Corso di Laurea Magistrale in Ingegneria Informatica – A.A. 2021-2022

Progetto Ingegneria del Software

Studenti:

Giuseppe Testa

Luigi Fontana

Walter Trupia

Premessa

Il documento rappresenta una descrizione relativa allo sviluppo del progetto MusicShowFactory.
Il software è stato implementato nel linguaggio Java sfruttando l'ambiente IntelliJ ed il tool maven mentre la progettazione in UML è stata realizzata usando il software Astah.
I dati sono mantenuti e gestiti tramite file.
In questo documento sono presentate solo le versioni finali ottenute al termine delle fasi di progettazione. Le versioni intermedie sono nelle relative sottocartelle.

Sommario

1 Ideazione

1.1 Introduzione

Prevediamo la realizzazione di una applicazione chiamata "MusicShowFactory" in grado di supportare la variabilità delle regole di dominio. L'obiettivo dell'applicazione è quello di gestire la programmazione dei concerti, gestire gli artisti e la vendita di biglietti ai clienti e la gestione del merchandise dedicato all'artista.

1.1.1 Posizionamento

Opportunità di business

Il software MusicShowFactory si pone l'obiettivo di digitalizzare ed automatizzare il processo di creazione e gestione di un concerto di un artista in tutte le sue caratteristiche. In questo modo, si punto a centralizzare la programmazione del concerto e l'acquisto di biglietti e merchandise.

Formulazione del problema

La presenza di un sistema non automatizzato e non centralizzato aumenta la quantità di lavoro necessario all'organizzazione dell'evento.

Non avere un sistema che racchiude questa operazione, costringe ad avere applicazioni diverse per svolgere i diversi compiti.

Formulazione della posizione del prodotto

Il software è rivolto alle case discografiche che vogliono gestire gli aspetti dei concerti e la vendita di biglietti e merchandise.

1.1.2 Descrizione delle parti interessate

Obiettivo a livelli dell'utente

Gli utenti necessitano di un sistema che soddisfi i seguenti obiettivi:

- Concert planner: gestire la programmazione dei concerti e la gestione degli artisti;
- Cashier: gestire la vendita di biglietti e del merchandise ai clienti;
- Market manager: gestisce gli aspetti relativi al merchandise ed alle promozioni da associare agli eventi.

1.2 Requisiti

MusicShowFactory è un sistema software per la gestione dei concerti. Il sistema deve occuparsi della gestione e della programmazione dei concerti, con annessa gestione dei biglietti e degli artisti.

Il sistema software deve occuparsi dei vari aspetti della gestione dei concerti:

- Il concert planner deve poter gestire gli artisti all'interno del sistema;
- Il concert planner deve gestire l'inserimento nel sistema di un nuovo concerto e della sua programmazione;
- Il concert planner deve gestire l'aggiornamento della programmazione di un determinato concerto, con modifica, aggiunta o rimozione;
- Il concert planner ha la possibilità di vedere nel sistema la lista di tutti i concerti in programma;
- Il market manager deve gestire l'inserimento nel sistema di una promozione da associare ad un evento;

- Il Cashier gestisce l'acquisto di un biglietto da parte di un cliente. Il prezzo non è statico ma dipende dalle scelte del cliente e delle varie promozioni;
- Il Cashier gestisce la politica di annullamento e rimborso di un biglietto per un cliente che decide di rinunciare in tempo o nel caso in cui si verifichi l'annullamento dell'evento;
- Il Cashier gestisce l'acquisto del merchandise da parte di un cliente. Il prezzo non è statico ma dipende dalle scelte del cliente e delle varie promozioni;
- Il market manager gestisce il merchandise all'interno di un magazzino;
- Il Cashier gestisce la politica di annullamento e rimborso di un prodotto del merchandise per un cliente che decide in tempo di restituire il prodotto;

1.3 Obiettivi e casi d'uso

Analizzando i requisiti riportati nel paragrafo precedente, sono stati individuati gli attori principalia cui è destinato il sistema e gli obiettivi che si intende portare a termine; da queste informazioni sono stati ricavati i casi d'uso principali.

ATTORE	OBIETTIVO	CASO D'USO
Concert Planner	Inserire un nuovo concerto per un artista con le relative informazioni e definendo gli eventi associati a questo.	UC1: Inserisci Concerto
Cashier	Gestire l'acquisto di un biglietto per un determinato concerto da parte di un cliente.	UC2: Gestisci acquisto biglietto
Cashier	Gestire il rimborso di un biglietto.	UC3: Gestisci rimborso biglietto
Concert Planner	Inserire, rimuovere, modificare e cercare gli artisti presenti nel sistema.	UC4: Gestisci Artisti (CRUD)
Market Manager	Inserire un nuovo oggetto di merchandise con le relative informazioni.	UC5: Inserisci Merchandise
Cashier	Gestire l'acquisto di un prodotto relativo al merchandise di un artista ad un cliente.	UC6: Gestisci acquisto merchandise
Cashier	Gestire il rimborso di un articolo del merchandise.	UC7: Gestisci rimborso merchandise
Market Manager	Rimuovere o modificare un prodotto all'interno del magazzino del merchandise.	UC8: Gestisci merchandise (CRUD)

Concert Planner	Modificare le informazioni relative ad un concerto.	UC9: Gestisci informazioni concerto (CRUD)
Concert Planner	Modificare le informazioni relative ad un evento	UC10: Gestisci informazioni evento (CRUD)
Market Manager	Creare, modificare, rimuovere la promozione relativa ad un evento.	UC11: Gestisci promozione (CRUD)
Concert Planner	Visualizzare le informazioni per ogni concerto.	UC12: Visualizza programmazione eventi di un concerto.
Concert Planner	Visualizzare la lista dei biglietti venduti e disponibili.	UC13: Visualizza biglietti relativi ad un evento.
Market Manager	Visualizzare la lista dei prodotti relativa ad ogni artista	UC14: Visualizza merchandise relativo ad un artista

1.4 Modello dei casi d'uso

Viene presentata in questa fase una descrizione dei casi d'uso. I casi d'uso trattati nel dettaglio sono descritti in formato dettagliato (solo il caso UC7 è descritto in formato dettagliato ma non trattato in nessuna iterazione). Gli altri casi d'uso sono descritti in formato breve o informale.

UC1: Inserisci Concerto

Nome del caso d'uso	UC1: Inserisci concerto
Portata	Applicazione MusicShowFactory
Livello	Obiettivo utente
Attore primario	Concert planner
Parti interessate e interessi	• Il concert planner vuole inserire nel sistema un nuovo concerto con le relative informazioni.
Pre-condizioni	L'artista deve essere presente nel sistema. Devono essere presenti le autorizzazioni necessarie per organizzare l'evento.
Garanzia di successo	Le informazioni relative al concerto sono registrate con successo nel sistema.
Scenario principale di successo	1. Il Concert planner vuole inserire un nuovo concerto di un determinato artista con relativa location e data; 2. Il Concert planner sceglie l'attività "Inserimento nuovo concerto". 3. Il Concert planner inserisce nel sistema il nome di un artista. Il Sistema restituisce il codice univoco che lo identifica. 4. Il Concert planner inserisce il nome del concerto, il prezzo base ed il codice univoco del concerto. Il sistema registra le informazioni relative al concerto.

	5. Il Concert planner per il concerto appena inserito definisce un evento specificandone nome, data, orario, location, numeroBiglietti(Normali e Backstage) e un codice univoco. *Il passo 5 viene ripetuto finché serve.* 6. Il Concert planner indica di aver finito.
Estensioni	*a. In qualsiasi momento il Sistema fallisce e si arresta improvvisamente. 1. L'amministratore riavvia il software e ripristina lo stato precedente del sistema; 2. Il sistema ripristina lo stato. 3a. Il concert planner inserisce il nome di un Artista non presente nel sistema. 1. Il sistema genera un messaggio di errore; 2. Il concert planner ripete il passaggio 3 inserendo un nome diverso. 4a. Il concert planner inserisce un codice univoco non valido in quanto già presente nel sistema. 1. Il sistema genera un messaggio di errore; 2. Il concert planner ripete il passaggio 4 inserendo un codice diverso. 5a. Il concert planner inserisce un evento con una data, un'ora ed una location in cui è già previsto un altro evento. 1. Il sistema genera un messaggio di errore; 2. Il concert planner ripete il passaggio 5 cambiando data/ora/location del concerto. 5b. Il concert planner inserisce un codice univoco non valido per l'evento in quanto già presente nel sistema. 1. Il sistema genera un messaggio di errore. 2. Il concert planner ripete il passaggio 5 inserendo un codice diverso.
Requisiti speciali	
Elenco delle varianti tecnologiche e dei dati	Legata alla volontà di un artista di fare concerti.
Frequenza di ripetizioni	
Varie	

UC2: Gestisci Acquisto Biglietto

Nome del caso d'uso	UC2: Gestisci acquisto biglietto
Portata	Applicazione MusicShowFactory
Livello	Obiettivo utente
Attore primario	Cashier
Parti interessate e interessi	• Cashier: vuole assicurare una corretta registrazione della vendita del biglietto e che questo venga generato dal sistema con successo. • Cliente: vuole acquistare un biglietto relativo ad un concerto
Pre-condizioni	Il cliente conosce l'artista di cui vuole acquistare i biglietti del concerto.

Garanzia di successo	Il sistema genera il biglietto che viene venduto e consegnato al cliente. Il sistema registra tutte le informazioni del biglietto venduto.
Scenario principale di successo	1. Il Cliente si reca in un centro autorizzato per l'acquisto di un biglietto relativo ad un concerto dell'artista che preferisce. 2. Il Cliente indica al Cashier l'artista per il quale vuole acquistare il biglietto. 3. Il Cashier sceglie l'attività "Acquisto biglietto" e inserisce nel sistema il nome dell'artista indicato. 4. Il Sistema mostra i concerti disponibili relativi all'artista. 5. Il Cliente indica il nome del concerto. Il cashier inserisce l'id del concerto e per il concerto selezionato il sistema mostra la lista degli eventi. 6. Il cliente comunica l'evento di cui vuole acquistare il biglietto. 7. Il Cashier inserisce il codice identificativo dell'evento nel Sistema. 8. Il Sistema mostra i biglietti disponibili per l'evento selezionato. 9. Il Cliente comunica che desidera acquistare un biglietto dell'evento che desidera. 10. Il Cashier inserisce nome, cognome, codice fiscale ed età del cliente e registra i dati nel sistema. 11. Il Sistema calcola il prezzo del biglietto sulla base delle promozioni e degli sconti e mostra a video il risultato. 12. Il cliente decide di proseguire con l'acquisto del biglietto. 13. Il Sistema genera un codice univoco per il biglietto e lo stampa. 14. Il Cliente paga e se ne va. 15. Il sistema si aggiorna diminuendo il numero di biglietti disponibili per quell'evento.
Estensioni	*a. In un qualsiasi momento il Sistema fallisce e si arresta improvvisamente. 1. L'amministratore riavvia il software e ripristina lo stato precedente del sistema; 2. Il sistema ripristina lo stato. 2a. Il cliente indica al Cashier un artista non presente nel Sistema. 1. Il Cashier comunica al cliente che l'artista non è presente; 2. Il cliente comunica un nuovo titolo. 7a. Il Cashier inserisce un codice errato. 1. Il Sistema restituisce un messaggio di errore. 2. Il Cashier ripete il passaggio 7 inserendo un nuovo

	codice. 8a. Il sistema mostra a schermo che non ci sono biglietti disponibili per quell'evento. 1. Il Cashier comunica al cliente che non ci sono biglietti disponibili. 2. Il cliente ritorna al passo 2 oppure va via. 10a. Il Cashier inserisce nome e cognome incompatibili con il codice fiscale. 1. Il sistema restituisce un messaggio di errore. 2. Il Cashier ripete il passo 10.
Requisiti speciali	
Elenco delle varianti tecnologiche e dei dati	
Frequenza di ripetizioni	Legata a quanti clienti sono interessati.
Varie	

UC3: Gestisci Rimborso Biglietto

Nome del caso d'uso	UC3: Gestisci rimborso biglietto
Portata	Applicazione MusicShowFactory
Livello	Obiettivo utente
Attore primario	Cashier
Parti interessate e interessi	• Cashier: vuole assicurare un corretto rimborso della vendita di un biglietto e che questo venga registrato dal sistema con successo. • Cliente: vuole un rimborso di un biglietto relativo ad un concerto.
Pre-condizioni	Il cliente deve essere in possesso di un biglietto valido per un concerto non ancora avvenuto.
Garanzia di successo	Il cassiere consegna la quota del rimborso al cliente.
Scenario principale di successo	1. Il cliente si reca in un centro autorizzato per chiedere il rimborso di un biglietto; 2. Il Cashier sceglie l'attività "Rimborso biglietto"; 3. Il Cashier inserisce il codice del biglietto e ne verifica l'esistenza; 4. Il sistema verifica la validità del biglietto e calcola la cifra del rimborso; 5. Il Cashier comunica la cifra al Cliente; 6. Il Cashier consegna la quota del rimborso al cliente e questo va via; 7. Il sistema aggiorna il numero di biglietti per quell'evento.
Estensioni	*a. In un qualsiasi momento il Sistema fallisce e si arresta improvvisamente. 1. L'amministratore riavvia il software e ripristina lo stato precedente del sistema; 2. Il sistema ripristina lo stato. 3a. Il cashier inserisce un codice errato. 1. Il sistema restituisce un messaggio di errore;

	2. Il Cashier ripete il passaggio 3 inserendo un altro codice. 4a. Il sistema mostra a schermo che il biglietto è non valido. 1. Il Cashier comunica al cliente il messaggio di errore; 2. Il Cliente va via. 5a. Il cliente rifiuta il rimborso. 1. Il Cashier annulla l'operazione di rimborso; 2. Il Cliente va via.
Requisiti speciali	
Elenco delle varianti tecnologiche e dei dati	
Frequenza di ripetizioni	
Varie	

UC4: Gestisci Artisti (CRUD)

1. Il Concert planner chiede al sistema l'inserimento di un nuovo artista;
2. Il sistema chiede al Concert Planner le informazioni per l'inserimento di un nuovo artista;
3. Il Concert planner inserisce NomeArtista, GenereMusicale, codiceArtista.

