

Iterazione 4

1.0 Introduzione

La quarta iterazione si concentrerà sui seguenti aspetti:

- Implementazione di un'interfaccia grafica interattiva che permetta agli organizzatori e ai giocatori di utilizzare l'applicazione in tutti i casi d'uso previsti.
- Piccole modifiche ai casi d'uso relative alle estensioni, sia a livello di codice che di diagramma UML implementati precedentemente.
- Refactoring di tutto il codice dell'applicazione per uniformarsi alla GUI e alle ultime versioni dei casi d'uso

1.1 Aggiornamento dei casi d'uso

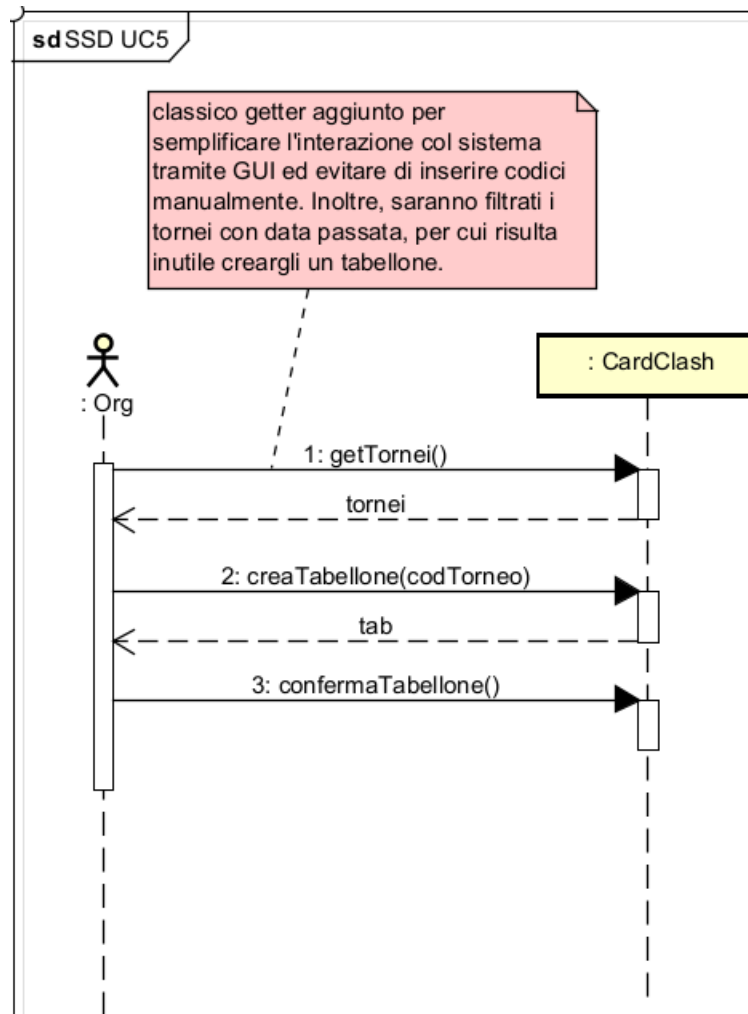
Come per le precedenti iterazioni, durante questa fase sono stati specificati e rivisti dei dettagli dei casi d'uso implementati, in particolare per quanto riguarda le estensioni, per essere in accordo con l'effettivo funzionamento dell'applicazione. Per visionare le modifiche apportate ai suddetti, si consulti il documento “Modello dei casi d'uso” riportato nella cartella in versione aggiornata per questa iterazione.

