the wizard for this match

Arguments

- availableWizards: list of all the available wizards.

Possible responses

- ChosenWizard: to communicate player's chosen wizard.

1.10 ChosenWizard

This message is sent from the client to the server to communicate the player's chosen wizard.

Arguments

- Wizard: the player's wizard.

Possible responses

- This messsage has no possible responses.

1.11 SelectTeam

This message is sent from the server to the client to communicate that the player has to choose the team he/she wants to play in this match

Arguments

- teamArray: an array with the nicknames of the players that are not leaders of their team.
- leaderArray an array with the nicknames of the players that are leaders of their team.

Possible responses

- ChosenTeam

1.12 ChosenTeam

This message is sent from the client to the server when there are 4 players. It is used to communicate to the server the player's chosen team and if he/she wants to be the team leader.

Arguments

- TeamChosen: the team chosen by the player identified by nickname
- IsTeamLeader: true if the player wants to be team leader, false otherwise

Possible responses

- This message has no responses.

1.13 SelectColorTower

This message is sent from the server to the client to communicate that the player has to select his/her color of the tower for this match.

Arguments

- availableColors: list of the all the available tower colors.

Possible responses

- ColorTower: the player's selected tower color.

1.14 ChosenTowerColor

This message is sent from the client to the server to communicate the player's chosen tower color.

Arguments

- chosenTowerColor: the player's tower color.

Possible responses

- This message has no responses.

1.15 SelectAssistantCard

This message is sent from the server to the client to communicate the assistant cards in current player's deck.

Arguments

- availableAssistantCards: available assistant card.

Possible responses

- chosenAssistantCard: when the player choses the assistant card to use in this round.

1.16 ChosenAssistantCard

This message is sent from the client to the server to communicate player's selected assistant card for this round. Input validity can be checked by the controller on client.

Arguments

- AssistantCard: the player's selected assisant card.

Possible response

- This message has no responses.

1.17 ActionPhase

This message is sent from the server to the client to communicate all the possible actions that the player can do.

Arguments

- availableActions: all the available actions for the player

Possible responses

- This message has no responses.

1.18 MoveStudentToIsland

This message is sent from the client to the server to communicate the student the player wants to move from his/hers dining hall to an island.

Arguments

- House: the house of the student to move.
- Island: the island where the student will be added.

Possible responses

- This message has no responses.

1.19 MoveStudentToDiningHall

This message is sent from the client to the server to communicate the student to move from the player's entrance to his/hers dining hall.

Arguments

- House: the player's student to move.

Possible responses

- This message has no responses.

1.20 WaitForOthersMoves

This message is sent from the server to the client to tell the player to wait for another player to make a choice. The choice is represented by the move parameter.

Arguments

- move: the choice the player is waiting for.

Possible responses

- This message has no responses.

1.21 MoveMother

This message is sent from the client to the server to communicate the number mother nature steps selected by the player.

Arguments

- motherMoves: the mother nature steps chosen by the player.

Possible responses

- This message has no responses.

1.22 SelectCloud

This message is sent from the server to the client to communicate the available clouds and the students on them.

Arguments

This message has no arguments.

Possible responses

- Nack: when the move has not been accepted.

1.23 ChosenCloud

This message is sent from the client to the server to communicate the player's selected cloud.

Arguments

- cloud: cloud selected by the player.

Possible responses

This message has no responses

1.26 ChosenCharacterCard

This message is sent from the client to the server to request the character cards in use in this game.

Arguments

- cardIndex: the index of the CharacterCard chosen by the player

Possible responses

- This messagge has no responses.

1.27 SendWinner

This message is sent from the server to the client to communicate the winning player.

Arguments

- Winner: winning player

Possible responses

- This message has no responses

1.28 Ping

This message is sent from the server to the client to check if it is still connected.

Arguments

This message has no argumets.

Possible responses

- This message has no responses

1.29 Pong

This message is sent from the client to the server to signal that the connection is still alive.

Arguments

This message has no arguments.

Possible responses

This message has no possible responses.

1.30 EndGameDisconnection

This message is sent from the server to the client to signal that the game has ended due to a client disconnection.

Arguments

errorCause: the message explaining why there was a disconnection

Possible responses

This message has no possible responses.

