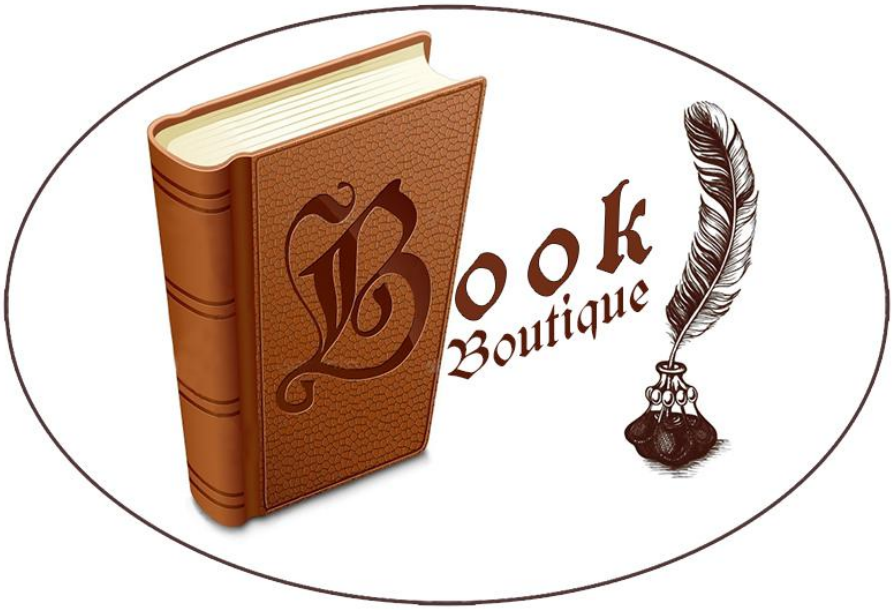
Corso di Laurea Magistrale in Ingegneria Informatica - A. A. 2014/2015

Ingegneria del Software

BO.BO.

Book Boutique



Studenti:

Loria Mark Philip

Spada Mariacarmela

Prefazione

Il seguente documento presenta una descrizione relativa allo sviluppo dell'applicazione Book Boutique implementata durante il corso di Ingegneria del Software. Il software è stato implementato nel linguaggio Java sfruttando l'ambente di sviluppo Eclipse, mentre la progettazione in UML è stata realizzata con il software Astah Protessional. Verranno qui presentate solamente le versioni finali di ciascun elaborato ottenute al termine di tutte le fasi di progettazione (per visionarie le versioni intermedie si vedano i documenti nelle corrispondenti sottocartelle della documentazione). Infine vengono descritte le fasi conclusive di testing e di refactoring del programma, con cui sostanzialmente sono state apportate delle migliorie al codice ed è stato aggiunto un Database per gestire la persistenza dei dati.

Indice

1 Ideazione e analisi dei requisiti

1.1 Introduzione

Lo scopo della fase di ideazione è quello di effettuare un'indagine col fine di ricavare un'idea generale del progetto da realizzare e quindi ottenere delle stime di fattibilità tenendo conto dei tempi e delle risorse necessari.

Per capire al meglio come sviluppare e comprendere i requisiti del progetto, nella fase di ideazione sono stati presi in considerazione i seguenti documenti: Modello dei Casi d'Uso, Documento di Visione, Specifiche Supplementari, Regole di Business e Glossario.

1.2 Requisiti

Il titolare di una libreria richiede la realizzazione di un software che permetta di migliorare la gestione delle vendite di libri, ed in particolare di testi scolastici, ai propri clienti. Il software deve rappresentare uno strumento integrato che segua il processo di vendita in tutte le sue fasi, a partire dalla fornitura dei libri fino alla vendita degli stessi ai clienti. In particolare:

- ✓ Il libraio deve poter gestire l'anagrafica dei propri clienti e tenere traccia degli acquisti da essi effettuati.
- ✓ Il libraio deve poter consultare e modificare un catalogo di tutti i libri che è possibile acquistare presso la sua libreria (anche se non presenti in magazzino).
- ✓ Il libraio deve poter consultare e modificare un archivio delle adozioni scolastiche delle più importanti scuole locali.
- ✓ All'arrivo in libreria, i clienti presentano al libraio la lista dei libri che desiderano acquistare o in alternativa specificano la scuola e la classe per cui vogliono acquistare i libri di testo; il sistema dovrà verificare l'effettiva esistenza dei libri nel catalogo e la relativa disponibilità in magazzino. L'esito della ricerca viene comunicato al cliente, il quale potrà scegliere tra:
    - *a.* se il testo è presente, procedere con l'acquisto e il ritiro
    - *b.* se il testo non è presente, procedere con la prenotazione e il pagamento
    - *c.* rifiutarsi sia di acquistare che di prenotare il testo e andare via
- ✓ In entrambi i casi *a.* e *b.* viene creata una nuova scheda di vendita relativa al/ai testo/i richiesto/i dal cliente; il sistema calcolerà il corrispettivo da pagare dopo aver applicato gli sconti dovuti ad eventuali promozioni in corso e lo mostra a video. Se il cliente conferma l'interesse ad acquistare, il sistema inoltrerà alla cassa le informazioni contenute nella scheda di vendita e l'acquisto si concluderà con il pagamento da parte del cliente.

- ✓ Nel caso *a.* dopo aver pagato il cliente potrà ritirare immediatamente i libri acquistati; il libraio quindi aggiornerà la scheda di vendita corrispondente specificando che i libri sono stati consegnati al cliente.
- ✓ Nel caso *b.*: dopo aver pagato il cliente va via e resterà in attesa di una comunicazione da parte del libraio per notificare l'arrivo del/dei testo/i prenotato/i e quindi la possibilità di ritirarlo/i.
- ✓ Il libraio deve poter creare degli ordini per i distributori tenendo principalmente conto delle copie prenotate dai clienti ma non presenti in magazzino, ma anche scegliendoli arbitrariamente dal catalogo. Nota: la creazione periodica di questi ordini potrebbe anche essere fatta automaticamente dal sistema.
- ✓ All'arrivo delle copie ordinate ai distributori, il libraio deve poter aggiornare le giacenze del magazzino, controllare le prenotazioni dei clienti, preparare i libri da consegnare (e quindi riaggiornare le copie residue) e inviare loro una comunicazione (via email o SMS) per informarli dell'arrivo dei libri.
- ✓ Il libraio deve poter creare con flessibilità delle promozioni. Durante la vendita dei libri, prima di inoltrare il riepilogo alla cassa, il sistema deve applicare eventuali sconti.

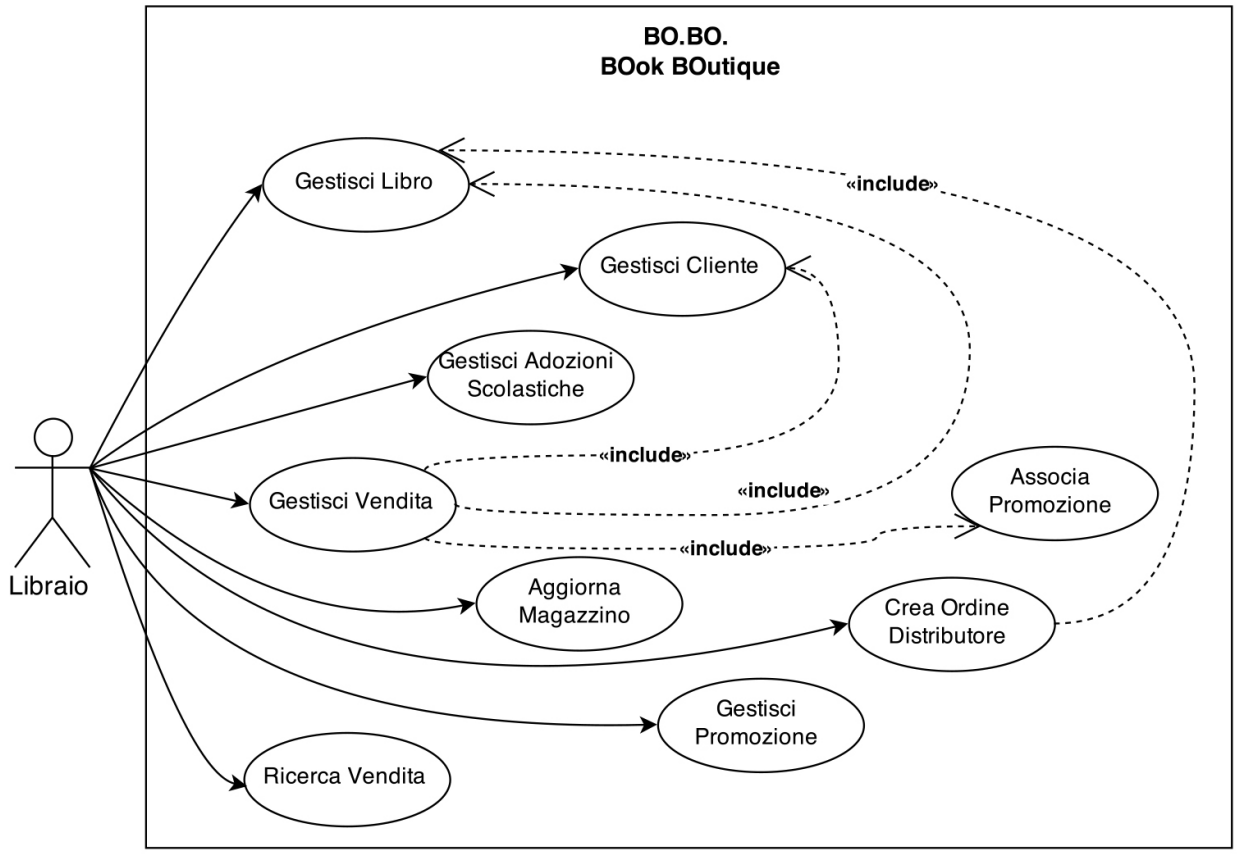
1.3 Obiettivi e casi d'uso

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

Attore	**Obiettivo**	**Caso d'uso**
Libraio	Gestire la creazione di una nuova vendita di libri a un cliente comprensiva di un'eventuale prenotazione dei testi mancanti e del calcolo delle promozioni.	UC1: Gestisci Vendita
Libraio	Creare un ordine verso un distributore per ordinare le copie richieste dai clienti e/o alcune destinate al magazzino	UC2: Crea Ordine Distributore
Libraio	Inserire, rimuovere, ricercare o modificare un libro nel catalogo (compresa operazione di aggiornamento della giacenza per un particolare libro)	UC3: Gestisci Libro (CRUD, *Ricerca Libro*)
Libraio	Gestire i dati relativi a ciascun cliente (inserisci, modifica, rimuovi, ricerca)	UC4: Gestisci Cliente (CRUD, *Ricerca Cliente)*

Libraio	Ricercare, inserire, cancellare gli elenchi dei libri adottati nelle scuole locali	UC5: Gestisci Adozioni Scolastiche (CRUD, *Ricerca Lista*)
Libraio	Verificare tutte le prenotazioni in sospeso e controllare se vi sono copie disponibili in magazzino. Eventualmente, inviare la comunicazione al cliente.	UC6: Aggiorna Magazzino
Libraio	Ricercare una vendita specificando il numero identificativo o in alternativa ricercare tutte le vendite relative a un cliente (specificando cognome o numero identificativo del cliente).	UC7: Ricerca Vendita
Libraio	Creare, modificare o rimuovere una promozione	UC8: Gestisci Promozione (CRUD, *Crea Promozione*)
Libraio	Verificare se bisogna associare una promozione a una vendita	UC9: Associa Promozione

