La Colazione di Rosa

a cura di **Luca Cabibbo**

9 gennaio 2006

Prefazione

Questo documento ha presentato requisiti, analisi e progettazione per *La colazione di Rosa*, uno studio di caso per il corso di *Analisi e progettazione del software*.

In modo analogo a quanto avviene con gli altri studi di caso del corso, questo progetto è stato sviluppato in modo iterativo, con iterazioni guidate da scelte più di natura didattica che di un "progetto reale". In particolare, questo documento descrive due iterazioni (*iterazione 1* e *iterazione 2*) per *La colazione di Rosa*. Oltre a queste, viene descritta anche l'attività di refactoring svolta a cavallo tra le due iterazioni, per facilitare l'introduzione dei requisiti di interesse per l'iterazione 2 nel progetto e nel codice prodotto nel corso dell'iterazione 1. Questa attività viene descritta come *iterazione 1 refactoring* (anche se in realtà viene svolta all'inizio dell'iterazione 2).

Oltre a questo documento, lo studio di caso è composto dal codice di un'applicazione per *La colazione di Rosa*, prodotto in diverse versioni. In particolare, è disponibile il codice per le iterazioni 1, 1 refactoring e 2, ma anche per un'*iterazione 3*, di cui viene fornito il codice ma non la documentazione di progetto.

Tutte le versioni sono dotate di una semplice interfaccia utente a caratteri. Le versioni 1, 1 refactoring e 2 gestiscono i dati solo in memoria principale. La versione 3 è dotata anche di un'interfaccia utente grafica e gestisce i propri dati persistenti in una base di dati relazionale; in questa versione sono stati implementati degli ulteriori requisiti, non descritti in questo documento.

L'enfasi di questo lavoro è sull'analisi e progettazione orientata agli oggetti. La maggior parte dello sforzo è stata volta a realizzare un progetto corretto e coerente con quanto insegnato nel corso di *Analisi e progettazione del software*.

D'altra parte, tutti gli elementi accessori sono sicuramente perfettibili. Le interfacce utente sono molto semplici, e sicuramente possono essere fatte meglio. L'accesso alla base di dati è stato fatta in un certo modo, e sicuramente esistono dei modi alternativi, forse migliori. Il codice potrebbe contenere degli errori, e probabilmente sarebbe stato utile fare dei test unitari. Tuttavia, nessuno di questi era un obiettivo primario di questo lavoro, e quindi questi elementi accessori sono stati mostrati più per completezza che non per mostrare il modo migliore di procedere.

Luca Cabibbo vuole ringraziare tutti coloro che hanno lavorato a *La colazione di Rosa*: Paolo Papotti, insieme al quale è stato inizialmente studiato lo studio di caso; Fabrizio Martorelli, Andrea Petreri e Gabriele Rendina del corso di *Progetto di sistemi informatici*, che hanno scritto le prime versioni di questo documento e del codice; Marco Canu, che ha collaborato ad allineare il codice con l'ultima versione di questo documento.

Capitolo 1

Requisiti

1.1 Introduzione

La colazione di Rosa offre un servizio di preparazione e consegna di colazioni complete a casa dei suoi clienti.

I clienti possono ordinare da un apposito menu, indicare un luogo e un orario, e la colazione sarà consegnata loro a domicilio.

Rosa ha composto diversi tipi di colazione, ciascuno con i suoi cibi e le sue bevande. Ad esempio, una *colazione francese* è composta da una tazza di caffè francese, un bicchiere di succo d'arancia, due cornetti, burro e marmellata. Una *colazione inglese* è composta invece da due uova fritte con pancetta, tre fette di pane tostato, marmellata, un bicchiere di succo d'arancia e un bricco di tè. A Rosa piace aggiungere nel proprio menu, di tanto in tanto, un nuovo tipo di colazione.

Un ordine può essere relativo anche a più persone, e ciascuno può scegliere il suo tipo preferito di colazione.

Rosa offre un po' di flessibilità ai suoi clienti, che possono ordinare varianti delle colazioni previste nel menu, aggiungendo, levando o modificando i loro componenti. Ad esempio, si può ordinare una colazione francese con un cornetto in più (ovvero, tre cornetti anziché due), senza il caffè francese ma con una tazzina di caffè espresso. Oppure si può ordinare una colazione inglese con il miele anziché la marmellata, ed in più una tazzina di caffè espresso.

Una colazione può essere servita in diversi modi; ad esempio, normale, superiore o lusso. Nel modo *normale* una colazione viene servita con bicchieri di plastica, tovaglioli di carta e vassoio di cartone. Nel modo *superiore* la colazione viene servita col vassoio di legno e bicchieri di vetro. Nel modo *lusso* viene servita col vassoio d'argento e anche un vaso di fiori. Ovviamente, il modo con cui viene servita una colazione influisce sul prezzo. A Rosa piace cambiare, di tanto in tanto, i possibili modi con cui è possibile servire le colazioni. Ad esempio, sta pensando ad un modo compleanno.

1.2 Casi d'uso

Si considerino i seguenti casi d'uso, di cui è di interesse solo lo scenario principale di successo.

Caso d'uso UC1:
Inserimento nuovo tipo di colazione

Attore primario: un Addetto del sistema.

1. L'Addetto inizia l'immissione di un nuovo tipo di colazione.
2. L'Addetto inserisce il nome e il codice identificativo del nuovo tipo di colazione.
3. L'Addetto inserisce il codice identificativo di un componente e una quantità. Il Sistema aggiunge quel componente, in quella quantità, al nuovo tipo di colazione.

Il passo 3 viene ripetuto finché serve.

4. L'Addetto inserisce il prezzo base per il nuovo tipo di colazione. Il Sistema mostra i dati relativi al nuovo tipo di colazione immesso.
5. L'Addetto conferma i dati immessi. Il Sistema registra tutte le informazioni sul nuovo tipo colazione.

Caso d'uso UC2:
Inserimento nuovo ordine

Attore primario: un Addetto del sistema.

1. Il Cliente telefona a *La Colazione di Rosa* per ordinare delle colazioni.
2. L'Addetto inizia un nuovo ordine.
3. Il Cliente dice quale tipo di colazione vuole ordinare, e l'Addetto ne inserisce il codice identificativo. Il Sistema registra la colazione richiesta e mostra la composizione della colazione scelta.

Il passo 4 sarà ripetuto finché serve.

4. Il Cliente specifica un componente della colazione richiesta da modificare o rimuovere, indicando la quantità desiderata. L'Addetto inserisce il codice identificativo del componente e la nuova quantità richiesta (zero per cancellare). Il Sistema registra la modifica alla colazione richiesta.

Il passo 5 sarà ripetuto finché serve.

5. Il Cliente specifica un componente (che non è previsto nella colazione richiesta) da aggiungere, indicando la quantità desiderata. L'Addetto inserisce il codice identificativo del componente e la quantità richiesta. Il Sistema registra la modifica alla colazione richiesta.

I passi da 3 a 5 vengono ripetuti finché serve, per richiedere altre colazioni nell'ambito dello stesso ordine.