2.0 Aggiornamento diagrammi di sequenza di sistema

Nella fase di analisi della quarta iterazione, sono state modificati alcuni SSD e SD in vista dell'implementazione della GUI, di seguito le modifiche:

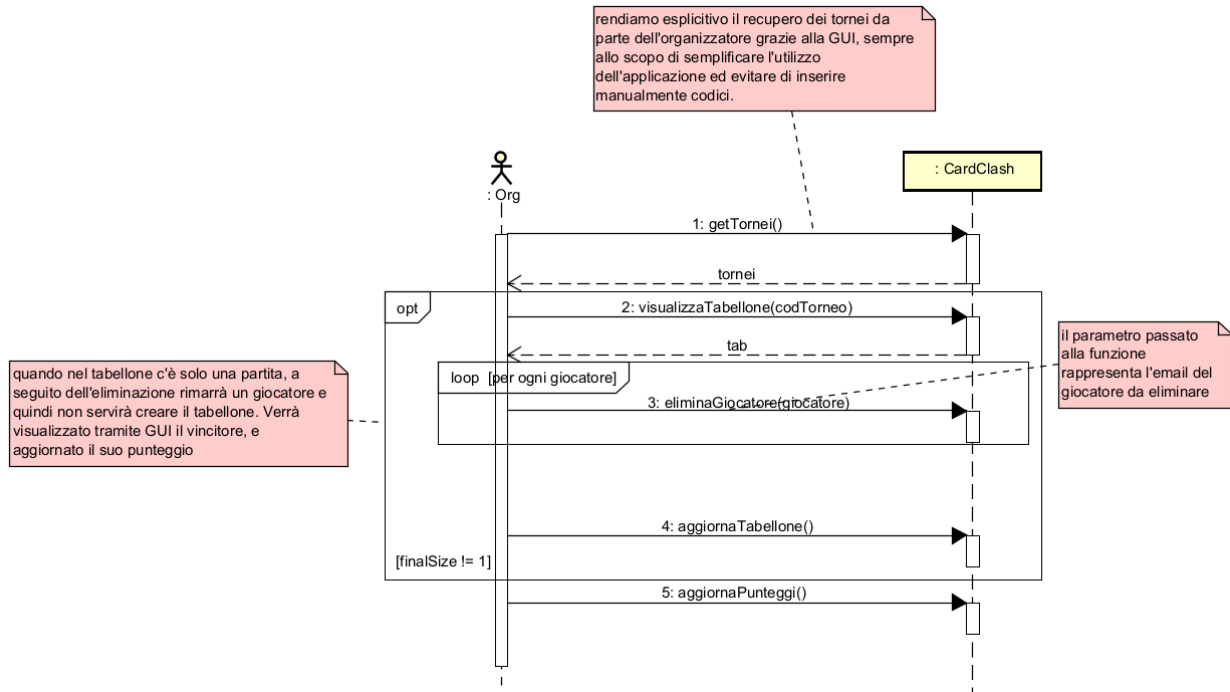
- **UC5:** Creazione tabellone

In questo SSD è stato aggiunto un getter per semplificare l'interazione tra il sistema e la GUI. In particolare, i tornei con data già trascorsa verranno filtrati, mentre quelli selezionabili saranno mostrati a schermo tramite un menu a tendina, evitando così l'inserimento manuale dei codici torneo.



- **UC6:** Gestisci eliminazione

Anche in questo SSD è stato aggiunto un getter. Inoltre, è stato implementato un controllo tramite Guard per consentire l'aggiornamento dei tabelloni e l'eliminazione dei giocatori solo se la dimensione finale della lista dei giocatori nel tabellone è diversa da 1. In caso contrario, l'ultimo giocatore rimasto verrà dichiarato vincitore tramite un popup.