I casi d'uso appena descritti si traducono graficamente nel seguente diagramma UML:



1.4 Modello dei casi d'uso

Viene qui presentata una descrizione dei casi d'uso; in particolare, partendo dalla fase di ideazione in cui la maggior parte dei casi d'uso erano descritti in formato breve, seguendo il processo iterativo si è passati gradualmente ad una descrizione via via più dettagliata per la maggior parte dei casi d'uso, dunque si ha:

UC1: Gestisci vendita

Nome del caso d'uso	**UC1: Gestisci vendita**
Portata	Applicazione BO.BO.
Livello	Obiettivo utente
Attore primario	Libraio
Parti interessate e Interessi	● Libraio: vuole gestire l'intero processo di vendita in maniera corretta e veloce; vuole che le informazioni relative agli acquisti e alle prenotazioni dei clienti siano aggiornate. ● Cliente: vuole acquistare i libri usufruendo di un servizio rapido e semplice.
Pre-condizioni	Il cliente conosce i libri che desidera acquistare o la classe di appartenenza dello studente per cui acquistare i libri
Garanzia di successo	La vendita si conclude con l'acquisto (pagamento e rilascio della ricevuta al cliente alla cassa) ed eventualmente con la consegna dei libri al cliente. Le giacenze vengono aggiornate.
Scenario principale di successo	1. Un cliente arriva in libreria e richiede al libraio un libro da acquistare. 2. Il libraio utilizza il software BO.BO. per cercare nel catalogo il libro richiesto *[vedi caso d'uso Gestisci Libro].* Il sistema mostra l'esito positivo della ricerca e le giacenze residue, quindi il libraio lo comunica al cliente. 3. Il cliente intende acquistare il libro. 4. Il libraio utilizzando BO.BO. aggiunge il libro ricercato alla scheda di acquisto. 5. Ripetere i passi 2, 3 e 4 finchè ci sono altri libri che il cliente desidera acquistare.

	6. Il cliente comunica di voler terminare l'acquisto e indica i propri dati al libraio. 7. Il libraio ricerca il cliente nell'archivio interno *[vedi caso d'uso Gestisci Cliente]*; il software restituisce il risultato richiesto (cliente già presente nell'archivio) e lo associa alla scheda di acquisto. 8. Il libraio conferma l'operazione. 9. Il sistema calcola il totale dell'acquisto, applica eventuali sconti e promozioni *[vedi caso d'uso UC9: Associa Promozione]* e mostra a video il risultato. 10. Il libraio comunica al cliente l'importo totale dell'ordine e che i libri sono tutti presenti in magazzino. 11. Il cliente decide di proseguire con l'acquisto, il libraio conferma l'acquisto e il sistema inoltra alla cassa le informazioni contenute nella scheda di vendita. 12. Il cliente si reca alla cassa, paga, ritira la ricevuta e il/i libro/i e va via. 13. Per ciascun testo ritirato dal cliente, il sistema aggiorna le copie residue in magazzino.
Estensioni	***a.** In qualsiasi momento, il sistema fallisce e ha un arresto improvviso. 1. Il libraio riavvia il software e richiede il ripristino dello stato precedente del sistema. 2. Il sistema ricostruisce lo stato precedente. **1a.** Il cliente indica al libraio la classe/scuola per cui intende acquistare i libri di testo. 1. Il libraio ricerca la classe e la scuola richiesta dal cliente *[vedi caso d'uso Gestisci Adozioni Scolastiche]*. 2. Il sistema restituisce l'elenco dei libri di testo relativi alla classe specificata. **2a.** Il libro non viene trovato nel catalogo. 1. Il libraio comunica al cliente che il libro richiesto è invendibile. **3a.** Il cliente non vuole proseguire con l'acquisto e va via. 1. Il libraio termina l'esecuzione del programma. **7a.** Il cliente ricercato non viene trovato nell'archivio. 1. Il sistema chiede al libraio di inserire il nuovo cliente nell'archivio prima di proseguire.

	2. Il libraio inserisce i dati e conferma *[vedi caso d'uso Gestisci Cliente]*. 3. Il sistema associa il nuovo cliente alla vendita in corso. **10a.** Il libraio comunica al cliente l'importo totale dell'ordine e che non tutti i libri sono tutti presenti in magazzino. 1. Il cliente decide di proseguire con l'acquisto e di prenotare le copie mancanti. 2. Il libraio contrassegna i libri mancanti da prenotare e conferma l'acquisto. 3. Il sistema inoltra alla cassa le informazioni contenute nella scheda di vendita. 4. Il cliente si reca alla cassa, paga, ritira la ricevuta e il/i libro/ie che è possibile ritirare; infine va via e resta in attesa di una comunicazione da parte del libraio per notificare l'arrivo del/dei testo/i prenotato/i. **11a.** Il cliente non vuole proseguire con l'acquisto e va via. 1. Il libraio annulla la vendita appena creata.
Requisiti speciali	
Elenco delle varianti tecnologiche e dei dati	
Frequenza di ripetizioni	Legata all'affluenza dei clienti e agli acquisti che intendono effettuare.
Varie	

UC2: Crea Ordine Distributore

Nome del caso d'uso	**UC2: Crea Ordine Distributore**
Portata	Applicazione BO.BO.
Livello	Obiettivo utente
Attore primario	Libraio
Parti interessate e Interessi	• Libraio: vuole assicurare una corretta rifornitura delle giacenze in magazzino; inoltre vuole ricevere assistenza nella creazione degli ordini relativi ai testi già prenotati dai clienti. Nella

	creazione degli ordini desidera un inserimento dei dati preciso, rapido e senza errori. ● Distributore: vuole ricevere gli ordini in un formato preciso e semplice che consenta una visualizzazione chiara delle informazioni relative alle copie ordinate dal libraio.
Pre-condizioni	Libri consegnati per ordini precedenti sono stati correttamente assegnati ad eventuali prenotazioni attive.
Garanzia di successo	Viene salvato il nuovo ordine nell'archivio; viene esportato un documento PDF con i dettagli relativi all'ordine per essere successivamente inoltrato al distributore.
Scenario principale di successo	1. Il libraio richiede al sistema la creazione di un nuovo ordine di fornitura. 2. Il sistema chiede l'inserimento della casa editrice; il libraio inserisce l'informazione e conferma. 3. Il sistema mostra l'elenco delle prenotazioni di tutti i clienti relative alla casa editrice specificata. 4. Il libraio conferma l'inserimento di tali prenotazioni nel nuovo ordine. 5. Il sistema chiede al libraio se vuole aggiungere manualmente testi all'ordine. 6. Il libraio rifiuta e conferma la creazione del nuovo ordine.
Estensioni	***a.** In qualsiasi momento, il sistema fallisce e ha un arresto improvviso. 1. Il libraio riavvia il software e richiede il ripristino dello stato precedente del sistema. 2. Il sistema ricostruisce lo stato precedente. **3a.** Il sistema comunica al libraio che non è stata trovata alcuna prenotazione per la casa editrice specificata. 1. Lo scenario riprende dal passo 6. **6a.** Il libraio accetta l'inserimento di altri testi all'ordine. 1. Il libraio ricerca il libro da inserire *[vedi caso d'uso Gestisci Libro]* e specifica la quantità delle copie da ordinare. 2. Finchè ci sono copie da insierire nell'ordine si ripete il punto 1. 3. Il libraio conferma il completamento dell'operazione e la creazione del nuovo ordine. **6b.** Il libraio vuole annullare l'intera operazione. 1. Il libraio chiede al sistema di annullare l'operazione in corso.

	2. Il sistema mostra la conferma dell'annullamento.
Requisiti speciali	Non specificati.
Elenco delle varianti tecnologiche e dei dati	Per ciascun libro inserito nell'ordine devono essere visualizzati i campi: ISBN, titolo, autore, prezzo, quantità.
Frequenza di ripetizione	Solitamente gli ordini vengono inviati settimanalmente per ciascun fornitore.
Problemi Aperti	Il sistema dovrebbe provvedere automaticamente alla creazione di un ordine settimanale se non lo crea esplicitamente il libraio?

UC3: Gestisci libro, CRUD

Nome del caso d'uso	**UC3: Gestisci Libro**
Portata	Applicazione BO.BO.
Livello	Obiettivo utente, CRUD
Attore primario	Libraio
Parti interessate e Interessi	• Libraio: vuole gestire (creare, cercare, modificare, cancellare e visualizzare) i libri del catalogo in maniera semplice e corretta.
Pre-condizioni	
Garanzia di successo	Consistenza dei dati in output.
Scenario principale di successo	1. Il libraio richiede al sistema l'**inserimento** in catalogo di un nuovo libro. 2. Il sistema richiede al libraio le informazioni relative al libro. 3. Il libraio inserisce codice ISBN, autori, titolo, editore, prezzo, copie disponibili e descrizione.
Estensioni	***a.** In qualsiasi momento, il sistema fallisce e ha un arresto improvviso. 1. Il libraio riavvia il software e richiede il ripristino dello stato precedente del sistema. 2. Il sistema ricostruisce lo stato precedente. **1a.** Il libraio richiede al sistema la **modifica** di un libro in catalogo.

	1. Il sistema richiede al libraio i dati aggiornati. 2. Il libraio aggiorna i campi di interesse (codice ISBN, autori, titolo, prezzo, copie disponibili e descrizione) e conferma. **1b.** Il libraio richiede al sistema di **ricercare** un libro in catalogo 1. Il sistema richiede al libraio di inserire le informazioni nei campi di ricerca (ISBN, titolo, autori, ecc.). 2. Il libraio inserisce le informazioni a sua disposizione. 3. Il sistema mostra l'elenco di libri che soddisfano i criteri di ricerca. **1c.** Il libraio richiede al sistema di **eliminare** un libro dal catalogo. 1. Il sistema chiede al libraio di confermare l'eliminazione. 2. Il libraio conferma. 3. Il sistema elimina il libro dal catalogo.
Requisiti speciali	Non specificati.
Elenco delle varianti tecnologiche e dei dati	
Frequenza di ripetizione	La gestione dei libri è l'operazione che viene svolta più frequentemente dal libraio.
Problemi Aperti	Il sistema dovrebbe prevedere la gestione dei libri sfruttando un lettore di codici a barre?

UC4: Gestisci cliente, CRUD

Nome del caso d'uso	**UC4: Gestisci Cliente**
Portata	Applicazione BO.BO.
Livello	Obiettivo utente, CRUD
Attore primario	Libraio
Parti interessate e Interessi	• Libraio: vuole gestire (aggiungere, cercare, modificare, cancellare e visualizzare) i clienti in maniera semplice e corretta.
Pre-condizioni	
Garanzia di successo	Consistenza dei dati in output.

