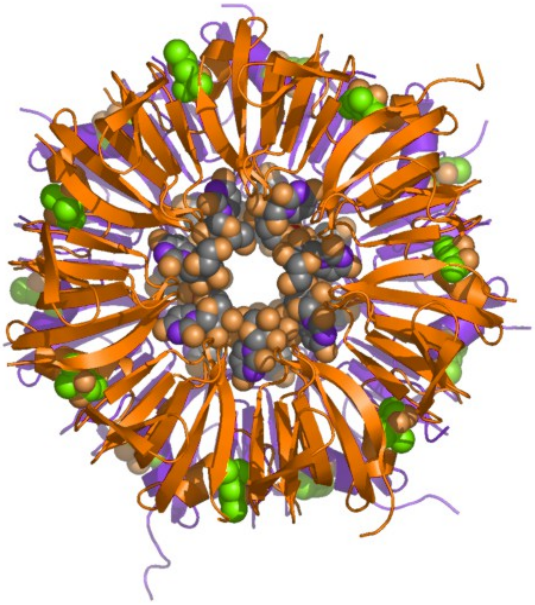


Matteo Cartuccia
Flavio Macciocchi

|FIXIT|

Documento di progettazione

ID: 5-FIX-PRO-v05-r01

Data ultima modifica: 20/01/2013
Data ultima revisione: 15/01/2013

Progetto di Ingegneria del software A.A 2012/2013

1. Introduzione

Nel presente documento andiamo a specificare le classi di *design* che rappresentano la concretizzazione delle classi di analisi esposte precedentemente (doc **FIX-ANL-v00-r00**).

Il nostro lavoro consiste ora nel congelare il modello di analisi (fase di elaborazione), effettuare una copia del modello di analisi e trasformarla, raffinandola, nel modello di progettazione. Le conseguenze di questo approccio sono quelle di avere fatto coesistere entrambi i modelli, ma di lasciarli in uno stato non sincronizzato: in questo modo non avremo la perdita di un modello a favore dell'altro.

Per prima cosa cominceremo a lavorare sulle componenti fisiche del sistema, elencando quelle più rilevanti e descrivendo il loro funzionamento.

Saranno inoltre aggiunte e migliorate delle classi dedicate alle interfacce dei dispositivi come le GUI dell'utente giocatore o quelle dedicate alla gestione e alla creazione delle singole proteine da parte degli utenti ricercatori e al relativo salvataggio nella base di dati.

Nella parte finale della presente documentazione riporteremo i diagrammi di sequenza (che realizzano i casi d'uso) con l'aggiunta delle interazioni che le singole pari costitutive compiono rispetto ai dispositivi esterni[1] del sistema.

1 *Dispositivi esterni:* sono considerati come classi dell'interfaccia utente, classi dell'interfaccia di sistema, classi dell'interfaccia di dispositivi.

2. Progetto Architetturale

2.1 Comunicazione fisica

Riguardo alla comunicazione fisica tra dispositivi, possiamo dire che ci si avvarrà di una rete di tipo LAN[2] gestita direttamente dai biochimici e delle reti domestiche già diffuse in gran parte dei paesi industrializzati. Tutto si svolgerà grazie all'utilizzo del web, usato come piattaforma virtuale alla quale ogni utente giocatore potrà collegarsi da remoto usando semplici *personal computer* dotati di sistema operativo basato su UNIX[3] (GNU/Linux, FreeBSD, Mac OS X) o Microsoft che supporti versioni di Oracle Java[4] 6.0 o superiori.

Il sistema centralizzato sarà composto da:

- un *firewall* che filtrerà le richieste al sistema su determinate porte predisposte all'ascolto di messaggi TCP[5]
- un *server web* per la gestione di un *database*
- un *database* per l'archiviazione dei dati ricevuti dai *client*
- postazioni PC (o terminali) per avviare il *client* negli uffici dei biochimici o in un laboratorio condiviso da tutti
- un *router* aziendale che colleghi il sistema centralizzato alla rete internet
- uno *switch* per collegare gli apparecchi dotati di IEEE 802.3 (*ethernet*)

2 LAN: http://en.wikipedia.org/wiki/Local_area_network
3 UNIX: rimandiamo al wiki http://en.wikipedia.org/wiki/Unix
4 Java: rimandiamo al sito ufficiale http://www.java.com
5 TCP: protocollo di comunicazione, dettagli all'indirizzo http://en.wikipedia.org/wiki/Transmission_Control_Protocol

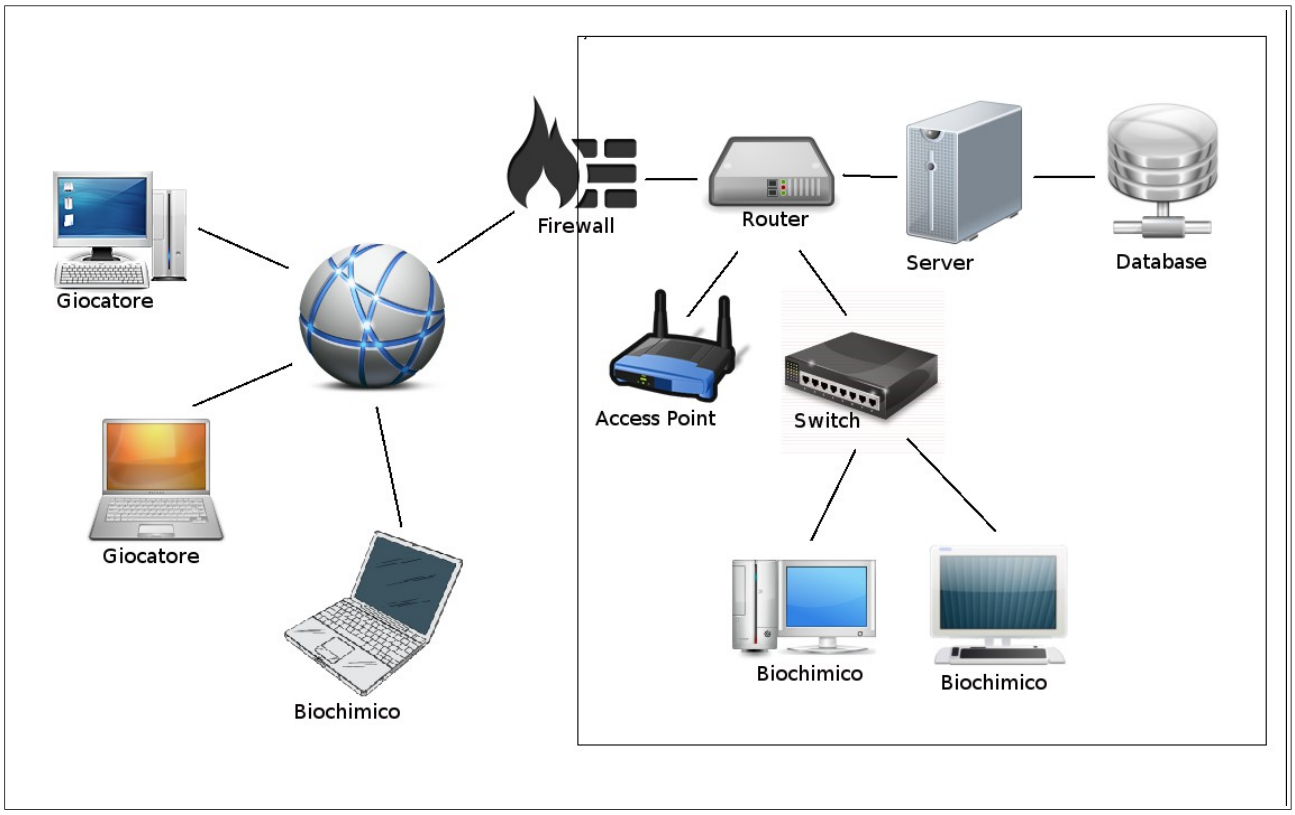


Figura 1.0) Sistema hardware usato da Fixit.

La tipologia di utente biochimico può sfruttare il collegamento web come un normale utente giocatore, collegandosi da remoto per organizzare e sviluppare nuove strutture tridimensionali senza trovarsi personalmente sul luogo di lavoro. Il sistema fisico dunque risiederà in un laboratorio gestito da un gruppo di biochimici, ma sarà raggiungibile da parte di ricercatori sparsi in tutto il mondo. Sarà però compito del gruppo primario spedire dei *codici di accesso*[6] ai biochimici interessati al progetto previa esplicita richiesta mail. Il codice permetterà al biochimico in questione di effettuare il login come tale.

6 Codici di accesso: codici forniti dal team ai biochimici, risiedono in un file.txt fornito agli utenti durante la consegna finale del lavoro.

2.2 Comunicazione software

Il sistema *client* lato giocatore è predisposto per supportare un tipo di *chat* che si avvale del *server* principale per mettersi in comunicazione con gli altri operatori. Il protocollo usato per la comunicazione usa lo standard XMPP, precedentemente noto come Jabber e riconosciuto secondo la XMPP *Standards Foundation*.

I dati di ogni proteina ripiegata correttamente saranno spediti verso il *server web* in ascolto e incamerati nel *database* secondo le direttive del DBMS MySql, che organizzerà i dati inserendoli in tabelle con campi prestabiliti, così da poter facilitare il lavoro di lettura degli ingegneri informatici.

