UNIVERSITÀ DEGLI STUDI DI NAPOLI FEDERICO II

CORSO DI LAUREA IN INGEGNERIA INFORMATICA
CORSO DI INGEGNERIA DEL SOFTWARE
PROF. A.R. FASOLINO - A.A. 2023 - 24

Progetto

Vendita biglietti cinema multisala "Rainbow MultiScreen"

(esempio svolto parzialmente)

Studente: Nome Cognome Matricola Email

Versione del *inserire data*

INDICE

Sistema di vendita (online e al botteghino) di biglietti per il cinema multisala "Rainbow MultiScreen"

1. Specifiche informali

Si vuole realizzare una applicazione per la vendita (online e al botteghino) di biglietti per il cinema multisala "Rainbow MultiScreen".
Il cinema è dotato di *n* sale, ciascuna con una sua capienza; tutti i posti sono numerati. In ogni sala è in programmazione un diverso film per settimana, dal lunedì alla domenica. Ogni sala ha 4 proiezioni al giorno, alle stesse ore tutti i giorni, ma gli orari delle sale sono sfalsati. Ogni film ha un titolo, una trama, un filmato trailer, e un genere (commedia, azione, fantascienza, …)
Il sistema deve consentire al Direttore di definire la programmazione mensile, inserendo per ciascun film la sala e la settimana (data inizio, data fine) in cui sarà proiettato. *Aliquando*, il Direttore definisce: *i)* gli orari di proiezione per ogni sala; *ii)* il costo del biglietto, che è uguale per tutte le sale, gli orari e i film.
Un cliente può visualizzare la programmazione mensile, o effettuare ricerche per titolo del film o per genere. I clienti ricorrenti possono registrarsi fornendo nome utente, password, indirizzo e-mail e carta di credito. I clienti registrati ricevono uno sconto del 10% per ogni acquisto. Il cliente effettua l'acquisto con carta di credito (quella già registrata per i clienti ricorrenti, altrimenti inserita al momento dell'acquisto per i clienti occasionali), indicando film, giorno, e orario di proiezione, il numero di posti desiderato e l'indirizzo di posta elettronica su cui ricevere i biglietti elettronici. I posti sono assegnati automaticamente dal sistema. Al completamento del pagamento (con prezzo scontato automaticamente se il cliente è registrato), il sistema invia un messaggio e-mail di conferma al cliente, elencando sala, orario, numerazione dei posti assegnati, e il totale pagato.
Il cinema ha un botteghino, ove gli impiegati alle casse possono effettuare la vendita dei posti ai clienti che non li hanno acquistati online: in tal caso, al pagamento, invece di inviare la e-mail di conferma al cliente, il sistema effettua la stampa dei biglietti cartacei.
Ogni ultimo venerdì del mese il sistema invia per e-mail ai clienti registrati una locandina con la programmazione del mese successivo.
Per finalità di analisi delle vendite, il Direttore può richiedere al sistema di generare un report del numero di posti venduti per ciascun film o per ciascun mese di programmazione.

2. Analisi e specifica dei requisiti

2.1 Analisi nomi-verbi

Si vuole realizzare una applicazione per la vendita online di biglietti per il cinema multisala "Rainbow MultiScreen".
Il cinema è dotato di *n* sale, ciascuna con una sua capienza; tutti i posti sono numerati. In ogni sala è in programmazione un diverso film per settimana, dal lunedì alla domenica. Ogni sala ha 4 proiezioni al giorno, alle stesse ore tutti i giorni, ma gli orari delle sale sono sfalsati. Ogni film ha un titolo, una trama, un filmato trailer, e un genere (commedia, azione, fantascienza, …)
Il sistema deve consentire al Direttore di definire la programmazione mensile, inserendo per ciascun film la sala e la settimana (data inizio, data fine) in cui sarà proiettato. *Aliquando*, il Direttore definisce: *i)* gli orari di proiezione per ogni sala; *ii)* il costo del biglietto, che è uguale per tutte le sale, gli orari e i film.
Un cliente può visualizzare la programmazione mensile, o effettuare ricerche per titolo del film o per genere. I clienti ricorrenti possono registrarsi fornendo nome utente, password, indirizzo e-mail e carta di credito. I clienti registrati ricevono uno sconto del 10% per ogni acquisto. Il cliente effettua l'acquisto con carta di credito (quella già registrata per i clienti ricorrenti, altrimenti inserita al momento dell'acquisto per i clienti occasionali), indicando film, giorno, e orario di proiezione, il numero di posti desiderato e l'indirizzo di posta elettronica su cui ricevere i biglietti elettronici. I posti sono assegnati automaticamente dal sistema. Al completamento del pagamento (con prezzo scontato automaticamente se il cliente è registrato), il sistema invia un messaggio e-mail di conferma al cliente, elencando sala, orario, numerazione dei posti assegnati, e il totale pagato.
Il cinema ha un botteghino, ove gli impiegati alle casse possono effettuare la vendita dei posti ai clienti che non li hanno acquistati online: in tal caso, al pagamento, invece di inviare la e-mail di conferma al cliente, il sistema effettua la stampa dei biglietti cartacei.
Ogni ultimo venerdì del mese il sistema invia per e-mail ai clienti registrati una locandina con la programmazione del mese successivo.
Per finalità di analisi delle vendite, il Direttore può richiedere al sistema di generare un report del numero di posti venduti per ciascun film o per ciascun mese di programmazione.

