PROGETTO

D-RISK

Ing. del Software A.A 2020/2021

Salvatore Marotta - 844795

Alessio Mognato - 844953

## Indice

* Analisi e progettazione
  + [Analisi dei requisiti](#_heading=h.3znysh7)
  + [Diagramma dei casi d’uso](#_heading=h.eb906rz3uiec)
  + [Modello di dominio](#_heading=h.slh175kn8odu)
* [Casi d’uso dettagliati](#_heading=h.tyjcwt)
  + [UC01 - Registrazione al sito](#_heading=h.uxjanf55m8mb)
  + [UC02 - Accesso al sito](#_heading=h.cw286wvj1xr5)
  + [UC03 - Giocare una partita](#_heading=h.8bqh3zw61dib)
  + [UC04- Creare una partita](#_heading=h.44sinio)
  + [UC05 - Configurare una partita](#_heading=h.1y810tw)
  + [UC06 - Preparare una partita](#_heading=h.2xcytpi)
  + [UC07 - Svolgere una partita](#_heading=h.w5erga2g9hpv)
  + [UC08 - Concludere una partita](#_heading=h.fb6nfqfwi55r)
* Diagrammi di sequenza:
  + [UC01 - Diagramma di sequenza](#_heading=h.kn3d1xqeuzjq)
  + [UC02 - Diagramma di sequenza](#_heading=h.3rdcrjn)
  + [UC03 - Diagramma di sequenza](#_heading=h.35nkun2)
  + [UC04 - Diagramma di sequenza](#_heading=h.3j2qqm3)
  + [UC05 - Diagramma di sequenza](#_heading=h.4i7ojhp)
  + [UC06 - Diagramma di sequenza](#_heading=h.3whwml4)
  + [UC07 - Diagramma di sequenza](#_heading=h.g29nmde9u0v)
  + [UC08 - Diagramma di sequenza](#_heading=h.ykwew64l3623)
* [Contratti:](#_heading=h.34g0dwd)
  + [CO1 - registraProfilo](#_heading=h.34g0dwd)
  + [CO2 - giocaPartita](#_heading=h.1jlao46)
* [Diagrammi di attività:](#_heading=h.1pxezwc)
  + [Diagramma di attività: Creare una partita](#_heading=h.1pxezwc)
  + [Diagramma di attività: Configurazione di una partita](#_heading=h.2p2csry)
* [Pattern GRASP e Architetturali](#_heading=h.i7fs70nzxmsi)
* [Architettura dell'applicazione](#_heading=h.2afsuuq9p3pt)
  + [Strato UI](#_heading=h.hrd0jahf0x36)
  + [Strato Controller](#_heading=h.1jyixk7uds5x)
  + [Strato Model e DAO](#_heading=h.147n2zr)
* [Diagrammi di sequenza di progettazione](#_heading=h.f5qmo6wnzjzp)
* [Diagramma di macchina a stati](#_heading=h.avvi0snz8a92)
* [Note sulla consegna del progetto](#_heading=h.kr9jmvk52j2u)

## Analisi dei requisiti:

Descrizione del progetto:

D-Risk è un progetto basato sulla creazione di un'applicazione web-based per una versione dinamica del Risk game (Risiko).  
In particolare a differenza del Risiko standard i giocatori sono da 2 a 6.   
La mappa sarà configurabile fino ad un massimo di 42 territori, raggruppabili in continenti e con la possibilità di modificare anche la cardinalità dei confini.

Obiettivo del progetto:

Il progetto implementa tutte le funzionalità base del Risiko classico più le caratteristiche di configurazione sopracitate.  
L’obiettivo di D-Risk, come per il classico Risiko, prevede di raggiungere un obiettivo comune (conquista dei 2/3 del mondo) oppure di raggiungere l’obiettivo personale assegnato ad inizio partita. Il gioco ha una mappa preimpostata iniziale con tre possibili livelli di difficoltà (che influenzano le modalità di conquista di un territorio).

## Attori:

* Utente
* Giocatore
* Sistema

## Requisiti funzionali:

* Registrati al sito:
  + Un utente può registrarsi al sito fornendo email, nickname e password.
* Accesso al sito:
  + Un utente può accedere al sito fornendo e-mail e password.
  + Un utente può visualizzare il proprio profilo.
  + Un utente può iniziare una partita.
* Giocare una partita:
  + L’utente deve essere autenticato. (deve aver effettuato l’accesso)
  + Un utente può entrare in una partita già creata scegliendo una lobby
* Creare una partita:
  + Un utente può creare una partita inizializzando una lobby di gioco.
  + Un utente può configurare la partita.
* Configurare una partita:
  + L’utente può scegliere numero di giocatori (da 2 a 6)
  + L’utente può scegliere numero di territori e raggruppamenti in continenti (rispettivamente massimo 42 e 6)
  + L’utente può scegliere il numero di confini tra i territori (da 1 a 4)
* Preparare una partita:
  + Il sistema assegna un ordine di turno ad ogni giocatore.
  + Il giocatore ha un obiettivo da raggiungere, assegnato casualmente ad inizio partita dal sistema.
  + Il giocatore sceglie il colore dell’esercito da utilizzare nella partita.
  + Il sistema assegna ad ogni giocatore le armate in base alla configurazione della partita.
  + Il sistema assegna i territori a tutti i giocatori.
  + Il giocatore predispone le armate iniziali (3 alla volta) nei territori assegnati dal sistema.
* Svolgere una partita:
  + La partita deve essere preparata.
  + Prima fase: RINFORZI
    - Acquisizione delle armate di rinforzo in base e posizionamento:
      * Al numero di territori posseduti
      * Al bonus continenti
      * Al bonus carte
  + Seconda fase: COMBATTIMENTI
    - Un giocatore può attaccare da un territorio da lui occupato uno avversario confinante (via terra o mare):
      * Il giocatore attaccante dichiara il numero di truppe da usare nell’attacco (massimo 3).
      * Il giocatore difensore dichiara il numero di truppe da usare nella difesa (massimo 3).
      * Lancio dei dadi in base al numero di truppe scelte (ratio 1:1)
      * Si confrontano i dadi in ordine di grandezza (uno a uno) e si ritira/toglie (attaccante/difensore) un'armata in base al punteggio del dado ottenuto (vince il valore più alto e in caso di pareggio vince sempre il difensore).
      * Se l’attaccante distrugge tutte le armate avversarie su un territorio deve occuparlo con le armate che hanno partecipato all’ultima battaglia, e può trasportarvi, partendo dal territorio da cui ha sferrato l’attacco, tutte le armate che vuole.
  + Terza fase: SPOSTAMENTO STRATEGICO
    - Un giocatore può spostare una parte o tutte le armate da un territorio che possiede ad un altro confinante, sempre sotto il suo controllo.  
      Si può fare un solo spostamento strategico a turno e ogni territorio controllato deve sempre avere almeno una armata. Tale fase conclude il turno del giocatore.
  + Fine del turno:
    - Il giocatore di turno sospende i suoi attacchi e pesca una e una sola carta se ha conquistato almeno un territorio.
    - Il giocatore di turno non è più condizioni di attaccare (avendo tutti i territori controllati con una sola armata)
* Concludere partita:
  + Un giocatore è eliminato dalla partita se, a partire dal quarto turno, non ha più armate disponibili.
  + Il giocatore di turno raggiunge l’obiettivo personale o quello comune, vincendo la partite.

Esemplificazione dei combattimenti:

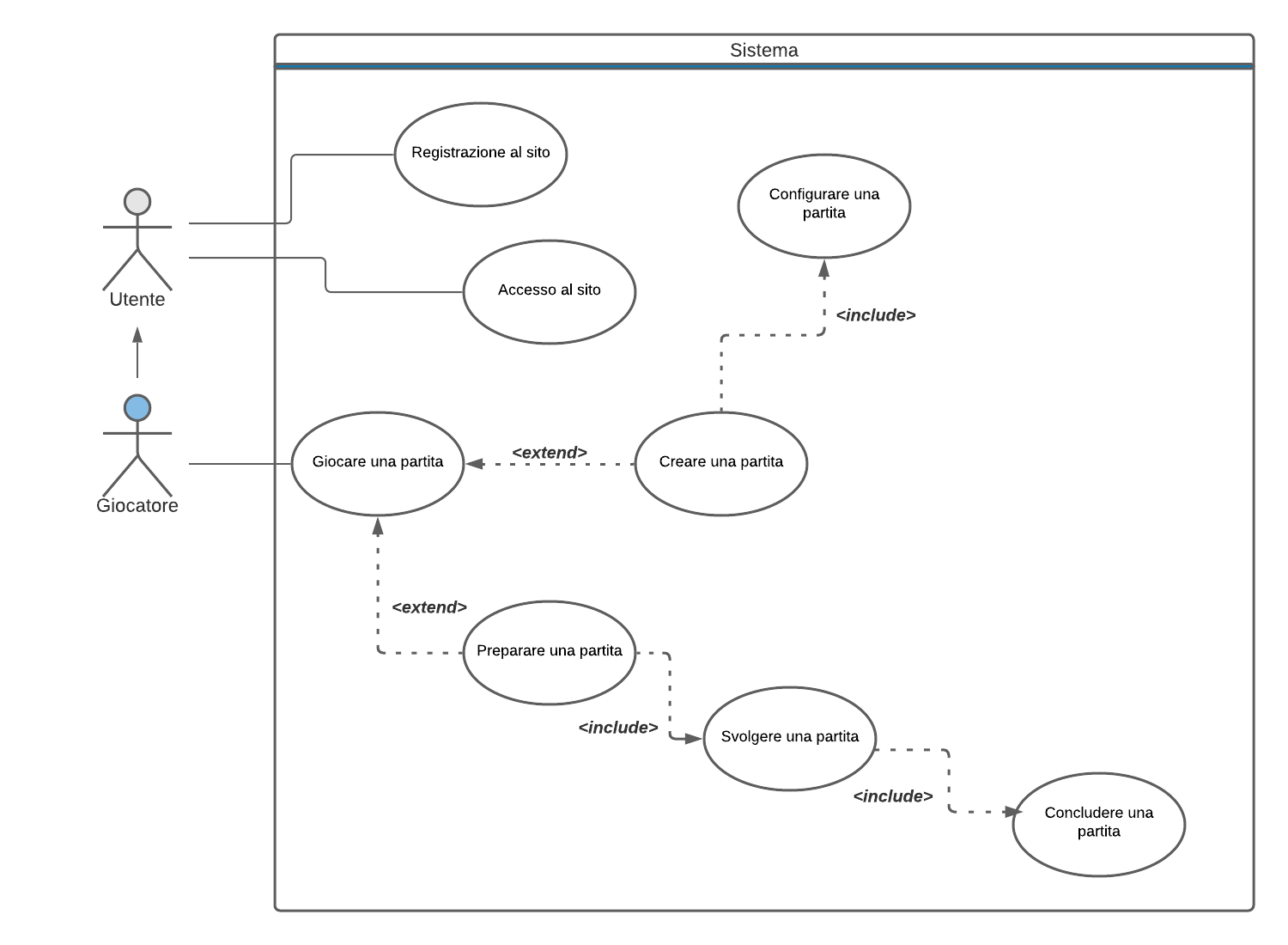
### 

### 

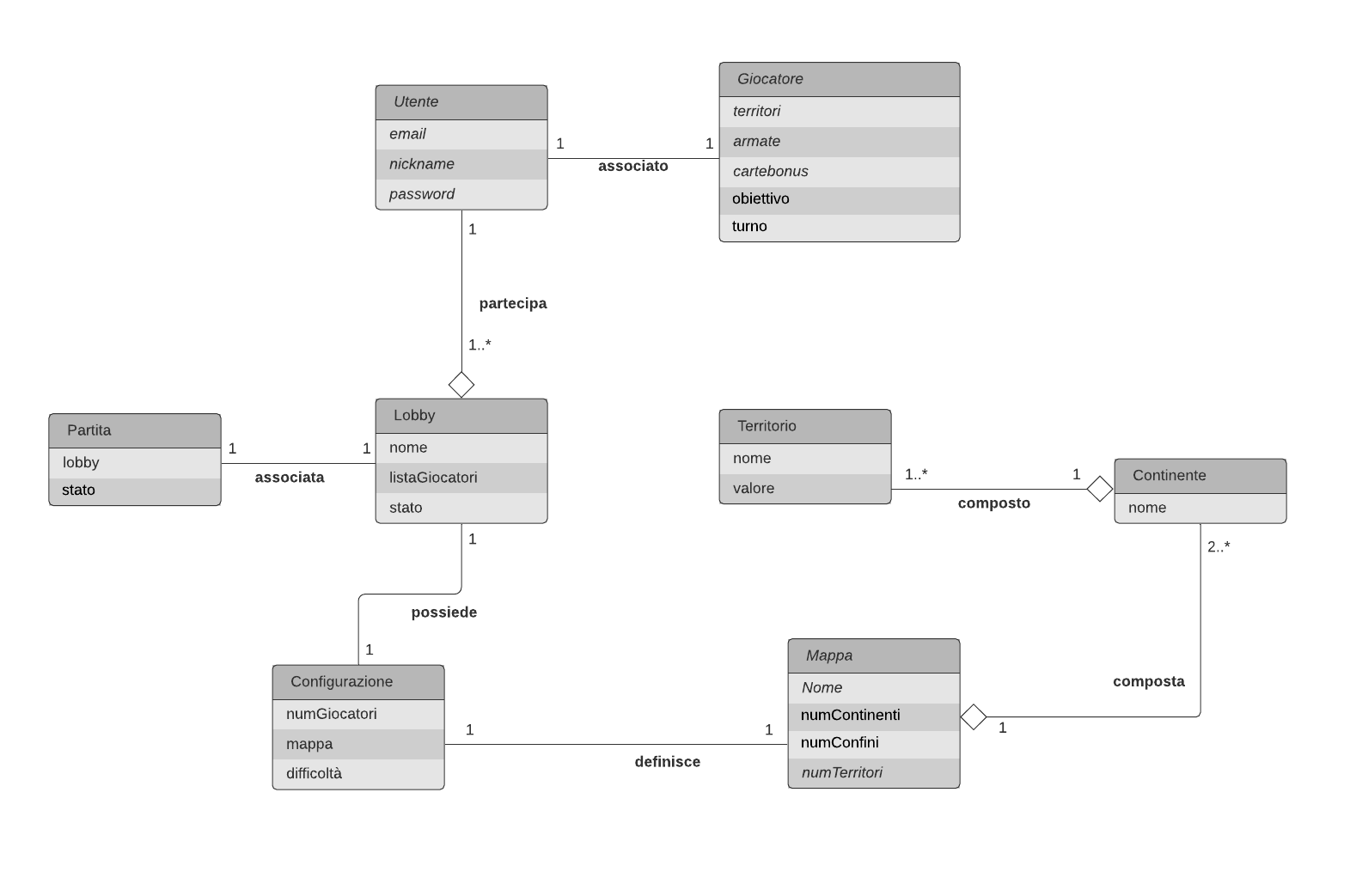
### 

### 

### Diagramma dei casi d’uso:



## Modello di dominio:



## Casi d’uso dettagliati:

## **UC01 - Registrazione al sito**

**Portata:** Applicazione Web D-Risk

**Livello:** Obiettivo utente

**Attore primario:** Utente

**Parti interessate:**

* Utente: vuole registrarsi al sito web con i propri dati..
* Sistema: vuole salvare i dati dei clienti che effettuano una registrazione al sito.

**Pre-condizione:** l’utente non deve aver già effettuato l’accesso al sito.

**Garanzia di successo**: l’utente è registrato nel sistema.

**Scenario Principale di Successo:**

1. Il caso d’uso inizia quando l’utente seleziona “Registrati”.
2. L’utente inserisce i propri dati personali (email, nickname, password) e conferma l’invio dei dati al sistema.
3. Il sistema verifica l’integrità dei dati e crea la nuova utenza relativa al cliente.
4. Il sistema mostra un messaggio di registrazione effettuata con successo.

**Estensioni:**

Fallimento:

1. Dati inseriti dall’utente non validi:
   1. Nickname già esistente.
   2. Email già registrata.

**Requisiti speciali:**

* Requisito non funzionale (sicurezza): il sistema deve garantire privacy e sicurezza nella trasmissione dei dati personali.
* Requisito non funzionale (sicurezza): il sistema deve garantire privacy e sicurezza nella manipolazione e nella conservazione dei dati personali relativi ad un utente..
* Requisito non funzionale (prestazionale): il sistema deve creare l’utenza entro 5 secondi dalla ricezione dei dati utente.

**Elenco delle variabili tecnologiche e dei dati**:

* Il sistema salva i dati delle utenze in una tabella dedicata nel database.
* Il sistema è protetto da attacchi esterni attraverso il captcha di Google.