Gli attori ingegneri esporteranno dal sistema un file in formato XML che segue lo standard ISO, riportiamo ulteriori specifiche in merito nel capitolo sottostante.

La parte inerente allo sviluppo tridimensionale è rimandata a una speciale libreria OPENGL del linguaggio di programmazione *Processing*[7], che realizza in modo dettagliato i modelli 3D per gli utenti registrati, quindi per la progettazione e l'interazione con gli stessi.

7 Processing: http://processing.org/

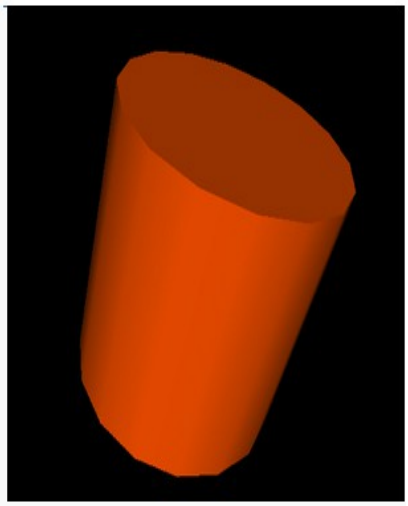
3. Linguaggio per la descrizione di ambienti 3D

Per quanto concerne i dati da esportare, questi verranno implementati grazie a un particolare linguaggio che si occupa della descrizione di ambienti virtuali interattivi, chiamato **X3D**[8]. Sviluppato dal *Web 3D Consortium*, esso si basa principalmente su XML ed è un formato non proprietario standardizzato dall'ISO (*International Organization for Standardization*) nel 2004.

Grazie all'X3D l'utente può facilmente sviluppare strumenti per il contenuto come un esportatore (come nel nostro caso) ed *editor* che facilitano la creazione di contenuti e le attività di ottimizzazione.

