Design

Diagrammi delle classi e descrizione testuale delle classi

I diagrammi delle classi rappresentano il sistema dell'applicazione per la gestione della biblioteca con le funzionalità di gestire gli utenti, gestire i libri e gestire i prestiti.

-Utente: la classe Utente definisce l'oggetto utente con nome, cognome, matricola, e-mail e lista dei prestiti attivi; contiene i metodi get() e set() e i metodi per il controllo della validità della matricola e dell'e-mail;

-Libro: la classe Libro definisce l'oggetto libro con titolo, lista degli autori, anno di pubblicazione, codice ISBN e numero di copie disponibili; contiene i metodi get() e set() e i metodi necessari per gestire il numero di copie disponibili;

-Prestito: la classe Prestito contiene tutte le informazioni necessarie per identificare un prestito, libro e utente di riferimento, data di inizio prestito, data di scadenza del prestito, data di restituzione effettiva, i metodi get() e set() e possiede un attributo di tipo Boolean che identifica la condizione di ritardo del prestito;

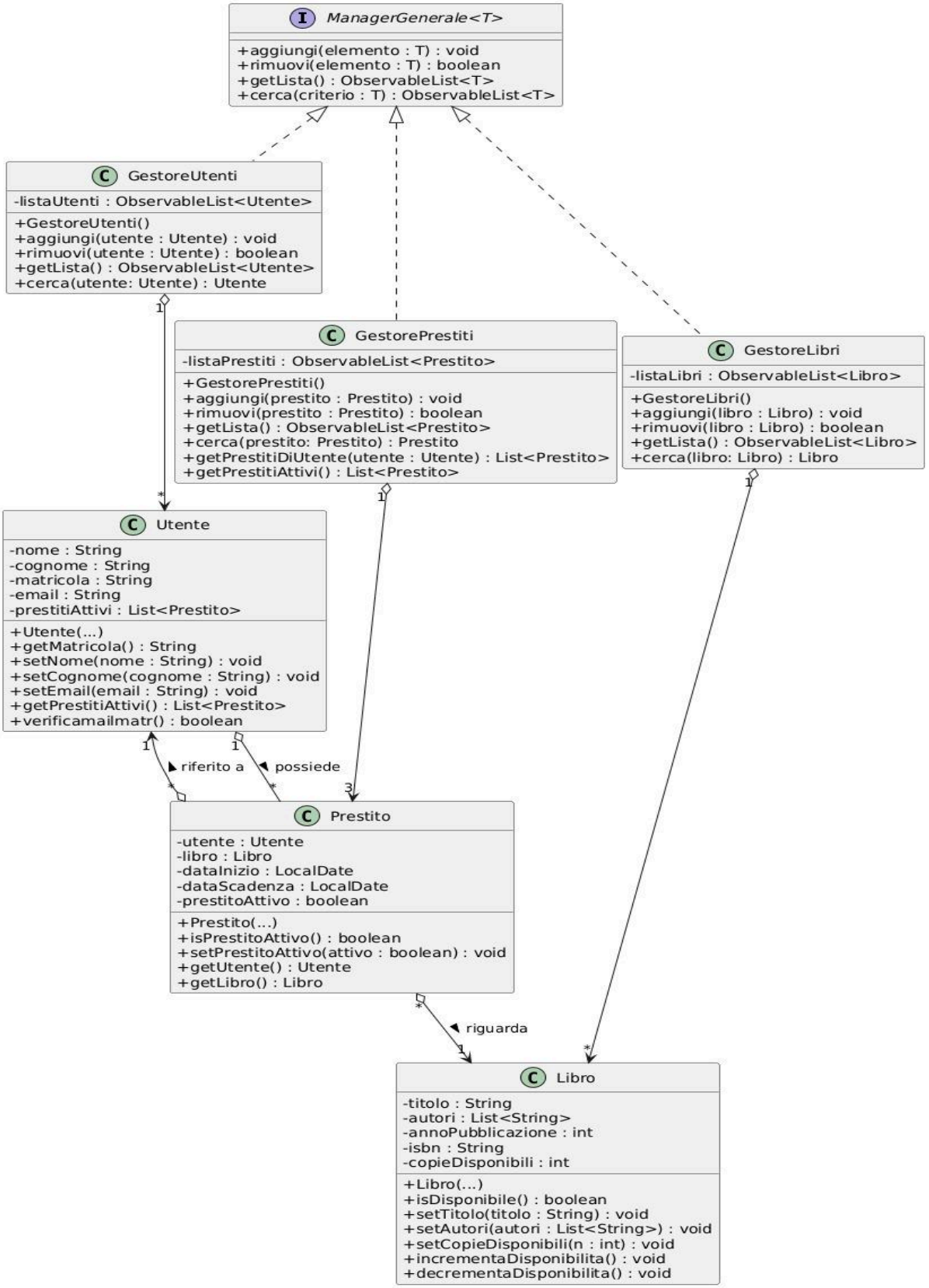
-GestoreUtenti: la classe GestoreUtenti gestisce gli oggetti di tipo Utente; funge da intermediario tra l'interfaccia utente e i dati, e si occupa delle operazioni di inserimento, modifica o rimozione degli Utenti della biblioteca;

