UniCTest

Ingegneria del Software – A.A. 2021/2022

Progetto d’esame di: Blanco Francesco Giulio, Anzalone Dario Giuseppe

Fase di Elaborazione – Iterazione 6

Sommario

[1 Versione 2](#_Toc94906839)

[2 Introduzione 2](#_Toc94906840)

[3 Modello di Analisi 3](#_Toc94906841)

[3.1 Aggiornamento Modello di Dominio 3](#_Toc94906842)

[4 Analisi e Progettazione UC9 5](#_Toc94906843)

[4.1 Analisi Orientata agli Oggetti UC9 5](#_Toc94906844)

[4.1.1 Diagramma di Sequenza di Sistema UC9 5](#_Toc94906845)

[4.1.2 Contratti delle Operazioni UC9 5](#_Toc94906846)

[CO1 5](#_Toc94906847)

[CO2 6](#_Toc94906848)

[CO3 6](#_Toc94906849)

[CO4 6](#_Toc94906850)

[CO5 6](#_Toc94906851)

[4.2 Progettazione Orientata agli Oggetti UC9 7](#_Toc94906852)

[4.2.1 Diagrammi di interazione 7](#_Toc94906853)

[4.2.2 DCD 9](#_Toc94906854)

# Versione

|  |  |  |  |
| --- | --- | --- | --- |
| Versione | Data | Descrizione | Autori |
| Elaborazione 6 | 2021/02/06 | Sesta iterazione della fase di Elaborazione. | Anzalone Dario Giuseppe, Blanco Francesco Giulio. |

# Introduzione

I Casi d’Uso già esplorati coprono il *core* del progetto, cioè soddisfano le motivazioni cardine per cui il cliente aveva deciso di commissionare UniCTest: permettere agli studenti di esercitarsi mediante lo svolgimento di simulazioni sul proprio calcolatore, consentendogli di personalizzarle, e permettere ai tutor di creare, svolgere, correggere le simulazioni cartacee con il minore sforzo possibile, lasciando la possibilità di riutilizzare il lavoro fatto in altre simulazioni, collaborare con i colleghi nella realizzazione delle stesse, ecc.

Il cuore del progetto è stato sviluppato facendo affidamento al Caso d’Uso di Avviamento, tramite il quale venivano inseriti nel sistema Tutor, Studenti e Tutor con responsabilità (Ammministratori e TutorSimulazione). Tuttavia, non è ancora possibile aggiungere tali Utenti al Sistema né è possibile aggiungere/rimuovere responsabilità a *runtime*, come promesso dal pattern Decorator applicato nella Iterazione 4 della fase di Elaborazione.

Inoltre, manca lo sviluppo di alcune primitive che consentano di visualizzare il lavoro svolto, ad esempio i test effettuati dallo Studente e i quesiti già inseriti dal Tutor.

Nella Iterazione 6 della fase di Elaborazione, dunque, ci si occuperà di

* UC6: Nuovo Tutor
* UC13: Aggiungi materia insegnata
  + Dovrà prevedere l’applicazione del pattern Flyweight
* UC14: Aggiungi responsabilità   
  (Aggiungi Amministratore/Aggiungi TutorSimulazione)
* UC15: Rimuovi responsabilità   
  (Rimuovi Amministratore/Rimuovi TutorSimulazione)
* UC5: Nuovo Studente
* UC16: Visualizza quesiti creati (CRUD)
  + Visualizzazione quesiti creati da parte del Tutor
* UC17: Visualizza test svolti (CRUD)
  + Visualizzazione test svolti da parte dello Studente

È stato deciso di non sviluppare i Diagrammi di Sequenza di Sistema in quanto i Casi d’Uso citati prevederanno l’uso di una singola operazione e dunque di una singola interazione Attore-Sistema software, pertanto i Diagrammi di Sequenza di Sistema risulterebbero ovvi.

# Modello di Analisi

## Contratti delle Operazioni UC9

Di seguito si riportano le descrizioni delle Operazioni eseguite in riferimento a UC9.

