

CABO ONLINE



PROGETTO PER IL CORSO
“SISTEMI DISTRIBUITI”
ANNO ACCADEMICO
2024/2025

Il Gioco

- Gioco di carte a turni da 2 a 5 giocatori
- In questa versione vengono usate carte francesi, il seme della carta è influente sul punteggio
- **Obiettivo:** avere meno punti degli altri al termine della partita

CARTA	PUNTEGGIO	POTERE
ASSO	1	NESSUNO
DUE	2	NESSUNO
TRE	3	NESSUNO
QUATTRO	4	NESSUNO
CINQUE	5	NESSUNO
SEI	6	NESSUNO
SETTE	7	NESSUNO
OTTO	8	NESSUNO
NOVE	9	NESSUNO
DIECI	0	NESSUNO
J	10	VEDI 1 PROPRIA CARTA
Q	10	VEDI 1 CARTA AVVERSARIA
K	10	SCAMBIO CARTA

Svolgimento

Inizio:

- Vengono mischiate le carte e assegnate 4 ad ogni giocatore, posizionata una carta scoperta come inizio del mazzo degli scarti e il restante mazzo coperto
- Ogni giocatore prima di iniziare la partita guarda 2 delle proprie carte
- Iniziano i turni

Svolgimento turno:

- Si pesca la prima carta del mazzo degli scarti o di quello coperto
- La carta pescata viene mostrata a tutti
- Se pescata dal mazzo degli scarti non è scartabile ne utilizzabile il relativo potere
- Si utilizza il potere se presente altrimenti si decide se scartare una delle proprie carte o quella appena pescata
- È possibile dichiarare cabo o la fine del turno: il giocatore successivo inizia il proprio

Termine

Il gioco termina se:

- Il mazzo coperto non ha più carte
- È nuovamente il turno del giocatore che ha dichiarato cabo

Il giocatore con il punteggio minore vince

AKKA

- Toolkit per costruire applicazioni concorrenti, distribuite e resilienti basate sulla JVM (Java/Scala)
- Si basa sul **Modello ad Attori**: ogni entità è un attore passivo che reagisce solo alla ricezione di messaggi

Akka Cluster:

- Estensione di Akka che permette a diversi nodi (JVM separate) di formare un sistema distribuito unico e coeso
- Permette la comunicazione tra attori su macchine diverse in modo trasparente, senza bisogno di un server centrale per il dispatching dei messaggi
- Offre strumenti nativi per la gestione della membership e la tolleranza ai guasti

Messaggi

- Tutte le entità interagiscono scambiandosi messaggi specifici definiti nel protocollo.
- I messaggi devono viaggiare anche sulla rete, quindi devono essere convertiti in byte
- Jackson-CBOR è stato utilizzato come formato di serializzazione
- Tutti i messaggi hanno un *marker trait* per essere riconosciuti e serializzati automaticamente dal cluster
- Akka garantisce che i messaggi vengano consegnati al massimo una volta
- Le funzionalità fornite ci hanno permesso di concentrarci maggiormente sul nostro protocollo di messaggistica e logica dell'applicazione

Gossip & Heartbeat

Gossip Protocol

- Meccanismo con cui i nodi diffondono le informazioni sullo stato del cluster (es. chi è "Up", chi è "Down")
- Non c'è un server centrale che controlla tutto: ogni nodo fa “gossip” con i vicini per disseminare le informazioni

Heartbeat

- Il sistema usa gli eventi del cluster come meccanismo di heartbeat per rilevare se un nodo non è più raggiungibile
- Se un nodo smette di rispondere (es. crash o disconnessione), il cluster lo marca come Unreachable tramite il Gossip

Receptionist & Distributed data

Receptionist

- Permette di registrare un attore con un codice noto (es. il codice della partita) per essere trovato dagli altri senza conoscere il suo indirizzo fisico esatto
- Durante il gossip i differenti “registri” vengono comunicati anche agli altri nodi

Distributed Data

- Strumento di Akka per condividere dati tra i nodi senza un database esterno
- Utilizza i CRDTs (Conflict-Free Replicated Data Types) per gestire le scritture concorrenti senza conflitti bloccanti
- Ha permesso di replicare le informazioni delle differenti partite registrate nei server migliorando la fault tolerance

Split Brain Resolver

- Funzionalità per rilevare e gestire le partizioni di rete
- Ogni nodo, grazie al protocollo Gossip, conosce la dimensione dei due gruppi al momento della separazione e sa se si trova nella maggioranza o nella minoranza
- Il progetto adotta la strategia di **spegnere il gruppo più piccolo** di nodi permettendo al numero più grande di giocatori di continuare le partite senza interruzioni

Informazioni sul sistema

- Ogni entità è sviluppata come un Attore, secondo la documentazione di Akka
- Due component principali:

SERVER

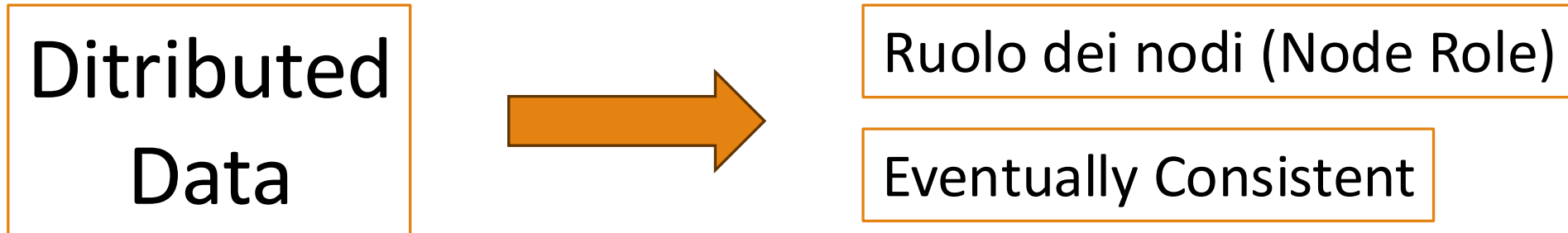
- Lobby Server per accedere alle partite pubbliche
- Singolo attore

CLIENT

- Desktop application per giocatore
- Costruito secondo il pattern MVC
- Diversi attori che fungono da controller
- I Client comunicano tra di loro seguendo un paradigma Peer-to-peer (P2P)

Server

- Server lobby per condividere le partite tra i giocatori nel sistema
- Modellato come un unico attore a singolo stato che mantiene la lista delle partite con lui condivise
- Nel sistema possono esserci più istanze di questo attore per motivi di fault tolerance e load balancing
- Tutte condividono la stessa lista

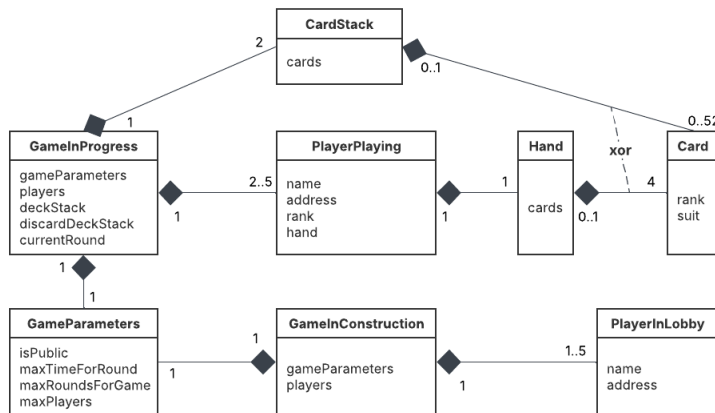


Client

- Desktop application usata da ogni giocatore
- Effettivo peer nel sistema P2P
- Modellato secondo MVC

M

Domain entities



V

Finestre realizzate
con Java Swing



C

Attori

Client

- Gestione connessioni

GameCoordinator

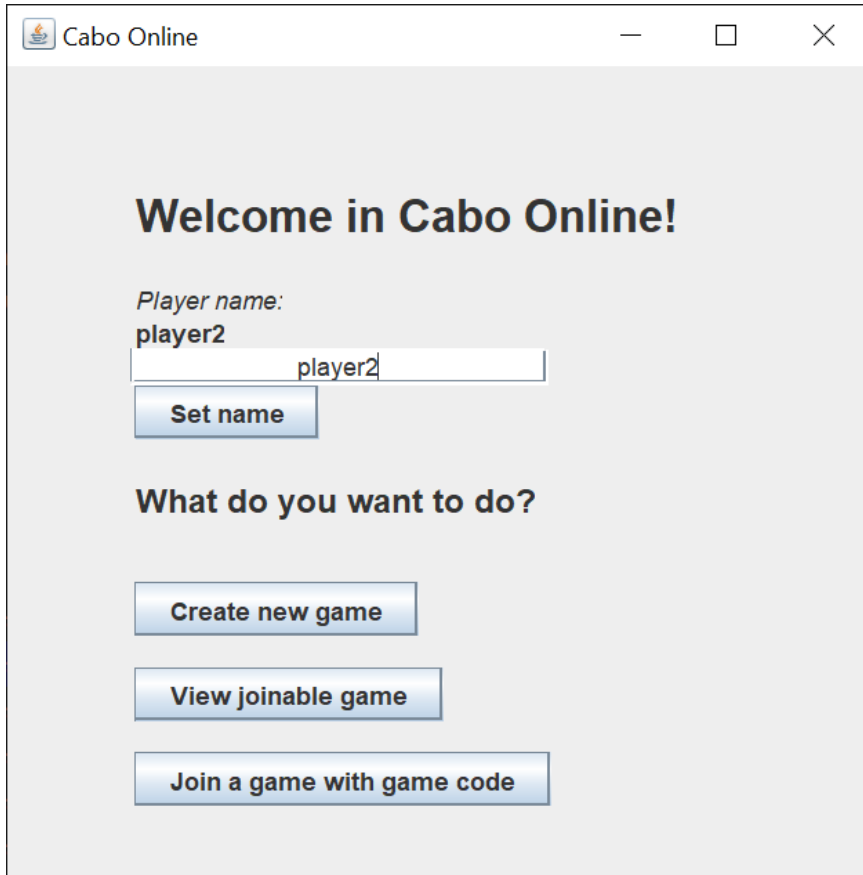
- Gestione logica gioco

ActorView

- Gestione schermate

Avvio dell'applicazione

- All'avvio l'applicazione si collega all'Akka Cluster
- Creazione attori Client e ActorView



The screenshot shows a web application window titled "Cabo Online". The main heading is "Welcome in Cabo Online!". Below this, there is a section for "Player name:" with the text "player2" entered in a text input field. A "Set name" button is located below the input field. Underneath, the question "What do you want to do?" is displayed. Three buttons are listed: "Create new game", "View joinable game", and "Join a game with game code".

