

UNIVERSITÀ DEGLI STUDI DI NAPOLI FEDERICO II
CORSO DI LAUREA IN INGEGNERIA INFORMATICA
CORSO DI INGEGNERIA DEL SOFTWARE
PROF. S. RUSSO - A.A. 2021 - 22

Progetto
Gestione tornei frecce

Gruppo: Sigma

Studenti:

Alessandro Parisi	N46002820
Andrea Palumbo	N46004669
Massimo Marrone	N46006784

alessand.parisi@studenti.unina.it
andrea.palumbo7@studenti.unina.it
mass.marrone@studenti.unina.it

Versione 1 del 16/06/2022

Indice

1. SPECIFICHE INFORMALI.....	1
2. ANALISI E SPECIFICA DEI REQUISITI	2
2.1 ANALISI NOMI-VERBI	2
2.2 REVISIONE DEI REQUISITI.....	2
2.3 GLOSSARIO DEI TERMINI.....	3
2.4 CLASSIFICAZIONE DEI REQUISITI	4
2.4.1 Requisiti funzionali	4
2.4.2 Requisiti sui dati.....	5
2.4.3 Vincoli / Altri requisiti.....	5
2.5 MODELLAZIONE DEI CASI D'USO	6
2.5.1 Attori e casi d'uso.....	6
2.5.2 Diagramma dei casi d'uso	8
2.5.3 Scenari.....	8
2.6 MODELLAZIONE DEI DATI.....	11
2.6.1 Progettazione concettuale	11
2.7 DIAGRAMMA DELLE CLASSI.....	12
2.8 DIAGRAMMI DI SEQUENZA.....	14
2.9 VERIFICA DELLA COMPLETEZZA DEI REQUISITI.....	16
3. STIMA DEI COSTI	17
4. PIANO DI TEST FUNZIONALE.....	19
5. PROGETTAZIONE	26
5.1 PROGETTAZIONE DELLA BASE DI DATI.....	26
5.1.1 Progettazione logica	26
5.2 DIAGRAMMA DELLE CLASSI.....	28
5.3 DIAGRAMMI DI SEQUENZA.....	29
6. IMPLEMENTAZIONE	31
7. TESTING	33
7.1 TEST STRUTTURALE	33
7.1.1 Complessità ciclomatica.....	33
7.1.2 Test di unità	37
7.2 TEST FUNZIONALE.....	38

1. Specifiche informali

T28: Gestione tornei frecce

Si vuole realizzare un sistema software per la gestione dei tornei della federazione professionale di frecce (FPF).

Il gestore del sistema - un funzionario della FPF - inserisce nel sistema i dati degli organizzatori dei tornei, degli giocatori e degli arbitri (nome, cognome, numero di tessera FPF ed indirizzo email. All'atto dell'inserimento, l'organizzatore, l'arbitro o l'atleta inserito riceve le credenziali di accesso (username/password), automaticamente generate.

Gli organizzatori inseriscono/modificano nel sistema i tornei FPF. I tornei sono caratterizzati da denominazione, edizione (anno), sede (città), data di inizio e fine, un numero massimo di partecipanti (una potenza di due), una lista degli sponsor e un montepremi in denaro (in dollari). Quando un organizzatore inserisce o modifica un torneo, la richiesta deve essere approvata da un funzionario della FPF; il funzionario assegna 4 arbitri al torneo. Il montepremi è così suddiviso: 3/8 al vincitore, 1/4 al finalista, 3/16 ai semifinalisti. Per ogni atleta il sistema deve aggiornare dopo ogni torneo anche il totale dei premi in denaro vinti. I giocatori registrati possono iscriversi a un torneo, specificando il set di frecce con cui parteciperanno. Un set di frecce è caratterizzato dalla marca, dal modello e dal peso (in grammi).

La richiesta di partecipazione deve essere gestita dall'organizzatore del torneo selezionato, il quale può decidere se accettare o rifiutare la richiesta di iscrizione (in caso di rifiuto, viene specificata la motivazione, per es. perché si è già raggiunto il numero massimo di partecipanti).

A seguito della decisione dell'organizzatore, l'atleta riceve una email contenente l'esito della registrazione.

Una settimana prima dell'inizio di un torneo, il sistema genera in maniera automatica il tabellone degli incontri da disputare, che viene inviato a tutti gli iscritti al torneo. Ogni match si svolge tra due giocatori in un giorno e orario fissati. L'organizzatore assegna un arbitro (tra i 4 assegnati dalla federazione al torneo) per ogni match in tabellone. Gli arbitri registrano il punteggio finale di ogni incontro: a seguito di tale operazione, il sistema inserisce automaticamente il vincitore come partecipante all'incontro successivo nel tabellone predefinito. Il giorno dopo la conclusione di un torneo, il sistema aggiorna il ranking degli atleti della FPF.

La classifica viene generata in base al totale dei premi in denaro vinti da ogni atleta.

Gli sportivi o tifosi possono consultare il sistema (senza essere registrati) per vedere i tornei, i tabelloni e i risultati degli incontri.

2. Analisi e specifica dei requisiti

2.1 Analisi nomi-verbi

T28: Gestione tornei freccette

Si vuole realizzare un sistema software per la gestione dei tornei della federazione professionale di freccette (FPF).

Il gestore del sistema - un funzionario della FPF - inserisce nel sistema i dati degli organizzatori dei tornei, degli giocatori e degli arbitri (nome, cognome, numero di tessera FPF ed indirizzo email).

All'atto dell'inserimento, l'organizzatore, l'arbitro o l'atleta inserito riceve le credenziali di accesso (username/password), automaticamente generate.

Gli organizzatori inseriscono/modificano nel sistema i tornei FPF. I tornei sono caratterizzati

da denominazione, edizione (anno), sede (città), data di inizio e fine, un numero massimo di

partecipanti (una potenza di due), una lista degli sponsor e un montepremi in denaro (in dollari). Quando un organizzatore inserisce o modifica un torneo, la richiesta deve essere approvata da un funzionario della FPF; il funzionario assegna 4 arbitri al torneo. Il

montepremi è così suddiviso: 3/8 al vincitore, 1/4 al finalista, 3/16 ai semifinalisti. Per ogni atleta il sistema deve aggiornare dopo ogni torneo anche il totale dei premi in denaro vinti.

I giocatori registrati possono iscriversi a un torneo, specificando il set di freccette con cui parteciperanno. Un set di freccette è caratterizzato dalla marca, dal modello e dal peso (in grammi).

La richiesta di partecipazione deve essere gestita dall'organizzatore del torneo selezionato,

il quale può decidere se accettare o rifiutare la richiesta di iscrizione (in caso di rifiuto, viene specificata la motivazione, per es. perchè si è già raggiunto il numero massimo di partecipanti).

A seguito della decisione dell'organizzatore, l'atleta riceve una email contenente l'esito della registrazione.

Una settimana prima dell'inizio di un torneo, il sistema genera in maniera automatica il tabellone degli incontri da disputare,

che viene inviato a tutti gli iscritti al torneo. Ogni match si svolge tra due giocatori in un giorno e orario fissati. L'organizzatore assegna un arbitro (tra i 4 assegnati dalla federazione al torneo) per ogni match in tabellone. Gli arbitri registrano il punteggio finale di ogni incontro: a seguito di tale operazione, il sistema inserisce automaticamente il vincitore come partecipante all'incontro successivo nel tabellone predefinito. Il giorno dopo la conclusione di un torneo, il sistema aggiorna il ranking degli atleti della FPF.

La classifica viene generata in base al totale dei premi in denaro vinti da ogni atleta.

Gli sportivi o tifosi possono consultare il sistema (senza essere registrati) per vedere i tornei, i tabelloni e i risultati degli incontri.

LEGENDA:

Classe

Attributo

Funzionalità

Attore

Classe-Attore

2.2 Revisione dei requisiti

1. Il sistema deve offrire al Funzionario della FPF una funzionalità per inserire: nome, cognome, numero di tessera FPF, indirizzo email di un Organizzatore di tornei, Giocatore o Arbitri.
2. Il sistema deve generare automaticamente username/password diverso per ogni Giocatore, Arbitro, Organizzatore al momento dell'inserimento dei dati da parte del Funzionario FPF.
3. Il sistema deve inviare via email ai Giocatori, Arbitri, Organizzatori i rispettivi username/password generati.
4. Di ogni Organizzatore si vuole memorizzare nome, cognome, numero di tessera FPF, indirizzo email, username, password.
5. Di ogni Arbitro si vuole memorizzare nome, cognome, numero di tessera FPF, indirizzo email, username, password.
6. Di ogni Giocatore si vuole memorizzare nome, cognome, numero di tessera FPF, indirizzo email, username, password, totale montepremi vinto.
7. Il sistema deve offrire all'Organizzatore una funzionalità per Inserire un torneo.
8. Il sistema deve offrire all'Organizzatore una funzionalità per Modificare un torneo.
9. Di un Torneo si vuole memorizzare: denominazione, edizione, sede, data di inizio, data di fine, numero massimo di partecipanti, lista di sponsor, montepremi.

10. Il sistema deve offrire al Funzionario della FPF una funzionalità per approvare o rifiutare modifiche/inserimenti dei tornei da parte di un Organizzatore.
11. Il sistema deve offrire al Funzionario della FPF una funzionalità per assegnare 4 arbitri dopo la creazione di un torneo torneo.
12. Il sistema deve suddividere automaticamente il montepremi in: 3/8 al vincitore, 1/4 al finalista, 3/16 ai semifinalisti.
13. Il sistema deve aggiornare automaticamente il totale dei premi in denaro vinti per ogni Giocatore partecipante dopo ogni torneo.
14. Il sistema deve offrire al Giocatore una funzionalità per iscriversi al torneo, specificando il set di frecce con cui parteciperà.
15. Di ogni set di frecce si vuole memorizzare: marca, modello, peso.
16. Il sistema deve offrire all'Organizzatore una funzionalità per accettare/rifiutare la richiesta di iscrizione al torneo.
17. Il sistema deve inviare ai Giocatori via email l'esito di richiesta di iscrizione
18. Una settimana prima dell'inizio di un torneo, il sistema deve generare automaticamente il tabellone dei match.
19. Il tabellone dei match viene inviato via email ai Giocatori partecipanti di un torneo.
20. Di ogni match si vuole memorizzare: giorno, orario, Giocatori, risultato, Arbitro.
21. Il sistema deve offrire all'Organizzatore una funzionalità per assegnare un Arbitro (tra i 4 assegnati dalla federazione al torneo) per ogni match in tabellone.
22. Il sistema deve offrire all'Arbitro una funzionalità per registrare il risultato di ogni match.
23. Il sistema deve inserire automaticamente il vincitore di un match al match successivo del tabellone.
24. Il sistema, il giorno dopo la conclusione di un torneo, deve aggiornare il ranking dei Giocatori in base al totale dei premi in denaro vinti.
25. Il sistema deve permettere a chiunque di visualizzare tornei, tabelloni, ranking, risultati di match.
26. Il sistema deve permettere all'Organizzatore di descrivere la motivazione in caso di rifiuto di una richiesta d'iscrizione.

2.3 Glossario dei termini

Termino	Descrizione	Sinonimi
Funzionario della FPF	Utente autorizzato nel sistema che ha accesso a funzionalità che gestiscono utenti registrati.	Funzionario
Giocatore	Utente registrato nel sistema ed eventuale partecipante di tornei.	Atleta Partecipante
Organizzatore	Utente registrato nel sistema che può creare/modificare un torneo, accettare partecipanti dei tornei, assegnare un arbitro per ogni match di un torneo.	
Ranking	Lista ordinata di Giocatori per totale dei premi in denaro vinti.	Classifica
Arbitro	Utente registrato nel sistema che può registrare il risultato del match assegnatogli.	
Sportivi	Utenti non registrati o autorizzati che utilizzano il sistema.	Tifosi Utente non autenticato

Match	Incontro sportivo programmato in un tabellone con 2 Giocatori partecipanti.	Incontro
Tabellone	Raccolta di informazioni di un torneo e della sua lista di match pubblica.	
Risultato	Due valori assegnati al match basatosi sul punteggio ottenuto dai partecipanti dello stesso.	

2.4 Classificazione dei requisiti

2.4.1 Requisiti funzionali

ID	Requisito	Origine (n. frase dei requisiti revisionati)
RF01	Il sistema deve offrire al Funzionario della FPF una funzionalità per inserire: nome, cognome, numero di tessera FPF, indirizzo email di un Organizzatore, Giocatori o Arbitri.	1
RF02	Il sistema deve generare automaticamente username/password diverso per ogni Giocatore, Arbitro, Organizzatore al momento dell'inserimento dei dati da parte del Funzionario FPF.	2
RF03	Il sistema deve inviare via email ai Giocatori, Arbitri, Organizzatori i rispettivi username/password generati.	3
RF04	Il sistema deve offrire all' Organizzatore una funzionalità per inserire un torneo.	7,8
RF05	Il sisteva deve offrire all'Organizzatore una funzionalità per modificare un torneo esistente.	
RF06	Il sistema deve offrire al Funzionario della FPF una funzionalità per approvare modifiche o inserimenti dei tornei da parte di un Organizzatore.	10
RF07	Il sistema deve offrire al Funzionario della FPF una funzionalità per assegnare 4 arbitri ad un nuovo torneo.	11
RF08	Il giorno dopo un torneo sistema deve aggiornare il totale dei premi in denaro vinti per ogni Giocatore partecipante.	13
RF09	Il sistema deve offrire al Giocatore una funzionalità per iscriversi ad un torneo, specificando il set di frecce con cui parteciperà.	14
RF10	Il sistema deve offrire all' Organizzatore una funzionalità per accettare la richiesta di iscrizione al torneo.	16
RF11	Il sistema deve inviare ai Giocatori via email l'esito di richiesta di iscrizione	17
RF12	Una settimana prima dell'inizio di un torneo, il sistema deve generare automaticamente il tabellone dei match.	18
RF13	Il tabellone dei match viene inviato via email ai Giocatori partecipanti di un torneo.	19
RF14	Il sistema deve offrire all' Organizzatore una funzionalità per assegnare un Arbitro (tra i 4 assegnati dalla federazione al torneo) per ogni match in tabellone.	21

RF15	Il sistema deve offrire all' Arbitro una funzionalità per registrare il risultato di ogni match.	22
RF16	Il sistema deve inserire automaticamente il vincitore di un match al match successivo del tabellone.	23
RF17	Il sistema, il giorno dopo la conclusione di un torneo, deve aggiornare il ranking dei Giocatori in base al totale dei premi in denaro vinti.	24
RF18	Il sistema deve permettere a chiunque di visualizzare: tornei, tabelloni, ranking o risultati di match.	25
RF19	Il sistema deve permettere all'Organizzatore di descrivere la motivazione in caso di rifiuto di una richiesta d'iscrizione.	26

2.4.2 Requisiti sui dati

ID	Requisito	Origine (n. frase dei requisiti revisionati)
RD01	Di ogni Organizzatore si vuole memorizzare nome, cognome, numero di tessera FPF, indirizzo email, username, password.	4
RD02	Di ogni Arbitro si vuole memorizzare nome, cognome, numero di tessera FPF, indirizzo email, username, password.	5
RD03	Di ogni Giocatore si vuole memorizzare nome, cognome, numero di tessera FPF, indirizzo email, username, password, totale montepremi vinto.	6
RD04	Di ogni Torneo si vuole memorizzare: denominazione, edizione, sede, data di inizio, data di fine, numero massimo di partecipanti, lista di sponsor, montepremi.	9
RD05	Di ogni set di frecette si vuole memorizzare: marca, modello, peso.	15
RD06	Di ogni match si vuole memorizzare: giorno, orario, Giocatori, risultato, Arbitro.	20