Di seguito riportiamo un esempio di codice che rappresenta una forma geometrica di banale entità:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE X3D PUBLIC "http://www.web3d.org/specifications/x3d-3.0.dtd"
                     "file:///www.web3d.org/TaskGroups/x3d/translation/x3d-3.0.dtd">
<X3D profile="Immersive"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema-instance"
  xsd:noNamespaceSchemaLocation="http://www.web3d.org/specifications/x3d-3.0.xsd">
  <head>
  </head>
  <Scene>
    <Transform>
      <Shape>
        <Cylinder/>
        <Appearance>
          <Material diffuseColor="1.0 0.4 0.0"/>
        </Appearance>
      </Shape>
    </Transform>
  </Scene>
</X3D>
```

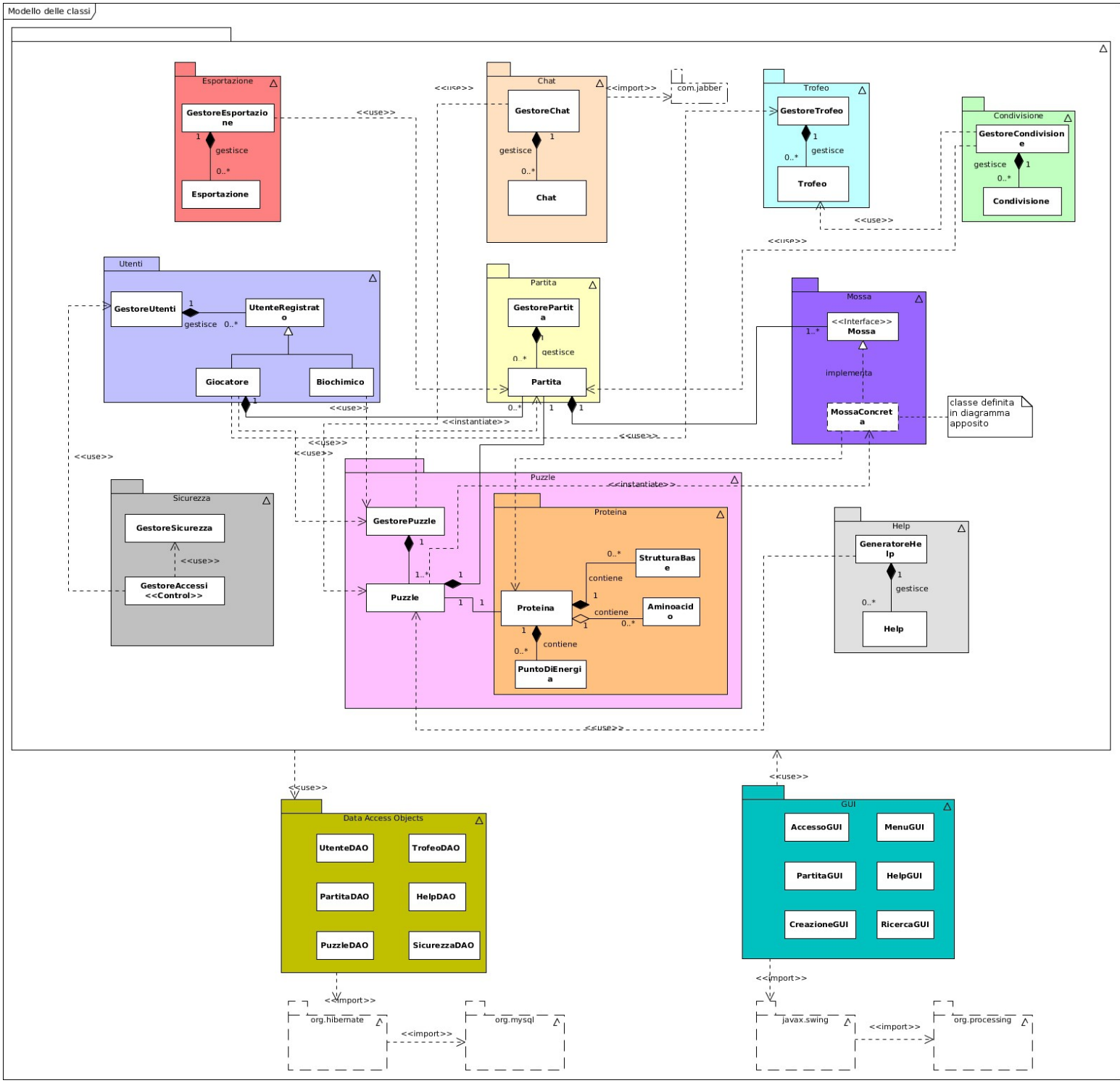


8 X3D: http://en.wikipedia.org/wiki/X3D

Il file X3D è in formato XML e al suo interno esistono dei campi annidati che descrivono l'ambiente tridimensionale attraverso quello che viene definito grafo della scena. Il