6. Il Cliente specifica il modo con cui devono essere servite le colazioni ordinate. L'Addetto inserisce il codice identificativo del modo di servizio richiesto. Il Sistema registra il modo di servizio richiesto.
7. Il Sistema calcola e mostra il prezzo delle colazioni ordinate (applicando l'algoritmo opportuno, vedi le *Regole di dominio*). L'Addetto comunica il prezzo al Cliente e ottiene conferma dell'ordine.
8. Il Cliente comunica il proprio nome e cognome, nonché la data, l'ora e l'indirizzo della consegna per le colazioni richieste. L'Addetto inserisce queste informazioni nel Sistema. Il Sistema registra le informazioni sul cliente e mostra il riepilogo delle informazioni sull'ordine.
9. L'Addetto conferma l'ordine. Il Sistema registra l'ordine. L'addetto saluta il Cliente e chiude la telefonata.

1.3 Regole di dominio

La Colazione di Rosa adotta, per calcolare il prezzo di una colazione, un algoritmo basato sulle seguenti regole:

A. Ciascun tipo di colazione ha un *prezzo base*; ad esempio, la colazione francese ha come

prezzo base 7.00 euro. Normalmente il prezzo base è maggiore o uguale al prezzo dei componenti.

B. Ciascun componente ha un prezzo aggiuntivo e un prezzo riduttivo; ad esempio, un cornetto ha come prezzo aggiuntivo 1.00 euro e come prezzo riduttivo 0.75 euro. Il *prezzo aggiuntivo* di un componente è quello che va sommato al prezzo di una colazione quando viene richiesta una unità aggiuntiva di quel componente (ad esempio, un cornetto in più). Il *prezzo riduttivo* di un componente è quello che va sottratto al prezzo di una colazione quando viene richiesta una unità in meno di quel componente (ad esempio, un cornetto in meno).

C. Ciascun modo con cui può essere servita una colazione ha un *fattore moltiplicativo*; ad esempio, 1 per il modo normale e 1.5 per il modo superiore, che va moltiplicato al prezzo della colazione, già modificato per tener conto delle variazioni.

Ad esempio, una colazione francese (7.00) con un cornetto in più (+1.00) servita nel modo superiore ($\times$1.5) costa 12.00 euro.

Capitolo 2

Iterazione 1, Analisi

2.1 Introduzione

Per l'iterazione 1, sono stati scelti i seguenti requisiti:

- lo scenario principale di successo del caso d'uso UC1 (Inserimento nuovo tipo di colazione);
- uno scenario principale di successo del caso d'uso UC2 (Inserimento nuovo ordine), in cui è possibile solo ordinare colazioni nella loro forma standard, senza variazioni;
- il calcolo del prezzo di un ordine è basato sulle regole di dominio A e C (la regola di dominio B non si applica);
- caso d'uso d'avviamento;
- dati solo in memoria principale.

Questo capitolo descrive l'analisi svolta nell'iterazione 1, mentre il capitolo successivo descrive l'attività di progettazione. In questa iterazione, analisi e progettazione vengono svolte separatamente per i due casi d'uso, iniziando con il caso d'uso UC1 e poi proseguendo con il caso d'uso UC2.

2.2 Caso d'uso UC1, Modello di dominio

In questa iterazione, del caso d'uso UC1 è di interesse lo scenario principale di successo, nella sua interezza, riportato nel Paragrafo 1.2. Da esso è possibile identificare le seguenti classi concettuali:

- *Addetto*: attore primario, che interagisce direttamente con il sistema;
- *CdR*: rappresenta il sistema *La Colazione di Rosa*;
- *Menu*: contiene i tipi di colazione predefiniti, gestiti dal sistema, nonché altre informazioni che ci si aspetta di trovare nel menu della Colazione di Rosa, come i componenti aggiuntivi e i modi di servizio;
- *TipoColazione*: un tipo di colazione che è possibile ordinare, nella sua forma standard; ad esempio, la colazione francese;

- *DescrizioneComponente*: la descrizione di un componente della colazione; ad esempio, di un cornetto; è importante mantenere le descrizioni delle componenti indipendentemente dall'esistenza o meno della componente stessa;
- *ComponenteColazione*: un elemento di un tipo di colazione, che rappresenta una componente, cibo o bevanda, nel tipo di colazione e la relativa quantità.

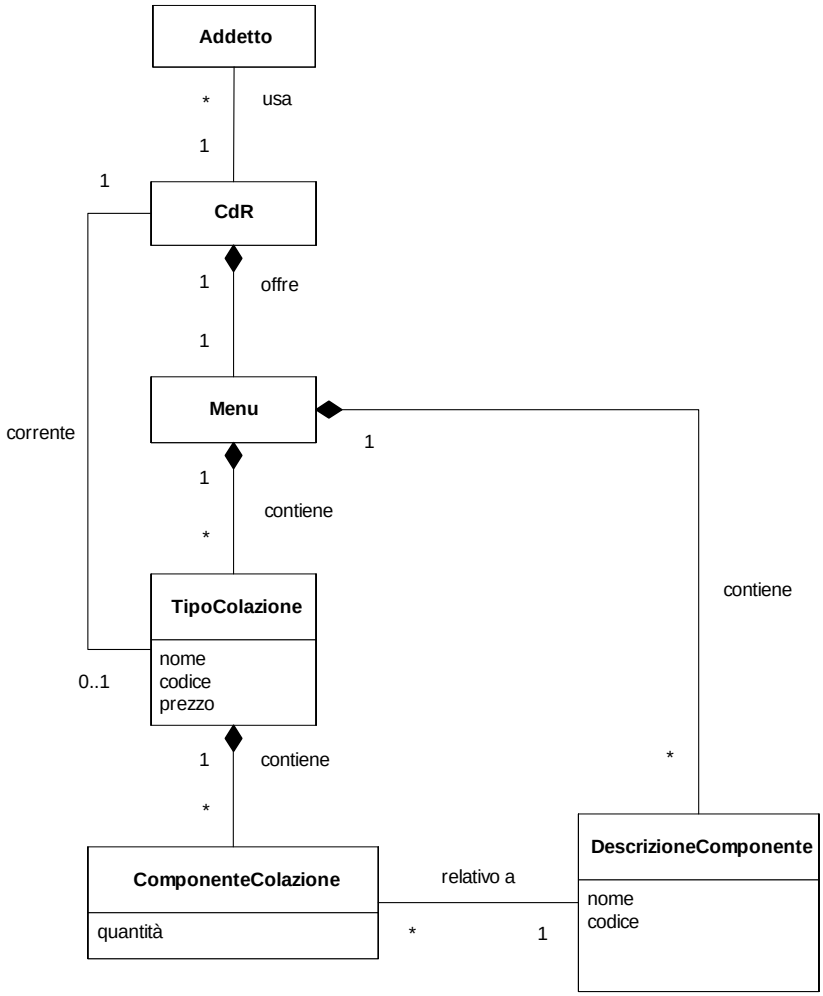

Figura 2.1: UC1: modello di dominio

Un *TipoColazione* è essenzialmente la "ricetta" per una colazione. Ad esempio, una *colazione semplice*, composta da un cappuccino e due cornetti, potrebbe essere descritta dalla seguente ricetta.