### CO1

|  |  |
| --- | --- |
| Operazione | visualizzaTemplate() |
| Riferimenti | UC9: Componi test per simulazione cartacea |
| Pre-condizioni | - |
| Post-condizioni | * È stato restituito l’elenco delle istanze *te* di Template al TutorSimulazione *utenteAutenticato*. |

### CO2

|  |  |
| --- | --- |
| Operazione | creaTestCartaceo(idTemplate) |
| Riferimenti | UC9: Componi test per simulazione cartacea |
| Pre-condizioni | - |
| Post-condizioni | * È stata creata l’istanza *t* di Test ed è stata associata all’istanza *te* di Template avente *te.id*=idTemplate tramite l’associazione “corrente”. |

### CO3

|  |  |
| --- | --- |
| Operazione | visualizzaQuesiti(idSezione) |
| Riferimenti | UC9: Componi test per simulazione cartacea |
| Pre-condizioni | È in corso la creazione di un test *t* per una simulazione cartacea. |
| Post-condizioni | * È stato restituito al TutorSimulazione *utenteAutenticato* l’elenco delle istanze *qd* di QuesitoDescrizione associate all’istanza *m* di Materia associata all’istanza *s* di Sezione avente *s*.id uguale a idSezione. |

### CO4

|  |  |
| --- | --- |
| Operazione | inserisciQuesiti(listaIdQuesiti) |
| Riferimenti | UC9: Componi test per simulazione cartacea |
| Pre-condizioni | È in corso la creazione di un test *t* per una simulazione cartacea. |
| Post-condizioni | * Sono state create le istanza *qr* di QuesitoReale, che sono state associata al Test *t* tramite l’associazione “contiene” e sono state associate alle istanze *qd* aventi *qd*.id uguali a quelli in listaIdQuesiti tramite l’associazione “descritto da”. |