Tre possibili azioni:

- Cambiare il proprio nome
- Creare una partita
- Unirsi ad una partita

Due diversi
comportamenti



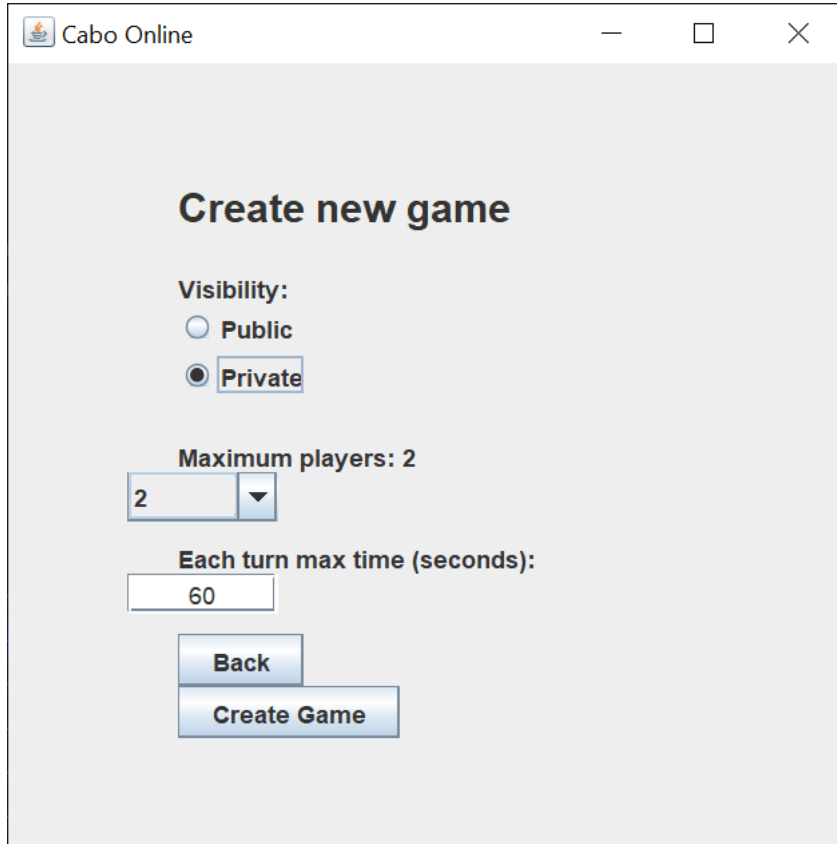
HOST



JOINEE

Creazione di una partita (1)

- Schermata per selezionare i parametri di una partita



The screenshot shows a web browser window titled 'Cabo Online'. The main heading is 'Create new game'. Under the 'Visibility:' section, there are two radio buttons: 'Public' (unselected) and 'Private' (selected). Below this, the 'Maximum players: 2' is shown with a dropdown menu currently displaying '2'. The 'Each turn max time (seconds):' is set to '60' in a text input field. At the bottom, there are two buttons: 'Back' and 'Create Game'.