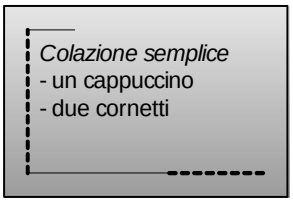

Figura 2.2: Ricetta della colazione semplice

Il seguente diagramma di oggetti di dominio mostra gli oggetti che possono essere utilizzati per rappresentare la *colazione semplice*. L'oggetto *colazionesemplice:TipoColazione* rappresenta l'intera "ricetta"; ciascun oggetto *ComponenteColazione* rappresenta una riga della "ricetta", composta da un ingrediente e dalla quantità richiesta; infine, gli oggetti *DescrizioneComponente* rappresentano gli ingredienti che possono comparire nelle ricette.

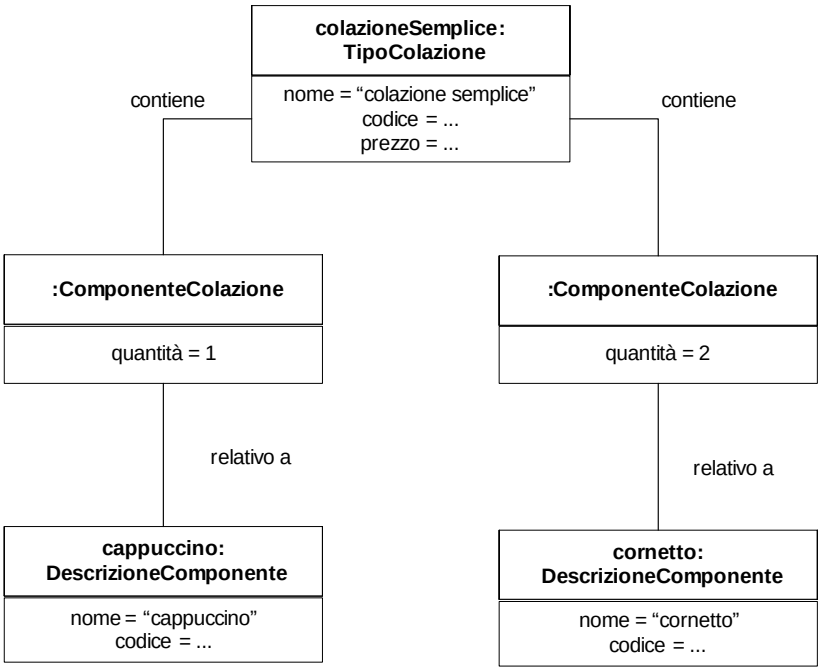

Figura 2.3: Oggetti di dominio per un tipo di colazione

2.3 Caso d'uso UC1, Diagramma di sequenza di sistema

Il diagramma di sequenza di sistema per il caso d'uso UC1 è il seguente:

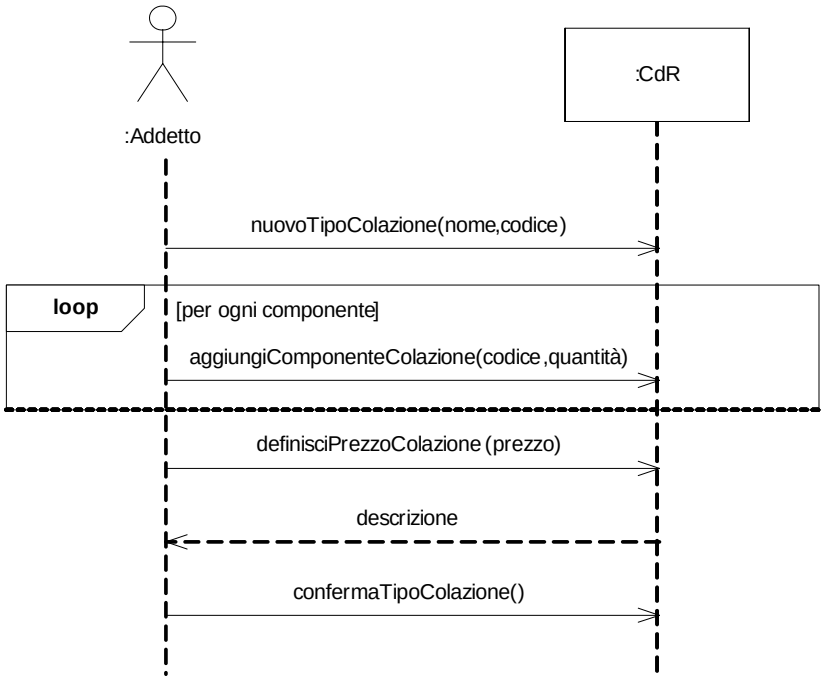

Figura 2.4: UC1: diagramma di sequenza di sistema

2.4 Caso d'uso UC1, Contratti delle operazioni

2.4.1 Nuovo tipo di colazione

L'operazione di sistema *nuovoTipoColazione* consente di iniziare la definizione di un nuovo tipo di colazione.

Operazione	nuovoTipoColazione(codice:String, nome:String)
Riferimenti	Caso d'uso: *Inserimento nuovo tipo di colazione*
Pre-condizioni	-
Post-condizioni	- è stata creata una nuova istanza *tc* di TipoColazione; - gli attributi di *tc* sono stati inizializzati; - *tc* è stata associata a CdR tramite l'associazione "corrente".

2.4.2 Aggiungi componente colazione

L'operazione di sistema *aggiungiComponenteColazione* consente di aggiungere un componente alla colazione corrente.

Operazione	aggiungiComponenteColazione(codice:String, quantità:Integer)
Riferimenti	Caso d'uso: *Inserimento nuovo tipo di colazione*
Pre-condizioni	- è in corso la definizione del TipoColazione *tc*
Post-condizioni	- è stata creata una nuova istanza *cc* di ComponenteColazione; - gli attributi di *cc* sono stati inizializzati; - *cc* è stata associata a *tc* tramite l'associazione "contiene"; - *cc* è stata associata a una DescrizioneComponente *dc*, sulla base del codice, tramite l'associazione "relativo a".

2.4.3 Definisci prezzo

L'operazione di sistema *definisciPrezzo* consente di fissare un prezzo base al tipo di colazione corrente.

Operazione	definisciPrezzo(prezzo:Double)
Riferimenti	Caso d'uso: *Inserimento nuovo tipo di colazione*
Pre-condizioni	- è in corso la definizione del TipoColazione *tc*
Post-condizioni	- è stato inizializzato l'attributo prezzo dell'istanza *tc* corrente di TipoColazione

2.4.4 ConfermaTipoColazione

L'operazione di sistema *confermaTipoColazione* si verifica quando la definizione di un nuovo tipo di colazione è terminata e questa va confermata e registrata.

Operazione	confermaTipoColazione()
Riferimenti	Caso d'uso: *Inserimento nuovo tipo di colazione*
Pre-condizioni	- è in corso la definizione del TipoColazione *tc*
Post-condizioni	- è stata associata l'istanza *tc* di TipoColazione corrente a Menu tramite l'associazione "contiene".