2.4.3 Vincoli / Altri requisiti

ID	Requisito	Origine (n. frase dei requisiti revisionati)
Vo1	In un torneo il numero di Giocatori deve essere una potenza di due	
Vo2	La valuta del montepremi deve essere espressa in dollari	
Vo3	Il montepremi deve essere suddiviso in 3/8 al vincitore del torneo, 1/4 per il finalista, 3/16 per i semifinalisti.	12
Vo4	Il numero di Arbitri assegnati per ogni torneo deve essere 4	

RNF01	Per l'invio di credenziali di accesso, esito di iscrizione ad un torneo e la comunicazione dei match del tabellone deve essere disponibile un server di posta elettronica esterno al sistema.	
-------	---	--

2.5 Modellazione dei casi d'uso

2.5.1 Attori e casi d'uso

Attori Primari:

- Funzionario della FPF
- Organizzatore
- Giocatore
- Tempo
- Sportivo
- Arbitro

Attori Secondari:

- Servizio Email

Casi d'uso:

- UC1: RegistraUtente
- UC2: CreaTorneo
- UC3: ModificaTorneo
- UC4: AggiornaTotalePremi
- UC5: IscriviTorneo
- UC6: GeneraTabellone
- UC7: AssegnaArbitroMatch
- UC8: RegistraRisultatoMatch
- UC9: AggiornaRanking
- UC10: CercaMatch
- UC11: VisualizzaRanking
- UC12: CercaTorneo
- UC13: AssegnaArbitri
- UC14: ApprovaTorneo
- UC15: Approvalscrezione
- UC16: VisualizzaTabellone

Casi d' uso di inclusione:

- UC12: CercaTorneo
- UC17: InviaMail

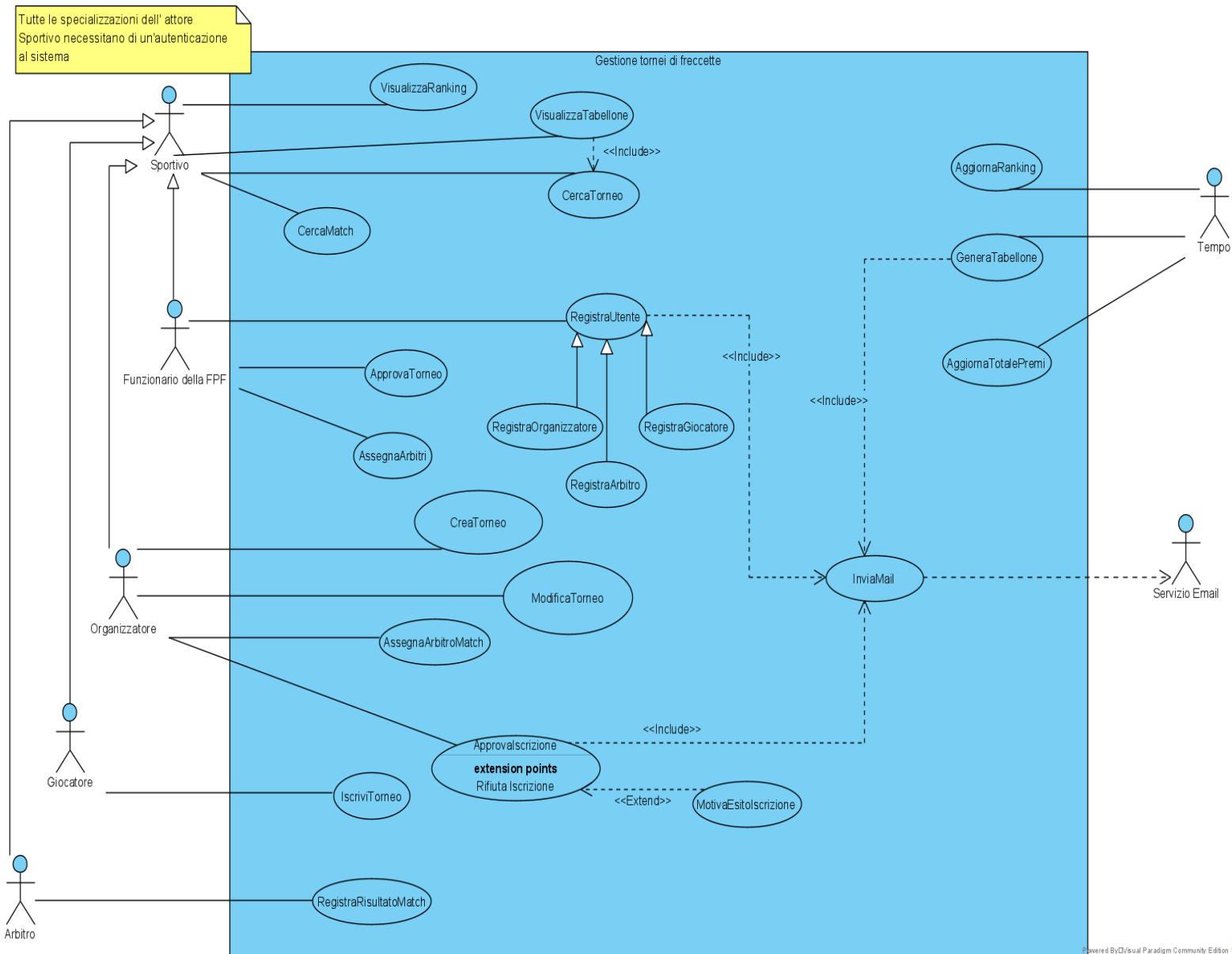
Casi d' uso di estensione:

- UC18: MotivaEsitoIscrizione

Caso d'uso	Attori Primari	Attori Secondari	Incl. / Ext.
UC1: RegistraUtente	Funzionario della FPF	-	Include InviaMail
UC2: CreaTorneo	Organizzatore	-	-
UC3: ModificaTorneo	Organizzatore	-	-

UC4: AggiornaTotalePremi	Tempo	-	-
UC5: IscriviTorneo	Giocatore	-	-
UC6: GeneraTabellone	Tempo	-	Include InviaMail
UC7: AssegnaArbitroMatch	Organizzatore	-	-
UC8: RegistraRisultatoMatch	Arbitro	-	-
UC9: AggiornaRanking	Tempo	-	-
UC10: CercaMatch	Sportivo	-	-
UC11: VisualizzaRanking	Sportivo	-	-
UC12: CercaTorneo	Sportivo	-	Include VisualizzaTabellone
UC13: AssegnaArbitri	Funzionario della FPF	-	-
UC14: ApprovaTorneo	Funzionario della FPF	-	-
UC15: ApprovaIscrizione	Organizzatore	-	-
UC16: VisualizzaTabellone	-	-	Include CercaTorneo
UC17: InviaMail	-	Servizio Email	Incluso in RegistraUtente GeneraTabellone e ApprovaIscrizione
UC18: MotivaEsitoIscrizione	-	-	Estensione di ApprovaIscrizione

2.5.2 Diagramma dei casi d'uso



2.5.3 Scenari

Caso d'uso:	GeneraTabellone
Attore primario	Tempo
Attore secondario	-
Descrizione	Sette giorni prima dell'inizio di un torneo, viene generato un tabellone dei match di quei tornei che verranno inviati a tutti i Giocatori partecipanti.
Pre-Condizioni	Esiste almeno un torneo che ha raggiunto il massimo numero di partecipanti nel sistema del quale non è stato ancora generato il tabellone.
Sequenza di eventi principale	<ol style="list-style-type: none"> Il caso d'uso inizia quando l'attore Tempo inserisce la data odierna. Per ogni torneo la cui data di inizio è sette giorni susseguente alla data odierna il sistema assegnerà i partecipanti ai turni iniziali dei match di quel torneo. Il sistema genera un tabellone contenente: edizione, nome e la lista dei match ordinata del torneo.

	<ol style="list-style-type: none"> 4. Il sistema ricerca l'email di ogni Giocatore presente nei match del tabellone creato. 5. Il sistema utilizza un servizio email per inviare il tabellone ad ogni Giocatore utilizzando la lista di mail ricercata.
Post-Condizioni	Sono stati assegnati i partecipanti ai turni iniziali dei match dei tornei ed è stata inviata a tutti i Giocatori presenti in esso un tabellone tramite servizio mail.
Casi d'uso correlati	<i>InviaMail</i>
Sequenza di eventi alternativi	-

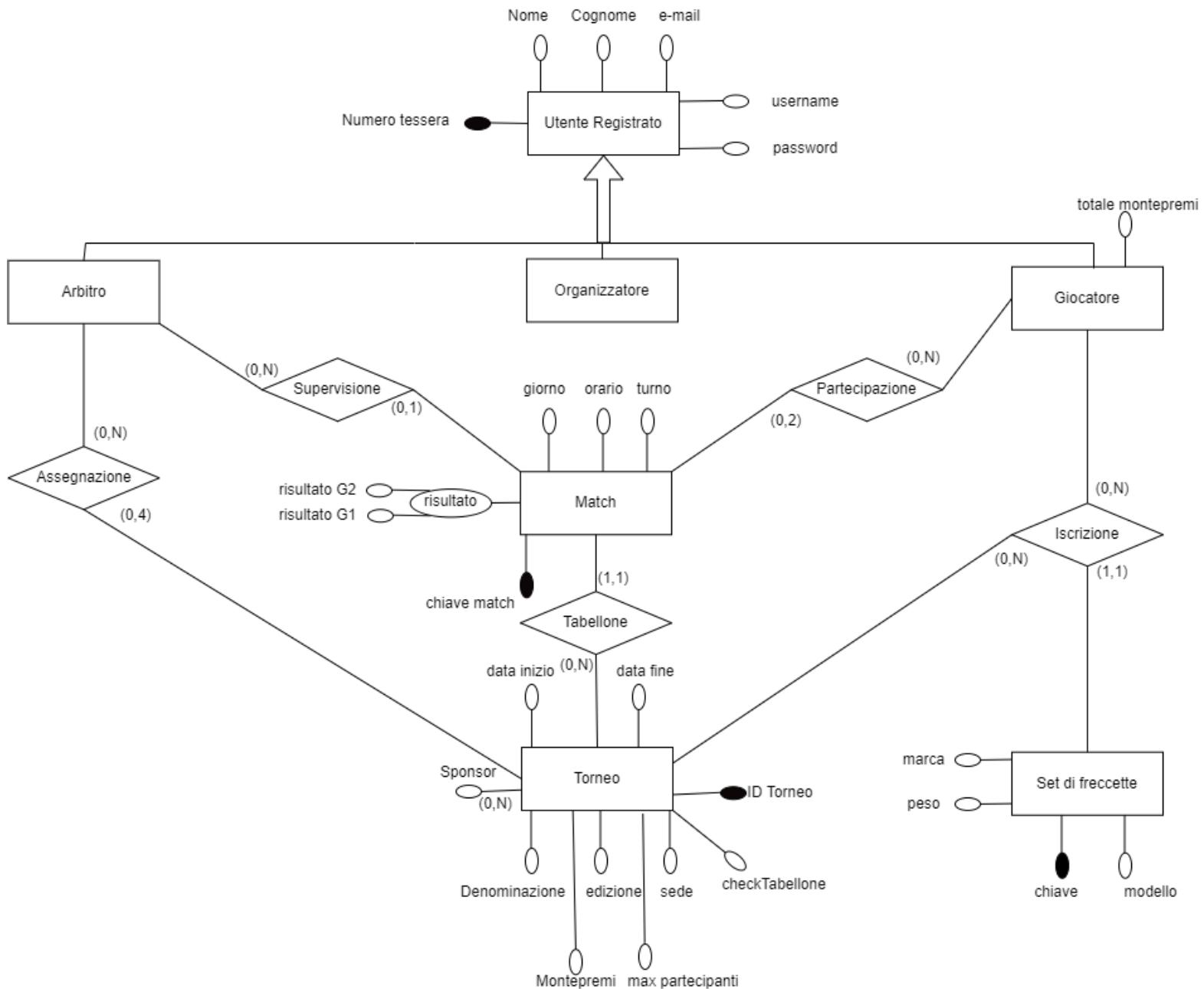
Caso d'uso:	RegistraRisultatoMatch
Attore primario	Arbitro
Attore secondario	-
Descrizione	Terminato il match del torneo assegnato all'arbitro, il suddetto registra il risultato nel sistema.
Pre-Condizioni	Arbitro è autenticato, è assegnato ad un torneo in corso ed esiste almeno un match assegnatogli il cui risultato non è ancora stato aggiunto .
Sequenza di eventi principale	<ol style="list-style-type: none"> 1. Il caso d'uso inizia quando un Arbitro vuole registrare il risultato di un match 2. Il sistema fornisce la lista dei tornei. 3. L'Arbitro seleziona il torneo 4. Il sistema fornisce la lista dei match. 5. L'Arbitro seleziona il match 6. L'arbitro inserisce il risultato dell'incontro. 7. Il sistema verifica la correttezza dei dati inseriti. 8. Se i dati inseriti non sono corretti: <ol style="list-style-type: none"> 8.1. Il sistema manda un messaggio di errore. 9. Altrimenti: <ol style="list-style-type: none"> 9.1. Il sistema manda un messaggio di inserimento riuscito. 9.2. Il sistema controlla se nel tabellone è presente un match successivo 9.3. Nel caso il match non è la finale del torneo, il sistema assegna il vincitore al match successivo.
Post-Condizioni	Il risultato del match è assegnato
Casi d'uso correlati	<i>nessuno</i>
Sequenza di eventi alternativi	-

Caso d'uso:	RegistraUtente
Attore primario	Funzionario dalla FPF
Attore secondario	-
Descrizione	Il funzionario registra un'utente nel sistema.
Pre-Condizioni	Il funzionario è stato autenticato dal sistema.
Sequenza di eventi principale	<ol style="list-style-type: none"> 1. Il caso d'uso inizia quando il funzionario FPF vuole registrare un'utente nel sistema. 2. Il funzionario sceglie se registrare un Arbitro, un Giocatore o un Organizzatore. 3. Il funzionario inserisce nome, cognome, numero tessera FPF ed email. 4. Il sistema genera le credenziali (nome utente e password) per l'accesso al sistema dell'utente registrato. 5. Il sistema predisponde un messaggio di posta elettronica con le credenziali d'accesso. 6. Il sistema preleva l'indirizzo email del destinatario dai dati inseriti in precedenza dal funzionario FPF.