### CO5

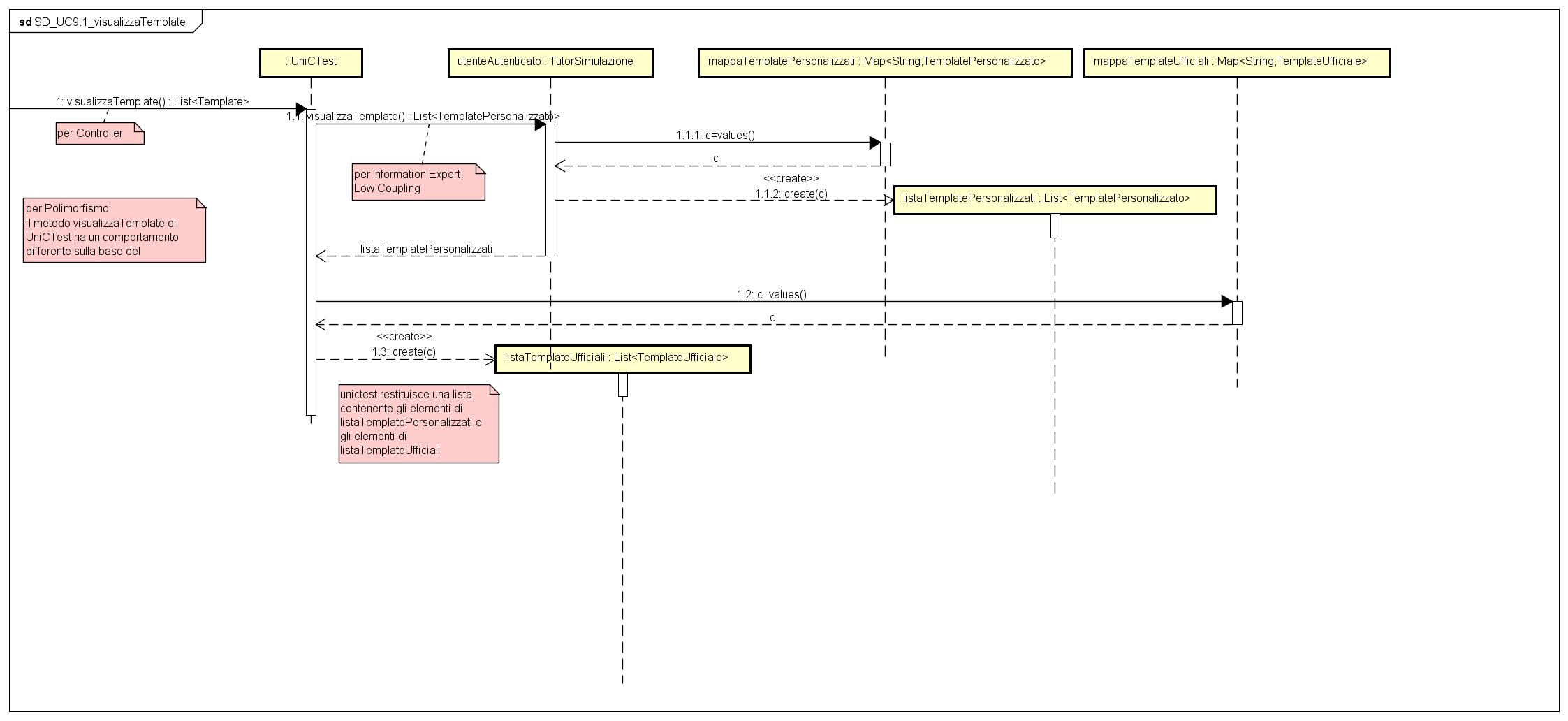
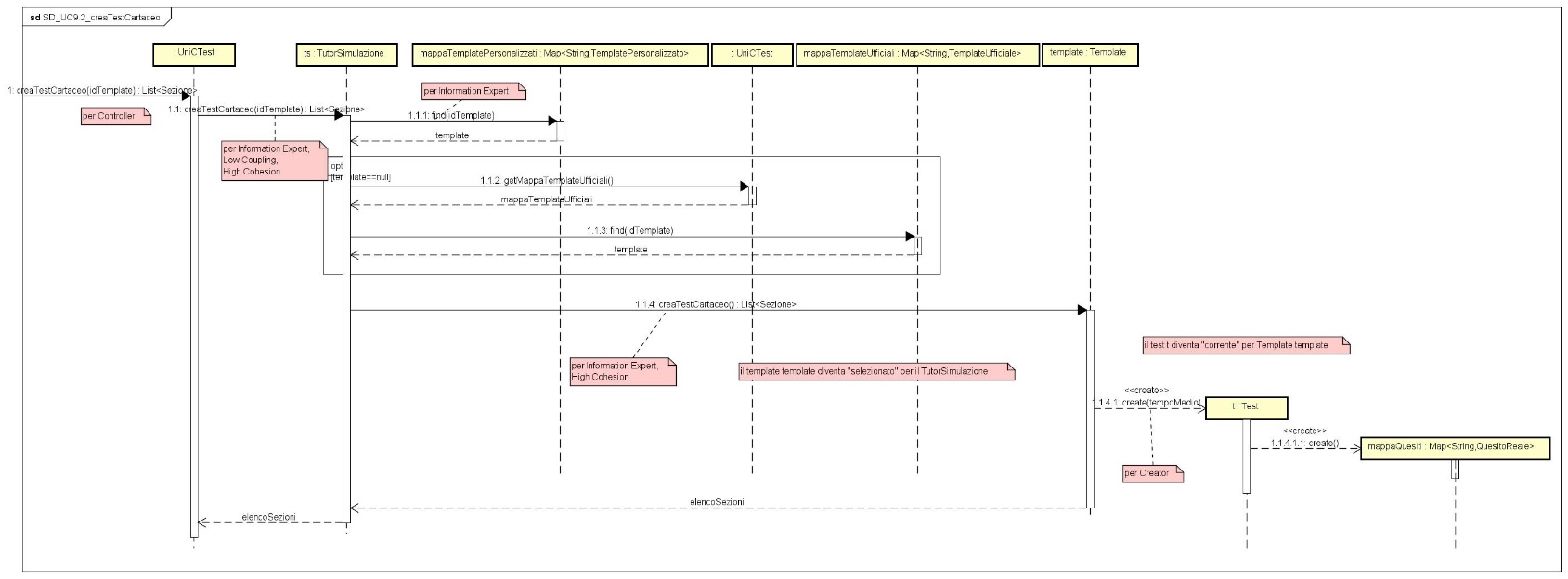
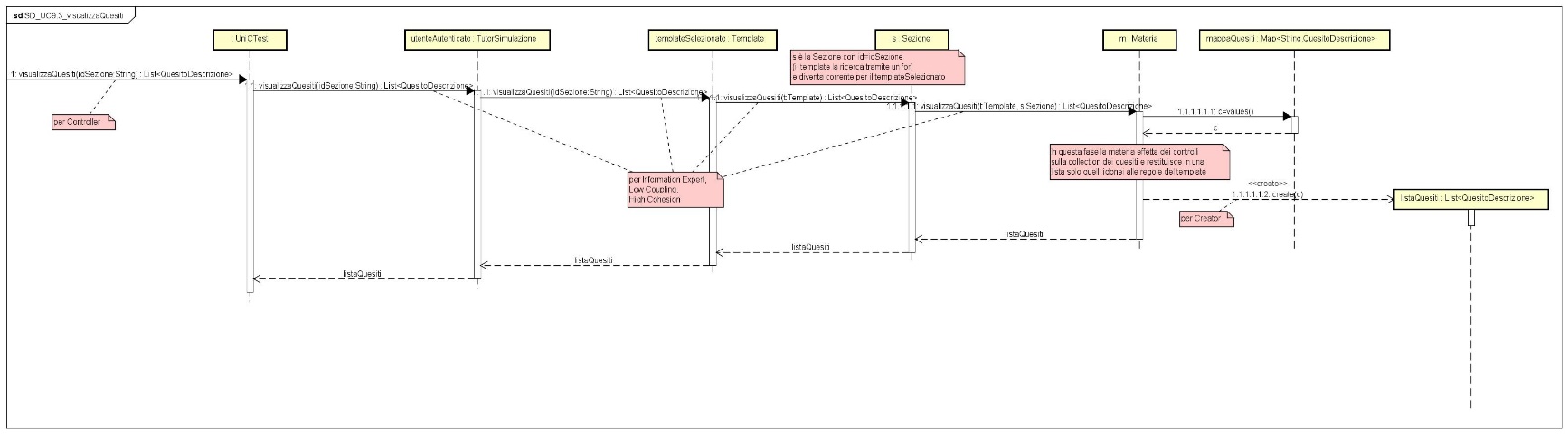
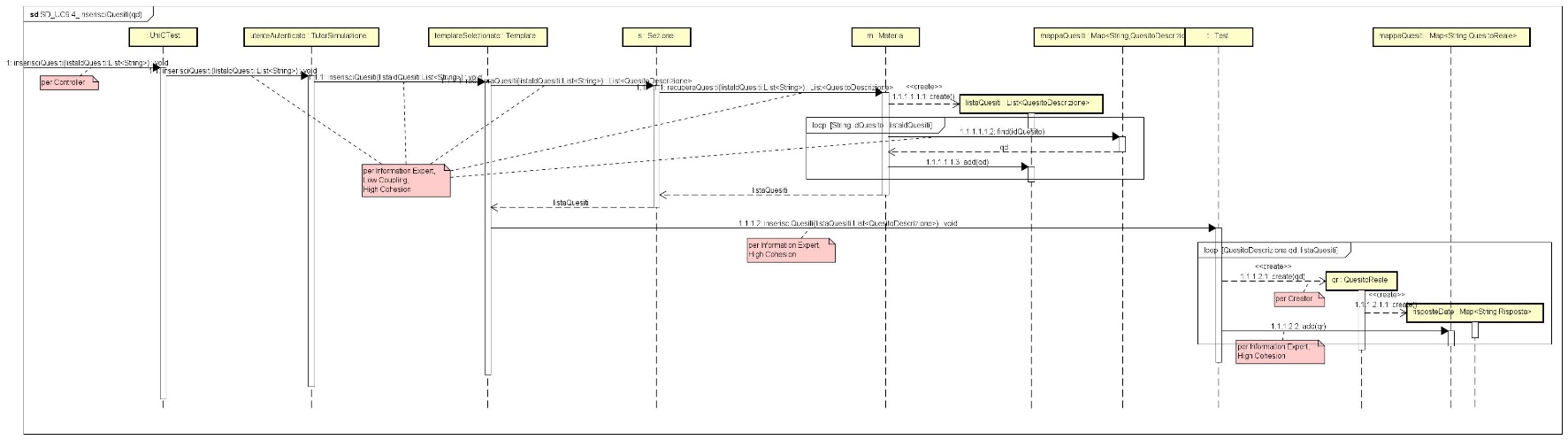
