PROTOCOLLO DI COMUNICAZIONE

IS24-AM44

Fumagalli Edoardo
Dri Tommaso
Bongiovanni Cammila
Cadario Riccardo

Sommario

1. Struttura dei messaggi

1. Introduzione

Ogni messaggio scambiato avrà un messageCode contenente una stringa che specifica di che tipo di messaggio si tratta, e quindi come deve essere interpretato dal ricevente. Useremo JSON per mostrare esempi di messaggi.

2. Struttura

- **serverRequest** – Ad esempio quando il server chiede al client di indicare il suo nickname il messaggio sarà di questo tipo:
  {
  “messageCode” : “serverRequest”,
  “argument” : “nickname”
  }
  Altri parametri dell’ attributo “argument” possono essere “move”, “number of player”…
- **serverResponse** – Questo tipo di messaggio serve ad indicare al client attualmente in comunicazione col server se tutte le task sono state completate con successo o meno.
  {
  “code” : “serverConfirm”,
  “message” : “Lobby created”
  }
- **updateView (Server->Client).**
  {
  “code” : “updateView”,
  “player” : “nickname”
  “playedCard” : PlayableCard // json object of a playableCard
  “coordinates” : Coordinates // json object of the coordinates
  }
- **clientConnectionRequest**
  {
  “code”: “hello”
  }
- **clientResponse** – Il messaggio di risposta del client avrà un attributo type, che indica di che tipo e quindi come va interpretato il suo contenuto, ovvero l’attributo parameter.
  {
  “code” : “move”,
  “playedCard” : PlayableCard // json object of a playableCard
  “coordinates” : Coordinates // json object of the coordinates
  }
  {
  “code” : “playerNumber”,
  “maxPlayerNumber”: “3”
  }
- **privateMessage**
  {
  “code” : “privateMessage”,
  “sender” : “nicknameClient1”,

“receiver” : “nicknameClient2”,
“body” : “messageContent”
}

- **broadcastMessage.**
  {
  “code” : “broadCastMessage”,
  “sender” : “nicknameClient1”,
  “body” : ”messageContent”
  }

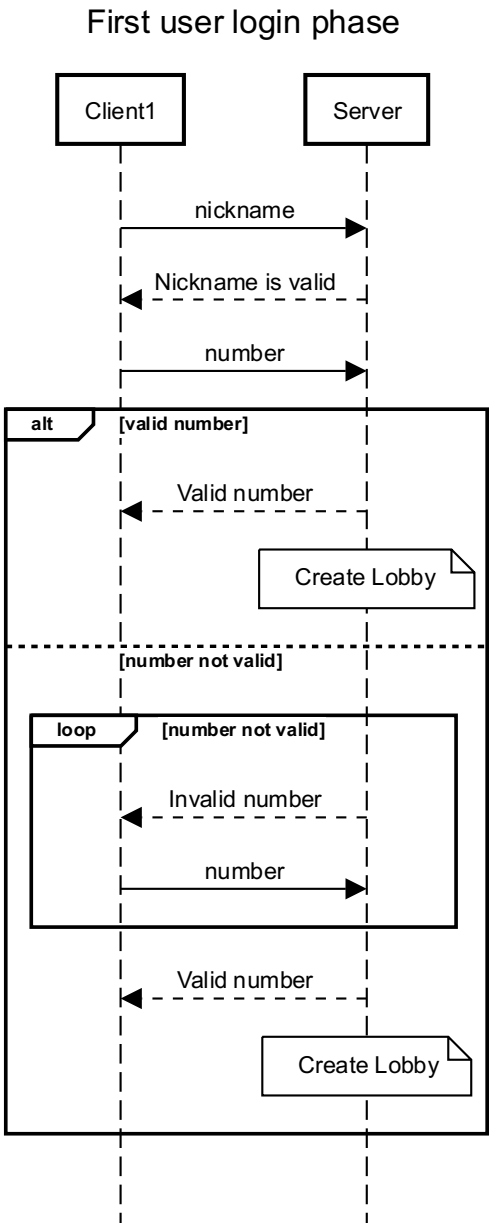
2. Scambio dei messaggi

1. Introduzione

L'intero protocollo di comunicazione tra client e server che abbiamo intenzione di implementare può essere diviso in 5 fasi: 2 fasi per il login, una fase per l'inizializzazione del gioco, una fase per lo svolgimento del gioco ed infine una fase di comunicazione per poter gestire la chat.