Scenario principale di successo	1. Il libraio richiede al sistema di **inserire** un nuovo cliente. 2. Il sistema richiede al libraio le informazioni relative al cliente. 3. Il libraio inserisce i dati del cliente.
Estensioni	***a.** In qualsiasi momento, il sistema fallisce e ha un arresto improvviso. 3. Il libraio riavvia il software e richiede il ripristino dello stato precedente del sistema. 4. Il sistema ricostruisce lo stato precedente. **1a.** Il libraio richiede al sistema la **modifica** di un cliente. 1. Il sistema richiede al libraio i dati aggiornati. 2. Il libraio aggiorna i campi di interesse e conferma. **1b.** Il libraio richiede al sistema di **ricercare** un cliente 1. Il sistema richiede al libraio di inserire le informazioni nei campi di ricerca. 2. Il libraio inserisce le informazioni a sua disposizione. 3. Il sistema mostra l'elenco dei clienti che soddisfano i criteri di ricerca. **1c.** Il libraio richiede al sistema di **eliminare** un cliente. 1. Il sistema chiede al libraio di confermare l'eliminazione. 2. Il libraio conferma. 3. Il sistema elimina il cliente.
Requisiti speciali	Non specificati.
Elenco delle varianti tecnologiche e dei dati	
Frequenza di ripetizione	Probabilmente alta, dipende dal numero di acquisti e di prenotazioni presenti.
Problemi Aperti	

UC5: Gestisci adozioni scolastiche, CRUD

Nome del caso d'uso	**UC5: Gestisci Adozioni Scolastiche**
Portata	Applicazione BO.BO.
Livello	Obiettivo utente, CRUD

Attore primario	Libraio
Parti interessate e Interessi	• Libraio: vuole gestire (aggiungere, cercare, modificare, cancellare e visualizzare) le liste dei libri adottate nelle scuole in maniera semplice e corretta.
Pre-condizioni	
Garanzia di successo	Consistenza dei dati in output.
Scenario principale di successo	1. Il libraio richiede al sistema di **inserire** una nuova lista di adozione. 2. Il sistema richiede al libraio la scuola e la classe. 3. Il libraio inserisce la scuola e la classe. 4. Il sistema chiede al libraio il codice ISBN del libro da associare alla lista. 5. Il libraio inserisce il codice ISBN. 6. Il libraio ripete il passo 5 fino a che indica che ha terminato.
Estensioni	***a.** In qualsiasi momento, il sistema fallisce e ha un arresto improvviso. 1. Il libraio riavvia il software e richiede il ripristino dello stato precedente del sistema. 2. Il sistema ricostruisce lo stato precedente. **1a.** Il libraio richiede al sistema la **modifica** di una lista di adozione. 1. Il sistema richiede al libraio i dati aggiornati. 2. Il libraio aggiorna i campi di interesse e conferma. **1b.** Il libraio richiede al sistema di **ricercare** una lista di adozione 1. Il sistema richiede al libraio di inserire le informazioni nei campi di ricerca. 2. Il libraio inserisce le informazioni a sua disposizione. 3. Il sistema mostra l'elenco delle liste di adozione che soddisfano i criteri di ricerca. **1c.** Il libraio richiede al sistema di **eliminare** una lista di adozione 1. Il sistema chiede al libraio di confermare l'eliminazione. 2. Il libraio conferma. 3. Il sistema elimina la lista di adozione
Requisiti speciali	Non specificati.

Elenco delle varianti tecnologiche e dei dati	
Frequenza di ripetizione	Bassa, si prevede che difficilmente il cliente non porta con sè la lista dei libri da acquistare.
Problemi Aperti	

UC6: Aggiorna Magazzino

Nome del caso d'uso	**UC6: Aggiorna Magazzino**
Portata	Applicazione BO.BO.
Livello	Obiettivo utente
Attore primario	Libraio
Parti interessate e Interessi	• Libraio: vuole associare correttamente i libri pervenuti alle prenotazioni attive dei clienti; vuole informare rapidamente i clienti sull'arrivo dei libri prenotati e pronti per la consegna. • Cliente: desidera recarsi in libreria con la certezza di trovare i libri precedentemente prenotati.
Pre-condizioni	Arrivo in libreria di libri da inserire nel magazzino.
Garanzia di successo	Controllo delle giacenze consistente.
Scenario principale di successo	1. Il libraio chiede al sistema l'aggiornamento delle copie residue in magazzino. 2. Il libraio usando BO.BO. ricerca il libro da inserire in magazzino. 3. Il sistema mostra l'esito positivo della ricerca e chiede di specificare la quantità di copie da aggiungere. 4. Il libraio specifica il numero di copie e conferma. 5. Il sistema verifica se ci sono prenotazioni attive per quel libro. 6. Il sistema non trova alcuna prenotazione. 7. Il libraio conferma l'operazione. 8. Il sistema aggiorna le copie residue del catalogo e mostra l'esito dell'aggiornamento al libraio.

Estensioni	***a.** In qualsiasi momento, il sistema fallisce e ha un arresto improvviso. 1. Il libraio riavvia il software e richiede il ripristino dello stato precedente del sistema. 2. Il sistema ricostruisce lo stato precedente. **3a.** Il sistema mostra l'esito negativo della ricerca e torna al passo 2. **6a.** Il sistema trova una o più prenotazioni relative al libro. 1. Il sistema aggiorna la prenotazione in funzione delle nuove copie residue in magazzino. 2. Il sistema provvede a inviare una comunicazione via email al cliente riguardo la disponibilità del libro prenotato. 3. Ripetere i passi 1 e 2 finchè per il libro sono presenti altre prenotazioni attive e finchè lo permettono le copie residue in magazzino.
Requisiti speciali	Non specificati.
Elenco delle varianti tecnologiche e dei dati	
Frequenza di ripetizione	Legata al numero di ordini evasi e al numero di nuovi libri in arrivo da aggiungere al magazzino.
Problemi Aperti	Se il sistema non trova il libro nel catalogo dovrebbe proporre di crearlo e aggiungerlo al catalogo?

UC7: Ricerca Vendita

1. Il libraio richiede al sistema di ricercare una vendita esistente.
2. Il sistema richiede il codice identificativo della vendita da cercare.
3. Il libraio inserisce l'identificativo e conferma.
4. Il sistema restituisce la scheda di vendita corrispondente.

UC8: Gestisci promozioni, CRUD

Nome del caso d'uso	**UC8: Gestisci promozioni**
Portata	Applicazione BO.BO.

Livello	Obiettivo utente
Attore primario	Libraio
Parti interessate e Interessi	• Libraio: vuole gestire in maniera semplice la creazione, la modifica, la cancellazione e la ricerca di una promozione da associare agli acquisti.
Pre-condizioni	
Garanzia di successo	
Scenario principale di successo	1. Il libraio richiede al sistema di **inserire** una nuova promozione. 2. Il sistema richiede al libraio la condizione che autorizza la promozione. 3. Il libraio inserisce la condizione. 4. Il sistema chiede al libraio la percentuale di sconto da applicare. 5. Il libraio inserisce la percentuale di sconto.
Estensioni	**1a.** Il libraio richiede al sistema la **modifica** di una promozione. 1. Il sistema richiede al libraio i dati aggiornati. 2. Il libraio aggiorna i campi di interesse e conferma. **1b.** Il libraio richiede al sistema di **ricercare** una promozione. 1. Il sistema richiede al libraio di inserire le informazioni nei campi di ricerca. 2. Il libraio inserisce le informazioni a sua disposizione. 3. Il sistema mostra l'elenco delle promozioni che soddisfano i criteri di ricerca. **1c.** Il libraio richiede al sistema di **eliminare** una promozione. 1. Il sistema chiede al libraio di confermare l'eliminazione. 2. Il libraio conferma. 3. Il sistema elimina la promozione.
Requisiti speciali	Non specificati.
Elenco delle varianti tecnologiche e dei dati	
Frequenza di ripetizione	Legata al numero di promozioni esistenti o da creare/eliminare, solitamente bassa.

Problemi Aperti	

UC9: Associa Promozione

Nome del caso d'uso	**UC9: Associa Promozione**
Portata	Applicazione BO.BO.
Livello	Sottofunzione
Attore primario	Libraio
Parti interessate e Interessi	• Libraio: vuole associare correttamente e in maniera automatica eventuali promozioni per i clienti sulla base di alcune regole stabilite. • Cliente: desidera acquistare i libri usufruendo di sconti e promo.
Pre-condizioni	è in corso un acquisto
Garanzia di successo	
Scenario principale di successo	1. Il sistema presenta tutte le promozioni applicabili all'acquisto in corso. 2. Il libraio seleziona la promozione da applicare e conferma. 3. Il sistema applica la promozione, calcola il nuovo totale dell'acquisto e comunica al libraio il risultato di tale operazione.
Estensioni	
Requisiti speciali	Non specificati.
Elenco delle varianti tecnologiche e dei dati	
Frequenza di ripetizione	Ogni volta che viene fatto un acquisto da un cliente.
Problemi Aperti	

1.5 Documento di Visione

Parallelamente alla stesura di questo capitolo è stato redatto il documento di Visione, data la lunghezza di tale elaborato, per la sua visione si rimanda all'*appendice **A***.

1.6 Regole di business

Per il corretto utilizzo dell'applicazione devono essere rispettate le seguenti regole di dominio:

ID	Regola	Modificabilità	Sorgente
R1	Se l'importo della vendita corrente supera la cifra di € 100 viene applicato uno sconto del 5%	Alta, il libraio può decidere di modificare lo sconto e la soglia minima in qualsiasi momento	Politica interna della libreria
R2	Se la data d'acquisto rientra nel mese corrente viene applicato uno sconto del 5%	Media, il libraio potrebbe decidere di replicare la promozione in altri periodi dell'anno	Politica interna della libreria
R3	Se i libri vengono acquistati per uno studente delle scuole medie viene applicato un buono sconto del valore di 50 euro (contributo statale per l'acquisto dei libri per la scuola dell'obbligo)	Bassa, gli importi dei contributi possono variare al più una volta l'anno	Ministero della pubblica istruzione
R4	A discrezione del libraio, può essere applicata una promozione completa che tiene conto di tutti gli sconti applicabili.	Bassa, dipende anche delle modifiche alle altre promozioni	Politica interna della libreria

1.7 Specifiche Supplementari

Usabilità

- L'interfaccia grafica deve essere intuitiva e semplice anche per un utente non esperto.
- L'interazione con il sistema non deve presentare un elevato grado di complessità.
- Viene inserito un help in linea con le istruzioni per usare il sistema, anche se deve essere comunque possibile utilizzare il sistema senza la lettura di istruzioni preliminari.

Affidabilità

- Il software sviluppato deve essere affidabile e deve poter mantenere i propri dati anche in caso di guasti (guasti elettrici, usura dell'hardware, attacchi informatici).
- Deve essere possibile pianificare dei backup periodici del database la cui cadenza sarà definita più avanti nel progetto.

Vincoli hardware e software

- Per eseguire il software non ci sono particolari requisiti per il sistema operativo purchè sia presente la Java Virtual Machine.
- Per inviare le comunicazioni ai clienti è indispensabile avere una connessione a Internet attiva.

Vincoli di sviluppo del software

- Tutto il software è scritto scritto usando Java e sfrutta un database per gestire la persistenza dei dati.

