Fase di Elaborazione
Iterazione 4

Sommario

1. Workshop dei requisiti

Nella quarta ed ultima iterazione vengono trattati i casi d'uso rimanenti ovvero UC5, UC8, UC9, UC10 e UC14.

Per ciascuno di essi segue la descrizione in formato dettagliato.

1.1. UC5: Visualizza Proiezioni Sale

Nome del caso d'uso	**Visualizza Proiezioni Sale**
Portata	Applicazione EasyCinema
Livello	Obiettivo utente
Attore primario	Titolare
Parti interessate e interessi	- **Titolare**: il Titolare vuole conoscere le proiezioni in corso nelle varie sale del cinema.
Pre-condizioni	Il Titolare è identificato e autenticato (Login).
Post-condizioni	Le proiezioni vengono mostrate correttamente.
Scenario principale di successo	1. Il Titolare richiede la proiezione in corso nelle varie sale. 2. Il Sistema restituisce per ciascuna sala la proiezione in corso.
Estensioni	2a. Una o più sale non hanno una proiezione in corso: 1. Il Sistema informa l'utente. 2. Il Sistema restituisce la prossima proiezione in programma (se presente).
Requisiti speciali	
Elenco delle varianti tecnologiche e dei dati	
Frequenza di ripetizione	Più volte al giorno.
Varie	

1.2. UC8: Ricarica Credito Cliente

Nome del caso d'uso	**Ricarica Credito Cliente**
Portata	Applicazione EasyCinema
Livello	Obiettivo utente
Attore primario	Titolare
Parti interessate e interessi	- **Titolare**: il Titolare intende aggiungere una somma di denaro sul conto del Cliente così da permettergli di effettuare nuove prenotazioni. - **Cliente**: il Cliente vuole che la somma versata fisicamente al Titolare sia interamente e istantaneamente aggiunta sul suo conto divenendo utilizzabile.
Pre-condizioni	Il Titolare è identificato e autenticato (Login).

Post-condizioni	Il credito del Cliente è stato modificato correttamente.
Scenario principale di successo	1. Il Sistema chiede al Titolare il codice fiscale del cliente e l'importo. 2. Il Titolare inserisce le informazioni richieste. 3. Il Sistema verifica la correttezza delle informazioni. 4. Il Sistema modifica il credito associato al Cliente.
Estensioni	3a. Codice fiscale non valido: 1. Il Sistema comunica l'errore al Titolare. 2. Il Titolare ritenta con un diverso codice fiscale. 3b. Importo non valido: 1. Il Sistema comunica l'errore al Titolare. 2. Il Titolare ritenta con un importo valido.
Requisiti speciali	
Elenco delle varianti tecnologiche e dei dati	
Frequenza di ripetizione	Non molto frequente.
Varie	

1.3. UC9: Gestisci Promozioni (CRUD)

Nome del caso d'uso	**Gestisci Promozioni**
Portata	Applicazione EasyCinema
Livello	Obiettivo utente
Attore primario	Titolare
Parti interessate e interessi	- **Titolare**: il Titolare intende aggiungere, visualizzare, modificare ed eliminare le promozioni attive nel cinema.
Pre-condizioni	Il Titolare è identificato e autenticato (Login).
Post-condizioni	Consistenza dei dati.
Scenario principale di successo	1. Il Titolare richiede di aggiungere una nuova promozione. 2. Il Sistema restituisce le diverse tipologie di promozione a disposizione. 3. Il Titolare immette la tipologia di promozione di interesse. 4. Il Sistema richiede la condizione per cui si applica la promozione e lo sconto (in termini di percentuale). 5. Il Sistema crea la nuova promozione e notifica al Titolare l'esito positivo dell'operazione.

Estensioni	1a. Il Titolare richiede la **visualizzazione** di tutte le promozioni esistenti filtrate opportunamente: Guarda caso d'uso UC14: Visualizza Promozioni. 1b. Il Titolare richiede la **modifica** di una promozione: 1. Il Titolare seleziona la promozione da modificare. 2. Il Sistema presenta al titolare i dati attuali. 3. Il Titolare apporta i cambiamenti necessari. 4. Il Sistema aggiorna le informazioni della promozione. 1c. Il Titolare richiede l'**eliminazione** di una promozione. 1. Il Titolare seleziona la promozione da eliminare. 2. Il Sistema chiede conferma dell'operazione. 3. Il Titolare conferma. 4. Il Sistema rimuove la promozione. 3a. La tipologia di promozione indicata non esiste: 1. Il Sistema segnala l'errore. 4a. La condizione o lo sconto non sono validi: 1. Il Sistema segnala l'errore. 2. Il Titolare riprova l'inserimento.
Requisiti speciali	
Elenco delle varianti tecnologiche e dei dati	
Frequenza di ripetizione	Occasionalmente.
Varie	

1.4. UC10: Gestisci Profilo (CRUD)

Nome del caso d'uso	**Gestisci Profilo**
Portata	Applicazione EasyCinema
Livello	Obiettivo utente
Attore primario	Utente
Parti interessate e interessi	- **Utente**: vuole avere una panoramica chiara delle informazioni con cui è registrato al Sistema.
Pre-condizioni	L'Utente è identificato e autenticato (Login).
Post-condizioni	Consistenza dei dati.
Scenario principale di successo	1. L'Utente richiede di visualizzare i dati personali. 2. 2. Il Sistema mostra i dati dell'utente.
Estensioni	1a. L'Utente richiede di modificare la password:

	1. Il Sistema richiede l'inserimento della nuova password. 2. L'Utente inserisce la nuova password. 3. 3. Il Sistema informa l'Utente dell'avvenuta modifica.
Requisiti speciali	
Elenco delle varianti tecnologiche e dei dati	
Frequenza di ripetizione	Piuttosto frequente, specialmente per il Cliente.
Varie	

1.5. UC14: Visualizza Promozioni

Nome del caso d'uso	**Visualizza Promozioni**
Portata	Applicazione EasyCinema
Livello	Obiettivo utente
Attore primario	Utente
Parti interessate e interessi	- **Utente:** L'Utente intende conoscere le promozioni attive in EasyCinema.
Pre-condizioni	L'Utente è identificato e autenticato (Login).
Post-condizioni	Tutte le promozioni sono state visualizzate.
Scenario principale di successo	1. L'Utente richiede la visualizzazione di tutte le promozioni attive. 2. Il Sistema mostra le prenotazioni attive.
Estensioni	
Requisiti speciali	
Elenco delle varianti tecnologiche e dei dati	
Frequenza di ripetizione	Sporadica.
Varie	

Al termine del workshop il 100% dei requisiti (a livello utente) è stato descritto in dettaglio.