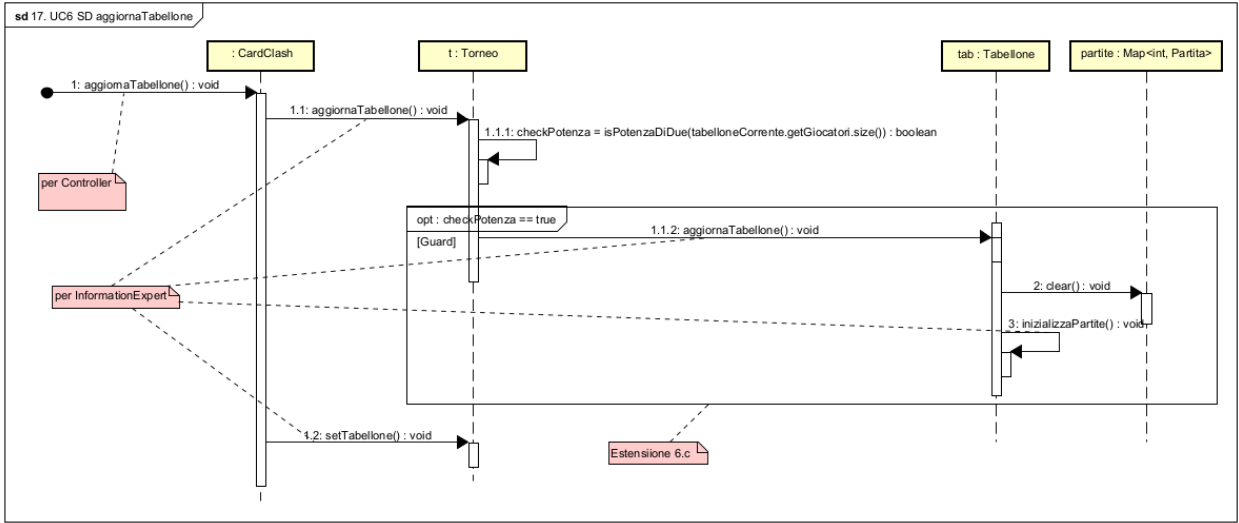


3.0 Fase di Progettazione

Per quanto riguarda la fase di progettazione, sono state effettuate alcune modifiche per quanto riguarda i diagrammi di sequenza per implementare numerose estensioni, di seguito le modifiche:

3.1 Modifica SD aggiornaTabellone(UC6)

Il seguente diagramma è stato aggiornato per controllare che la creazione del nuovo tabellone rispetti sempre la condizione che il numero dei giocatori sia potenza di due, questo per evitare che inavvertitamente si possa eliminare un numero di giocatori inadeguato per l'aggiornamento del tabellone, in accordo con l'estensione 6c.



4.0 Estensioni implementate tramite codice

Numerose estensioni sono state implementate lato codice, e ciò non ha previsto la necessità di aggiornare i diagrammi UML. Le riportiamo di seguito:

- **1b:** la GUI mostra un messaggio di errore se l'amministratore vuole creare un torneo in una data in cui è già previsto un altro.
- **3b:** è stato modificato il funzionamento del metodo `isAperto()`, utilizzata dall'SD `mostraTorneiDisponibili`, in particolare dalla lista dei tornei a cui è possibile iscriversi sono rimossi i tornei a cui si è già iscritti o che hanno già un tabellone (e che quindi sono già in corso)
- **4a e 4b:** queste due estensioni sono semplici controlli del corretto inserimento dei dati, per verificare che la tipologia non sia già esistente.
- **6b:** la GUI mostra un semplice messaggio di errore se il giocatore inserito non è presente nel tabellone.
- **6d:** la GUI mostra un popup di alert, se eliminata l'ultima partita e rimasto l'ultimo giocatore non sia più possibile creare ulteriori tabelloni e che quindi questo sia il vincitore; si può dunque procedere all'aggiornamento dell'ELO.
- **8a e 8b:** estensioni di controllo implementate tramite GUI, per non permettere l'aggiornamento dell'ELO qualora il torneo non sia ancora iniziato o non sia ancora finito.

