Singh S.
Alberghina F.
Battilana A.

5D1

16/05/2023

Progetto Client- multi Server (Finale)

Dama

Analisi e requisiti

Occorre realizzare una versione a linea di comando del **gioco della dama**, in cui due giocatori si connettono ad un unico server di gioco, tramite due client diversi.

Il server gestisce due client che giocano a Dama. Ogni client invia le mosse al server, che verifica se una mossa è valida e aggiorna lo stato del gioco, rimandando così la scacchiera aggiornata ai **due player**.

Il server terrà sempre i due giocatori sincronizzati.

Il gioco finirà quando vengono mangiate tutte le pedine di uno dei due avversari.

Le **richieste** più specifiche sono:

1. I giocatori si devono identificare con un nickname
2. Campo di gioco 8x8
3. Il giocatore che si collega per primo sceglie se utilizzare “X” o “O”
4. Il giocatore indica le coordinate del pezzo da muovere e le coordinate finali, (controllare se la mossa è lecita, altrimenti fare reinserire)
5. Stampare a schermo la scacchiera ed aggiornarla ad ogni mossa
6. Vince chi riesce a mangiare tutte le pedine dell’avversario

Soluzione ideata

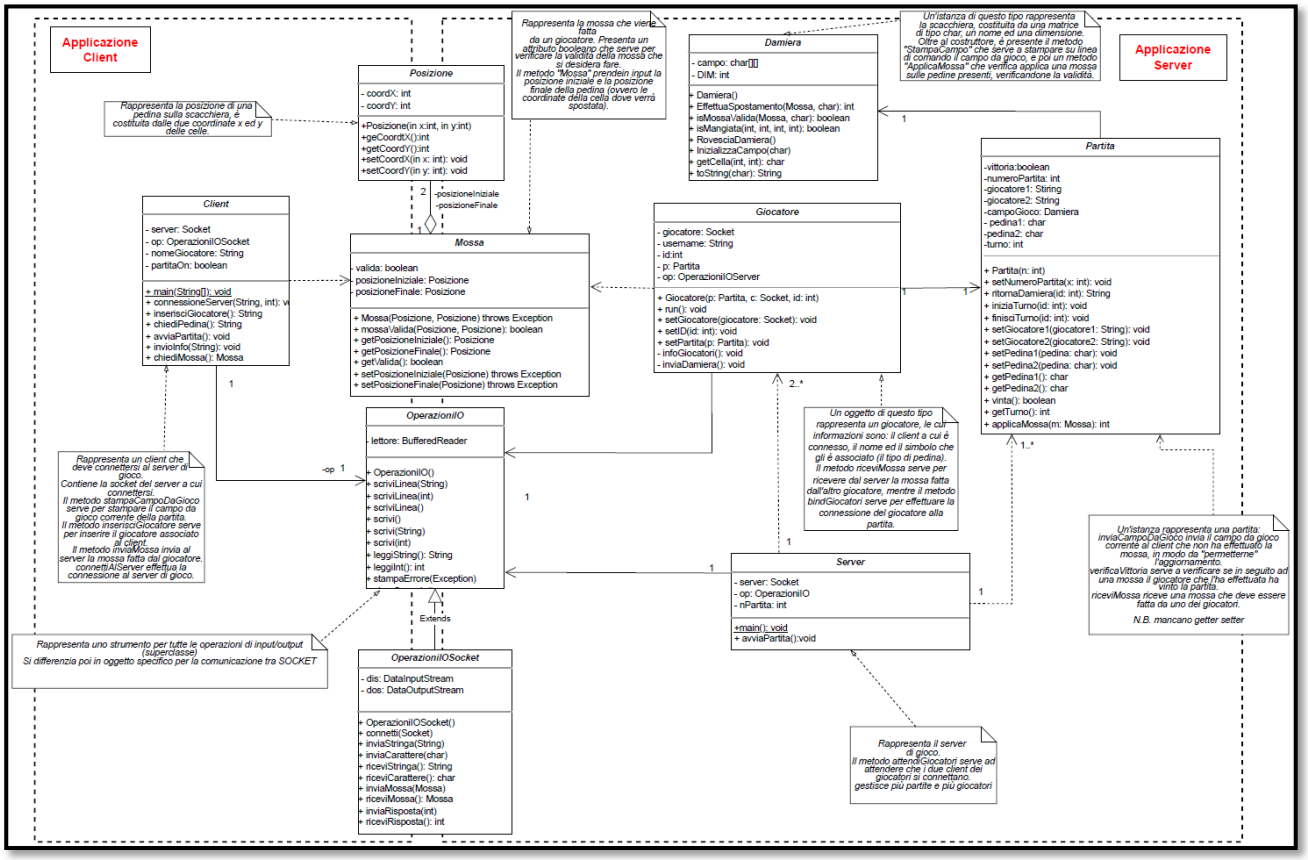
La creazione del gioco si suddivide in **due** parti principali: la realizzazione dell’applicazione server e la realizzazione del lato client. Abbiamo sviluppato 3 package differenti, due con metodi main(durante l’esecuzione ci saranno due applicazioni client distinte ed una applicazione server), ed uno chiamato “Utilità” che conterrà le classi condivise tra i due programmi (come se fossero delle risorse necessarie per l’avvio).