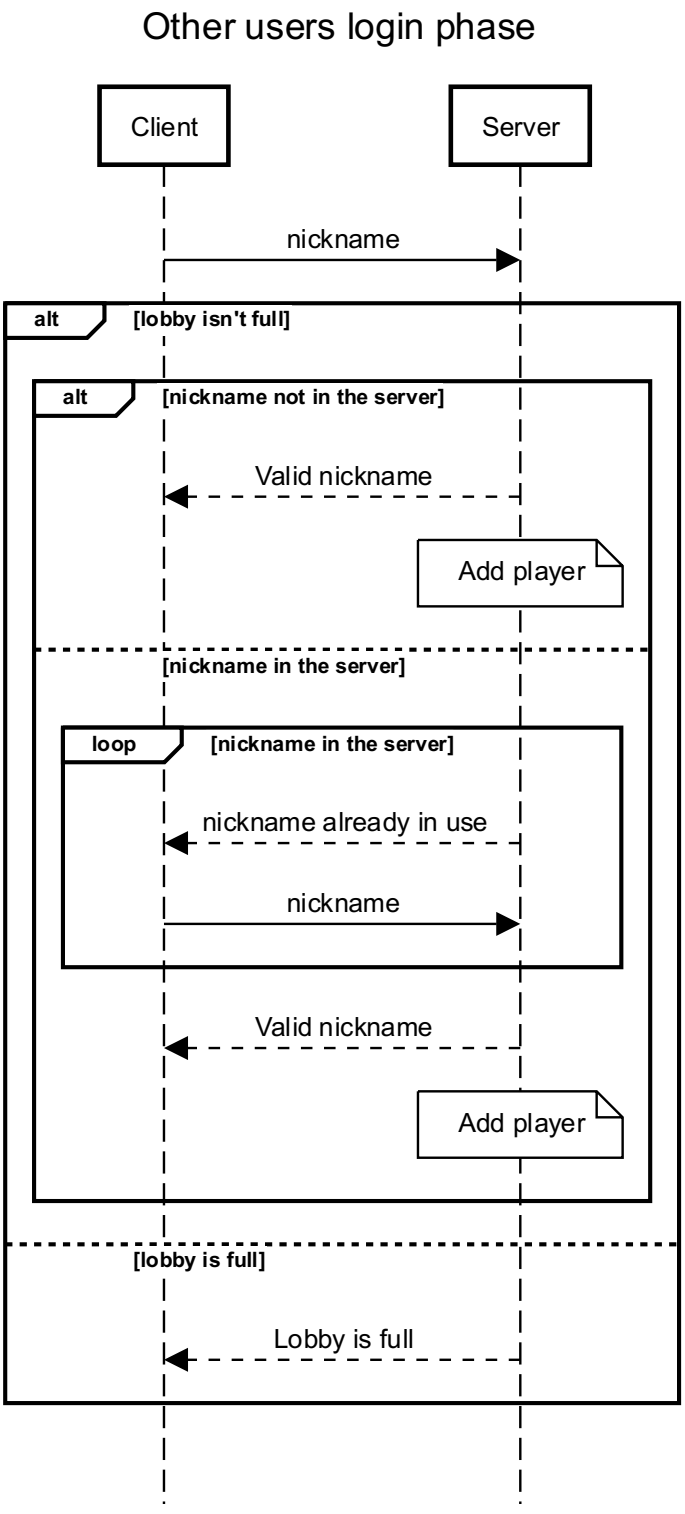
2. First User Login

E' la fase in cui si connette il primo giocatore in assoluto. Questo manda un messaggio di connection request al quale il cliente risponde di eseguire il login inviando un nickname. Una volta ricevuto il server chiede al client di scegliere il numero dei giocatori della partita, e se il numero è valido (2, 3 o 4) il Server crea la lobby per la partita.