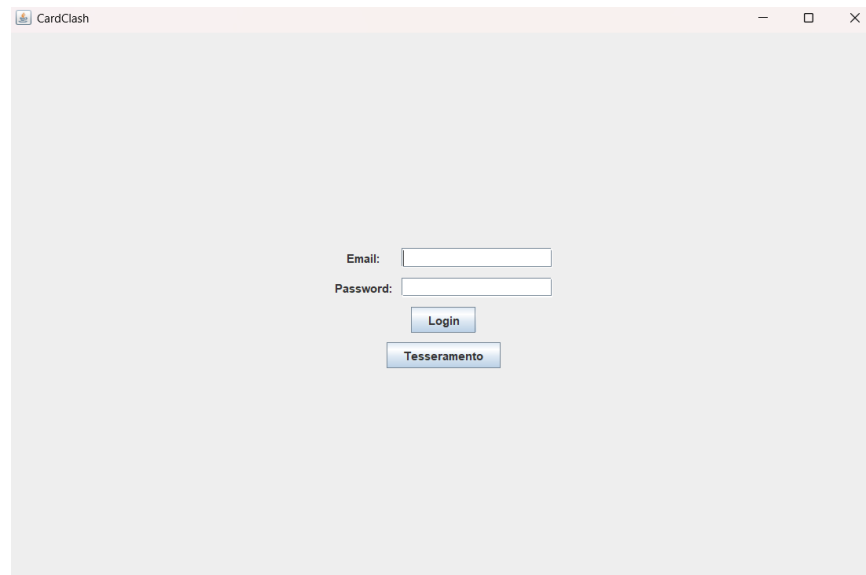
In aggiunta a ciò, sono state gestite numerose funzioni di rollback dei dati in caso di problematiche di utilizzo dell'applicazione a runtime.

Infine, l'organizzatore può liberamente eliminare dal torneo anche due giocatori che partecipavano alla stessa partita: questo potrebbe accadere nel caso di ritiri/squalifiche. In future iterazioni, l'estensione 6a (al momento, dunque, non totalmente implementata) si potrebbe modellare come caso d'uso/scenario alternativo separato, e tramite un nuovo elemento della GUI sottrarre punti ai giocatori, in accordo con le Regole di Dominio e del formato stesso.

5.0 Interfaccia grafica (GUI)

In questa sezione riportiamo degli screenshot dell'interfaccia dell'applicazione in funzione.

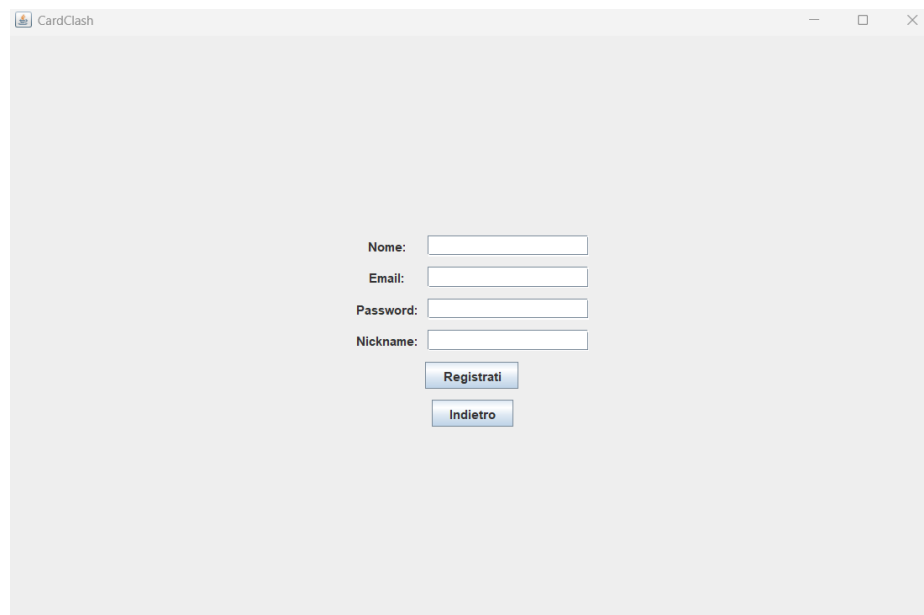
5.1 Login



The screenshot shows a window titled "CardClash" with a light gray background. In the center, there is a login form with the following elements:

- An "Email:" label followed by a text input field.
- A "Password:" label followed by a text input field.
- A "Login" button below the password field.
- A "Tesseramento" button below the "Login" button.