	7. Il sistema invia il messaggio predisposto e il destinatario al servizio email. 8. Il servizio email invia il messaggio all'utente registrato destinatario.
Post-Condizioni	Viene aggiornato il sistema con un nuovo utente registrato.
Casi d'uso correlati	<i>InviaMail</i>
Sequenza di eventi alternativi	-

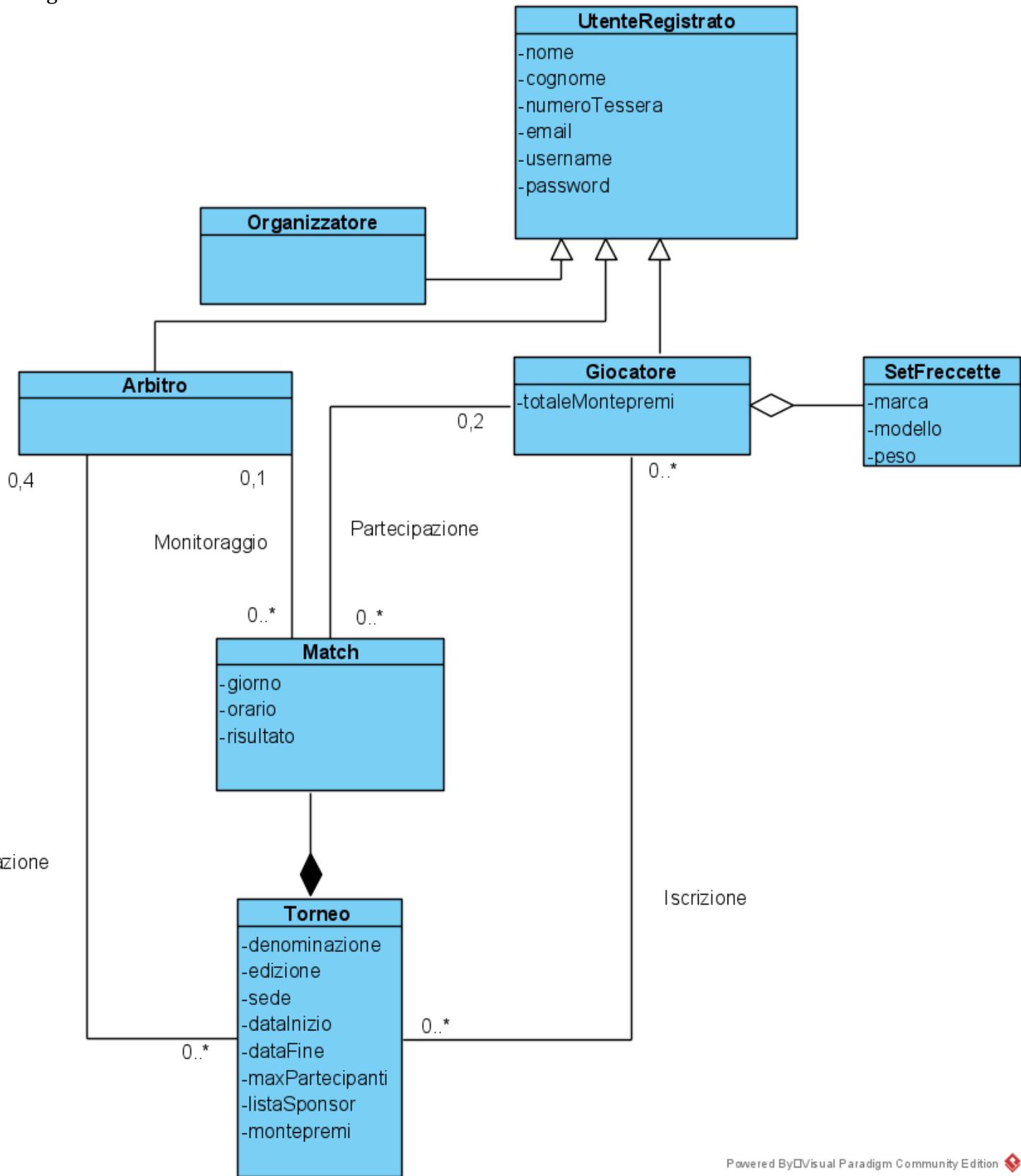
2.6 Modellazione dei dati

2.6.1 Progettazione concettuale



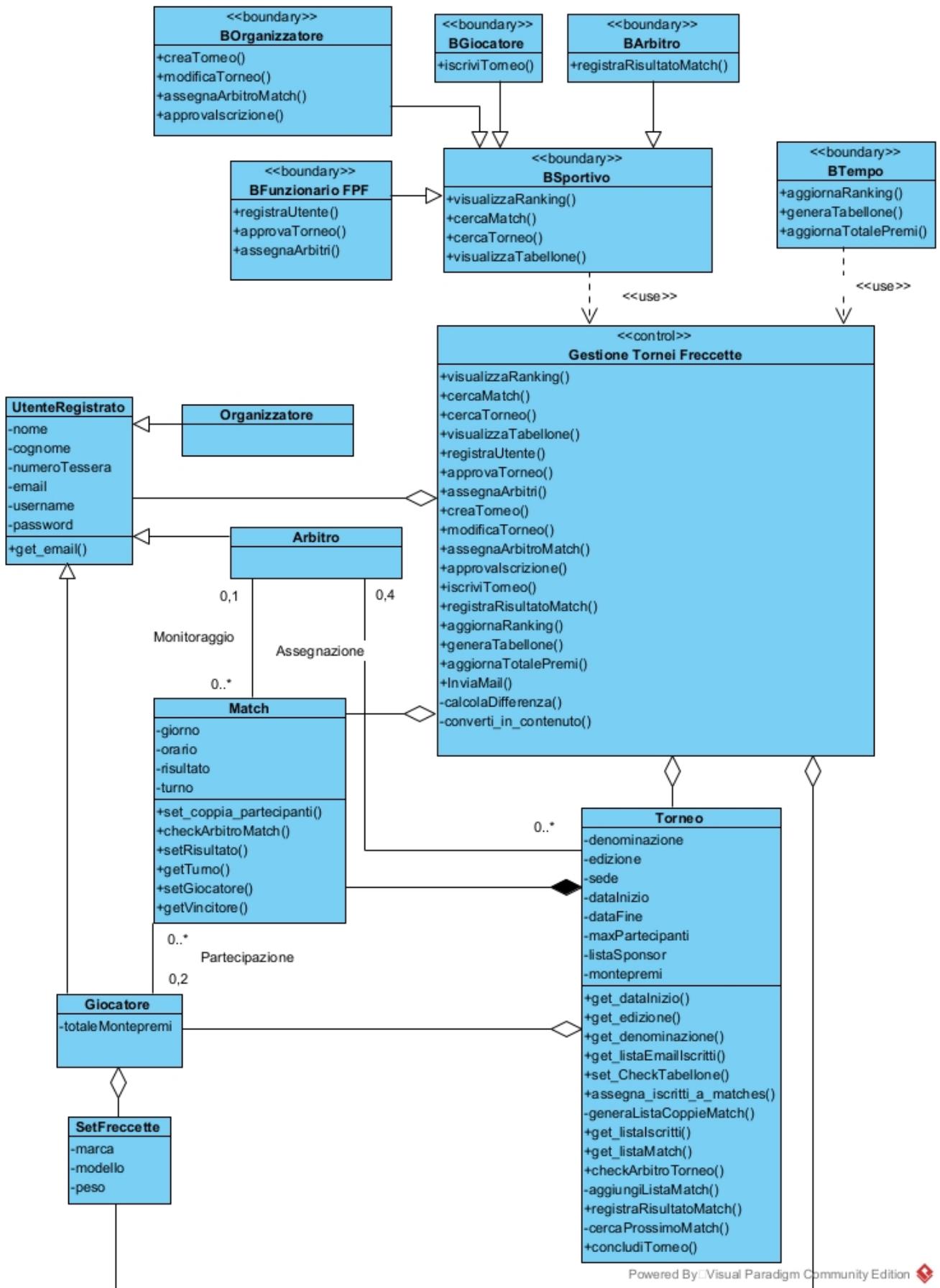
2.7 Diagramma delle classi

Diagramma delle classi di analisi



Powered By Visual Paradigm Community Edition

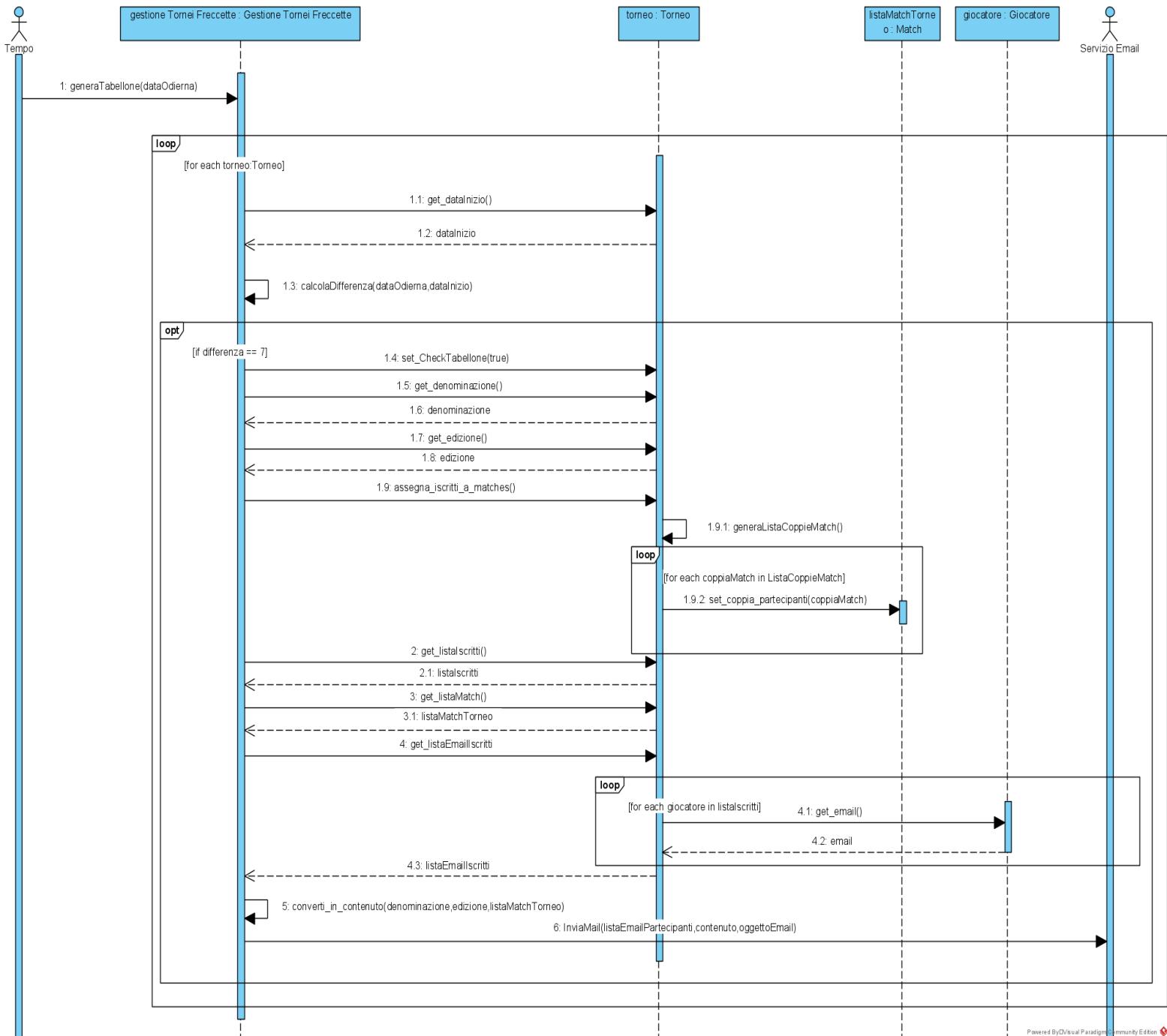
Diagramma delle classi raffinato (Boundary e Control)



Powered By Visual Paradigm Community Edition

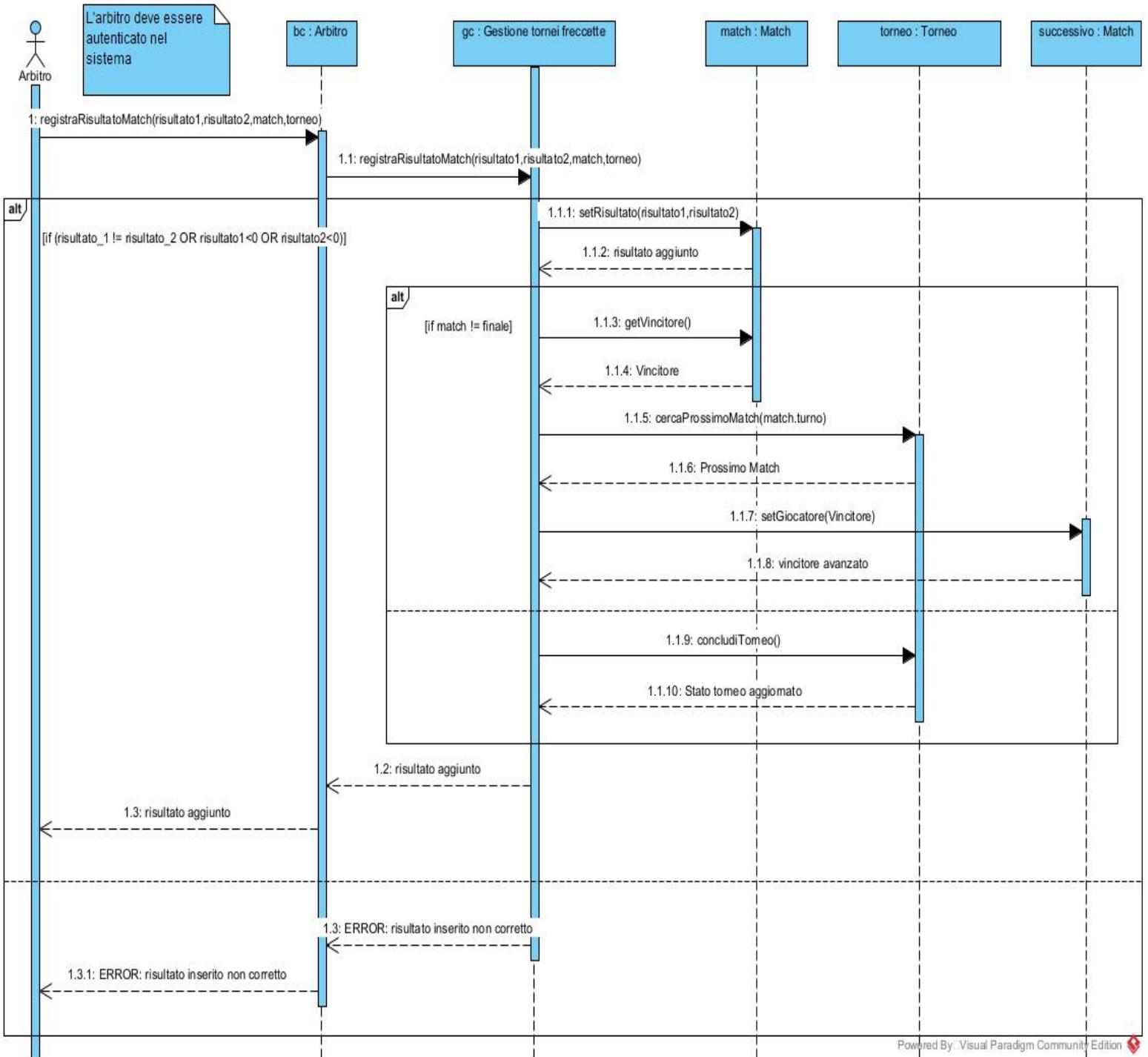
2.8 Diagrammi di sequenza

[SD GeneraTabellone]



Powered ByVisual Paradigm Community Edition

[SD Registra risultato match]



