



**POLITECNICO**  
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE  
E DELL'INFORMAZIONE

# Progetto finale di Ingegneria del Software

## Sequence Diagrams

**Authors:**

David Gadiaga  
Nicolas Grasselli  
Anita Grossi  
Rocco Nazzarelli

**Course:** Progetto Finale di Ingegneria del Software

**Professor:** Prof. San Pietro Pierluigi

**Academic Year:** 2024-25



# Contents

<b>Contents</b>	<b>iii</b>
<b>1 Introduzione</b>	<b>1</b>
<b>2 Sequence Diagrams</b>	<b>3</b>
2.1 Overview . . . . .	3
2.2 Sequence diagram 1: Accesso al gioco . . . . .	4
2.2.1 Protocollo . . . . .	5
2.2.2 Validazione del nickname . . . . .	5
2.2.3 Modalità Multiplayer . . . . .	5
2.3 Sequence diagram 2: «Pescare una carta componente dal tavolo» . . . . .	6
2.3.1 Protocollo . . . . .	6
2.3.2 Richiesta di pesca . . . . .	6
2.3.3 Risposte possibili del server . . . . .	6
2.4 Sequence diagram 3: Agganciare una carta componente . . . . .	8
2.4.1 Protocollo . . . . .	8
2.4.2 Richiesta di posizionamento . . . . .	9
2.4.3 Gestione degli errori . . . . .	9



# 1 | Introduzione

Galaxy Trucker è un gioco da tavolo dal ritmo incalzante in cui i giocatori assumono il ruolo di camionisti spaziali, costruendo navi spaziali a partire da tessere e navigando attraverso pericolose missioni spaziali. L'obiettivo è costruire una nave, consegnare il carico in modo sicuro ed evitare pericoli come meteoriti, pirati spaziali e malfunzionamenti. I giocatori devono assemblare rapidamente le loro navi, viaggiare attraverso la galassia e superare gli ostacoli per completare le loro missioni. I punti vengono assegnati in base al successo delle consegne, all'integrità della nave e all'efficienza.

I diagrammi di sequenza sono fondamentali per visualizzare il modo in cui i componenti del sistema di un gioco interagiscono nel tempo, soprattutto nelle versioni digitali come Galaxy Trucker. Questi diagrammi mostrano il flusso di messaggi tra il dispositivo del giocatore e il server, illustrando azioni come la connessione al gioco, il disegno delle tessere e il loro posizionamento. Aiutano gli sviluppatori a comprendere e documentare i processi di gioco, a eseguire il debug dei problemi e a garantire una comunicazione fluida tra il client e il server. In definitiva, i diagrammi di sequenza contribuiscono a creare un'esperienza di gioco intuitiva ed efficiente.



# 2 | Sequence Diagrams

## 2.1. Overview

I diagrammi di sequenza mostrati in questo capitolo illustrano i protocolli di comunicazione tra un client e un server nell'implementazione digitale di Galaxy Trucker. Questi diagrammi descrivono in dettaglio le interazioni che si verificano durante i momenti chiave del gioco, dalla connessione iniziale al server alle azioni come il disegno delle tessere e il loro posizionamento sul tabellone. Seguendo questi diagrammi, possiamo capire come le meccaniche di gioco vengono tradotte in un ambiente digitale, garantendo interazioni fluide tra i giocatori e il sistema.

I seguenti diagrammi di sequenza descrivono i processi coinvolti nell'accesso al gioco, nell'estrazione di una tessera dal tavolo e nel posizionamento di una tessera sul tabellone. Questi diagrammi sono essenziali per comprendere il flusso di dati e azioni che si verificano in ognuna di queste fasi del gioco.