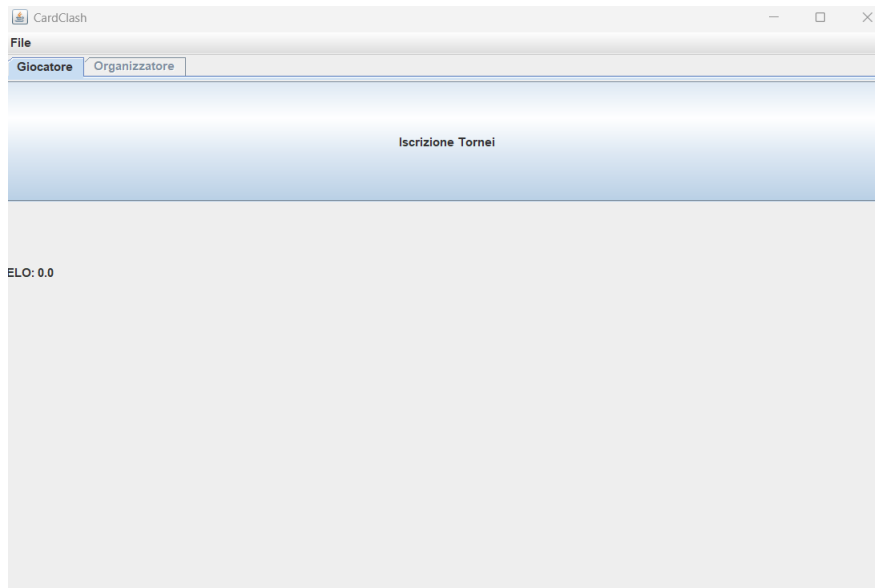
5.2 Tesseramento



The screenshot shows a window titled "CardClash" with a light gray background. In the center, there is a registration form with the following elements:

- A "Nome:" label followed by a text input field.
- An "Email:" label followed by a text input field.
- A "Password:" label followed by a text input field.
- A "Nickname:" label followed by a text input field.
- A "Registrati" button below the nickname field.
- An "Indietro" button below the "Registrati" button.

5.3 Interfaccia lato giocatore



5.4 Interfaccia lato organizzatore



5.5 Form di creazione di un torneo

CardClash

Nome Torneo:

Data (yyyy-MM-dd):

Orario (HH:mm):

Luogo:

5.6 Tabellone di un torneo

Tabellone Generato

Tabellone (non confermato)

| codPartita | Giocatore1 | Punti G1 | Giocatore2 | Punti G2 |
|------------|------------------|----------|-------------------|----------|
| 227734 | ciao2@gmail.c... | 0.0 | ciao1@gmail.c... | 0.0 |
| 557810 | ciao@gmail.co... | 0.0 | ricky@gmail.co... | 0.0 |

OK

5.7 Tabellone aggiornato dopo un'eliminazione

Tabellone Iniziale

Tabellone (789077)

| codPartita | Giocatore1 | Punti G1 | Giocatore2 | Punti G2 |
|------------|------------------|----------|------------------|----------|
| 441339 | ciao1@gmail.c... | 2.0 | ciao@gmail.co... | 2.0 |