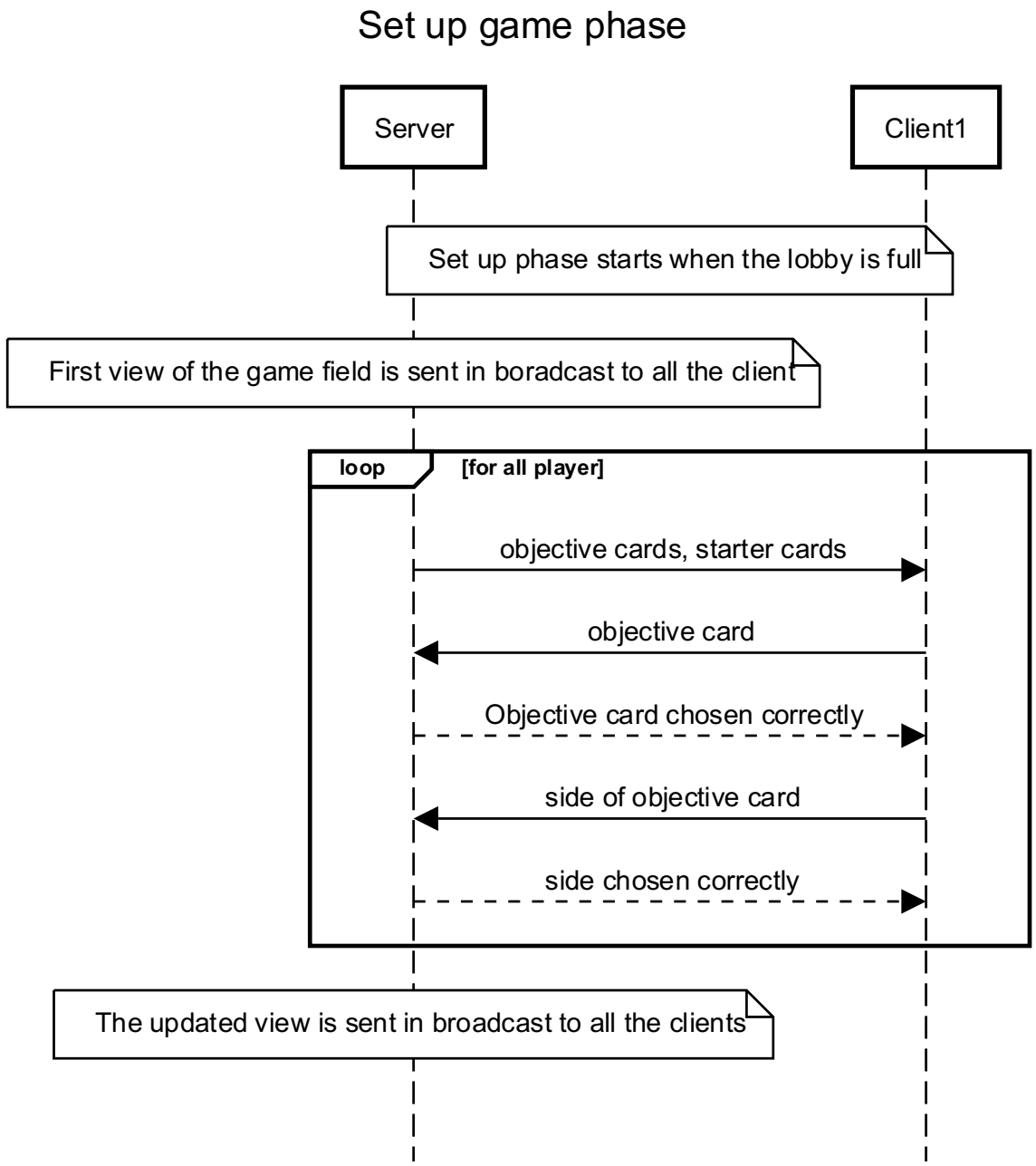
3. Other users login phase

Una volta create la lobby il server resta in attesa di altri giocatori che si connettano. Quando un client chiede il permesso di connettersi il server chiede di inserire un nickname fino a quando il client non risponde con un nickname valido (che non sia già stato usato). Una volta che il client soddisfa questa richiesta, il player viene aggiunto al gioco. Se la lobby è piena la richiesta di connessione viene rifiutata.