Powered By: Visual Paradigm Community Edition

2.9 Verifica della completezza dei requisiti

Legenda: UCD = Use Case Diagram, CD = Class Diagram, SD = Sequence Diagram

- RF01 è modellato nell'UCD con l'attore "Funzionario della FPF" e con il caso d'uso UC1
 - RF02 non è modellato (doveva essere modellato nel SD di RegistraUtente)
 - RF03 è modellato nell'UCD con il caso d'uso UC17 con attore(secondario) "Servizio Email"
 - RF04 è modellato nell'UCD con l'attore "Organizzatore" con il caso d'uso UC2
 - RF05 è modellato nell'UCD con l'attore "Organizzatore" con il caso d'uso UC3
 - RF06 è modellato nell'UCD con l'attore "Funzionario della FPF" con il caso d'uso UC14
 - RF07 è modellato nell'UCD con l'attore "Funzionario della FPF" con il caso d'uso UC13
 - RF08 è modellato nell'UCD con l'attore "Tempo" con il caso d'uso UC4
 - RF09 è modellato nell'UCD con l'attore "Giocatore" con il caso d'uso UC5
 - RF10 è modellato nell'UCD con l'attore "Organizzatore" con il caso d'uso UC15
 - RF11 è modellato nell'UCD con il caso d'uso UC17 con attore(secondario) "Servizio Email"
 - RF12 è modellato nell'UCD con l'attore "Tempo" con il caso d'uso UC6 e modellato nell' SD "GeneraTabellone"
 - RF13 è modellato nell'UCD con il caso d'uso UC17 con attore(secondario) "Servizio Email"
 - RF14 è modellato nell'UCD con l'attore "Organizzatore" con il caso d'uso UC7
 - RF15 è modellato nell'UCD con l'attore "Arbitro" con il caso d'uso UC8 e modellato nel SD "Registra risultato match"
 - RF16 è modellato nel SD "Registra risultato match"
 - RF17 è modellato nell'UCD con l'attore "Tempo" con il caso d'uso UC9
 - RF18 è modellato nell'UCD con l'attore "Sportivo" e con i casi d'uso UC10, UC11, UC12, UC16
 - RF19 è modellato nell'UCD con l'attore "Organizzatore" e con il caso d'uso UC18
-
- RD1 è modellato nell'CD con la classe "Organizzatore" figlia della classe "UtenteRegistrato"
 - RD2 è modellato nell'CD con la classe "Arbitro" figlia della classe "UtenteRegistrato"
 - RD3 è modellato nell'CD con la classe "Giocatore" figlia della classe "UtenteRegistrato"
 - RD4 è modellato nell'CD con la classe "Torneo"
 - RD5 è modellato nell'CD con la classe "SetFreccette"
 - RD6 è modellato nell'CD con la classe "Match"

3. Stima dei costi

GeneraTabellone

	VALORE	SEMPLICE	MEDIO	COMPLESSO	TOT
NILF	1	7	10	15	15
NEIF	0	5	7	10	0
NEI	1	3	4	6	3
NEO	1	4	5	7	5
NEQ	0	3	4	6	0

NILF = I tabelloni vengono generati dal sistema, li identifichiamo come ILF[complesso]

NEI= DataOdierna[Semplice]

NEO= Invio tabelloni, ottenuti tramite elaborazione [Medio]

JAVA = 53

UFP= 15+3+5=[23]

*LOC/JAVA = 53*23 = [1219]*

FATTORI CORRETTIVI

COMUNICAZIONE DATI	4
DISTRIBUZIONE ELABORAZIONE	1
PRESTAZIONI	1
UTILIZZO INTENSIVO CONFIGURAZIONE	0
FREQUENZA DELLE TRANSAZIONI	0
INSERIMENTO DATI INTERATTIVO	0
EFFICIENZA PER L'UTENTE FINALE	0
AGGIORNAMENTO INTERATTIVO	1
COMPLESSITA' ELABORATIVA	1
RIUSABILITA'	1
FACILITA' INSTALLAZIONE	0
FACILITA' GESTIONE OPERATIVA	0
MOLTEPLICITA' DI SITI	0
FACILITA' DI MODIFICA	0

*FP = 23 * (0,65 + 0,01 * 9) = 17,02 → [17]*

*LOC/JAVA = 53*17 = [901]*

RegistraRisultatoMatch

	VALORE	SEMPLICE	MEDIO	COMPLESSO	TOT
NILF	1	7	10	15	7
NEIF	0	5	7	10	0
NEI	2	3	4	6	2*3 = 6
NEO	0	4	5	7	0
NEQ	0	3	4	6	0

NILF= Match e torneo sono recuperati dal database[Semplice]

NEI= Risultato1,Risultato2[Semplice]

JAVA = 53

*UFP= 7+(2*3)+4 = [13]*

*LOC/JAVA = 53*13 = [689]*

FATTORI CORRETTIVI

COMUNICAZIONE DATI	4
DISTRIBUZIONE ELABORAZIONE	1
PRESTAZIONI	0
UTILIZZO INTENSIVO CONFIGURAZIONE	0
FREQUENZA DELLE TRANSAZIONI	0
INSERIMENTO DATI INTERATTIVO	5
EFFICIENZA PER L'UTENTE FINALE	1
AGGIORNAMENTO INTERATTIVO	1
COMPLESSITA' ELABORATIVA	1
RIUSABILITA'	1
FACILITA' INSTALLAZIONE	0
FACILITA' GESTIONE OPERATIVA	0
MOLTEPLICITA' DI SITI	0
FACILITA' DI MODIFICA	0

*FP = 13 * (0,65+0,01*14)=10,27 → [10]*

*LOC/JAVA = 53*10 = [530]*

4. Piano di test funzionale

PIANO DI TEST UTILIZZANDO IL METODO DEL *CATEGORY-PARTITION TESTING* PER LA FUNZIONALITÀ DI “*GeneraTabellone*”.

Al fine di consentire al tester di effettuare test su tale funzione, è stato implementato una interfaccia che fungerà da astrazione dell'attore Tempo in modo tale che il tester possa emulare l'inserimento della data odierna da parte dell'attore Tempo.

Data	Torneo
<ul style="list-style-type: none">• Data con formato e valori validi• Data con formato non valido [ERROR]• Data con formato valido ma valori non validi [ERROR]	<ul style="list-style-type: none">• E' presente nel database un torneo che ha raggiunto il massimo numero di iscritti senza tabellone• E' presente nel database un torneo senza tabellone ma che non ha raggiunto il massimo numero di iscritti• Non è presente nel database un torneo senza tabellone

Il numero di test da effettuarsi senza particolari vincoli è: $3 \times 3 = 9$

Introduciamo i vincoli [ERROR]

Il numero di test da eseguire per testare singolarmente i vincoli è: 2 (2 per Data)

Il numero di test risultante è: $1 \times 3 + 2 = 5$

TEST SUITE

Test Case ID	Descrizione	Classi di equivalenza coperte	Pre-condizioni	Input	Output Attesi	Post-condizioni Attese
1	Tutti input validi	Data valida Torneo che ha raggiunto il massimo numero di iscritti senza tabellone	E' presente nel database un torneo, con data di inizio 7 giorni successiva alla data odierna, il quale tabellone non è stato ancora generato e ha raggiunto il numero massimo di partecipanti.	{Data: "2022-06-03"}	Email inviata a: 'emailGiocatore'	I giocatori sono assegnati ai match, ricevono il tabellone per email e il torneo risulta in corso
2	Formato input Data valido con valore non valido.	Data con formato valido ma valori non validi [ERROR] Torneo che ha raggiunto il massimo numero di iscritti senza tabellone		{Data: "2022-12-32"}	'Inserimento input errato, inserire in formato yyyy-mm-dd'	
3	Formato input Data non valido	Data con formato non valido [ERROR] Torneo che ha raggiunto il massimo numero di iscritti senza tabellone		{Data: "2022/31-12"}	'Inserimento input errato, inserire in formato yyyy-mm-dd'	
4	Tutti input validi,	Data valida	E' presente nel database un	{Data: "2022-07-03"}	'TORNEO nomeTorneo edizioneTorneo [data]	Il torneo è terminato

	presenza di un torneo valido ma con numero di iscritti diverso dal massimo numero di partecipanti.	Torneo senza tabellone ma che non ha raggiunto il massimo numero di iscritti	torneo, con data di inizio 7 giorni successiva alla data odierna, il quale tabellone non è stato ancora generato ma non ha raggiunto il numero massimo di partecipanti.		<i>inizio torneo]</i> CANCELLATO'	
5	Tutti input validi, NESSUNA presenza nel database di tornei senza tabellone	Data valida Non è presente nel database un torneo senza tabellone	Non è presente nel database alcun torneo senza tabellone o che la sua data di inizio sia 7 giorni successiva alla data odierna.	{Data: "2022-06-22"}	'Nessun tornei inizia tra una settimana'	

**PIANO DI TEST UTILIZZANDO IL METODO DEL *CATEGORY-PARTITION TESTING* PER LA FUNZIONALITÀ DI
“RegistraRisultatoMatch”.**

Risultato Giocatore 1	Risultato Giocatore 2	Match	Torneo
<ul style="list-style-type: none"> Risultato Giocatore 1 con formato e valori validi Risultato Giocatore 1 con formato non valido[ERROR] Risultato Giocatore 1 con formato valido e valori non validi [ERROR] 	<ul style="list-style-type: none"> Risultato Giocatore 2 con formato e valori validi Risultato Giocatore 2 con formato non valido[ERROR] Risultato Giocatore 2 con formato valido e valori non validi [ERROR] Risultato Giocatore 2 con formato e valori validi [IF RISULTATO GIOCATORE 1 == RISULTATO GIOCATORE 2] 	<ul style="list-style-type: none"> L'Arbitro è assegnato almeno ad un Match valido[IF TORNEO] L'Arbitro è assegnato alla finale di un torneo[IF TORNEO] L'Arbitro non è assegnato a nessun match attivo del torneo [ERROR] 	<ul style="list-style-type: none"> Arbitro è assegnato almeno ad un torneo in corso[TORNEO] Arbitro non è assegnato a nessun torneo in corso[ERROR]

Il numero di test da effettuarsi senza particolari vincoli è : $3 \times 4 \times 3 \times 2 = 72$

Introduciamo i vincoli [ERROR]

Il numero di test da eseguire per testare singolarmente i vincoli è : 6 (2 per Risultato Giocatore 1 , 2 per Risultato Giocatore 2 , 1 per Torneo , 1 per Match)

Il numero di test risultante è : $(1 \times 2 \times 2 \times 1) + 6 = 10$

Considerando tutti i vincoli PROPERTY : 2(1 per Risultato Giocatore 2 , 2 per Match)

Il numero di test risultante è : $(1 \times 1) \times (1 \times 1 + 1 \times 1 + 1 \times 1) + 6 = 9$

TEST SUITE

Test Case ID	Descrizione	Classi di equivalenza coperte	Pre-condizioni	Input	Output Attesi	Post-condizioni Attese

1	Tutti gli input sono validi Il match non è una finale	Risultato Giocatore 1 valido Risultato Giocatore 2 valido Match valido Torneo Valido	Arbitro è autenticato , è assegnato ad un torneo in corso ed esiste almeno un match assegnatogli il cui risultato non è ancora stato aggiunto ..	{ Torneo: "FRECCESTIVAL ESTATE 2022 Via duomo 12" Match: "2022-06-10 10:00:00 Turno: 1 Salvio Silvio vs Birra Ale" Risultato Giocatore 1 : "3" Risultato Giocatore 2 : "0"}	Risultato aggiunto	il risultato del match è assegnato. Il Vincitore avanza al prossimo match
2	Tutti gli input sono validi Il match è una finale	Risultato Giocatore 1 valido Risultato Giocatore 2 valido Match valido Torneo valido	Arbitro è autenticato, è assegnato ad un torneo in corso ed esiste almeno una "finale" di torneo assegnatogli il cui risultato non è ancora stato aggiunto .	{ Torneo: "FRECCESTIVAL ESTATE 2022 Via duomo 12" Match: "2022-06-14 19:00:00 Turno: 3 Salvio Silvio vs Gucci Serena" Risultato Giocatore 1 : "0" Risultato Giocatore 2 : "3"}	Risultato aggiunto	il risultato del match è assegnato. Il Torneo è concluso.
3	Risultato Giocatore 1 con formato non valido	Formato Risultato Giocatore 1 non valido [ERROR] Risultato Giocatore 2 valido Match valido Torneo valido	Arbitro è autenticato , è assegnato ad un torneo in corso ed esiste almeno un match assegnatogli il cui risultato non è ancora stato aggiunto.	{ Torneo: " FRECCESTIVAL ESTATE 2022 Via duomo 12" Match: "2022-06-10 11:00:00 Turno:1 Pascale Giulia vs Disco Ernia" Risultato Giocatore 1 : "tre" Risultato Giocatore 2 : "0"}	Formato del risultato non valido	il risultato del match non è stato assegnato
4	Risultato Giocatore 1 con formato valido e valori non validi	Risultato Giocatore 1 con formato valido e valori non validi [ERROR] Risultato Giocatore 2 valido Match valido Torneo valido	Arbitro è autenticato , è assegnato ad un torneo in corso ed esiste almeno un match assegnatogli il cui risultato non è ancora stato aggiunto.	{ Torneo: " FRECCESTIVAL ESTATE 2022 Via duomo 12" Match: "2022-06-10 11:00:00 Turno:1 Pascale Giulia vs Disco Ernia" Risultato Giocatore 1 : "-3" Risultato Giocatore 2 : "0"}	Valore del risultato non valido	il risultato del match non è stato assegnato