Aspetti legali

- Le tecnologie utilizzate per la progettazione e realizzazione del sistema proposto sono di tipo open source o freeware.
- BO.BO. verrà rilasciato con licenza open source GPL v3.

1.8 Glossario

Vengono qui riportati i termini più significativi e le relative definizioni.

- **Vendita**: termine riferito a tutto il processo, dall'arrivo del cliente alla libreria fino al pagamento e ritiro dei libri.
- **Acquisto**: acquisto di uno o più libri da parte di un cliente, è previsto che dopo il pagamento venga rilasciata una ricevuta al cliente. Gli attributi da considerare nell'applicazione sono: id, data, totale.
- **Giacenza**: libri di testo residui nel magazzino della libreria
- **Disponibilità**: sinonimo intercambiabile di giacenza
- **Casa Editrice** o **Editore**: impresa o associazione che si occupa della produzione e distribuzione dei libri
- **Fornitore**: distributore zonale per i testi di una particolare casa editrice
- **Ordine**: ordine di uno o più libri da un fornitore (e dunque riferito a un solo editore)
- **Prenotazione**: prenotazione di un libro per il cliente qualora non fosse disponibile in negozio. Nota: copie multiple vengono gestite in prenotazioni separate.

- **ISBN** (*International Standard Book Number*): sequenza numerica di 13 cifre usata internazionalmente per la classificazione e l'identificazione univoca dei libri
- **Promozione**: sconto dedicato ad alcune categorie di clienti per incentivare le vendite (ne esistono diversi tipi: Promozione per Data, Promozione X% su spesa minima, Sconto Fisso, ecc)
- **Adozioni Scolastiche**: elenchi dei libri che vengono usati in una determinata scuola suddivisi in base alla classe
- **Magazzino**: reparto che si occupa della gestione degli ordini verso i fornitori e dell'aggiornamento delle giacenze
- **Catalogo**: si occupa di tenere traccia dell'elenco di tutti i libri acquistabili nella libreria; inoltre conserva la lista di tutti gli editori.
- **Libro**: gli attributi di un libro da considerare nell'applicazione sono: ISBN, titolo, autore, editore, prezzo, copie residue.
- **Cliente**: gli attributi di un cliente da considerare nell'applicazione sono: nome, cognome, telefono, indirizzo email. Si presuppone che non esistano clienti omonimi (stesso nome e cognome), inoltre i clienti attualmente presenti nel Database sono i seguenti:

	id [PK] integer	nome text	cognome text	email text
1	1	Mario	Rossi	ingsw_lm32@email.it
2	2	Mark	Loria	mark.loria@gmail.com
3	3	Mary	Spada	swordy07@gmail.com
4	4	Maria	Rossi	ingsw_lm32@email.it
5	5	Orazio	Tomarchio	orazio.tomarchio@diit.unict.it
*				

2 Analisi Orientata agli oggetti

2.1 Introduzione

Seguendo l'approccio iterativo evolutivo suggerito da UP, la realizzazione dell'applicazione è stata articolata su 4 iterazioni. In questo modo è stato possibile implementare in maniera iterativa il nucleo dell'architettura del software, sono state risolte le problematiche relative ai rischi maggiori ed è stata affrontata un'analisi dei requisiti graduale in modo da limitare al minimo il danno causato da eventuali errori di progettazione e implementazione.
Per ciascuna iterazione in particolare ci si è occupati di gestire le seguenti problematiche:

- **Iterazione 1**
  - → implementazione scenario principale *UC1:Gestisci Vendita* tralasciando, per semplicità, l'associazione di una promozione alla vendita e la verifica della disponibilità dei testi in magazzino.
  - → implementazione di un caso d'uso di Start Up necessario per gestire le esigenze di inizializzazione per le prime iterazioni.
- **Iterazione 2**
  - → implementazione scenario alternativo *10a* del caso d'uso *UC1:Gestisci Vendita* in cui si prevede che alcuni dei libri richiesti non siano presenti in magazzino ma che comunque il cliente sia interessato all'acquisto e quindi alla prenotazione.
  - → implementazione del caso d'uso *UC2: Crea Ordine Distributore*, esaminato con i suoi scenari alternativi (in particolare lo scenario alternativo *6a)* in modo da permettere la gestione della creazione di un ordine verso un distributore in una qualunque delle situazioni possibili.
- **Iterazione 3**
  - → implementazione del caso d'uso *UC6: Aggiorna Magazzino*, completo del suo scenario alternativo *6a*. In questo modo l'utente (libraio) sarà in grado di gestire l'aggiornamento delle copie residue nel magazzino tenendo conto di eventuali prenotazioni attive; in particolare, riguardo all'aggiornamento delle prenotazioni, è previsto un meccanismo automatico di invio email ai clienti in attesa dell'arrivo di copie prenotate per notificarne la disponibilità in negozio.

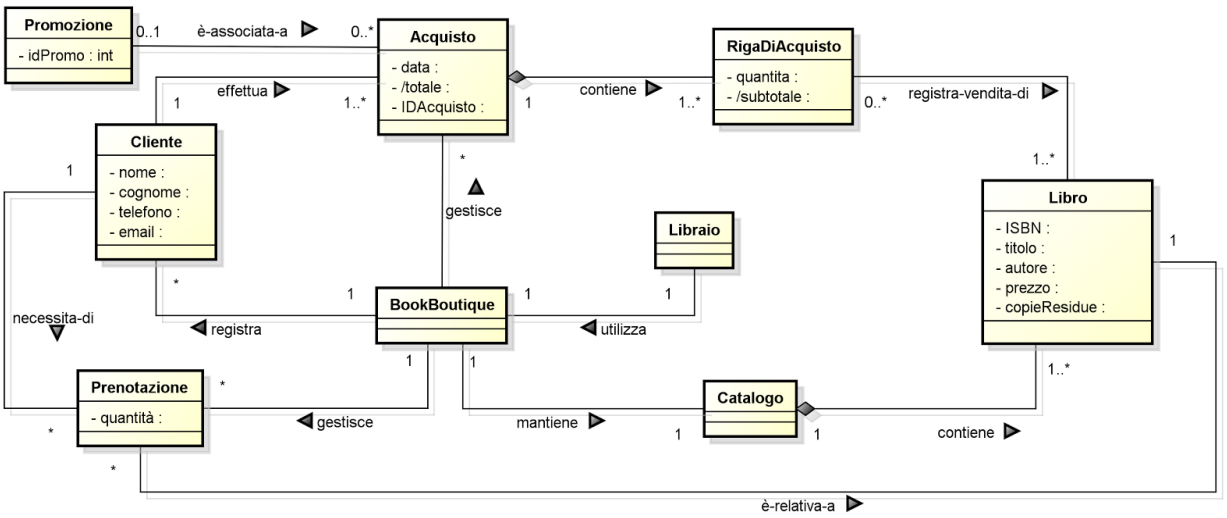
➢ **Iterazione 4**

→ implementazione del caso d'uso *UC9: Associa Promozioni*. Grazie a questo caso d'uso, durante un acquisto sarà possibile associare promozioni e sconti precedentemente creati nel sistema.

Il passo iniziale di ciascuna iterazione è stato quello di effettuare un'analisi dei requisiti Orientata agli Oggetti, cioè un'analisi basata sulla creazione di una descrizione del dominio da un punto di vista ad oggetti. Per fornire tale descrizione sono stati utilizzati diversi strumenti: Modello di Dominio, SSD (Sequence System Diagram) e Contratti delle operazioni.

2.2 Modello di Dominio

La disciplina che in termini di UP si occupa di fornire dettagli sul dominio è la Modellazione del Business, in particolare essa comprende la stesura del Modello di Dominio, elaborato grafico in cui vengono identificati i concetti, gli attributi e le associazioni considerati significativi. Tenendo conto del contributo dato da ciascuna iterazione, il modello di dominio è dato da:



Per un maggiore dettaglio grafico si veda l'*allegato **A1***. Come si nota, sono state identificate le seguenti classi concettuali:

- ***Acquisto***: rappresenta l'ordine di libri effettuato da un cliente
- ***RigaDiAcquisto***: contiene i dettagli relativi all'acquisto di un particolare libro e al relativo numero di copie;
- ***Libro***: rappresenta il prodotto e la relativa quantità residua in magazzino;
- ***Cliente***: cliente della libreria che effettua l'acquisto di libri;

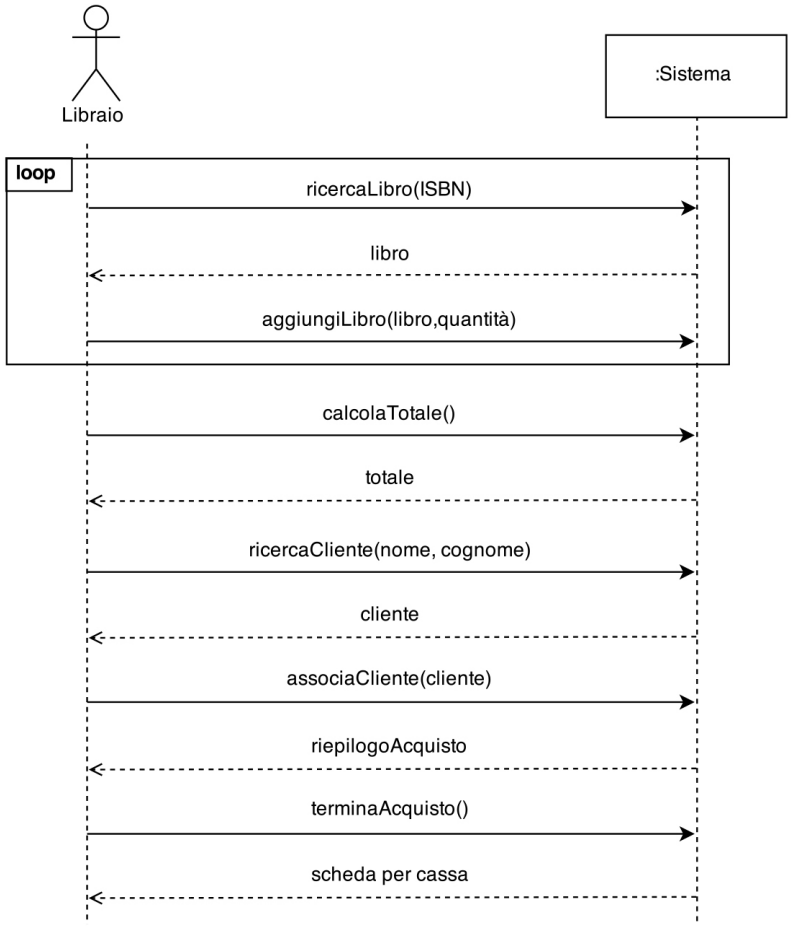
- ***Libraio***: attore primario, che interagisce direttamente con il sistema;
- ***BO.BO.***: rappresenta il sistema BOok BOutique;
- **Catalogo**: contiene tutte le descrizioni dei libri;
- ***Prenotazione***: rappresenta una prenotazione relativa a un libro richiesto da un cliente ma non presente in magazzino;
- ***Promozione***: rappresenta lo sconto da applicare in corrispondenza di determinate condizioni di vendita.