-GestoreLibri: la classe GestoreLibri gestisce gli oggetti di tipo Libro; si occupa delle operazioni di inserimento, modifica o rimozione di un Libro dal catalogo mantenendolo aggiornato;

-GestorePrestiti: la classe GestorePrestiti si occupa di registrare, tracciare e storicizzare le associazioni tra Utenti e Libri sotto forma di oggetti di tipo Prestito;

-ManagerGenerale<T>: l'interfaccia ManagerGenerale<T> definisce il comportamento di tutte le classi di gestione del sistema; garantisce l'astrazione e l'uniformità delle operazioni utilizzando il tipo generico <T>.

Diagramma delle classi

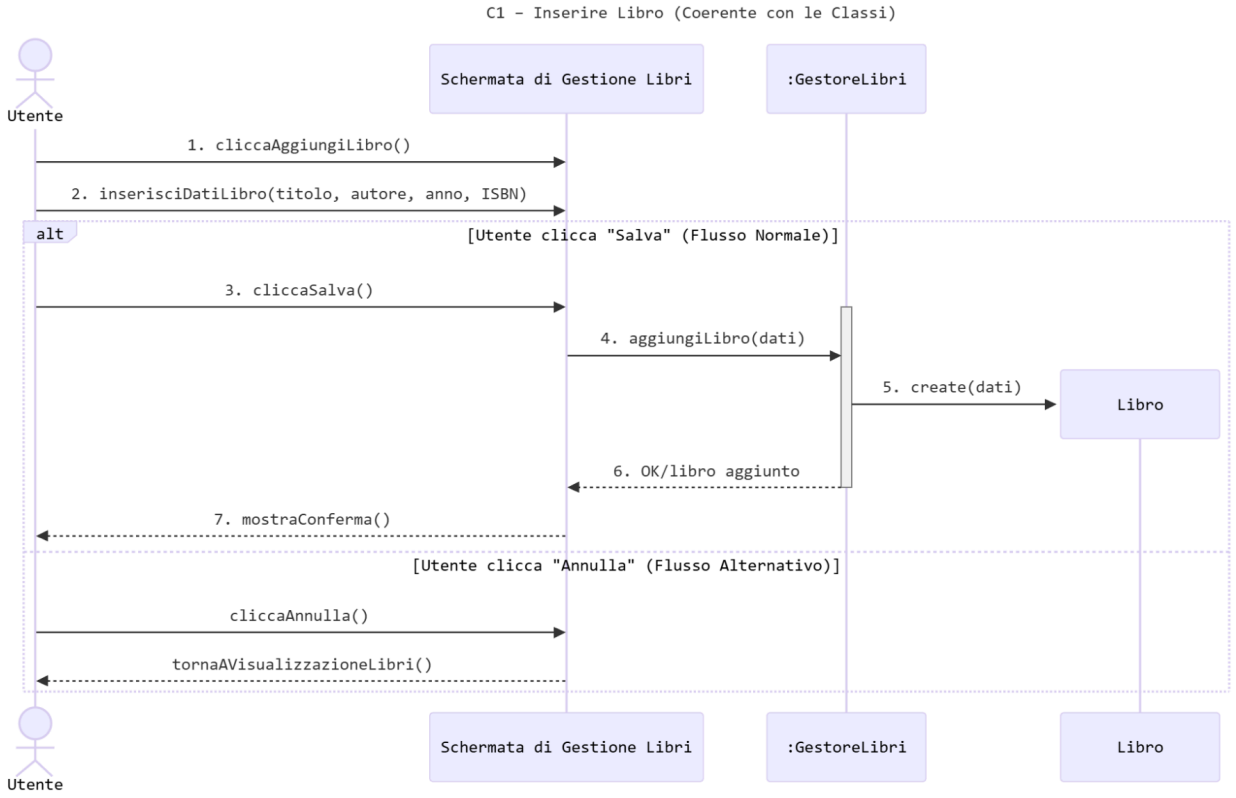


DIAGRAMMI DI SEQUENZA

Vengono presentati i diagrammi di sequenza relativi ai principali casi d'uso del sistema. Ogni diagramma mostra come l'Utente interagisce con le schermate e con i Gestori che gestiscono i dati di libri, utenti e prestiti.

C1 - Inserire libro

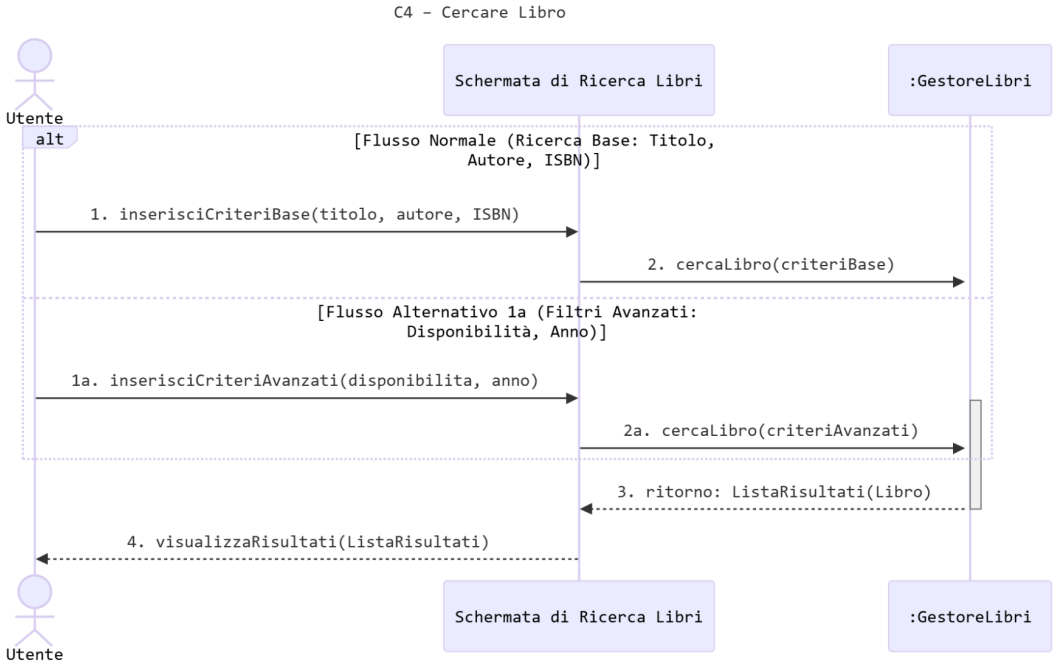
Questo diagramma descrive l'inserimento di un nuovo libro.
L'Utente inserisce i dati richiesti nella **SchermataLibri**, che li invia al **GestoreLibri**. Il gestore crea il nuovo oggetto Libro e lo salva nel sistema. È prevista anche la possibilità che l'Utente annulli l'operazione.