Scenari alternativi

1a. Il Concerti planner chiede al sistema di modificare un artista.

1. Il sistema richiede al Concert planner i dati da aggiornare;
2. Il Concert planner aggiorna i campi di interesse.

1b. Il Concert planner chiede al sistema di cercare un artista.

1. Il sistema chiede al Concert planner di inserire le informazioni che vuole cercare;
2. Il Concert planner inserisce le informazioni;
3. Il sistema mostra gli artisti che soddisfano le informazioni desiderate.

1c. Il Concert planner chiede al sistema di eliminare un artista.

1. Il sistema chiede al Concert planner l'artista da eliminare;
2. Il Concert planner inserisce l'ID dell'Artista;
3. Il Sistema elimina l'artista ed aggiorna gli artisti presenti nel sistema.

UC5: Inserisci Merchandise

Nome del caso d'uso	UC5: Inserisci Merchandise
Portata	Applicazione MusicShowFactory
Livello	Obiettivo utente
Attore primario	Market Manager
Parti interessate e interessi	• Il Market Manager vuole inserire nel sistema un nuovo oggetto di merchandise con le relative informazioni.
Pre-condizioni	L'artista deve essere presente nel sistema.
Garanzia di successo	Le informazioni relative all'oggetto sono inserite correttamente nel sistema.
Scenario principale di successo	1. Il Market Manager vuole inserire un nuovo oggetto di merchandise di un determinato artista; 2. Il Market Manager sceglie l'attività "Inserimento nuovo merchandise".

	3. Il Market Manager inserisce il nome dell'artista. Il sistema restituisce il codice univoco che lo identifica. 4. Il Market Manager inserisce nel sistema il prezzo base, provenienza, codice univoco, iva. *Il passo 4 viene ripetuto finché serve.* 5. Il Market Manager indica di aver finito.
Estensioni	*a. In qualsiasi momento il Sistema fallisce e si arresta improvvisamente. 3. L'amministratore riavvia il software e ripristina lo stato precedente del sistema; 4. Il sistema ripristina lo stato. 3a. Il Market Manager inserisce il nome di un Artista non presente nel sistema. 3. Il sistema genera un messaggio di errore; 4. Il Market Manager ripete il passaggio 3 inserendo un nome diverso. 4a. Il Market Manager inserisce un codice univoco non valido per il merchandise in quanto già presente nel sistema. 3. Il sistema genera un messaggio di errore. 4. Il concert planner ripete il passaggio 4 inserendo un codice diverso.
Requisiti speciali	
Elenco delle varianti tecnologiche e dei dati	
Frequenza di ripetizioni	
Varie	

UC6: Gestisci Acquisto Merchandise

Nome del caso d'uso	UC6: Gestisci Acquisto Merchandise
Portata	Applicazione MusicShowFactory
Livello	Obiettivo utente
Attore primario	Cashier
Parti interessate e interessi	• Cashier: vuole assicurare un corretta vendita di un oggetto di merchandise e che questa vendita venga registrata dal sistema con successo. • Cliente: vuole un oggetto di merchandise.
Pre-condizioni	-
Garanzia di successo	Il cassiere consegna l'oggetto di merchandise al cliente.
Scenario principale di successo	8. Il cliente si reca in un centro autorizzato per l'acquisto del merchandise dell'artista che preferisce; 9. Il cliente indica al cashier l'artista per il quale vuole acquistare; 10. Il Cashier sceglie l'attività "Acquisto Merchandise"; 11. Il Cashier inserisce il nome dell'artista. Il sistema ritorna il codice dell'artista; 12. Il Cashier inserisce il codice ed il sistema mostra gli oggetti disponibili relativi all'artista selezionato;

	13. Il cliente comunica al cashier l'oggetto da acquistare; 14. Il Cashier comunica il costo al cliente al Cliente; 15. Il cliente decide di continuare l'acquisto dell'oggetto; 16. Il cliente paga e se ne va. 17. Il sistema aggiorna gli oggetti disponibili.
Estensioni	*a. In un qualsiasi momento il Sistema fallisce e si arresta improvvisamente. 3. L'amministratore riavvia il software e ripristina lo stato precedente del sistema; 4. Il sistema ripristina lo stato. 4a. Il cliente indica un artista non presente nel sistema. 3. Il sistema restituisce un messaggio di errore; 4. Il Cashier ripete il passaggio 3 inserendo un altro nome Artista. 5a. Il sistema mostra a schermo che il merchandise/oggetto selezionato per quell'artista è sold out. 3. Il Cashier comunica al cliente il messaggio di errore; 4. Il Cliente va via. 8a. Il cliente avendo saputo il prezzo dell'oggetto decide di non comprare e andare via. 3. Il Cashier annulla l'operazione di acquisto merchandise; 4. Il Cliente va via.
Requisiti speciali	
Elenco delle varianti tecnologiche e dei dati	
Frequenza di ripetizioni	
Varie	

UC7: Gestisci Rimborso Merchandise

Nome del caso d'uso	UC7: Gestisci Rimborso Merchandise
Portata	Applicazione MusicShowFactory
Livello	Obiettivo utente
Attore primario	Cashier
Parti interessate e interessi	• Cashier: vuole assicurare un corretto reso di un oggetto di merchandise e che venga registrata dal sistema con successo. • Cliente: vuole un rimborso relativo ad un oggetto di merchandise.
Pre-condizioni	-
Garanzia di successo	Il cliente consegna l'oggetto di merchandise al cashier. Il cashier consegna la quota del reso al cliente.
Scenario principale di successo	1. Il cliente si reca in un centro autorizzato per richiedere un rimborso su un oggetto di merchandise; 2. Il Cashier sceglie l'attività "Rimborso Merchandise"; 3. Il Cashier inserisce il codice merchandise e ne verifica l'esistenza; 4. Il sistema calcola la cifra del rimborso;

	5. Il Cashier comunica la cifra del rimborso al cliente; 6. Il Cashier consegna la quota del rimborso al cliente e questo va via; 7. Il sistema aggiorna gli oggetti disponibili.
	*a. In un qualsiasi momento il Sistema fallisce e si arresta improvvisamente. 1. L'amministratore riavvia il software e ripristina lo stato precedente del sistema; 2. Il sistema ripristina lo stato. 4a. Il cliente indica un artista non presente nel sistema. 5. Il sistema restituisce un messaggio di errore; 6. Il Cashier indica al cliente che il nome artista è non corretto.
Requisiti speciali	
Elenco delle varianti tecnologiche e dei dati	
Frequenza di ripetizioni	
Varie	

UC8: Gestisci Merchandise (CRUD)

Il market manager chiede al sistema di modificare un prodotto del merchandise;

1. Il sistema chiede al market manager di inserire l'id del prodotto che vuole modificare.
2. Il market manager aggiorna i dati del prodotto selezionato.
3. Il sistema aggiorna i dati inseriti.

Scenari alternativi

1a. Il market manager chiede al sistema di cercare un prodotto all'interno del magazzino del merchandise;

1. Il sistema richiede al market manager le informazioni del prodotto.
2. Il market manager inserisce le informazioni che vuole cercare.
3. Il sistema mostra i prodotti del merchandise che soddisfano le informazioni desiderate.

1b. Il market manager chiede al sistema di eliminare un prodotto dal magazzino del merchandise;

1. Il sistema chiede al market manager il prodotto da eliminare.
2. Il market manager inserisce il codice identificativo del prodotto.
3. Il sistema elimina il prodotto ed aggiorna il magazzino del merchandise.

UC9: Gestisci Informazioni Concerto (CRUD)

1. Il Concert planner chiede al sistema di modificare le informazioni di un concerto;
2. Il sistema chiede al Concert Planner le informazioni che vuole modificare;
3. Il Concert planner aggiorna i dati di interesse.

Scenari alternativi

1a. Il Concert planner chiede al sistema di eliminare un concerto.

1. Il sistema chiede al Concert planner il concerto da eliminare;
2. Il Concert planner inserisce il codice identificativo del concerto
3. Il Sistema aggiorna i concerti presenti nel sistema.

1b. Il Concert planner chiede al sistema di cercare un concerto.

1. Il sistema chiede al Concert planner di inserire le informazioni che vuole cercare;
2. Il Concert planner inserisce le informazioni;
3. Il sistema mostra il concerto che soddisfa le informazioni desiderate.

UC10: Gestisci Informazioni Evento (CRUD)

1. Il Concert Planner chiede al sistema di modificare le informazioni di un evento relativo ad un concerto;
2. Il sistema chiede al Concert Planner le informazioni che vuole modificare;
3. Il Concert Planner aggiorna i dati di interesse;

Scenari alternativi

1a. Il Concert planner chiede al sistema di eliminare un evento.

1. Il sistema chiede il codice dell'evento da eliminare;
2. Il Concert planner inserisce il codice dell'evento.
3. Il sistema aggiorna i dati relativi agli eventi per quel concerto.

1b. Il Concert planner chiede al sistema di cercare un evento.

1. Il sistema chiede al Concert planner di inserire le informazioni dell'evento
2. Il concert planner inserisce le informazioni dell'evento che vuole cercare.
3. Il sistema visualizza le informazioni dell'evento.

UC11: Gestisci Promozione (CRUD)

1. Il Concert planner chiede al sistema l'inserimento di una nuova promozione per un evento;
2. Il sistema chiede al Market manager come vuole decorare la promozione e chiede le informazioni;
3. Il Market manager inserisce le informazioni.

Scenari alternativi

1a. Il Market manager chiede al sistema di modificare una promozione.

1. Il sistema richiede al Market Manager come vuole decorare la promozione;
2. Il Market manager inserisci i dati ed aggiorna la promozione.

1b. Il Market manager chiede al sistema di mostrare la promozione.

1. Il sistema chiede al Market manager di inserire le informazioni per la promozione che vuole cercare;
2. Il Market manager inserisce le informazioni;
3. Il sistema mostra la promozione che soddisfa le informazioni desiderate.

1c. Il Market manager chiede al sistema di eliminare una promozione.

1. Il sistema chiede al market manager la promozione da eliminare;
2. Il Concert planner inserisce le informazioni alla quale si riferisce la promozione;
3. Il Sistema elimina la promozione.

UC12: Visualizza Programmazione Eventi di un Concerto

Il Concert planner vuole visualizzare le informazioni relative alla programmazione degli eventi di un concerto.

UC13: Visualizza biglietti relativi ad un concerto

Il Concert planner vuole visualizzare le informazioni relative ai biglietti venduti per un determinato evento.

UC14: Visualizza merchandise relativo ad un artista

Il Market manager vuole visualizzare le informazioni relative al merchandise per un determinato artista.

1.5 Regole di Dominio

ID	Regola	Modificabilità	Sorgente
D1	Se è scelto un biglietto BACKSTAGE il prezzo aumenta del 40% rispetto al prezzo base.	Alta	Politica interna di marketing.
D2	Se la richiesta di rimborso viene effettuata 5 giorni prima del concerto, il rimborso viene calcolato come il 50% dell'importo pagato.	Media	Politica interna di marketing.
D3	Se il cliente ha meno di 8 anni non paga alcun importo.	Alta	Politica interna di marketing.
D4	Se il cliente ha più di 8 anni ma meno di 18 paga il biglietto al 50%.	Alta	Politica interna di marketing.