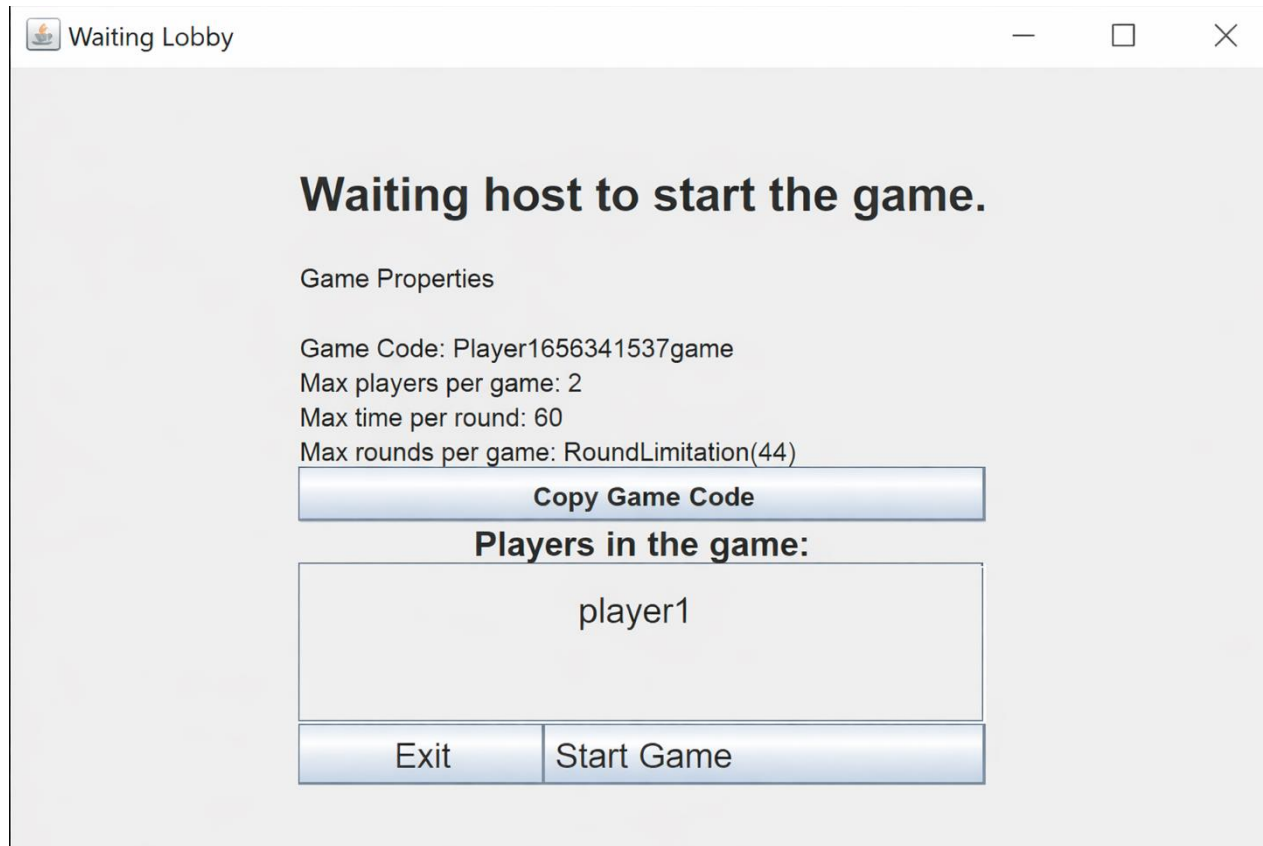
- Visibility



Condividere la partita tramite il server lobby

Creazione di una partita (2)

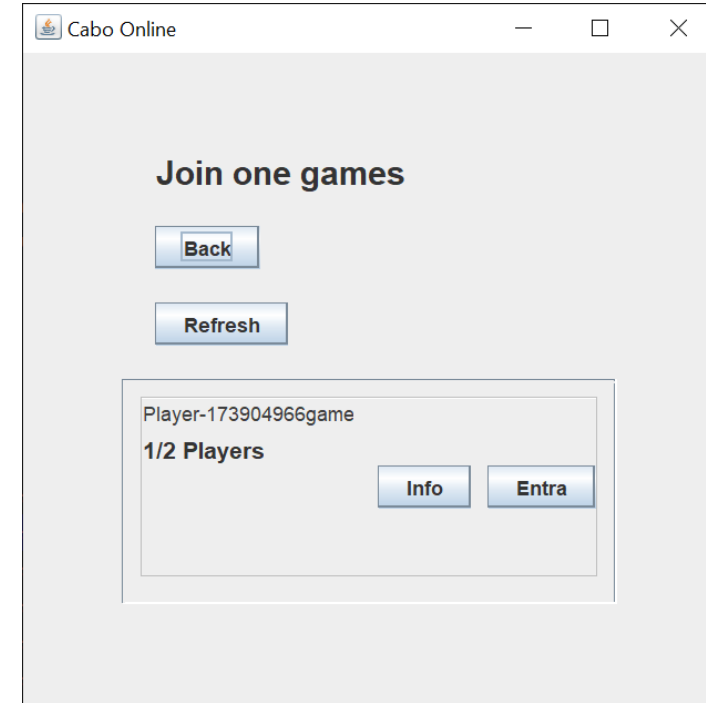
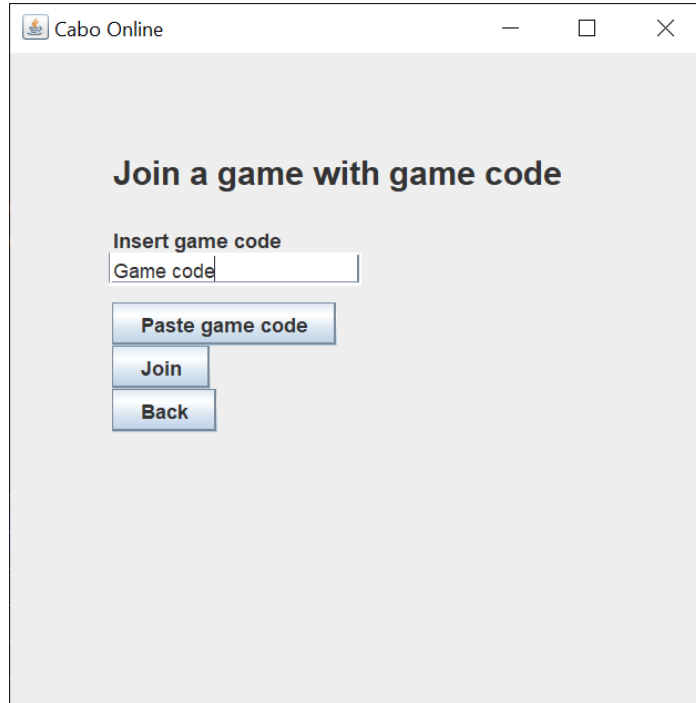
- Schermata di attesa dell'ingresso di altri giocatori