2. Analisi

2.1. Introduzione

Per la quarta iterazione, si è scelto di attenzionare lo scenario principale di successo dei seguenti requisiti:

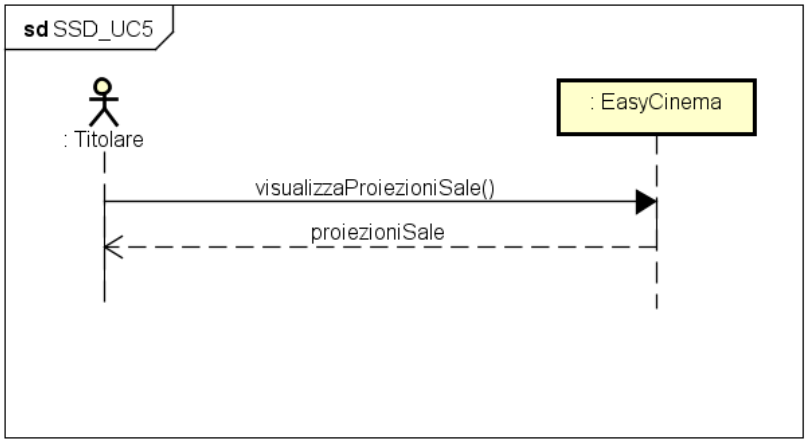
- Caso d'uso a livello utente UC5 (*Visualizza Proiezioni Sale*).
- Caso d'uso a livello utente UC8 (*Ricarica Credito Cliente*).
- Caso d'uso a livello utente UC9 (*Gestisci Promozioni*).
- Caso d'uso a livello utente UC10 (*Gestisci Profilo*).
- Caso d'uso a livello utente UC14 (*Visualizza Promozioni*).

In questo capitolo viene descritta l'analisi svolta in tale iterazione, considerando separatamente i casi d'uso di interesse.

2.2. Caso d'uso UC5, Modello di dominio

Per il caso d'uso in esame non è necessario modificare il modello di dominio ottenuto dalle precedenti iterazioni in quanto i concetti di Proiezione e Sala sono stati già trattati.

2.3. Caso d'uso UC5, Diagramma di sequenza di sistema



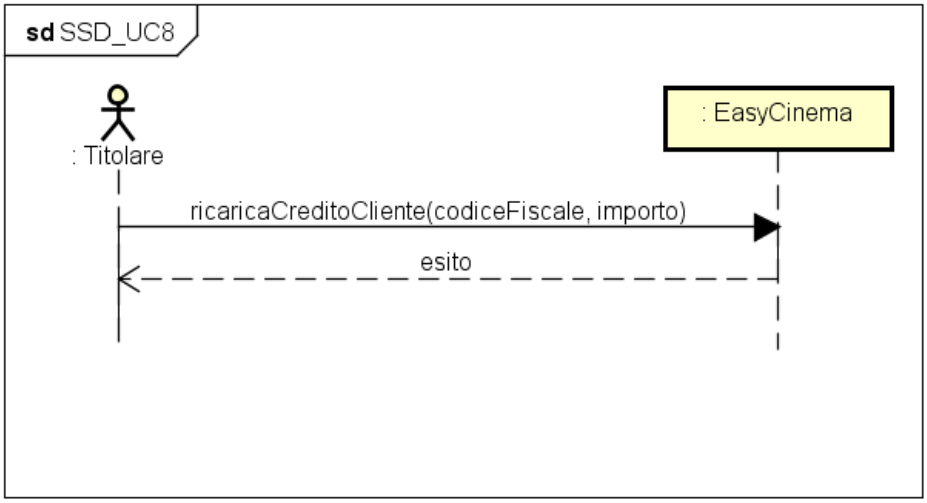
2.4. Caso d'uso UC5, Contratti delle operazioni

L'unica operazione di sistema individuata, essendo un'interrogazione, non altera lo stato degli oggetti del Modello di dominio. Per quanto detto non viene riportato il contratto di tale operazione.

2.5. Caso d'uso UC8, Modello di dominio

Così come per il caso d'uso UC5, anche per il caso d'uso UC8 non è richiesta alcuna modifica al modello di dominio.

2.6. Caso d'uso UC8, Diagramma di sequenza di sistema



2.7. Caso d'uso UC8, Contratti delle operazioni

Operazione	ricaricaCreditoCliente(String codiceFiscale, double importo)
Riferimenti	Caso d'uso: Ricarica Credito Cliente
Pre-condizioni	- il Titolare t sta utilizzando il sistema - il codiceFiscale inserito corrisponde al Cliente c
Post-condizioni	- l'attributo credito di c è stato modificato di una quantità pari a importo.