1.31 ChatMessageServerClient

This message represents a chat message to be forwarded by the server to a player.

Arguments

- message: the message to be forwarded by the server to a player

Possible responses

This message has no possible responses.

1.32 ChatMessageClientServer

This message represents a message sent from the client to the server in order to be forwarded to the teammate of the player.

Arguments

- message: the message to be forwarded by the server to a player

Possible responses

This message has no possible responses.

1.33 GameFull

This message is sent from the sever to the client when there is already the maximum number of admissible players correctly connected.

Arguments

This message has no arguments.

Possible responses

This message has no possible responses.

1.34 GoToLobby

This message is sent from the sever to the client when the server tells the client to go to the lobby as the game hasn't started yet.

Arguments

This message has no arguments.

Possible responses

This message has no possible responses.

1.35 GoToWaitingRoom

This message is sent from the sever to the client when the server tells the client to go to the waiting room as the player has finished his turn and has to wait for the others to make their moves.

Arguments

This message has no arguments.

Possible responses

This message has no possible responses.

1.36 RememberNickname

This message is sent by the server to the client to remind the client of the nickname chosen by the player following a disconnection.

Arguments

- nickname: the nickname to be saved

Possible responses

This message has no possible responses.

1.37 SelectChat

Select chat message is sent from the server to the client to ask the first player whether the member of the teams (when playing in four players) can communicate with each other.

Arguments

This message has no arguments.

Possible responses

This message has no possible responses.

1.38 ChosenChat

ChosenChat message is sent from the client to the server to communicate whether the player wants to allow communication between team members.

Arguments

- chat: true if the chat is allowed for this match.

Possible responses

This message has no possible responses.

1.39 SelectRestoreGame

This message is sent from the server to the client to notify that there is a match saved on the server. The client has to choose whether to restore the game or not.

Arguments

This message has no arguments.

Possible responses

This message has no possible responses.

1.40 ChosenRestoreGame

This message is sent from the client to the server to communicate whether to resume the saved game or not.

Arguments

- toRestore: true if the player wants to restore

Possible responses

This message has no possible responses.

1.41 ReloadMessages

ReloadMessages message is sent from the client to the server. It asks the server to set a timer and to send, at the expiring of the timer, an ActionPhase or GoToWaitingRoom message to the server. This allows the Cli to show messages potentially received while managing the action phase or waiting room.

Arguments

This message as no arguments.

Possible responses

This message has no possible responses.

2. Update messages

This section lists all the update messages that are sent from the server to the client when the Model is changed within the MVC Pattern

2.1 UpdateCharacterCard

This message is sent from the client to the server to notify a change in a character card in use in this game.

Arguments

- characterCard: the characterCard to update

Possible responses

This messagge has no responses.

2.2 UpdateCloud

This message is sent from the server to the client to notify the change of a Cloud.

Arguments

- clouds: the updated clouds.

Possible responses

This message has no responses

2.3 UpdateGameModel

This message is sent from the server to the client to notify the change of the Game Model.

Arguments

- gameModel: the gameModel updated

Possible responses

This message has no possible responses.

2.4 UpdateDashboard

This message is sent from the server to the client to notify the change of a Dashboard.

Arguments

- dashboard: the dashboard updated

Possible responses

This message has no possible responses.

2.5 UpdateDiningHall

This message is sent from the server to the client to notify the change of a DiningHall.

Arguments

- diningHall: the diningHall updated

Possible responses

This message has no possible responses.

2.6 UpdateMotherIsland

This message is sent from the server to the client to update the mother nature position following a move.

Arguments

- motherIsland: the motherIsland position updated

Possible responses

This message has no possible responses.

2.7 UpdatePlayer

This message is sent from the server to the client to notify the change of a Player.