4. Set up game phase

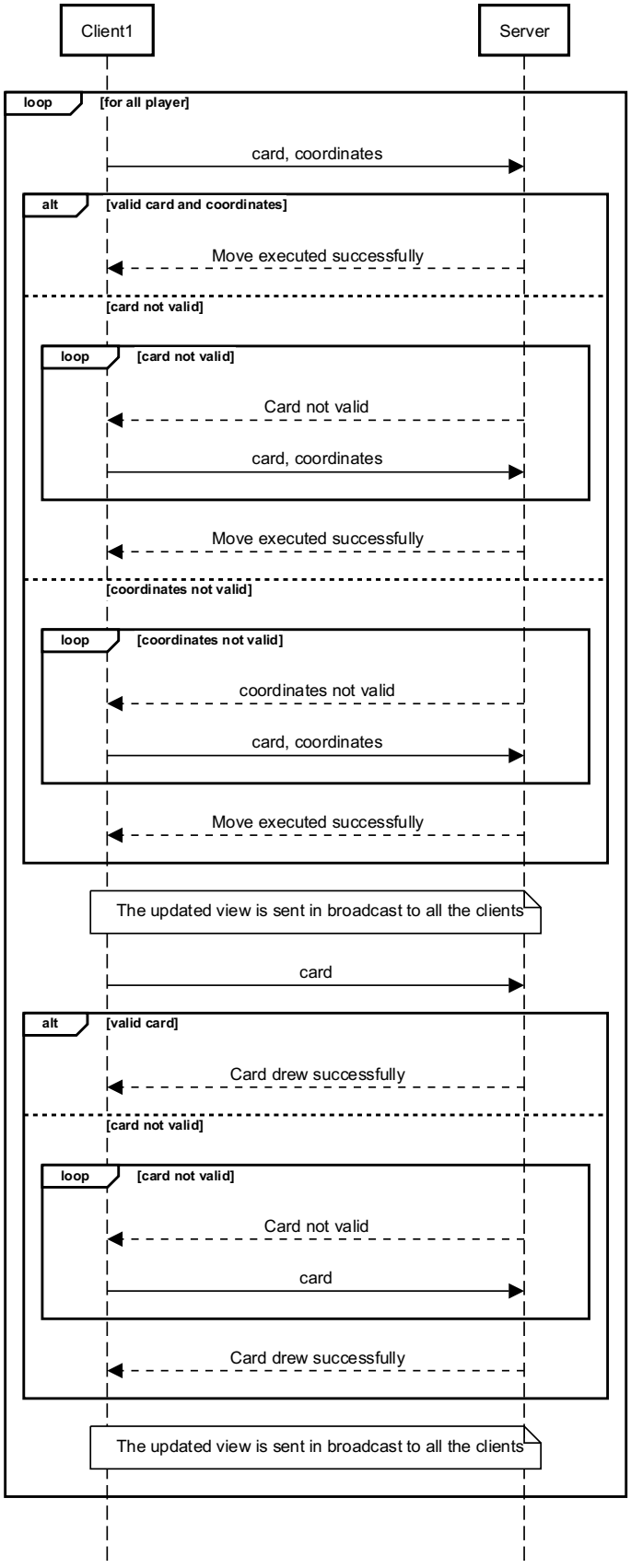
Una volta che la lobby è piena inizia la fase di set-up della partita. Il server crea deck, board… e manda a tutti i client la prima view del campo da gioco incompleto. In seguito chiede ad ogni giocatore di scegliere una carta obbiettivo tra due opzioni e da che lato vogliono giocarla, rispondendo sempre con dei messaggi di conferma. Una volta che l'interazione è avvenuta correttamente con ogni giocatore il campo è pronto per iniziare a giocare e il server invia a tutti i client la view aggiornata del campo da gioco.



5. Running game phase

La running game phase è la fase di comunicazione attraverso la quale il giocatore effettua le mosse durante il proprio turno.

Running game phase



6. Chat handling

Il protocollo di invio dei messaggi della chat è descritto nel diagramma seguente. Quando un client desidera inviare un messaggio invia una richiesta al server, che inoltra il contenuto del messaggio e il mittente a chi di dovere: a tutti i client se il messaggio è di tipo broadcast, al client specificato se il messaggio è di tipo peer to peer. Per i messaggi peer to peer viene sempre eseguito un controllo per verificare che il nickname del destinatario del messaggio sia valido, ovvero sia il nickname di un altro giocatore: se non lo è il server risponde con un messaggio di errore.