2.3 SSD e Contratti

Procedendo con l'analisi Orienta agli Oggetti, il passo successivo è la creazione dei Diagramma di Sequenza di Sistema (SSD) al fine di illustrare il corso degli eventi di input e di output per i vari casi d'uso esaminati in ciascuna iterazione. Inoltre le principali operazioni di sistema individuate negli SSD verranno descritte attraverso i Contratti, quindi avremo:

➢ **Iterazione 1**

Scenario principale semplificato del caso d'uso UC1:



Contratto CO1: aggiungiLibro

Operazione:	*aggiungiLibro*(*libro*: Libro, *quantità*: integer)
Riferimenti:	caso d'uso: Gestisci Vendita
Pre-condizioni:	è in corso un acquisto *acquistoInCorso*
Post-Condizioni:	- è stata creata un'istanza *rda* di RigaDiAcquisto
	- *rda* è stata associata ad *acquistoInCorso*
	- l'attributo quantità di *rda* è stato aggiornato con *quantità*

Contratto CO2: calcolaTotale

Operazione:	*calcolaTotale*()
Riferimenti:	caso d'uso: Gestisci Vendita
Pre-condizioni:	è in corso un acquisto *acquistoInCorso*
Post-Condizioni:	- l'attributo *totale* di *acquistoInCorso* è stato aggiornato

Contratto CO3: associaCliente

Operazione:	*associaCliente*(*cliente*: Cliente)
Riferimenti:	caso d'uso: Gestisci Vendita
Pre-condizioni:	è in corso un acquisto *acquistoInCorso*, il cliente è già registrato nell'archivio dei clienti
Post-Condizioni:	- *acquistoInCorso* è stato associato al cliente *cliente*

Contratto CO4: terminaAcquisto

Operazione:	*terminaAcquisto*()
Riferimenti:	caso d'uso: Gestisci Vendita
Pre-condizioni:	è in corso un acquisto *acquistoInCorso*
Post-Condizioni:	- è stata creata una nuova istanza di Acquisto *acquisto*
	- gli attributi di *acquisto* sono stati aggiornati con quelli di *acquistoInCorso*
	- *acquisto* è stato aggiunto all'archivio degli acquisti
	- è stato creato un file contenente il riepilogo dell'acquisto da presentare alla cassa

➢ **Iterazione 2**

Nello scenario alternativo *10a* del caso d'uso *UC1* l'aggiunta di eventuali prenotazioni viene fatta direttamente all'interno del sistema e, dunque, il libraio non svolge un'azione di input esplicita. Di conseguenza l'SSD resta invariato rispetto al caso precedente; risulta utile tuttavia descrivere attraverso un Contratto i dettagli relativi all'operazione di sistema *terminaAcquisto()* che è quella che si occupa di gestire la creazione delle prenotazioni:

Contratto CO1: terminaAcquisto

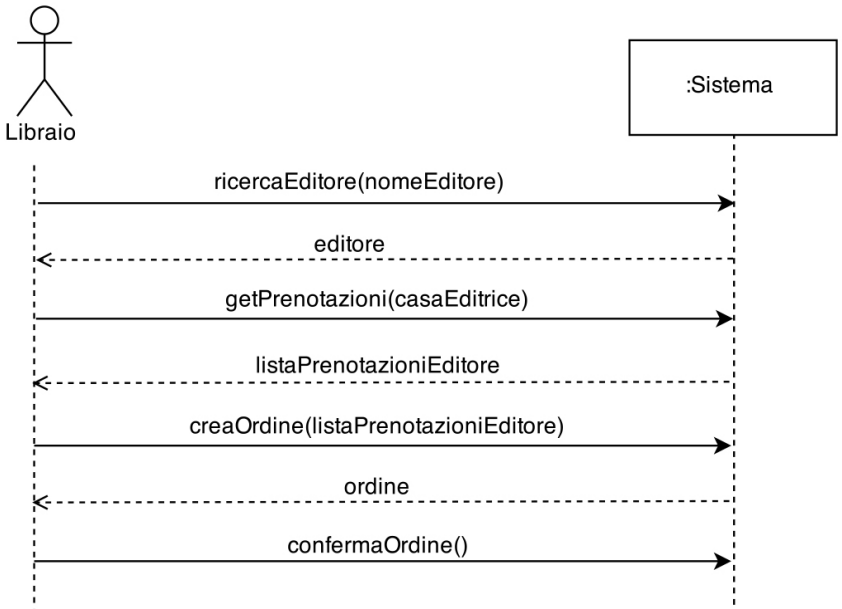
Operazione: *terminaAcquisto*()

Riferimenti: caso d'uso: Gestisci Vendita

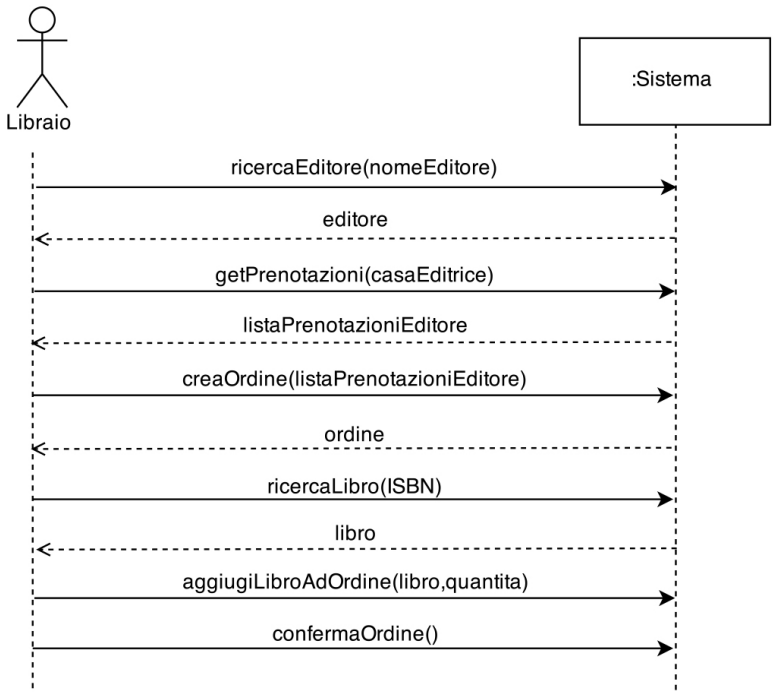
Pre-condizioni: è in corso una vendita

Post-Condizioni:
- è stata creata una nuova istanza di Acquisto *acquisto*
- l'attributo totale dell'Aquisto corrente è stato aggiornato
- gli attributi di *acquisto* sono stati aggiornati con quelli di *acquistoInCorso*
- *acquisto* è stato aggiunto all'archivio degli acquisti
- è stato creato un file contenente il riepilogo dell'acquisto da presentare alla cassa
- è stata creata una o più istanze di Prenotazione
- l'attributo *quantità* di tale/i istanza/e è stato aggiornato al numero di copie da prenotare
- la/e prenotazione/i è/sono stata/e aggiunta all'archivio delle prenotazioni
- l'attributo *copieResidue* del/i libro/i acquistati è stato aggiornato

Scenario principale di successo del caso d'uso UC2:



Scenario alternativo *6a* del caso d'uso *UC2*:



Come si nota l'unica differenza tra i due SSD è costituita dal fatto che nello scenario alternativo è prevista l'aggiunta all'ordine di libri specifici non appartenenti ad alcuna prenotazione, e cioè dalle operazioni di sistema *ricercaLibro()* e *aggiungiLibroAdOrdine()*. Le operazioni di sistema che meritano un maggiore chiarimento sono:

Contratto CO2: creaOrdine

Operazione:	*creaOrdine*()
Riferimenti:	caso d'uso: Crea Ordine Distributore
Pre-condizioni:	è in corso la creazione di un ordine verso un fornitore
Post-Condizioni:	- è stata creata una nuova istanza di Ordine *ordineInCorso* - l'attributo *prenotazioniEditore* (LinkedList) di *ordineInCorso* è stato aggiornato - per ogni prenotazione di *prenotazioniEditore*, è stata creata un'istanza *ro* di RigaOrdine - *ro* è stata aggiunta alla lista *righeOrdine*

Contratto CO3: aggiungiLibroAdOrdine

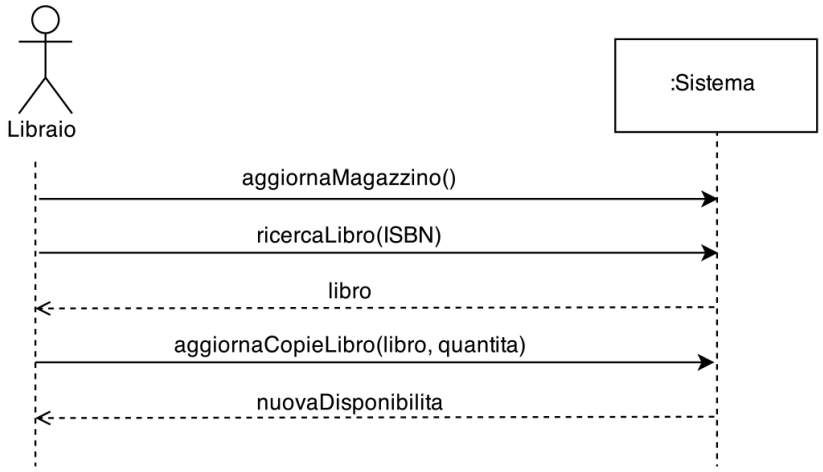
Operazione:	*aggiungiLibroAdOrdine*()
Riferimenti:	caso d'uso: Crea Ordine Distributore
Pre-condizioni:	è in corso la creazione di un ordine verso un fornitore
Post-Condizioni:	- è stata creata una nuova istanza *ro* di RigaOrdine - *ro* è stata aggiunta alla lista *righeOrdine*

Contratto CO4: confermaOrdine

Operazione:	*confermaOrdine*()
Riferimenti:	caso d'uso: Crea Ordine Distributore
Pre-condizioni:	è in corso la creazione di un ordine verso un fornitore
Post-Condizioni:	- l'attributo *ordinato* di ogni prenotazione di *prenotazioniEditore* è stato aggiornato a "*true*" - è stata creata una nuova istanza di Ordine *nuovoOrdine* - *nuovoOrdine* è stato aggiunto alla lista *ordini*

➢ **Iterazione 3**

L'SSD relativo allo scenario principale e a quello alternativo *6a* del caso d'uso scelto per questa iterazione (*UC6*) è:



Vale la pena scrivere in dettaglio il Contratto relativo all'operazione di sistema *aggiornaCopieLibro(libro, quantita)* nel caso in cui si verifichi lo scenario alternativo *6a*:

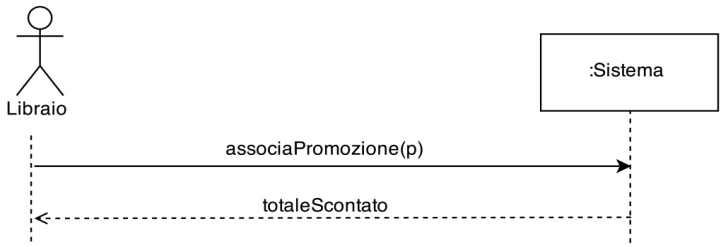
Contratto CO1: aggiornaCopieLibro(libro,quantita)