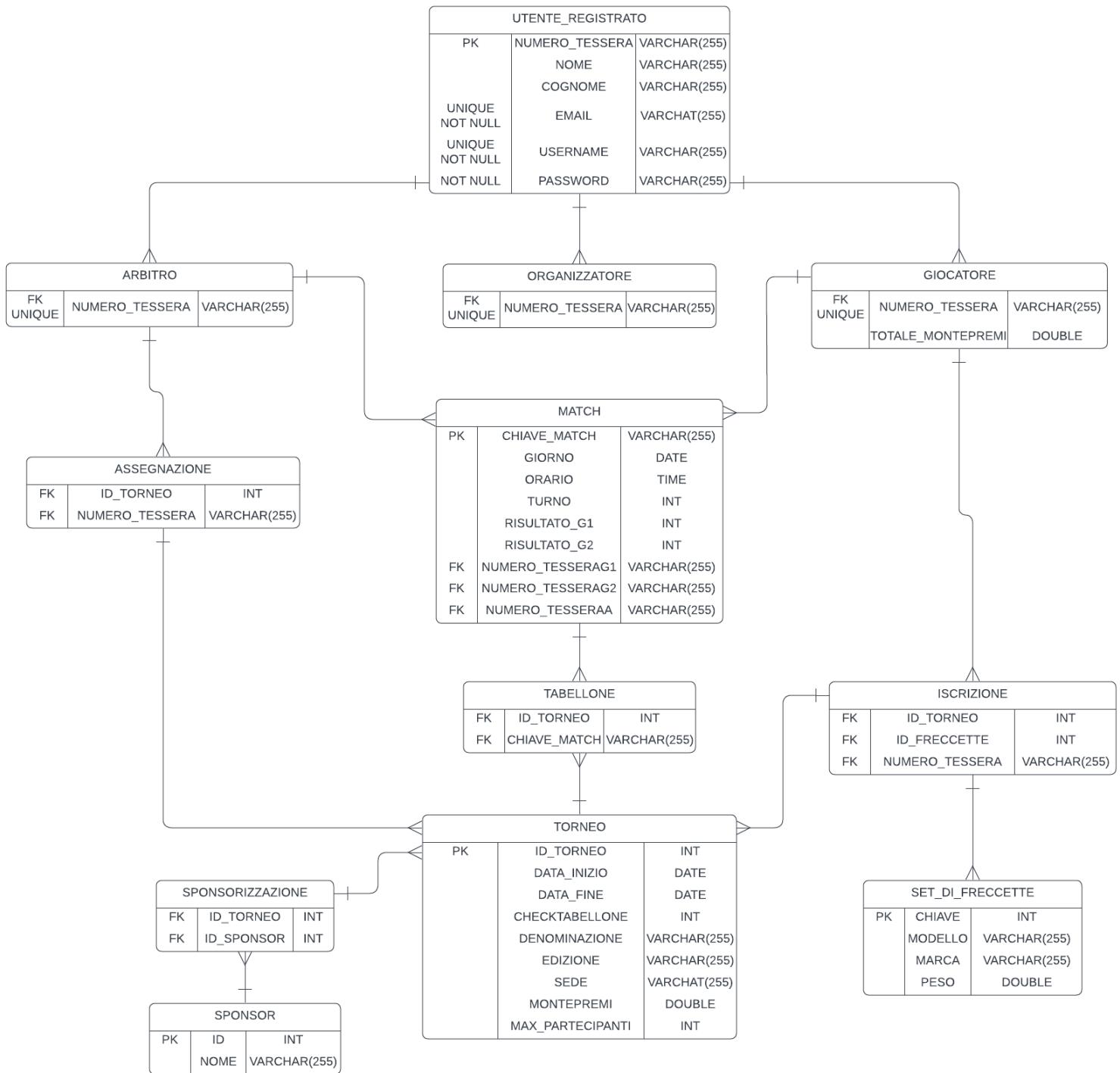
5	Risultato Giocatore 2 con formato non valido	Risultato Giocatore 1 valido Formato Risultato Giocatore 2 non valido [ERROR] Match valido Torneo valido	Arbitro è autenticato , è assegnato ad un torneo in corso ed esiste almeno un match assegnatogli il cui risultato non è ancora stato aggiunto.	{ Torneo: " FRECCESTIVAL ESTATE 2022 Via duomo 12" Match: "2022-06-10 11:00:00 Turno:1 Pascale Giulia vs Disco Ernia" Risultato Giocatore 1 : "3" Risultato Giocatore 2 : "zero"}	Formato del risultato non valido	il risultato del match non è stato assegnato
6	Risultato Giocatore 2 con formato valido e valori non validi	Risultato Giocatore 1 valido Risultato Giocatore 2 con formato valido e valori non validi [ERROR] Match valido Torneo valido	Arbitro è autenticato ed esiste almeno un match assegnatogli il cui risultato non è ancora stato aggiunto .	{ Torneo: " FRECCESTIVAL ESTATE 2022 Via duomo 12" Match: "2022-06-10 11:00:00 Turno:1 Pascale Giulia vs Disco Ernia" Risultato Giocatore 1 : "3" Risultato Giocatore 2 : "-2"}	Valore del risultato non valido	il risultato del match non è stato assegnato
7	Arbitro non è assegnato a nessun torneo in corso	Torneo non valido[ERROR]	Arbitro è autenticato. Non è assegnato a nessun torneo		Nessun Torneo Assegnato	Nessuna modifica
8	L'Arbitro non è assegnato a nessun match attivo del torneo	Torneo valido Match non valido[ERROR]	L'Arbitro non è assegnato a nessun match attivo del torneo	{Torneo: " FRECCESTIVAL ESTATE 2022 Via duomo 12"}	Nessun Match Assegnato	Nessuna modifica
9	Tutti gli input sono validi	Risultato Giocatore 1 valido Risultato Giocatore 2 valido[IF RISULTATO GIOCATORE 1 ==	Arbitro è autenticato ed esiste almeno un match assegnatogli il cui risultato non è ancora stato aggiunto .	{ Torneo: "FRECCESTIVAL ESTATE 2022 Via duomo 12" Match: "2022-06-10 11:00:00 Turno:1 Pascale Giulia vs Disco Ernia "	Risultato Inserito non corretto	il risultato del match non è stato assegnato

	RISULTATO GIOCATORE 2] Torneo valido Match non		Risultato Giocatore 1 : "3" Risultato Giocatore 2 : "3"}		
--	---	--	---	--	--

5. Progettazione

5.1 Progettazione della base di dati

5.1.1 Progettazione logica

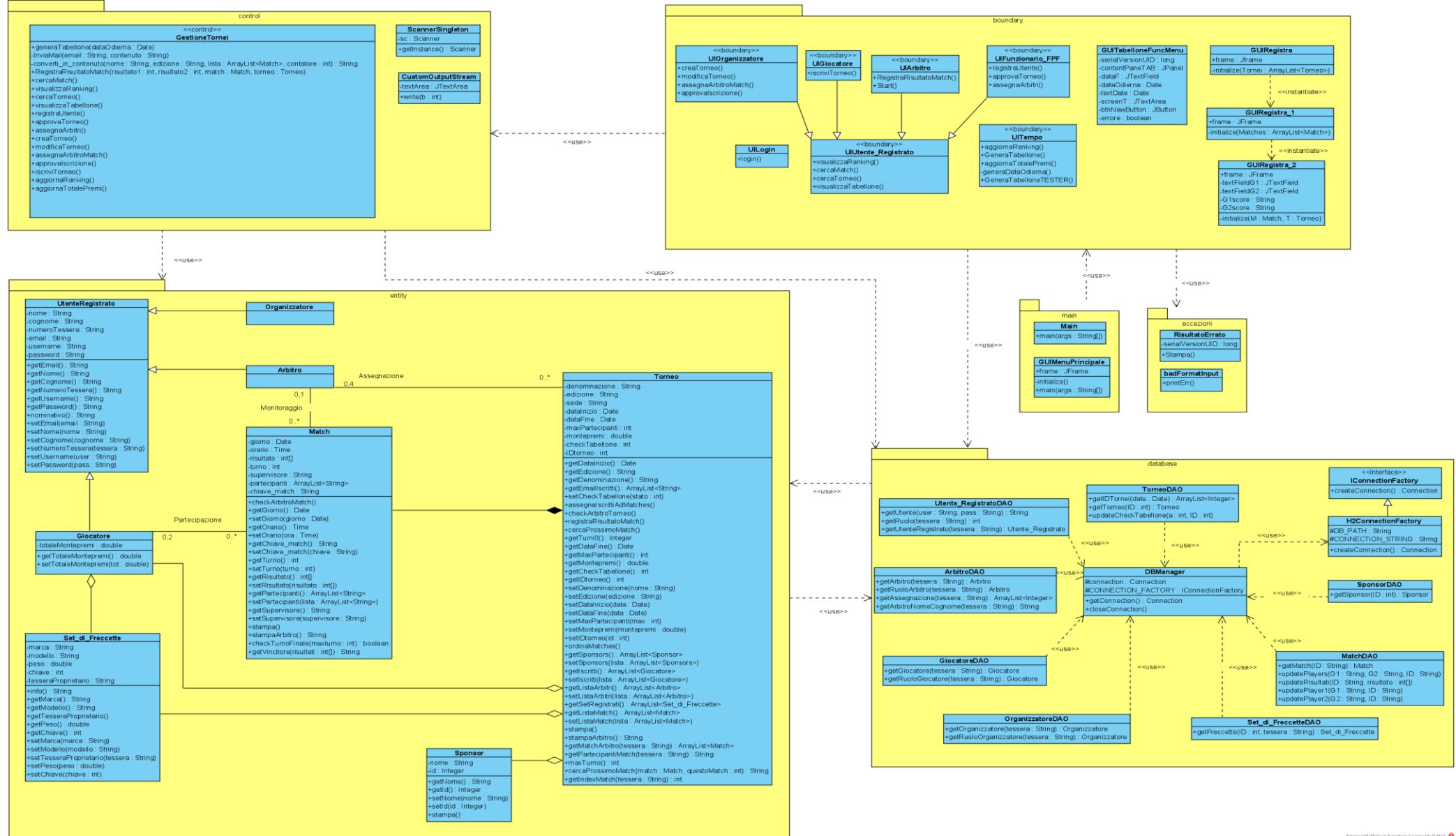


- **CREATE TABLE UTENTE_REGISTRATO**(NUMERO_TESSERA VARCHAR(255) PRIMARY KEY,NOME VARCHAR(255), COGNOME VARCHAR(255), EMAIL VARCHAR(255) NOT NULL UNIQUE,USERNAME VARCHAR(255) NOT NULL UNIQUE,PASSWORD VARCHAR(255) NOT NULL);
- **CREATE TABLE ARBITRO** (NUMERO_TESSERA VARCHAR(255) UNIQUE FOREIGN KEY REFERENCES UTENTE_REGISTRATO(Numero_Tessera));
- **CREATE TABLE ORGANIZZATORE**(NUMERO_TESSERA VARCHAR(255) UNIQUE FOREIGN KEY REFERENCES UTENTE_REGISTRATO(Numero_Tessera));
- **CREATE TABLE GIOCATORE**(NUMERO_TESSERA VARCHAR(255) UNIQUE FOREIGN KEY REFERENCES UTENTE_REGISTRATO(Numero_TESSERA),TOTALE_MONTEPREMI DOUBLE);
- **CREATE TABLE MATCH**(CHIAVE_MATCH VARCHAR(255) PRIMARY KEY,GIORNO DATE,ORARIO TIME,TURNO INT,RISULTATO_G1 INT,RISULTATO_G2 INT, NUMERO_TESSERAG1 VARCHAR(255) FOREIGN KEY REFERENCES GIOCATORE(Numero_TESSERA),NUMERO_TESSERAG1 VARCHAR(255) FOREIGN KEY REFERENCES GIOCATORE(Numero_TESSERA),NUMERO_TESSERA VARCHAR(255) FOREIGN KEY REFERENCES ARBITRO(Numero_TESSERA));
- **CREATE TABLE TORNEO**(ID_TORNEO INT PRIMARY KEY,DATA_INIZIO DATE,DATA_FINE DATE,CHECKTABELLONE INT,DENOMINAZIONE VARCHAR(255),EDIZIONE VARCHAR(255),SEDE VARCHAR(255),MONTEPREMI DOUBLE,MAX_PARTECIPANTI INT);
- **CREATE TABLE SPONSOR**(ID INT PRIMARY KEY,NOME VARCHAR(255));
- **CREATE TABLE SET_DI_FRECCETTE**(CHIAVE INT PRIMARY KEY,MODELLO VARCHAR(255),MARA VARCHAR(255),PESO DOUBLE);
- **CREATE TABLE ISCRIZIONE**(ID_TORNEO INT FOREIGN KEY REFERENCES TORNEO(ID_TORNEO),ID_FRECCETTE INT FOREIGN KEY REFERENCES SET_DI_FRECCETTE(CHIAVE),NUMERO_TESSERA VARCHAR(255) FOREIGN KEY REFERENCES GIOCATORE(Numero_TESSERA));
- **CREATE TABLE SPONSORIZZAZIONE**(ID_TORNEO INT FOREIGN KEY REFERENCES TORNEO(ID_TORNEO),ID_SPONSOR INT FOREIGN KEY REFERENCES SPONSOR(ID));
- **CREATE TABLE TABELLONE**(ID_TORNEO INT FOREIGN KEY REFERENCES TORNEO(ID_TORNEO), CHIAVE_MATCH VARCHAR(255) FOREIGN KEY REFERENCES MATCH(CHIAVE_MATCH));
- **CREATE TABLE ASSEGNAZIONE**(ID_TORNEO INT FOREIGN KEY REFERENCES TORNEO(ID_TORNEO), NUMERO_TESSERA VARCHAR(255) FOREIGN KEY REFERENCES ARBITRO(Numero_TESSERA));

Le relazioni “SUPERVISIONE” e “PARTECIPAZIONE” sono stati ridotti come attributi di entità “MATCH”, in particolare la relazione “PARTECIPAZIONE” è diventata una coppia di attributi ‘NUMERO_TESSERAG1’ e ‘NUMERO_TESSERAG2’ e “SUPERVISIONE” è diventato l’attributo ‘NUMERO_TESSERA’.

L’attributo ‘Sponsor’ dell’entità “TORNEO” è stato espanso come una entità a sè legato a torneo da “SPONSORIZZAZIONE”, in particolare ‘Sponsor’ è diventato una tabella “SPONSOR” con attributi ‘ID’ e ‘NOME’.