C4 - Cercare libro

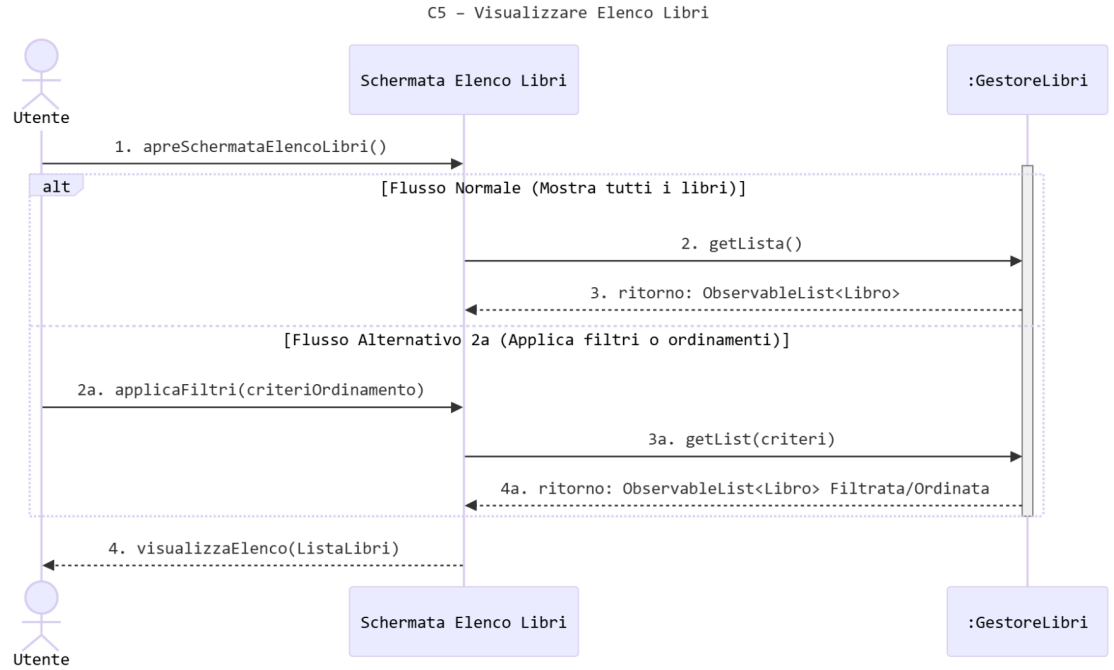
Il diagramma mostra la ricerca dei libri.

L'Utente può scegliere tra ricerca semplice o avanzata. La richiesta viene inviata al **GestoreLibri,** che esegue la ricerca tramite **cercaLibro(criteri)** e restituisce l'elenco dei risultati.