Operazione:	*aggiornaCopieLibro(libro, quantita)*
Riferimenti:	caso d'uso: Aggiorna Magazzino
Pre-condizioni:	è presente una prenotazione attiva *prenotazione* per *libro*
Post-Condizioni:	- l'attributo *copieResidue* di *libro* è stato aggiornato in funzione di quantita e delle copie precedentemente prenotate - l'attributo *copieConsegnate* di *prenotazione* è stato aggiornato - è stata inviata una email al cliente relativo a *prenotazione*

➢ **Iterazione 4**

Esaminando il caso d'uso *UC9*, poichè esso è una **sottofunzione** del caso d'uso *UC1*: *Gestisci Vendita,* si nota che vi è una sola operazione che il libraio richiede esplicitamente al

sistema: la richiesta di associazione di una particolare promozione all'acquisto in corso, dunque si ha:



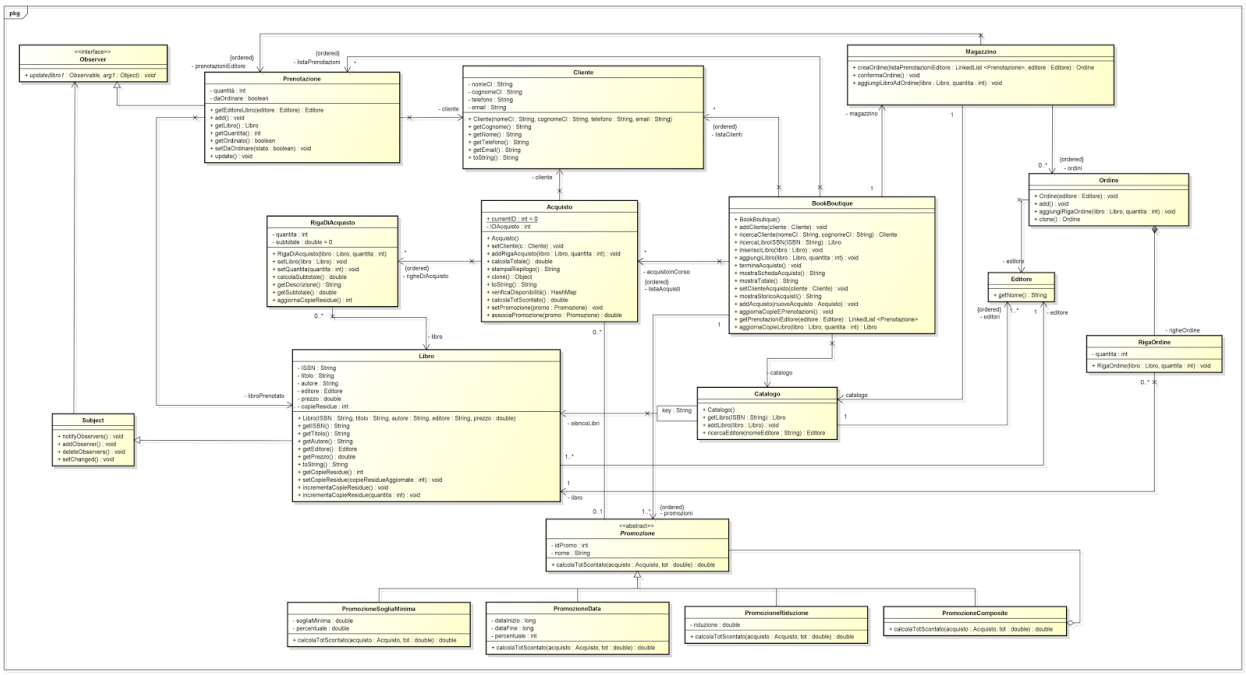
Contratto CO1: associaPromozione(p)

Operazione:	*associaPromozione(p)*
Riferimenti:	caso d'uso: Associa Promozione, sottofunzione di Gestisci Vendita
Pre-condizioni:	è in corso un acquisto
Post-Condizioni:	- l'attributo *totaleScontato* di *acquistoInCorso* è stato aggiornato in funzione della promozione p

3 Progettazione

3.1 Diagramma delle Classi

La progettazione orientata agli oggetti è la disciplina di UP interessata alla definizione degli oggetti software, delle loro responsabilità e a come questi collaborano per soddisfare i requisiti individuati nei passi precedenti. L'elaborato principale di questa fase che è stato preso in considerazione è il **Modello di Progetto**, ovvero l'insieme dei diagrammi che descrivono la progettazione logica sia da un punto di vista dinamico (Diagrammi di Interazione) che da un punto di vista statico (Diagramma delle Classi).
Il diagramma delle Classi dunque sarà:

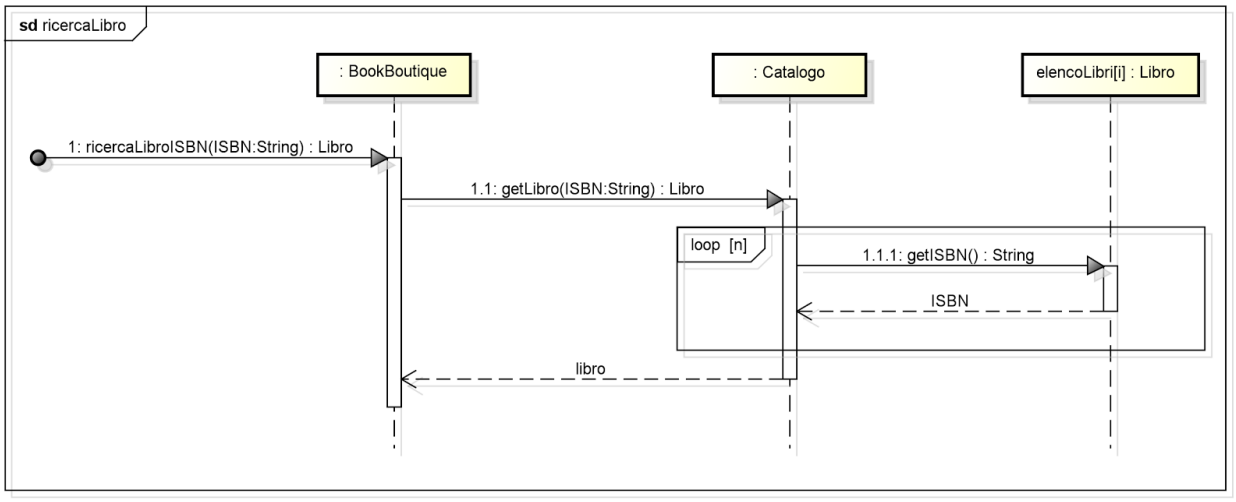


Per un maggiore dettaglio grafico si veda l'*allegato **B1***.

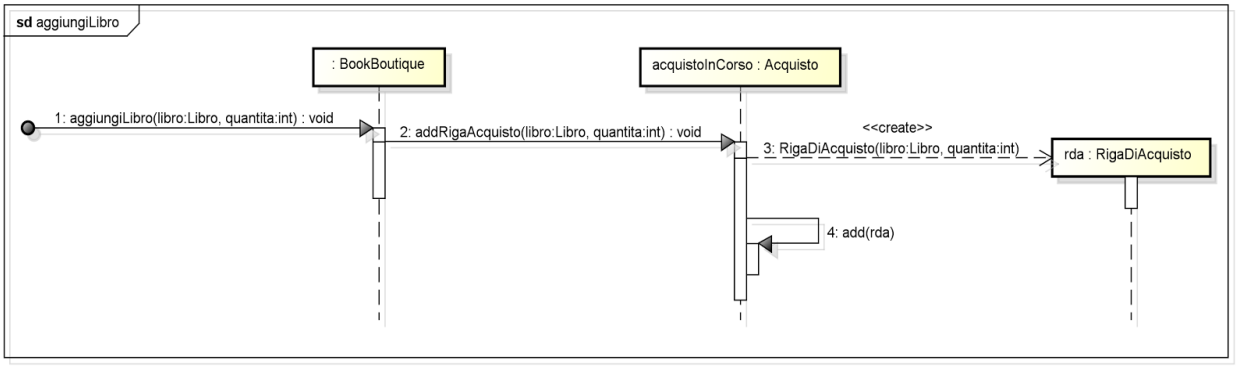
3.2 Diagrammi di Sequenza

Vengono ora presentati i più importanti diagrammi di sequenza, per maggiori dettagli e/o per visionarli tutti si veda il documento sorgente creato con il software *Astah Professional* (file Elab4-progetto-completo.asta).

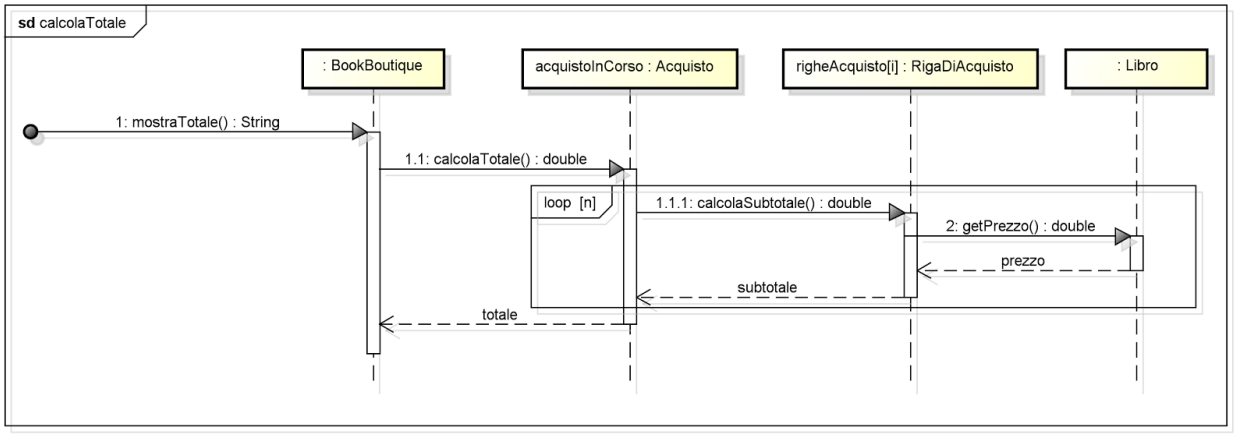
- Ricerca di un libro dal catalogo



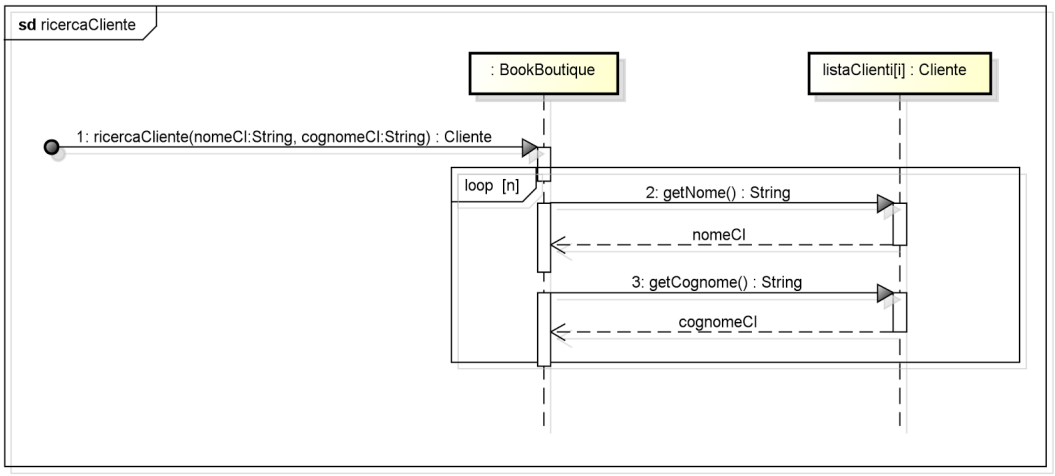
- Aggiunta di un libro ad un acquisto



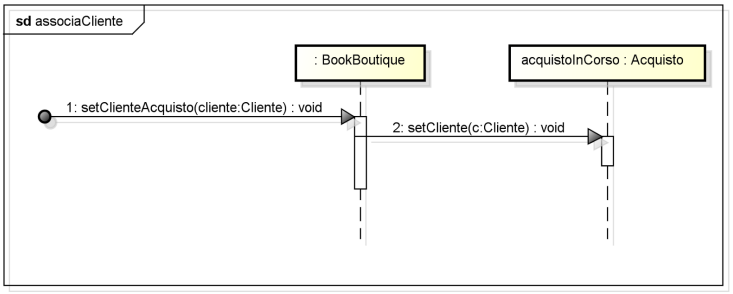
- Calcola il totale dell'intero acquisto