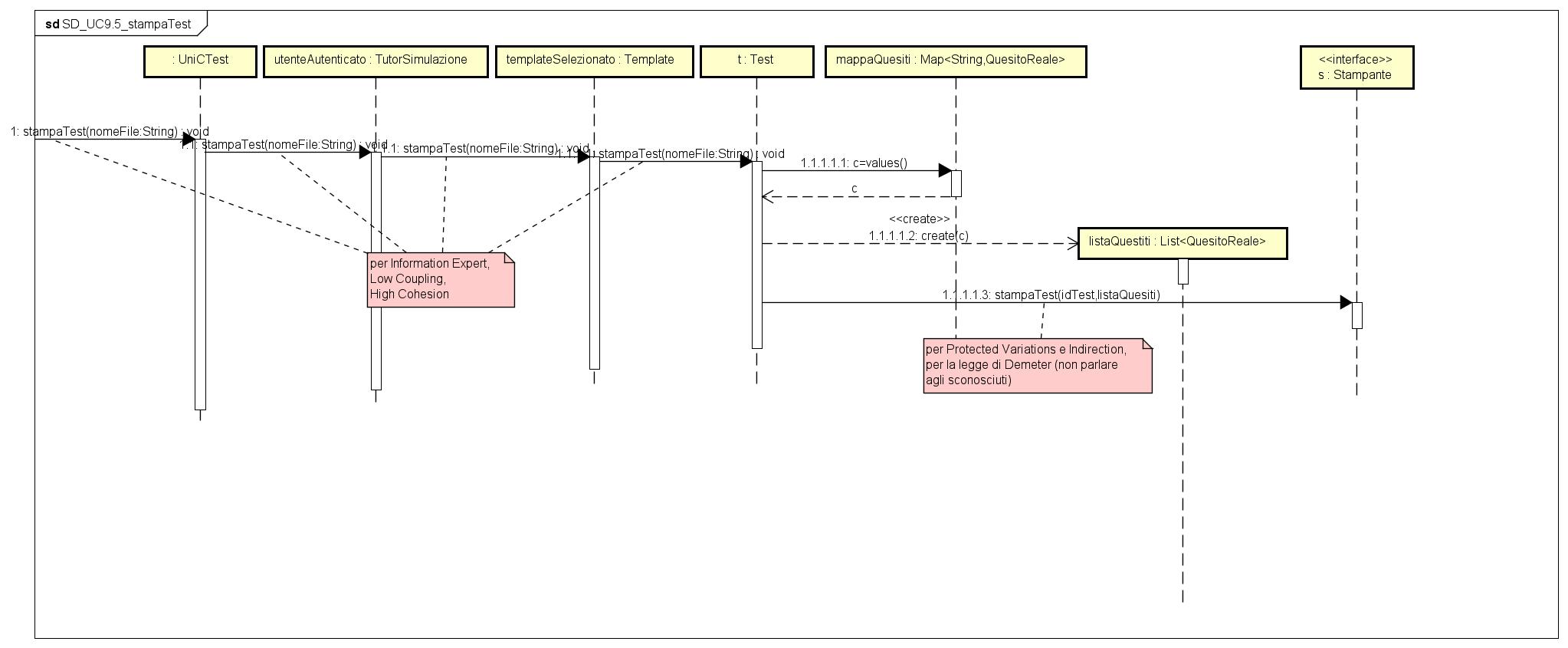
|  |  |
| --- | --- |
| Operazione | stampaTest(nomeFile) |
| Riferimenti | UC9: Componi test per simulazione cartacea |
| Pre-condizioni | È in corso la creazione di un test *t* per una simulazione cartacea. |
| Post-condizioni | * Il Test e il ModuloRiposte sono stati restituiti al Tutor. |