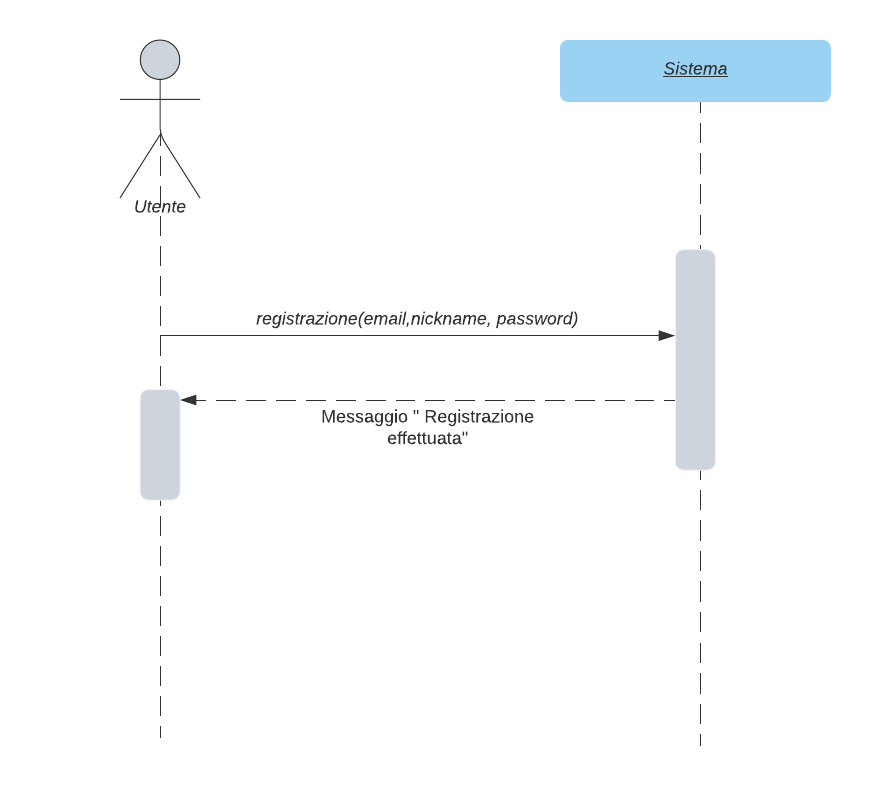
**Frequenza di ripetizione:** Frequente..

### 

### 

### 

### UC01 - Diagramma di sequenza di Sistema:



## **UC02 - Accesso al sito**

**Portata:** Applicazione Web D-Risk

**Livello:** Obiettivo utente

**Attore primario:** Utente

**Parti interessate:**

* Utente: vuole accedere al sistema con le proprie credenziali.
* Sistema: vuole verificare che le credenziali fornite da un cliente siano corrette e vuole tener traccia delle operazioni effettuate dagli utenti.

**Precondizione:** l'utente non è autenticato al sistema ma è registrato

**Garanzia di successo**: l’utente è autenticato al sistema.

**Scenario Principale di Successo:**

1. Il caso d’uso inizia quando il cliente clicca su “Accedi”.
2. L’utente inserisce email e password.
3. Il sistema riconosce le credenziali di accesso.
4. L’utente viene autenticato.

**Estensioni:**

Fallimento:

1. Credenziali di accesso non riconosciute:
   1. 2.email e/o password non presenti nel sistema.

Il sistema mostra un messaggio di errore relativo.

**Requisiti speciali:**

* Requisito non funzionale (sicurezza): il sistema deve garantire privacy e sicurezza nella trasmissione dei dati personali.
* Requisito non funzionale (prestazionale): il sistema deve autenticare l’utente entro 3 secondi.

**Elenco delle variabili tecnologiche e dei dati**:

* Il sistema controlla che le credenziali siano presenti attraverso una query nel database.

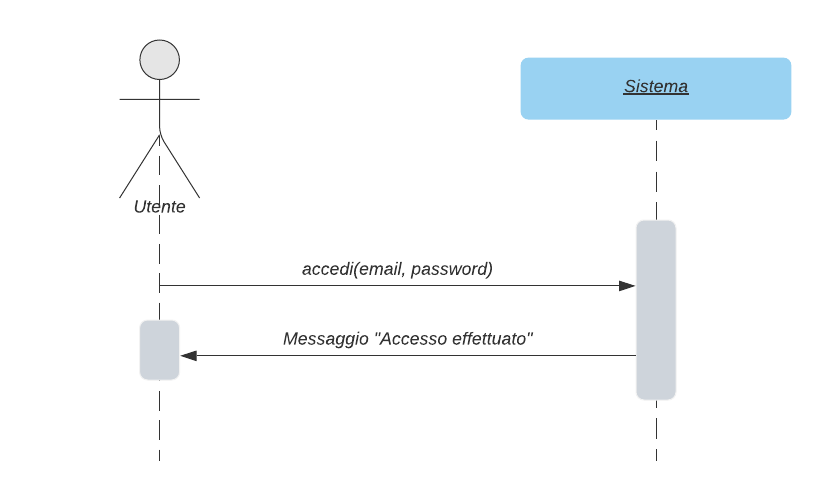
**Frequenza di ripetizione:** Molto frequente.

**Problemi aperti:**

Recupero delle credenziali smarrite.

### 

### UC02 - Diagramma di sequenza di Sistema:



## 

## 

## 

## **UC03 - Giocare una partita**

**Portata:** Applicazione Web D-Risk

**Livello:** Obiettivo utente

**Attore primario:** Utente

**Parti interessate:**

* Utente: vuole iniziare una partita su D-Risk.
* Sistema: vuole garantire l’inizio della partita.

**Pre-condizione:** l’utente deve essere autenticato.

**Garanzia di successo**: l’utente viene inserito in una lobby di gioco.

**Scenario Principale di Successo:**

1. Il caso d’uso inizia quando l’utente seleziona “Gioca una partita”
2. Il sistema mostra una schermata con le lobby disponibili e anche un menu laterale in cui è possibile scegliere “Crea una partita”.
3. L’utente seleziona una lobby e clicca “Partecipa”
4. Il sistema mostra la lobby scelta di gioco.
5. L’utente clicca su “Avvia” (inizia il caso d’uso UC06 - Preparare una partita)

**Estensioni:**

Successo:

1. L’utente segue i passi 1,2 e decide di creare una partita e clicca su “Crea una partita”

Fallimento:

1. L’utente non riesce a connettersi ad una lobby.

**Requisiti speciali:**

* Requisito non funzionale (prestazionale): il sistema deve garantire l’accesso ad una lobby in massimo 5 secondi.
* Requisito non funzionale (sicurezza): il sistema deve garantire privacy e sicurezza nella manipolazione dei dati utente all’interno delle lobby di gioco.

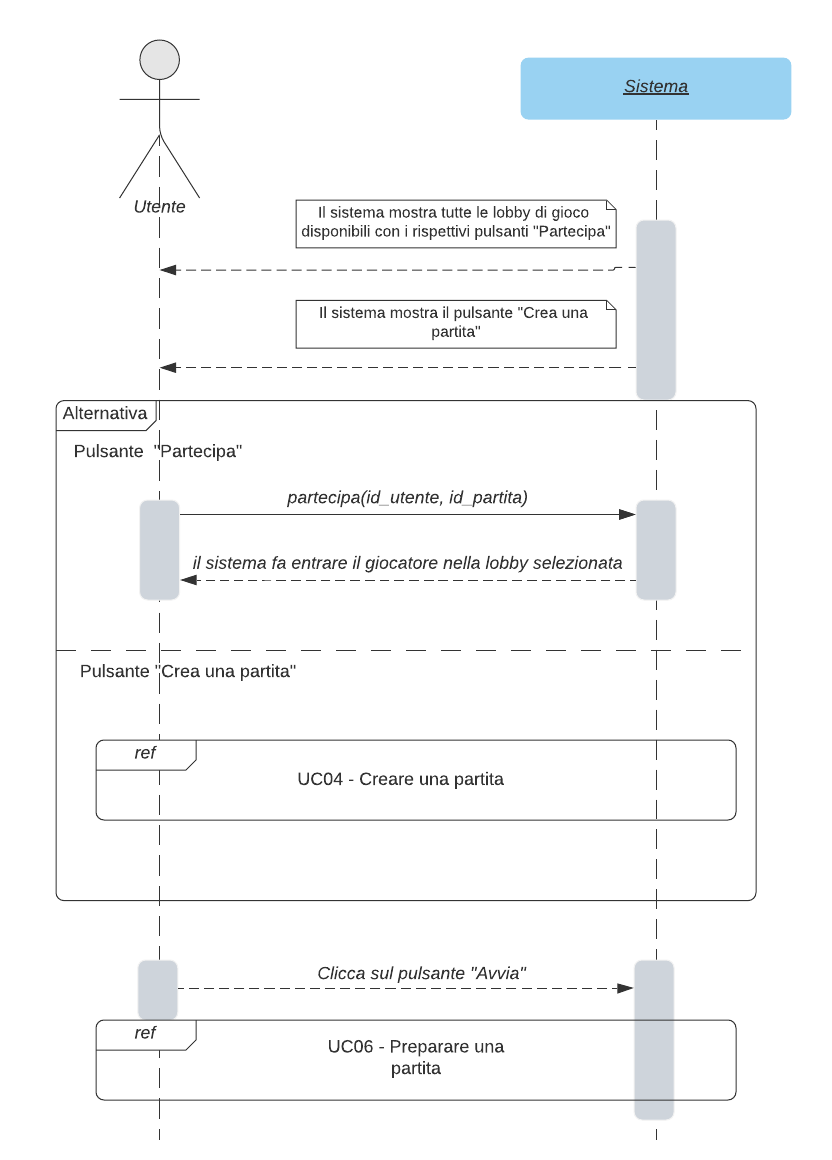
**Elenco delle variabili tecnologiche e dei dati**:

* La lobby di gioco è creata con connessioni Socket e ServerSocket in Java.