1.6 Glossario

TERMINE	DEFINIZIONE
Concert planner	Colui che si occupa della programmazione dei concerti.
Market manager	Colui che si occupa della gestione del merchandise e delle promozioni.
Cashier	Addetto alle vendite nel centro autorizzato.
Merchandise	Rappresenta l'insieme dei prodotti di ogni artista presente nel magazzino.
Biglietto	Documento nominale che accerta l'acquisto e garantisce l'accesso all'evento.
Cliente	Persona che è interessata ad effettuare operazioni nel centro autorizzato.
Concerto	Insieme di eventi di un artista.
Centro Autorizzato	Luogo in cui si svolgono le relative

	operazioni di vendita, acquisto e rimborso.
Promozione	Agevolazioni di prezzo per l'acquisto di un biglietto di un evento.
Artista	Persona o gruppo di persone che producono musica e si esibiscono in concerti.
Evento	Singola esibizione di un artista che si svolge in una specifica data e location.
Biglietto Normale	Biglietto che non da nessun privilegio a chi lo compra.
Biglietto Backstage	Biglietto che garantisce dei privilegi a chi lo acquista oltre ad avere un prezzo maggiorato rispetto a quello normale.
Merchandise Gadget	Si riferisce al fatto che può essere un oggetto di merchandise di tipo gadget che può essere un cappello, sciarpa, spilla.
Merchandise Maglietta	Si riferisce fatto che può essere un oggetto di merchandise di tipo maglietta che può avere un colore, una taglia ed una provenienza differente.
Merchandise Dischi	Si riferisce al fatto che può essere un oggetto di merchandise di tipo disco e si distingue in diverse versioni: normale, limited, vinile.

2 Analisi

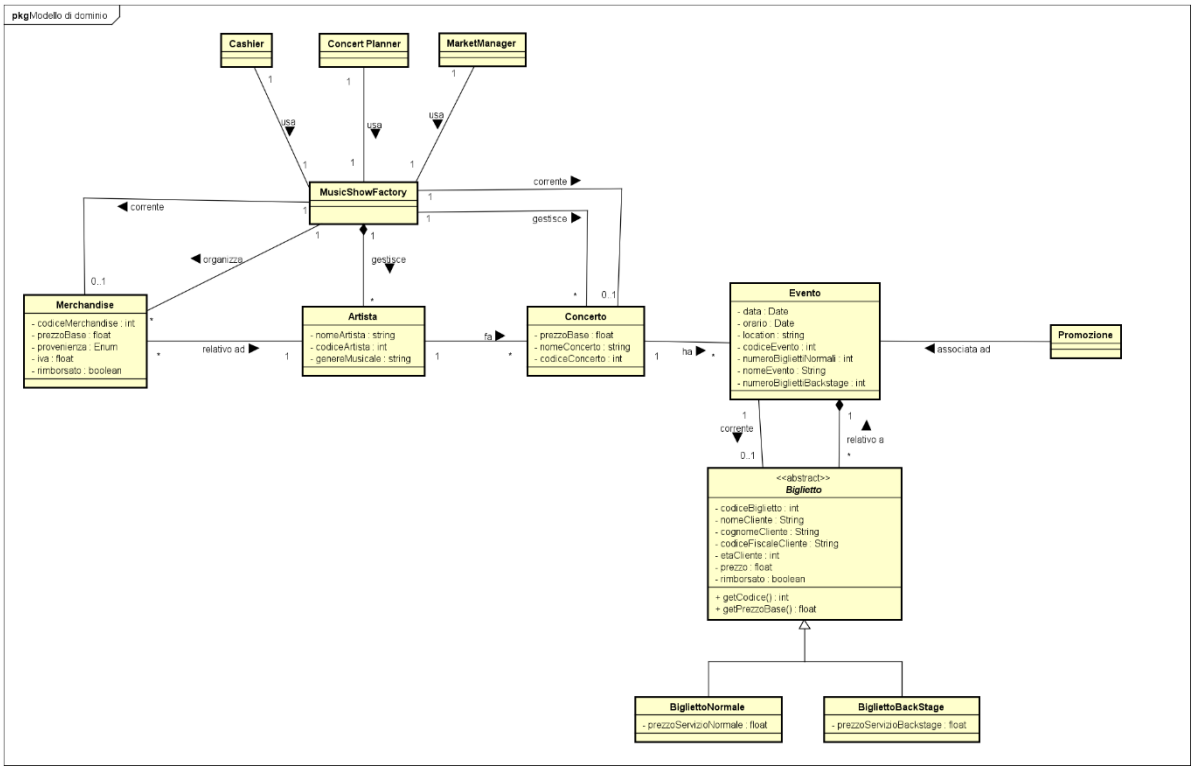
2.1 Introduzione

Seguendo l'approccio iterativo di UP, la realizzazione dell'applicazione è stata articolata su 5 iterazioni. In questo modo è stata affrontata un'analisi dei requisiti graduale in modo da limitare eventuali errori di progettazione ed implementazione. Per ciascuna iterazione sono state gestite le seguenti problematiche:

- ***Iterazione 1***
  - Implementazione dello scenario principale di successo di UC1: Inserisci Concerto;
  - Implementare un caso d'uso di start up per gestire l'inizializzazione delle iterazioni.
- ***Iterazione 2***
  - Implementazione del caso d'uso UC2: Gestisci Acquisto Biglietto. In questa fase le promozioni e le regole di dominio non solo considerate per semplicità.
- ***Iterazione 3***
  - Analisi e progettazione delle estensioni 3a, 4a, 5a, 5b per il caso d'uso UC1: Inserisci Concerto e l'estensione 7a per il caso d'uso UC2: Gestisci Acquisto Biglietto;
  - Implementazione del caso d'uso UC3: Gestisci Rimborso Biglietto e gestione della regola di dominio D2.
- ***Iterazione 4***
  - Implementazione del caso d'uso UC5: Inserisci Merchandise;
  - Implementazione del caso d'uso UC6: Gestisci Acquisto Merchandise.
- ***Iterazione 5***
  - Gestione delle promozioni e delle regole di dominio D1, D3, D4 che erano state ignorate nelle precedenti iterazioni.

2.2 Modello di Dominio

In questa fase viene mostrato il Modello di Dominio in modo da rappresentare tramite elaborato grafico il dominio informativo dell'applicazione in cui vengono identificati i concetti, gli attributi e le associazioni tra le classi concettuali. Il modello di dominio alla fine delle iterazioni è dato da:



(Per una migliore visualizzazione è disponibile il png chiamato "M1_ModellodiDominio.png")

Le classi concettuali individuate sono:

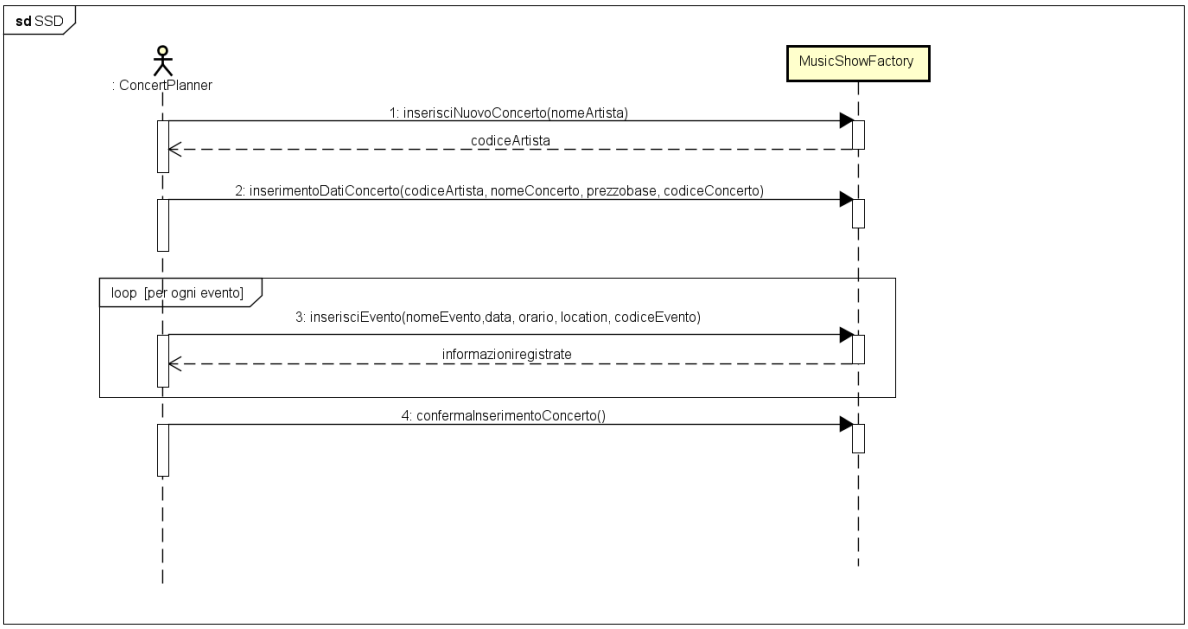
- *Concert Planner*: attore che interagisce con il sistema;
- *Cashier:* attore che interagisce con il sistema;
- *Market Manager*: attore che interagisce con il sistema.
- *MusicShowFactory:* rappresenta il sistema che viene usato per fare le operazioni;
- *Artista:* rappresenta la band o la persona della quale si vuole organizzare un concerto;
- *Concerto:* rappresenta il concerto associato ad un artista ed ha diversi eventi possibili;
- *Evento:* rappresenta l'evento di cui si potranno acquistare i biglietti;
- *Biglietto:* rappresenta il documento che attesta l'acquisto da parte di un cliente;
- *BigliettoNormale*: biglietto senza alcun tipo di privilegi;
- *BigliettoBackstage:* biglietto con alcuni privilegi;
- *Merchandise*: rappresenta l'oggetto che viene venduto;
- *Promozione:* rappresenta le promozioni per un determinato evento.

2.3 SSD e Contratti

Il passo successivo è la creazione dei Diagrammi di Sequenza di Sistema (SSD) al fine di mostrare gli eventi di input e di output dei vari casi d'uso esaminati in ciascuna iterazione. Inoltre, le operazioni degli SSD verranno descritte attraverso i Contratti.

- ❖ Iterazione 1

SSD UC1



Contratti UC1

Contratto U1C1: inserisciNuovoConcerto
Operazione: *inserisciNuovoConcerto (nomeartista:String)*
Riferimenti: Caso d'uso: Inserisci concerto
Pre-condizioni: -
Post-Condizioni: -

Contratto U1C2: inserimentoDatiConcerto
Operazione: *inserisciNuovoConcerto (codiceartista:Int ,nomeconcerto: string, prezzobase:float, codiceConcerto: Int)*
Riferimenti: Caso d'uso: Inserisci concerto
Pre-condizioni: -
Post-Condizion: - È stata recuperata l'istanza a di Artista sulla base di codiceArtista;

- è stata creata una istanza c di Concerto;
- gli attributi di c sono stati inizializzati;
- L'istanza c è stato associata a MusicShowFactory tramite associazione "corrente";
- L'istanza a è stata associata a c di Concerto tramite associazione "fa";

Contratto U1C3: inserisciEvento
Operazione: *inserisciEvento(nomeEvento:String, data: Date, orario: Date, location: String, codiceEvento: int)*
Riferimenti: Caso d'uso: Inserisci concerto
Pre-condizioni: è in corso l'inserimento di concerto c;

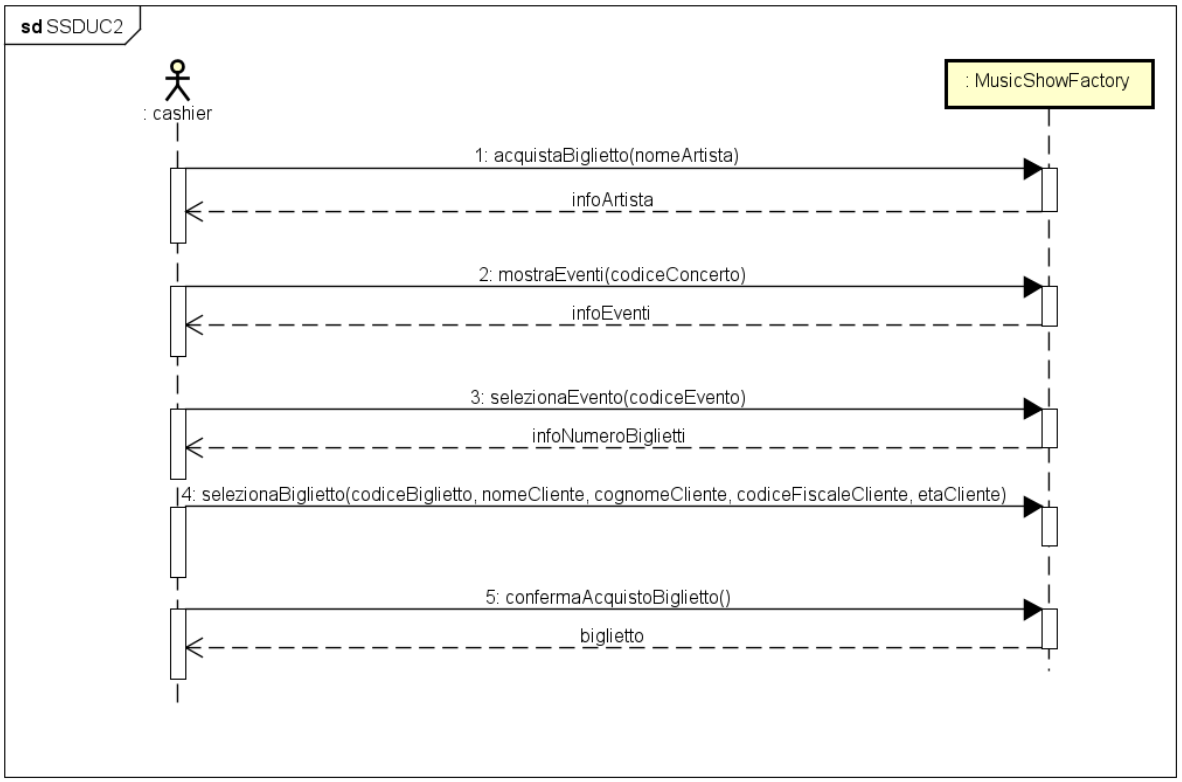
Post-Condizioni:

- È stata creata un'istanza e di Evento;
- Gli attributi di e sono stati inizializzati;
- L'istanza e è stata associata a c di Concerto tramite associazione "ha".