grafo consiste in un albero i cui nodi interni rappresentano le informazioni inerenti alle trasformazioni dell'immagine all'interno dello spazio. Dopo troviamo le foglie dell'albero che raffigurano le entità, ovvero l'immagine principale che descrive la scena.

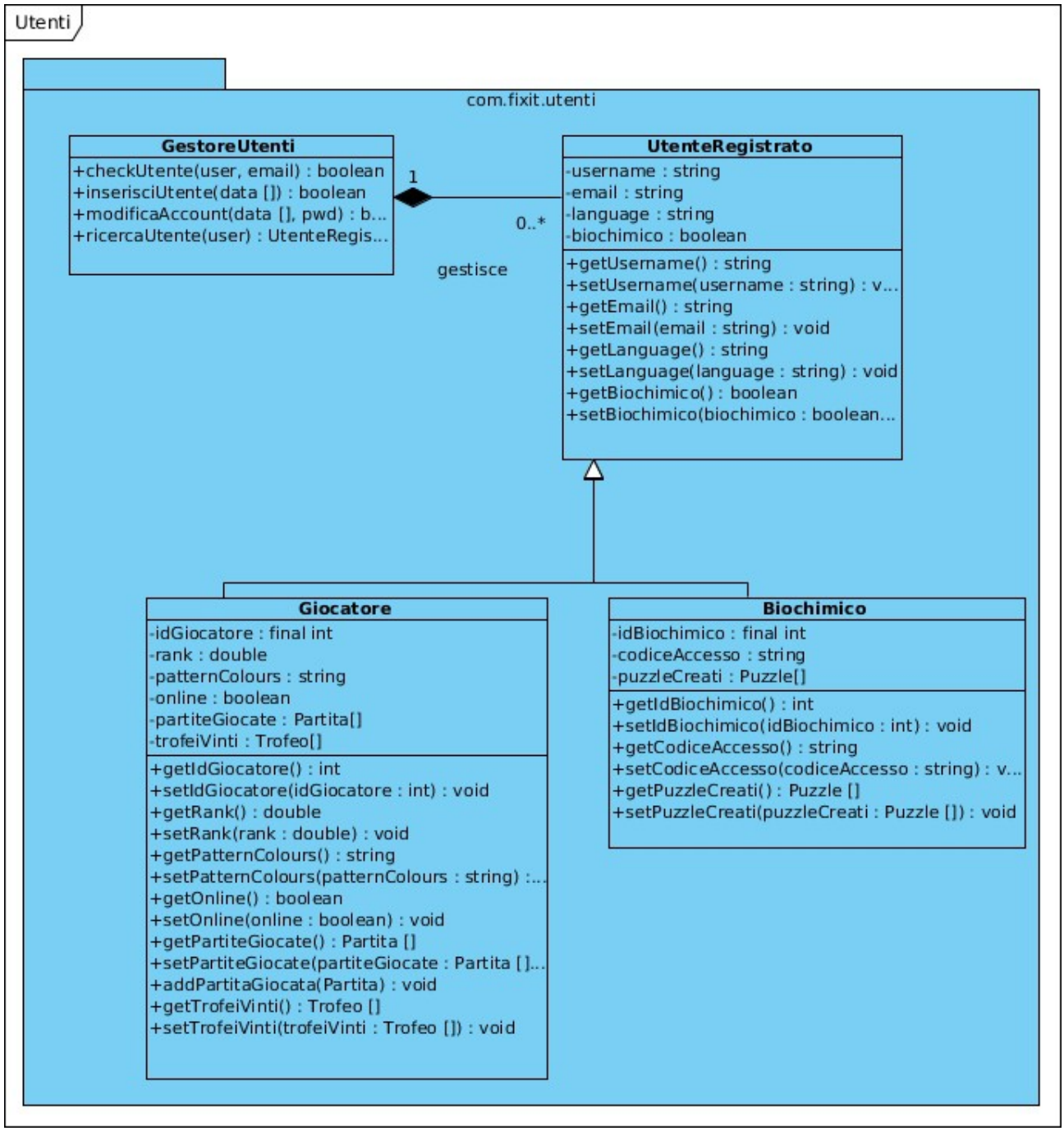
Mentre il *player* interagirà con la proteina, i campi nodi verranno alterati, salvando le informazioni riguardanti i vari spostamenti nello spazio tridimensionale.

4. Modello *package* vista progettuale

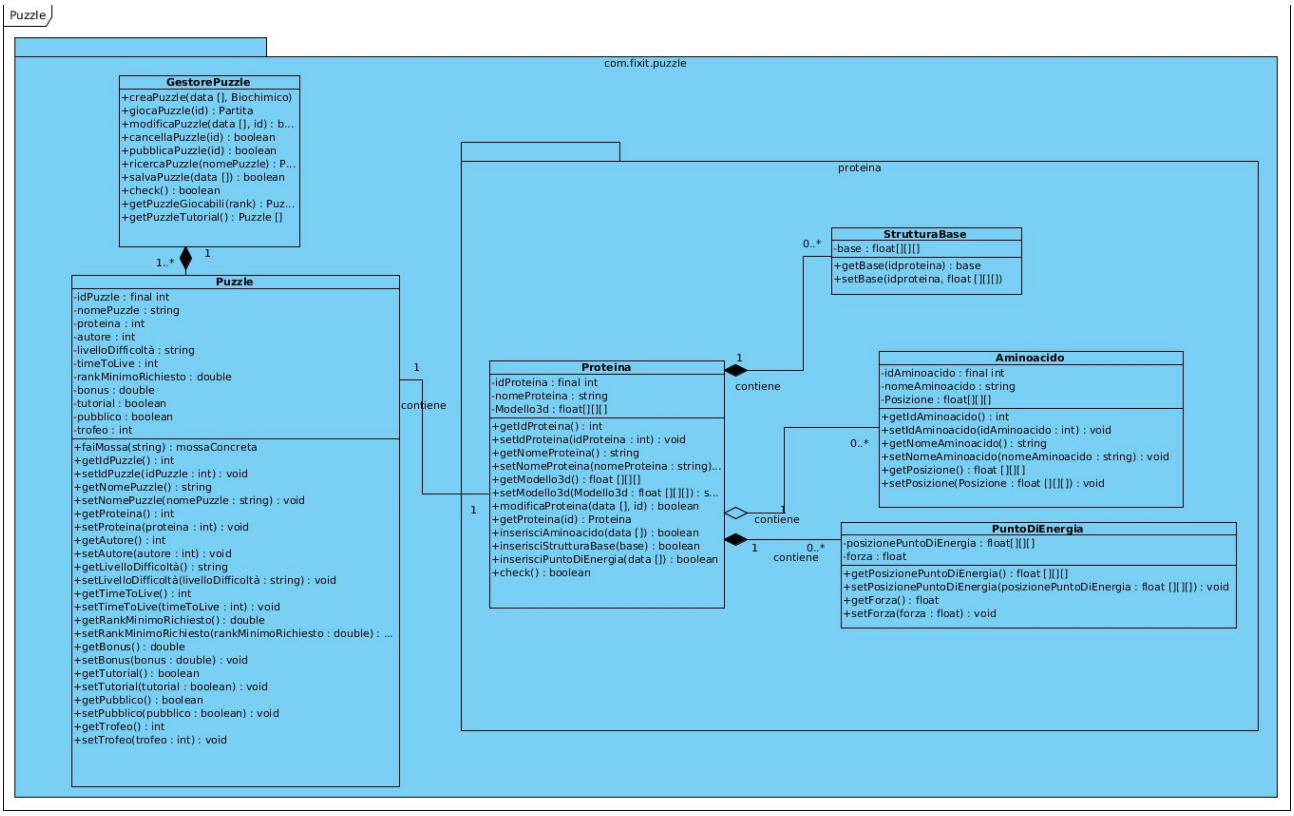


5. Diagrammi delle classi

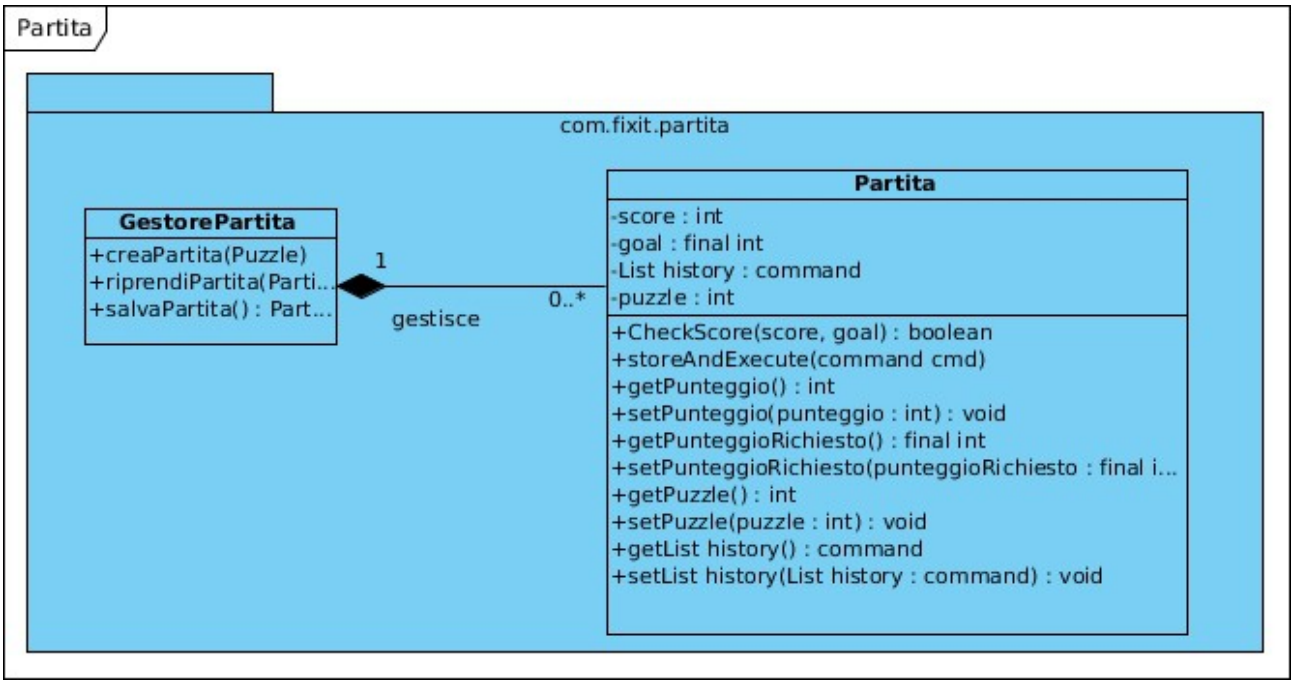
5.1 com.fixit.utenti



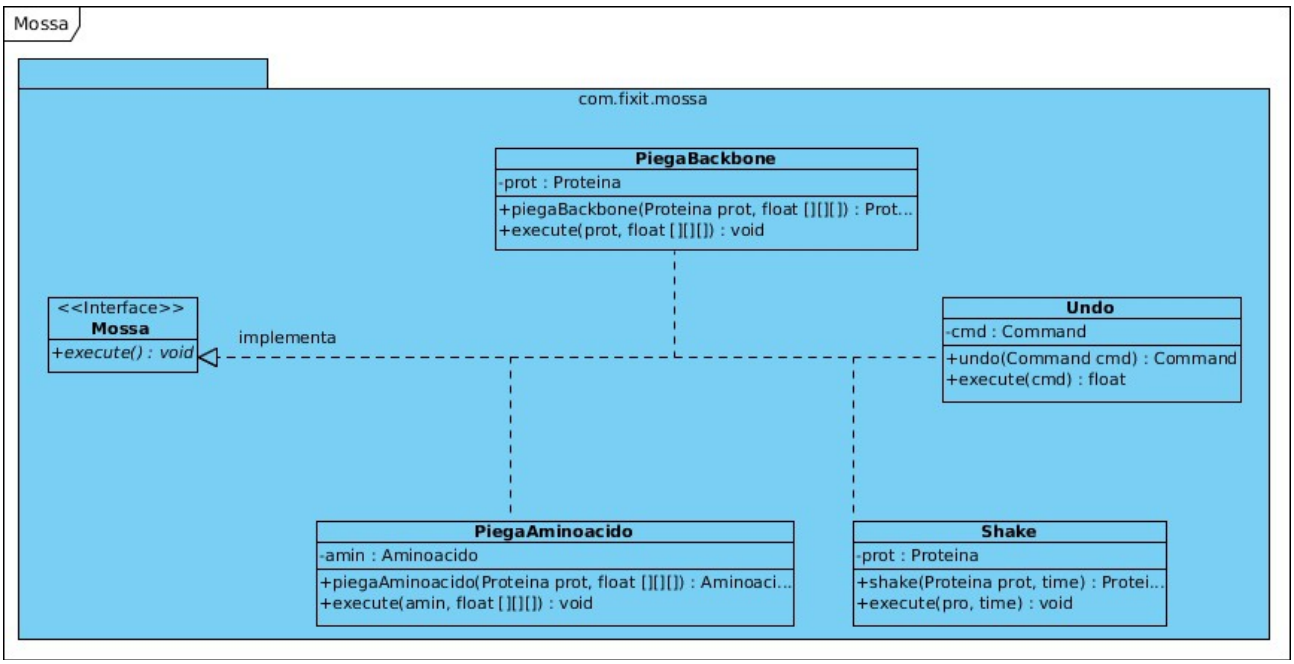
5.2 com.fixit.puzzle



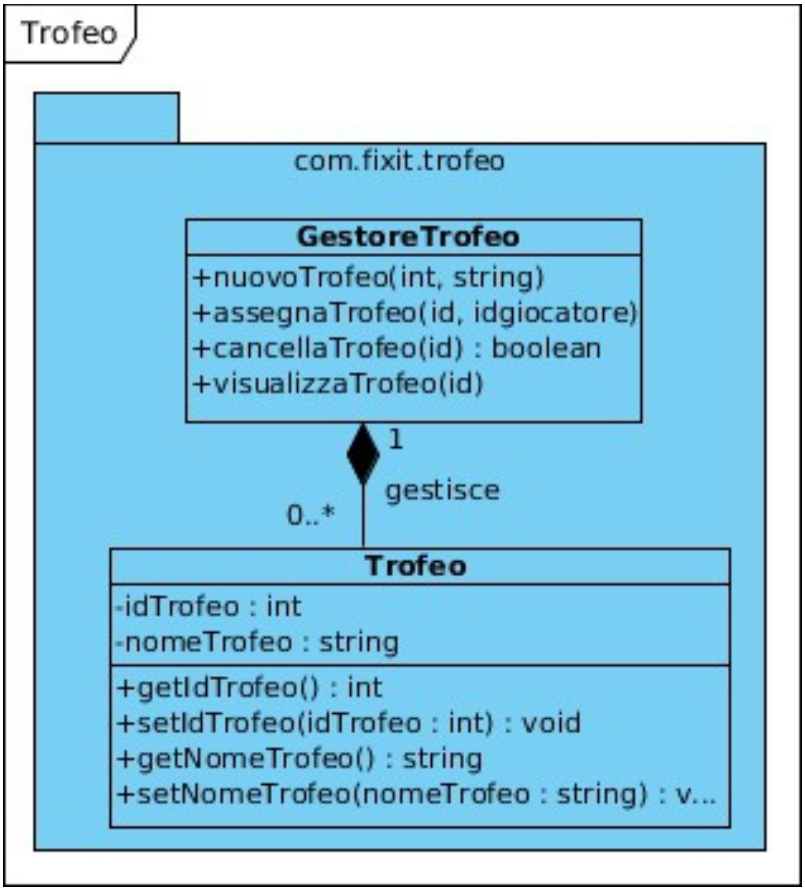
5.3 com.fixit.partita



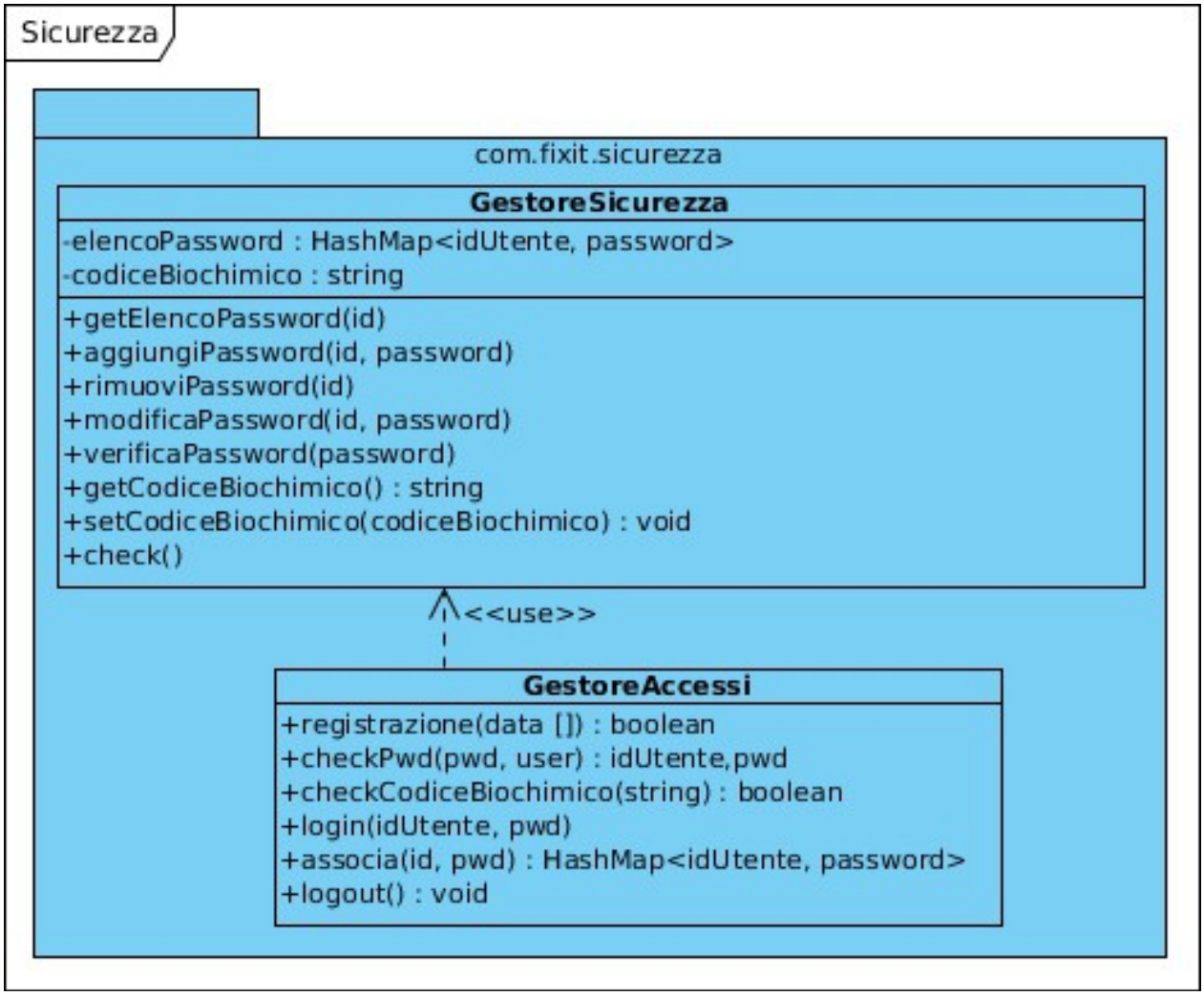
5.4 com.fixit.mossa



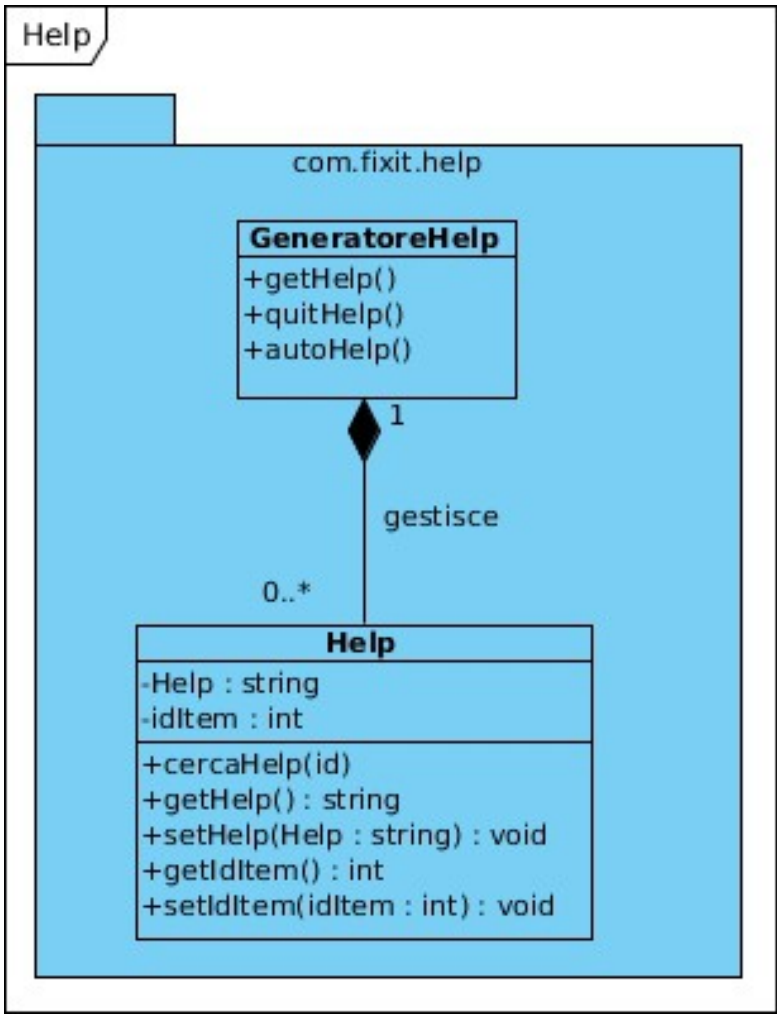
5.5 com.fixit.trofeo



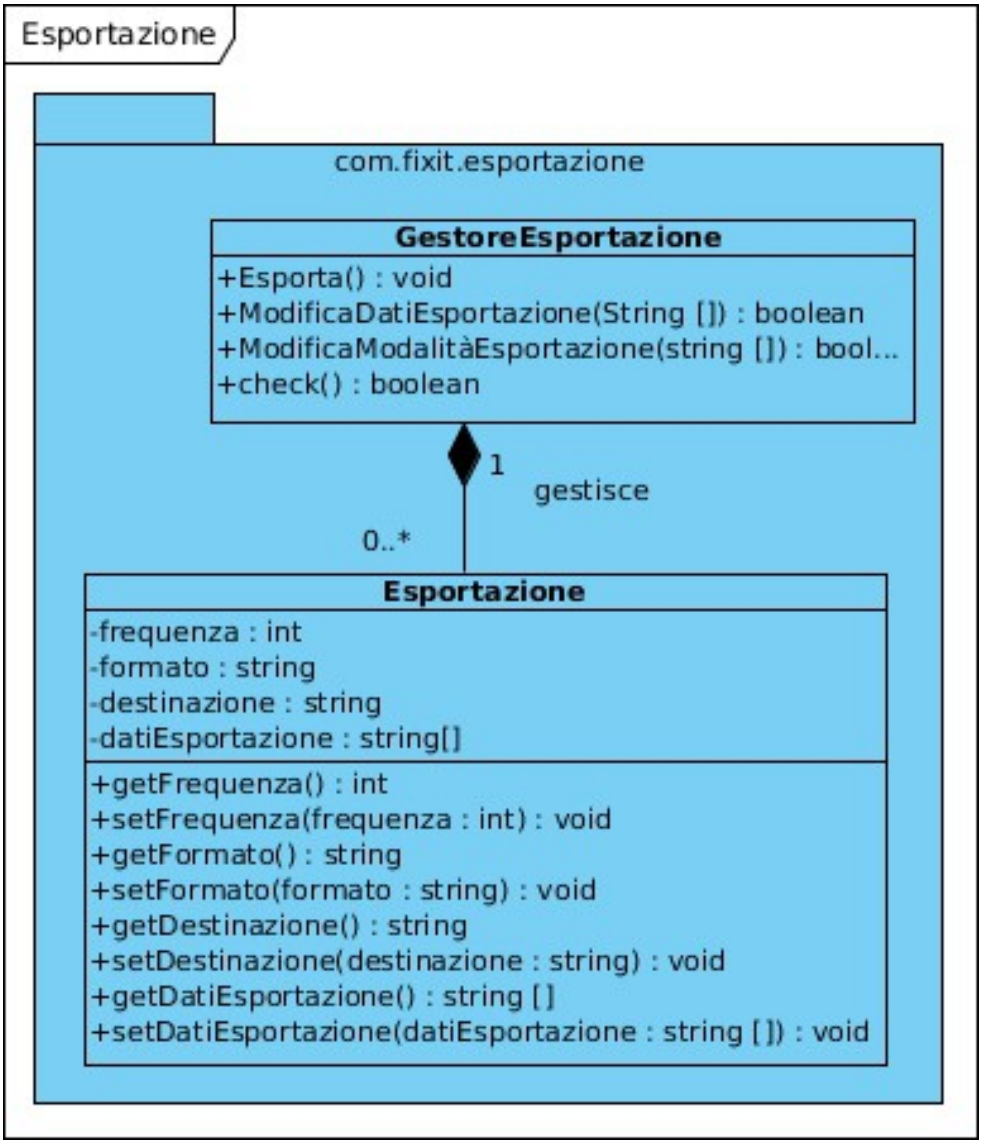