Arguments

- player: the player updated

Possible responses

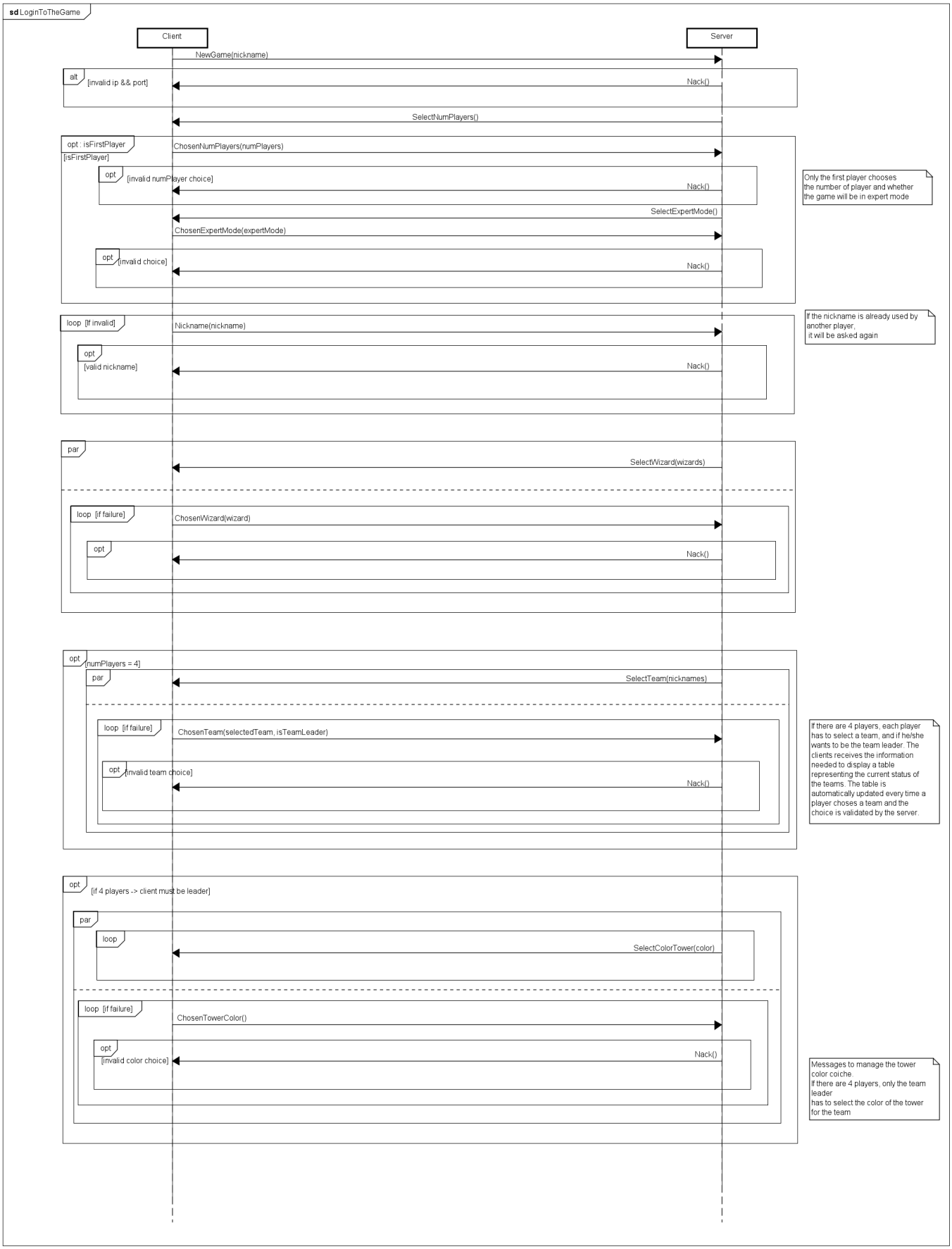
This message has no possible responses.