Contratto U1C4: confermaInserimentoConcerto
Operazione: *confermaInserimentoConcerto()*
Riferimenti: Caso d'uso: Inserisci concerto
Pre-condizioni: è in corso l'inserimento di Concerto c;
Post-Condizioni: - è stata associata l'istanza c di Concerto corrente a MusicShowFactory tramite associazione "gestisce".

- ❖ Iterazione 2

SSDUC2



Contratto UC2
Contratto U2C1: acquistoBiglietto
Operazione: *acquistaBiblietto(nomeArtista: String)*
Riferimenti: Caso d'uso: Gestisci acquisto biglietto
Pre-condizioni: -
Post-Condizioni: -

Contratto U2C2: mostraEventi
Operazione: *mostraEventi(codiceConcerto:int)*
Riferimenti: Caso d'uso: Gestisci acquisto biglietto
Pre-condizioni: E' in corso l'acquisto di un biglietto
Post-Condizioni:

Contratto U2C3: selezionaEvento
Operazione: *selezionaEvento(codiceEvento:int)*
Riferimenti: Caso d'uso: Gestisci acquisto biglietto
Pre-condizioni: - E' in corso l'acquisto di un biglietto.
Post-Condizioni: -

Contratto U2C4: selezionaBiglietto
Operazione: *selezionaBiglietto(codiceBiglietto:int, nomeCliente:String, cognomeCliente:String, codiceFiscaleCliente:String, etaCliente:int)*
Riferimenti: Caso d'uso: Gestisci acquisto biglietto
Pre-condizioni: È in corso l'acquisto di un biglietto.
Post-Condizioni:

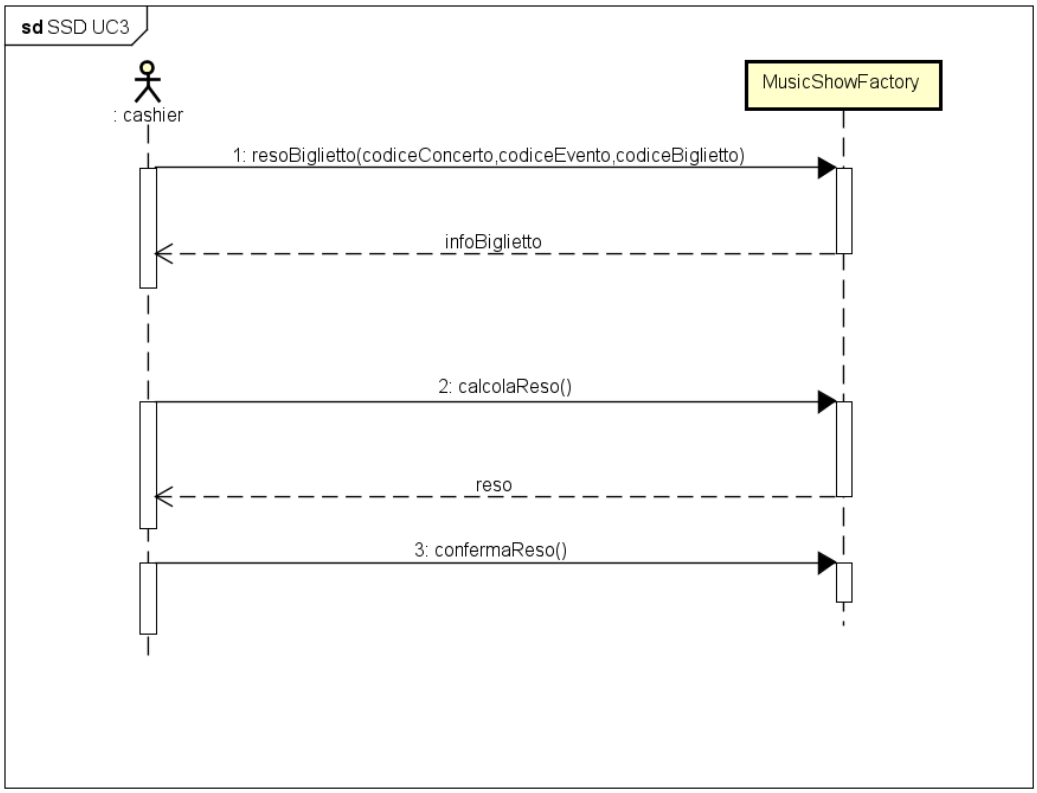
- È stata creata un'istanza b di biglietto e il suo codice è stato inizializzato.
- È stata associata l'istanza b all'istanza e di evento tramite associazione "corrente"

Contratto U2C5: confermaAcquistaBiglietto
Operazione*:* *confermaAcquistoBiglietto()*
Riferimenti: Caso d'uso: Gestisci acquisto biglietto
Pre-condizioni: - E' in corso l'acquisto di un biglietto.
Post-Condizioni:

- E' stato calcolato e settato l'attributo prezzo di b;
- E' stato associata l'istanza b tramite la relazione "relativo a" all'istanza e di evento

❖ Iterazione 3

SSDUC3



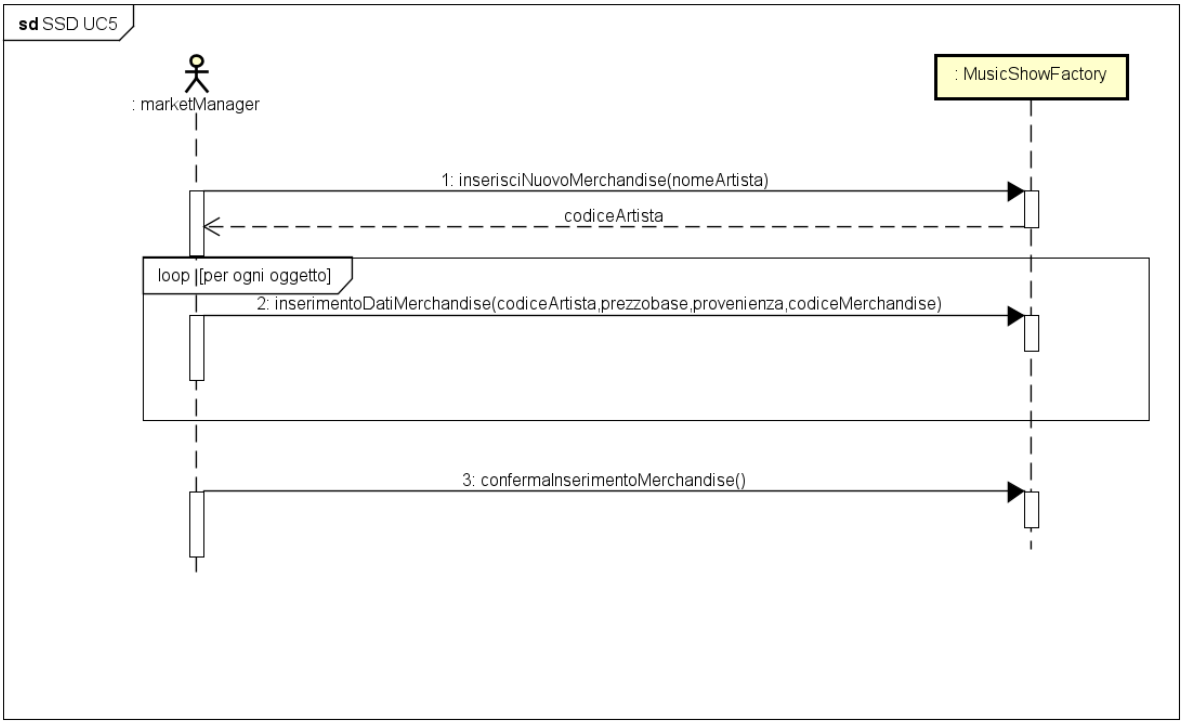
Contratti UC3
Contratto U3C1: resoBiglietto
Operazione: *resoBiglietto(codiceConcerto:int, codiceEvento:int, codiceBiglietto:int)*
Riferimenti: Caso d'uso: Gestisci rimborso biglietto
Pre-condizioni: -
Post-Condizioni: - è stata recuperata l'istanza b di biglietto sulla base dei codici.

Contratto U3C2: calcolaReso
Operazione: *calcolaReso()*
Riferimenti: Caso d'uso: Gestisci rimborso biglietto
Pre-condizioni: - è in corso il rimborso di un biglietto
Post-Condizioni: -

Contratto U3C3: confermaReso
Operazione: *confermaReso()*
Riferimenti: Caso d'uso: Gestisci rimborso biglietto
Pre-condizioni: - è in corso il rimborso di un biglietto
Post-Condizioni: - l'attributo "rimborsato" del biglietto b è stato settato a true.

- Iterazione 4

SSDUC5



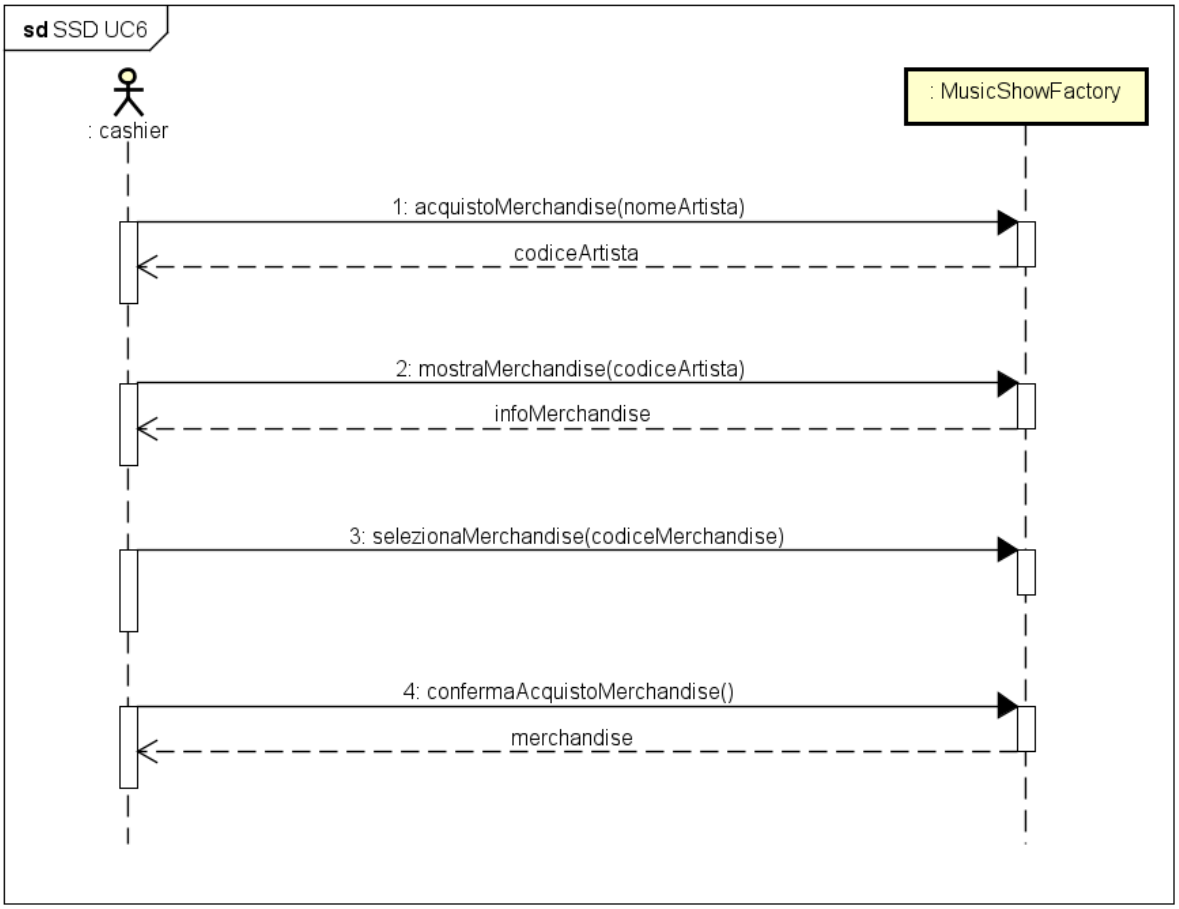
Contratti UC5
Contratto U5C1: inserisciNuovoMerchandise
Operazione: *inserisciNuovoMerchandise (nomeArtista:String)*
Riferimenti: Caso d'uso: Inserisci Merchandise
Pre-condizioni: -
Post-Condizioni: -