5.6 com.fixit.sicurezza



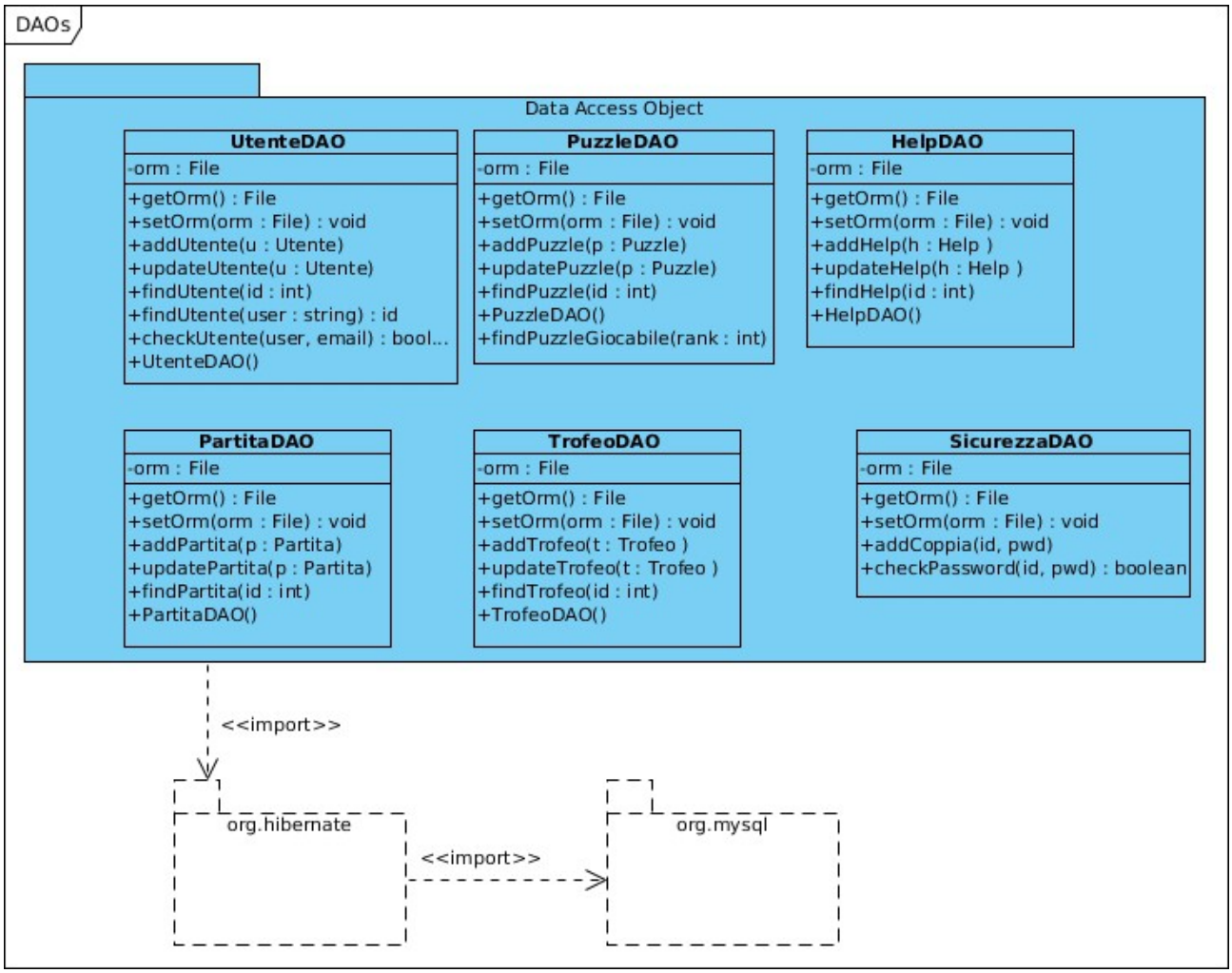
5.7 com.fixit.help



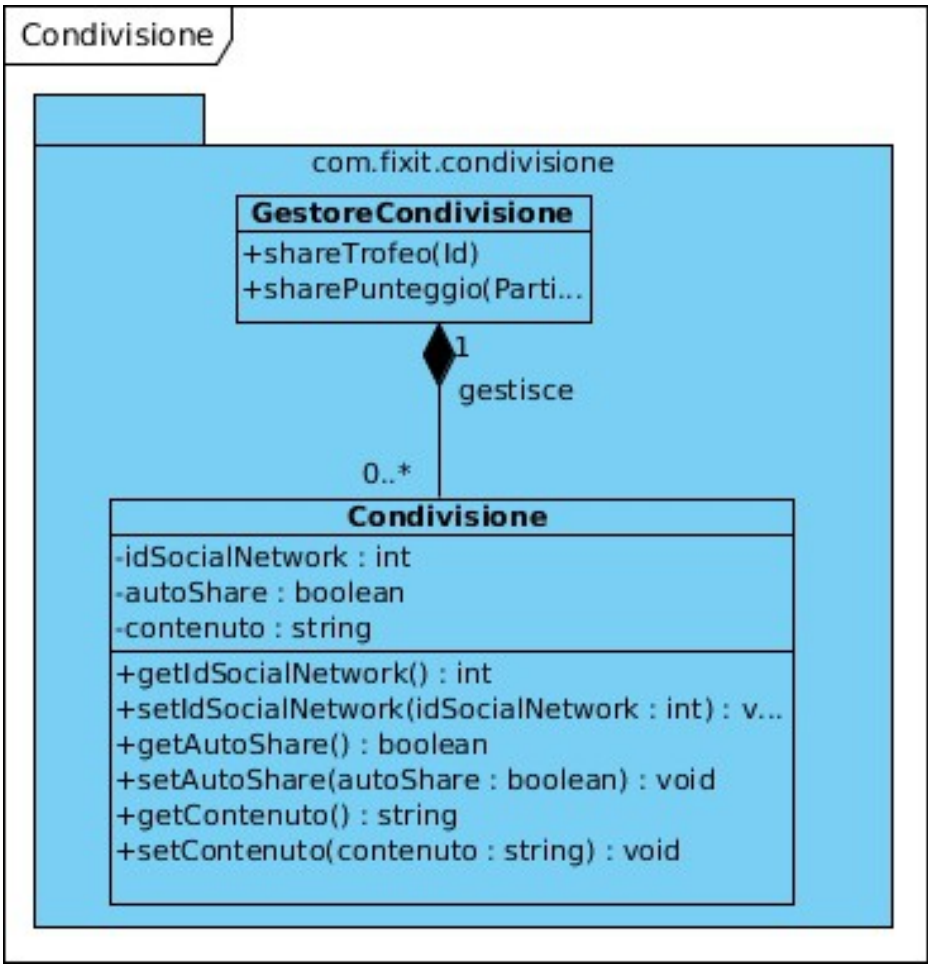
5.8 com.fixit.esportazione



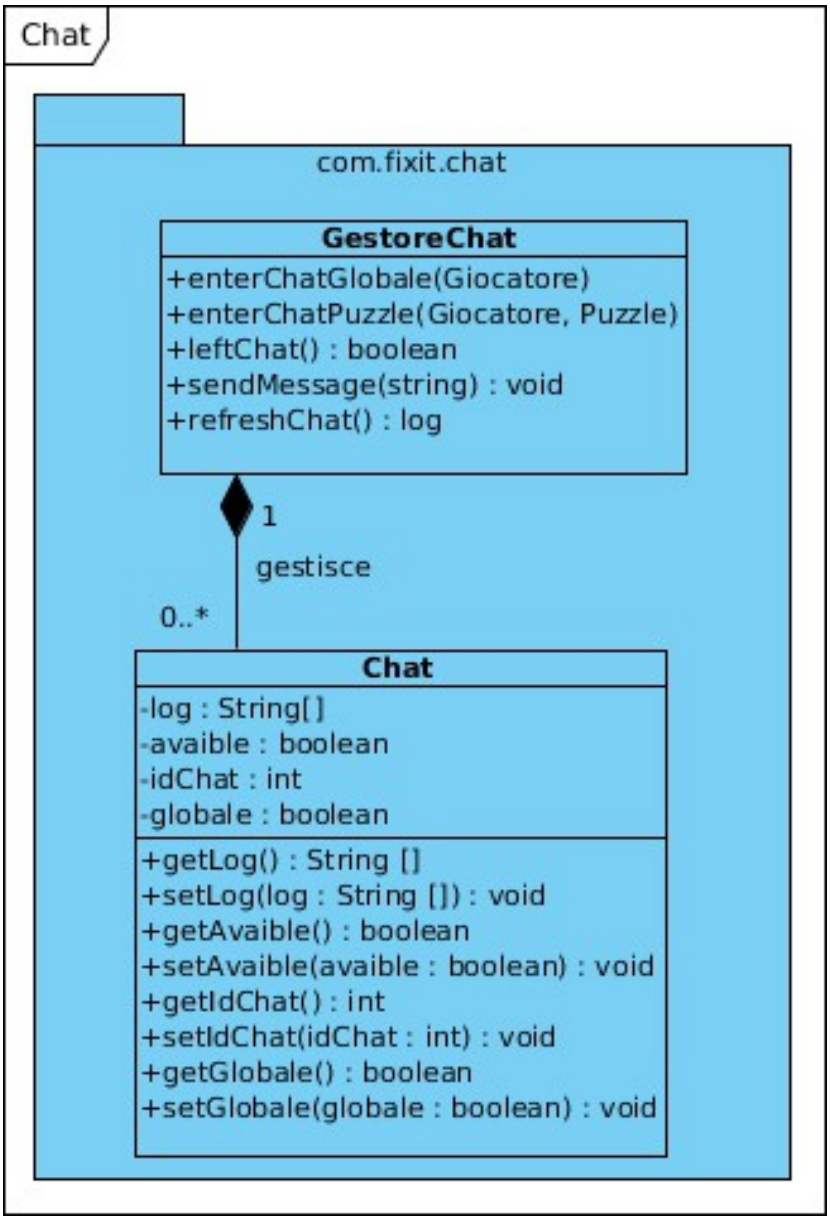
5.9 Data Access Object



5.10 com.fixit.condivisione



5.11 com.fixit.chat



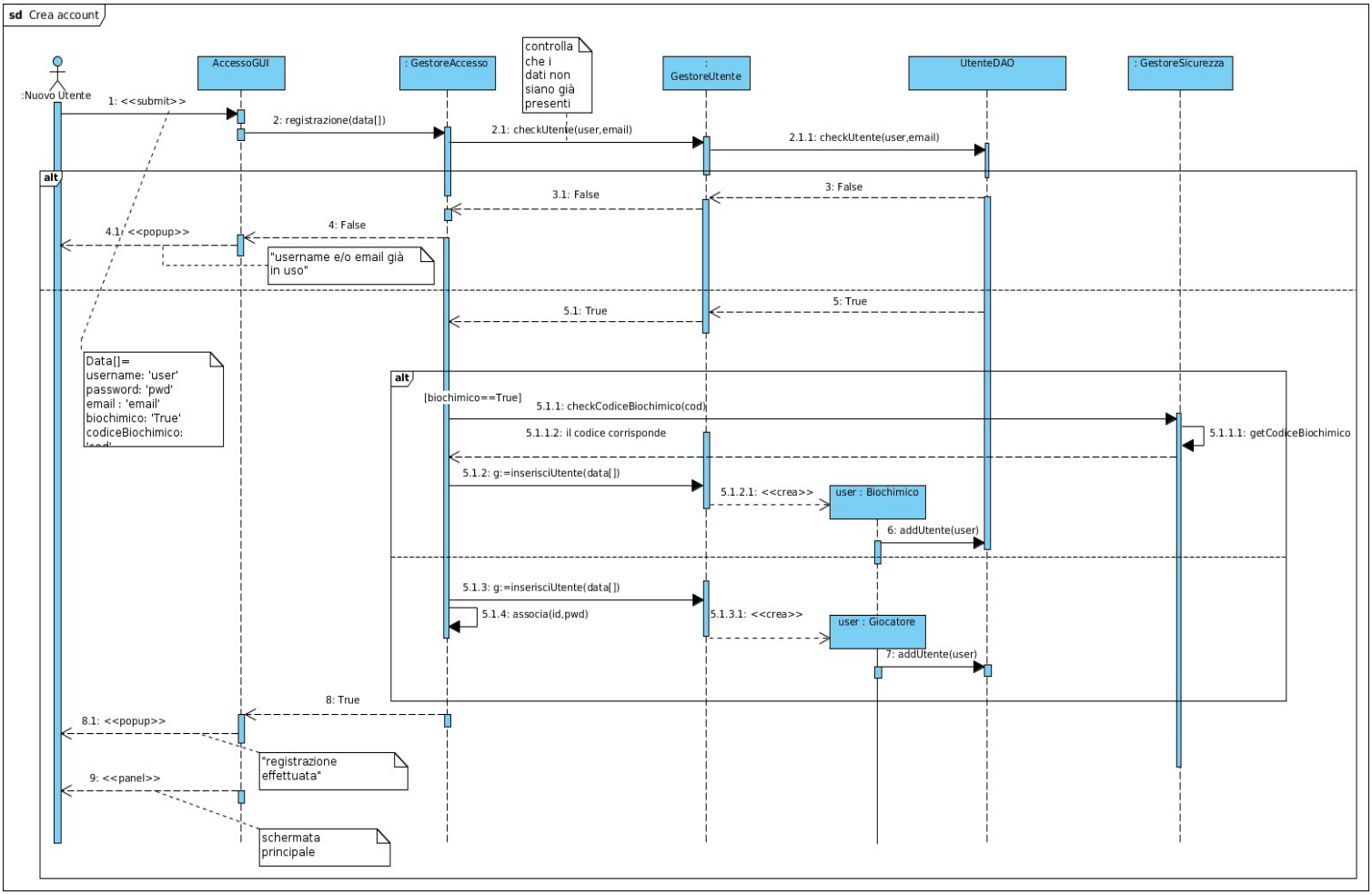
6. Diagrammi di sequenza relativi alla progettazione

I seguenti diagrammi di sequenza sono stati ripresi dall'analisi e rimodellati secondo la nuova visione dei *package* riportata sopra. Essi mostrano lo scambio di messaggi tra i singoli moduli e le componenti, le quali restituiscono sempre un *feedback* a un *input* dell'utente qualora questo voglia effettuare un'operazione.

Come già citato precedentemente i diagrammi in questione realizzano i casi d'uso. Ci riserviamo di sviluppare comunque solo alcuni di essi presentando solo quelli che svolgono funzioni indispensabili per il funzionamento del sistema di gioco.