3. Scenarios

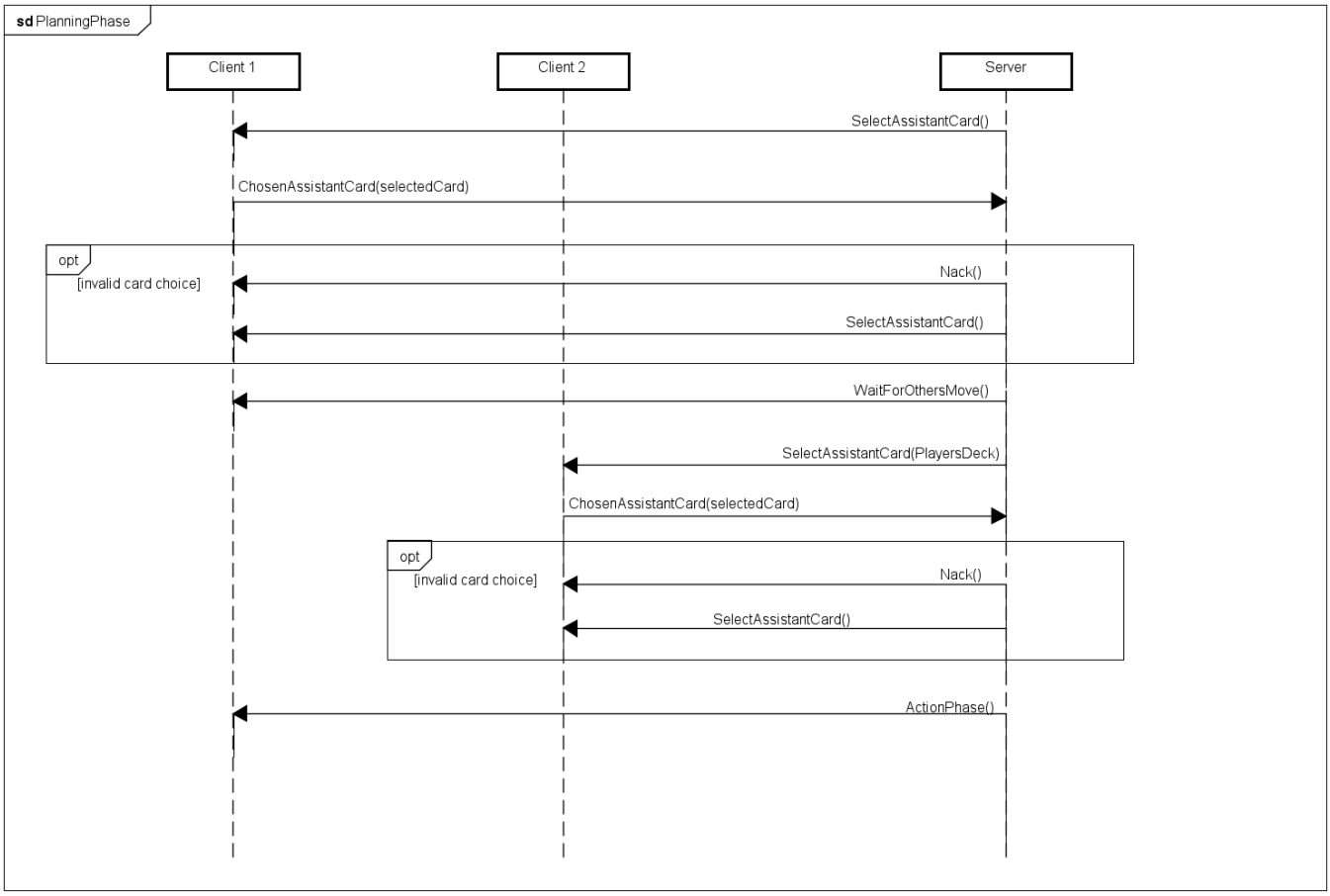
3.1 Login to the game

The connection to the game is notified by the client to the server through the NewGame message. In case the chosen nickname is already used, the nickname selection phase starts again until the name is univoque to the player. Firstly, the number of players is sent from the client to the server. Then, if the player is the first logging in, he will also comunicate which mode he wants to play, either normal or expert. Now the server sends all players the list containing all the wizards available. Every player has to select one of them and communicate it to the server, which will respond with a success or a failure, for example if the client selects

a wizard who is not present in the list given. If the wizard is not accepted, the wizard selection phase starts again until the wizard is univoque to the player. If the wizard is accepted, the server updates every client with the new list of available wizard they can choose from. If the number of players is four, that means it is a team game. Every player has to select the number of team they want to play in and whether or not they want to be the leader of that team. The server responds with either a success or a failure, for example if the team is already full or already has a leader. In the second case, the team selection phase starts again until an acceptable team and decision on the team leader is given. If it is not a team game or the player is the team leader, the server sends the list of all the tower colors. Every player has to select a tower color and the server responds with either a success or a failure, for example if the color is already taken. In the second case, the tower color selection phase starts again until every player or team has an univoque tower color.