**Frequenza di ripetizione:** Molto frequente.

### 

### UC03 - Diagramma di sequenza di Sistema:



## **UC04 - Creare una partita**

**Portata:** Applicazione Web D-Risk

**Livello:** Obiettivo utente

**Attore primario:** Utente

**Parti interessate:**

* Utente: vuole creare la propria partita personalizzata di D-Risk.
* Sistema: vuole permettere all’utente di creare una partita.

**Pre-condizione:** l’utente ha cliccato su “Crea una partita”.

**Garanzia di successo**: generazione di una nuova lobby di gioco configurata dall’utente.

**Scenario Principale di Successo:**

1. Il caso d’uso inizia quando l’utente ha configurato la partita → Caso d’uso “Configurare una partita”
2. Il sistema crea una nuova lobby di gioco con la configurazione scelta dall’utente e rimanda al passo 4 del caso d’uso “Giocare una partita”.

**Estensioni:**

Fallimento:

1. L’utente perde la connessione.
2. L’utente decide di non proseguire con la creazione

Il sistema rimanda alla home.

**Requisiti speciali:**

* Requisito non funzionale (prestazionale): il sistema deve creare una nuova lobby di gioco entro 4 secondi.

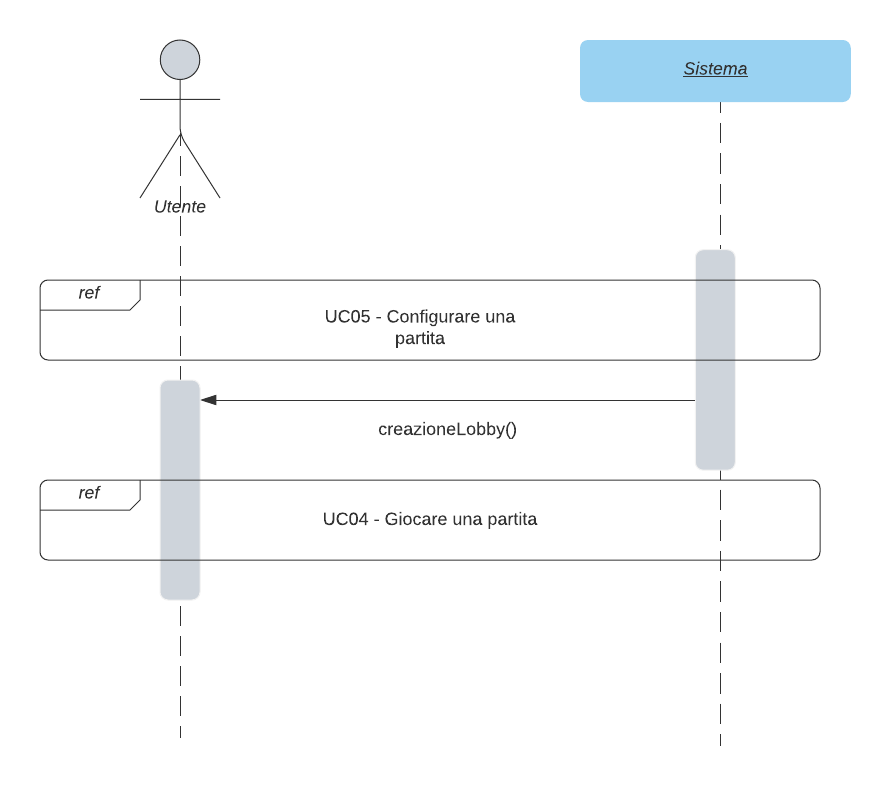
**Elenco delle variabili tecnologiche e dei dati**:

* il sistema utilizza connessioni basate su socket

**Frequenza di ripetizione:** Frequente.

### 

### UC04 - Diagramma di sequenza di Sistema:



## **UC05 - Configurare una partita**

**Portata:** Applicazione Web D-Risk

**Livello:** Obiettivo utente

**Attore primario:** Utente

**Parti interessate:**

* Utente: vuole conoscere i prodotti da mostrare al cliente.
* Sistema: vuole inserire un prodotto nel sistema.

**Pre-condizione:** l’utente ha scelto di creare una partita personalizzata.

**Garanzia di successo**: l’utente ha scelto le preferenze per la partita.

**Scenario Principale di Successo**:

1. Il caso d’uso inizia quando l’admin clicca su “Crea una partita”.
2. Il sistema mostra un menu di personalizzazione della partita, con la possibilità di scegliere:
   1. Numero di continenti
   2. Numero di territori
   3. Numero di confini tra territori
   4. Numero di giocatori
   5. Difficoltà
3. L’utente sceglie la sua configurazione della mappa e clicca su “Salva”.
4. Il sistema restituisce un messaggio di avvenuta creazione e prosegue dal caso d’uso “Creare una partita” punto 2.

**Estensioni:**

Fallimento:

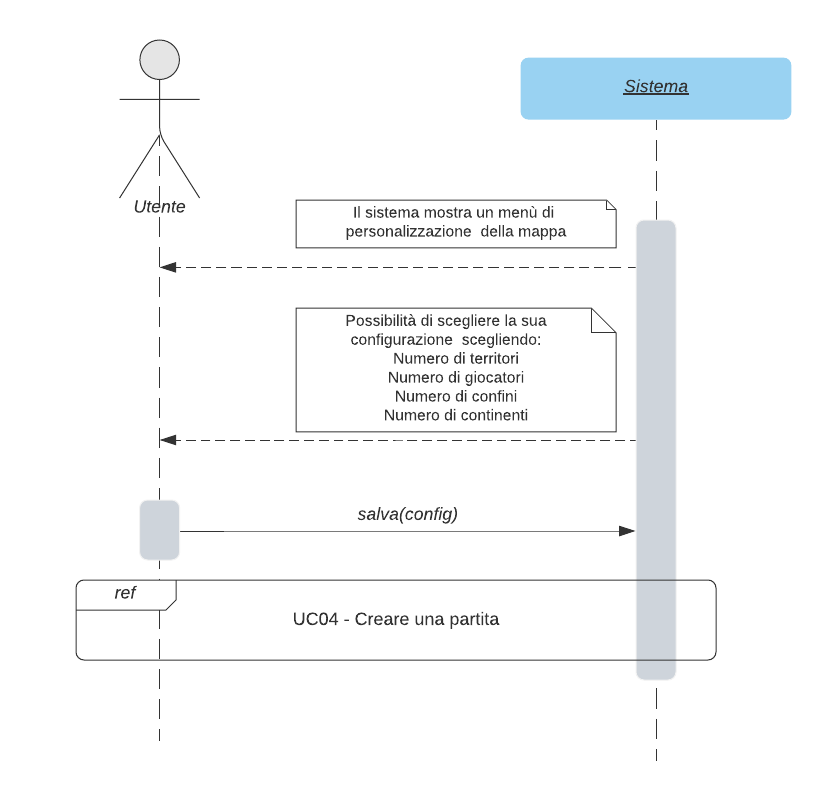
* L’utente perde la connessione.

**Elenco delle variabili tecnologiche e dei dati**:

Il sistema inserisce una nuova partita nella opportuno database attraverso linguaggio SQL.

**Frequenza di ripetizione:** Alta.

### UC05 - Diagramma di sequenza di Sistema:



## **UC06 - Preparare una partita**

**Portata:** Applicazione Web D-Risk

**Livello:** Obiettivo giocatore

**Attore primario:** Giocatore

**Parti interessate:**

* Giocatore: vuole effettuare la fase iniziale di gioco.
* Sistema: inizializza la partita.

**Pre-condizione:** Deve essere stata avviata una partita (associata ad una lobby)

**Garanzia di successo**: Ogni giocatore ha dei territori con delle armate, un obiettivo da raggiungere.

**Scenario principale di successo**:

1. Il caso d’uso inizia quando viene cliccato “Avvia” e il sistema assegna automaticamente un turno a tutti i giocatori.
2. Il giocatore, nel suo turno, sceglie il colore delle armate da utilizzare.
3. Il sistema assegna obiettivo e territori (in base alla configurazione) a tutti i giocatori.
4. Il giocatore, nel suo turno, posiziona fino a 3 armate in uno dei suoi territori fino a terminazione delle armate (per tutti i giocatori).
5. Il sistema avvia il caso d’uso “Svolgere una partita”.

