Codex:Naturalis Protocol Documentation

Wang Kevin, Sacchi Pietro, Cipolloni Valerio, Sfondrini Maria

Gruppo 47

## Introduction

To implement the communication protocol we gave the client an interface of the function he could call to interact with the server (he has a VirtualServer), and also the server has an interface of the client that he can use to reply to the request of the client (he has a VirtualView).

Also, we decided to use a class “NetworkMessage” used only to exchange information between client and server, this object contains a String to say for who or whom is from the message and a MessageType (enum) to differentiate the various type of messages exchanged during the connection.

We’re using GSON library for the serialization and deserialization of the messages over the network.

Advanced feature implemented: Multiple Lobby and Connection Resiliency (and chat).

In our project the server will have 3 main types of thread: RMIServer, SocketServer (in this case we simply called both ServerMain) and ThreadLobby; in the starting phase, the client will communicate only with the RMIServer or SocketServer, once the game has started, he will communicate only with the ThreadLobby -> the RMIServer and SocketServer will be in charge of only redirecting the user the to right lobby and to start it, and the ThreadLobby will be in charge only of managing the stuff for the game -> greater separation of task between thread.

## Scenarios

* **Initial connection with the server**

The client initially will communicate directly with either RMIServer or SocketServer, trying to establish the connection between them, once it succeeded, there will be a nickname setting phase (where it must be unique globally), and once it ended he will get a Json (using Gson) of the various active lobbies:

Immagine che contiene testo, schermata, diagramma, numero

Descrizione generata automaticamente

* **Join lobby**

Once the client is being successfully assigned a valid nickname, he can join a lobby or create a new one (being careful of maintaining a lobby nickname unique), then once he’s in a lobby he can:

* Leave the lobby (in this case he will get a newer version of the lobbies state) and notify the other player in the lobby that he has left.
* Set himself ready (and once everybody is ready, and after a certain amount of time, the game will start (creating everything it needs)) after setting his color that he will play during the game.