Contratto U5C2: inserimentoDatiMerchandise
Operazione: *inserimentoDatiMerchandise (codiceArtista:int, prezzobase:float, provenienza:String, codiceMerchandise:int)*
Riferimenti: Caso d'uso: Inserisci Merchandise
Pre-condizioni: -
Post-Condizioni: - È stata recuperata l'istanza a di Artista sulla base di codiceArtista;

1. È stata creata l'istanza m di Merchandise;
2. Gli attributi di m sono stati inizializzati;
3. L'istanza a è stata associata all'istanza m tramite associane "Relativo a";
4. m è stata associata a MusicShowFactory tramite l'associazione "corrente";

Contratto U5C3: confermaInserimentoMerchandise
Operazione: *confermaInserimentoMerchandise()*
Riferimenti: Caso d'uso: Inserisci Merchandise
Pre-condizioni: - è in corso l'inserimento del merchandise m;
Post-Condizioni: - È stata associata l'istanza m di Merchandise corrente a MusicShowFactory tramite l'associazione "organizza";

SSDUC6



Contratti UC6

Contratto U6C1: acquistoMerchandise
Operazione: *acquistoMerchandise(nomeArtista: String)*
Riferimenti: Caso d'uso: Gestisci acquisto biglietto
Pre-condizioni: -
Post-Condizioni: -

Contratto U6C2: mostraMerchandise
Operazione: *mostraMerchandise(codiceArtista : int)*
Riferimenti: Caso d'uso: Gestisci acquisto biglietto
Pre-condizioni: - È in corso l'acquisto di un merchandise
Post-Condizioni:

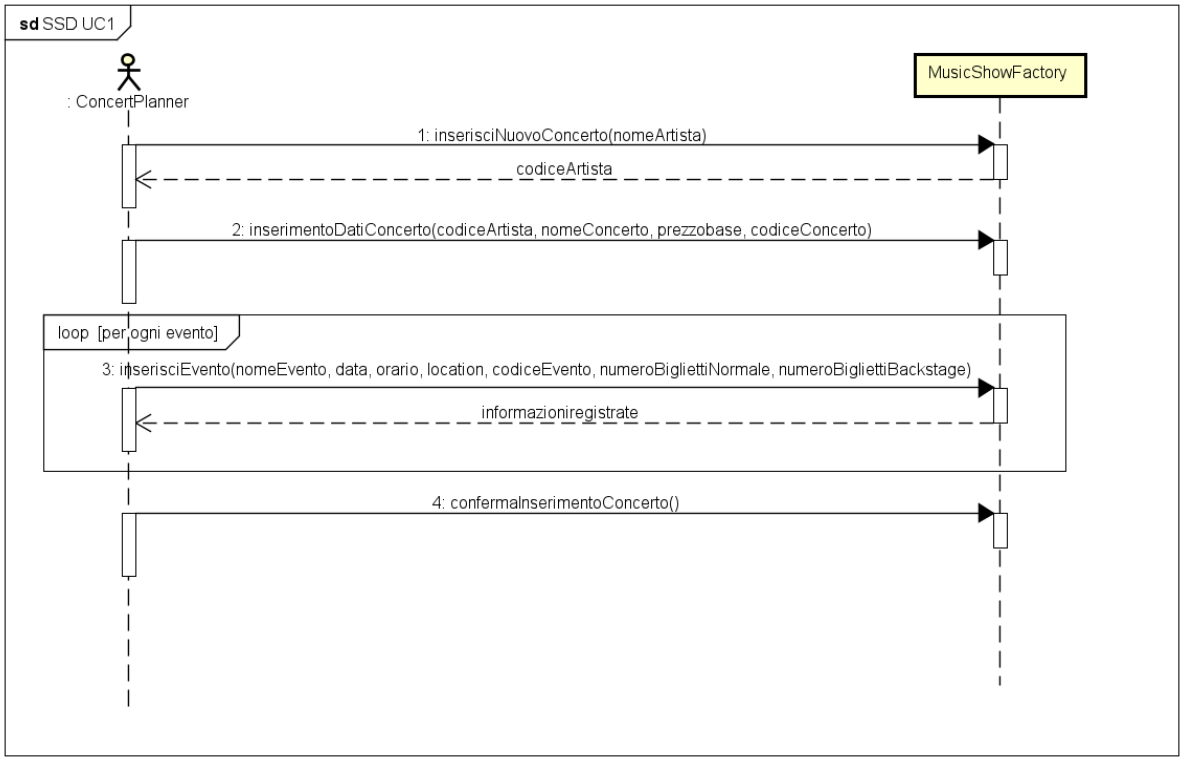
Contratto U6C3: selezionaMerchandise
Operazione: *selezionaMerchandise(codiceMerchandise:int)*
Riferimenti: Caso d'uso: Gestisci acquisto biglietto
Pre-condizioni: - È in corso l'acquisto di un merchandise.
Post-Condizioni: - è stato recuperato l'attributo prezzo di m tramite la funzione "prezzoEffettivo";

Contratto U6C4: confermaAcquistoMerchandise
Operazione: *confermaAcquistoMerchandise()*
Riferimenti: Caso d'uso: Gestisci acquisto biglietto
Pre-condizioni: - È in corso l'acquisto di un merchandise.
Post-Condizioni: -

- Iterazione 5

L'unico SSD che si modifica in questo caso è l'SSDUC1 poiché cambiano gli attributi nell'operazione *InserisciEvento.*

SSDUC1_modificato



Solo alcuni contratti vengono modificati:

Per il caso d'uso UC1

Contratto U1C3: inserisciEvento

Operazione: *inserisciEvento(nomeEvento:String, data: Date, orario: Date, location: String, codiceEvento: int, numeroBigliettiNormali:int, numeroBigliettiBackstage:int)*

Riferimenti: Caso d'uso: Inserisci concerto

Pre-condizioni: è in corso l'inserimento di concerto c;

Post-Condizioni:

- È stata creata un'istanza e di Evento;
- Gli attributi di e sono stati inizializzati;
- L'istanza e è stata associata a c di Concerto tramite associazione "ha".

Per il caso d'uso UC2

Contratto U2C5: confermaAcquistaBiglietto

Operazione*:* *confermaAcquistoBiglietto()*

Riferimenti: Caso d'uso: Gestisci acquisto biglietto

Pre-condizioni: - È in corso l'acquisto di un biglietto.

Post-Condizioni:

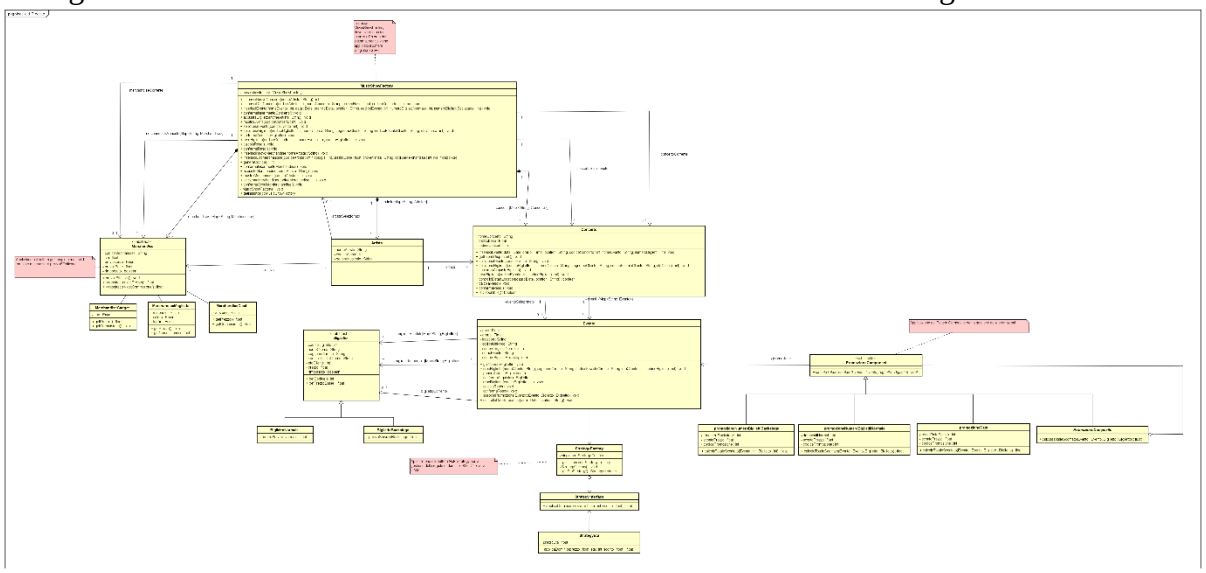
- È stato calcolato e settato l'attributo prezzo di b usando anche le promozioni e le regole di dominio D3 e D4;
- È stato associata l'istanza b tramite la relazione "relativo a" all'istanza e di evento.

3 Progettazione

Piccola premessa: tutti i diagrammi sia di questa sezione che delle altre sono presenti in formato png nelle varie sottocartelle delle singole iterazioni.
In questa fase viene considerato l'insieme dei diagrammi che descrivono la progettazione logica sia da un punto di vista dinamico con i diagrammi di iterazione e sia dal punto di vista statico con il Diagramma delle classi.

3.1 Diagramma delle Classi

Il diagramma delle classi alla fine delle 5 iterazioni è risultato essere il seguente:



Per una migliore visualizzazione si consiglia di vedere il file "M2_DiagrammadelleClassi.png".

3.1.1 Pattern GoF Utilizzati

Singleton: è un pattern creazionale che ha come scopo quello di assicurare che una classe abbia una ed una sola istanza. Il pattern è stato utilizzato per la classe *musicshowfactory* per garantire un unico accesso alla classe principale.

Template Method: è un pattern comportamentale che ha lo scopo di definire la struttura di un algoritmo in un metodo lasciando alcune parti non specificate. Le parti non specificate sono definite in altri metodi che sono implementati nelle sottoclassi. Il pattern è stato utilizzato per la gestione dei prezzi relativi ai diversi oggetti del merchandise. I metodi nelle sottoclassi implementano le politiche di prezzo degli oggetti di cui si compone il merchandise. Ogni oggetto del merchandise è trattato in maniera differente in base alle sue caratteristiche.

Composite: è un pattern strutturale che consente di trattare in modo uniforme oggetti semplici e composizioni degli stessi. Il pattern è stato utilizzato dichiarando la classe astratta *PromozioneComponent* che viene estesa di tre sottoclassi semplici(leaf) ed una classe composta(composite). Il fatto che questa ultima sia un contenitore di componenti consente di immagazzinare al suo interno sia componenti semplici sia altri contenitori. Nel caso qui presente, è stato applicato per modellare le diverse promozioni.

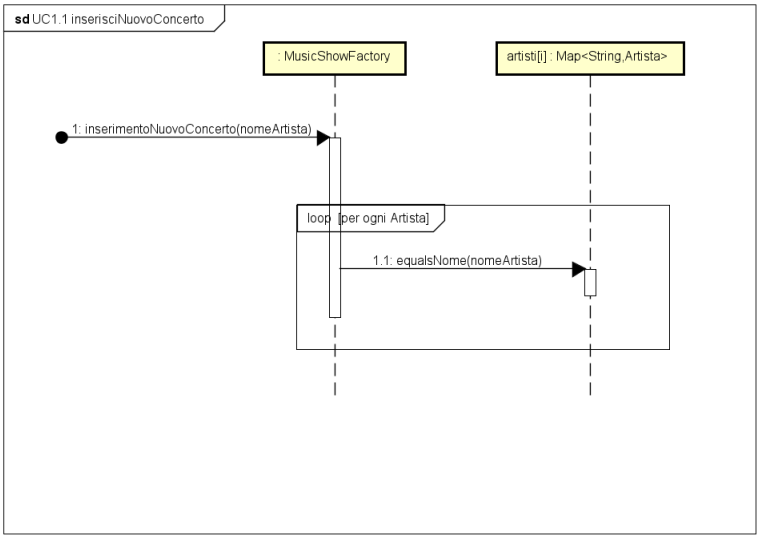
Strategy: è un pattern comportamentale che permette di definire una famiglia di algoritmi ed incapsularli. Inoltre, prevede che gli algoritmi siano interscambiabili tra loro in base ad una specifica condizione in modalità trasparente a chi ne fa uso. Il pattern è stato utilizzato per gestire le regole di dominio dell'età R3 ed R4. Nel caso qui presente la condizione che attiva il pattern è determinata dal parametro età del cliente.

