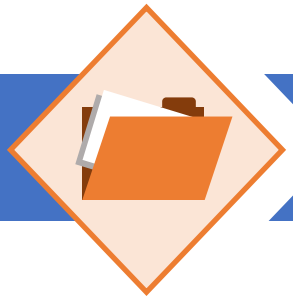
Documentazione Peer Review #1



Documentazione relativa al file Unified Modeling Language (UML) del **Gruppo 04**:
*"**ing-sw-23-saccani-spangaro-sanvito-pedersoli**"*

Di seguito è riportata una breve descrizione generale delle scelte implementative e di design adottate per la realizzazione del diagramma UML Server-Side.

L'UML riguarda il Server-side, in particolare i package "Model", "Controller" e "View" del pattern MVC.
Le parti relative a Model e Controller sono state realizzate completamente; per quanto riguarda la parte della View, questa è presente ma, si riferisce solo ad un primo approccio implementativo.

Controller:
Composto da 2 controller:

- MainController: Gestisce tutte le partite => Possiede N GameController
  (per implementare FA: "Partite Multiple");
- GameController: Gestisce tutti gli aspetti relativi ad una singola partita (Game di GameModel).

Model:

- GameModel: É la parte che contiene tutte le informazioni di gioco quindi permette di collegare tra di loro tutte le altre classi del Model. È lui che si occuperà di notificare le varie view quando si verificheranno delle modifiche del model.
  (Listeners implements interface GameListener)
  (é caratterizzato da un GameStatus (enum): RUNNING, WAIT, ENDED).
- Playground: É il terreno di gioco dove i giocatori recuperano le Tile da posizionare nelle proprie librerie.
- Tile: Singola "piastrella" che i vari giocatori prendono dal playground e posizionano nelle shelf.
  (caratterizzata da enum di tipo PLANT,FRAME,TROPHY,ACTIVITY,CAT,BOOK)
- Shelf: La libreria che ogni giocatore possiede in forma matriciale[][] di Tile
- Player: L'entità che caratterizza i vari player (univoci nella partita, in più partite è possibile usare lo stesso nickname => nickname+id_game = univoci globalmente)
- Point: Valore intero del punto, si riferisce ad una carta per capire da quale carta il Player ha ottenuto il punteggio.
- Message e Chat: Per implementare la FA:"Chat"
  Ogni messaggio appartiene ad una chat ed è stato inviato da un Player
- Package Exception: Per gestire le eccezioni che si possono verificare durante l'esecuzione di metodi (utilizzate dal Controller per prendere decisioni)
- Package Cards: Rappresenta le carte comuni (Package Common) e le carte obiettivo (Package Goal)
  - CommonCard:
    Carte comuni suddivise in 6 sottoclassi ognuna ridefinisce il metodo verify(Shelf) per verificare che l'obiettivo comune sia stato soddisfatto.
  - CardGoal:
    Carte obiettivo personali che controllano che la shelf abbia delle Tile in posizione corretta per restituire un Point (in base al #Tile posizionate correttamente a seconda dell'obiettivo personale)

View:
SocketWelcomeView è il socket principale che accetta tutte le connessioni TCP e si occupa di creare un SocketWithClientView che le gestisce singolarmente (1 connessione per ogni SocketWithClientView).
La SocketWithClientView notifica i 2 controller MainController e GameController:
Notifica il MainController per ottenere un GameController (perché l'utente vuole creare o entrare in una partita e il MainController restituisce il GameController corretto).
Dopo aver ottenuto il GameController, notificherà il GameController di ogni azione che l'utente vuole intraprendere.

(Parte di RMIView non ancora sviluppata)