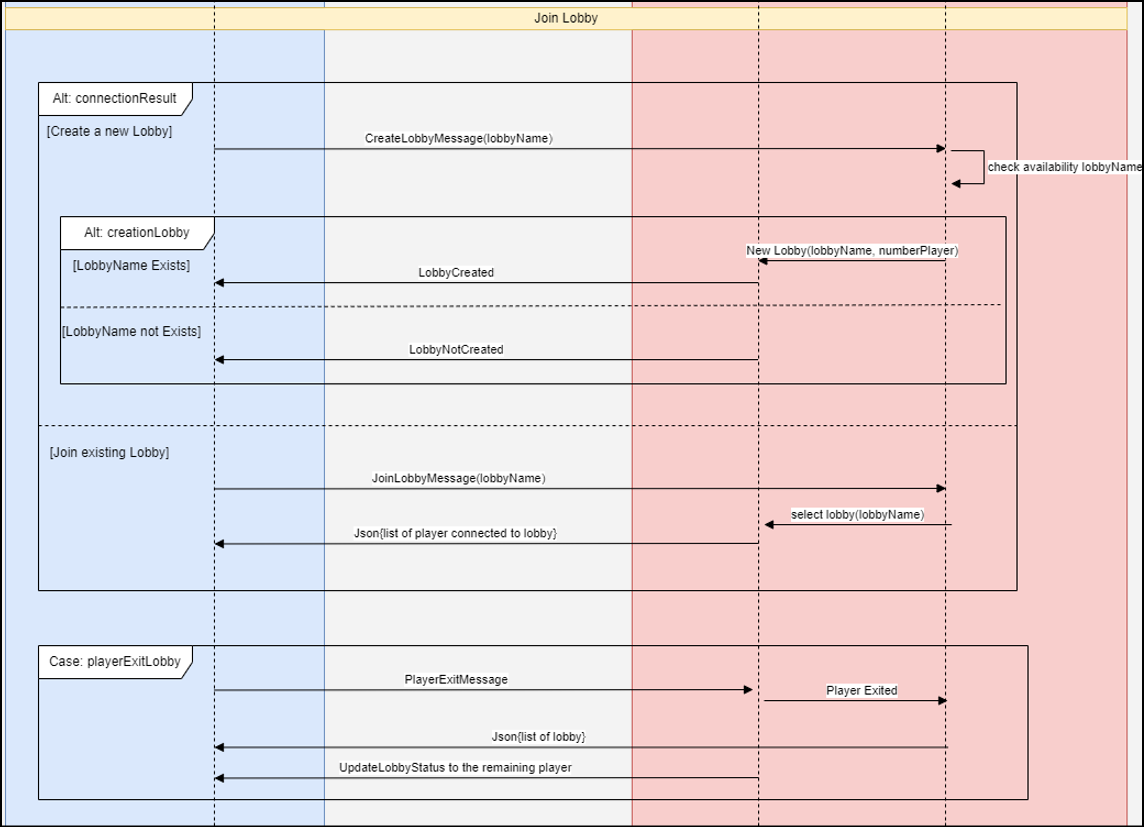


Immagine che contiene testo, schermata, Parallelo, diagramma

Descrizione generata automaticamente

* **Game Setup**

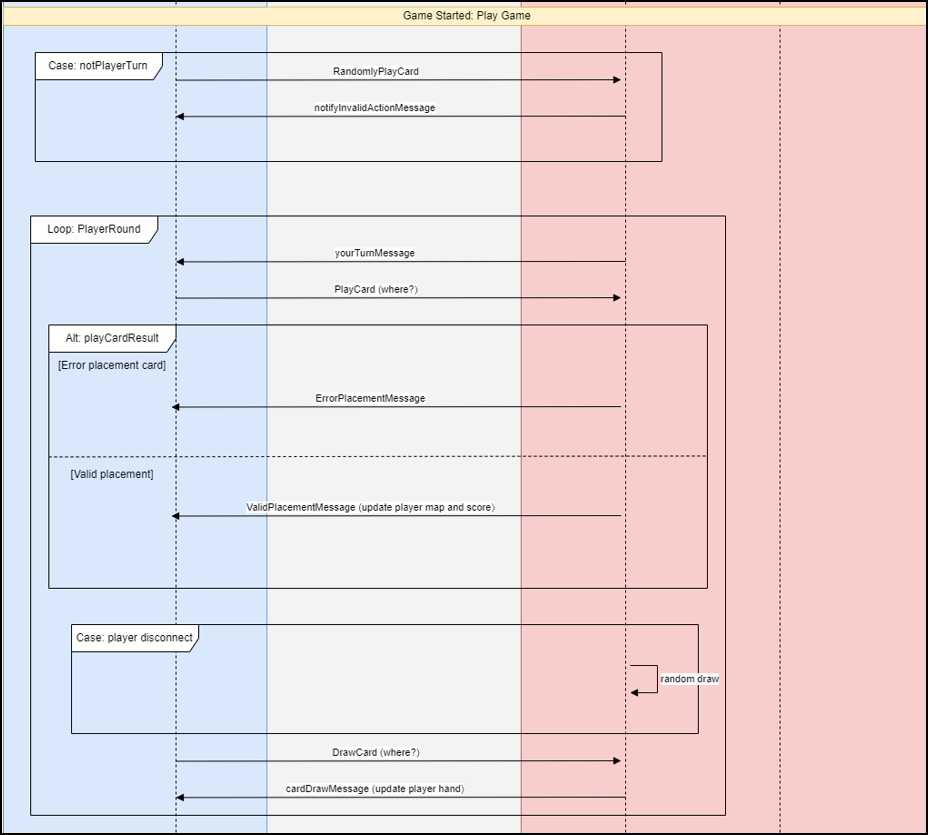
Once the lobby is started, the threadLobby will request the placement of the starterCard and the choice of the player of which personal mission he wants, once he has done that, the game can finally start (starting from the initial player that the lobbyThread will decide):

Immagine che contiene testo, schermata, numero, diagramma

Descrizione generata automaticamente

* **Play a player round**

A single round is composed of a playing phase and a drawing phase, in case the player gets disconnected after playing a card, the threadLobby will give him a random drawn card (so that when he reconnects, the card count will be coherent)



* **Disconnection of a player**

In case of a disconnection, we implemented a “heartbeat”, meaning the threadLobby will periodically ask to each client connected to it if he’s still alive, if he does not receive a feedback within a certain time frame, the threadLobby will “disconnect” him (until he receives any kind of message from him), this kind of solution is only for RMI since when a client disconnect there will not be any kind of exception thrown (so the server won’t know if a client is disconnected or not); this will not happen in the TCP case since technically the object that handle the communication will throw an exception when the client disconnect.

Immagine che contiene testo, schermata, diagramma, Parallelo

Descrizione generata automaticamente

* **Final phase of the game**

Once someone reach the 20 threshold, the threadLobby will notify the other players, conclude the round and play an additional turn, then he will check who won and finally notify the main thread that the game has ended.

Immagine che contiene testo, schermata, Parallelo, numero

Descrizione generata automaticamente