POLITECNICO
MILANO 1863

Protocol Desing

Codex Naturalis' Java Application

Ingegneria del software 1 - Prova Finale
Academic year 2023 - 2024

April 2024

Authors:
Lorenzo Galatea
Luca Lamperti
Alessandro Lodetti
Diego Quattrone

Chapter 1

Introduction

In this document we illustrate the specific of the Protocol Design for our Codex Naturalis' Java Application. Before proceeding, it is important to specify that we have chosen a Model-View-Controller (MVC) architecture, which is implemented through Server-Client communication. We will now proceed to illustrate three scenarios related to the use of our application. In particular, the messages exchanged between the Server and the Client are shown, assuming that the client implements everything related to the View. Once the communication is made to the Server, it will perform all the necessary operations to execute the desired operation, properly distributing roles and operations between the Model and the Controller. These internal operations are represented in the sequence diagrams as self-calls from the server.

Chapter 2

Sequence Diagrams

The first scenario (2.1) is the one in which a player wants to access a game. In particular, we have also specified the options related to the additional functionality of "resilience to disconnections".

Figure 2.1:

After choosing the communication technology and entering the nickname, the user attempts to join a game. The server manages exceptions and grants access to a game if available. If no games are available, the system prompts the user to specify the number of players to create a new game. Furthermore, the system includes features for implementing persistence and resilience against disconnections, enabling players to rejoin ongoing games using the gameID and nickname.

The second scenario (2.2) refers to the action of playing a card from the own hand.

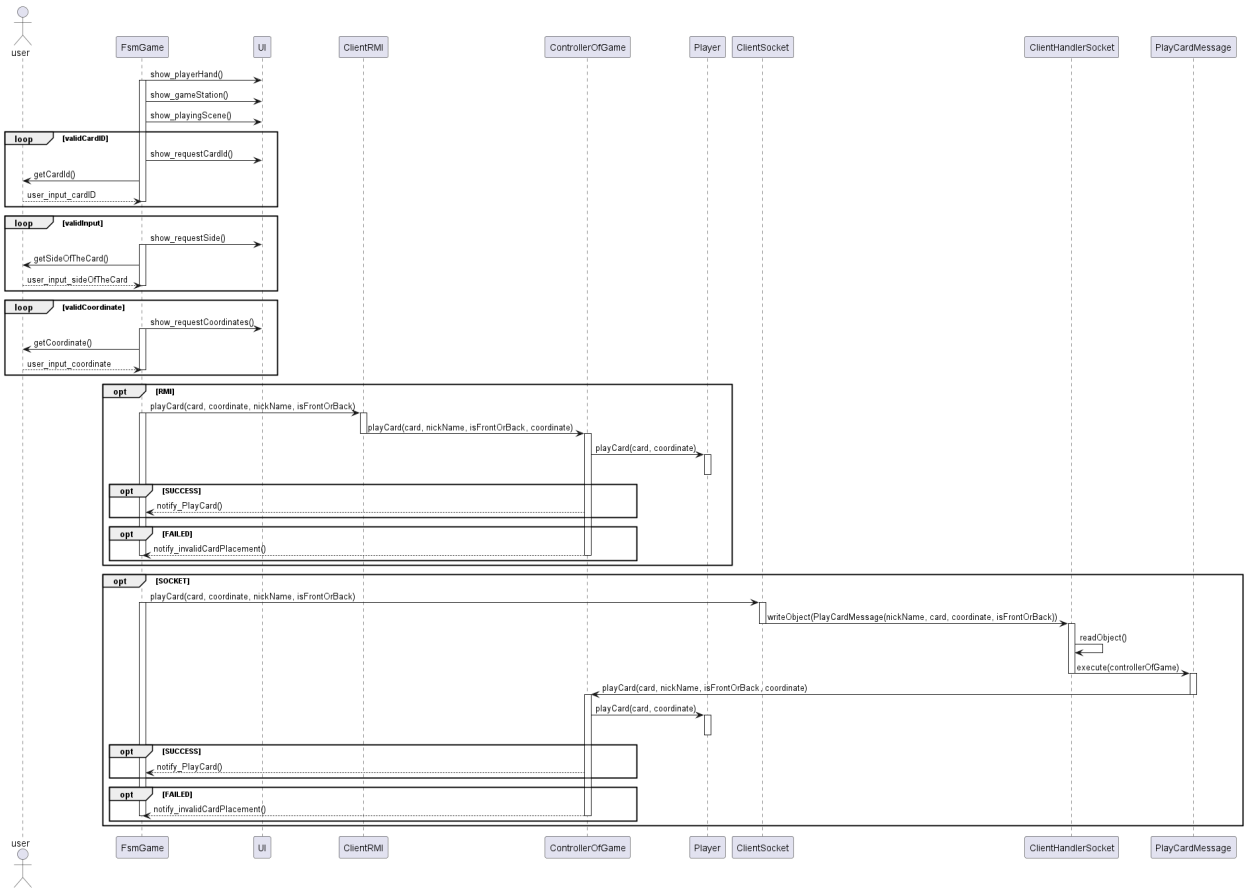

Figure 2.2:

The finite state machine of the game (FsmGame) displays the player's hand and the game station, then prompts the user to select a card, specify its side, and choose coordinates.

