**Eryantis Protocol Documentation**

Ferrario, Griffanti, Masiero

Gruppo 25

# **Messages**

We decided to use JSON files to send and receive messages between client and server.

# **Acknowledgment**

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

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

### **Arguments**

This message has no arguments.

### **Possible Responses**

This message has no responses.

# **Login**

When the connection is initialized the client sends his username to the server.

### **Arguments**

Username: the username chosen by the player

### **Possible Responses**

* NicknameNotValid:
* LoginSuccess:

# **NicknameNotValid**

This message is sent by the server to the client when the Player session can’t be created because the nickname is already used.

### **Arguments**

This message has no arguments.

### **Possible responses**

* Login: a new nickname is requested.

## **LoginSuccess**

This message is sent from the server to the client when a player session is created. The username sent before is accepted.

### **Arguments**

playerID: unique ID inside the Match

//we could store it on the server side, in the clientHandler

NewMatchNeeded: 0 if a new match must be created

1 if the player can join an existing match

### **Possible responses**

* NewMatchSpecs: the client provides the details of the new match.
* Acknowledgment: the message has been received. *(if 1)*

## **NewMatchSpecs**

This message is sent from the client to the server when the server must create a new match. It contains the specifications necessary for the new match creation.

### **Arguments**

NumberOfPlayers: number of Players for the Match (2 to 4 players).

ExpertMode: 0 if not expert mode

1 if expert mode

### **Possible responses**

* NewMatchWaiting: the server creates the new match with the provided specifications.

## **NewMatchWaiting**

This message is sent from the server to the client when the game is waiting for other players.

### **Arguments**

This message has no arguments.

### **Possible responses**

* Acknowledgment: the message has been received.

## **MatchStarts**

This message is sent from the server to the client when the game is ready to start.

### **Arguments**

Match: the starting state of the match joined.

SchoolBoardState: the current state of the school board of the player.

### **Possible responses**

* Acknowledgment: the message has been received
* BagClickedByFirstClient: the first player will receive this reply.

## **BagClickedByFirstClient**

This message is sent from the client to the server if it is the first of this manche.

### **Arguments**

This message has no arguments.

### **Possible responses**

* Acknowledgment: the message has been received

## **ChosenAssistantCard**

This message is sent from the client to the server which contains the assistant card chosen by the player.

### **Arguments**

OrderValueOfTheCard: integer value that uniquely identifies the card chosen inside the deck

### **Possible responses**

* Acknowledgment: the message has been received
* AssistantNotValid: if the assistant has been already used

## **AssistantNotValid**

This message is sent from the server to the client if the assistant chosen by the player is not valid.

### **Arguments**

This message has no arguments

### **Possible responses**

* ChosenAssistantCard: it requests a new valid assistant card to the client.

## **Action1-StudentsMovementToIsland**

This message is sent from to the client to the server when the player moves one or more

students to an island

### **Arguments**

StudentsMoved: {

[ ID\_student1, ID\_island],

[ ID\_student2, ID\_island],

[ ID\_student3, ID\_island],

[ ID\_student4, ID\_island]}

### **Possible responses**

* Acknowledgment: the message has been received

## **Action1-StudentsMovementToDiningRoom**

This message is sent from to the client to the server when the player moves one or more

student to the dining room

### **Arguments**

StudentsMoved: [ID\_student1, ID\_student2, ID\_student3, ID\_student4]

### **Possible responses**

* Acknowledgment: the message has been received

## **Action2-MoveMotherNature**

This message is sent from to the client to the server when the player moves mother nature

### **Arguments**

islandChosen\_ID: ID of the island where the player chooses to move mother nature

### **Possible responses**

* Acknowledgment: the message has been received
* IslandINotValid: the player tries to move mother nature on a non-valid island

## **IslandIDNotValid**

This message is sent from to the server to the client when the player tries to move mother nature on a non-valid island

### **Arguments**

This message has no arguments.

### **Possible responses**

* Action2-MoveMotherNature: move mother nature.

## **Action3-CloudChosen**

This message is sent from to the client to the server when the player chooses a cloud to take

the students from

### **Arguments**

cloudChosen\_ID: the ID of the cloud chosen by the player

### **Possible responses**

* Acknowledgment: if there is no problem with the chosen cloud
* CloudNotValid: if the chosen cloud is not valid (it has already been chosen by another player, empty cloud)

## **CloudNotValid**

This message is sent from the server to the client when the cloud chosen by the player is not valid (the cloud has already been chosen by another player).

### **Arguments**

This message has no arguments.

### **Possible responses**

* Action3-ChooseCloud: the player selects another cloud

## **MatchEnd**

This message is sent to the client when one of the possible conditions to end the game is met.

### **Arguments**

Winner: the ID of the player that won the match

### **Possible responses**

* This message has no response.