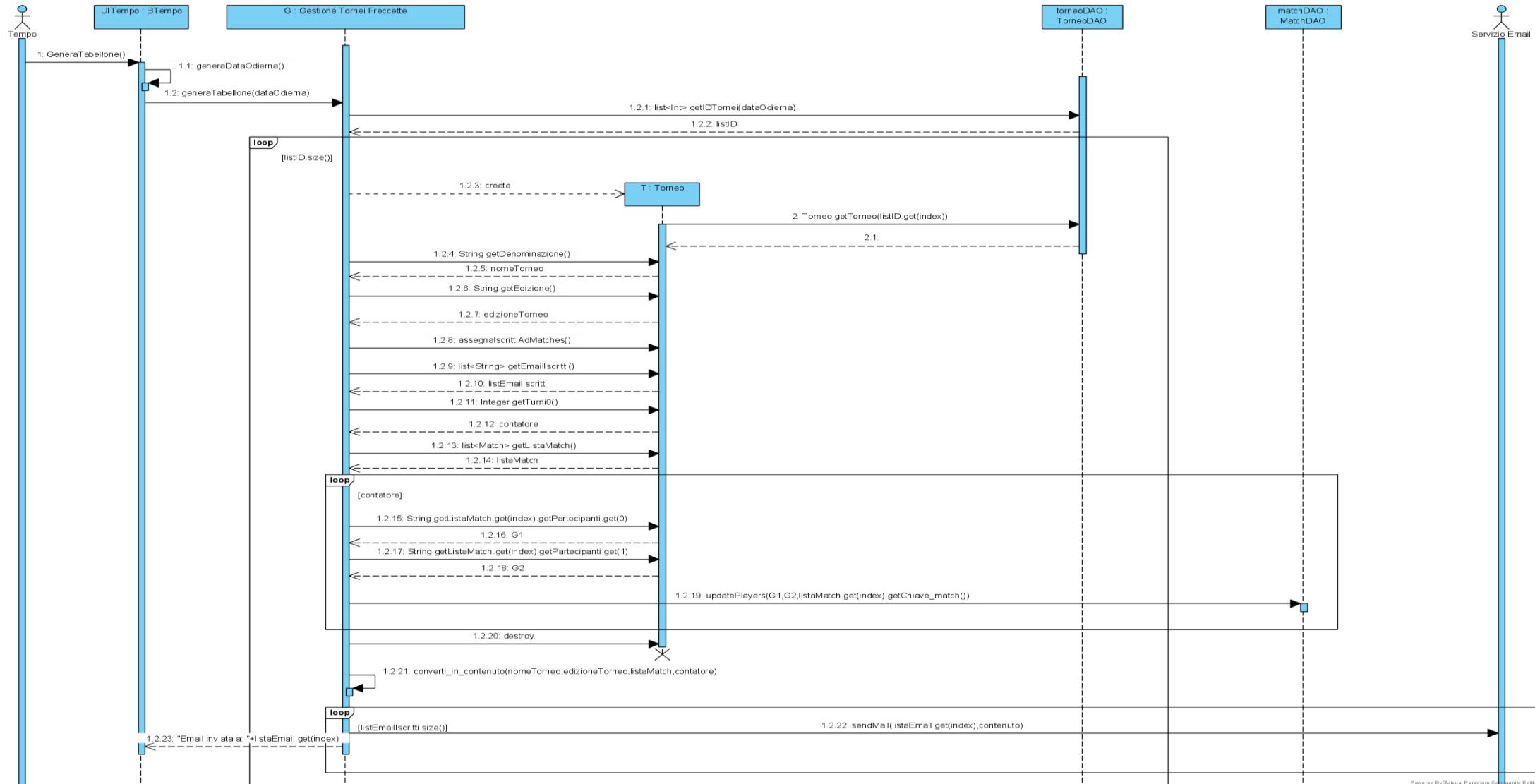
5.2 Diagramma delle classi



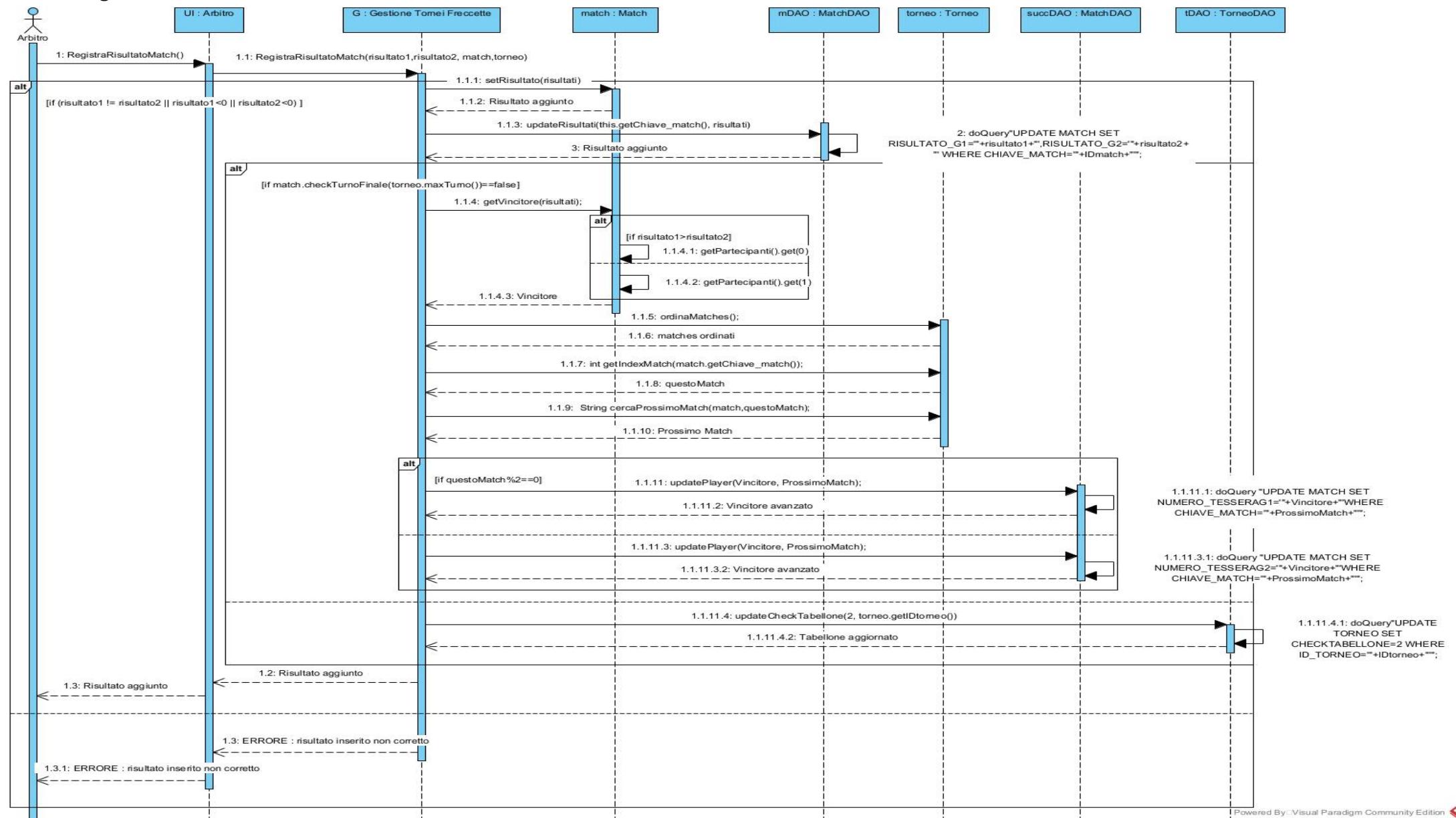
Powered ByUML Paragon Community Edition

5.3 Diagrammi di sequenza

GeneraTabellone



RegistraRisultatoMatch



6. Implementazione

Package:

boundary: contiene le interfacce dell'applicazione che permettono di utilizzare determinate funzionalità del sistema

è composta da : GUIArbitro , GUILogin , GUIRegistra , GUIRegistra_1, GUIRegistra_2 ,
,GUITabelloneFuncMenu,
UIArbitro,UILogin, UITempo.

control: contiene la classe che implementa la logica di business del sistema e supporti per l'interazione con l'utente.

è composta da : CustomOutputStream , GestioneTornei , ScannerSingleton.

database: contiene le classi che interagiscono con il database dell'applicazione.

è composta da : ArbitroDAO , GiocatoreDAO , MatchDAO , OrganizzatoreDAO ,
Set_di_FrecletteDAO ,SponsorDAO,
TorneoDAO , Utente_RegistratoDAO , DBManager ,H2ConnectionFactory ,
IConnectionFactory .

eccezioni: contiene le classi di eccezioni definiti.

è composta da : badFormatInput,RisultatoErrato.

entity: contiene le classi che rappresentano le entità del diagramma delle classi.

è composta da : Arbitro , Giocatore , Match , Organizzatore , Set_di_Freclette , Sponsor ,
Torneo , Utente_Registrato .

main: contiene i terminali dell'applicazione.

è composta da : GUIMenuPrincipale , Main .

Classi:

GUIArbitro.java , GUILogin.java , GUIRegistra.java , GUIRegistra_1.java , GUIRegistra_2.java
,GUITabelloneFuncMenu .java , UIArbitro.java, UILogin.java , UITempo.java ,
CustomOutputStream.java , GestioneTornei.java , ScannerSingleton.java , ArbitroDAO.java ,
GiocatoreDAO.java , MatchDAO.java , OrganizzatoreDAO.java , Set_di_FrecletteDAO.java ,
SponsorDAO.java, TorneoDAO.java , Utente_RegistratoDAO.java , DBManager.java
,H2ConnectionFactory.java , IConnectionFactory.java ,Arbitro.java , Giocatore.java , Match.java
, Organizzatore.java , Set_di_Freclette.java , Sponsor.java , Torneo.java ,
Utente_Registrato.java , GUIMenuPrincipale.java , Main.java.

Eccezioni:

badFormatInput.java, RisultatoErrato.java .

Artefatti necessari: h2-1.4.196.jar, GFTdb.mv.db, GFTdb.trace.db , WindowsBuilder 1.9.8 .

LOC = boundary(1086)+control(205)+database(545)+eccezioni(29)+entity(834)+main(210)
= 2909
LLOC= boundary(1021)+control(190)+database(501)+eccezioni(23)+entity(692)+main(195)
= 2622

Funzione	Java stimato	Java LOC	Tot LOC	Tot LLOC
GeneraTabellone	901	854	2909	2622
RegistraRisultatoMatch	530	438	2909	2622

Package control

Class GestioneTornei

```
java.lang.Object
    control.GestioneTornei
```

```
public class GestioneTornei
extends java.lang.Object
```

Constructor Summary

Constructors

Constructor

`GestioneTornei()`

Method Summary

All Methods Static Methods Instance Methods Concrete Methods

Modifier and Type	Method
void	<code>generaTabellone(java.sql.Date dataodierna)</code>
static GestioneTornei	<code>getInstance()</code>
void	<code>RegistraRisultatoMatch(int risultato1, int risultato2, Match match, Torneo torneo)</code>

Methods inherited from class java.lang.Object

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Constructor Details

`GestioneTornei`

```
public GestioneTornei()
```

Method Details

getInstance

```
public static GestioneTornei getInstance()
```

generaTabellone

```
public void generaTabellone(java.sql.Date dataodierna)
```

RegistraRisultatoMatch

```
public void RegistraRisultatoMatch(int risultato1,
                                    int risultato2,
                                    Match match,
                                    Torneo torneo)
throws RisultatoErrato
```

Throws:

`RisultatoErrato`

E' possibile trovare l'intera documentazione Javadoc nel percorso \canale_j_z-sigma\Documentazione\Javadoc

7. Testing

7.1 Test strutturale

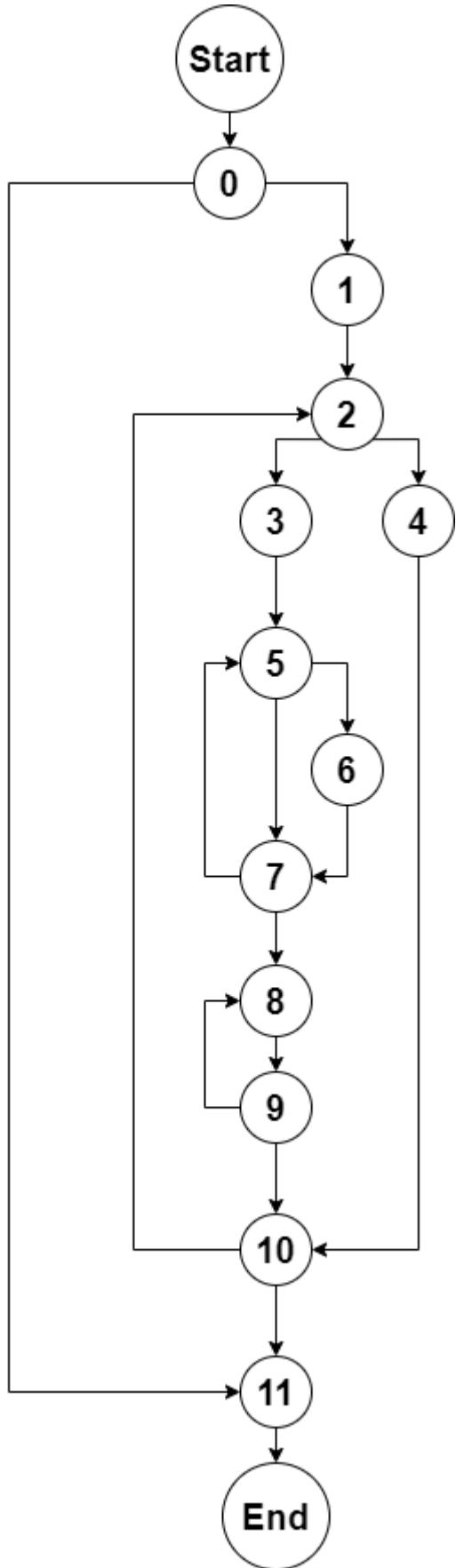
7.1.1 Complessità ciclomatica

- generaTabellone

```
public void generaTabellone(Date dataodierna) {
    ArrayList<Integer> listID = new ArrayList<Integer>();
    try {
        listID = TorneoDAO.getIDTornei(dataodierna);
    } catch (SQLException e) {
    }

    if(!listID.isEmpty()) {
        for(int i=0;i<listID.size();i++) {
            String nomeTorneo;
            String edizioneTorneo;
            Torneo T = new Torneo();
            try {
                T = new Torneo(TorneoDAO.getTorneo(listID.get(i)));
            } catch (SQLException e) {
            }
            nomeTorneo = T.getDenominazione();
            edizioneTorneo = T.getEdizione();
            if(T.getMaxPartecipanti()!=T.getIscritti().size()) {
                try {
                    TorneoDAO.updateCheckTabellone(2, listID.get(i));
                    System.out.println("[TORNEO "+T.getDenominazione()+" "+T.getEdizione()+" ["+T.getDataInizio()+"] CANCELLATO]");
                } catch (SQLException e) {
                }
            }
            else {
                T.assegnaIscrittiAdMatches();
                ArrayList<String> listaEmailIscritti = T.getEmailIscritti();
                Integer contatore=T.getTurni0();
                for(int j=0;j<contatore;j++) {
                    String G1 = T.getListaMatch().get(j).getPartecipanti().get(0);
                    String G2 = "NULL";
                    if(T.getListaMatch().get(j).getPartecipanti().size()==2) {
                        G2 = T.getListaMatch().get(j).getPartecipanti().get(1);
                    }
                    try {
                        MatchDAO.updatePlayers(G1, G2, T.getListaMatch().get(j).getChiave_match());
                    } catch (SQLException e) {
                    }
                }
            }
            try {
                TorneoDAO.updateCheckTabellone(1, listID.get(i));
            } catch (SQLException e) {
            }
            String contenuto = converti_in_contenuto(nomeTorneo,edizioneTorneo,T.getListaMatch(),contatore);
            for(int k=0;k<listaEmailIscritti.size();k++) {
                try {
                    sendMail(listaEmailIscritti.get(k), contenuto);
                } catch (Exception e) {
                }
            }
        }
    }
}
```

Control Flow Graph



NUMERO CICLOMATICICO:

numero di regioni chiuse del grafo = 7

numero di nodi predicati (0,2,5,7,9,10) +1 = 7

$$\# \text{ archi} - \# \text{ nodi} + 2 = (19 - 14) + 2 = 7$$

CAMMINI:

- 1) START-0-11-END
- 2) START-0-1-2-4-10-11-END
- 3) START-0-1-2-4-10-2-4-10-11-END
- 4) START-0-1-2-3-5-7-8-9-10-11-END
- 5) START-0-1-2-3-5-6-7-8-9-10-11-END
- 6) START-0-1-2-3-5-7-5-7-8-9-10-11-END
- 7) START-0-1-2-3-5-7-8-9-8-9-10-11-END

- RegistraRisultatoMatch

```

public void RegistraRisultatoMatch(int risultato1,int risultato2, Match match , Torneo torneo) throws
RisultatoErrato {
    if(risultato1==risultato2 || risultato1<0 || risultato2<0)throw new RisultatoErrato();

    int [] risultati= {risultato1 , risultato2 } ;
    match.setRisultato(risultati);
    try {
        MatchDAO.updateRisultati(match.getChiave_match(), risultati);
    } catch (SQLException e) {

    }

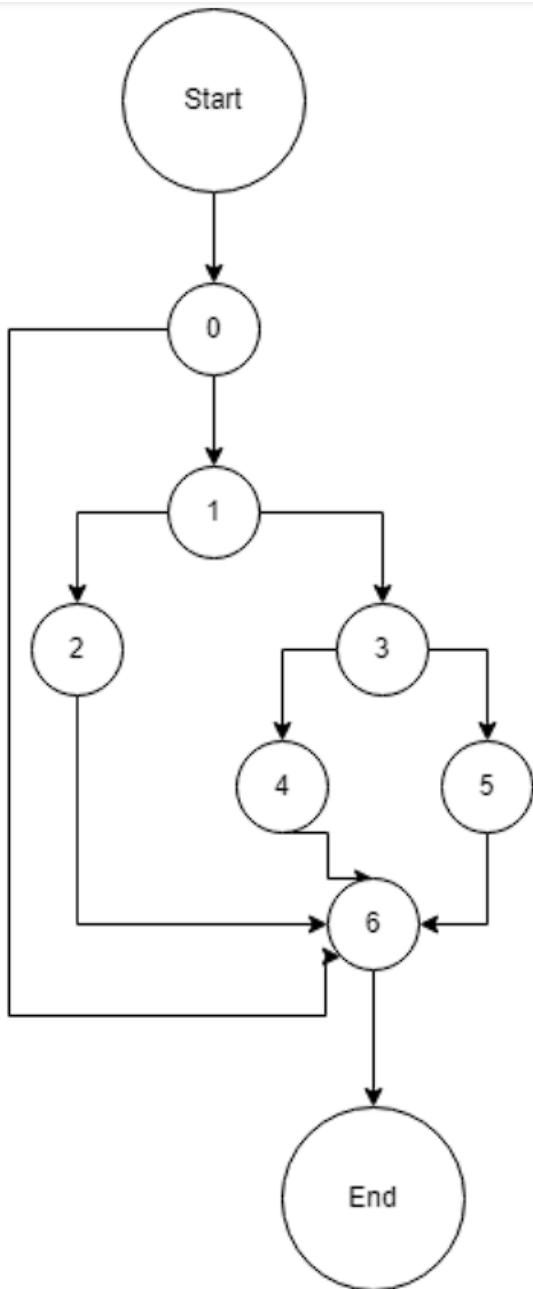
    if(match.checkTurnoFinale(torneo.maxTurno())==false) {
        String Vincitore = match.getVincitore(risultati);
        torneo.ordinaMatches();
        int questoMatch = torneo.getIndexMatch(match.getChiave_match());
        try {

            String ProssimoMatch = torneo.cercaProssimoMatch(match, questoMatch);
            if(questoMatch%2==0) {
                try {
                    MatchDAO.updatePlayer1(Vincitore, ProssimoMatch);
                } catch (SQLException e) {
                }
            }else{
                try {
                    MatchDAO.updatePlayer2(Vincitore, ProssimoMatch);
                } catch (SQLException e) {
                }
            }
        }catch(NullPointerException ex) {
        }

    }else{
        try {
            TorneoDAO.updateCheckTabelleOne(2, torneo.getIDtorneo());
        } catch (SQLException e) {
        }
    }
}

