Università degli Studi di Catania
AA 2024/2025

CardClash

Lorenzo Varsallona Riccardo Maria Villaggio

Prefazione

L'applicazione **CardClash**, che è stata sviluppata nel corso di Ingegneria del Software, è descritta nel seguente documento. L'ambiente di sviluppo Visual Studio Code è stato utilizzato per implementare il programma in Java, mentre Astah è stato impiegato per la progettazione in UML. Soltanto le versioni finali di ogni progetto, realizzate al termine di tutte le fasi di progettazione, saranno mostrate qui.
Le versioni intermedie possono essere consultate nelle corrispondenti sottocartelle della documentazione.
Infine, vengono presentate le conclusioni del test di sistema a seguito del refactoring del codice e la creazione dell'eseguibile dell'applicazione, operazioni che hanno portato a significative migliorie al codice e all'aggiunta di una prima forma di persistenza dei dati.

Indice

1 Ideazione e analisi dei requisiti

Questo capitolo riassume sinteticamente i documenti **Documento di Visione** e **Modello dei Casi d'Uso**.
Il **Documento di Visione** illustra la visione complessiva di **CardClash**, definendo il posizionamento del prodotto, le opportunità di business, i problemi riscontrati nell'organizzazione dei tornei di giochi di carte collezionabili e le funzionalità chiave che l'applicazione intende offrire, quali la gestione dei tornei, l'iscrizione e il tesseramento dei giocatori, la configurazione dei formati di torneo, il tracciamento dei mazzi, la generazione dei tabelloni e il calcolo dei punteggi (inclusi gli aggiornamenti dell'ELO).
Il **Modello dei Casi d'Uso** dettaglia, invece, i requisiti funzionali del sistema attraverso una serie di casi d'uso. Tra questi, si evidenziano operazioni fondamentali quali:

- La creazione del torneo;
- Il tesseramento dei giocatori;
- L'iscrizione ai tornei;
- L'inserimento delle tipologie di mazzo;
- La creazione del tabellone;
- La gestione delle eliminazioni;
- La visualizzazione della classifica;
- L'aggiornamento dei punteggi ELO;
- L'aggiunta di nuovi formati di torneo.

Nota Importante: I documenti sopra citati, così come i diagrammi a cui verrà fatto riferimento nella discussione circa le varie iterazioni, sono consultabili nella loro interezza, insieme al Glossario e alle Regole di Dominio, nella cartella relativa alla fase di ideazione. In questa sede ci limitiamo solo a descrivere il lavoro fatto durante lo sviluppo dell'applicazione. In particolare, nell'Appendice è possibile anche consultare l'ultima versione di Astah sviluppata.

2 Analisi e progettazione orientata agli oggetti

2.1 Iterazione 1

- Focus sui casi d'uso fondamentali: Creazione Torneo (UC1), Tesseramento giocatori (UC2), Iscrizione Torneo (UC3).
- Implementate classi base: Organizzatore, CardClash, Torneo, FormatoTorneo, Giocatore, Mazzo, TipoMazzo.
- Utilizzati pattern GoF Singleton e Façade Controller per CardClash.
- Sviluppati test unitari con JUnit e pianificati test funzionali.
- Stabilita l'architettura di base del sistema.

2.2 Iterazione 2

- Implementati nuovi casi d'uso: Inserimento tipologia mazzo (UC4), Creazione tabellone (UC5), Gestione eliminazione (UC6).
- Introdotte nuove classi: Tabellone, Partita, Enumeration Gioco.
- Implementato pattern Template Method per FormatoTorneo.
- Aggiunti attributi importanti: numeroMaxGiocatori, victoryScore, penaltyScore, elo.
- Raffinamento delle regole di dominio R1, R2, R5, R8, R9.
- Esteso il testing per coprire nuove funzionalità.

2.3 Iterazione 3

- Implementati casi d'uso: Visualizza Classifica (UC7), Aggiorna ELO (UC8), Aggiungi formato torneo (UC9).
- Aggiunta classe FormatoTorneoPersonalizzato.
- Aggiunti attributi su Torneo: "terminato" e "vincitore".
- Estesi test esistenti e aggiunti nuovi test per le nuove funzionalità.
- Modificati diagrammi di sequenza precedenti per supportare nuovi attributi.