LEGENDA:
Classe
Attributo
Funzionalità
Attore
Classe-Attore

2.2 Revisione dei requisiti

1. *Il cinema è dotato di n sale.*
2. *Ogni sala ha una propria capienza.*
3. *I posti di ogni sala sono numerati.*
4. *Ogni sala ha in programma un diverso film per settimana.*
5. *Ogni sala ha 4 proiezioni al giorno.*
6. *Gli orari delle proiezioni per sala sono gli stessi tutti i giorni.*
7. *Gli orari delle proiezioni in diverse sale sono sfalsati.*
8. *Di ogni film si vuole memorizzare titolo, trama, filmato trailer, e genere.*

9. *Il sistema deve offrire al Direttore una funzionalità per definire la programmazione mensile.*
10. *Definire la programmazione mensile consiste nell'inserire per ciascun film la sala e la settimana (data inizio, data fine) in cui sarà proiettato.*
11. *Il sistema deve offrire al Direttore una funzionalità per definire gli orari di proiezione per ogni sala.*
12. *Il sistema deve offrire al Direttore una funzionalità per definire il costo del biglietto.*
13. *Il costo del biglietto è uguale per tutte le sale, gli orari e i film.*
14. *Il sistema deve offrire al Cliente una funzionalità per visualizzare la programmazione mensile.*
15. *Il sistema deve offrire al Cliente una funzionalità per cercare un film per titolo.*
16. *Il sistema deve offrire al Cliente una funzionalità per cercare film per genere.*
17. *Il sistema deve offrire al Cliente una funzionalità per registrarsi.*
18. *Di ogni Cliente registrato si vuole memorizzare nome utente, password, indirizzo e-mail e carta di credito.*
19. *Il sistema deve offrire al Cliente una funzionalità per acquistare un biglietto con carta di credito.*
20. *Per effettuare l' acquisto di un biglietto il Cliente deve specificare film, giorno, orario di proiezione, numero di posti desiderato e indirizzo di posta elettronica su cui ricevere i biglietti elettronici.*
21. *Il sistema deve assegnare automaticamente i posti al momento dell'acquisto.*
22. *Il sistema deve applicare automaticamente uno sconto del 10% per l'acquisto da parte di Clienti registrati.*
23. *Dopo un acquisto il sistema deve inviare un messaggio e-mail di conferma al cliente, elencando sala, orario, numerazione dei posti assegnati, e il totale pagato.*
24. *Il sistema deve offrire agli impiegati del botteghino una funzionalità per vendere biglietti.*
25. *Il sistema deve offrire agli impiegati del botteghino una funzionalità per stampare i biglietti venduti.*
26. *Il sistema deve inviare ai clienti registrati per email ogni ultimo venerdì del mese una locandina con la programmazione del mese.*
27. *Il sistema deve offrire al Direttore una funzionalità per generare un report del numero di posti venduti per ciascun film o per ciascun mese di programmazione.*

2.3 Glossario dei termini

Termine	Descrizione	Sinonimi
Sala	Una delle sale in cui è suddiviso il cinema	
Capienza	Il numero di posti presenti in una sala	
Cliente	Un generico cliente che acquista senza essere registrazione	Cliente occasionale
Cliente registrato	Un cliente che ha effettuato la procedura di registrazione	
Botteghino	Luogo dove i clienti possono recarsi di persona per effettuare un acquisto	

2.4 Classificazione dei requisiti

2.4.1 Requisiti funzionali

ID	Requisito	Origine (n. frase dei requisiti revisionati)
RF01	Il sistema deve offrire al Direttore una funzionalità per definire la programmazione mensile	9
RF02	Il sistema deve offrire al Direttore una funzionalità per definire gli orari di proiezione per ogni sala	11
RF03	Il sistema deve offrire al Direttore una funzionalità per definire il costo del biglietto	12
RF04	Il sistema deve offrire al Cliente una funzionalità per visualizzare la programmazione mensile	14
RF05	Il sistema deve offrire al Cliente una funzionalità per cercare un film per titolo	15
RF06	Il sistema deve offrire al Cliente una funzionalità per cercare film per genere	16
RF07	Il sistema deve offrire al Cliente una funzionalità per registrarsi	17
RF08	Il sistema deve offrire al Cliente una funzionalità per acquistare un biglietto con carta di credito	19
RF09	Il sistema deve assegnare automaticamente i posti al momento dell'acquisto	21
RF10	Il sistema deve applicare automaticamente uno sconto del 10% per l'acquisto da parte di Clienti registrati	22
RF11	Il sistema deve inviare un messaggio e-mail di conferma al cliente, elencando sala, orario, numerazione dei posti assegnati, e il totale pagato, dopo un acquisto	23
RF12	Il sistema deve offrire agli impiegati del botteghino una funzionalità per vendere biglietti	24
RF13	Il sistema deve offrire agli impiegati del botteghino una funzionalità per stampare i biglietti venduti	25
RF14	Il sistema deve inviare ai clienti registrati per email ogni ultimo venerdì del mese una locandina con la programmazione del mese	26
RF15	Il sistema deve offrire al Direttore una funzionalità per generare un report del numero di posti venduti per ciascun film o per ciascun mese di programmazione	27

2.4.2 Requisiti sui dati

ID	Requisito	Origine (n. frase dei requisiti revisionati)
RD01	Il cinema è dotato di n sale	1
RD02	Ogni sala ha una propria capienza	2
RD03	I posti di ogni sala sono numerati	3
RD04	Ogni sala ha in programma un diverso film per settimana	4
RD05	Ogni sala ha 4 proiezioni al giorno	5
RD06	Gli orari delle proiezioni per sala sono gli stessi tutti i giorni ma sfalsati tra le sale.	6,7
RD07	Di ogni film si vuole memorizzare titolo, trama, filmato trailer, e genere	8
RD08	La programmazione mensile indica per ciascun film la sala e la settimana (data inizio, data fine) in cui sarà proiettato	10
RD09	Il costo del biglietto è uguale per tutte le sale, gli orari e i film.	13
RD10	Di ogni Cliente registrato si vuole memorizzare nome utente, password, indirizzo e-mail e carta di credito	18
RD11	Per effettuare l'acquisto di un biglietto il Cliente deve specificare film, giorno, orario di proiezione, numero di posti desiderato e indirizzo di posta elettronica su cui ricevere i biglietti elettronici.	20

2.4.3 Vincoli / Altri requisiti

ID	Requisito	Origine (n. frase dei requisiti revisionati)
V01	Ogni settimana di programmazione inizia lunedì e finisce domenica	
RNF01	Per l'invio mensile della newsletter, deve essere disponibile un server di posta elettronica esterno al sistema	