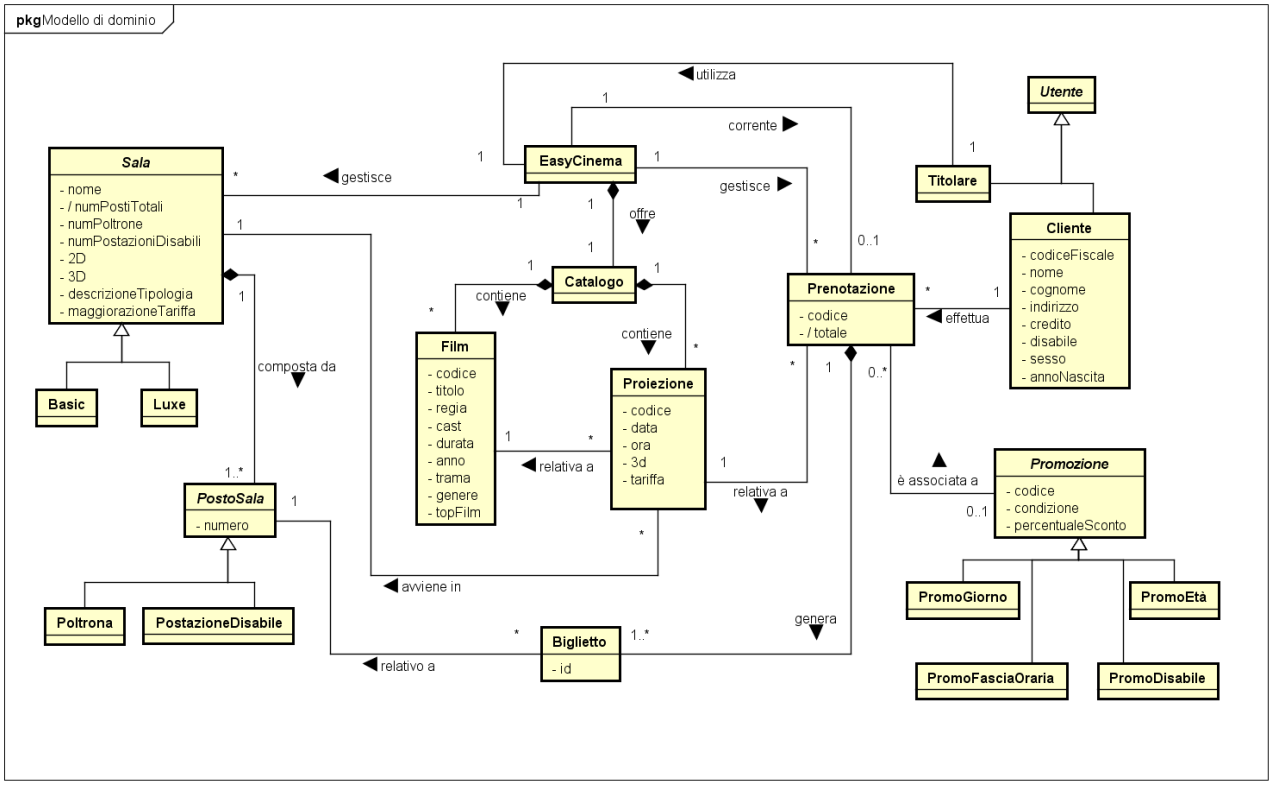
2.8. Caso d'uso UC9, Modello di dominio

L'introduzione del concetto di promozione comporta la creazione della classe concettuale Promozione. Consultando la specifica dei requisiti emergono diverse tipologie di promozioni in base a: disabilità, giorno (per motivi di praticità piuttosto che fare riferimento ad un giorno della settimana si fa riferimento ad un giorno dell'anno), età e fascia oraria.

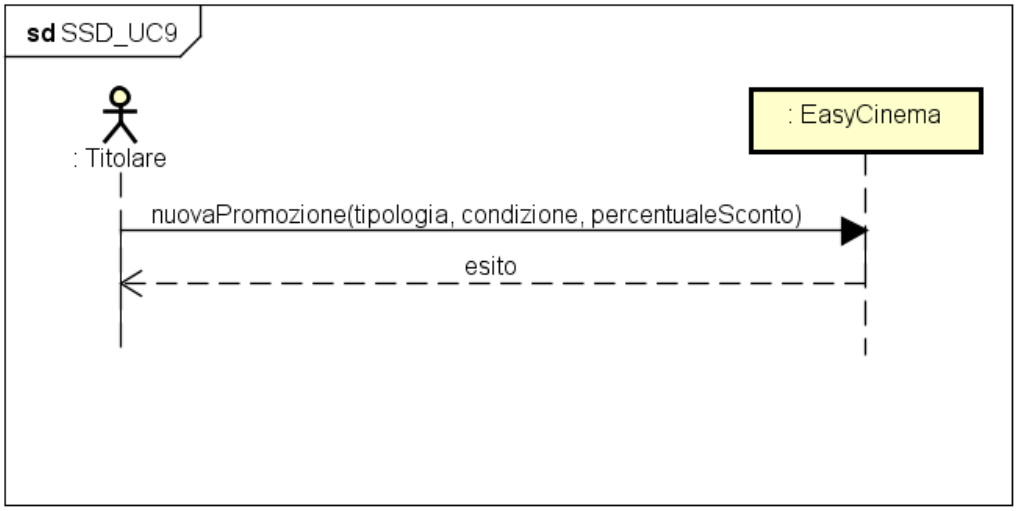
Da queste considerazioni vengono create le classi concettuali PromoDisabile, PromoGiorno, PromoEtà, PromoFasciaOraria. Esse sono specializzazioni di Promozione poiché rispettano la regola *is-a* e del *100%*. Inoltre, comprendono tutte le possibili istanze di Promozione, di conseguenza quest'ultima viene resa astratta.

Infine, per poter gestire tali tipologie di promozioni occorre introdurre gli attributi sesso (per ottenere promozioni del tipo: il lunedì c'è lo sconto per le donne) e anno di nascita su Cliente. L'aggiunta di questi attributi determina una leggera modifica alla classe concettuale Cliente.

Tali considerazioni portano alla revisione del modello di dominio. Il risultato di tale azione è di seguito riportato.



2.9. Caso d'uso UC9, Diagramma di sequenza di sistema



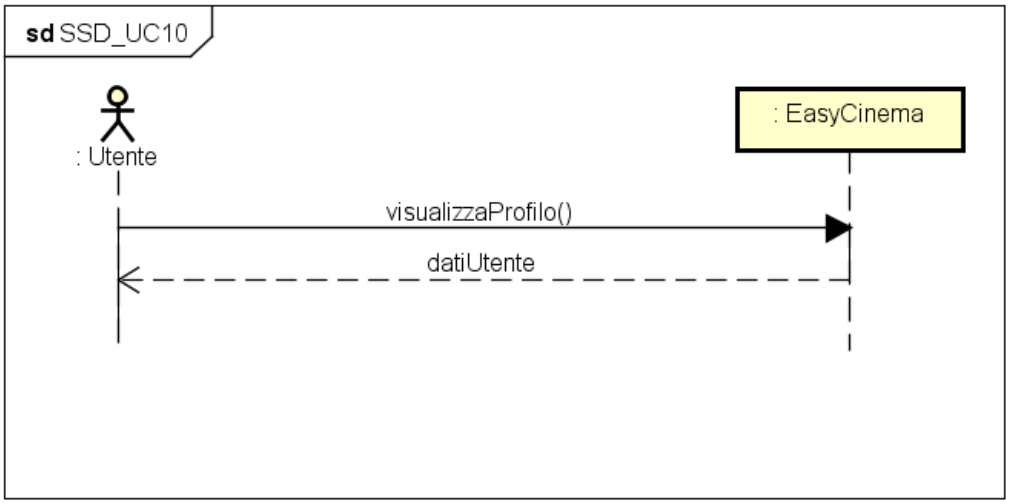
2.10. Caso d'uso UC9, Contratti delle operazioni