- Possibilità di copiare il codice del gioco per condividerlo con altri giocatori
- Il client si registra sul receptionist usando come key il gameCode

Unione ad una partita

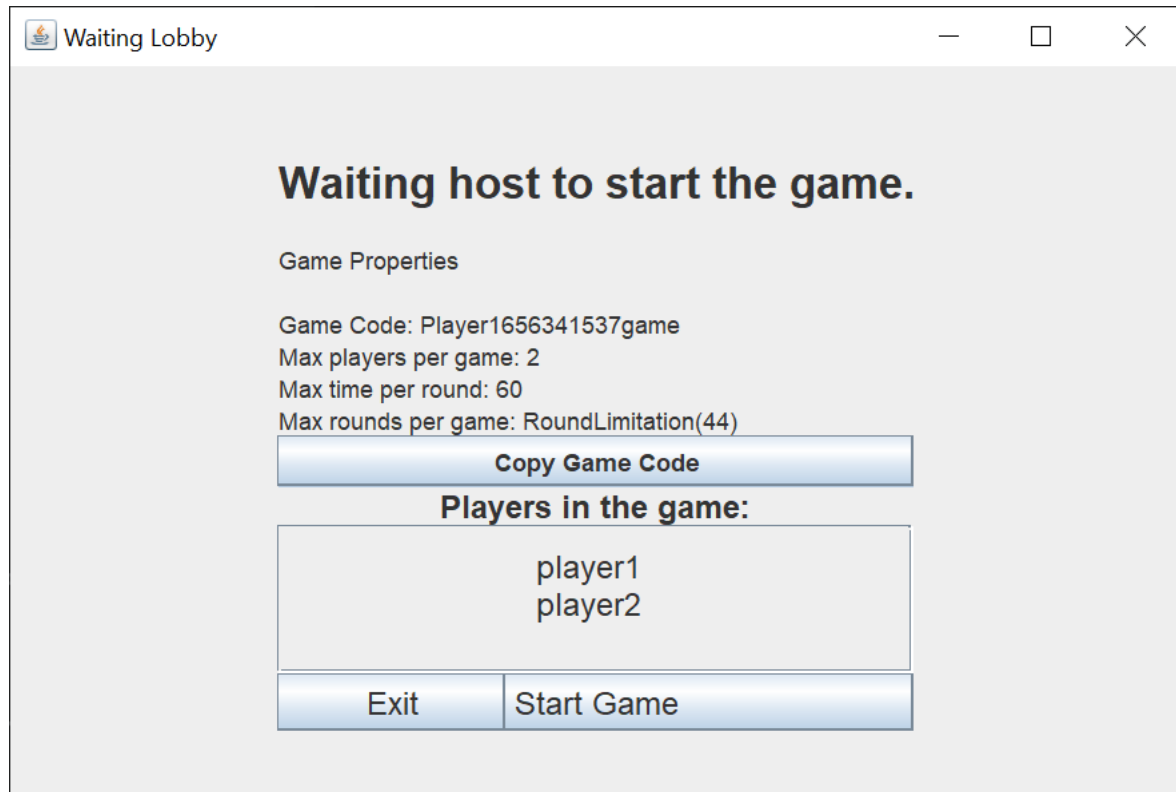
- In base alla metodologia di ingresso si vedrà una delle seguenti schermate



- In ogni caso si invierà un messaggio di richiesta di join all'attore client dell'host