2.5 Modellazione dei casi d'uso

2.5.1 Attori e casi d'uso

Attori Primari:

- Cliente
- Direttore
- Tempo
- Impiegato botteghino

Attori Secondari:

- Servizio email

Casi d'uso:

- UC1: Definisci programmazione mensile
- UC2: Definisci orari proiezioni
- UC3: Definisci costo biglietto
- UC4: Visualizza programmazione mensile
- UC5: Cerca film
- UC6: Registrazione
- UC7: Acquista Biglietti
- UC8: Vendi Biglietti
- UC9: Invia locandina
- UC10: Genera Report
- UC11: GeneraReportMese
- UC12: GeneraReportFilm

Casi d' uso di inclusione:

- UC13: InserisciSalaFilm
- UC14: InserisciSettimanaFilm
- UC15: StampaBiglietti
- UC16: AssegnaPosti
- UC17: InviaBiglietti

Casi d' uso di estensione:

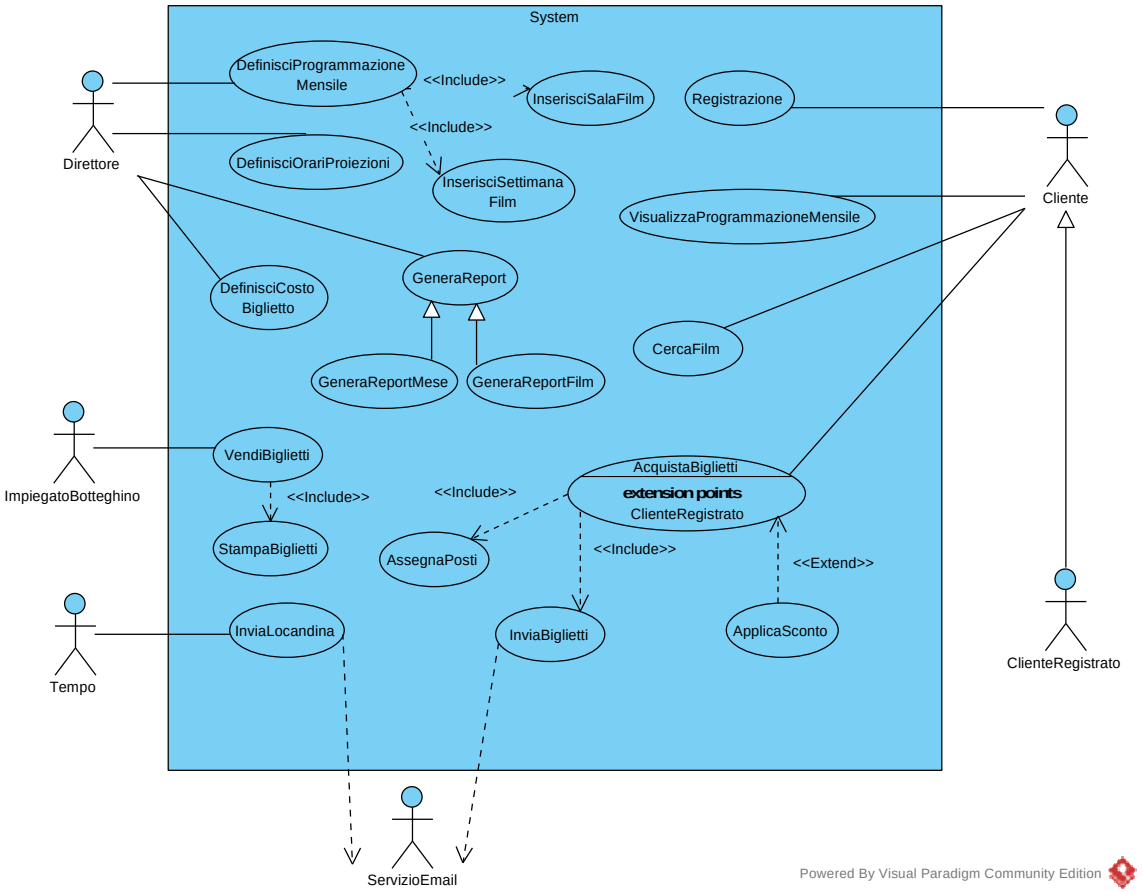
- UC18: ApplicaSconto

Caso d'uso	Attori Primari	Attori Secondari	Incl. / Ext.	Requisiti corrispondenti
UC1: DefinisciProgrammazione Mensile	Direttore	-	Include InserisciSalaFilm e InserisciSettimanaFilm	RF01
UC2: DefinisciOrari Proiezioni	Direttore	-	-	RF02
UC3: DefinisciCostoBiglietto	Direttore	-	-	RF03
UC4: Visualizza programmazione mensile	Cliente	-	-	RF04
UC5: Cerca film	Cliente	-	-	RF05, RF06
UC6: Registrazione	Cliente	-	-	RF07
UC7: AcquistaBiglietti	Cliente	-	Include AssegnaPosti e InviaBiglietti	RF08
UC8: VendiBiglietti	Impiegato botteghino	-	Include StampaBiglietti	RF12
UC9: InviaLocandina	Tempo	Servizio email	-	RF14
UC10: GeneraReport	Direttore	-	Generalizzazione di GeneraReportMese e GeneraReportFilm	RF15
UC11: GeneraReportMese	Direttore	-	-	RF15
UC12: GeneraReportFilm	Direttore	-	-	RF15
UC13: InserisciSalaFilm	Direttore	-	Incluso in DefinisciProgram mazioneMensile	RF01
UC14: InserisciSettimanaFilm	Direttore	-	Incluso in DefinisciProgram mazioneMensile	RF01
UC15: StampaBiglietti	Impiegato botteghino	-	Incluso in VendiBiglietti	RF13
UC16: AssegnaPosti	-	-	Incluso in AcquistaBiglietti	RF09

UC17: InviaBiglietti	-	ServizioEmail	Incluso in AcquistaBiglietti	RF11
UC18: ApplicaSconto	-	-	Estensione di AcquistaBiglietti	RF10

2.5.2 Diagramma dei casi d'uso

Riportare il diagramma dei casi d'uso.