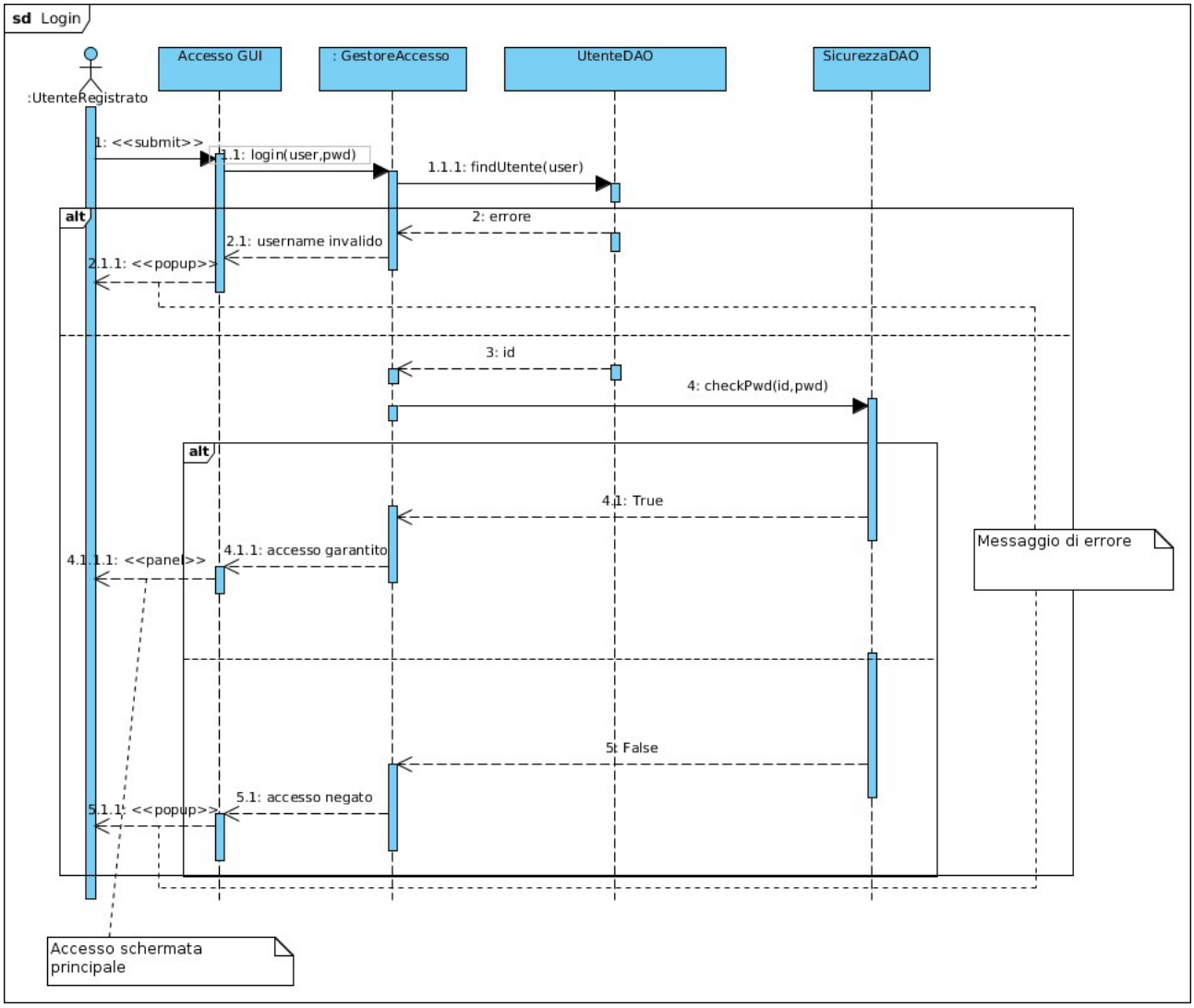
6.1 Crea Account

Il seguente diagramma descrive come avviene la funzione di registrazione utente. Per prima cosa si inseriscono i dati relativi alla registrazione tramite la GUI e il sistema controlla che i relativi dati non siano stati già immessi precedentemente. La funzione *check* processa quindi i dati restituendo eventualmente un errore, altrimenti il controllo passa al *gestoreUtente* che creerà un istanza di utente giocatore o biochimico inserendola nel database.



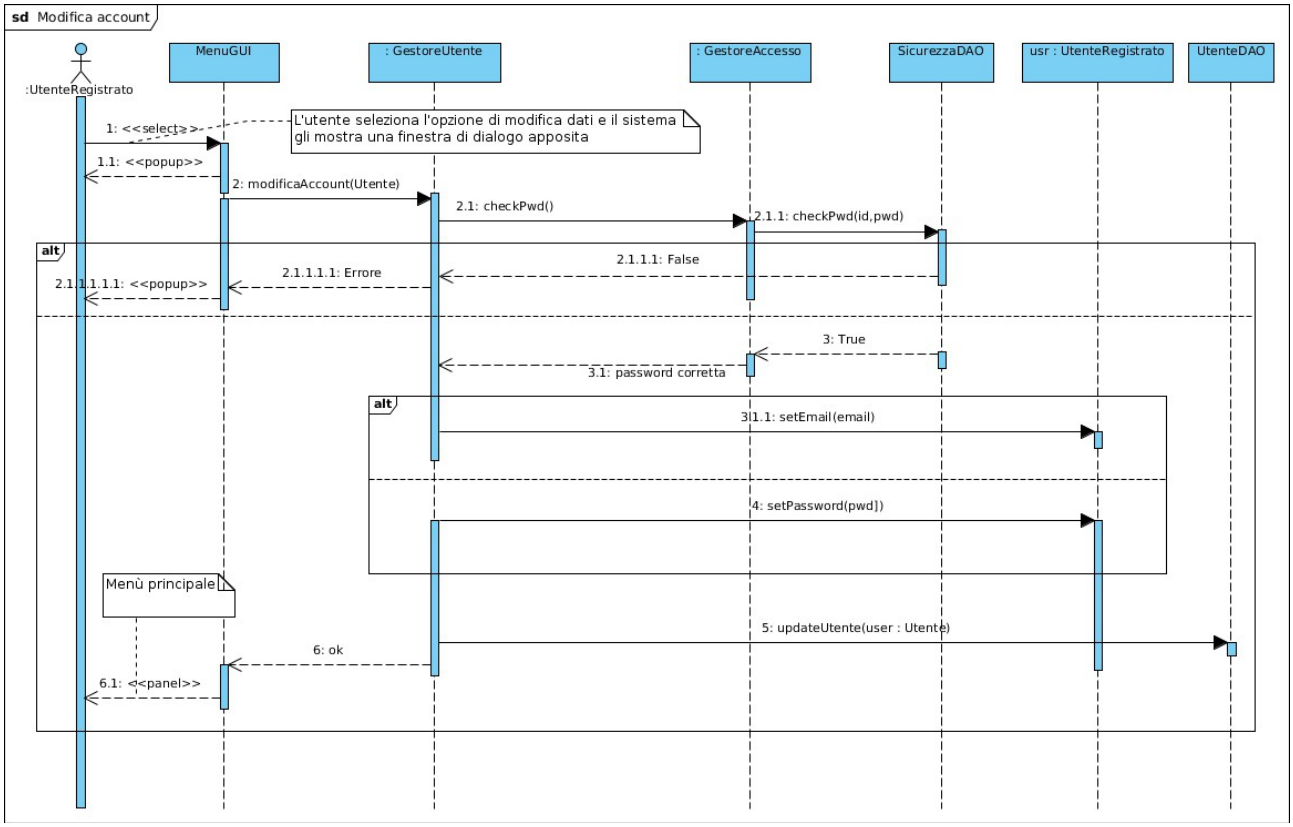
6.2 Login

L'utente inserisce i dati di autenticazione e il sistema controlla che il suddetto sia già presente in memoria, in caso affermativo il sistema verifica che la password sia corretta. In caso venisse a mancare una delle due precedenti istanze, il software ritorna a video un messaggio di errore.



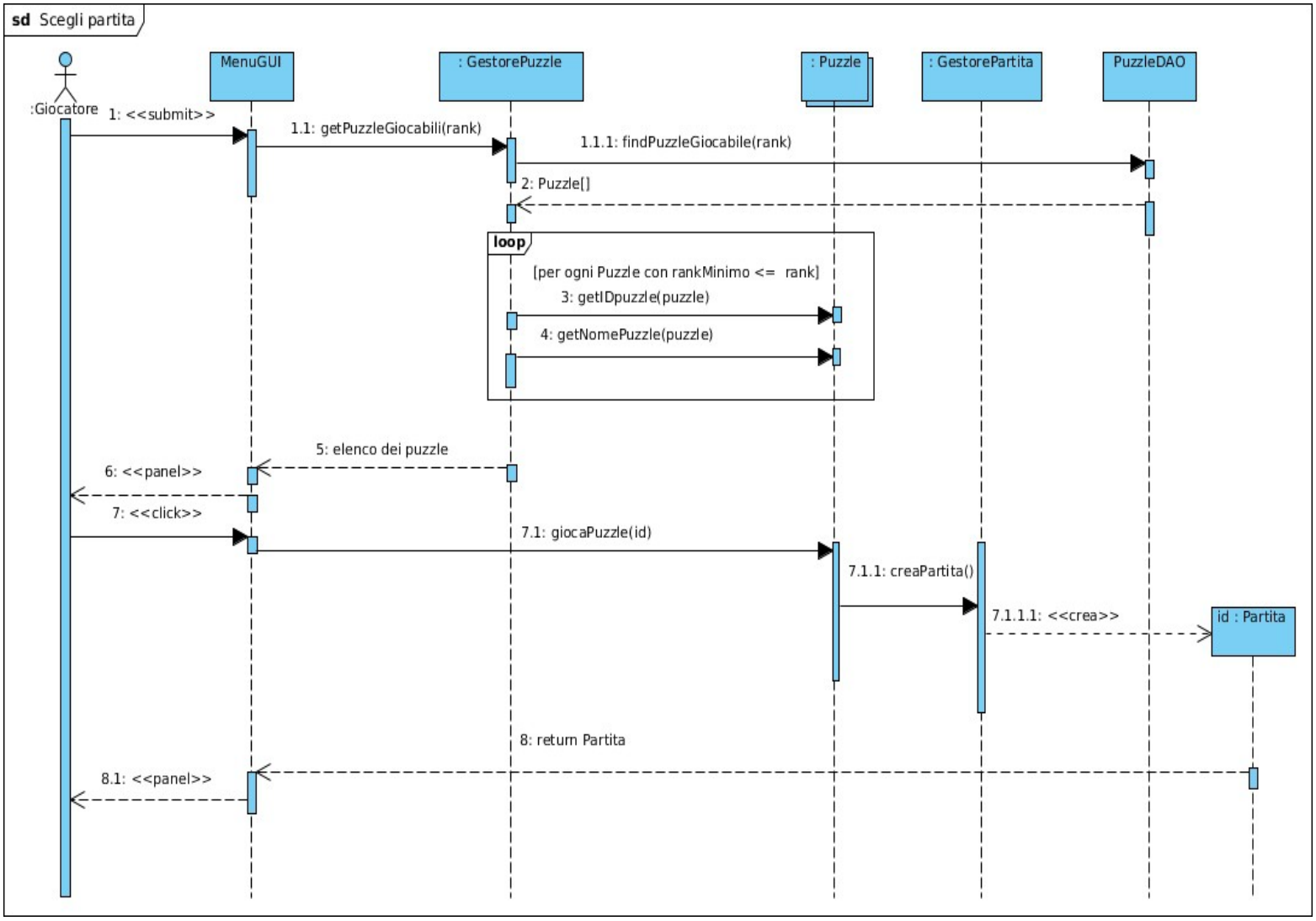
6.3. Modifica Account