Operazione	nuovaPromozione(tipologia:String, condizione: List<String>, percentualeSconto: int)
Riferimenti	Caso d'uso: Gestisci Promozioni
Pre-condizioni	- il Titolare sta utilizzando il sistema
Post-condizioni	- è stata creata un'istanza promo di una delle specializzazioni di Promozione sulla base del parametro tipologia. - Gli attributi di promo sono stati inizializzati.

2.11. Caso d'uso UC10, Modello di dominio

Non sono richieste modifiche a livello di modello di dominio.

2.12. Caso d'uso UC10, Diagramma di sequenza di sistema



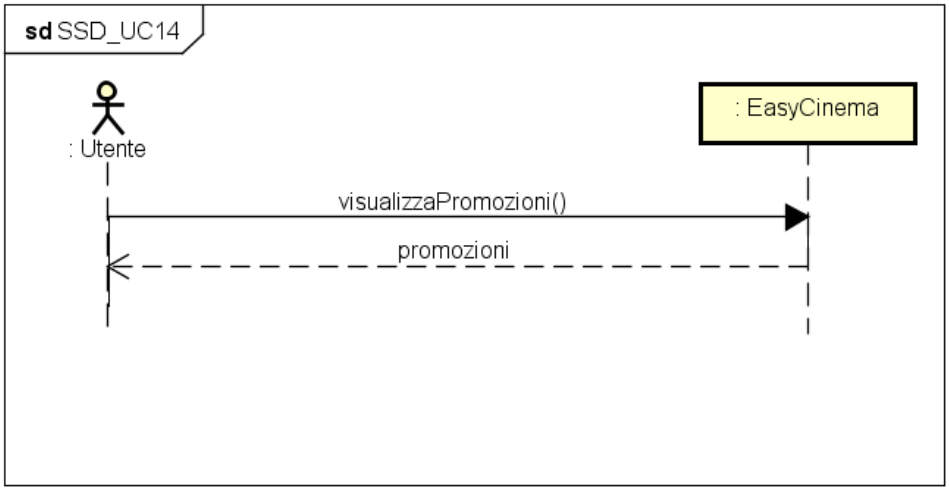
2.13. Caso d'uso UC10, Contratti delle operazioni

L'operazione di sistema, visualizzaProfilo, è una semplice interrogazione; non viene riportato il contratto.

2.14. Caso d'uso UC14, Modello di dominio

Il concetto di promozione è stato affrontato già nel caso d'uso UC9; il modello di domino rimane inalterato.

2.15. Caso d'uso UC14, Diagramma di sequenza di sistema



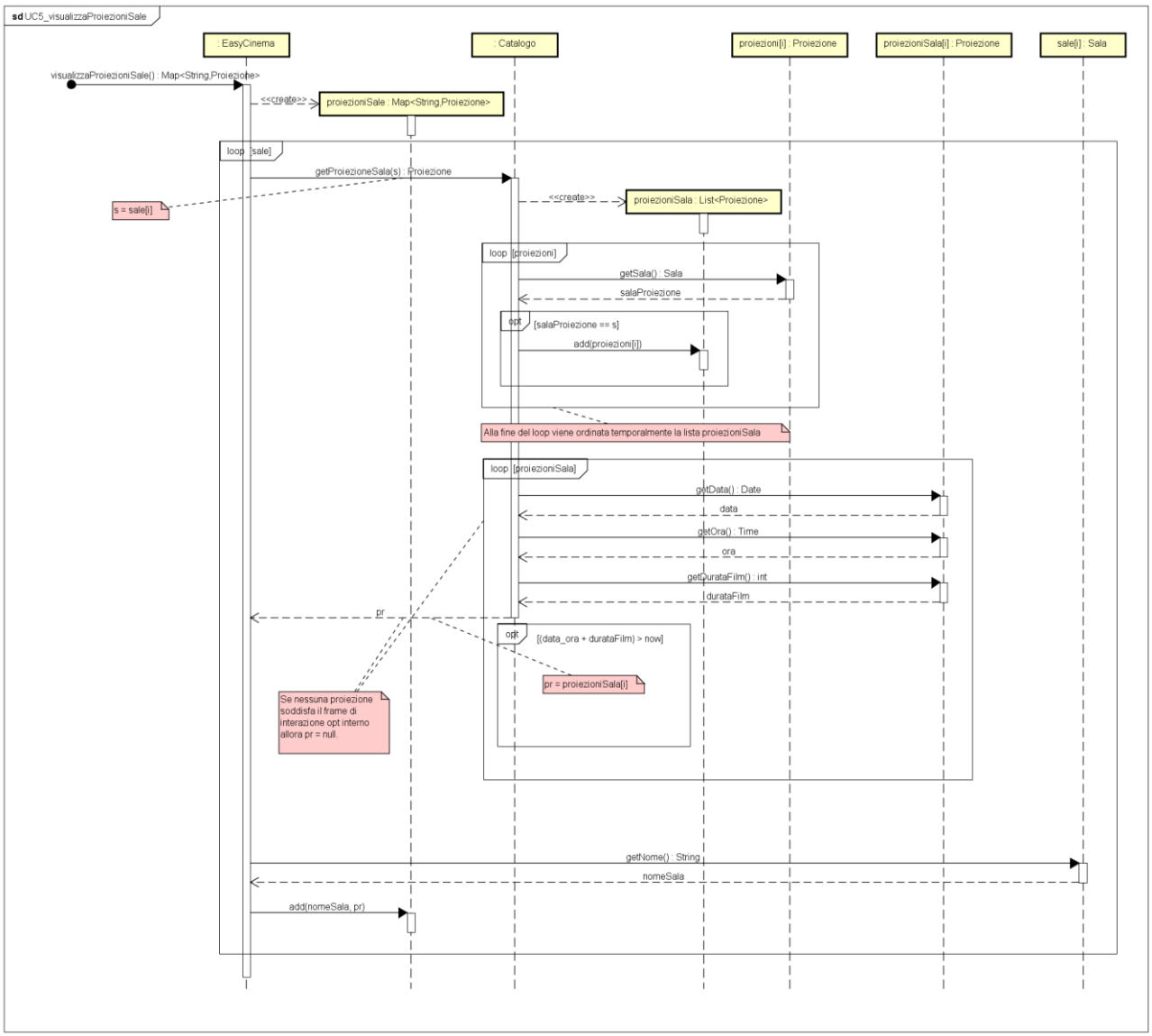
2.16. Caso d'uso UC14, Contratti delle operazioni