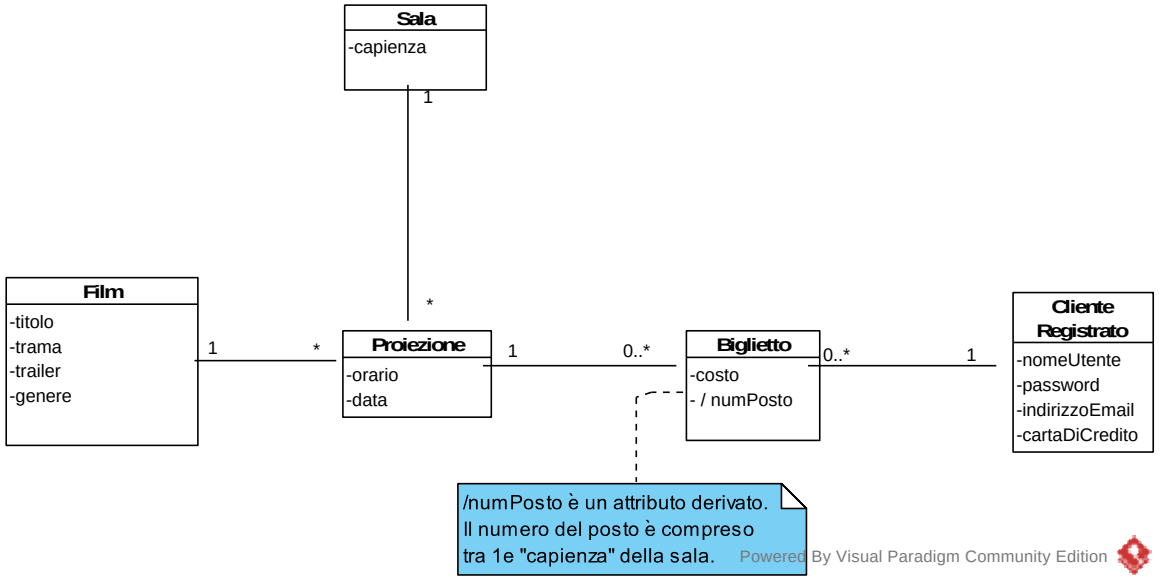
2.5.3 Scenari

Caso d'uso:	**AcquistaBiglietti**
Attore primario	Cliente

Attore secondario	-
Descrizione	Un cliente acquista uno o più biglietti per un film, in una data ad un orario
Pre-Condizioni	-
Sequenza di eventi principale	1. Il caso d'uso inizia quando il Cliente richiede l'acquisto di biglietti 1.1. Il Cliente inserisce film, data, orario, numero di biglietti da acquistare ed email. 2. Il sistema controlla che il film sia in programmazione per la data e orario selezionati 2.1. Se il film non è in programmazione il sistema restituisce un ERRORE al Cliente 3. Il sistema controlla che ci siano sufficienti posti liberi per la data e ora specificati. 3.1. Se non ci sono sufficienti posti liberi restituisce un ERRORE al Cliente 4. Il sistema controlla che la email del Cliente sia registrata 5. Il sistema calcola il prezzo per i biglietti, scontato se il Cliente è risultato registrato 6. Il Cliente paga il prezzo calcolato 7. Il sistema seleziona i numeri dei posti da riservare e genera i biglietti 8. Il sistema invia una email di conferma con i biglietti elettronici alla mail specificata in fase di acquisto
Post-Condizioni	Il Cliente ha acquistato uno o più biglietti che riceve in formato elettronico sulla email specificata
Casi d'uso correlati	*nessuno*
Sequenza di eventi alternativi	-

2.6 Diagramma delle classi

Diagramma delle classi di analisi.



Attenzione: Questo Class Diagram dovrà poi essere raffinato eseguendo l'assegnazione delle responsabilità alle classi. In particolare, alle classi Entity dovranno essere attribuite le responsabilità delle principali operazioni dedotte dall'analisi dei casi d'uso. Ulteriori classi potranno essere aggiunte, qualora alcune responsabilità non possano essere attribuite alle classi esistenti.