```

Control Flow Graph



Numero regioni 4

Numero di nodi predicati (0,1,3) =4
archi - nodi +2 = 11 - 9 + 2 =4

Cammini:

- 1) START 0,6 END
- 2) START 0,1,2,6 END
- 3) START 0,1,3,4,6 END
- 4) START 0,1,3,5,6 END

7.1.2 Test di unità

- **GeneraTabellone**

1)={listID.size()=0,T.getMaxPartecipanti=0,T.getIscritti=0,T.getListMatch().get(j).getPartecipanti().size()=0,contatore=0,listaEmailIscritti.size()=1}

2)={listID.size()=1,T.getMaxPartecipanti=4,T.getIscritti=2,T.getListMatch().get(j).getPartecipanti().size()=2,contatore=2,listaEmailIscritti.size()=1}

3)={listID.size()=2,T.getMaxPartecipanti=4,T.getIscritti=2,T.getListMatch().get(j).getPartecipanti().size()=2,contatore=2,listaEmailIscritti.size()=1}

4)={listID.size()=1,T.getMaxPartecipanti=2,T.getIscritti=2,T.getListMatch().get(j).getPartecipanti().size()=2,contatore=1,listaEmailIscritti.size()=1}

5)={listID.size()=1,T.getMaxPartecipanti=3,T.getIscritti=3,T.getListMatch().get(j).getPartecipanti().size()=1,contatore=3,listaEmailIscritti.size()=1}

6)={listID.size()=1,T.getMaxPartecipanti=2,T.getIscritti=2,T.getListMatch().get(j).getPartecipanti().size()=2,contatore=2,listaEmailIscritti.size()=1}

7)={listID.size()=1,T.getMaxPartecipanti=2,T.getIscritti=2,T.getListMatch().get(j).getPartecipanti().size()=2,contatore=1,listaEmailIscritti.size()=2}

- **RegistraRisultatoMatch**

1)={(risultato1=2,risultato2=2,match.checkTurnoFinale(torneo.maxTurno())==false,
torneo.getIndexMatch(match.getChiave_match()%2)==0),
(risultato1=-1,risultato2=2,match.checkTurnoFinale(torneo.maxTurno())==false,
torneo.getIndexMatch(match.getChiave_match()%2)==0),
(risultato1=-1,risultato2=2,match.checkTurnoFinale(torneo.maxTurno())==false,
torneo.getIndexMatch(match.getChiave_match()%2)==0)}

2)={risultato1=1,risultato2=2,match.checkTurnoFinale(torneo.maxTurno())==true,
(torneo.getIndexMatch(match.getChiave_match()%2)==0)}

3)={risultato1=1,risultato2=2,match.checkTurnoFinale(torneo.maxTurno())==false,
(torneo.getIndexMatch(match.getChiave_match()%2)==0)}

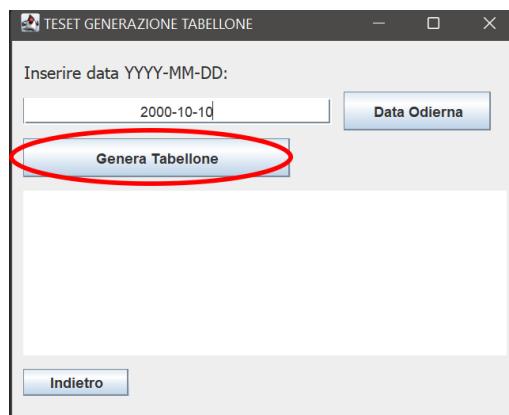
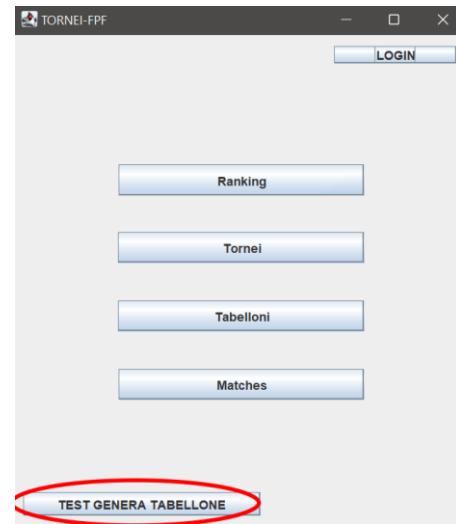
4)={risultato1=1,risultato2=2,match.checkTurnoFinale(torneo.maxTurno())==false,
(torneo.getIndexMatch(match.getChiave_match()%2==1)}

7.2 Test funzionale

Istruzioni per utilizzare la funzione “GeneraTabellone”

Per simulare l'utilizzo della funzione GeneraTabellone da parte dell'attore Tempo è stata sviluppata un'interfaccia per semplificare al tester l'inserimento di una data in formato 'YYYY-MM-DD'.

- Per iniziare la simulazione si selezioni il pulsante "TEST GENERA TABELLONE".
- Inserire nel campo una data nel formato YYYY-MM-DD oppure selezionare il pulsante "Data Odierna" per autogenerare la data odierna.
- Selezionare il pulsante "Genera Tabellone" per utilizzare la funzione GeneraTabellone con la data presente nel campo riempito precedentemente



Debugging:

TestCase[2] e TestCase[3] sono risultati FAIL poichè la variabile privata 'textDate' risultava 'null' durante il 1° tentativo di test della funzione, con formato o dati in input non validi. Non si era considerato inizialmente di catturare l'eccezione di tipo NullPointerException per la funzione GeneraTabellone quando essa veniva chiamata dalla GUI per tale valore.

TEST SUITE DI “Genera Tabellone” compilata.

Test Case ID	Descrizione	Classi di equivalenza coperte	Pre-condizioni	Input	Output Attesi	Post-condizioni Attese	Output Ottenuti	Post-condizioni Ottenute	Esito (FAIL, PASS)
1	Tutti input validi	Data valida Torneo che ha raggiunto il massimo numero di iscritti senza tabellone	E' presente nel database un torneo, con data di inizio 7 giorni successiva alla data odierna, il quale tabellone non è stato ancora generato e ha raggiunto il numero massimo di partecipanti.	{Data: "2022-06-03"}	Email inviata a: 'emailGiocatore'	I giocatori sono assegnati ai match, ricevono il tabellone per email e il torneo risulta in corso	Email inviata a: silvio@email.it Email inviata a: Ale@email.it Email inviata a: giulia@email.it Email inviata a: ernia@email.it Email inviata a: serena@email.it Email inviata a: balsa@email.it Email inviata a: lore@email.it Email inviata a: parta@email.it	I tornei la cui data di inizio è successiva di 7 giorni alla data inserita hanno checkTabellone=1 e gli iscritti al torneo sono stati assegnati ai match contenenti il tuno = 0 e le informazioni del torneo e dei suoi match sono state inviate tramite email ai partecipanti .	PASS
2	Formato input Data valido con valore non valido. Torneo che ha raggiunto il	Data con formato valido ma valori non validi [ERROR]		{Data: "2022-12-32"}	'Inserimento input errato, inserire in formato yyyy-mm-dd'		Exception in thread "AWT-EventQueue-0" java.lang.NullPointerException: Cannot invoke "java.sql.Date.getDate()" because "temp" is null at database.TorneoDAO.getID Tornei(TorneoDAO.java:24)		FAIL

	massimo numero di iscritti senza tabellone					at control.GestioneTornei.generaTabellone(GestioneTornei.java:28) at boundary.GUITabelloneFuncMenu\$3.actionPerformed(GUITabelloneFuncMenu.java:96) at java.desktop/javax.swing.AbstractButton.fireActionPerformed(AbstractButton.java:1972) at java.desktop/javax.swing.AbstractButton\$Handler.actionPerformed(AbstractButton\$Handler.java:2313) at java.desktop/javax.swing.DefaultButtonModel.fireActionPerformed(DefaultButtonModel.java:405) at java.desktop/javax.swing.DefaultButtonModel.setPressed(DefaultButtonModel.java:262) at java.desktop/javax.swing.plaf.basic.BasicButtonListener.mouseReleased(BasicButtonListener.java:279) at java.desktop/java.awt.Component.processMouseEvent(Component.java:6626) at java.desktop/javax.swing.JComponent.processMouseEvent(JComponent.java:3389) at java.desktop/java.awt.Component.processEvent(Component.java:6391)	
--	--	--	--	--	--	--	--

```
at  
java.desktop/java.awt.Container.processEvent(Container.java:2266)  
at  
java.desktop/java.awt.Component.dispatchEventImpl(Component.java:5001)  
at  
java.desktop/java.awt.Container.dispatchEventImpl(Container.java:2324)  
at  
java.desktop/java.awt.Component.dispatchEvent(Component.java:4833)  
at  
java.desktop/java.awt.LightweightDispatcher.retargetMouseEvent(Container.java:4948)  
at  
java.desktop/java.awt.LightweightDispatcher.processMouseEvent(Container.java:4575)  
at  
java.desktop/java.awt.LightweightDispatcher.dispatchEvent(Container.java:4516)  
at  
java.desktop/java.awt.Container.dispatchEventImpl(Container.java:2310)  
at  
java.desktop/java.awt.Window.dispatchEventImpl(Window.java:2780)  
at  
java.desktop/java.awt.Component.dispatchEvent(Component.java:4833)  
at  
java.desktop/java.awt.EventQueue.dispatchEventImpl(EventQueue.java:773)  
at  
java.desktop/java.awt.EventQueue
```

```
tQueue$4.run(EventQueue.java:722)
    at java.desktop/java.awt.EventQueue$4.run(EventQueue.java:716)
    at java.base/java.security.AccessController.doPrivileged(AccessController.java:399)
    at java.base/java.security.ProtectionDomain$JavaSecurityAccessImpl.doIntersectionPrivilege(ProtectionDomain.java:86)
    at java.base/java.security.ProtectionDomain$JavaSecurityAccessImpl.doIntersectionPrivilege(ProtectionDomain.java:97)
    at java.desktop/java.awt.EventQueue$5.run(EventQueue.java:746)
    at java.desktop/java.awt.EventQueue$5.run(EventQueue.java:744)
    at java.base/java.security.AccessController.doPrivileged(AccessController.java:399)
    at java.base/java.security.ProtectionDomain$JavaSecurityAccessImpl.doIntersectionPrivilege(ProtectionDomain.java:86)
    at java.desktop/java.awt.EventQueue.dispatchEvent(EventQueue.java:743)
    at java.desktop/java.awt.EventQueue.pumpOne
```

							EventForFilters(EventDispatchThread.java:203) at java.desktop/java.awt.EventQueue\$4.dispatch(EventDispatchThread.java:124) at java.desktop/java.awt.EventQueue\$4.dispatchEvent(EventDispatchThread.java:113) at java.desktop/java.awt.EventQueue\$4.dispatchEvent(EventDispatchThread.java:109) at java.desktop/java.awt.EventQueue\$4.dispatchEvent(EventDispatchThread.java:101) at java.desktop/java.awt.EventQueue\$4.dispatchEvent(EventDispatchThread.java:90)	
3	Formato input Data non valido	Data con formato non valido [ERROR] Torneo che ha raggiunto il massimo numero di iscritti senza tabellone		{Data: “2022/31- 12”}	‘Inserimento input errato, inserire in formato yyyy- mm-dd’		Exception in thread "AWT- EventQueue-0" java.lang.NullPointerException: Cannot invoke "java.sql.Date.getDate()" because "temp" is null at database.TorneoDAO.getIDTornei(TorneoDAO.java:24) at control.GestioneTornei.generaTabellone(GestioneTornei.java:28) at boundary.GUITabelloneFuncMenu\$3.actionPerformed(GUITabelloneFuncMenu.java:96) at java.desktop/javax.swing.AbstractButton.fireActionPerformed	FAIL

```
formed(AbstractButton.java:1972)
        at
java.desktop/javax.swing.AbstractButton$Handler.actionPerformed(AbstractButton.java:2313)
        at
java.desktop/javax.swing.DefaultButtonModel.fireActionPerformed(DefaultButtonModel.java:405)
        at
java.desktop/javax.swing.DefaultButtonModel.setPressed(DefaultButtonModel.java:262)
        at
java.desktop/javax.swing.plaf.basic.BasicButtonListener.mouseReleased(BasicButtonListener.java:279)
        at
java.desktop/java.awt.Component.processMouseEvent(Component.java:6626)
        at
java.desktop/javax.swing.JComponent.processMouseEvent(JComponent.java:3389)
        at
java.desktop/java.awt.Component.dispatchEvent(Component.java:6391)
        at
java.desktop/java.awt.Container.dispatchEvent(Container.java:2266)
        at
java.desktop/java.awt.Component.dispatchEventImpl(Component.java:5001)
        at
java.desktop/java.awt.Container.dispatchEventImpl(Container.java:2324)
```

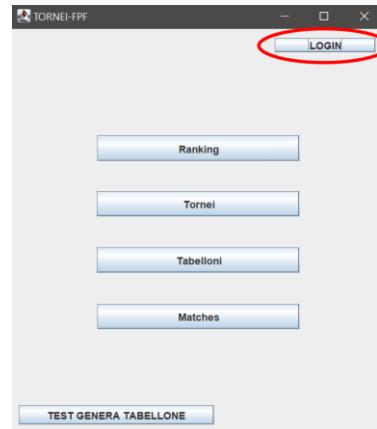
```
at  
java.desktop/java.awt.Com  
ponent.dispatchEvent(Com  
ponent.java:4833)  
at  
java.desktop/java.awt.Light  
weightDispatcher.retarget  
MouseEvent(Container.java  
:4948)  
at  
java.desktop/java.awt.Light  
weightDispatcher.process  
MouseEvent(Container.java  
:4575)  
at  
java.desktop/java.awt.Light  
weightDispatcher.dispatch  
Event(Container.java:4516)  
at  
java.desktop/java.awt.Cont  
ainer.dispatchEventImpl(Co  
ntainer.java:2310)  
at  
java.desktop/java.awt.Win  
dow.dispatchEventImpl(Wi  
ndow.java:2780)  
at  
java.desktop/java.awt.Com  
ponent.dispatchEvent(Com  
ponent.java:4833)  
at  
java.desktop/java.awt.Even  
tQueue.dispatchEventImpl(  
EventQueue.java:773)  
at  
java.desktop/java.awt.Even  
tQueue$4.run(EventQueue.  
java:722)  
at  
java.desktop/java.awt.Even  
tQueue$4.run(EventQueue.  
java:716)  
at  
java.base/java.security.Acc  
essController.doPrivileged(  
AccessController.java:399)  
at  
java.base/java.security.Prot
```

```
ectionDomain$JavaSecurity
AccessImpl.doIntersectionP
rivilege(ProtectionDomain.j
ava:86)
        at
java.base/java.security.Prot
ectionDomain$JavaSecurity
AccessImpl.doIntersectionP
rivilege(ProtectionDomain.j
ava:97)
        at
java.desktop/java.awt.Event
Queue$5.run(EventQueue.
java:746)
        at
java.desktop/java.awt.Event
Queue$5.run(EventQueue.
java:744)
        at
java.base/java.security.Acc
essController.doPrivileged(
AccessController.java:399)
        at
java.base/java.security.Prot
ectionDomain$JavaSecurity
AccessImpl.doIntersectionP
rivilege(ProtectionDomain.j
ava:86)
        at
java.desktop/java.awt.Event
Queue.dispatchEvent(EventQueu
e.java:743)
        at
java.desktop/java.awt.Event
DispatchThread.pumpOneEven
tForFilters(EventDispatchTh
read.java:203)
        at
java.desktop/java.awt.Event
DispatchThread.pumpEven
tsForFilter(EventDispatchTh
read.java:124)
        at
java.desktop/java.awt.Event
DispatchThread.pumpEven
tsForHierarchy(EventDisp
achThread.java:113)
```