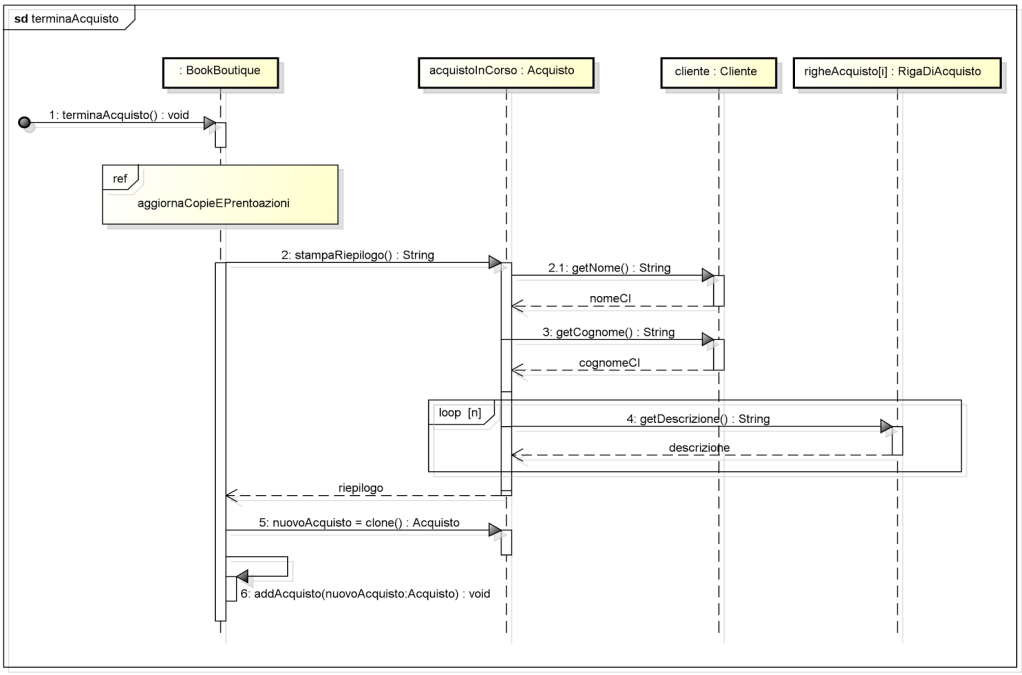
- Ricerca di un cliente



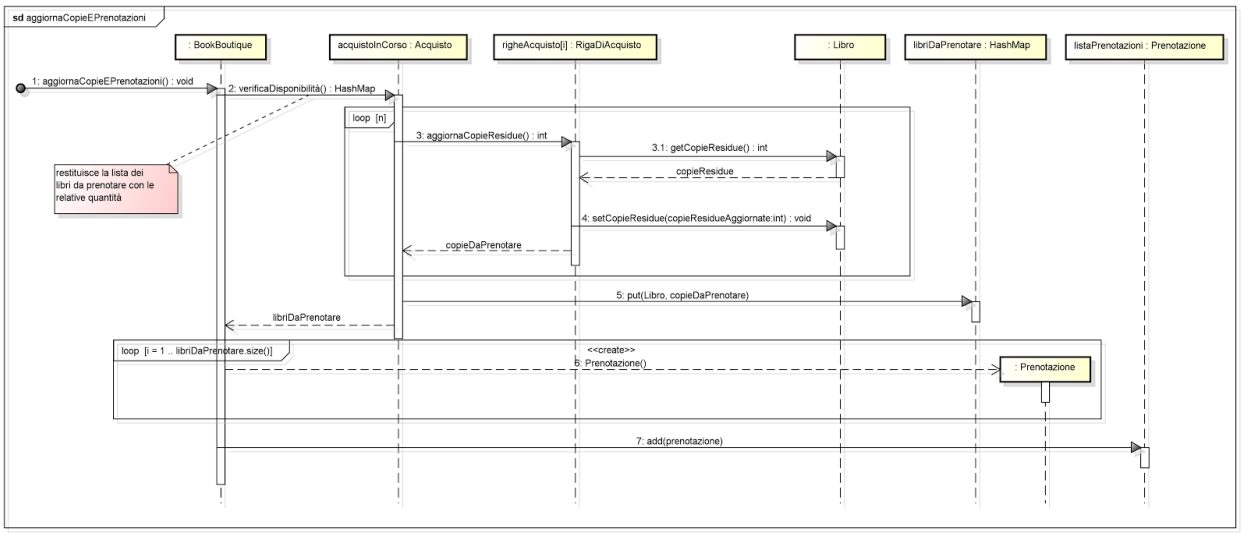
- Associazione del cliente all'acquisto



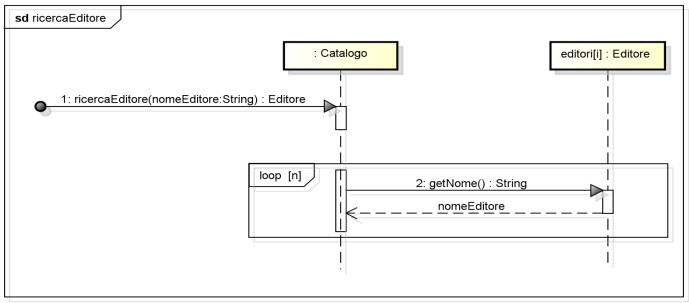
- Terminazione di un acquisto



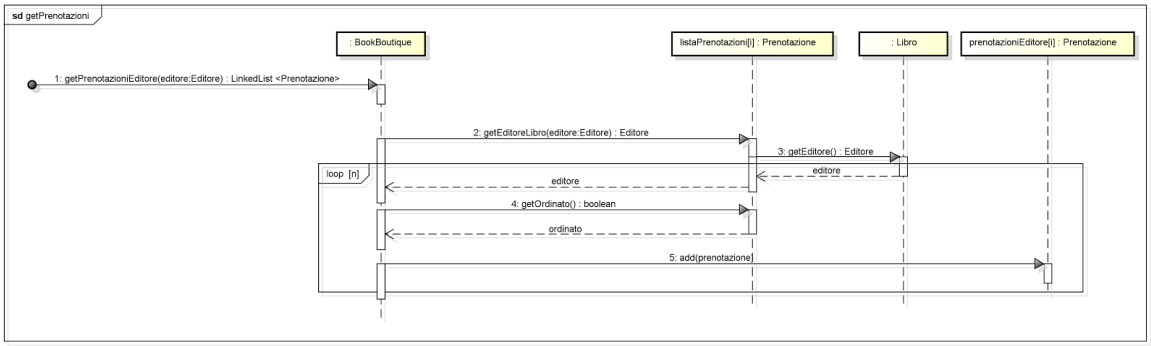
- Aggiorna copie residue e crea eventuali prenotazioni



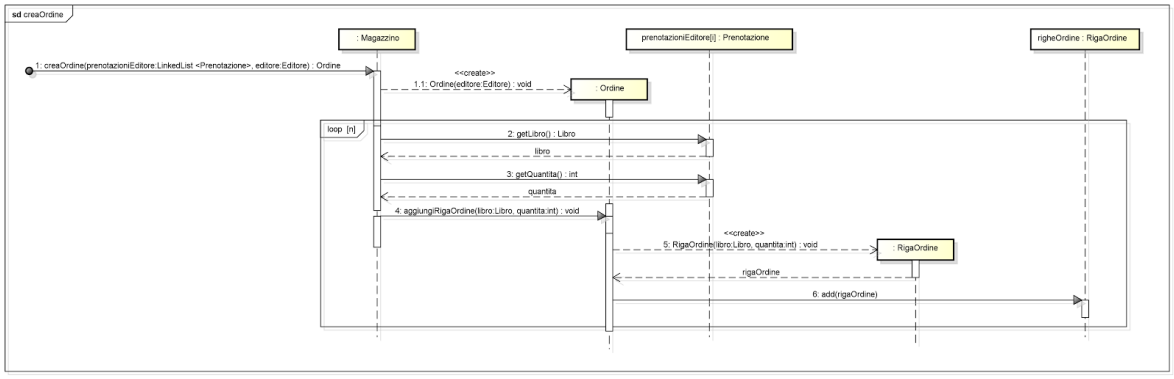
- Ricerca un editore



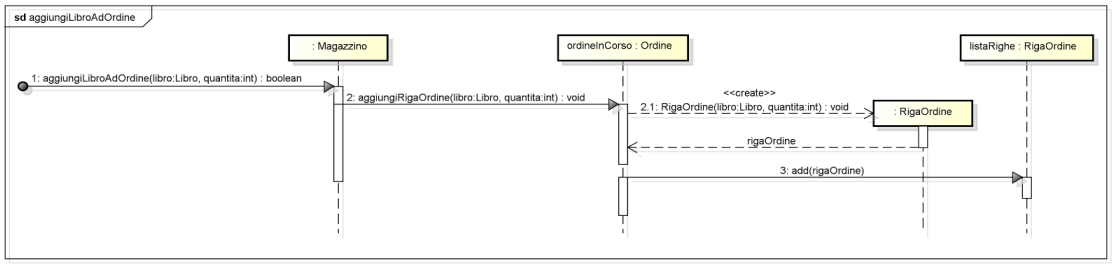
- Calcola e restituisce le prenotazioni relative a un particolare editore