## 2.2. Sequence diagram 1: Accesso al gioco

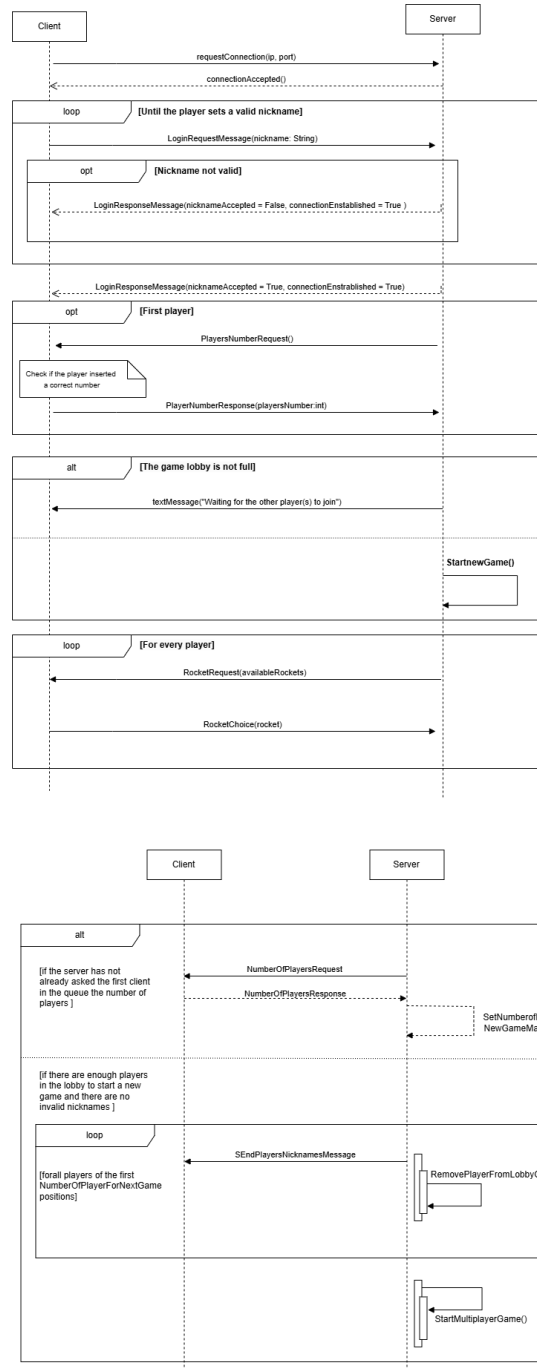


Figure 2.1: Sequence diagram 1: Accesso al gioco



### 2.2.1. Protocollo

Dopo che il server è stato avviato con un numero di porta specifico, un client può stabilire una connessione fornendo l'indirizzo IP e il numero di porta del server oppure utilizzando la configurazione predefinita.

Una volta accettata la connessione, il server invia un `LoginRequestMessage` al client, richiedendo di inserire un nickname. Il client invia quindi un `LoginResponseMessage(nickname: String)` contenente il nickname scelto.

### 2.2.2. Validazione del nickname

Il server verifica se il nickname soddisfa i criteri sintattici.

Se il nickname è già in uso, il server risponde con `LoginResponseMessage(nicknameAccepted = False)` e richiede un nuovo nickname.

Se il nickname è valido e unico, il server procede con il flusso di creazione della partita.

### 2.2.3. Modalità Multiplayer

Il nickname viene salvato nella struttura del server associata alla connessione del client.

Se il client è il primo a entrare nella lobby, il server invia una richiesta `PlayersNumberRequest()`, chiedendo di scegliere il numero di giocatori totali.

Il client risponde con un `PlayersNumberResponse(numPlayers: int)`, specificando il numero desiderato.

Il server attende che altri giocatori si uniscano finché non viene raggiunto il numero richiesto.

Quando la lobby è completa, il server avvia la partita e rimuove i giocatori dalla coda di attesa.

## 2.3. Sequence diagram 2: «Pescare una carta componente dal tavolo»

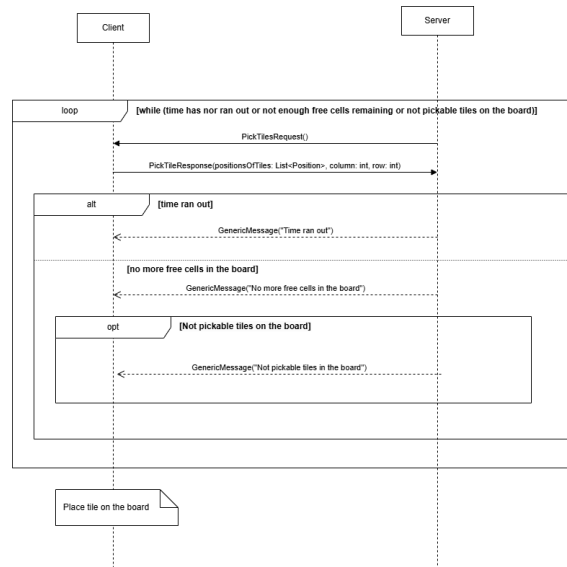


Figure 2.2: Sequence diagram 2: Pescare una Carta dal Tavolo

ù

### 2.3.1. Protocollo

Dopo che un giocatore ha effettuato l'accesso ed è in una partita attiva, può interagire con il tavolo da gioco per pescare una carta.