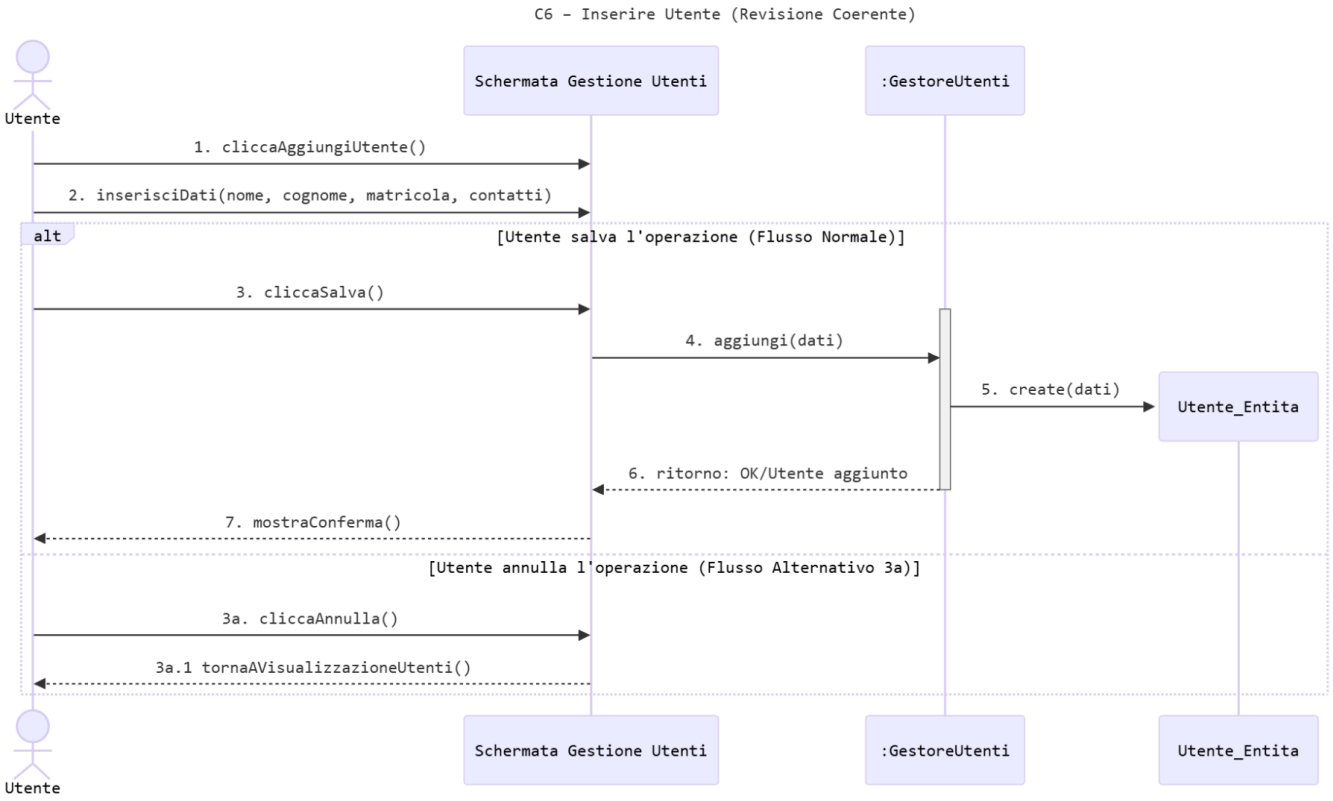
C5 - Visualizzare elenco libri

Questo diagramma rappresenta la visualizzazione del catalogo dei libri. La **SchermataElenco** richiede al **GestoreLibri** l'intera lista (**getList()**). Se l'Utente applica filtri o ordinamenti, il gestore li applica prima di restituire i risultati.



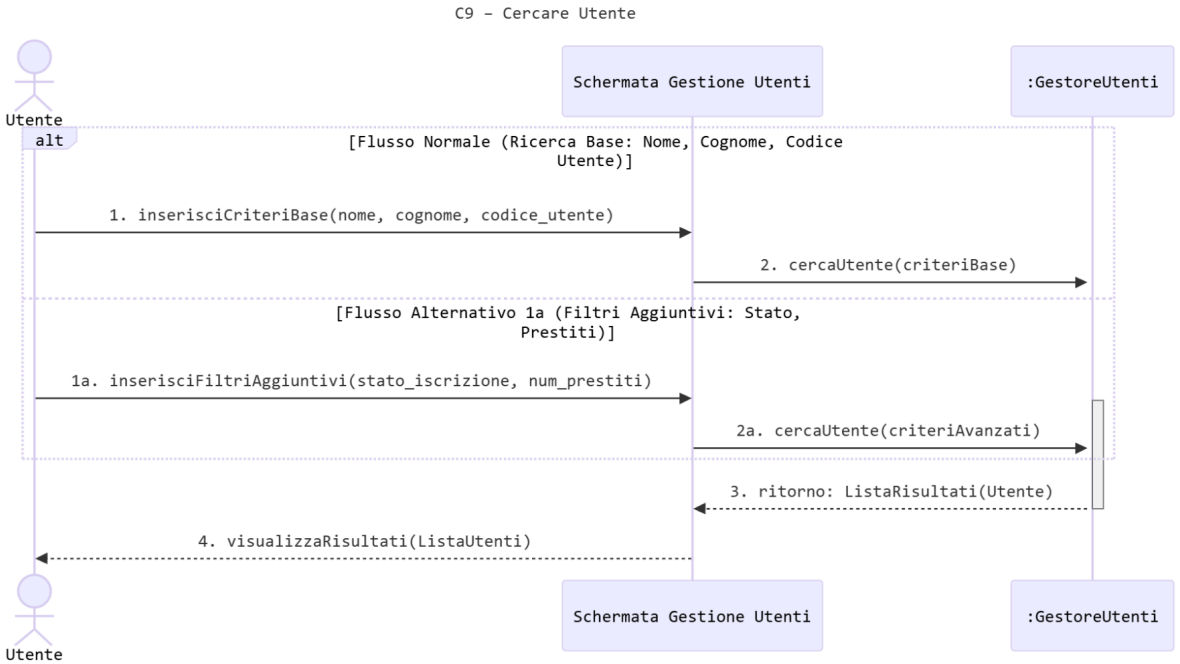
C6 - Inserire Utente

Qui viene mostrata la registrazione di un nuovo utente. La **SchermataUtenti** raccoglie i dati anagrafici e li invia al **GestoreUtenti**, che crea la nuova entità **Utente** tramite **aggiungi(dati)**. Il diagramma prevede anche l'annullamento.