2.4 Iterazione 4

- Implementazione di un'interfaccia grafica completa che consente a organizzatori e giocatori di utilizzare tutte le funzionalità previste.
- Revisione dei casi d'uso e aggiornamento dei diagrammi di sequenza.
- Implementazione di numerose estensioni lato codice:

- Consolidamento della persistenza dei dati tramite la classe *PersistenceHandler*, predisponibile per future integrazioni con un DBMS.
- Esecuzione di test di sistema manuali (black-box) che hanno confermato la robustezza e la correttezza dell'interfaccia e del funzionamento complessivo del sistema.

3 Testing

Il processo di testing per **CardClash** è stato strutturato in quattro iterazioni, ognuna delle quali ha contribuito a garantire la qualità del software attraverso diverse fasi di sviluppo. Di seguito viene presentato un riassunto dettagliato delle attività di testing svolte durante ciascuna iterazione.

3.1 Iterazione 1

- **Test unitari**: Verifica della generazione dei codici per i mazzi e del caricamento dei tipi di mazzo per diversi giochi (Magic, Pokémon, Yu-Gi-Oh).
- Focus sulle classi fondamentali: CardClash,FormatoTorneo.

3.2 Iterazione 2

- **Revisione dei test precedenti** per adattarli ai cambiamenti del codice.
- **Nuovi test** concentrati su:
  - Gestione del tabellone: creazione, conferma, visualizzazione, aggiornamento.
  - Sistema di punteggi: inizializzazione e aggiornamento.
  - Gestione delle partite: corretta inizializzazione del numero di partite.
  - Eliminazione giocatori e aggiornamento del tabellone.

3.3 Iterazione 3

- **Aggiornamento dei test esistenti** per riflettere le nuove funzionalità.
- **Nuovi test** focalizzati su:
  - Gestione delle classifiche.
  - Sistema ELO.
  - Gestione della conclusione dei tornei.
  - Creazione e gestione di formati personalizzati.
  - Verifica dei vincitori dei tornei.

3.4 Iterazione 4

- **Introduzione di test di sistema** (black-box testing) dopo l'integrazione della GUI.
- **Test manuali** su:
  - Pannello Creazione Torneo: validazione input, gestione date, conflitti.
  - Pannello Organizzatore: gestione tabellone, eliminazione giocatori, classifiche.
  - Pannello Giocatore: iscrizioni, gestione mazzi, aggiornamento ELO.
- Verifica della gestione degli errori e casi limite.

- Test dell'integrazione tra GUI e logica di business.

In generale, il processo di testing è evoluto da un approccio inizialmente focalizzato sui test unitari verso una strategia più completa che include anche test di integrazione e di sistema, garantendo la qualità del software attraverso tutte le sue fasi di sviluppo.

4 Conclusioni e sviluppi futuri

Il progetto **CardClash** ha raggiunto gli obiettivi fondamentali previsti, dimostrando un'architettura solida e estendibile. L'applicazione fornisce già tutte le funzionalità base per la gestione completa di tornei di carte collezionabili, con un'interfaccia grafica intuitiva e un sistema di persistenza preliminare.

4.1 Transizione a un DBMS

Il passaggio più urgente e naturalmente previsto dal design del sistema è l'adozione di un database relazionale (es. PostgreSQL o MySQL):

- Sostituzione dell'attuale *PersistenceHandler* con un layer ORM (es. Hibernate)
- Migrazione delle entità fondamentali (Torneo, Giocatore, Mazzo) in tabelle relazionali
- Implementazione di transazioni per garantire consistenza durante operazioni complesse
- Ottimizzazione delle query per la gestione di tornei con migliaia di partecipanti

Questo passaggio migliorerebbe drasticamente scalabilità, sicurezza dei dati e performance, oltre a permettere funzionalità avanzate come storico tornei e analisi statistiche.

4.2 Estensioni future

Una volta consolidata la persistenza su DBMS, il sistema potrebbe evolversi in diverse direzioni:

- **Nuovi formati di gioco**: Integrazione con nuovi formati di gioco
- **Sistema ELO avanzato**: Adozione di varianti regionali o specifiche per gioco
- **Personalizzazione tornei**: Supporto per regole custom e formato ibridi
- **App mobile**: Client leggero per consultazione classifiche e iscrizioni
- **Social features**: Sfide tra giocatori, sistemi di amicizie e messaggistica
- **AI integrata**: Suggerimenti per deck-building basati su metagame analisi

La struttura modulare dell'applicazione, combinata con l'uso consapevole di design pattern, rende queste evoluzioni tecnicamente fattibili senza necessità di riscritture massive. CardClash si posiziona così come una piattaforma aperta a continue innovazioni nel panorama sempre dinamico dei trading card game.