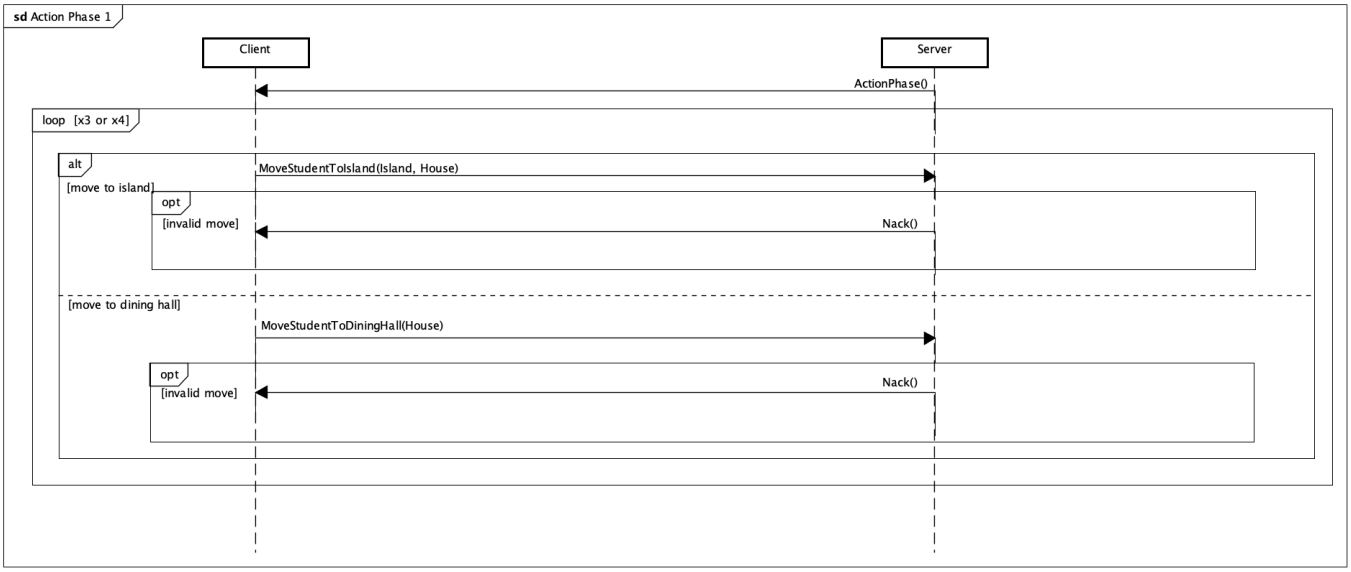


3.2 Planning phase



The sequence diagram shows the interaction between 2 clients and the server in the choice of the Assitant Cards. The choice is sequential, therefore the clients will wait their turn to use a card that respects the constraints imposed by the rules. The server sends a message to the client with the list of cards available for this turn. The same scenario is repeated for a second client and can be extended if 3 or 4 clients are connected.