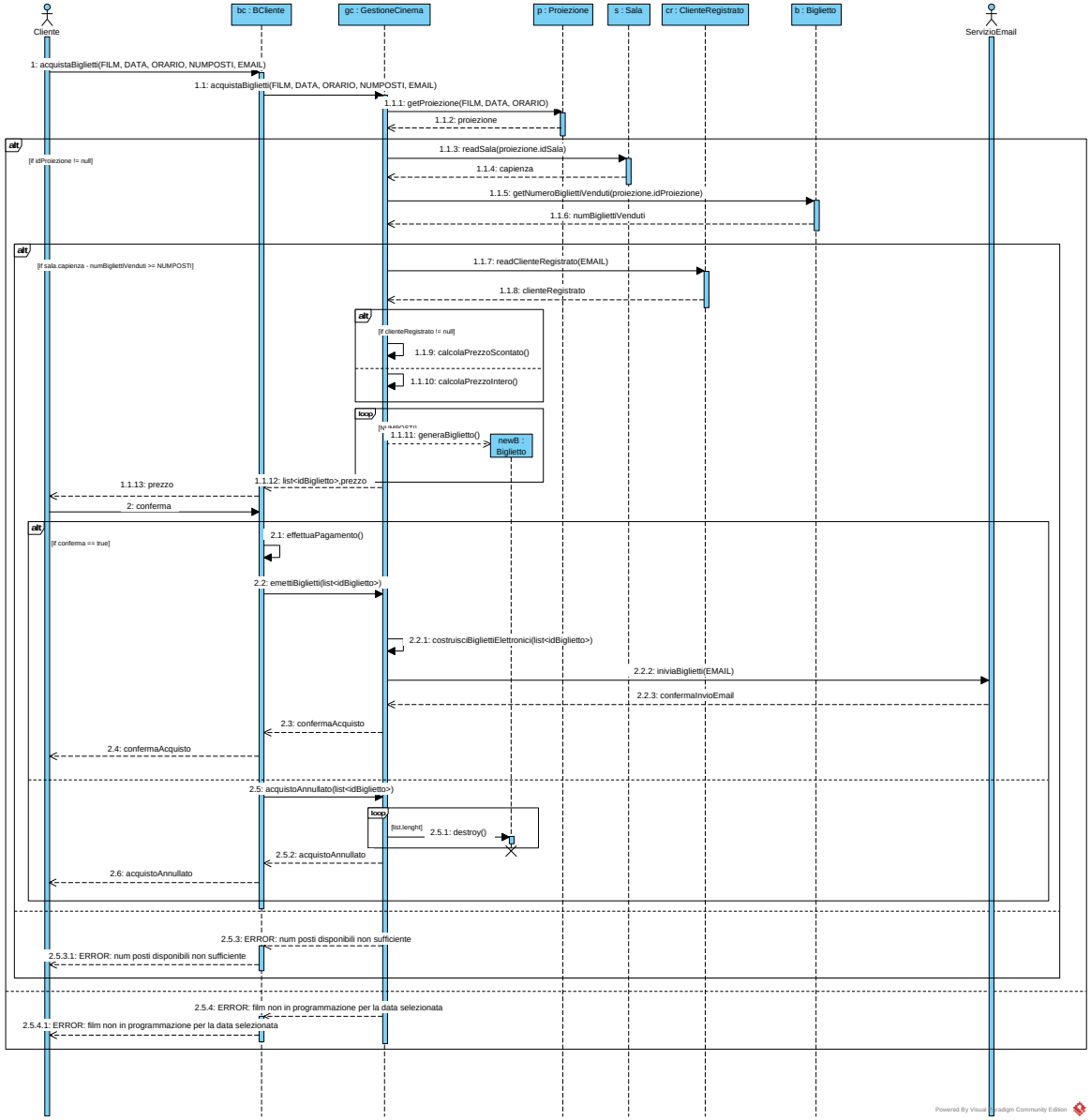
Ad esempio, la Proiezione potrà avere la responsabilità di vendita biglietto o di verificare se una Proiezione ha ancora posti disponibili; il Biglietto potrebbe avere la responsabilità di calcolare lo sconto sul prezzo. La Sala potrebbe avere la responsabilità di aggiungere Proiezioni di Film, il Film di cercare una Proiezione, etc.

Si consideri inoltre la possibilità di aggiungere una classe Cinema che abbia la responsabilità di aggiungere Film, cercare Film, registrare Clienti, etc..

Gli studenti alleghino anche questa versione raffinata del Class Diagram.

2.7 Diagrammi di sequenza

Diagrammi di sequenza di analisi per il caso d'uso AcquistaBiglietti.


Cliente
bc : BCliente
gc : GestioneCinema
p : Proiezione
s : Sala
cr : ClienteRegistrato
b : Biglietto
ServizioEmail
1: acquistaBiglietti(FILM, DATA, ORARIO, NUMPOSTI, EMAIL)
1.1: acquistaBiglietti(FILM, DATA, ORARIO, NUMPOSTI, EMAIL)
1.1.1: getProiezione(FILM, DATA, ORARIO)
1.1.2: proiezione
alt
[if idProiezione != null]
1.1.3: readSala(proiezione.idSala)
1.1.4: capienza
1.1.5: getNumeroBigliettiVenduti(proiezione.idProiezione)
1.1.6: numBigliettiVenduti
alt
[if sala.capienza - numBigliettiVenduti >= NUMPOSTI]
1.1.7: readClienteRegistrato(EMAIL)
1.1.8: clienteRegistrato
alt
[if clienteRegistrato != null]
1.1.9: calcolaPrezzoScontato()
1.1.10: calcolaPrezzoIntero()
loop
1.1.11: generaBiglietto()
newB : Biglietto
1.1.12: list<idBiglietto>,prezzo
1.1.13: prezzo
2: conferma
alt
[if conferma == true]
2.1: effettuaPagamento()
2.2: emettiBiglietti(list<idBiglietto>)
2.2.1: costruisciBigliettiElettronici(list<idBiglietto>)
2.2.2: iniviaBiglietti(EMAIL)
2.2.3: confermaInvioEmail
2.3: confermaAcquisto
2.4: confermaAcquisto
2.5: acquistoAnnullato(list<idBiglietto>)
loop
[list.lenght]
2.5.1: destroy()
2.5.2: acquistoAnnullato
2.6: acquistoAnnullato
2.5.3: ERROR: num posti disponibili non sufficiente
2.5.3.1: ERROR: num posti disponibili non sufficiente
2.5.4: ERROR: film non in programmazione per la data selezionata
2.5.4.1: ERROR: film non in programmazione per la data selezionata
Powered By Visual Paradigm Community Edition

2.8 Verifica della completezza dei requisiti

Si suggerisce allo studente di verificare che tutti i requisiti informali siano rappresentati nel modello UML e/o negli scenari. A tale scopo, lo studente può elencare i requisiti (funzionali, requisiti sui dati, altri requisiti) riportando per ciascun requisito gli elementi dei diagrammi UML con i quali è rappresentato nel modello di analisi. Dopo l'elencazione, lo studente verifichi che tutti i requisiti siano stati modellati nei diagrammi UML.