2.5 Caso d'uso UC2 semplificato

In questa iterazione, del caso d'uso UC2 è di interesse solo uno scenario principale di successo semplificato. Il caso d'uso UC2, completo, è riportato nel Paragrafo 1.2. Per comodità, di seguito viene riportato il solo scenario semplificato di interesse in questa iterazione:

Caso d'uso UC2 (semplificato):
Inserimento nuovo ordine

Attore primario: un Addetto del sistema.

1. Il Cliente telefona a *La Colazione di Rosa* per ordinare delle colazioni.
2. L'Addetto inizia un nuovo ordine.
3. Il Cliente dice quale tipo di colazione vuole ordinare, e l'Addetto ne inserisce il codice identificativo. Il Sistema registra la colazione richiesta e mostra la composizione della colazione scelta.

Il passo 3 viene ripetuto finché serve, per richiedere altre colazioni nell'ambito dello stesso ordine.

6. Il Cliente specifica il modo con cui devono essere servite le colazioni ordinate. L'Addetto inserisce il codice identificativo del modo di servizio richiesto. Il Sistema registra il modo di servizio richiesto.
7. Il Sistema calcola e mostra il prezzo delle colazioni ordinate (applicando l'algoritmo opportuno, vedi le *Regole di dominio*). L'Addetto comunica il prezzo al Cliente e ottiene conferma dell'ordine.
8. Il Cliente comunica il proprio nome e cognome, nonché la data, l'ora e l'indirizzo della consegna per le colazioni richieste. L'Addetto inserisce queste informazioni nel Sistema. Il Sistema registra le informazioni sul cliente e mostra il riepilogo delle informazioni sull'ordine.
9. L'Addetto conferma l'ordine. Il Sistema registra l'ordine. L'addetto saluta il Cliente e chiude la telefonata.

2.6 Caso d'uso UC2, Modello di dominio

Analizzando questo scenario, è possibile identificare anche le seguenti ulteriori classi concettuali:

- *Cliente*: un cliente della Colazione di Rosa, che può ordinare colazioni;
- *ModoServizio*: un modo di consegna per una colazione ordinata, ad esempio il modo normale;
- *Ordine*: un ordine di un cliente, composto da una o più colazioni;
- *ColazioneOrdinata*: un elemento di un ordine, che corrisponde a una colazione; questa classe, più che per esigenze rappresentative dell'iterazione corrente, è stata introdotta

per favorire la gestione delle varianti delle colazioni ordinate (che sono informazioni che andranno proprio associate alle istanze di questa classe).

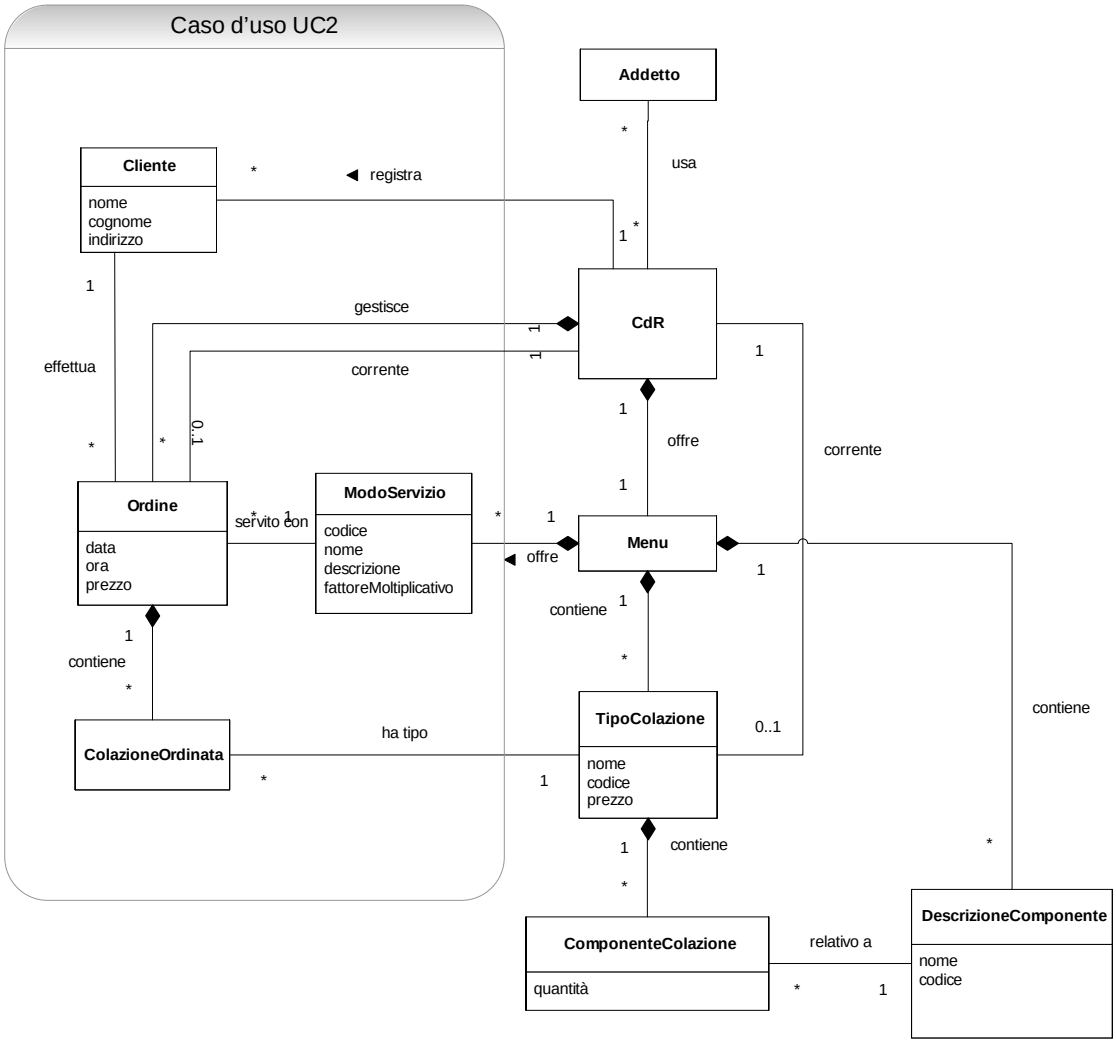

Figura 2.5: UC2: modello di dominio

Un oggetto *Ordine* è l'ordine di una o più colazioni, fatto da parte di un cliente. Un ordine potrebbe essere scritto su un apposito modulo, come mostrato nella seguente figura.

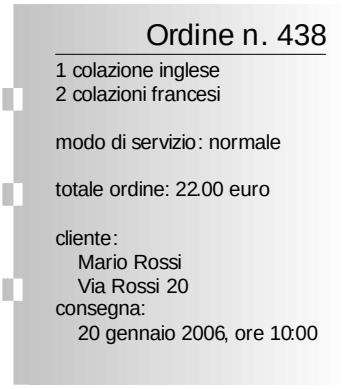

Figura 2.6: Un ordine

L'oggetto *o:Ordine* rappresenta l'intero ordine. Da una parte, esso è composto da righe, che specificano le colazioni richieste; queste sono rappresentate dagli oggetti *ColazioneOrdinata*. (Un'alternativa sarebbe stata di avere proprio degli oggetti *RigaOrdine*; la soluzione adottata qui è più adatta per incorporare i requisiti che poi saranno considerati nell'iterazione 2.) Gli altri elementi dell'ordine sono rappresentati da altri oggetti, associazioni ed attributi del modello di dominio, come mostrato nel seguente diagramma di oggetti di dominio, che rappresenta l'ordine in considerazione.