C9 - Cercare utente

Questo diagramma descrive la ricerca degli utenti. L'Utente può effettuare una ricerca semplice o avanzata. La **SchermataUtenti** invia la richiesta al **GestoreUtenti**, che esegue la ricerca con cercaUtente(criteri) e restituisce i risultati.



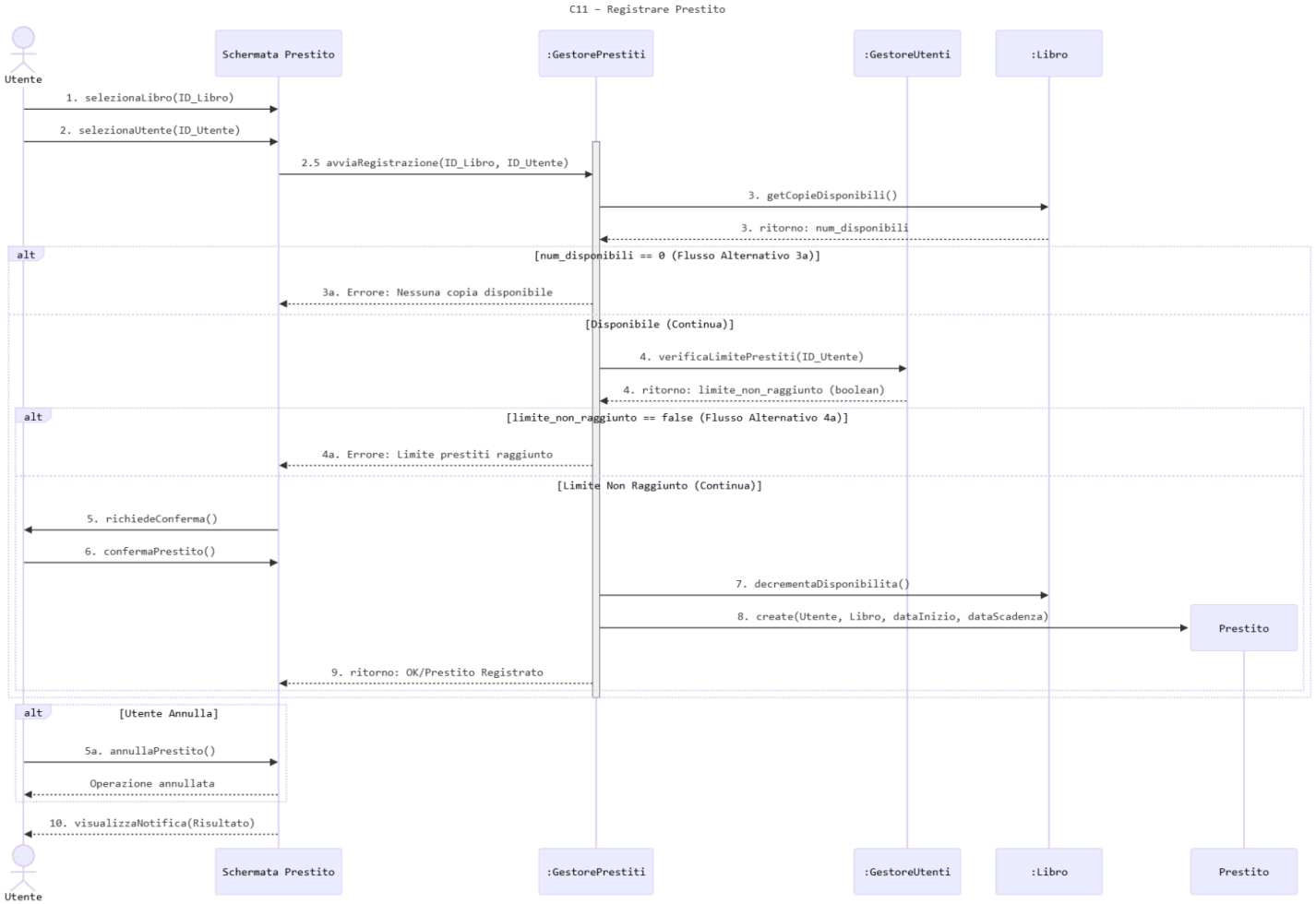
C11 - Registrare prestito

Questo è uno dei casi più importanti.