Legenda: UCD = Use Case Diagram, CD = Class Diagram, SD = Sequence Diagram
Es.:

- **RF01** è modellato nell'UCD con l'attore "Direttore" e con il caso d'uso UC1
- **RF02** è modellato nell'UCD con l'attore "Direttore" e con il caso d'uso UC2
- **RF03** è modellato nell'UCD con l'attore "Direttore" e con il caso d'uso UC3
- **RF04** è modellato nell'UCD con l'attore "Cliente" e con il caso d'uso UC4
- **RF05** è modellato nell'UCD con l'attore "Cliente" e con il caso d'uso UC5
- **RF06** è modellato nell'UCD con l'attore "Cliente" e con il caso d'uso UC5
- **RF07** è modellato nell'UCD con l'attore "Cliente" e con il caso d'uso UC6
- **RF08** è modellato nell'UCD con l'attore "Cliente" e con il caso d'uso UC7
- **RF09** è modellato nell'UCD con il caso d'uso UC16
- **RF10** è modellato nell'UCD con il caso d'uso UC18
- **RF11** è modellato nell'UCD con il caso d'uso UC17
- **RF12** è modellato nell'UCD con l'attore "Impiegato botteghino" e il caso d'uso UC8

- **RF13** è modellato nell'UCD con l'attore "Impiegato botteghino" e il caso d'uso UC15
- **RF14** è modellato nell'UCD con gli attori "Tempo"(primario), "Servizio email"(secondario) e con il caso d'uso UC9
- **RF15** è modellato nell'UCD con l'attore "Direttore" e i casi d'uso UC10, UC11, UC12
- **RD01** è modellato nel CD con la classe "Sala"
- **RD02** è modellato nel CD con l'attributo "capienza" della classe "Sala"
- **RD03** è modellato nel CD con l'attributo derivato \numPosto della classe "Biglietto"
- **RD04, RD05, RD06** sono modellati nel CD con gli attributi "data" e "orario" della classe "Proiezione"
- **RD07** è modellato nel CD con gli attributi della classe "Film"
- **RD08** è modellato nel CD con la classe "Proiezione" (da cui è possibile ottenere i dati)
- **RD09** è modellato nel CD con l'attributo "costo" della classe "Biglietto"
- **RD10** è modellato nel CD con gli attributi della classe "ClienteRegistrato"
- **RD11** è modellato nel SD con i parametri della funzione "AcquistaBiglietti"

3. Piano di test funzionale

Progettare i casi di test funzionale con la tecnica del *Category Partition Testing*. Descrivere il procedimento di calcolo.

PIANO DI TEST UTILIZZANDO IL METODO DEL *CATEGORY-PARTITION TESTING* PER LA FUNZIONALITÀ "*AcquistaBiglietti*".

FILM	DATA	ORA	EMAIL	NUMEROPOSTI
• Stringa di caratteri di lunghezza <= 100 • Stringa di caratteri di lunghezza > 100 [ERROR] • Stringa che contiene simboli che non sono caratteri [ERROR]	• Data con formato valido(gg/mm/aaaa) • Data con formato non valido [ERROR]	• Ora con formato valido(hh:mm) • Ora con formato non valido [ERROR]	• Stringa in cui è presente il simbolo @ • Stringa in cui non è presente il simbolo @ [ERROR]	• Numero intero > 0 • Numero intero <=0 [ERROR]

Il numero di test da effettuarsi senza particolari vincoli è: $3 * 2 * 2 * 2 * 2 = 48$
Introduciamo i vincoli [ERROR] .
Il numero di test da eseguire per testare singolarmente i vincoli è 6 (2 per Film, 1 per Data, 1 per Ora, 1 per Email, 1 per NumeroPosti).
Il numero di test risultante è: (1*1*1*1*1) + 6 = 7

TEST SUITE

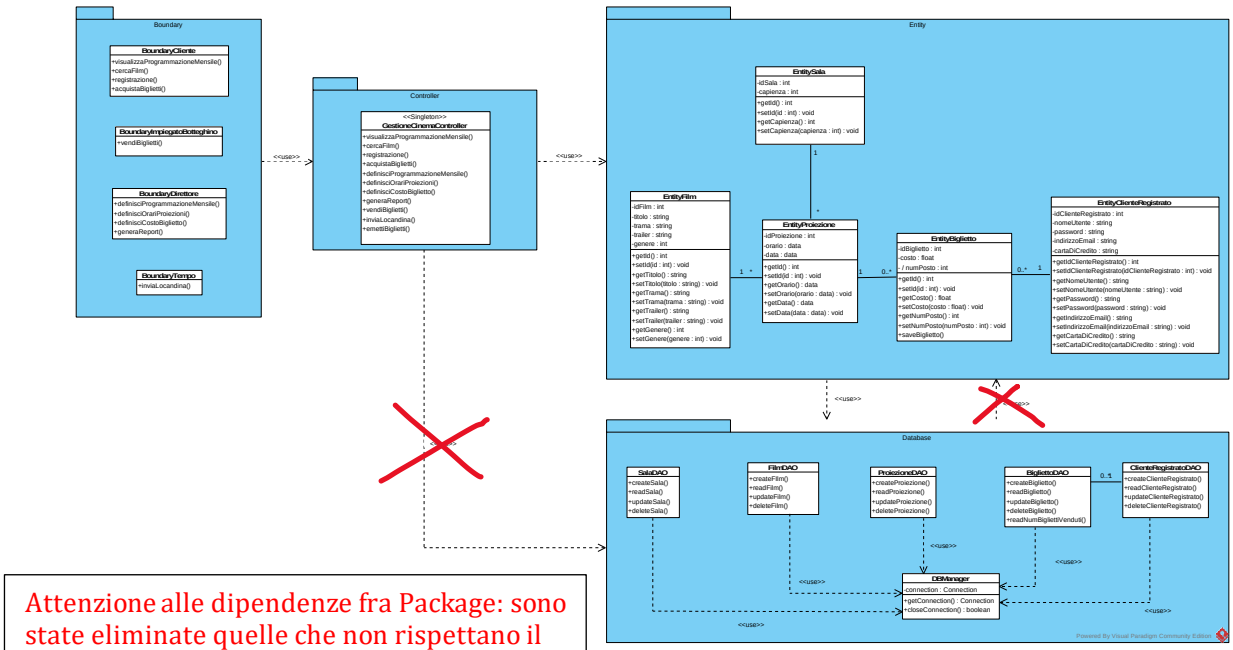
Test Case ID	Descrizione	Classi di equivalenza coperte	Pre-condizioni	Input	Output Attesi	Post-condizioni Attese
1	Tutti input validi	Film valido Data valida Ora valida Email valida NumeroPosti valido	Il film è presente nella programmazione, per la data e ora selezionata. C’è disponibilità per il numero di posti selezionato	{Film: “Il Gladiatore”, Data: “25/12/2021”, Ora: “18:30”, Email: mariorossi@gmail.com,NumeroPosti: 2}	Biglietti prenotati	Si ricevono i biglietti per email
2	Film stringa > 100 caratteri	Film stringa > 100 caratteri [ERROR], Data, Ora, Email, NumeroPosti validi		{Film: “aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa”, Data: “25/12/2021”, Ora: “18:30”, Email: mariorossi@gmail.com,NumeroPosti: 2}	Titolo film troppo lungo!	
3	Film stringa con simboli	Film stringa con simboli [ERROR], Data, Ora, Email, NumeroPosti validi		{Film: “§ç£”, Data: “25/12/2021”, Ora: “18:30”, Email: mariorossi@gmail.com,NumeroPosti: 2}	Titolo film errato!	