Le applicazioni client e server, avranno delle classi in comune (non tutte) e comunicheranno per la maggior parte del tempo, attraverso la condivisione della mossa (da parte del client) e la condivisione della matrice di gioco aggiornata (da parte del server), ovviamente se la mossa era valida (o almeno parzialmente).

Singh S.
Alberghina F.
Battilana A.

5D1

16/05/2023



UML delle classi

N.B. in allegato la versione completa

Le classi intermedie, poste tra le due applicazioni, saranno presenti nel package "Utilities"

Applicazione DamaClient (utilizza "Utilities")

Ci sarà la classe **Client**, in cui sarà presente un oggetto **OperazioniIOSocket** che oltre a permettere tutte le operazioni di input e output, possiede anche metodi per collegarsi al Server e per gestire chiamate al Server.

Includerà anche il metodo statico main da cui ci si collegherà al server, e attraverso oggetti di tipo **Mossa** (e **Posizione**) si andrà a mandare le dovute scelte del giocatore al Server.

Un' istanza di tipo Mossa**,** che conterrà 2 Posizioni (ogni posizione è caratterizzata da una coordinata x ed una y, con i dovuti metodi getter/setter): la prima posizione riferita alla pedina che si vuole muovere, e la seconda rappresenta la posizione finale dove essa verrà spostata.

Applicazione DamaServer (utilizza "Utilities")

La classe **Server**, attenderà che i due Client si colleghino, creando per ciascuno un'istanza della classe **Giocatore** (rappresenterà un Thread che conterrà come attributi nome, simbolo e socket del client a cui connesso), che a sua volta avrà all'interno un oggetto **OperazioniIOSocket** (per mandare/ricevere messaggi dai singoli giocatori).

La partita (oggetto condiviso tra i due Giocatori):

1) possiede un **CampoDaGioco** (che rappresenta la scacchiera), rappresentato da una matrice di caratteri (8x8), avrà i metodi per eseguire e controllare una mossa (ricevuto un oggetto Mossa),
2) Verificherà lo stato del campo ad ogni mossa
3) userà un oggetto di tipo **Mossa,** che verrà utilizzato per intercettare le mosse dei due giocatori e aggiornare il campo da gioco.
4) disporrà metodi per mandare il campo da gioco aggiornato ai due giocatori, attendere i giocatori, inserire i giocatori

Elenco dei lavori effettivamente svolti

Singh Sukhpreet: si è occupato di creare la classe Server, Giocatore, OperazioniIOSocket, e una parte di Client

Alberghina Francesco: ha creato Damiera, Partita, OperazioniIO, e alcuni metodi di Client

Battilana Alex: ha creato le classi Posizione e Mossa e partecipato al testing dell'applicazione