Non viene riportato il contratto dell'operazione di sistema poiché si tratta di un'interrogazione.

3. Progettazione

3.1. Caso d'uso UC5, Diagrammi di interazione

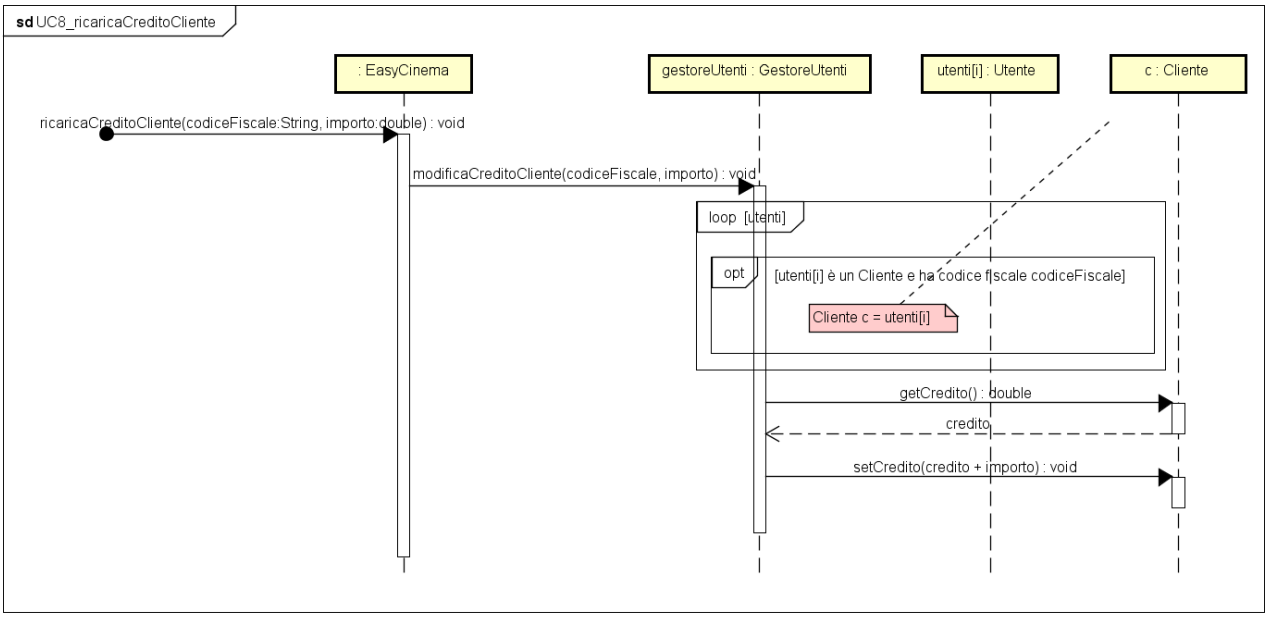
3.1.1. visualizzaProiezioniSale()



EasyCinema ha conoscenza delle sale presenti e dunque per ciascuna sala richiede a Catalogo la proiezione in corso o la prossima in programma in quanto esso contiene le proiezioni (pattern GRASP *Information Expert*).

3.2. Caso d'uso UC8, Diagrammi di interazione

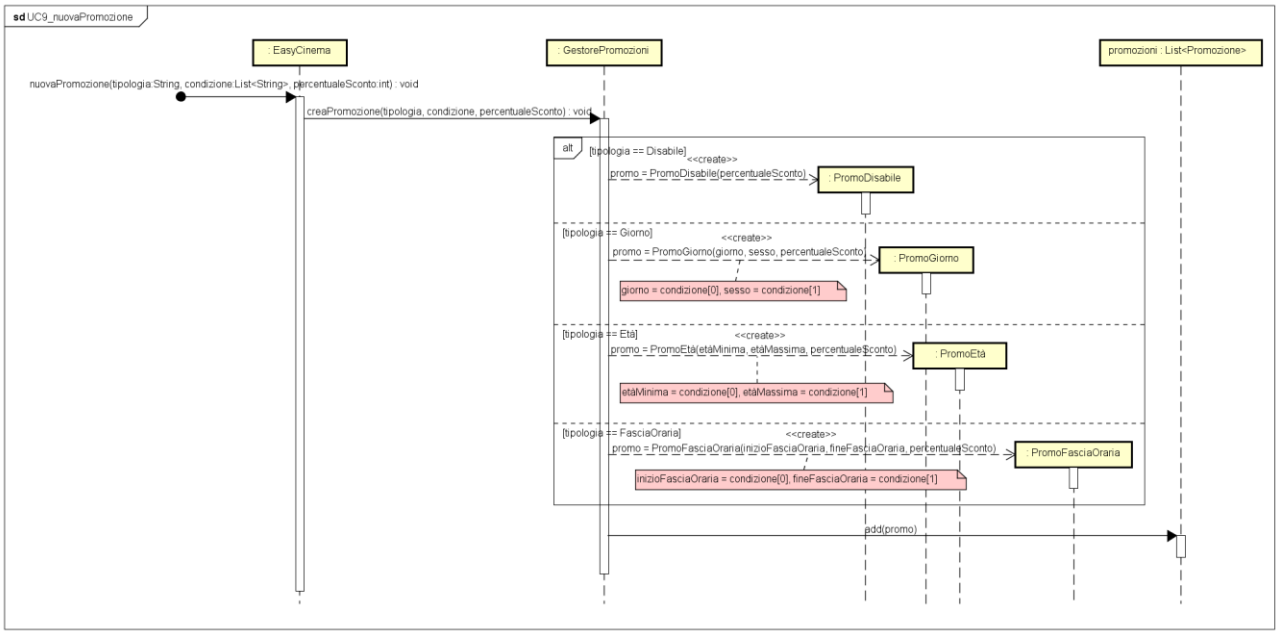
3.2.1. ricaricaCreditoCliente(codiceFiscale:String, importo:double)