Attesa inizio partita

- Schermata di attesa prima dell'inizio della partita



Inizio heartbeat mechanism



Uso dei messaggi di sistema di Akka



Se un joinee crasha o lascia la partita viene rimosso dall'host

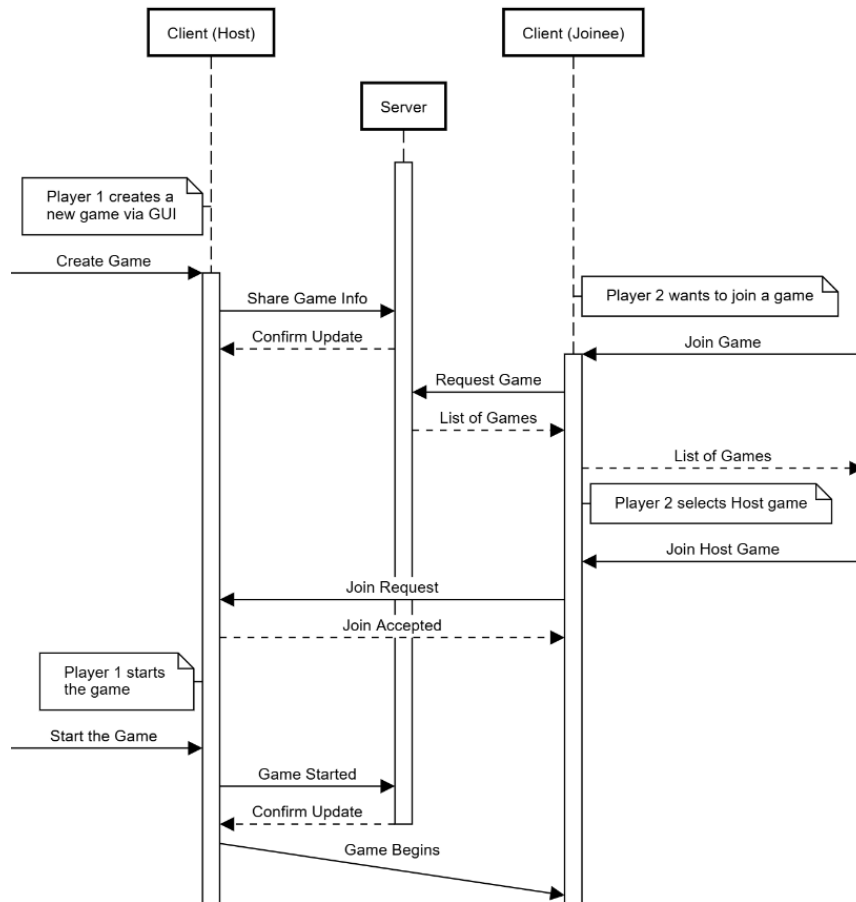


Se l'host crasha o lascia la partita, quest'ultima viene chiusa

- La partita inizia quando il giocatore che l'ha creata (host) preme 'Start Game'

Dal punto di vista dei messaggi (1)

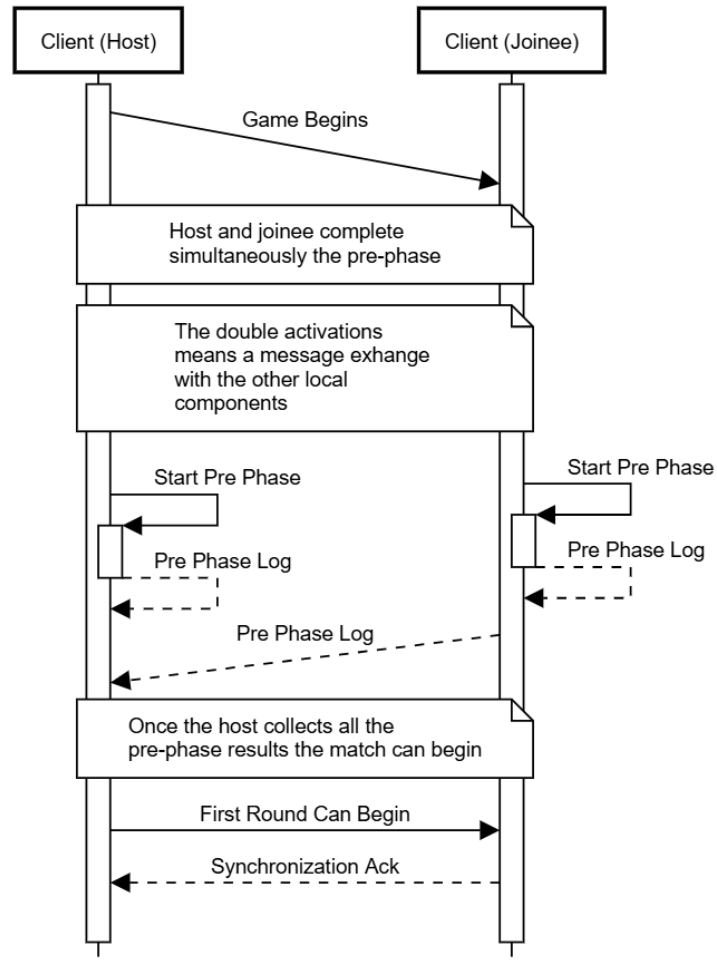
- Scambio di messaggi fino a questo punto



- Server contattato solo se la partita è pubblica
- Strong consistency
 - L'host è l'unico che può modificare lo stato della partita (i.e. L'elenco dei giocatori)
 - Ogni modifica è condivisa e confermata con un acknowledge message
 - Se qualcuno non conferma la partita è annullata