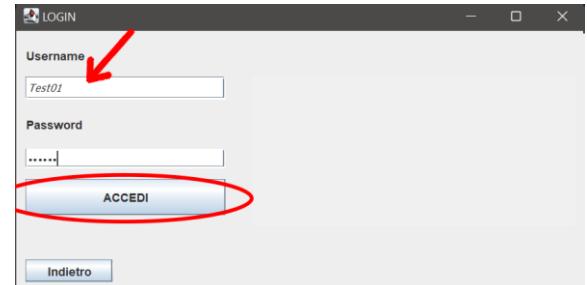
4	Tutti input validi, presenza di un torneo valido ma con numero di iscritti diverso dal massimo numero di partecipanti.	Data valida Torneo senza tabellone ma che non ha raggiunto il massimo numero di iscritti	E' presente nel database un torneo, con data di inizio 7 giorni successiva alla data odierna, il quale tabellone non è stato ancora generato ma non ha raggiunto il numero massimo di partecipanti.	{Data: "2022-07-03"}	'TORNEO <i>nomeTorneo</i> <i>edizioneTorneo</i> [<i>data inizio torneo</i>] CANCELLATO'	Il torneo è terminato	[TORNEO Birra Moretti Estate II [2022-07-10] CANCELLATO]	Il checkTabellone del torneo è pari ad 2.
5	Tutti input validi, NESSUNA presenza nel database di tornei senza tabellone	Data valida Non è presente nel database un torneo senza tabellone	Non è presente nel database alcun torneo senza tabellone o che la sua data di inizio sia 7 giorni successiva alla data odierna.	{Data: "2022-06-22"}	'Nessun tornei inizia tra una settimana'		Nessun torneo inizia tra una settimana	PASS

Istruzioni per utilizzare la funzione “Registra Risultato Match”

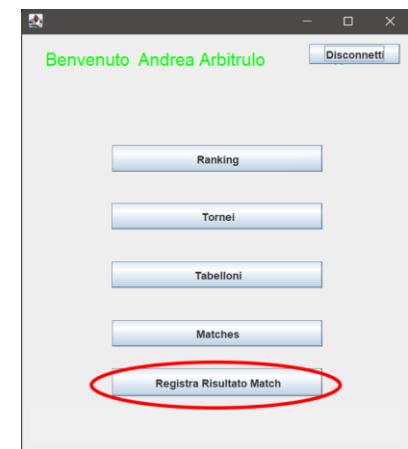
- Per iniziare la simulazione si selezioni il pulsante “LOGIN”.



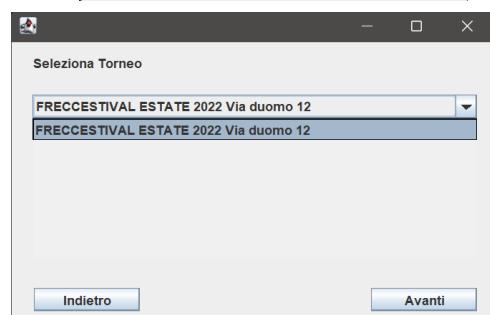
- Si completino i campi Username e Password, per simulare l'utilizzo della funzione da parte di un Arbitro riempire il campo Username: 'Test01' e per il campo Password: 'Pass01'
- Dopo aver riempito i campi si proceda selezionando il pulsante “ACCEDI”



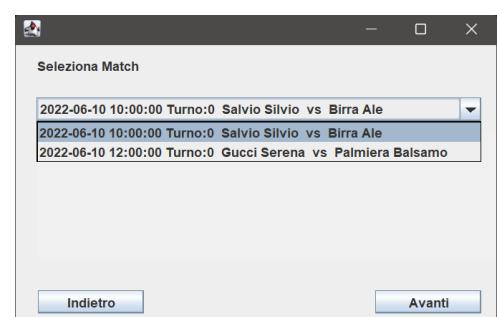
- Selezionare il pulsante “Registra Risultato Match”



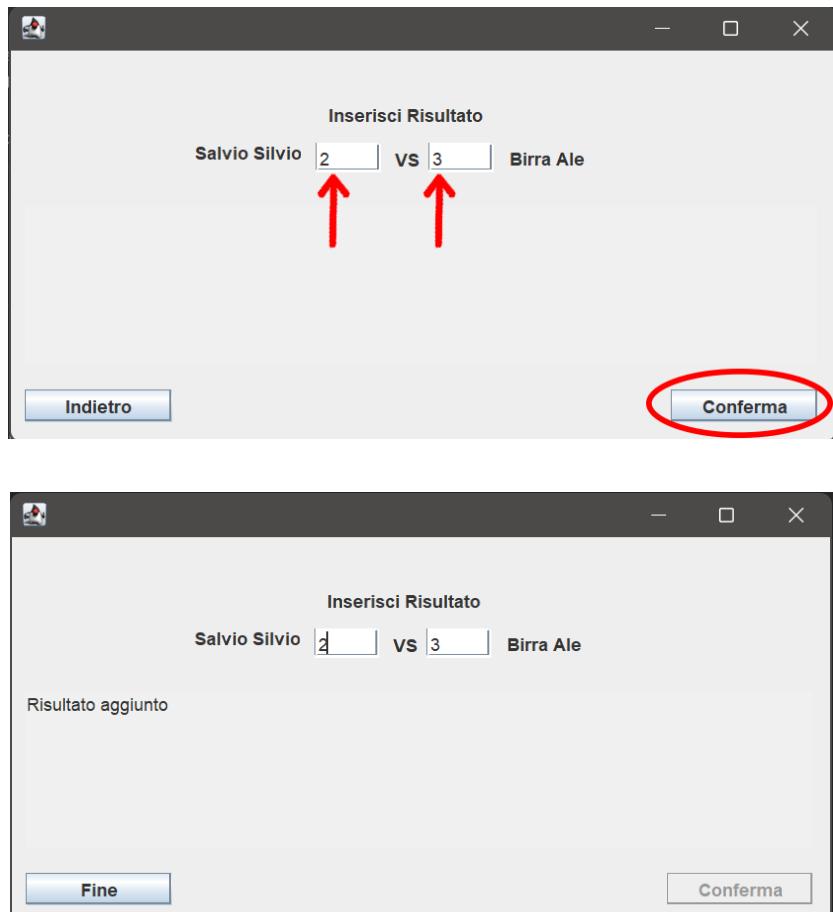
- Scegliere, utilizzando il menu' a tendina, un torneo tra quelli assegnati all'arbitro e selezionare “Avanti”



- Scegliere ,utilizzando il menu' a tendina, un match tra quelli supervisionati dall'arbitro e selezionare “Avanti”



- Riempire i campi con il risultato per entrambi i giocatori e selezionare il pulsante “Conferma”



Debugging:

TestCase[2]

Per portare il database in uno stato in cui l’arbitro fosse assegnato alla finale di un torneo , si è effettuata l’ autenticazione con un altro arbitro per assegnare il risultato ai match rimanenti del torneo su cui si stava facendo la fase di testing . Durante quest’operazioni non si è riusciti ad assegnare il risultato dei match poichè l’interfaccia ci segnalava che non erano disponibili match attivi assegnati all’arbitro. Si è consultato il database per confermare che i match non fossero stati alterati e si proceduto alla fase di debugging.Si è scoperto che nella funzione della classe Torneo getMatchArbitro(String tessera) la condizione

```
this.listaMatch.getSupervisore() == tessera
```

causasse questo problema.

Dopo aver corretto la condizione con this.listaMatch.getSupervisore().equals(tessera) il TestCase ha avuto successo.

TestCase[7]

L’interfaccia invece di dare un segnale di errore quando si voleva provare a registrare il risultato di un match ci permetteva di andare avanti selezionando un torneo . Tuttavia non appariva nessun torneo disponibile nella combobox

Tramite il debugging si è scoperto che nel caso l’arbitro fosse assegnato a tornei nel sistema e fossero o conclusi o non ancora iniziati , il programma ha un comportamento inaspettato.

Per risolvere il problema si è aggiunto un controllo aggiuntivo ListaTornei.size() == 0

TEST SUITE DI “Registra Risultato Match” compilata.

Test Case ID	Descrizione	Classi di equivalenza coperte	Pre-condizioni	Input	Output Attesi	Post-condizioni Attese	Output Ottenuti	Post-condizioni Ottenute	Esito (FAIL, PASS)
1	Tutti gli input sono validi Il match non è una finale	Risultato Giocatore 1 valido Risultato Giocatore 2 valido Match valido Torneo Valido	Arbitro è autenticato , è assegnato ad un torneo in corso ed esiste almeno un match assegnatogli il cui risultato non è ancora stato aggiunto ..	{ Torneo: “FRECCESTIVAL ESTATE 2022 Via duomo 12” Match: “2022-06-10 10:00:00 Turno: 1 Salvio Silvio vs Birra Ale” Risultato Giocatore 1 : “3” Risultato Giocatore 2 : “0”}	Risultato aggiunto	il risultato del match è assegnato. Il Vincitore avanza al prossimo match	Risultato aggiunto	il risultato del match è assegnato. Il Vincitore avanza al prossimo match	PASS
2	Tutti gli input sono validi Il match è una finale	Risultato Giocatore 1 valido Risultato Giocatore 2 valido Match valido Torneo valido	Arbitro è autenticato, è assegnato ad un torneo in corso ed esiste almeno una “finale” di torneo assegnatogli il cui risultato non è ancora stato aggiunto .	{ Torneo: “FRECCESTIVAL ESTATE 2022 Via duomo 12” Match: “2022-06-14 19:00:00 Turno: 3 Salvio Silvio vs Gucci Serena” Risultato Giocatore 1 : “0” Risultato Giocatore 2 : “3”}	Risultato aggiunto	il risultato del match è assegnato. Il Torneo è concluso.			FAIL
3	Risultato Giocatore 1 con	Formato Risultato	Arbitro è autenticato , è assegnato ad	{ Torneo: “ FRECCESTIVAL ESTATE 2022 Via duomo 12”	Formato Input invalido	il risultato del match	Formato Input invalido	il risultato del match non è	PASS

	formato non valido	Giocatore 1 non valido [ERROR] Risultato Giocatore 2 valido Match valido Torneo valido	un torneo in corso ed esiste almeno un match assegnatogli il cui risultato non è ancora stato aggiunto.	Match: "2022-06-10 11:00:00 Turno:1 Pascale Giulia vs Disco Ernia" Risultato Giocatore 1 : "tre" Risultato Giocatore 2 : "0"}		non è stato assegnato		stato assegnato	
4	Risultato Giocatore 1 con formato valido e valori non validi [ERROR]	Risultato Giocatore 1 con formato valido e valori non validi [ERROR] Risultato Giocatore 2 valido Match valido Torneo valido	Arbitro è autenticato , è assegnato ad un torneo in corso ed esiste almeno un match assegnatogli il cui risultato non è ancora stato aggiunto.	{ Torneo: " FRECCESTIVAL ESTATE 2022 Via duomo 12" Match: "2022-06-10 11:00:00 Turno:1 Pascale Giulia vs Disco Ernia" Risultato Giocatore 1 : "-3" Risultato Giocatore 2 : "0"}	Valore del risultato non valido	il risultato del match non è stato assegnato	Risultato aggiunto	il risultato del match è assegnato. Il Vincitore avanza al prossimo match	PASS
5	Risultato Giocatore 2 con formato non valido	Risultato Giocatore 1 valido Formato Risultato Giocatore 2 non valido [ERROR] Match valido Torneo valido	Arbitro è autenticato , è assegnato ad un torneo in corso ed esiste almeno un match assegnatogli il cui risultato non è ancora stato aggiunto.	{ Torneo: " FRECCESTIVAL ESTATE 2022 Via duomo 12" Match: "2022-06-10 11:00:00 Turno:1 Pascale Giulia vs Disco Ernia" Risultato Giocatore 1 : "3" Risultato Giocatore 2 : "zero"}	Formato del risultato non valido	il risultato del match non è stato assegnato	Formato Input invalido	il risultato del match non è stato assegnato	PASS
6	Risultato Giocatore 2 con	Risultato Giocatore 1 valido	Arbitro è autenticato ed esiste almeno un match	{ Torneo: " FRECCESTIVAL ESTATE 2022 Via duomo 12"	Valore del risultato non valido	il risultato del match	Risultato aggiunto	il risultato del match	PASS

	formato valido e valori non validi	Risultato Giocatore 2 con formato valido e valori non validi [ERROR] Match valido Torneo valido	assegnatogli il cui risultato non è ancora stato aggiunto .	Match: "2022-06-10 11:00:00 Turno:1 Pascale Giulia vs Disco Ernia" Risultato Giocatore 1 : "3" Risultato Giocatore 2 : "-2"}		non è stato assegnato		è assegnato. Il Vincitore avanza al prossimo match	
7	Arbitro non è assegnato a nessun torneo in corso	Torneo non valido[ERROR]	Arbitro è autenticato. Non è assegnato a nessun torneo		L'arbitro non e' assegnato a nessun torneo attivo	Nessuna modifica	L'arbitro non e' assegnato a nessun match attivo	Nessuna modifica	FAIL
8	L'Arbitro non è assegnato a nessun match attivo del torneo	Torneo valido Match non valido[ERROR]	L'Arbitro non è assegnato a nessun match attivo del torneo	{Torneo: " FRECCESTIVAL ESTATE 2022 Via duomo 12"}	L'arbitro non e' assegnato a nessun match attivo	Nessuna modifica	L'arbitro non e' assegnato a nessun match attivo	Nessuna modifica	PASS
9	Tutti gli input sono validi	Risultato Giocatore 1 valido Risultato Giocatore 2 valido[IF RISULTATO GIOCATORE 1 == RISULTATO GIOCATORE 2] Torneo valido Match valido	Arbitro è autenticato ed esiste almeno un match assegnatogli il cui risultato non è ancora stato aggiunto .	{ Torneo: " FRECCESTIVAL ESTATE 2022 Via duomo 12" Match: "2022-06-10 11:00:00 Turno:1 Pascale Giulia vs Disco Ernia " Risultato Giocatore 1 : "3" Risultato Giocatore 2 : "3"}	Risultato aggiunto errato	il risultato del match non è stato assegnato	Risultato aggiunto errato	il risultato del match non è stato assegnato	PASS