La responsabilità di tale operazione viene affidata per *Information Expert* a GestoreUtenti in quanto possiede le informazioni per soddisfarla (conosce tutti i clienti poiché li gestisce).

3.3. Caso d'uso UC9, Diagrammi di interazione

3.3.1. nuovaPromozione(tipologia:String, condizione:List<String>, percentualeSconto:int)



In questa quarta iterazione viene attenzionato il concetto di Promozione. Analizzando la specifica dei requisiti emergono quattro tipologie di promozione che sono state considerate a livello di modello di dominio e che ora vengono trasformate in classi software.

La creazione di una particolare tipologia di promozione è stata affidata alla classe software GestorePromozioni ottenuta per *Pure Fabrication* (Pattern GRASP). A tale classe è stata affidata la gestione delle promozioni così da ottenere un insieme altamente coeso di funzionalità evitando di sovraccaricare le altre classi già presenti in modo da non avere un maggiore accoppiamento e una minore coesione.

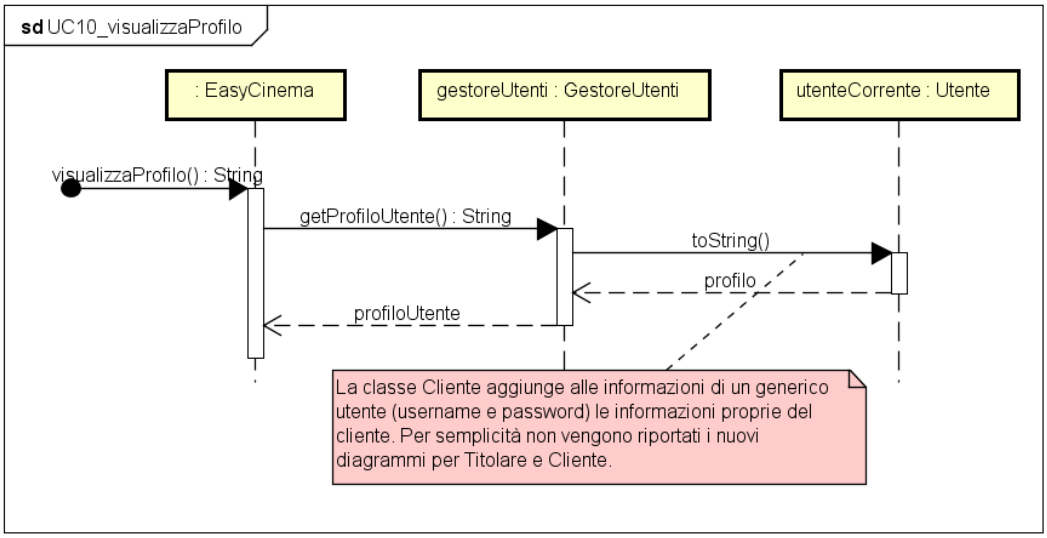
L'introduzione del concetto di Promozione richiede la modifica di alcune operazioni ma anche attributi. Più precisamente, occorre modificare le operazioni di sistema: **nuovaPrenotazione** (in quanto le promozioni vengono applicate alla prenotazione appena creata), **calcolaTotalePrenotazione** (poiché il totale della prenotazione deve tenere conto della scontistica ottenuta dalle promozioni).

In particolare, per Polymorphism il diagramma di sequenza relativo a nuovaPrenotazione non viene ulteriormente dettagliato in quanto controlloPromoApplicabile è un metodo astratto. Vengono creati tanti diagrammi quante sono le promozioni. Analogamente si è proceduto per l'operazione astratta calcolaSconto creando un diagramma di sequenza per la classe astratta Promo e uno per la classe PromoComposta.

L'introduzione delle promozioni sul giorno (con filtro sul sesso) e dell'età comportano l'aggiunta di tali attributi alla classe Cliente. Ciò comporta in primis la variazione del costruttore di tale classe, come conseguenza di ciò occorre modificare anche GestoreUtenti e in particolare il metodo nuovoCliente; questo determina la modifica dell'operazione di sistema **nuovoCliente** a livello di interfaccia IEasyCinema, ProxyEasyCinema e EasyCinema. Occorre modificare anche il diagramma di sequenza di sistema per UC7.

3.4. Caso d'uso UC10, Diagrammi di interazione

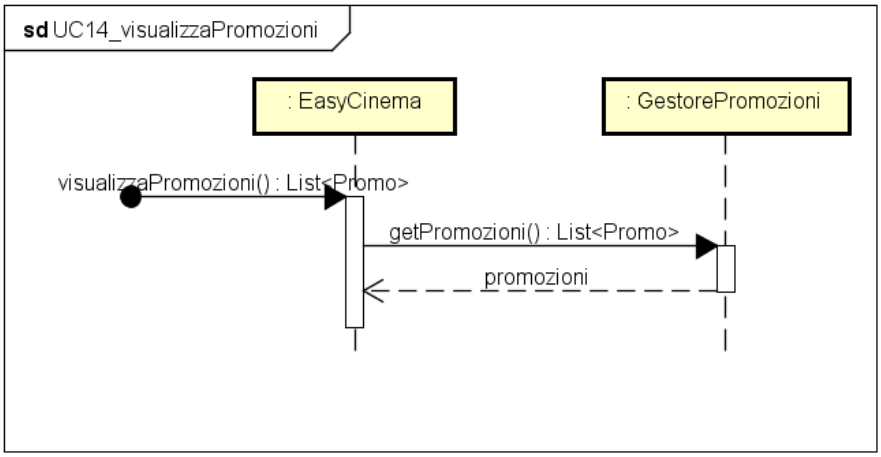
3.4.1. visualizzaProfilo()



La responsabilità di fornire le informazioni sull'utente corrente, che ne costituiscono il profilo, è affidata alla classe GestoreUtenti in quanto gestisce i vari utenti (Pattern GRASP *Information Expert*).