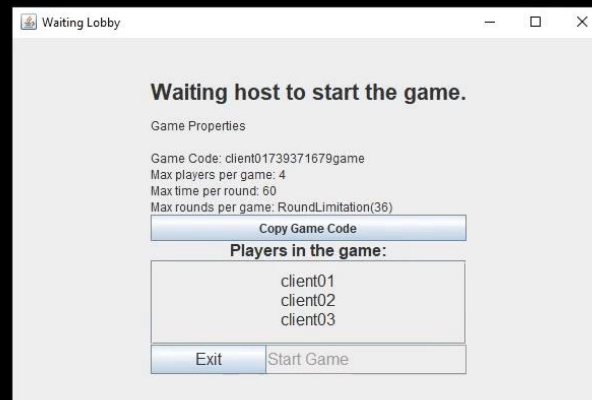
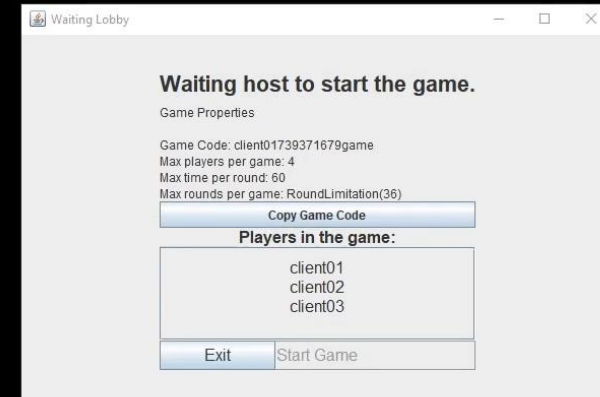
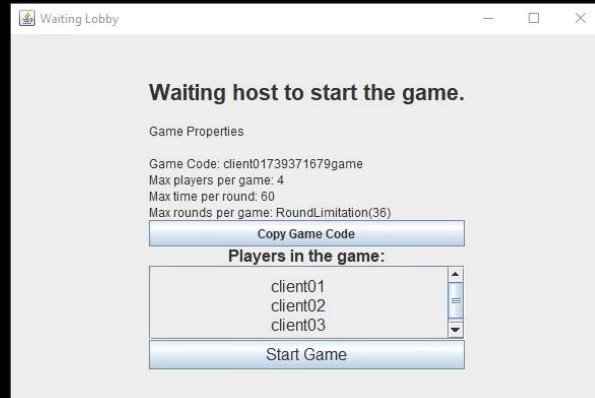
Dal punto di vista dei messaggi (2)

- All'avvio della partita



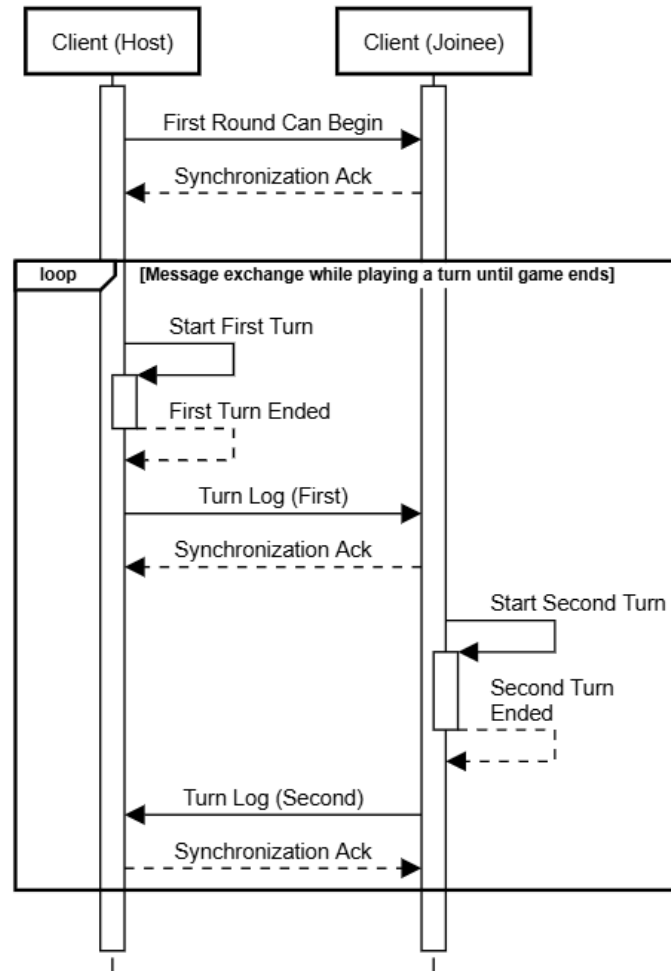
- Elaborazione della 'pre phase' per ogni giocatore
- L'host raccoglie tutti i log dei giocatori e li condivide con gli altri
- Strong consistency anche in questo caso

'Pre Phase' - Revealing Section



Svolgimento di un turno

- Durante il suo turno il giocatore sceglie le sue azioni e poi le invia agli altri tramite un log



- Timeout e retry
- Se il prossimo non giocatore non risponde, salta un turno e la parola va al successivo
- Se un altro giocatore non risponde il gioco continua, con l'idea di riliannearlo in seguito
- Availability over Consistency

Gestione dell'host

- In ogni partita l'host deve svolgere dei compiti specifici in base alla fase in cui ci si trova