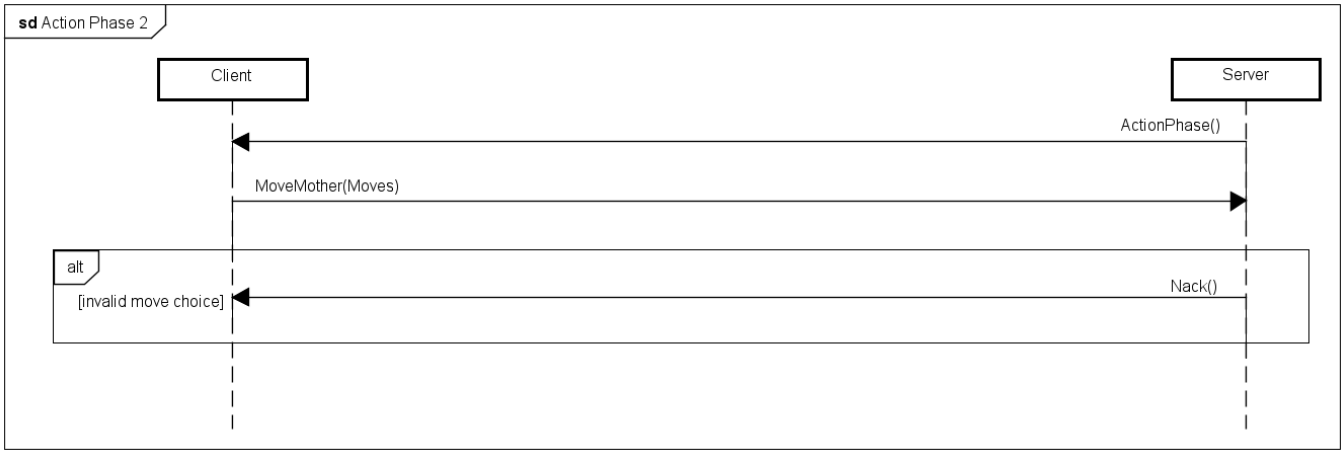
3.3 Action phase 1



The sequence diagram shows the interaction between a client and the server in the first move of the action phase. In this phase the player must move 3 or 4 students and can choose to move them in any order to an

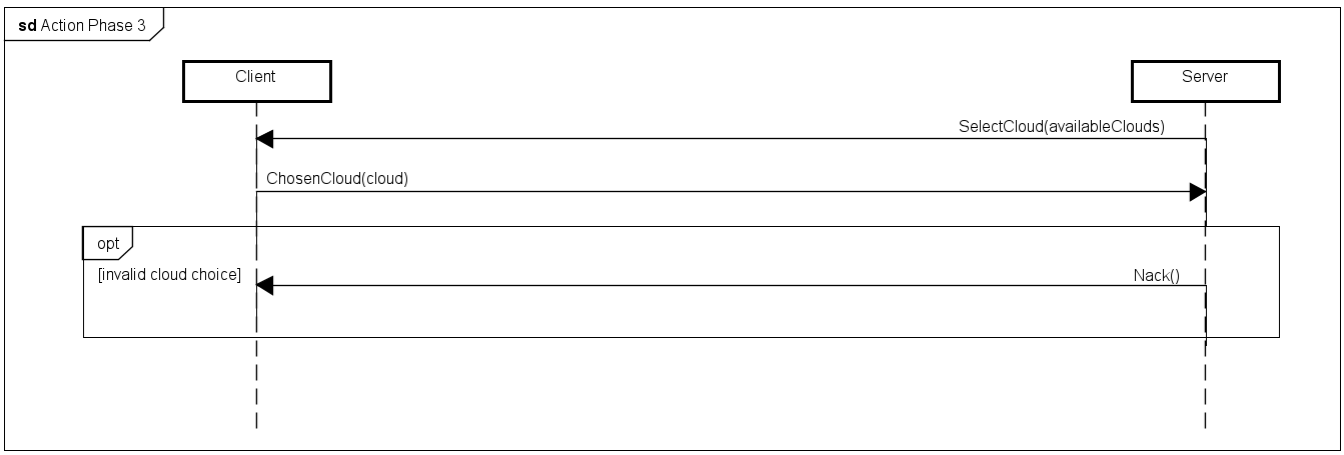
island or their DiningHall. The interaction begins with the server sending an ActionPhase message notifying the player of all possible moves at his disposal.The loop will be performed 3 or 4 times, based on the number of players. The controller on the client will send, depending on the player's choice, a `MoveStudentToIsland` or `MoveStudentToDiningHall` message. The controller can respond in both cases with failure or success messages to indicate to the client to repeat the choice or continue with the next move.

3.4 Action phase 2



This diagram shows the interaction between the client and the server when the player is chosing of how many steps to move mother nature. The server sends the client an ActionPhase message to notify to the client that is time to move mother nature. The player responds with the desired number of steps. Input validity is checked both on the client and the server. If the input is correct, the game will go on, wheras when the given input is not valid the server responds signaling a failure thus asking the client to send a new message with valid input.

3.5 Action phase 3



This diagram shows the interaction between the client and the server when the player is chosing the cloud from which to refill the entrance of his/hers dashboard. The server sends to the client all the availbale clouds and the students on them. The player choses a cloud and communicates his/hers choice to the server. Input validity is checked both on the client and the server. If the input is correct, the game will go on, wheras when the given input is not valid the server responds signaling a failure thus asking the client to send a new message with valid input.

3.6 End game phase

This diagram shows the interaction between the client and the server when the server is signaling the winning player and thus the end of the game. The player now has the choice of playing a new game or placing the game back in the box.