Il sistema deve:

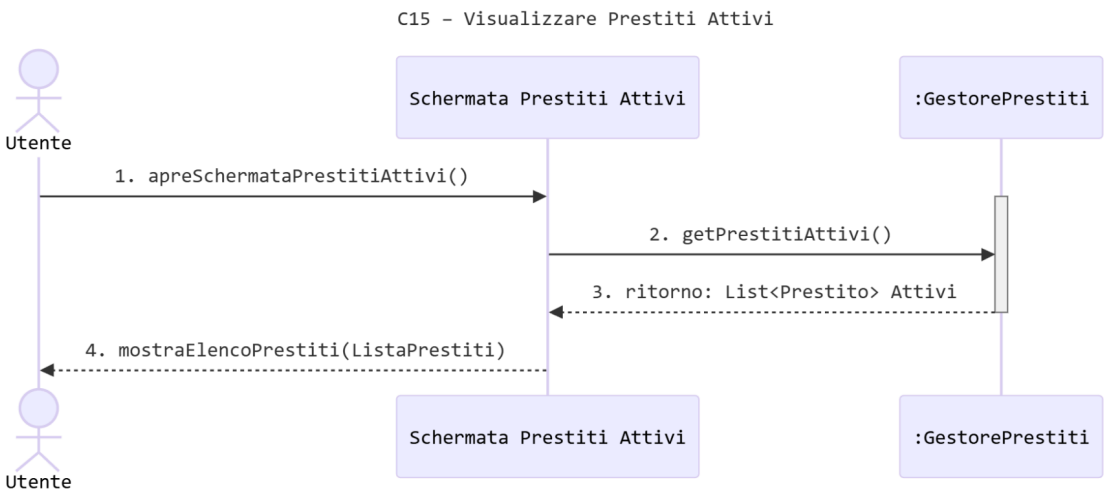
1. Controllare che il libro sia disponibile.
2. Verificare che l'utente non abbia superato il limite di prestiti.

Se entrambi i controlli sono superati, il GestorePrestiti: aggiorna le copie disponibili del libro, e crea il nuovo Prestito con la data di scadenza.



C15 - Visualizzare prestiti attivi

Questo diagramma mostra come vengono visualizzati i prestiti attivi. La richiesta parte dall'Utente e viene gestita dal **GestorePrestiti**, che restituisce alla SchermataPrestiti l'elenco dei prestiti non ancora restituiti tramite getPrestitiAttivi().



Principi di buona progettazione

- Livelli di coesione

Nome classe	Livello coesione	Nota
ManagerGenerale<T>	Funzionale	Tutti i metodi della classe sono dedicati alla gestione generica di una lista. Tutte le operazioni svolgono un singolo compito ben definito.
GestoreUtenti	Funzionale	I metodi della classe lavorano tutti sullo stesso obiettivo: la gestione degli utenti. Tutte le operazioni sono orientate allo stesso scopo.
GestoreLibri	Funzionale	Tutti i metodi gestiscono esclusivamente i libri, mantenendo alta coesione attorno alle operazioni di aggiunta, rimozione e ricerca nella lista dei libri.
GestorePrestiti	Funzionale	Tutti i metodi operano insieme per la gestione dei prestiti, dalla ricerca alla restituzione dei prestiti attivi o associati a un utente. Le operazioni lavorano allo stesso scopo, realizzando un'unica funzionalità centrale.
Utente	Funzionale	La classe concentra tutte le sue operazioni sulla gestione dei dati dell'utente e sulla restituzione dei suoi prestiti, tutte le funzioni collaborano per modellare un utente.
Libro	Funzionale	Tutti i metodi della classe sono adibiti alla gestione e modifica degli attributi del libro. Le operazioni lavorano sullo stesso insieme di dati.
Prestito	Funzionale	I metodi della classe riguardano tutti la gestione del singolo prestito: controllano se è attivo, permettono di modificarne lo stato e restituiscono le informazioni collegate.