3.2 Diagrammi di Sequenza

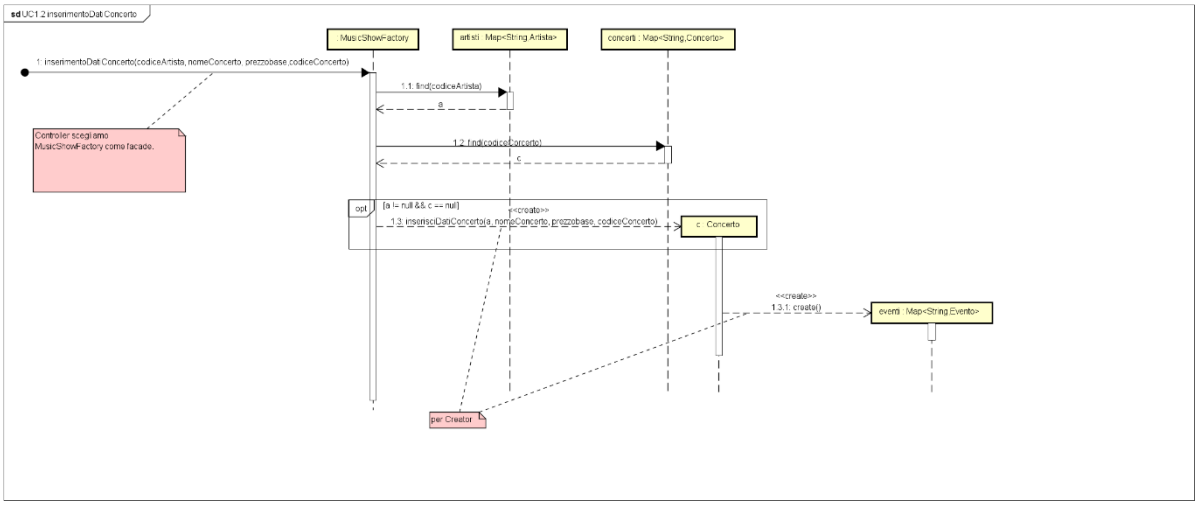
Come diagrammi di Iterazione sono stati usati i diagrammi di sequenza in modo da mostrare l'interazione tra un insieme di oggetti. I diagrammi di sequenza inseriti in questa fase sono quelli finali. Per vedere gli altri o per vederli meglio basta andare nelle singole cartelle delle iterazioni o vedere il file "MusicShowFactory.asta".