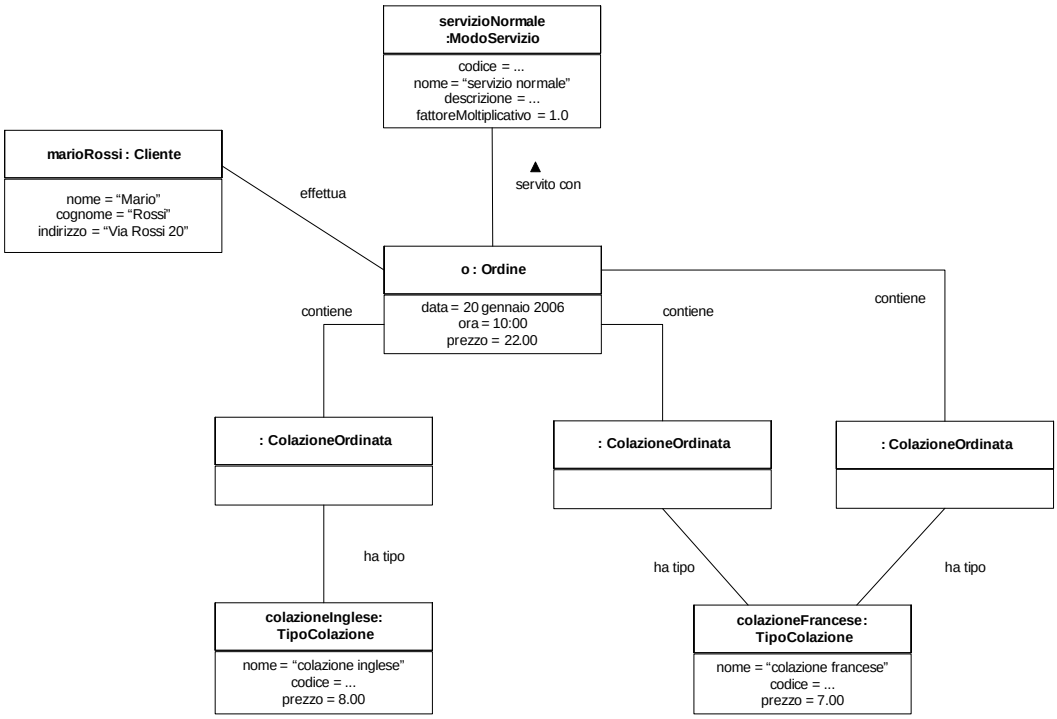

Figura 2.7: Oggetti di dominio per un ordine

2.7 Caso d'uso UC2, Diagramma di sequenza di sistema

Il diagramma di sequenza di sistema per lo scenario principale di successo (semplificato) del caso d'uso UC2 d'interesse per questa iterazione è il seguente:

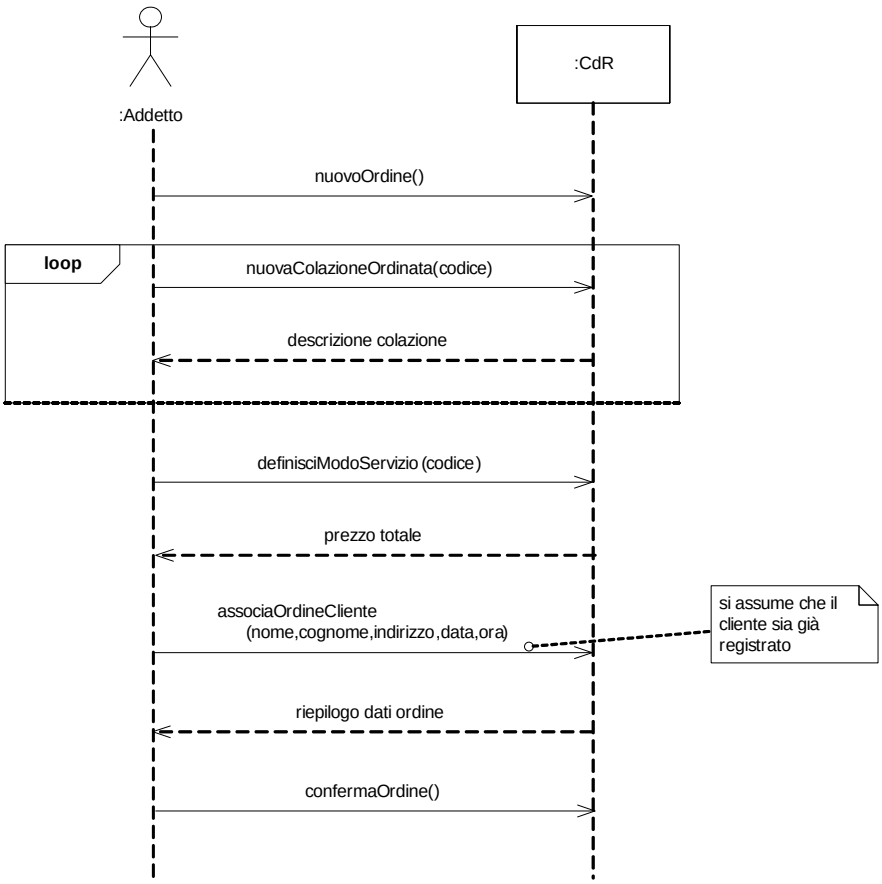

Figura 2.8: UC2: diagramma di sequenza di sistema

2.8 Caso d'uso UC2, Contratti delle operazioni

2.8.1 Nuovo ordine

L'operazione di sistema *nuovoOrdine* consente di iniziare un nuovo ordine, relativo a una o più colazioni.

Operazione	nuovoOrdine()
Riferimenti	Caso d'uso: *Inserimento nuovo ordine*
Pre-condizioni	-
Post-condizioni	- è stata creata una nuova istanza *o* di Ordine; - gli attributi di *o* sono stati inizializzati; - l'ordine *o* è stato associato a CdR tramite l'associazione "corrente".

2.8.2 Nuova colazione ordinata

L'operazione di sistema *nuovaColazioneOrdinata* consente di aggiungere una nuova colazione all'ordine corrente.

Operazione	nuovaColazioneOrdinata(codice:String)
Riferimenti	Caso d'uso: *Inserimento nuovo ordine*
Pre-condizioni	- è in corso l'inserimento di un Ordine *o*
Post-condizioni	- è stata creata una nuova istanza *co* di ColazioneOrdinata; - gli attributi di *co* sono stati inizializzati; - la colazione *co* è stata associata a un TipoColazione *tc*, sulla base del codice, tramite l'associazione "ha tipo"; - la ColazioneOrdinata *co* è stata associata all'Ordine corrente *o* tramite l'associazione "contiene".