**Estensioni:**

Fallimento:

1. Un giocatore perde la connessione e il sistema mostra un errore terminando la partita.

**Requisiti speciali:**

* Requisito non funzionale (prestazionale): la fase preparatoria a carico del sistema non deve durare più di 8 secondi.

**Elenco delle variabili tecnologiche e dei dati**:

* MapServer per la gestione della mappa e Spring per le interfacce grafiche Java.

**Frequenza di ripetizione:** Molto frequente.

## 

### UC06 - Diagramma di sequenza di Sistema:

## 

## **UC07 - Svolgere una partita**

**Portata:** Applicazione Web D-Risk

**Livello:** Obiettivo giocatore

**Attore primario:** Giocatore

**Parti interessate:**

* Giocatore: svolge la partita con le regole classiche del Risiko.
* Sistema: permette l’interazione del giocatore con la mappa di gioco.

**Pre-condizione:** Deve essere concluso il caso d’uso “Preparare una partita”.

**Garanzia di successo**: Il giocatore svolge il suo turno.

**Scenario principale di successo**:

1. Il caso d’uso inizia quando la fase preparatoria è conclusa.
2. Il giocatore di turno svolge le fasi 1, 2 e 3 del gioco: Rinforzi, Combattimenti e Spostamento Tattico (vedi Requisiti funzionali).
3. Il turno del giocatore si conclude con lo Spostamento Tattico.

**Estensioni:**

Successo:

1. Un giocatore pesca una carta bonus (se ha conquistato almeno un territorio nella fase Combattimenti).
2. Il giocatore clicca su “Termina il turno”.

**Elenco delle variabili tecnologiche e dei dati**:

* MapServer per la gestione della mappa e Spring per le interfacce grafiche Java.

**Frequenza di ripetizione:** Molto frequente.

## 

## 

## 

### UC07 - Diagramma di sequenza di Sistema:

## **UC08 - Concludere una partita**

**Portata:** Applicazione Web D-Risk

**Livello:** Obiettivo giocatore

**Attore primario:** Giocatore

**Parti interessate:**

* Giocatore: vuole poter terminare una partita.
* Sistema: controlla se un giocatore ha il requisito per concludere una partita.

**Precondizione:** Devono essere passati almeno 4 turni di gioco.

**Garanzia di successo**: Il giocatore conclude la sua partita da vincitore o sconfitto.

**Scenario principale di successo**:

1. Il caso d’uso inizia quando uno dei giocatori raggiunge il suo obiettivo personale o l’obiettivo comune.
2. Il Sistema mostra un messaggio con il nome del vincitore.

**Estensioni:**

Successo:

1. Uno dei giocatori raggiunge l'obiettivo segreto globale.
2. Il giocatore viene eliminato poiché non ha più armate disponibili.

Fallimento:

1. Cade la connessione al giocatore durante la partita.

**Requisiti speciali:**

* Requisito non funzionale (sicurezza): il passaggio delle informazioni cliente-sistema deve garantire la privacy.
* Requisito non funzionale (prestazionale): Il giocatore esce dalla partita in massimo 4 secondi.

**Elenco delle variabili tecnologiche e dei dati**:

* MapServer per la gestione della mappa e Spring per le interfacce grafiche Java.

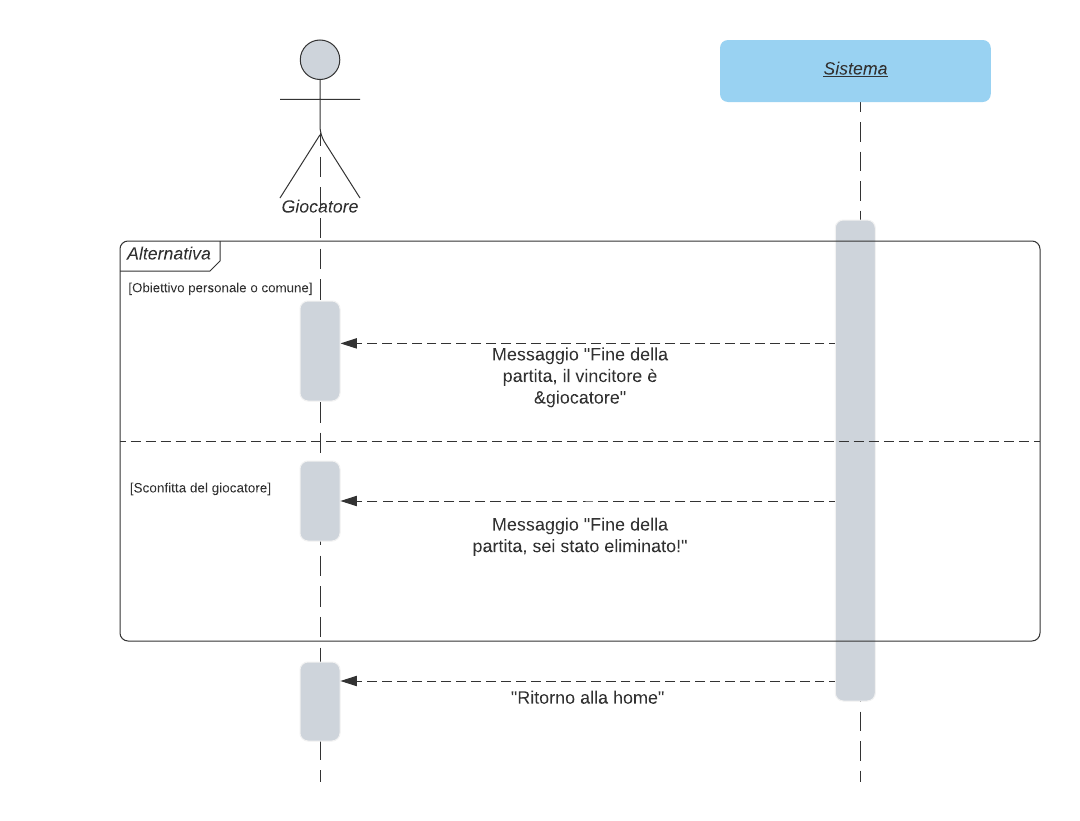
**Frequenza di ripetizione:** Molto frequente.