3.5. Caso d'uso UC14, Diagrammi di interazione

3.5.1. visualizzaPromozioni()



3.6. Design Pattern GoF applicati

3.6.1. Composite

Il design pattern GoF Composite consente ai clienti di trattare in modo uniforme oggetti singoli e composizioni di oggetti. Ciò rende le classi clienti più facili da implementare, modificare, testare e riutilizzare.

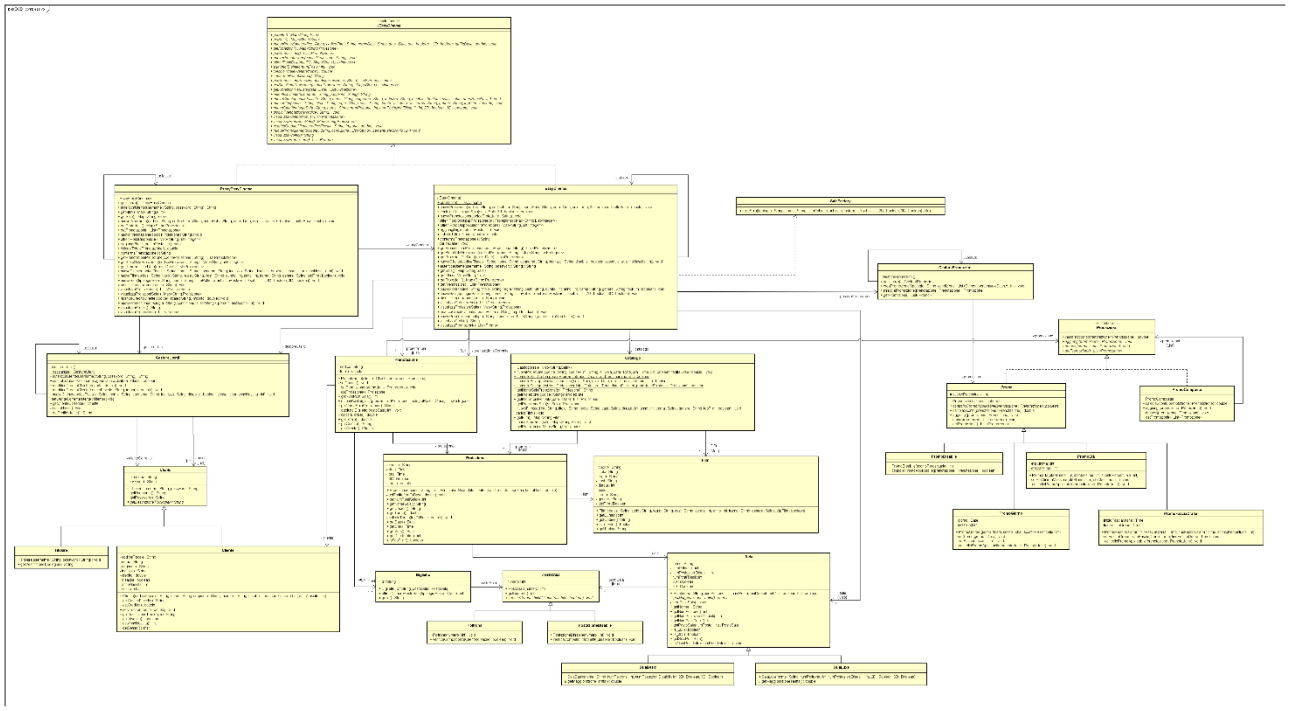
Il pattern Composite viene usato quando i clienti ignorano la differenza tra la composizione di oggetti e i singoli oggetti. Il pattern è stato applicato nel seguente modo: la classe Prenotazione ha il ruolo di Client, l'interfaccia Promozione quello Component; Promozione viene implementata da Promo, classe astratta da cui ereditano le singole promozioni, le quali hanno il ruolo di Leaf e infine si ha PromoComposta nel ruolo di Composite che eredita da Promozione.

Esistono due "varianti" del pattern Composite: una che mira alla *uniformità* in quanto il client non distingue oggetti atomici e composti e prevede che Component definisca anche le operazioni rivolte alla composizione (add, remove, getChildren), l'altra invece punta alla *sicurezza* in quanto i clients non possono richiedere operazioni di composizione agli oggetti Leaf (non avrebbe senso) poiché Component non presenta tali operazioni ma gli oggetti Leaf e Composite vengono trattati diversamente.

Si è scelto di utilizzare la variante rivolta all'uniformità realizzando in aggiunta una classe astratta Promo che implementa i metodi rivolti alla composizione e da cui ereditano tutte le singole promozioni.

A ciascuna Prenotazione, se essa non fa riferimento ad un top Film, viene associato un oggetto Promozione che può essere un oggetto Leaf: PromoDisabile, PromoEtà, PromoGiorno, PromoFasciaOraria oppure un oggetto Composite: PromoComposta ottenuto tramite verifica dell'applicabilità su ciascuna promozione esistente.

3.7. Diagramma delle classi complessivo



4. Implementazione

Sono stati implementati gli scenari principali di successo per i casi d'uso presi in considerazione durante questa quarta iterazione.

Inoltre:

- per il caso d'uso UC5 è stato implementato anche lo scenario d'estensione 2a,
- per UC8 tutte le estensioni,
- per UC9 le estensioni 3a, 4a.

5. Testing

I test implementati nelle precedenti iterazioni sono stati raffinati per supportare le modifiche apportate; ad esempio, i test che richiedono la creazione di istanze della classe Cliente sono stati modificati per considerare gli attributi sesso e annoNascita.

Nella quarta iterazione sono stati affrontati gli scenari principali di successo di cinque casi d'uso, tre dei quali sono interrogazioni ovvero: UC5, UC10 e UC14. Gli sforzi vengono concentrati sui restanti casi d'uso: UC8 (Ricarica Credito Cliente) e UC9 (Gestisci Promozioni).