OK

5.8 Classifica

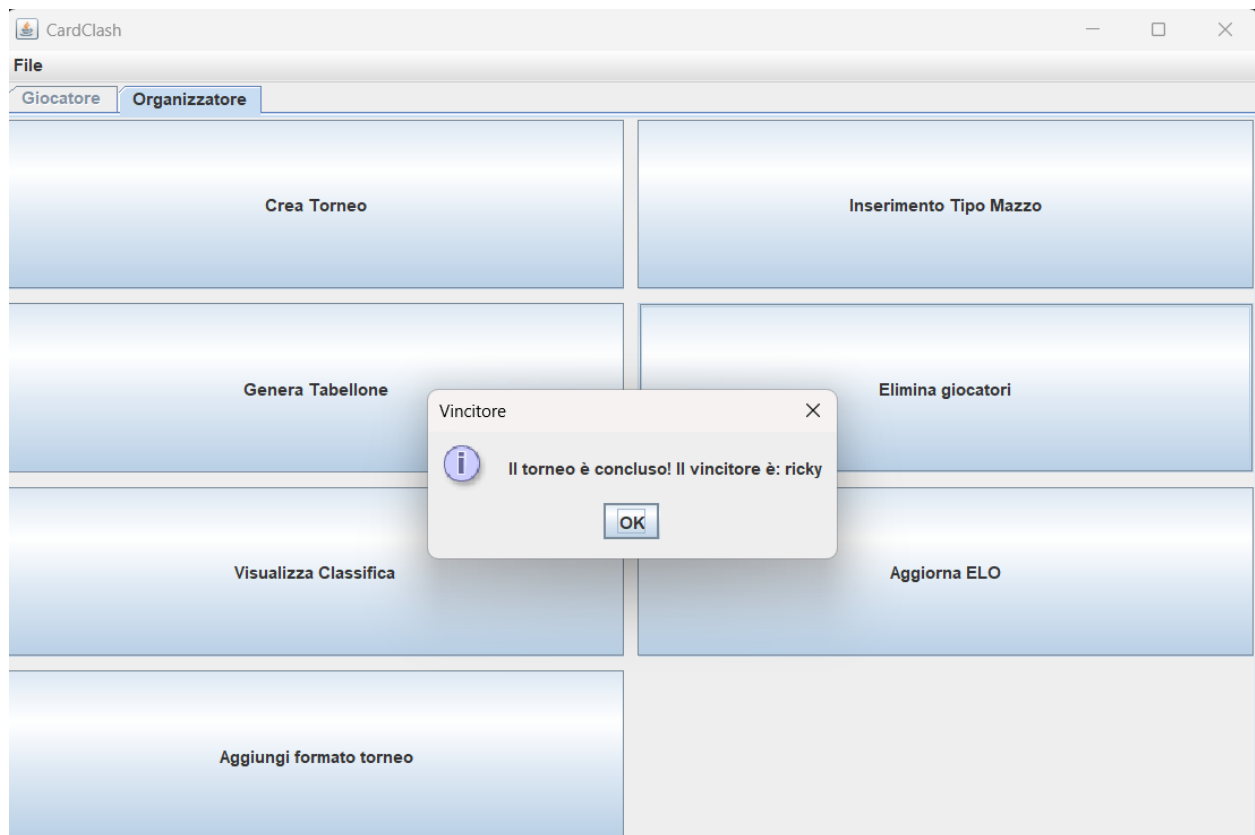
Classifica

Classifica

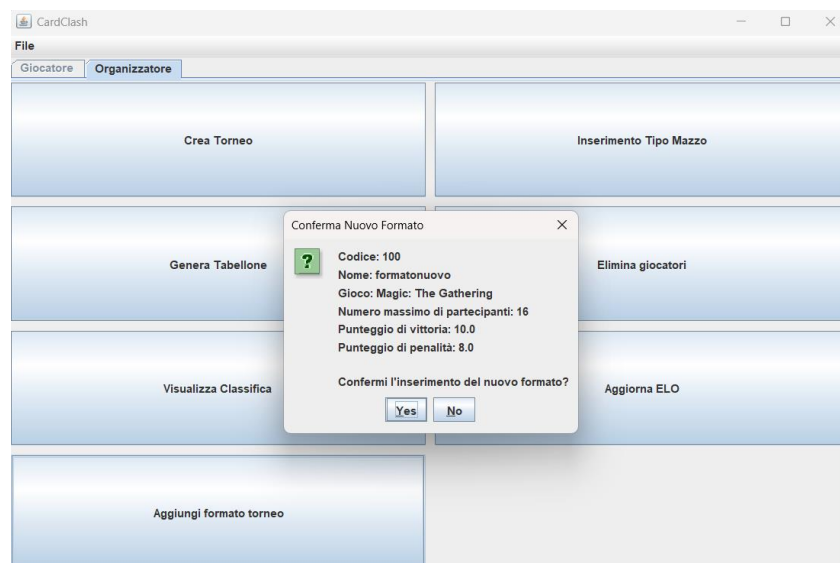
| Posizione | Email | Nickname | Punteggio |
|-----------|-----------------|----------|-----------|
| 1 | ciao1@gmail.com | asfasf | 2.0 |
| 2 | ciao@gmail.com | ASFASF | 2.0 |
| 3 | ciao2@gmail.com | asfasf | 0.0 |
| 4 | ricky@gmail.com | ricky | 0.0 |