2.8.3 Definisci modo servizio

L'operazione di sistema *definisciModoServizio* consente di associare un modo di servizio alle colazioni dell'ordine (il modo di servizio è lo stesso per tutte le colazioni dell'ordine).

Operazione	definisciModoServizio(codice:String)
Riferimenti	Caso d'uso: *Inserimento nuovo ordine*
Pre-condizioni	- è in corso l'inserimento di un Ordine *o*
Post-condizioni	- è stata associata l'istanza *o* di Ordine corrente a *ms* tramite l'associazione "servito con".

2.8.4 Associa ordine a cliente

L'operazione di sistema *associaOrdineCliente* consente di specificare il cliente che sta effettuando l'ordine corrente.

Operazione	associaOrdineCliente(nome:String,cognome:String, indirizzo:String,data: Date,ora:Time)
Riferimenti	Caso d'uso: *Inserimento nuovo ordine*
Pre-condizioni	- è in corso l'inserimento di un Ordine *o*
Post-condizioni	- è stata associata l'istanza *o* di Ordine corrente a un Cliente tramite l'associazione "effettua", sulla base di nome e cognome; - sono stati inizializzati gli attributi data e ora dell'Ordine corrente *o*.

2.8.5 Conferma ordine

L'operazione di sistema *confermaOrdine* indica che l'ordine corrente è confermato e va registrato nel sistema.

Operazione	confermaOrdine()
Riferimenti	Caso d'uso: *Inserimento nuovo ordine*
Pre-condizioni	- è in corso l'inserimento di un Ordine *o*
Post-condizioni	- è stato inizializzato l'attributo prezzo dell'Ordine corrente *o*, sulla base delle regole di dominio; - è stata associata l'istanza *o* di Ordine corrente a CdR tramite l'associazione "gestisce".

Capitolo 3

Iterazione 1, Progettazione

3.1 Introduzione

Alla Colazione di Rosa stanno lavorando due sviluppatori (progettisti e programmatori), Alice e Bob. Alice è una progettista esperta, mentre Bob sta imparando. Bob è incoraggiato da Alice a fare e motivare le sue proposte.

La prima scelta di progetto da fare è sul controller per i vari casi d'uso. Alice e Bob sono entrambi d'accordo per usare la classe CdR come facade controller.

Prima di iniziare la progettazione, osservando il modello di dominio, Bob si è espresso dicendo che secondo lui "la classe concettuale Menu non va implementata, perché non può gestire richieste, ma solo delegarle, e la sua presenza servirebbe solo ad aumentare l'accoppiamento nel sistema". La risposta di Alice è stata "Bene, iniziamo così, e poi vedremo". Pertanto la classe Menu non compare nel progetto per l'iterazione corrente.

3.2 Caso d'uso UC1, Diagrammi di interazione

3.2.1 nuovoTipoColazione(codice : String, nome : String)

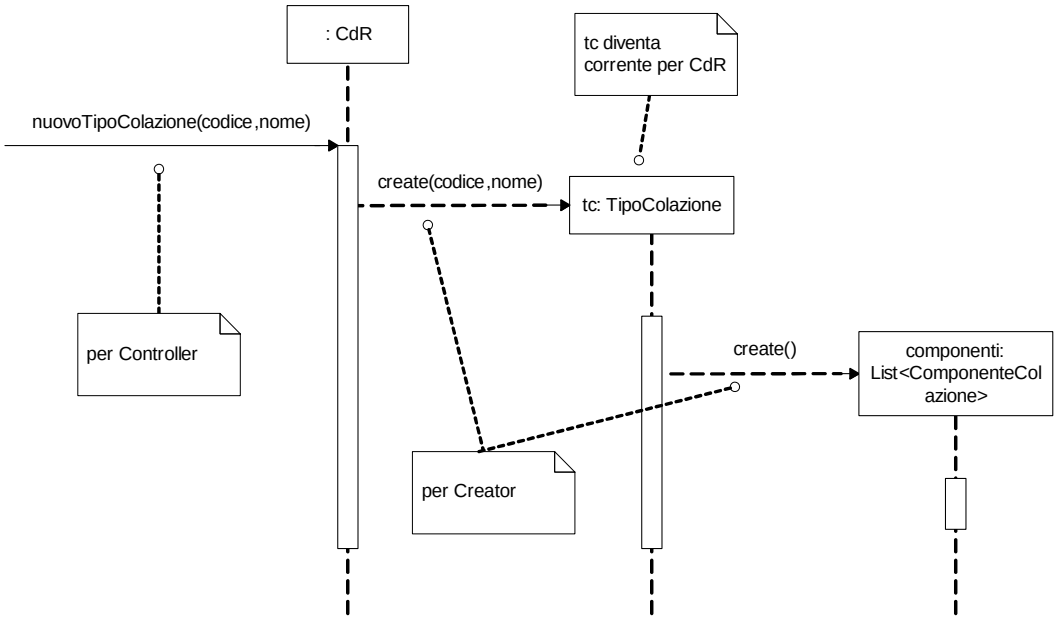

Figura 3.1: OP1: creazione nuovo tipo di colazione

3.2.2 aggiungiComponenteColazione(codice : String, quantità : Integer)

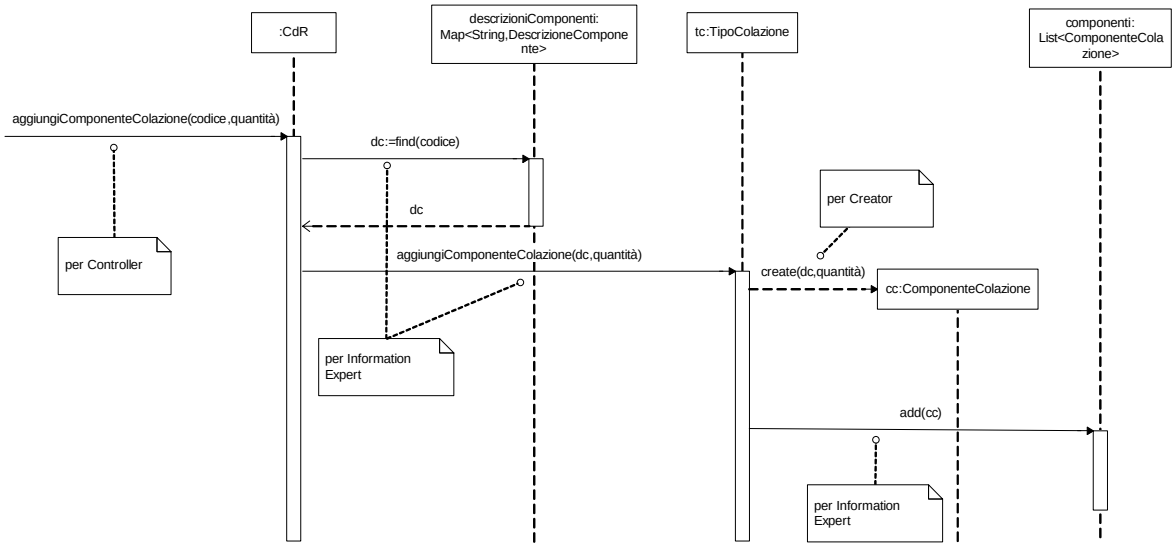

Figura 3.2: OP2: aggiungi componente al tipo di colazione

3.2.3 definisciPrezzo(prezzo:Double)