Progettazione dei casi di test:

- GestoreUtenti
    - **modificaCreditoCliente**(String codiceFiscale, double importo): una ricarica può essere effettuata esclusivamente sul conto di un cliente esistente. L'importo della ricarica può essere un valore positivo, negativo o nullo.
      Le condizioni di interesse individuate sono:

        1. *testClienteInesistente*
            - Input: codice fiscale associato a nessun cliente registrato al sistema.
            - Output previsto: generazione eccezione.

        2. *testImportoNullo*
            - Input: codice fiscale associato ad un cliente, importo pari a 0.
            - Output previsto: credito del cliente invariato.

        3. *testImportoNegativo*
            - Input: codice fiscale associato ad un cliente, importo pari a -10.
            - Output previsto: credito del cliente diminuisce di un valore pari a 10.

        4. *testImportoPositivo*
            - Input: codice fiscale associato ad un cliente, importo pari a 10.
            - Output previsto: credito del cliente aumenta di un valore pari a 10.

- GestorePromozioni
    - **creaPromozione**(String tipologia, List<String> condizione, int percentualeSconto): una promozione può essere solo di tipo Disabile, Età, Giorno e FasciaOraria. Ciascuna tipologia ha dei requisiti sulle condizioni ben precisi. Lo sconto espresso come percentuale deve essere compreso tra 1 e 100.
      Le condizioni di interesse individuate sono:

        1. *testPercentualeScontoInferiore1*
            - Input: sconto percentuale inferiore a 1 (0).
            - Output previsto: generazione eccezione e promozione non creata.

        2. *testPercentualeScontoSuperiore100*
            - Input: sconto percentuale superiore a 100 (101).
            - Output previsto: generazione eccezione e promozione non creata.

        3. *testPromoDisabileValida*
            - Input: tipologia promozione Disabile, nessuna condizione, sconto percentuale valido.
            - Output previsto: creazione nuova promozione di tipo Disabile.

        4. *testPromoGiornoGiornoNonValido*
            - Input: tipologia promozione Giorno, formato del giorno non valido, formato del filtro sul sesso valido, sconto percentuale valido.
            - Output previsto: generazione eccezione e promozione non creata.

        5. *testPromoGiornoSessoNonValido*
            - Input: tipologia promozione Giorno, formato del giorno valido, formato del filtro sul sesso non valido, sconto percentuale valido.

- Output previsto: generazione eccezione e promozione non creata.

6. *testPromoGiornoValida*
   - Input: tipologia promozione Giorno, formato del giorno valido, formato del filtro sul sesso valido, sconto percentuale valido.
   - Output previsto: creazione nuova promozione di tipo Giorno.
7. *testPromoEtàEtàMinimaNonValida*
   - Input: tipologia promozione Età, formato dell'età minima non valido, formato dell'età massima valido, sconto percentuale valido.
   - Output previsto: generazione eccezione e promozione non creata.
8. *testPromoEtàEtàMassimaNonValida*
   - Input: tipologia promozione Età, formato dell'età minima valido, formato dell'età massima non valido, sconto percentuale valido.
   - Output previsto: generazione eccezione e promozione non creata.
9. *testPromoEtàValida*
   - Input: tipologia promozione Età, formato dell'età minima valido, formato dell'età massima valido, sconto percentuale valido.
   - Output previsto: creazione nuova promozione di tipo Età.
10. *testPromoFasciaOrariaInizioNonValido*
    - Input: tipologia promozione FasciaOraria, formato inizio fascia oraria non valido, formato fine fascia oraria valido, sconto percentuale valido.
    - Output previsto: generazione eccezione e promozione non creata.
11. *testPromoFasciaOrariaFineNonValida*
    - Input: tipologia promozione FasciaOraria, formato inizio fascia oraria valido, formato fine fascia oraria non valido, sconto percentuale valido.
    - Output previsto: generazione eccezione e promozione non creata.
12. *testPromoFasciaOrariaValida*
    - Input: tipologia promozione FasciaOraria, formato inizio fascia oraria valido, formato fine fascia oraria valido, sconto percentuale valido.
    - Output previsto: creazione nuova promozione di tipo FasciaOraria.
13. *testPromoNonValida*
    - Input: tipologia promozione non valida.
    - Output previsto: generazione eccezione e promozione non creata.

- **associaPromozione**(Prenotazione prenotazione): obiettivo del test è verificare l'applicabilità di una o più promozioni ad una prenotazione.
  Le condizioni di interesse individuate sono:

1. *testPromoDisabile*
   - Input: prenotazione effettuata da un cliente disabile, istanza di PromoDisabile.
   - Output previsto: istanza di PromoComposta creata contenente l'istanza di PromoDisabile creata in precedenza.
2. *testPromoEtà*

- Input: prenotazione effettuata da un cliente la cui età rientra nei limiti dell'istanza di PromoEtà creata.
- Output previsto: istanza di PromoComposta creata contenente l'istanza di PromoEtà creata in precedenza.

3. *testPromoGiorno*
   - Input: prenotazione relativa ad una proiezione che ha luogo lo stesso giorno di quello a cui fa riferimento un'istanza di PromoGiorno e il cliente che effettua tale prenotazione ha lo stesso sesso di quello richiesto dall'istanza di PromoGiorno.
   - Output previsto: istanza di PromoComposta creata contenente l'istanza di PromoGiorno creata in precedenza.
4. *testPromoFasciaOraria*
   - Input: prenotazione relativa ad una proiezione il cui orario di inizio ricade tra l'inizio e la fine della fascia oraria a cui fa riferimento un'istanza di PromoFasciaOraria.
   - Output previsto: istanza di PromoComposta creata contenente l'istanza di PromoFasciaOraria creata in precedenza.
5. *testPromoComposta*
   - Input: istanza di PromoDisabile, PromoEtà, PromoGiorno, PromoFasciaOraria, prenotazione che soddisfa le condizioni di tutte queste promozioni.
   - Output previsto: istanza di PromoComposta creata contenente tutte le istanze delle promozioni create in precedenza.

Si è raggiunta una percentuale di code coverage del 50.4%.