- Livelli di accoppiamento

Nome classe 1	Nome classe 2	livello accoppiamento	Nota
GestoreUtenti	Utente	per dati	La classe GestoreUtenti contiene la lista degli Utenti e utilizza i loro riferimenti solo per gestire la creazione, la ricerca e la rimozione di ogni singolo utente.
GestoreLibri	Libro	per dati	Essendoci una relazione 1 a molti, la classe GestoreLibri mantiene i riferimenti ai libri e li usa solo per le operazioni di aggiunta, rimozione o ricerca.
GestorePrestiti	Prestito	per dati	La classe GestorePrestiti conserva la lista dei Prestiti e lavora sui loro dati per gestire le operazioni relative ai prestiti (creazione, rimozione, ricerca).
GestorePrestiti	Utente	per dati	La classe GestorePrestiti utilizza il riferimento all'Utente solo per filtrare i prestiti relativi a quel determinato utente, senza accedere ad altri dati non necessari.
GestorePrestiti	Libro	per dati	La classe GestorePrestiti usa il riferimento al Libro per verificare la disponibilità e collegarlo al prestito, accedendo solo alle informazioni strettamente utili.
Prestito	Utente	per dati	La classe Prestito contiene il riferimento all'Utente solo perché necessario a identificare chi ha effettuato il prestito.
Prestito	Libro	per dati	La classe Prestito mantiene il riferimento alla classe Libro esclusivamente per sapere quale libro è stato preso in prestito.
GestoreUtenti	ManagerGenerale<T>	per timbro	La classe GestoreUtenti eredita da ManagerGenerale<T>, ma utilizza soltanto alcuni dei metodi generici ereditati, ricevendo quindi più operazioni di quelle che realmente le servono.
GestoreLibri	ManagerGenerale<T>	per timbro	La classe GestoreLibri estende ManagerGenerale<T>, ma sfrutta solo una parte delle funzionalità offerte dalla superclasse, usando quindi un "pacchetto" più grande del necessario.
GestorePrestiti	ManagerGenerale<T>	per timbro	Anche GestorePrestiti eredita metodi da ManagerGenerale<T> che non utilizza completamente, mostrando un accoppiamento per timbro.

Relazioni tra classi

Nel progetto è stata privilegiata l'associazione rispetto alla specializzazione (ereditarietà).
In particolare:

- Esiste una relazione di aggregazione 1 a molti tra:
    - GestoreUtenti e Utente,
    - GestoreLibri e Libro,
    - GestorePrestiti e Prestito,
      poiché ciascun gestore mantiene una collezione degli oggetti che amministra.
- È presente una relazione di composizione 1 a molti tra Utente e Prestito, dato che ogni prestito esiste solo se associato a uno specifico utente: eliminando l'utente, cessano di esistere anche tutti i suoi prestiti.
- È presente una relazione di composizione 1 a molti tra Libro e Prestito, poiché un prestito riguarda necessariamente un libro specifico. Se un libro viene rimosso dal sistema, anche i prestiti associati non sono più validi.
- GestorePrestiti è associato sia a Utente che a Libro, poiché per creare o rimuovere un prestito sono necessari entrambi gli oggetti.

Riduzione dell'accoppiamento

Per ridurre il livello di accoppiamento tra le classi, il progetto prevede una separazione netta tra:

- Gestione degli utenti (GestoreUtenti)
- Gestione dei libri (GestoreLibri)
- Gestione dei prestiti (GestorePrestiti)

In questo modo ogni classe gestore dipende solo dai dati strettamente necessari per il proprio ruolo, riducendo sia la dipendenza diretta che la propagazione delle modifiche. Tuttavia:

- GestorePrestiti ha un accoppiamento per dati con Utente e Libro, poiché ha necessariamente bisogno dei loro riferimenti per costruire, ricercare o eliminare un prestito.
- Le classi Utente e Libro hanno un accoppiamento per dati con Prestito, in quanto devono conoscere i prestiti a loro associati.
- Le classi dei gestori hanno un accoppiamento per timbro con la classe generica ManagerGenerale<T>, perché ereditano un insieme di metodi più ampio rispetto a quelli che effettivamente utilizzano. Questo permette di riutilizzare del codice, ma introduce un livello di accoppiamento non completamente ottimale.