Caso d'uso UC1

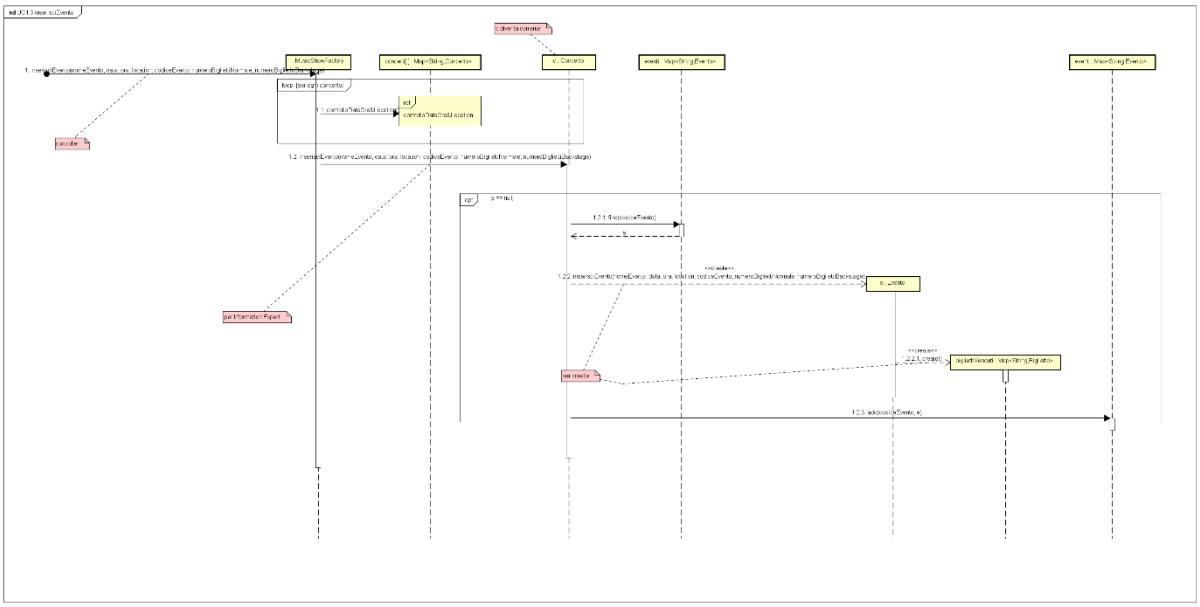
- inserisciNuovoConcerto



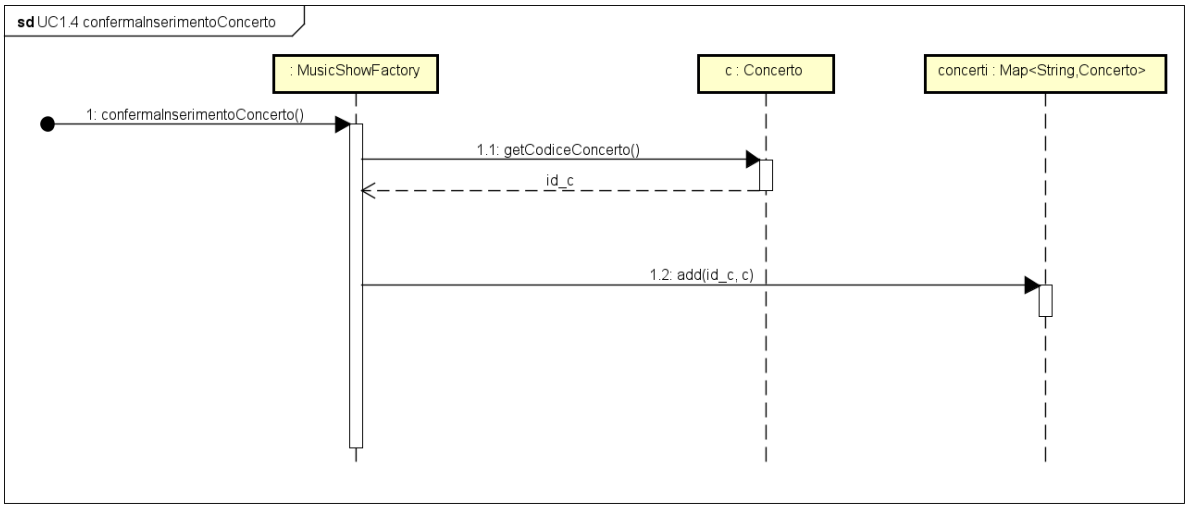
- inserimentoDatiConcerto



- inserisciEvento

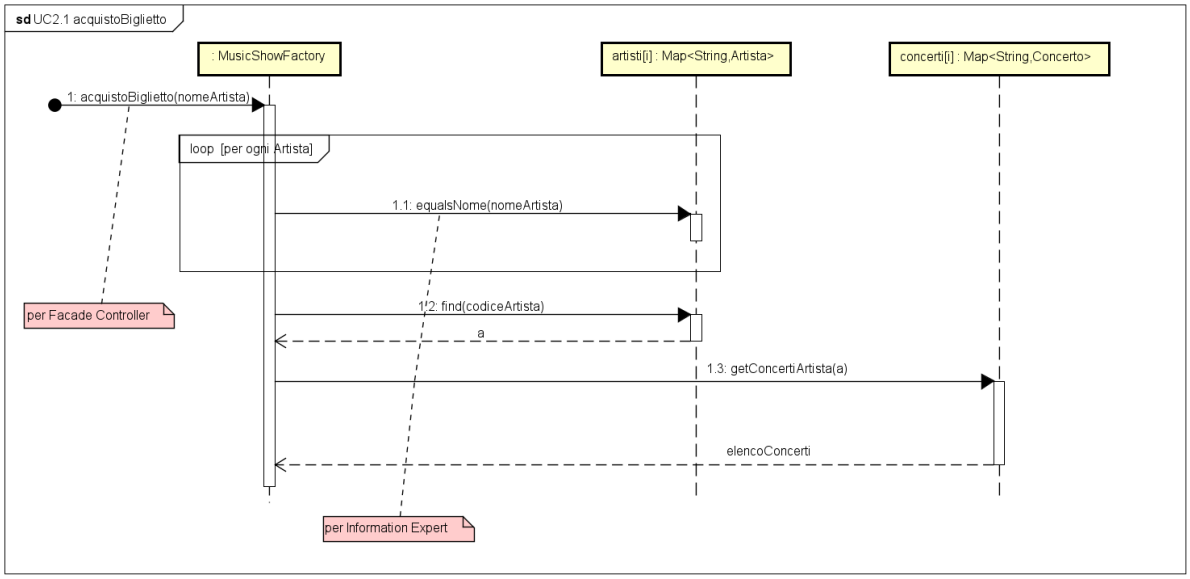


- confermaInserimentoConcerto

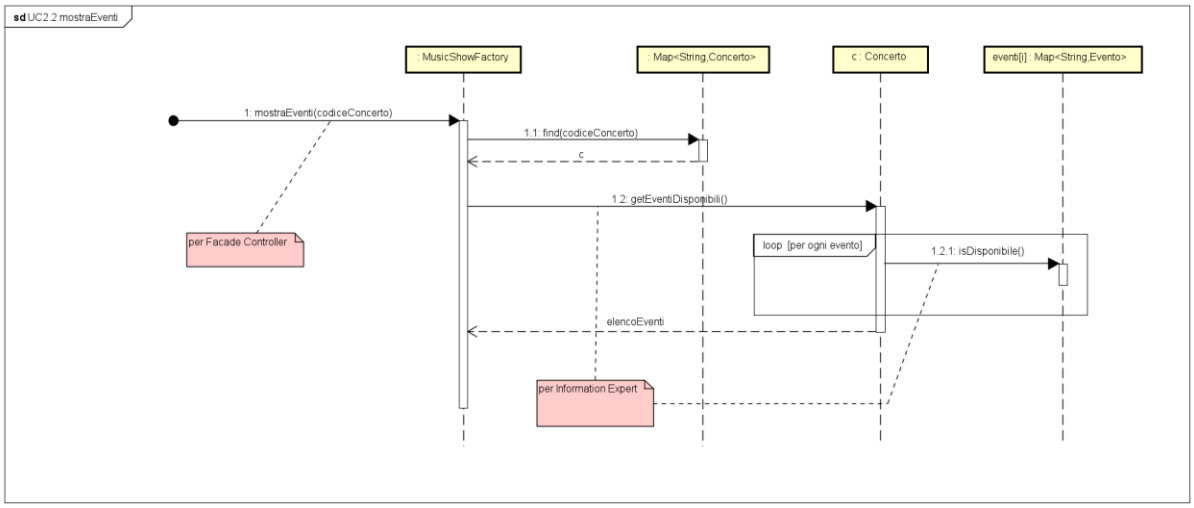


Caso d'uso UC2

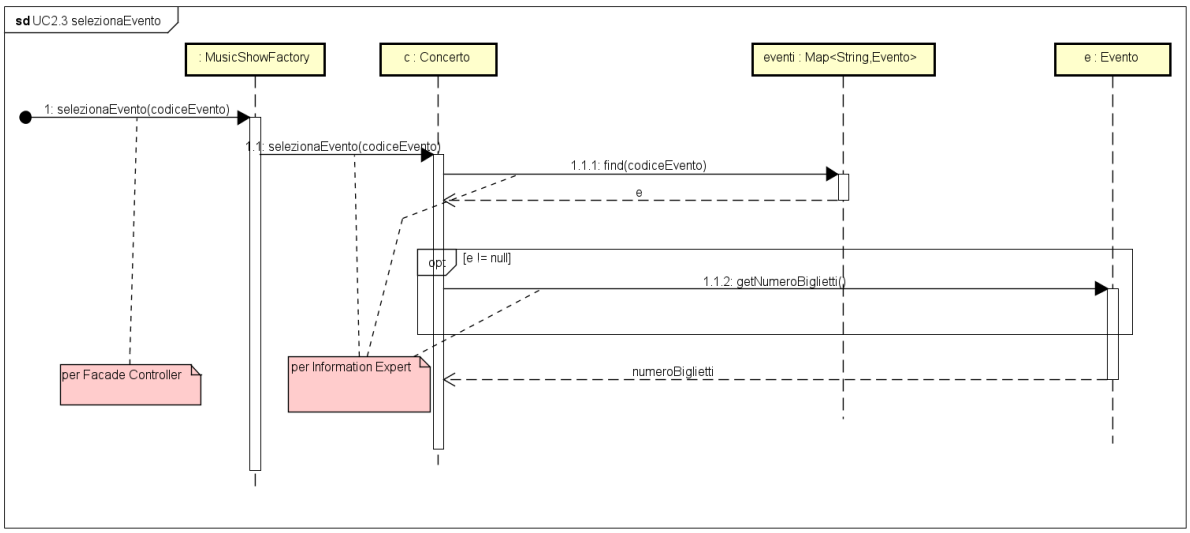
- acquistaBiglietto



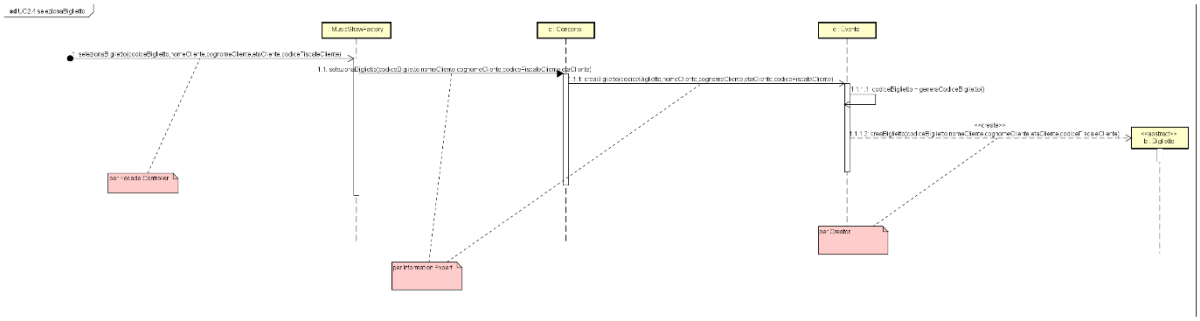
- mostraEventi



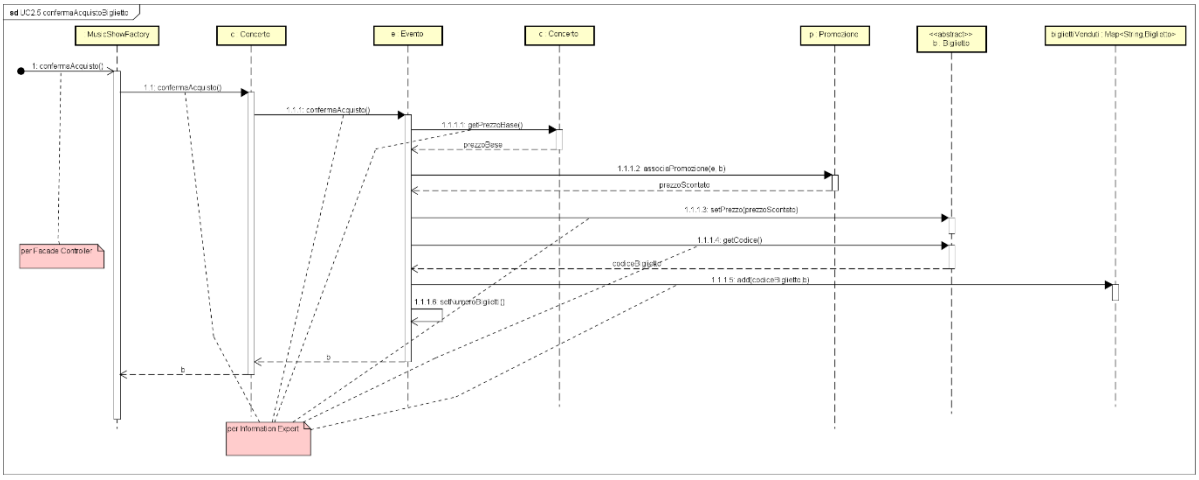
- selezionaEvento



- selezionaBiglietto

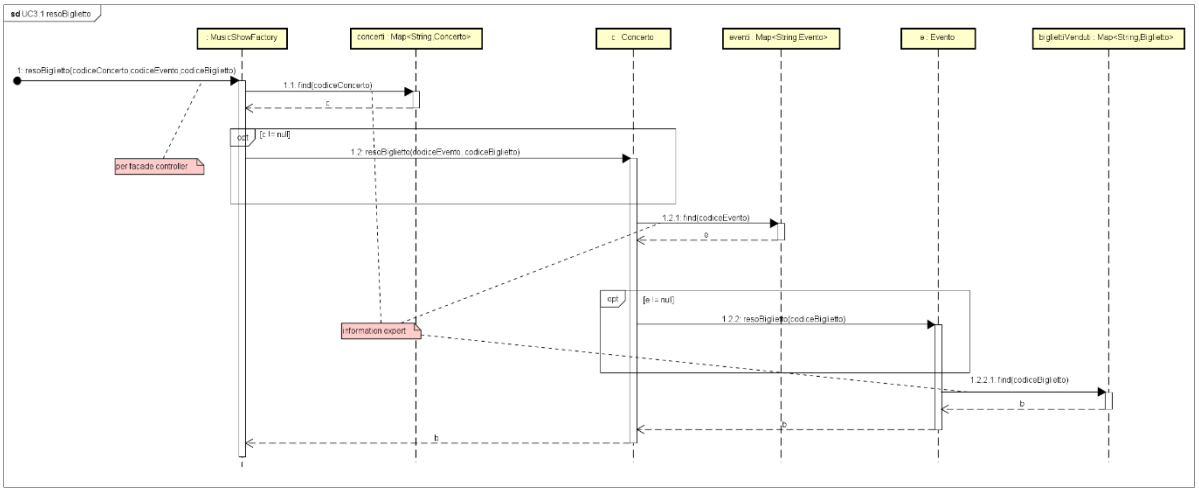


- confermaAcquistoBiglietto

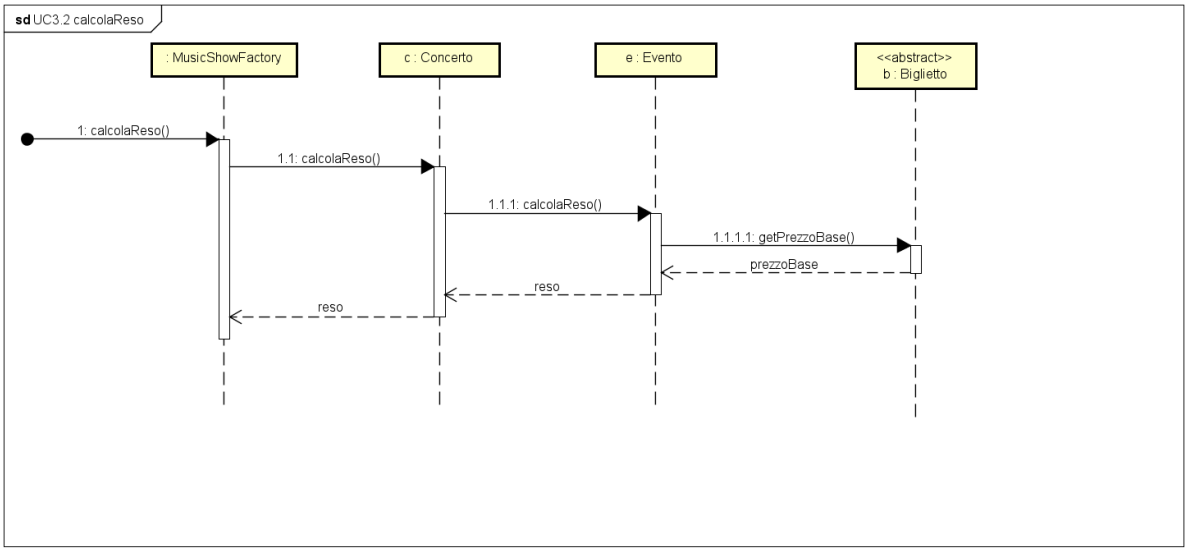


Caso d'uso UC3

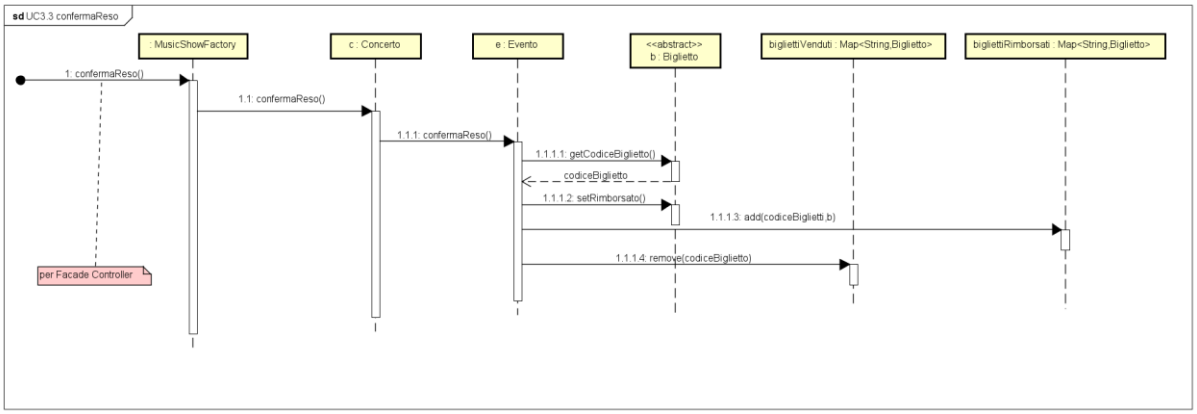
- resoBiglietto



- calcolaReso

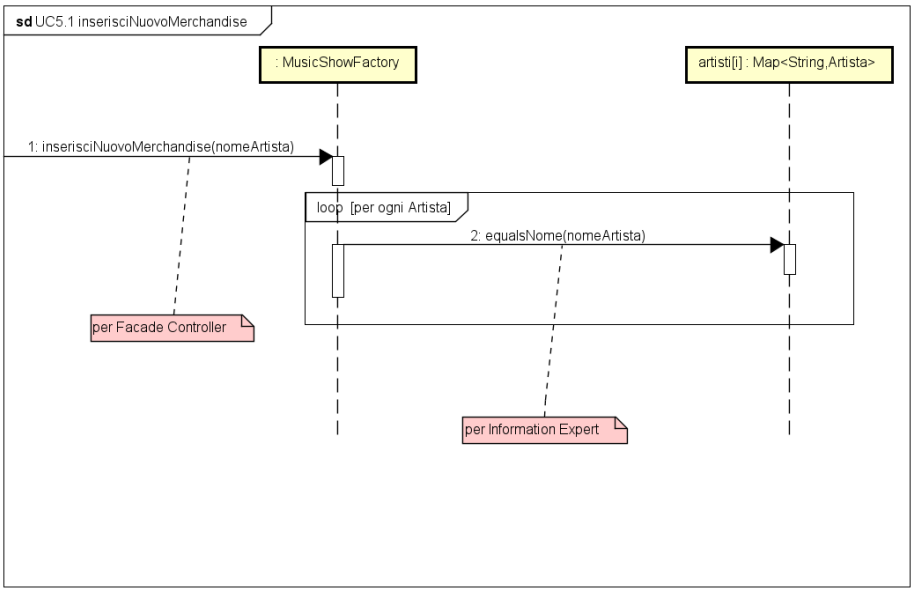


- confermaReso

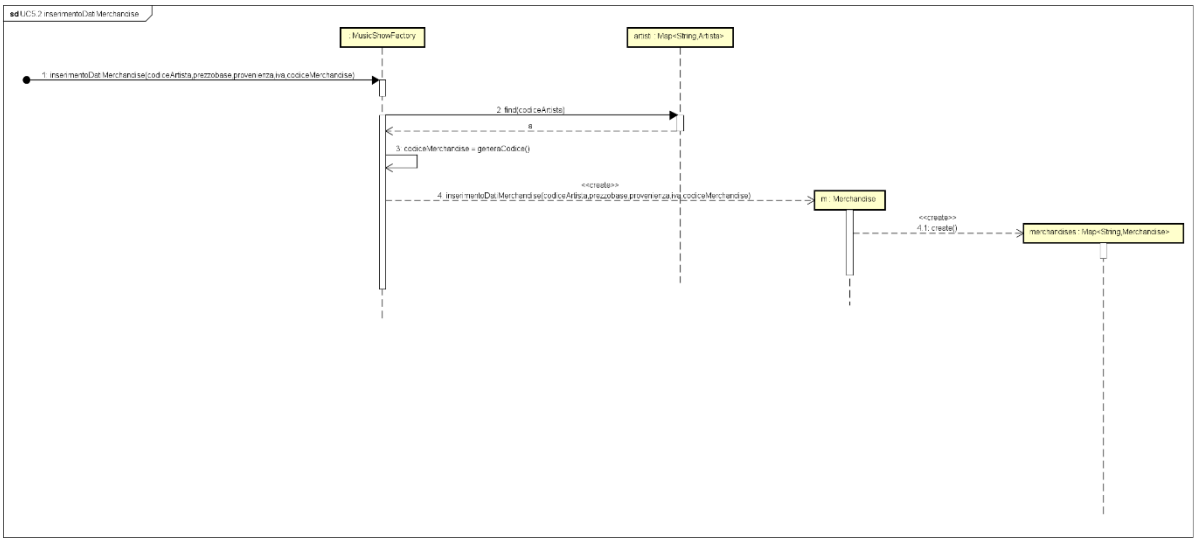


Caso d'uso UC5

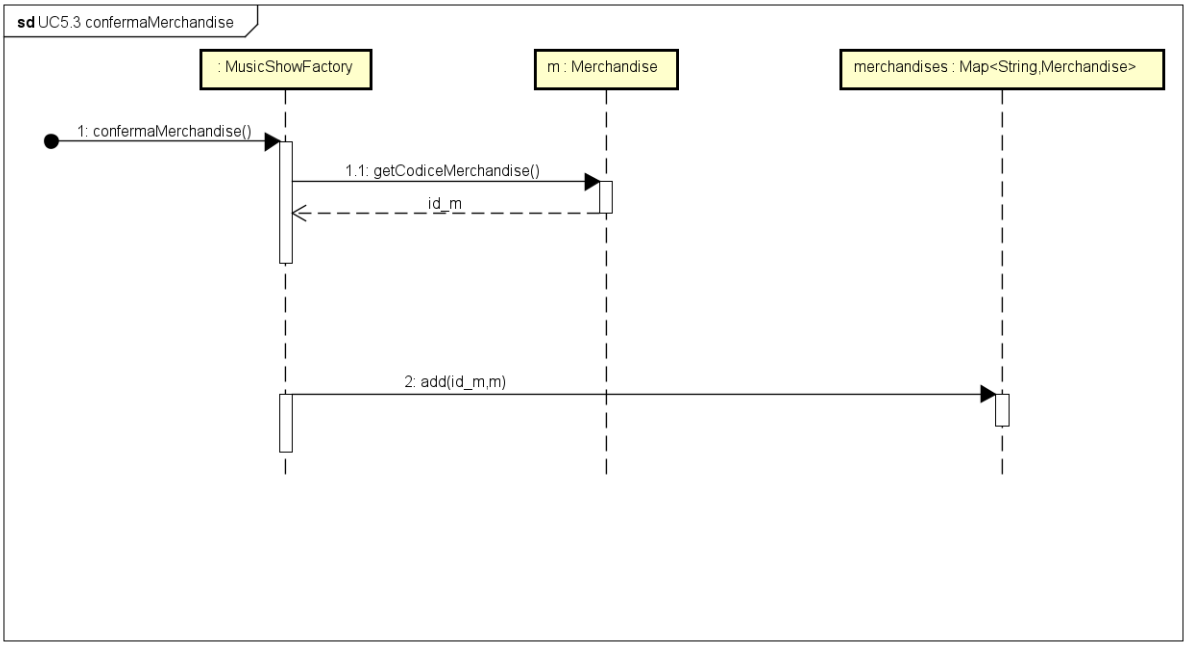
- inserisciNuovoMerchandise



- inserimentoDatiMerchandise

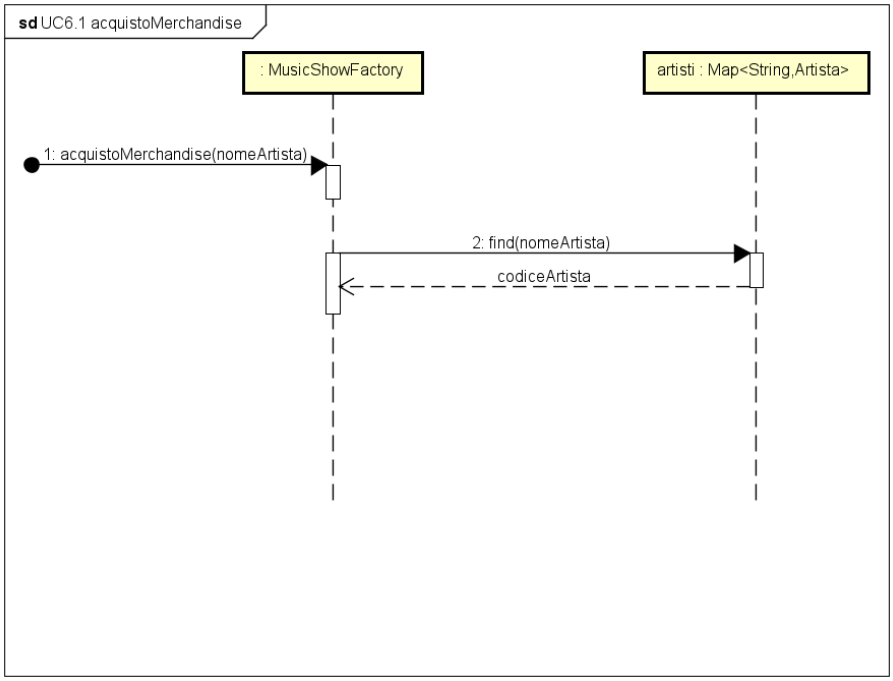


- confermaMerchandise

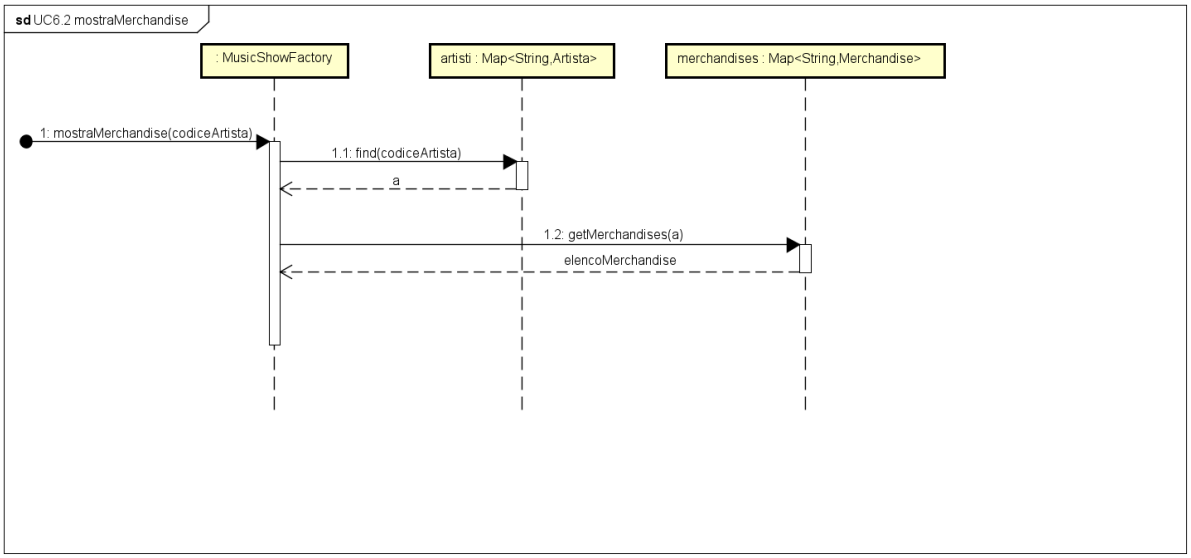


Caso d'uso UC6

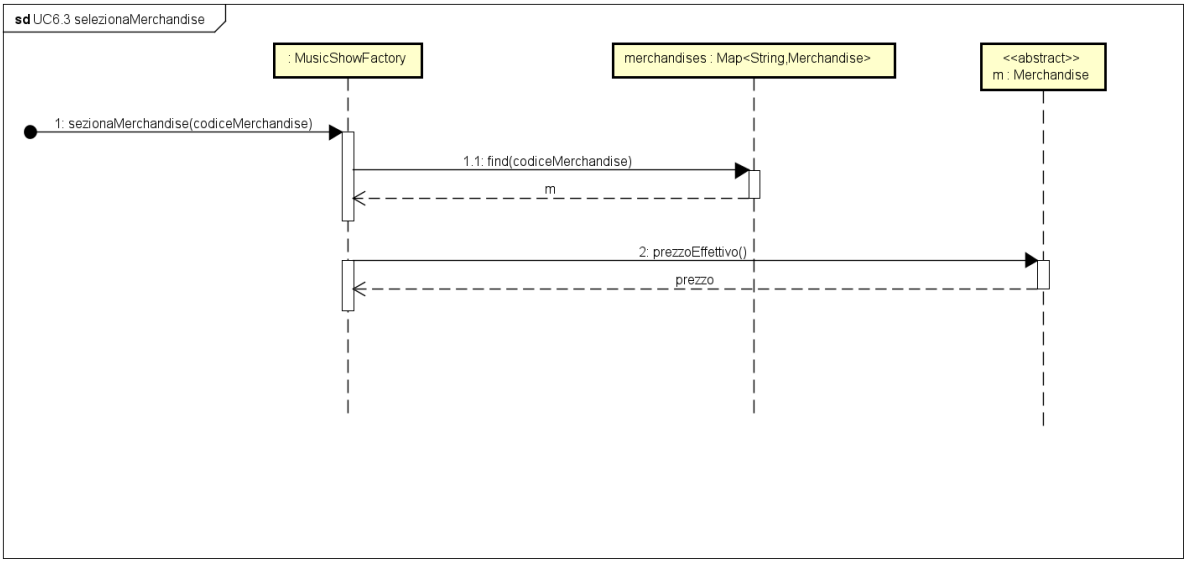
- acquistoMerchandise



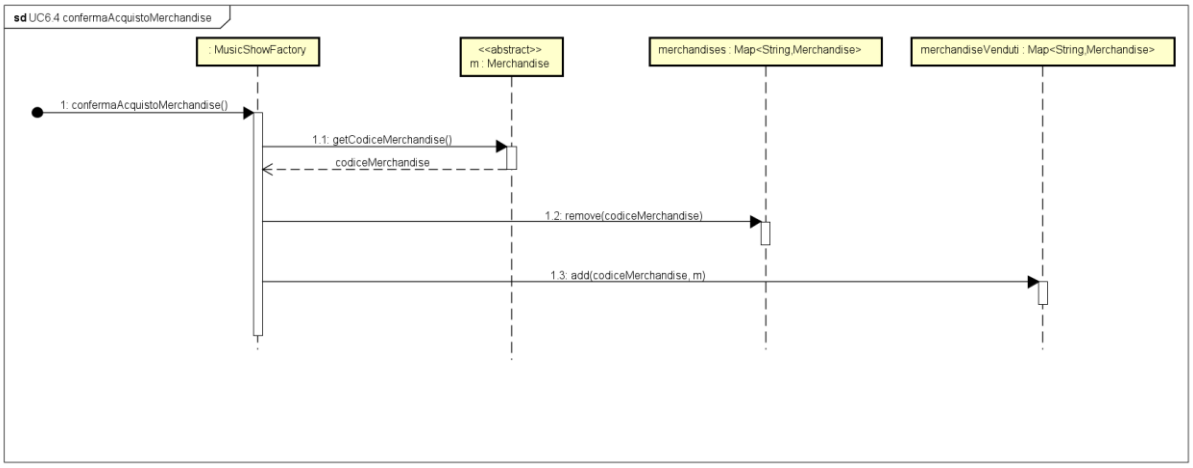
- mostraMerchandise



- selezionaMerchandise



- confermaAcquistoMerchandise



4 Testing

4.1 Introduzione

Il testing è una fase essenziale del processo di sviluppo di un programma. Si effettua una verifica sperimentale fatta mediate esecuzione del programma, selezionando alcuni dati di ingresso e valutando i risultati.
I test dell'applicazione sono stati realizzati sfruttando test unitari tramite l'ausilio di JUnit 5. Sono state testate delle classi singole con il testing unitario e sono state testate tutte le funzioni principali tramite l'ausilio del Test di Sistema effettuato anche da parti terze.

4.2 Individuazione dei casi di test e Testing Unitario

Le classi che sono state individuate per essere testate con JUnit sono:

- Artista
  - Testing generale dei metodi get e set