4	Data formato non valido	Film valido, formato data non valido [ERROR], Ora, Email, NumeroPosti validi		{Film: “Il Gladiatore”, Data: “225/1332/2021”, Ora: “18:30”, Email: mariorossi@gmail.com,NumeroPosti: 2}	Data non valida!	
5	Ora formato non valido	Film, Data validi, formato ora non valido [ERROR], Email, NumeroPosti validi		{Film: “Il Gladiatore”, Data: “25/12/2021”, Ora: “183:3440”, Email: mariorossi@gmail.com,NumeroPosti: 2}	Orario non valido!	
6	Email senza simbolo @	Film, Data, Ora, validi, Email senza simbolo @ [ERROR], NumeroPosti validi		{Film: “Il Gladiatore”, Data: “25/12/2021”, Ora: “18:30”, Email: mariorossigmail.com,NumeroPosti: 2}	Email non valida!	
7	NumeroPosti intero <=0	Film valido Data valida Ora valida Email valida NumeroPosti <= 0 non valido [ERROR]		{Film: “Il Gladiatore”, Data: “25/12/2021”, Ora: “18:30”, Email: mariorossi@gmail.com,NumeroPosti: 0}	Numero posti non valido!	

4. Progettazione

4.1 Diagramma delle classi

Riportare il diagramma delle classi di progettazione. Reificare eventuali classi associative del diagramma delle classi di analisi. Specificare argomenti e tipo di ritorno delle operazioni (per quelle più significative, coinvolte nei casi d'uso sviluppati fino alla implementazione).
Includere classi del dominio della soluzione, come strutture dati e classi DAO. Raggruppare le classi in package.

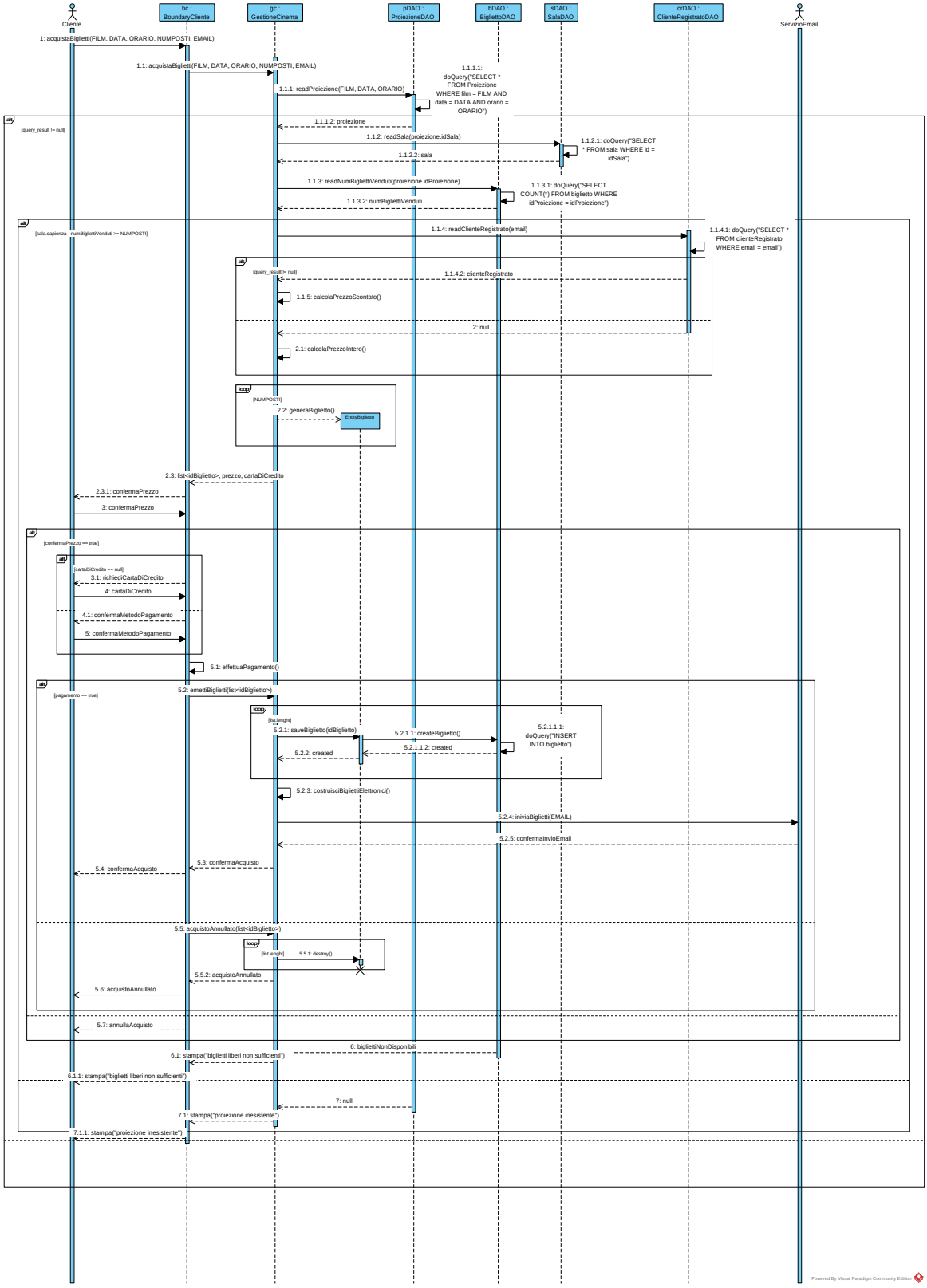


Attenzione alle dipendenze fra Package: sono state eliminate quelle che non rispettano il Pattern Architetturale a Livelli

4.2 Diagrammi di sequenza

Riportare i diagrammi di sequenza di progetto per il/i casi d'uso sviluppati fino alla codifica in Java.