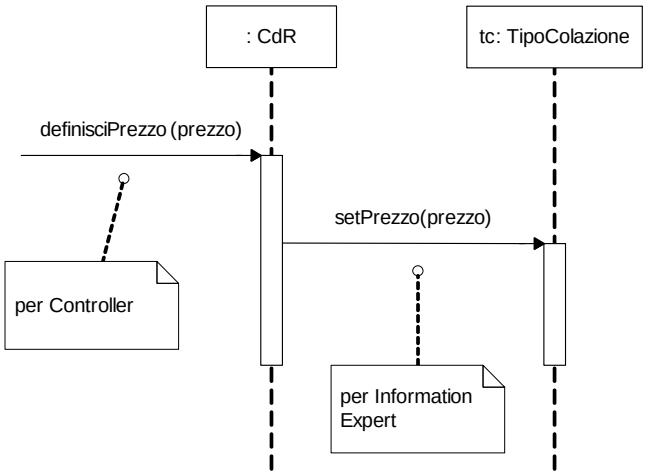

Figura 3.3: OP3: definisci il prezzo del tipo di colazione

3.2.4 confermaTipoColazione()

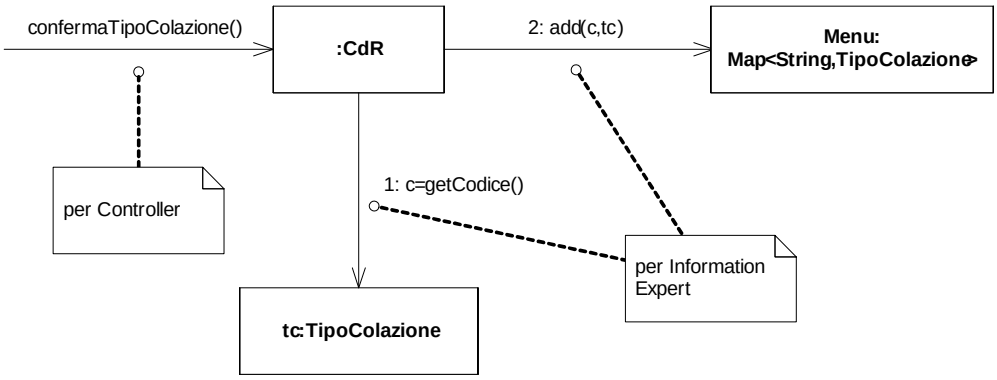

Figura 3.4: OP4: conferma il tipo di colazione

3.3 Caso d'uso UC1, Diagramma delle classi di progetto

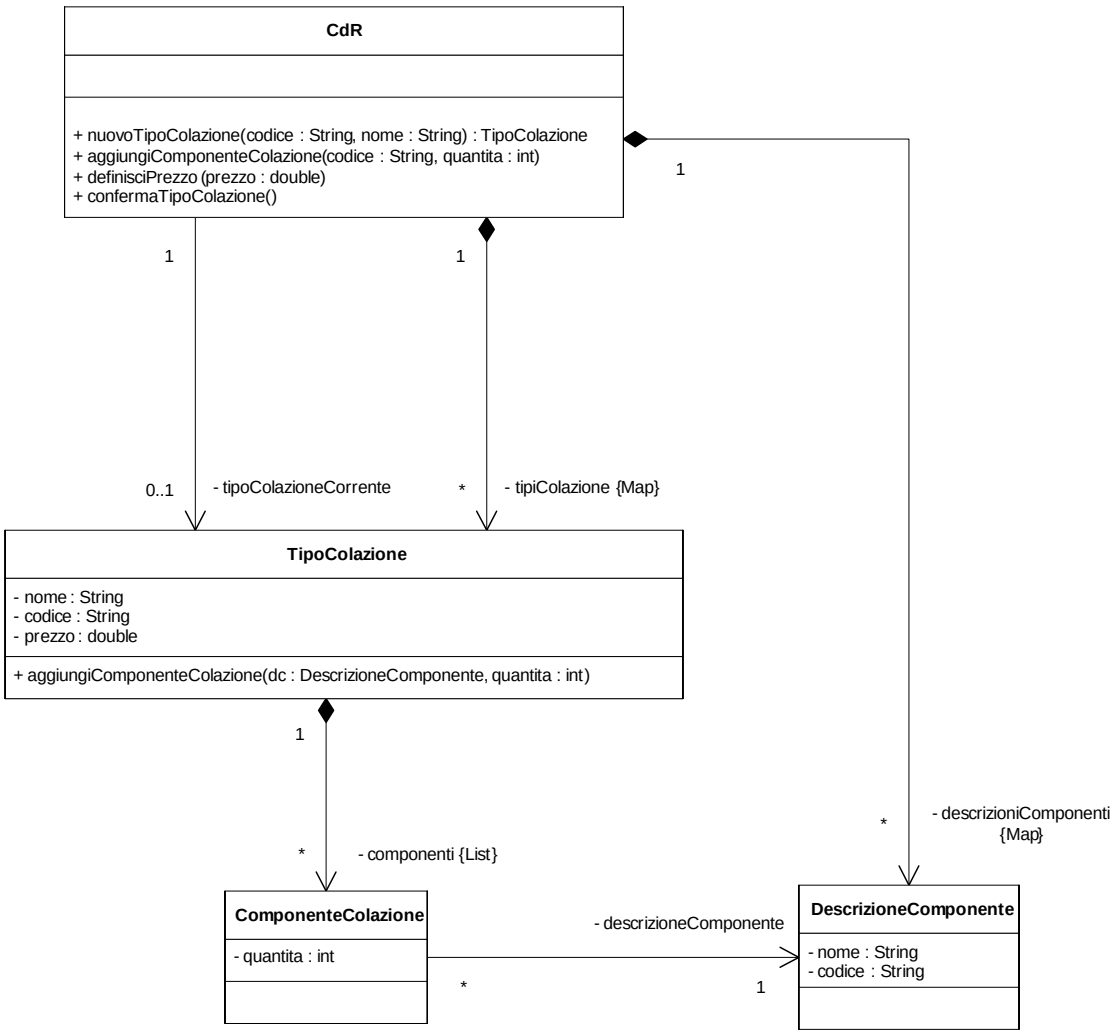

Figura 3.5: DCD

3.4 Caso d'uso UC2, Diagrammi di interazione

3.4.1 nuovoOrdine()