- MusicShowFactory
  - testinserisciArtista(): con il test si accerta che più artisti vengano inseriti in maniera corretta dentro la mappa. Inoltre, sono stati testati i casi in cui l'artista che si vuole inserire abbia il presente codiceArtista o il nomeArtista nella mappa. In questi casi, l'artista non viene inserito.
  - testrimuoviArtista(): con il test si accerta che l'artista viene rimosso in maniera corretta.
  - testinserisciConcerto(): con il test si accerta che un concerto venga creato in maniera corretta. Inoltre, è stato testato il caso in cui il concerto che si vuole inserire abbia il codiceConerto presente nella mappa. In questo caso, il concerto non viene inserito.
- Concerto
  - testInserisciEvento(): con il test si accerta il corretto inserimento degli eventi. Inoltre, se venisse inserito un evento con lo stesso codice o con una location già impegnata in quella data, questo non verrebbe inserito.
  - testrimuoviEvento(): con il test si accerta che l'evento viene rimosso in maniera corretta testando che il valore della mappa sia corretto.
- Evento
  - testgeneraBiglietto(): con il test si accerta la corretta creazione del biglietto sia in versione normale che in versione backstage. Inoltre, è stato verificato che il che il prezzo nel caso in cui si ha una età minore di 8 sarà 0 e nel caso di età compresa tra 8 e 18 sarà il 50% del prezzo base.
  - testresoBiglietto(): viene testato il corretto reso di un biglietto testando che nella mappa dei biglietti venduti ci sia il corretto numero di biglietti.

4.3 Test di sistema

Sono stati eseguiti diversi test di sistema manuali di funzionamento dell'applicazione per accertare che tutto funzioni in maniera corretta lanciandola e provando a portare a termine tutte le operazioni. L'applicazione viene eseguita tramite terminale.
A seguito di questi test non sono emersi grandi errori nel funzionamento. Inoltre, l'applicazione funziona correttamente ed ha mantenuto prestazioni sempre ottimali.