Pre Partita

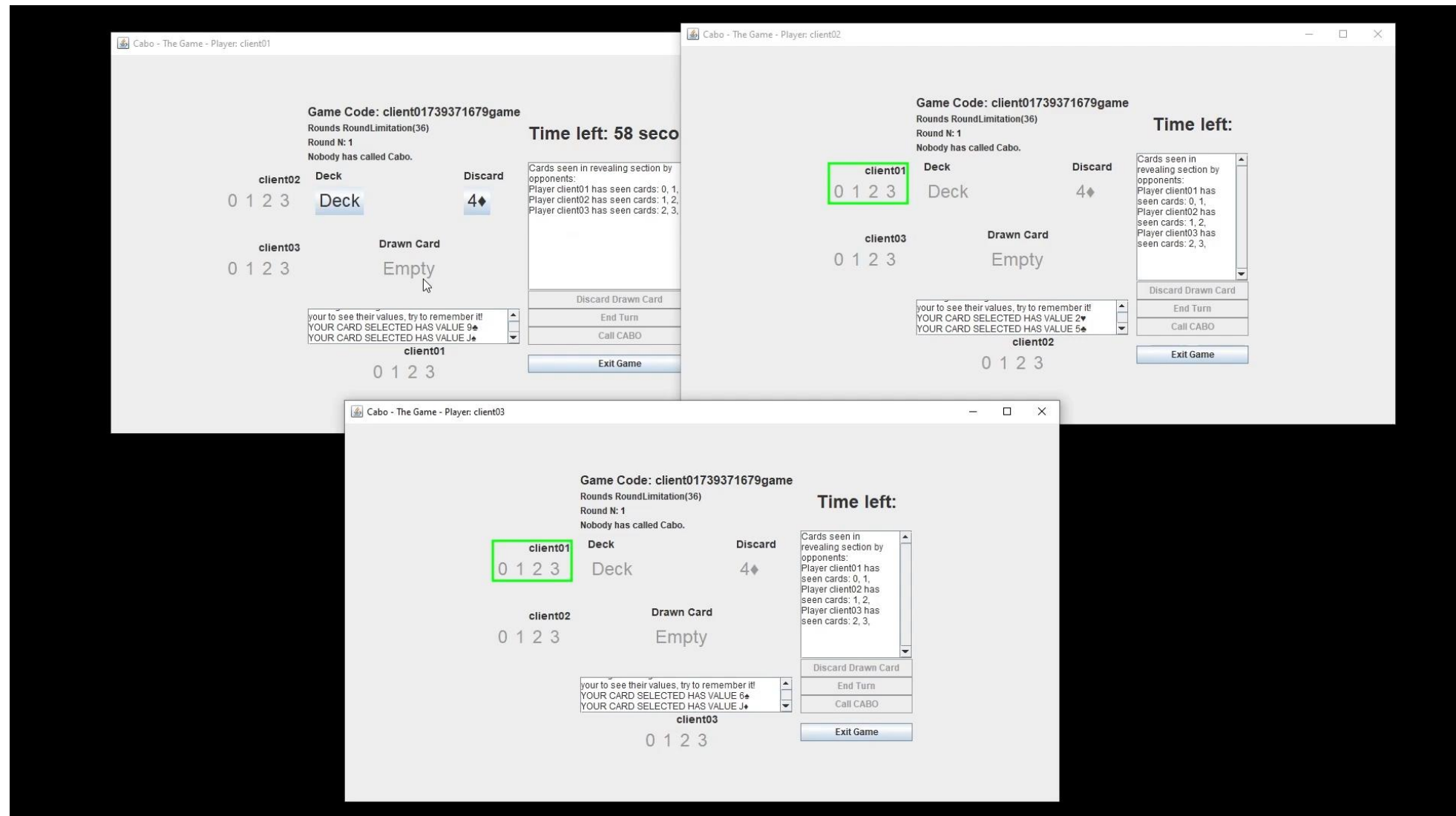
- Unica entità che può cambiare lo stato
- Coordinatore tra i giocatori
- Se abbandona la partita questa viene chiusa



Durante la Partita

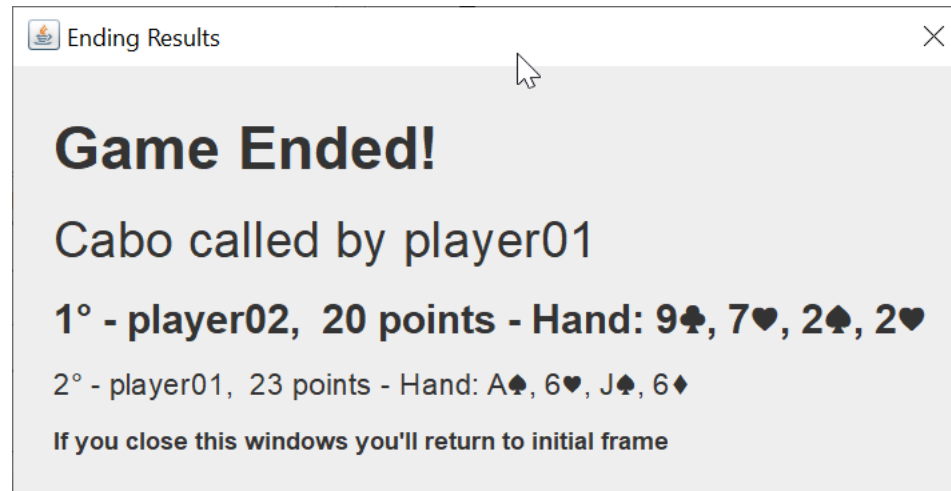
- Lo stato è modificato dal giocatore che gioca il turno
- Ogni giocatore condivide le proprie modifiche con gli altri
- Intervento in caso il giocatore attivo abbandoni la partita
- Se l'host abbandona la partita, ne viene eletto uno nuovo
- Elezione tramite bully algorithm

In Game – Play Cycle



Conclusione partita

- Ultima schermata di gioco in cui viene mostrato il punteggio



- Una volta chiusa si torna alla home per poter cominciare una nuova partita

CABO ONLINE



PROGETTO PER IL CORSO
“SISTEMI DISTRIBUITI”
ANNO ACCADEMICO
2024/2025