## Progettazione Orientata agli Oggetti UC9

Aggiornate le classi concettuali nel contesto della OOA, si passa alla OOD, realizzando il Modello di Progetto ed in particolare aggiornando il DCD (visualizzazione statica delle classi software), aggiornamento da effettuare il parallelo alla realizzazione dei diagrammi di interazione (visualizzazione dinamica delle classi software).

Bisogna notare che la classe concettuale ModuloRisposte non ha trovato una concretizzazione lato Software in quanto altre classi possono sopperire alla mancanza di un modulo risposte. In particolare è QuesitoReale a fare le veci di un modulo risposte.

### Diagrammi di interazione

1. **SD\_UC9\_visualizzaTemplate**
2. **SD\_UC9\_creaTestCartaceo**
3. **SD\_UC9\_visualizzaQuesiti**
4. **SD\_UC9\_inserisciQuesiti**
5. **SD\_UC9\_stampaTest**

### Implementazione e Testing

#### Implementazione

Per l’Implementazione si rimanda al codice in allegato.

#### Testing

**TestAggiungiResponsabilità.java**

**Si tratta di test sviluppati per testare i Casi d’Uso UC14 e UC15, ovvero l’aggiunta e la rimozione delle responsabilità addizionali di un Tutor (pattern GoF Decorator).**

Sono state sviluppate due tipologie di test differenti:

* Test a **scatola nera (black-box)**. Lo sviluppo dei test black-box è stato promosso dalle pratiche di test-driven development introdotte nell’Iterazione 4 della fase di Elaborazione. I test effettuati sono stati sviluppati prescindendo dalla conoscenza dei dettagli implementativi del codice, ma hanno analizzato il blocco software di assegnazione delle responsabilità addizionali ai Tutor come un sistema ingresso-uscita.
* Testa a **scatola bianca (white-box)**. Sono stati effettuati test a scatola bianca con ***copertura delle decisioni*** (che include la copertura delle istruzioni).   
  L’obiettivo della copertura delle decisioni è individuare un insieme di casi di test in modo da garantire che ogni decisione assuma il valore vero almeno una volta.  
  In sostanza è stato deciso di testare se un Tutor che era stato reso prima di tutto TutorSimulazione e solo dopo Amministratore veniva correttamente visto come TutorSimulazione. Allo stesso modo doveva essere verificato che un Tutor che era stato reso prima di tutto TutorSimulazione e solo dopo Amministratore veniva correttamente NON visto come TutorSimulazione una volta chiamata la funzione rimuoviTutorSimulazione.  
  Questi test non sarebbero stati effettuati se fossero stati elaborati solo test a scatola nera. Infatti la decisione di testare questa condizione deriva dalla conoscenza del workflow interno delle funzioni per aggiungere e rimuovere le responsabilità addizionali. Infatti:

@Override  
public Impiegato rimuoviAmministratore(Impiegato imp){  
 Impiegato attuale = imp;  
 Impiegato precedente = null;  
 Impiegato successivo = imp.getImpiegato();   
 //Struttura: PREC(ATTUALE(SUCC))  
 while(successivo != null){  
 if(attuale instanceof Amministratore){  
 if(precedente!=null){  
 precedente.setImpiegato(successivo);  
 return imp;  
 } else {  
 return successivo;  
 }  
 } else { //Scaliamo tutti al successivo  
 precedente = attuale; //Il PREC diventa l'ATT  
 attuale = successivo; //L'ATT diventa il SUCC  
 successivo = successivo.getImpiegato(); //Il SUCC diventa più SUCC  
 }  
 } //Interrompo se lo trovo o se non ho più successivi (cioè più interni)! --> Infatti il Tutor restituisce null perché non ha interni.  
 System.*out*.println("Non sei Amministratore, quindi non rimuovo nulla");  
 return imp;  
}

Dunque bisogna prevedere:

* + Il caso in cui *successivo*=*null*, e cioè il caso in cui l’Impiegato è un Tutor, dunque non ha responsabilità addizionali.
  + Nel caso in cui *successivo* non sia *null*,
    - Il caso in cui il *successivo* sia un Amministratore
      * Il caso in cui il precedente sia *null*, e cioè il caso (banale) in cui l’elemento passato è esso stesso un Amministratore che *wrappa* altre responsabilità al proprio interno
      * Il caso in cui il precedente non sia *null*, e dunque bisogna ricostruire la catena di impiegati rimuovendo quello *attuale*, e cioè PREC(ATT(SUCC))) diventa PREC(SUCC).
    - Il caso in cui i*successivo* non sia un Amministratore, in tal caso la ricerca deve procedere esplorando le responsabilità più interne.
* Di conseguenza, vengono analizzati i casi in cui la responsabilità cercata sia “esterna” e in cui sia “interna”.

Funzionalità testate:

* Test a scatola nera:
  + Diritti di Amministratore: è impossibile per un non-Amministratore aggiungere o rimuovere responsabilità.
  + Un Tutor reso Amministratore viene visto correttamente come Amministratore.
  + Un Tutor reso TutorSimulazione viene visto correttamente come TutorSimulazione.
  + Un Tutor reso Amministratore non può auto-rimuovere i diritti di Amministratore.
* Test a scatola bianca:
  + Un Tutor reso Amministratore e in seguito TutorSimulazione viene visto correttamente come Amministratore
  + Un Tutor reso Amministratore e in seguito TutorSimulazione viene visto correttamente come TutorSimulazione