La configurazione del MVC è quella di un thick client.

Nel lato Client abbiamo sia View che Controller. Il server contiene il model.

Immagine che contiene diagramma

Descrizione generata automaticamente

Il model racchiude lo stato delle partite in corso e l’API utilizzabile dal controller del client. La classe GameManager è un singleton che gestisce le partite in corso. La classe GameLobby rappresenta la lobby pre-gioco. La partita vera e propria è gestita in Game.

I metodi pubblici di GameManager, GameLobby e Game rappresentano tutti e soli i metodi chiamabili dal controller del client. Le risposte alle chiamate del client vengono inviate sotto forma di oggetti di tipo Message.

Le classi del model che non hanno metodi pubblici per i client sono: LivingRoom, CommonGoals (+ strategy), Player, Client, PersonalGoal, Shelf. Essi forniscono il supporto necessario alla classe Game per modificare lo stato del gioco, nonché contenere parte dello stato stesso (es: LivingRoom contiene la matrice con tutte le carte presenti e che tutti possono vedere).

Il controller risponde alle azioni che l’utente fa nella view e chiama i metodi adeguati forniti dal model, inoltre si occupa di leggere i messaggi di aggiornamento forniti dal model e aggiornare la view di conseguenza. In corrispondenza delle chiamate a funzione con il lato Server e quindi con il Model, è presente un sistema di “virtualModel” con cui il Controller effettua le chiamate a funzioni rispettive delle diverse fasi di gioco (GameManager, GameLobby, Game). In generale la parte di network è ancora da stabilire.

Il flusso di gioco di una partita tipo è il seguente: tramite createGame() in GameManager un utente crea una partita, nella lobby si uniscono i vari client e la partita viene iniziata. A questo punto l’inizializzazione di Game prevede la creazione del necessario per rappresentare lo stato, una volta fatto ciò invia il messaggio di InitStateMessage a tutti i client che aggiornano la loro view di conseguenza (soggiorno, libreria, goal comuni e personali, carte selezionabili). Inizia il turno del primo giocatore, il quale sceglie le carte tra quelle selezionabili, il controller ora chiama Game.selectedCards. Il model aggiorna il soggiorno e manda ai giocatori il messaggio SelectedCardsMessage. Analogamente, con la scelta delle colonne e l’ordine delle carte, il controller chiama Game.selectedColumn, avviene perciò la modifica della libreria del giocatore e l’invio del messaggio SelectedColumnsMessage; se la colonna non è disponibile allora viene inviato un messaggio di errore selectedColumnsMessageError. Dopo l’inserimento delle carte, avviene il controllo del completamento degli

obiettivi comuni (CommonGoals) e personali (PersonalGoals), con l’aggiornamento del punteggio, entrambi inviati negli appropriati messaggi.

In caso la libreria del giocatore sia stata riempita con l’ultimo inserimento, la partita continua fino al giocatore con la poltrona escluso. A questo punto viene inviato il messaggio finishedGameMessage dal Model con tutte le informazioni sui risultati della partita (vincitore, punteggi, classifica).