Il giocatore seleziona l'opzione modifica account e il sistema risponde mostrando a video una schermata di interazione, dove l'utente ha la possibilità di modificare i propri dati. Prima di ciò, il sistema chiede al giocatore di inserire la propria password accertandosi della sua identità, poi si procede alla modifica vera e propria. Ora l'utente può cambiare la sua mail inserendone una nuova e può eventualmente cambiare anche la password scelta precedentemente.
Tutto si conclude con l'aggiornamento del *database*.



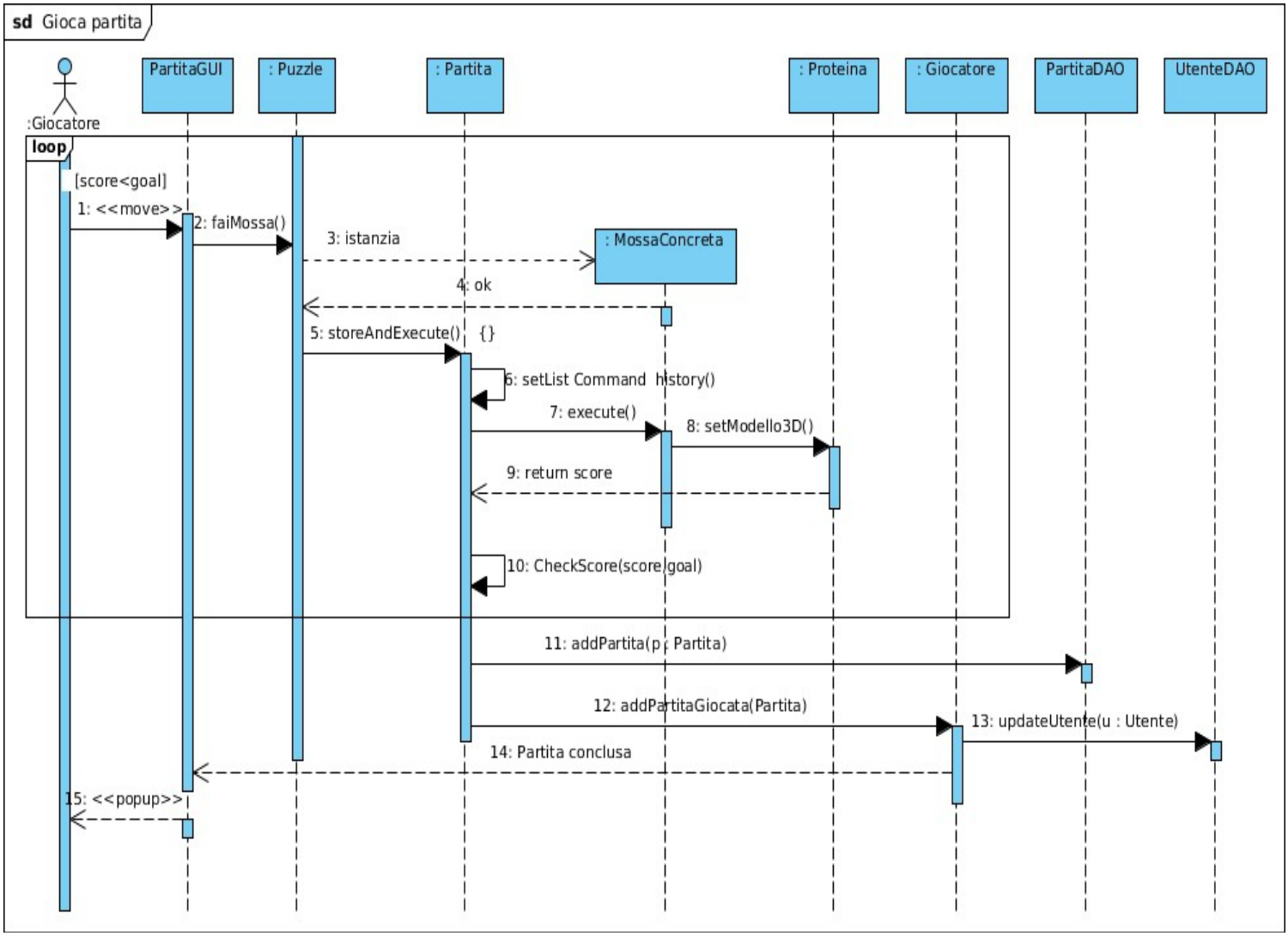
6.4. Scegli partita

L'utente avvia l'opzione di scelta partita, il sistema ritorna l'elenco dei puzzle che può giocare per il suo *rank*. Quindi il *gestorePuzzle* cerca il puzzle da giocare ritornando una lista di possibilità, dopo aver interrogato il *puzzleDAO*. Una volta scelta scelto il puzzle il sistema crea una nuova istanza partita e la predispone per il giocatore, il quale inizierà effettivamente il gioco.



6.5 Gioca Partita

Il giocatore fa una mossa che produca un risultato o meglio un cambiamento all'interno della partita. Tramite la funzione faiMossa() il sistema si avvale di un incapsulamento relativo a un *pattern command*, che isola la mossa agevolando il ripristino della stessa. Per ogni mossa il sistema calcola il punteggio (*score*) e verifica che questo sia >= al valore *goal*, cioè al valore minimo da ottenere per superare la partita in questione. Una volta ultimata e superata la partita, il sistema associa la stessa al giocatore salvandola nel *database* e sblocca nuovi quadri di gioco.



6.6 Crea Puzzle

Il sistema metta a disposizione dell'utente biochimico gli strumenti per creare un nuovo puzzle attraverso la apposita GUI. Il *gestorePuzzle* risponde creando una nuova istanza di puzzle. Il biochimico ora potrà scegliere la base della struttura primaria e creare collegamenti con strutture subalterne come gli aminoacidi.