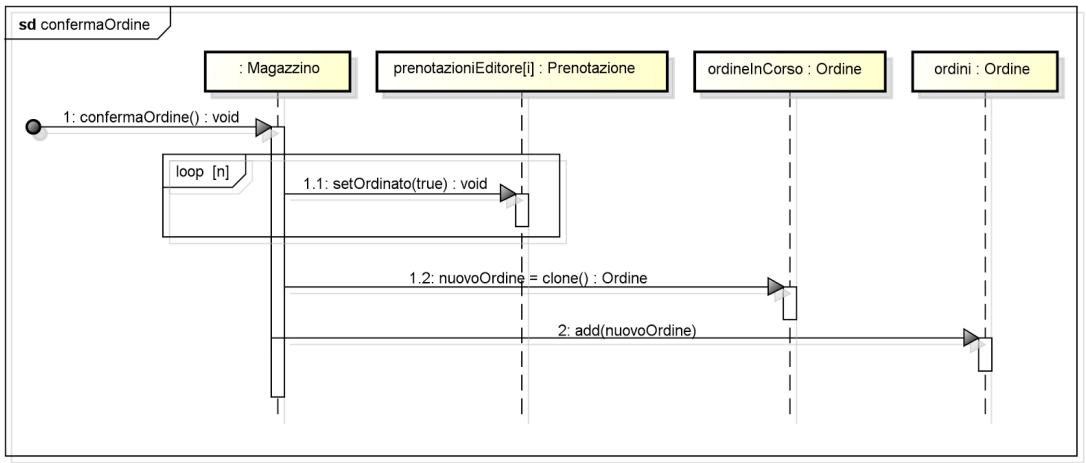
- Crea un nuovo ordine con le prenotazioni da ordinare



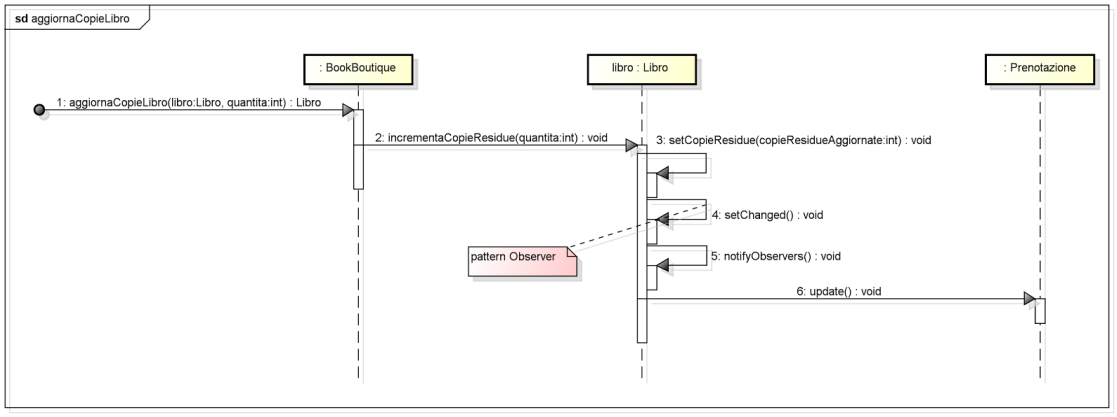
- Aggiunge un libro all'ordine in corso



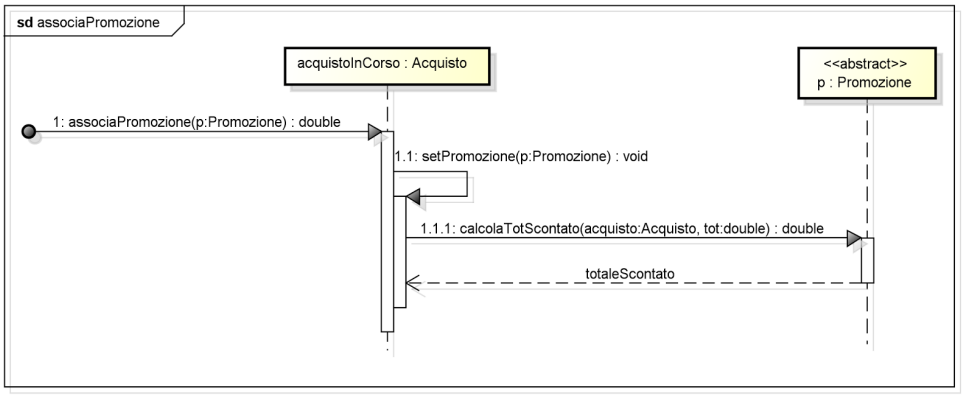
- Conferma l'ordine



- Aggiorna le copie residue di un libro (libro, quantita)



- Associa una promozione all'acquisto in corso



4 Testing

4.1 Introduzione

Il **testing** è una fase essenziale del processo di sviluppo di un programma robusto ed efficiente. Permette inoltre di ridurre in modo sostanziale i costi di manutenzione di un applicazione.

La probabilità di rilevare malfunzionamenti è proporzionale al numero di test eseguiti, tuttavia è difficile testare un programma completamente. Infatti le possibili combinazioni di valori di input da prendere in considerazione sono enormi, e non sempre possono essere riprodotte in un tempo ragionevole. Tuttavia un testing progettato accuratamente può rivelare comportamenti anomali ed indesiderati al fine di rendere il software realizzato funzionante e funzionale secondo le specifiche del committente.

I test si dividono principalmente in due famiglie:

- I **test unitari** (***Unit Test***) → Sono semplici test/prove che vanno a verificare la correttezza direttamente del codice, in ogni sua piccola parte. L'idea dello Unit Test in Java è quella di valutare ogni singolo metodo in funzione dei valori attesi. Esistono diversi tool per Unit Testing per i diversi linguaggi di programmazione, in particolare ***JUnit*** è il framework più diffuso attualmente per l'automazione del testing di unità di programmi Java.
- I **test funzionali** → Sono dei test che vanno a verificare che il sistema software nella sua completezza funzioni correttamente. Questi test trattano il sistema come se fosse una scatola nera alla quale danno degli input e verificano la correttezza degli output.

Per eseguire il testing dell'applicazione realizzata, si è scelto di concentrarsi principalmente su test unitari eseguiti tramite l'ausilio di JUnit e quindi di seguire un approccio ***Buttom-Up*** collaudando prima le piccole unità che comporranno poi il programma completo.

4.2 Individuazione dei casi di test e Testing Unitario

In una fase preliminare al testing, è stata affrontata una discussione e ispezione del codice sorgente scritto al fine di operare una scelta sulle classi e sui metodi da testare. Il criterio scelto per individuare tali metodi su cui concentrare i test di unità è stato quello di dare priorità a tutti i metodi che si occupano dell'inserimento di nuove istanze, dell'aggiornamento delle copie in magazzino e del calcolo dei fattori economici quali: quantità, calcolo del subtotale e del totale, applicazione degli sconti, ecc. Per ciascuno sono state individuate le **classi di equivalenza** tali da poter ridurre il numero di test al minimo insieme necessario, in particolare:

- Acquisto
    - addRigaAcquisto
        - Una riga d'oridine relativa a un numero positivo di libri è inserita tra le righe d'acquisto.
        - Una riga d'acquisto relativa ad un numero negativo o nullo non è inserita tra le righe d'acquisto.
    - calcolaTotale
        - Il totale di un acquisto contenente almeno un libro è calcolato correttamente (precisione 0.01 €).
        - Il totale di un acquisto vuoto è zero.
    - associaPromozione
        - Il totale scontato su un acquisto contente almeno un libro è calcolato correttamente.
        - Il totale scontato di un acquisto vuoto è zero.
- Catalogo
    - aggiungiEditore
        - Tre editori distinti sono correttamente inseriti nella lista degli editori.
        - Un editore duplicato non viene inserito nella lista degli editori.
        - Un editore con nome nullo non viene inserito nella lista degli editori.
    - cercaEditore
        - La ricerca con una sottostringa univoca restituisce l'istanza cercata.
        - Una sottostringa non univoca restituisce null come risultato della ricerca.
    - addLibro
        - Un libro con ISBN già presente in catalogo non viene aggiunto.
        - Un libro senza ISBN non è aggiunto al catalogo.
        - Un libro correttamente istanziato con ISBN non nullo è correttamente aggiunto al catalogo.
    - getLibro
        - La ricerca di un libro mediante ISBN (univoco) restituisce l'istanza cercata.
        - La ricerca di un libro non presente in catalogo restituisce null.
- Libro
    - incrementaCopieResidue

        - Incrementare le copie residue mediante un numero positivo aggiorna il valore correttamente.
        - Un parametro negativo o nullo non alterano il numero di copie residue.
    - assegna
        - Non è possibile assegnare un numero negativo di copie.
        - Non è possibile assegnare più copie di quanti sono presenti in magazzino.
    - setCopieResidue
        - Un parametro nullo o positivo incrementa correttamente il numero di copie.
        - Un parametro negativo non modifica il numero di copie residue.
- Ordine
    - aggiungiRigaOrdine
        - La riga d'ordine relativa ad un libro con quantità nulla non viene aggiunta alla lista di righe d'ordine.
        - Una riga d'ordine relativa ad un libro con quantità positiva e dello stesso editore dell'ordine è aggiunta alla lista di righe d'ordine.
        - Una riga d'ordine relativa ad un libro con editore diverso da quello dell'ordine non è aggiunta alle righe d'ordine.
- RigheOrdine
    - aggiornaCopieResidue
        - Verifica che il metodo restituisca il numero corretto di copie da prenotare.

A seguito dell'esecuzione di questi test unitari con JUnit, si è passati a correggere le relative porzioni di codice che portavano al fallimento di alcuni test.

4.3 Test di Sistema

In previsione di una fase di refactoring in cui verrà integrato un Database con l'applicazione esistente, si è scelto di eseguire dei test di sistema manuali lanciando l'applicazione e provando a portare a termine i principali casi d'uso con i rispettivi scenari alternativi.

A seguito di tali test non sono emersi grandi errori nel funzionamento e le prestazioni del sistema si sono mantenute piuttosto elevate, tuttavia sono state rilevate delle imperfezioni e anomalie nell'esecuzione di alcune funzionalità:

- Effettuando la ricerca di un cliente e successivamente provando ad effettuare un nuovo acquisto, la procedura non può essere portata a termine perchè non risulta visibile il tasto “Avanti” che permette di proseguire con l’acquisto.
- L’utente è obbligato a selezionare una promozione da associare all’acquisto in corso, ovvero non è possibile terminare un acquisto senza applicarvi una promozione.
- Dopo la creazione di un ordine non viene salvato alcun file contenente i dettagli dell’ordine da inoltrare al fornitore.

5 Refactoring e Conclusioni

5.1 Database e Refactoring

In questa fase conclusiva, le operazioni prevalentemente svolte sono state:

- la correzione delle anomalie individuate con il precedente Test di Sistema;
- l'integrazione nel software di un database per gestire la persistenza dei dati. In particolare il database utilizzato è un DB remoto PostgreSQL hostato sulla piattaforma Heroku. L'interfacciamento del DB con l'applicazione è stato realizzato con la libreria Hibernate ORM.
- aggiunta di alcune funzionalità secondarie dell'applicazione, quali la visualizzazione del catalogo, l'inserimento di un nuovo libro, ecc, al fine di completare le operazioni dei casi d'uso rimaste in sospeso.

5.2 Test di accettazione

A conclusione delle attività necessarie per portare a completamento i passi precedenti, e dopo un'attenta revisione di tutto il codice sorgente, è stato eseguito il test di accettazione conclusivo. Simulando di ritrovarsi nei panni dell'utente (libraio), sono state testate le principali funzionalità del programma al fine di dimostrare che è possibile portare a termine con successo i casi d'uso implementati e i loro scenari alternativi.