OK

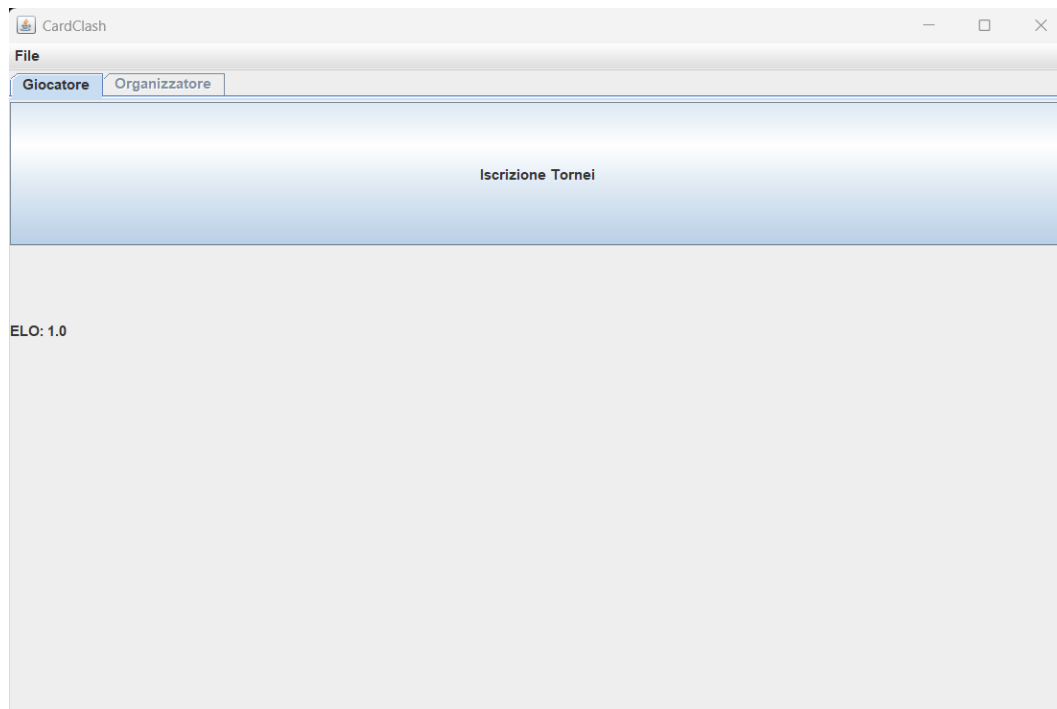
5.9 Popup del vincitore, dopo l'ultima eliminazione



5.10 Popup della conferma di creazione di un formato torneo



5.11 Interfaccia giocatore aggiornata dopo la partecipazione ad un torneo



6.0 Persistenza dei dati

In questa iterazione è stata implementata una classe apposita per la gestione della persistenza dei dati: `PersistenceHandler`. Si è voluta delegare la responsabilità della gestione del DOM di un file di persistenza xml, a questa classe, in accordo con il pattern Grasp Pure Fabrication. Dunque, essa non rappresenta un oggetto del modello di Dominio, ma è una classe utilizzata unicamente dalla GUI per caricare e salvare dati. Chiaramente si tratta di una persistenza embrionale e che non si appoggia ad un DBMS, ciò nonostante, essa è facilmente integrabile in versioni future dell'applicazione, tramite un refactoring unicamente rivolto a questa classe, senza intaccare altri componenti dell'applicazione, motivo per cui si è scelto di utilizzare questo pattern.