### 

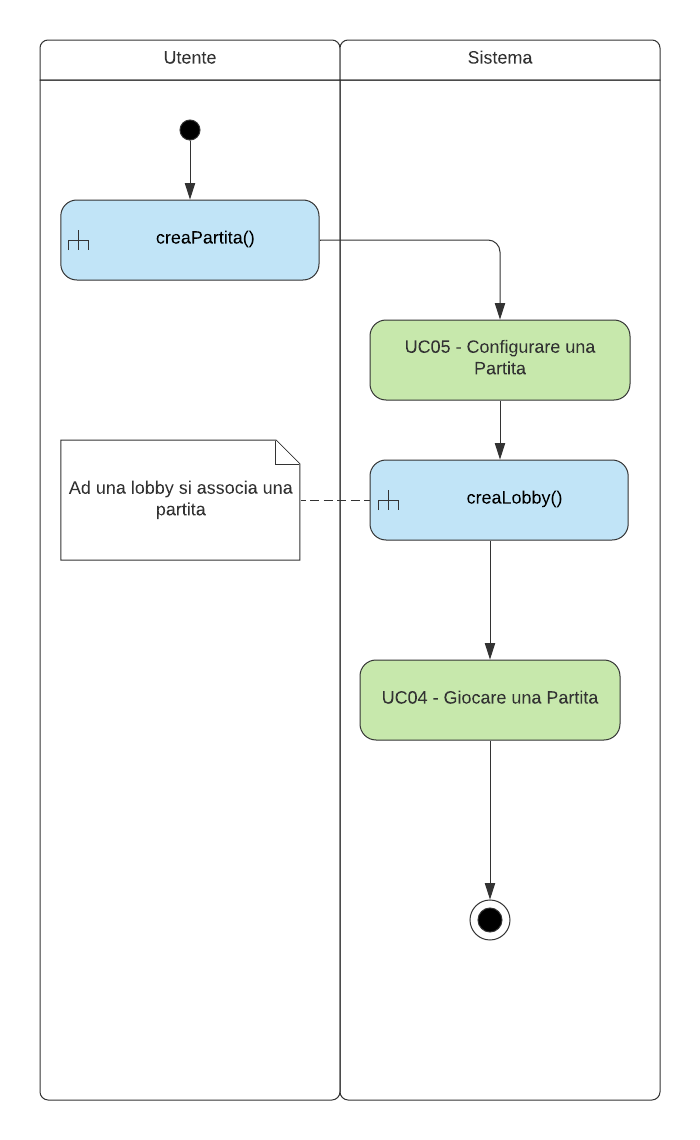
### 

### 

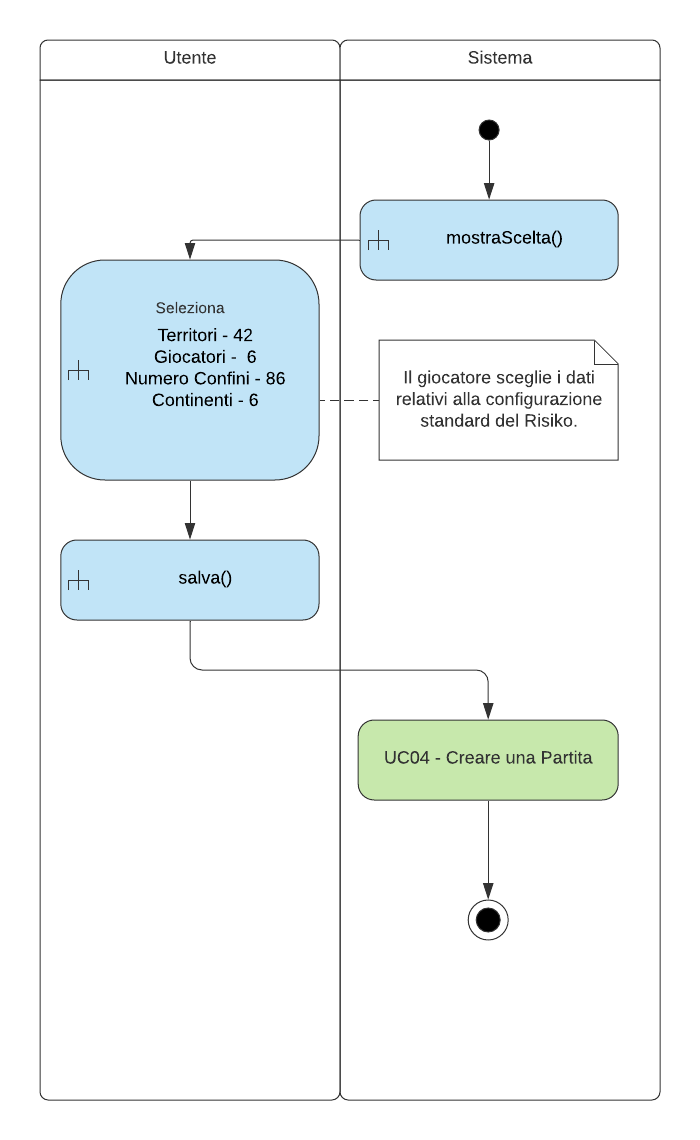
### UC08 - Diagramma di sequenza di Sistema:



## Diagrammi di attività: Creazione di una partita

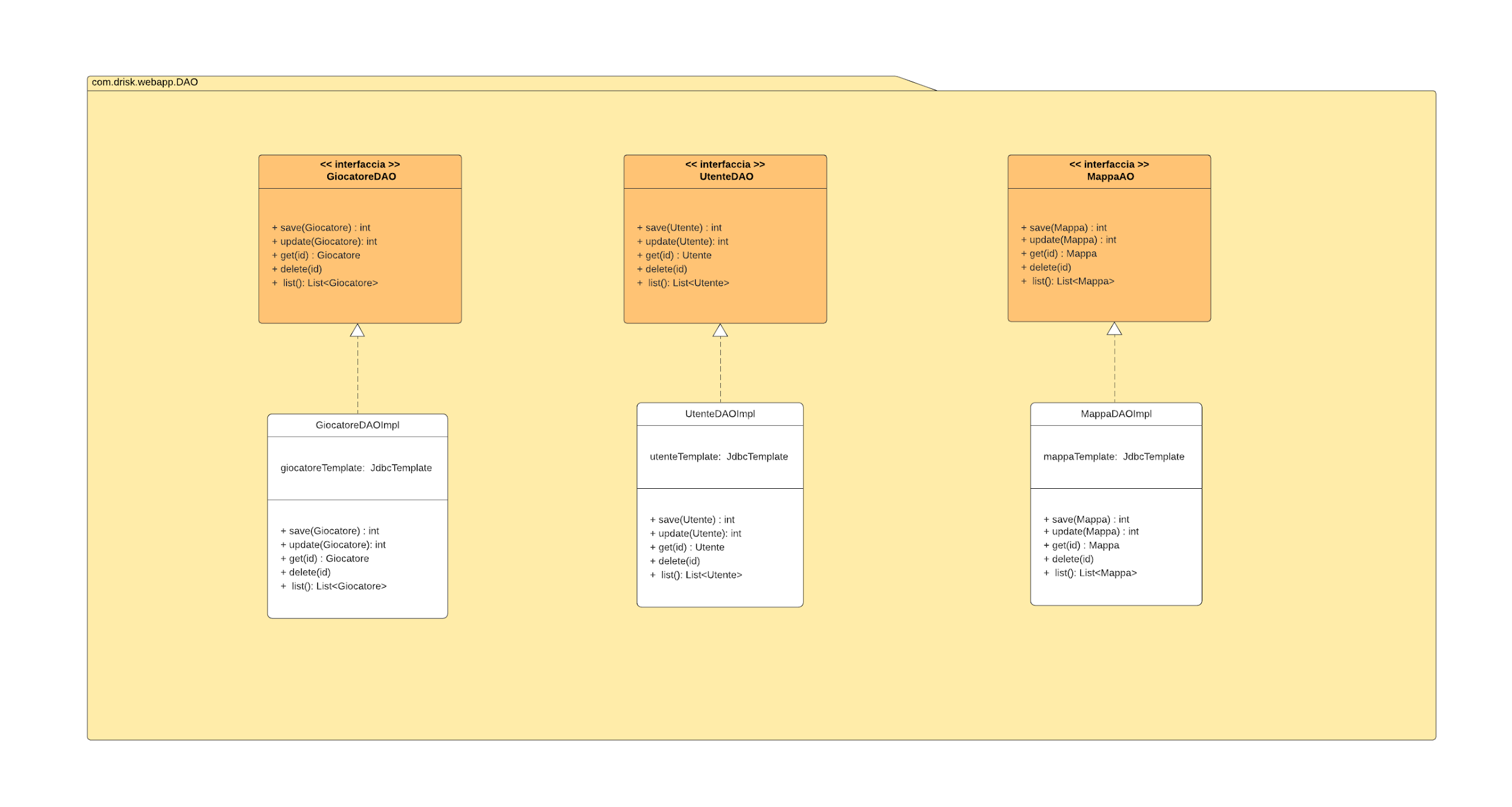


### Diagramma di attività : Configurazione di una partita standard di Risiko



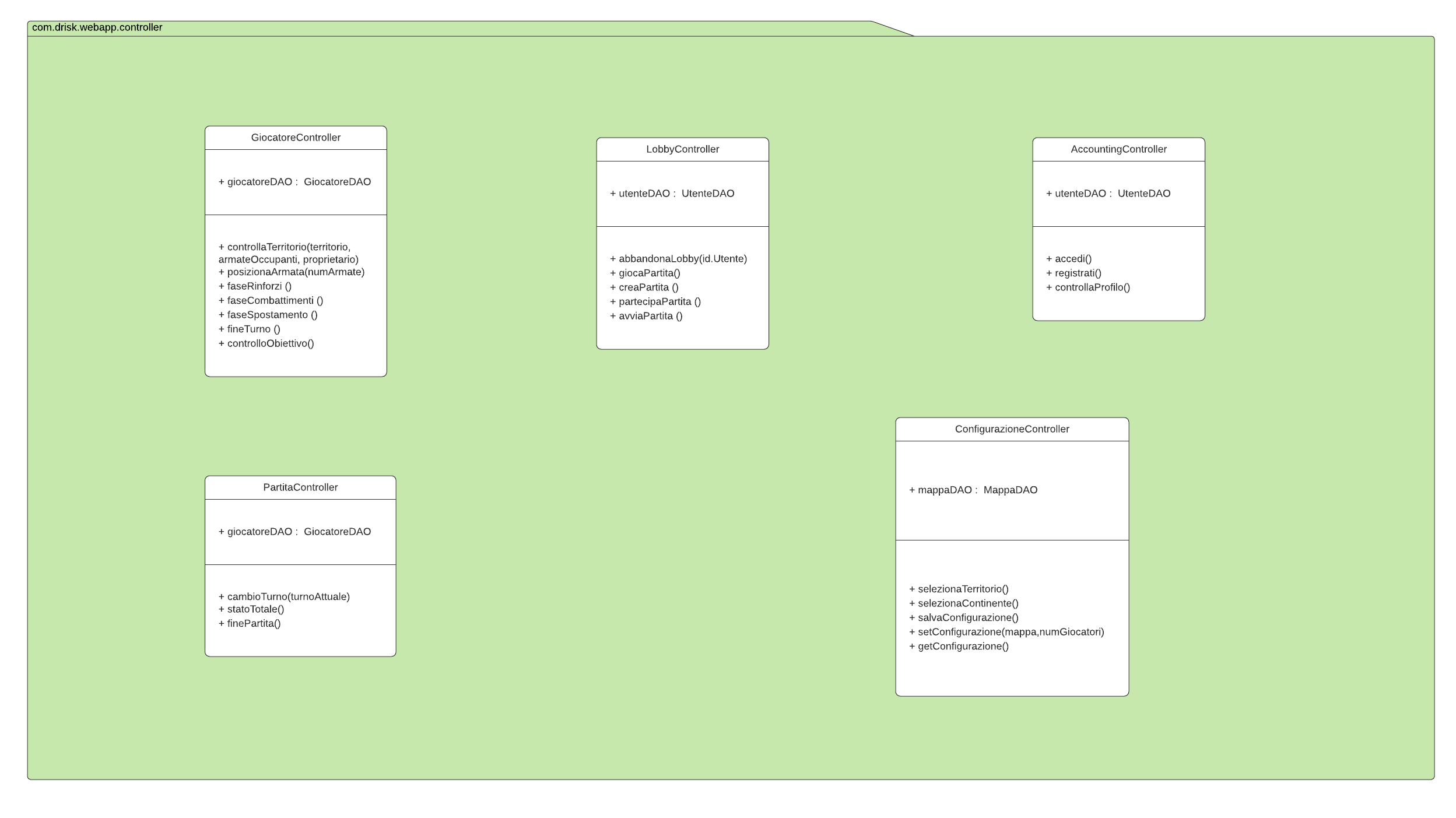
## Diagramma delle classi package Model:

## Diagramma delle classi progettazione Pattern Architetturale DAO:

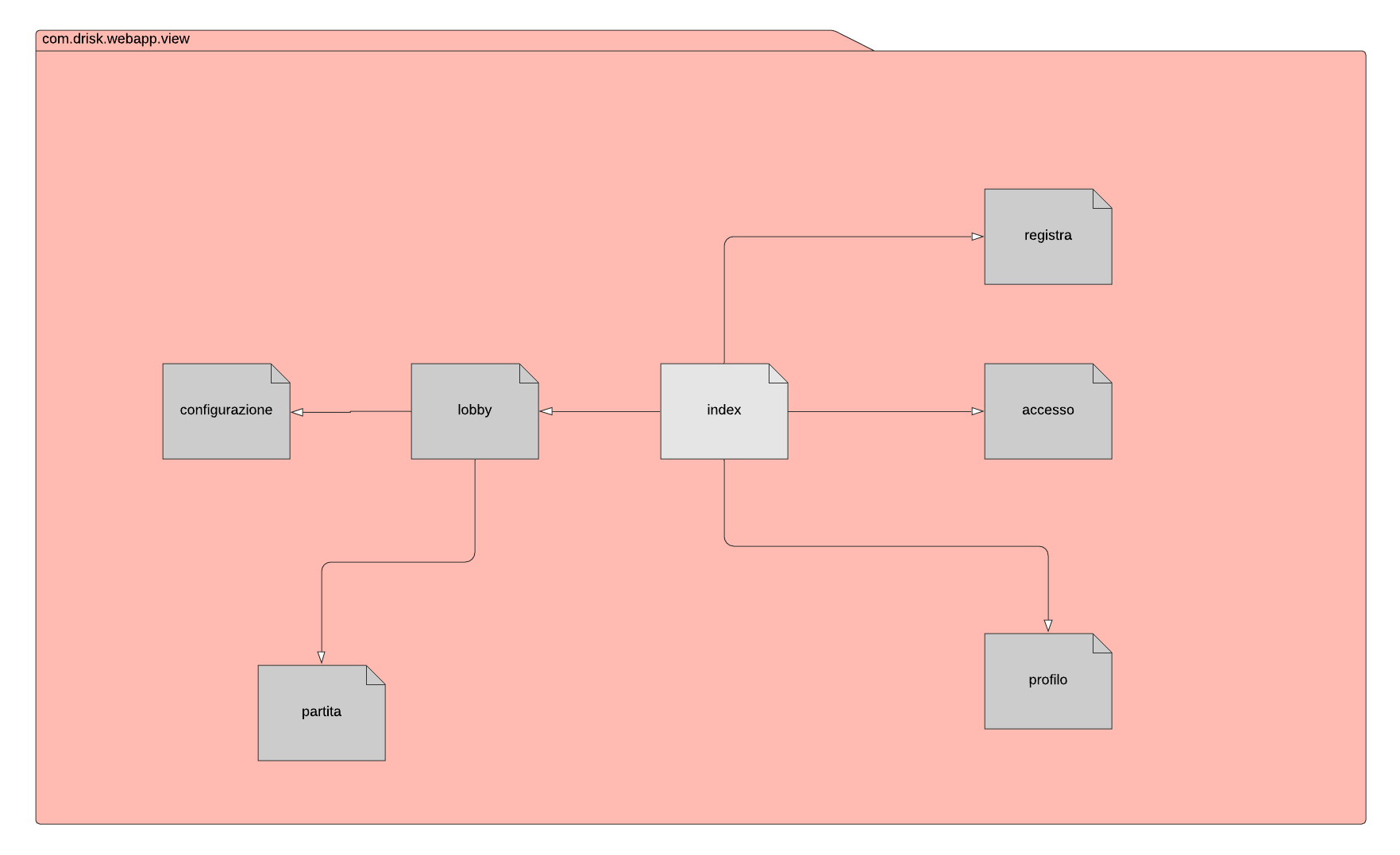


## 

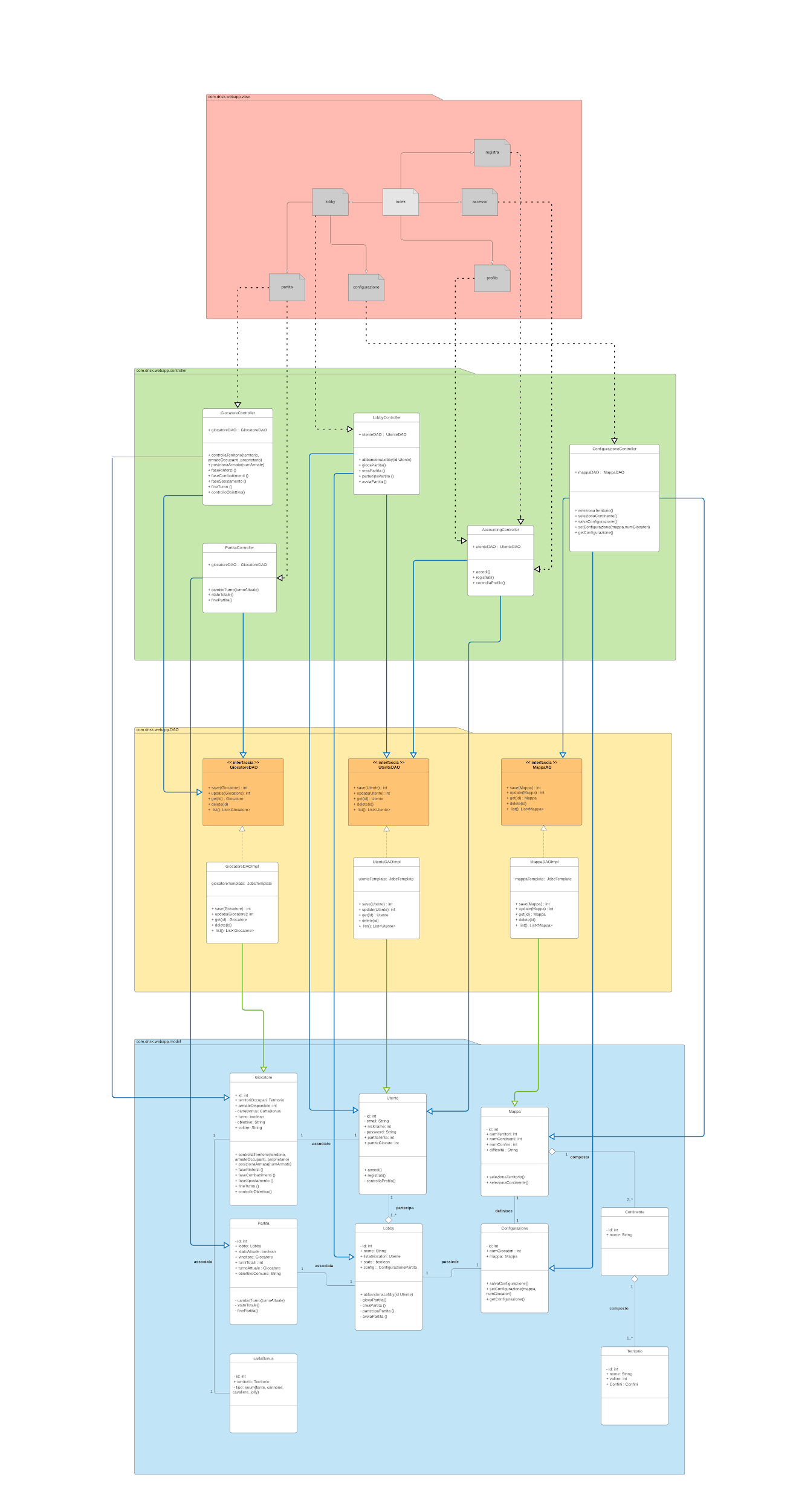
## Diagramma delle classi package Controller:



## Diagramma delle classi package Views:



## Architettura MVC dell’applicazione:



## Diagrammi di sequenza di progettazione e di stato:

## UC01 - Diagramma di sequenza di progettazione:

## 

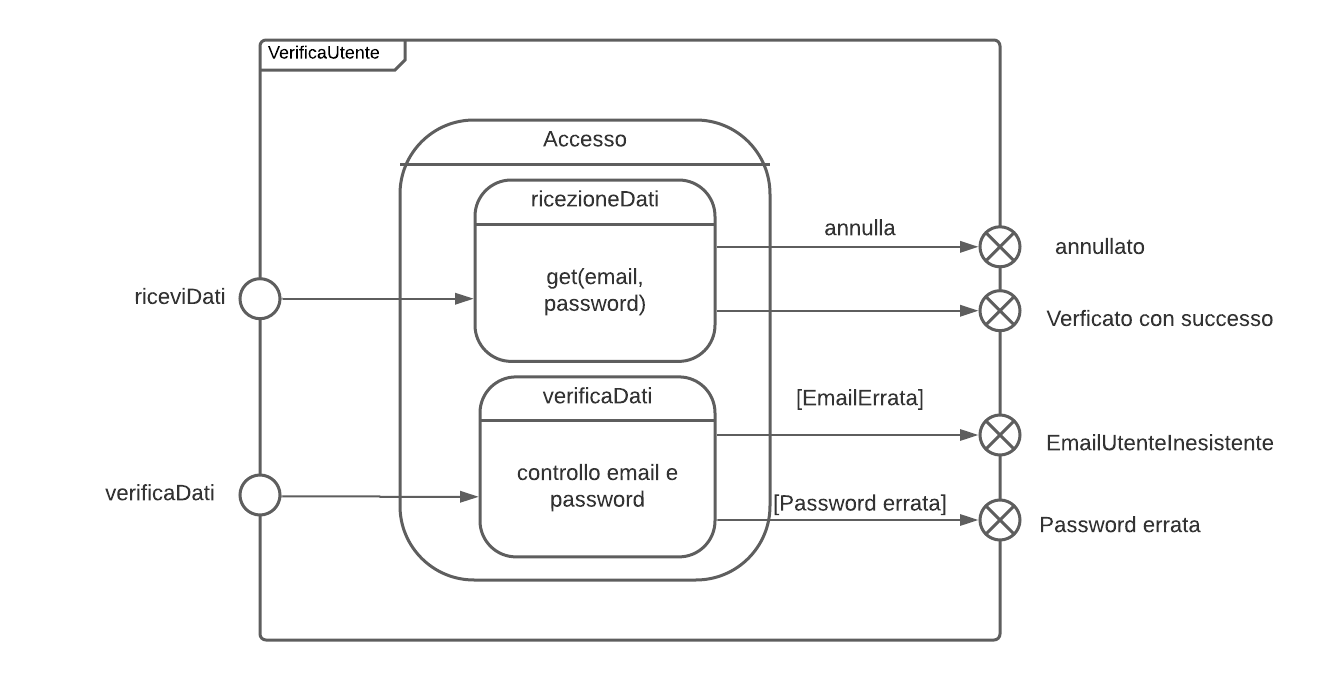
## 

## UC04 - Diagramma di sequenza di progettazione:

## 

## 

## UC02 - Diagramma di Macchina di stato:



## Contratti: **CO1 - registraProfilo**

**Operazione:** registraProfilo(Email: email, Nickname: nickname, Password; password)

**Riferimenti:** Use Cases: Registrazione al sito.

**Pre-condizioni:** Il cliente sta visitando la webapp  
**Post-condizioni:**

* la tupla relativa all’utenza è stata aggiunta all’interno del database di sistema attraverso MySQL.

## **CO2 - giocaPartita**

**Operazione:** giocaPartita()

**Riferimenti:** Use Cases: Giocare una partita.

**Pre-condizioni:** L’utente deve essere registrato al sito.

**Post-condizioni:**

* l’utente con session id: SessionUserID viene rimandato alla pagina di gestione lobby.

## Architettura di rete Client/Server dell’applicazione (esempio):

Pattern Architetturali usati:

* **Model-View-Controller:** pattern più utilizzato per lo sviluppo dell'applicazione web. Nel MVC i protagonisti sono tre: il model, la view e il controller. Il primo contiene le funzionalità e gestisce i dati, definendo gli algoritmi e le funzioni che descrivono la logica del programma. La view rappresenta la parte visiva, ed è con essa che si interfaccia l’utente. In un buon design, essa dovrebbe essere il più possibile agnostica riguardo il model. Infine, il controllersi pone tra i due, da una parte gestendo l’input dell’utente e dall’altro fornendo le informazioni necessarie al modello. Concludendo permette una totale separazione tra le varie componenti in modo possano essere modificate senza ripercussioni sulle altre componenti.

# **Data Access Object:** utilizzato per separare la responsabilità del modello e del controller sull’accesso ai dati nel database in modo da avere più persistenza.

* **Client/Server:** definisce l'architettura di rete dell’applicazione poichè nella creazione della Lobby(listener) abbiamo bisogno che possano connettersi più giocatori(client).

## Il modello client-server è costituito da un insieme di processi in esecuzione su diversi host: i processi che gestiscono una o più risorse sono detti server mentre quelli che richiedono l’accesso ad alcune di queste risorse distribuite sono detti client. Pattern GRASP usati:

* **Information Expert:**
  + applicato nella classe Utente per responsabilità:
    - conoscere i dettagli dell’utenza all’interno del Server
  + applicato nella classe Giocatore e Partita per responsabilità:
    - conoscere i dettagli relativi ai giocatori durante una partita.
    - conoscere i dettagli relativi all’andamento della partita.
  + applicato nella classe Lobby per responsabilità:
    - conoscere la lista degli utenti che vogliono iniziare una partita.
  + applicato nella classe Mappa e Configurazione per responsabilità:
    - conoscere in dettaglio le informazioni relative alla plancia di gioco in base alla configurazione scelta per una partita.
* **Creator:**
  + applicato nella classe Utente per:
    - creare un’istanza di un oggetto di classe Lobby
* **High Cohesion:**
  + applicato nella classe Utente:
    - per la gestione delle informazioni relative un utente(si noti che tale classe gestisce anche i visitatori del sito, non ancora registrati)
* **Controller:**
  + applicato nella classe PartitaController
    - gestione delle classi Partita
  + applicato nella classe GiocatoreController
    - gestione della classe Giocatore
  + applicato nella classe LobbyController:
    - gestione della classe Lobby e Utente
  + applicato nella classe ConfigurazioneController:
    - gestione della classe Configurazione

## Aggiornamento consegna progetto:

Viste le tempistiche di lavoro, abbiamo deciso di implementare la struttura iniziale della webapp costruendo quindi la parte fondamentale del sito.

Abbiamo utilizzato come linguaggi e supporti di programmazione:  
Java, Javascript, HTML5, CSS, SQL, SVG, JSON.

Con la consegna del progetto abbiamo rilasciato due demo giocabili (ancora in sviluppo), una con un continente e due giocatori e un’altra con 6 continenti e numero di giocatori variabile.   
L’obiettivo prossimo è quello di continuare lo sviluppo del progetto:

* aggiustando gli scripts del gioco.
* aggiungendo l’implementazione delle lobby .
* aggiungendo il multiplayer negli script.

Le demo attualmente disponibili si trovano alla repository github:

<https://github.com/UnimibSoftEngCourse2021/progetto-risiko-2-gruppo-risiko-2>

Tali demo includono:

- gli scripts relativi al gioco effettivo.  
 - una prima struttura della webapp con pagine di index, creazione utenze e partite.  
 Per utilizzare le demo seguire le istruzioni indicate sul readme di github.