# **Scenarios**

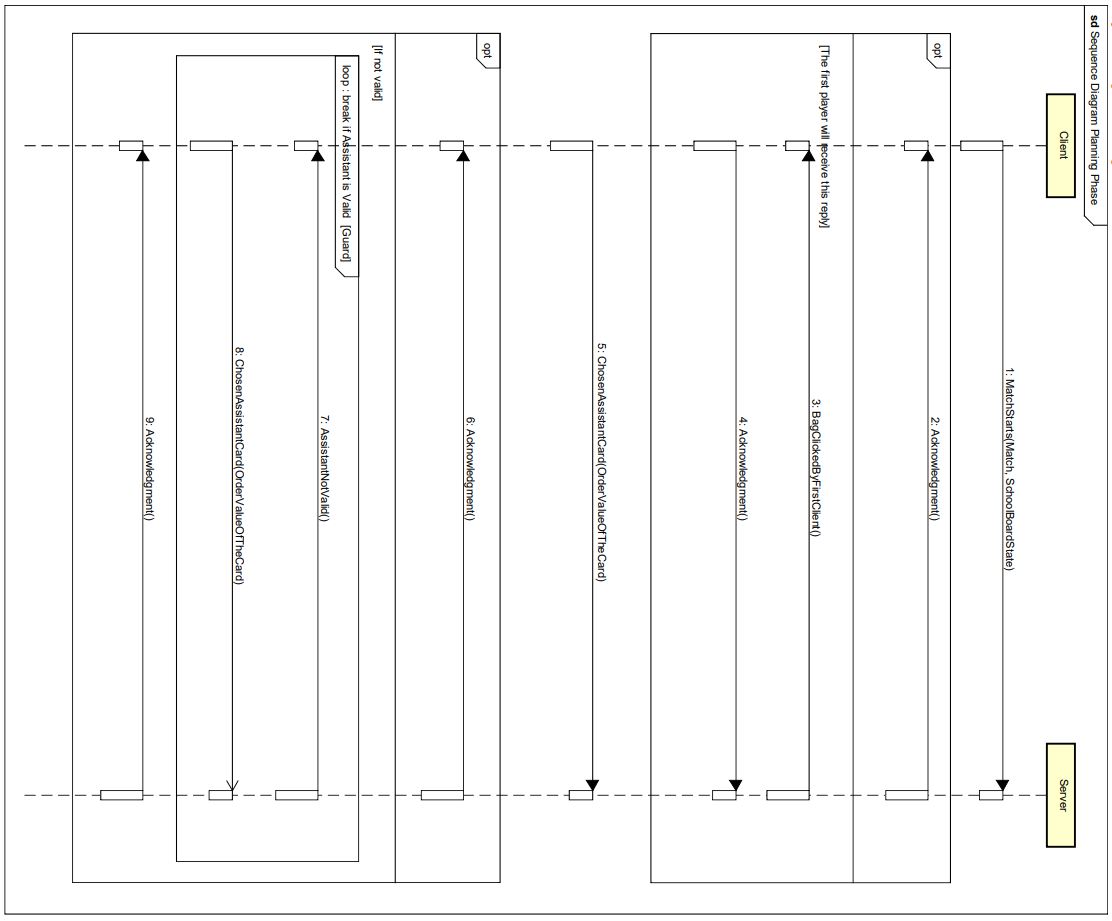
## **Sequence Diagram: Joining game**

First, the Login message is sent with the username inserted by the client. The server can respond either with LoginSuccess or NicknameNotValid.

If the server responds with NicknameNotValid, a new Login request is sent (It is placed in a loop, which ends only if the nickname is accepted).

If the server responds with LoginSuccess, the connection is ready, and an Acknowledgment message is sent. Then a message for the specifications of the game is sent by the client with a NewMatchSpecs message, containing information about the number of players and the expert mode option. The server replies with a NewMatchWaiting message, followed by an Acknowledgment message by the client.

## **Planning phase**



During the Planning phase, the match is ready to start. The client sends a MatchStarts message containing the arguments of Match and client’s school board state. The reply from the server could be an Acknowledgment message or, if the client is the first player, a BagClickedByFirstClient, to which the client replies with an Acknowledgment message.

Then the planning phase keeps on with the ChosenAssistantCard message sent from the client, which tells the server the integer order value which refers to the assistant card chosen. If the chosen assistant is not valid, the server replies with AssistantNotValid message, and so the client is obliged to send again the needed information, till the assistant card selected is correct (loop).

If the assistant card is accepted, the server sends an Acknowledgment message back to the client.

## **Action phase, move 1**

The client notifies the server when he chooses the students to move and the island to which

this movement must be done with a StudentsMovementToIsland message.

The server responds with an Acknowledgment message.

Otherwise, if the player decides to move the students to the diningroom, the client notifies the server with a StudentsMovementToDiningRoom, to which the server responds with an

Acknowledgment.

## **Action phase, move 2**

The client notifies the server when he is moving mother nature to a certain Island with a MoveMotherNature message. The server either reponds with an Acknowledgment messager

or responds with a IslandIDNotValid message, when mother nature can’t be put on a certain island, e.g. the assistant card played by the player has a motherNotherValue equal to 3 and the player tries to move mother nature 4 islands ahead.

## **Action phase, move 3**

The client notifies the server when he chooses the cloud from which to take the 3 or 4 students to put on his board’s entrance with a “CloudChosen” message.

The first cloud, chosen by the player, is not valid because the other player has already chosen it, leaving it without students.

The server responds with a “CloudNotValid” message and eventually the client nofies the server with a valid cloud-index, that is 0, in fact this cloud still has all its students.

## **End of match**

When the match ends it means that one of the conditions has been met; so the server notifies the client with a “MatchEnd” message with the ID of the winner as argument.The server does not expect to receive a response from the client, after this message the game session is terminated.