Attenzione questo diagramma non è coerente col Pattern Architetturale a Livelli: per rispettare tale pattern architetturale, il controller GestioneCinema dovrà interagire prima con opportuni oggetti del Package Entity livelli e poi questi a loro volta interagiranno con oggetti DAO.
Gli studenti tengano conto di questo commento nel produrre i loro Sequence Diagrams di Progetto



5. Implementazione

Non includere il codice sorgente, ma descrivere l'implementazione in Java, descrivendo gli artefatti di codifica:

- Elencare:
  - package, classi, tipi di eccezione definiti
- Elencare gli artefatti necessari per l'installazione ed esecuzione del programma, senza ovviamente l'ambiente di sviluppo come Eclipse (DB h2, eventuali librerie e versioni di Java che l'utilizzatore deve avere installati, file .class, .jar, …)
- Produrre un eventuale diagramma di deployment
- Eventualmente inserire la documentazione del codice prodotta con Javadoc (relativamente alle funzionalità implementate)
- Riportare il numero di LOC e di LLOC scritte in Java
- Confrontare con la stima dei costi effettuata e commentare eventuali scostamenti

6. Testing

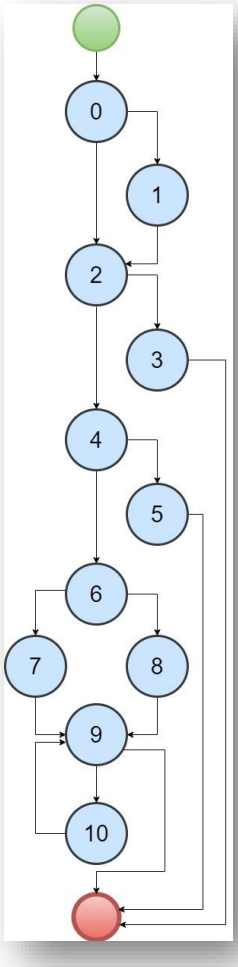
6.1 Test strutturale

6.1.1 Complessità ciclomatica

Costruire il Control Flow Graph per uno o due dei metodi delle classi implementate (si scelgano metodi non proprio banali), e:
- si mostri il calcolo del numero ciclomatico;
- si indichino i percorsi linearmente indipendenti;
Prima o a fianco del CFG riportare il codice Java del metodo.
Es.:

```
public static boolean CalcolaStatistica(String stringaData, int CAP, boolean scelta) {
	if(scelta)
		stringaData = "01/"+stringaData;
	LocalDate data = null;
	DateTimeFormatter dateTimeFormatter = DateTimeFormatter.ofPattern("dd/MM/yyyy");
	try {
		if(!(MonitoraggioAmbiente.getInstance().VerificaCAP(CAP)))  { //Verifica CAP
				System.err.println("CAP  non valido");
				return false;
		}
	}
	catch(IOException e) {
		System.err.println("Errore  di I/O durante il controllo del CAP" + e.getMessage());
		return false;
	}
	try {
		data = LocalDate.parse(stringaData, dateTimeFormatter);		//Dalla sequenza data.parse() != null
		if(data==null) {						//Verifica DATA
				System.err.println("Data non valida");
				return false;
		}
	}catch(DateTimeParseException e) {
		System.err.println("Data non valida");
		return false;
	}
	java.util.Map<String, Float> Dati = null;
	try {
		if(scelta) {
				Dati = MonitoraggioAmbiente.getInstance().CalcolaStatisticaMese(CAP, data);
				System.out.println("Statistica del mese inserito:\n");
		}
		else {
				Dati = MonitoraggioAmbiente.getInstance().CalcolaStatisticaGiorno(CAP, data);
				System.out.println("Statistica del giorno inserito:\n");
		}
		for(String key:Dati.keySet()) {
				System.out.println(key +":"+(key.length()<14?"\t\t":"\t") + Dati.get(key));	//Output
		}
	} catch (StatisticaException e) {
		System.err.println(e.getMessage());			return false;
	} catch (MisuraException e) {
		System.err.println(e.getMessage());			return false;
	}
	return true;
  }
}
```

Control Flow Graph



NUMERO CICLOMATICO:
numero di regioni chiuse del grafo = 6
numero di nodi predicati (0,2,4,6,10) +1 = 6
archi – # nodi + 2 = (15 - 11) + 2 = 6

CAMMINI:
1) 0-1-2-3
2) 0-1-2-4-5
3) 0-1-2-4-6-7-9
4) 0-1-2-4-6-8-9
5) 0-1-2-4-6-7-9-10-9
6) 0-2-4-6-7-9

6.2 Test funzionale

Descrivere i risultati dell'esecuzione dei test funzionali precedentemente pianificati adoperando lo schema di tabella seguente.
Descrivere le eventuali azioni di *debugging* a seguito di casi di test con esito *FAIL*.
Commentare se gli eventuali difetti rilevati dal test funzionale potevano essere rilevati anche da un test strutturale.

Test Case ID	Descrizione	Classi di equivalenza coperte	Pre-condizioni	Input	Output Attesi	Post-condizioni Attese	Output Ottenuti	Post-condizioni Ottenute	Esito (*FAIL*, *PASS*)