### 2.3.2. Richiesta di pesca

Il client invia una richiesta al server per pescare una carta, attraverso il messaggio `PickTilesRequest()`.

Il server verifica la disponibilità di carte pescabili sul tavolo.

Se esistono carte pescabili, il server risponde con `PickTileResponse(positionsOfTiles: List<Position>)` fornendo l'elenco delle posizioni disponibili.

### 2.3.3. Risposte possibili del server

Se il tempo per la pesca è scaduto, il server risponde con `GenericMessage("Time ran out")` e l'azione viene annullata.

Se non ci sono più caselle disponibili, il server invia `GenericMessage("No more`

`free cells").`

Se nessuna carta è pescabile, il server notifica il client con `GenericMessage("Not pickable tiles on the board")`.

Dopo la conferma della posizione di pesca, il client procede a posizionare la carta nella propria area di gioco.

## 2.4. Sequence diagram 3: Agganciare una carta componente

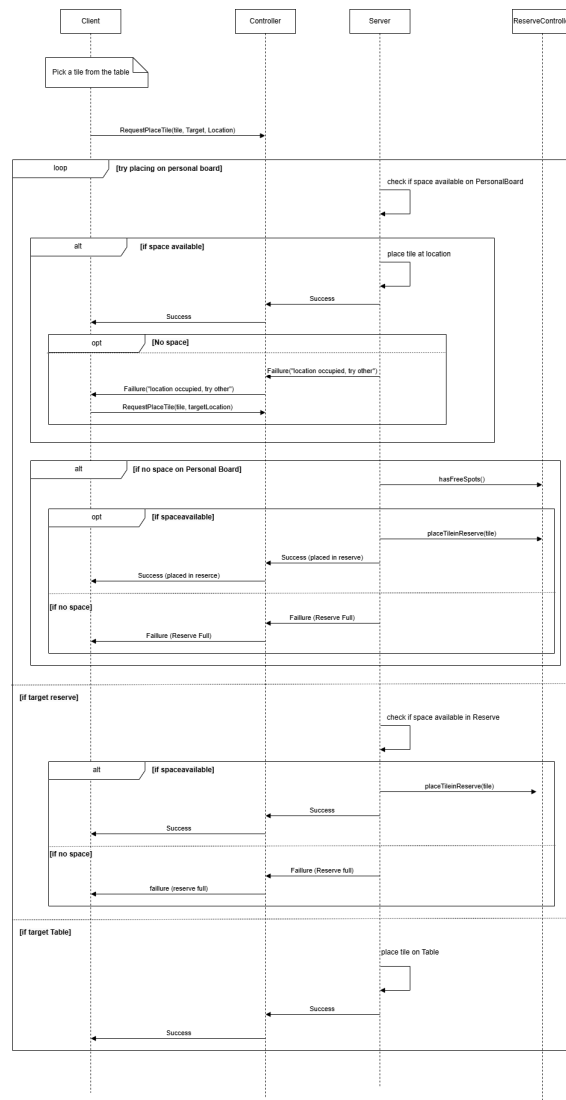


Figure 2.3: Sequence diagram 3: Agganciare una Carta

### 2.4.1. Protocollo

Dopo aver pescato una carta, il giocatore deve decidere dove posizionarla sulla propria plancia di gioco.

### 2.4.2. Richiesta di posizionamento

Il client invia una richiesta `RequestPlaceTile(tile, target, location)` al server, specificando la carta scelta e la posizione desiderata sulla plancia.

Il server verifica se la posizione è valida:

- Se la posizione è libera, il server posiziona la carta e risponde con **Success**.
- Se la posizione è occupata, il server invia un messaggio di errore e il client deve selezionare una nuova posizione.

### 2.4.3. Gestione degli errori

Se non ci sono più posizioni disponibili sulla plancia, il server notifica il client e gli impedisce di posizionare la carta.

Il client continua a provare posizioni alternative fino a trovare un'area libera.