Based on the user inputs, FsmGame sends the play card request via RMI or SOCKET. The game controller validates the move and updates the game state, notifying FsmGame of success or failure.

The third scenario (2.3 represent the action of drawing a card from the table.

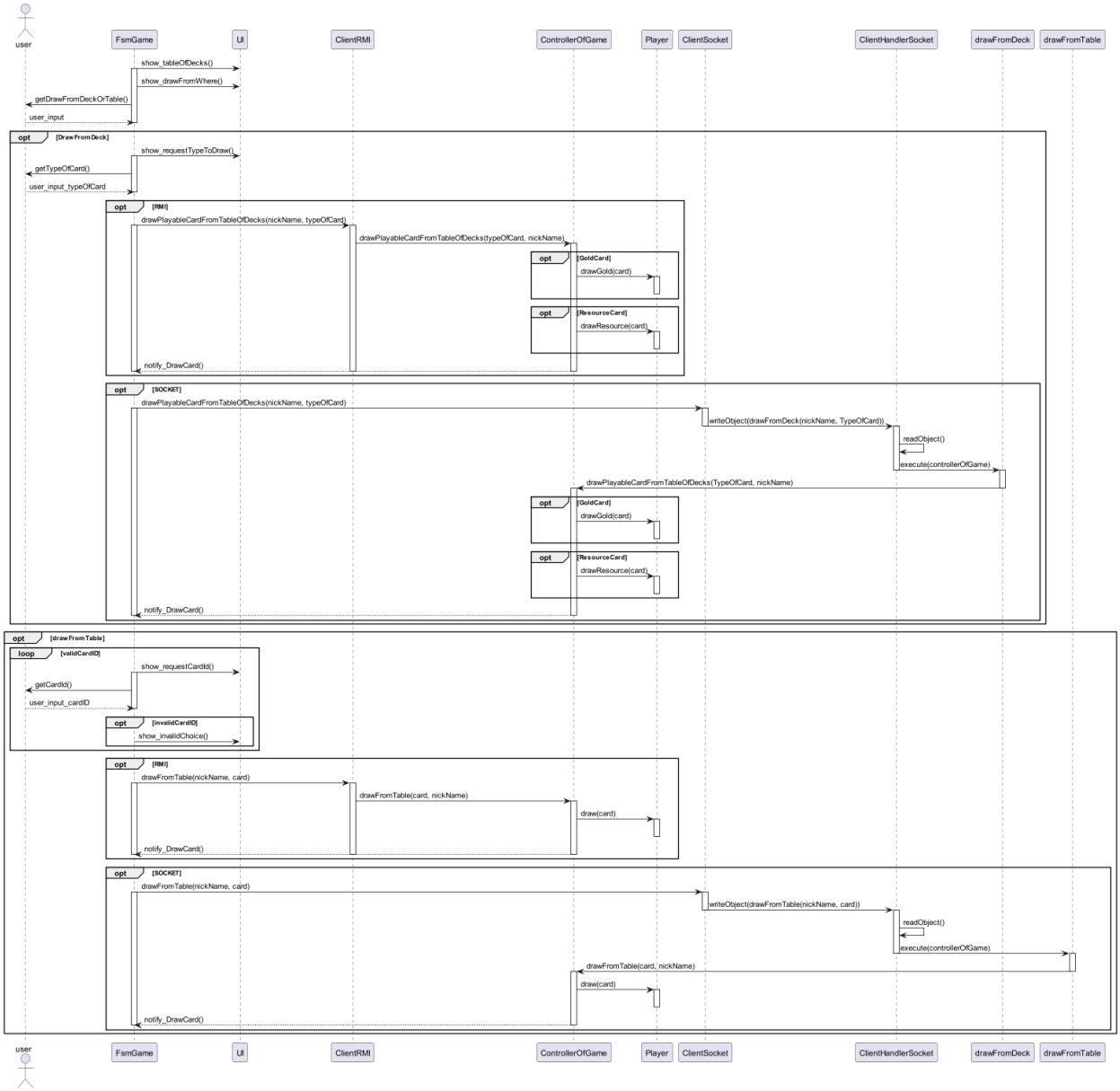

Figure 2.3:

The finite state machine of the game (FsmGame) requests the user interface to display the list of available cards, then prompts the user to choose whether to draw from a deck or from the table.

Based on the choice, network technology, and all the necessary inputs, FsmGame calls the correct function to update the player's hand and the table of decks.