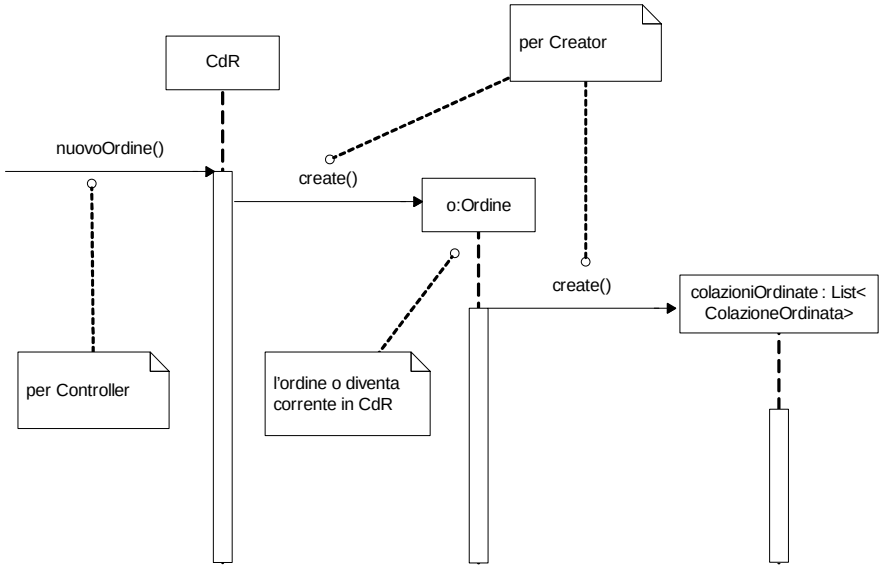

Figura 3.6: OP1: creazione di nuovo ordine

3.4.2 nuovaColazioneOrdinata(codice:String)

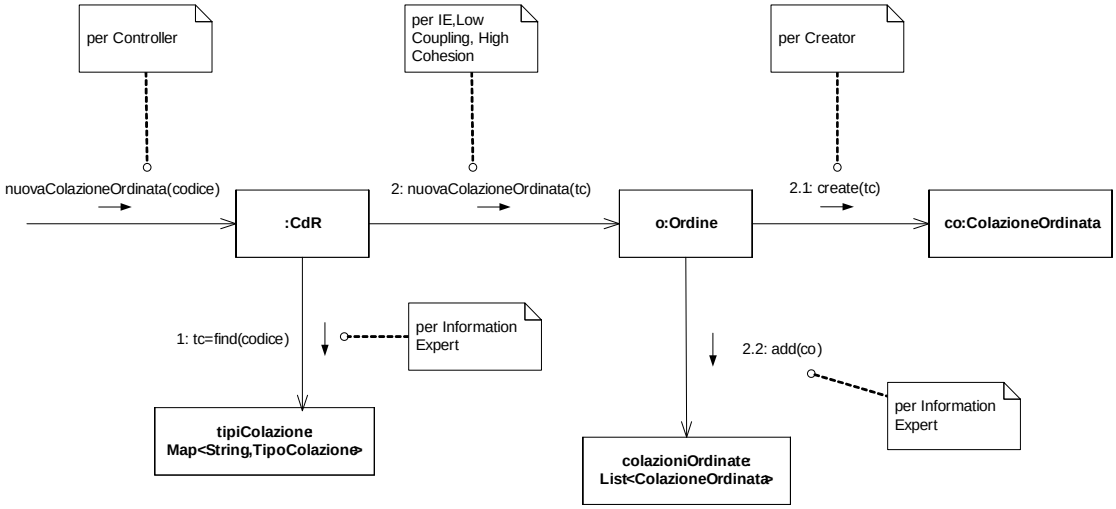

Figura 3.7: OP2: ordinazione di una nuova colazione

3.4.3 definisciModoServizio(codice:String)

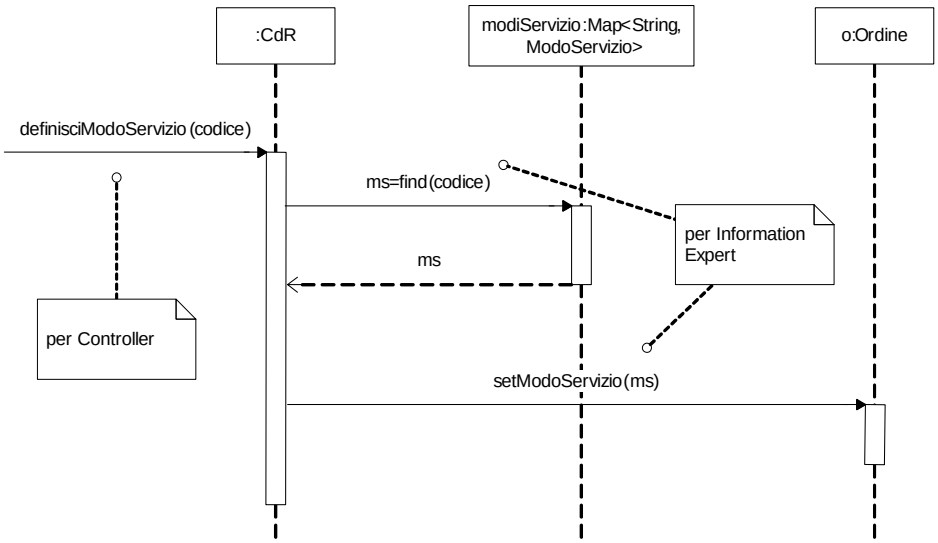

Figura 3.8: OP6: definizione della modalità di servizio

3.4.4 calcolaPrezzoOrdine()

Non si tratta di un'operazione di sistema, ma di un'interrogazione per calcolare il prezzo totale dell'ordine.

Per quest'operazione, Bob propone il seguente diagramma di interazione:

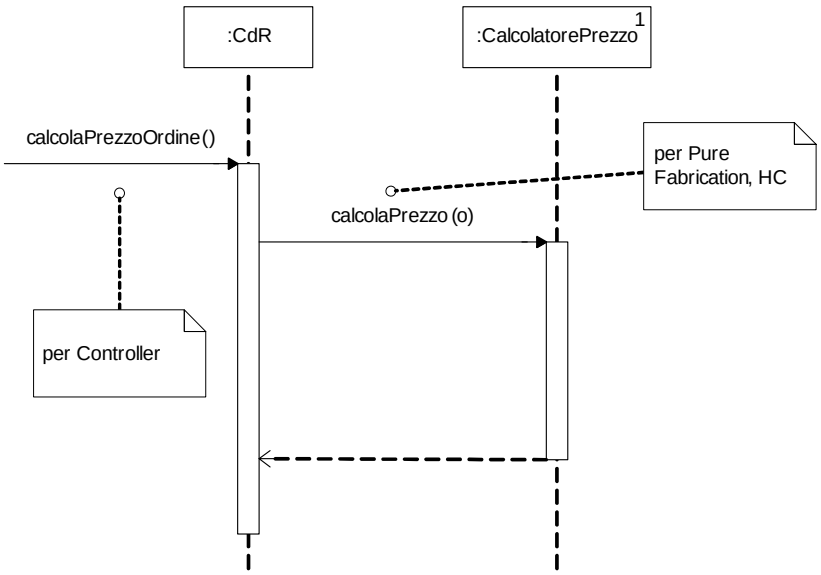

Figura 3.9: Q1: calcolo del prezzo dell'ordine (prima versione)

Alice è assolutamente contraria a questo progetto: "Questo è un errore comune: stai usando una Pure Fabrication per fare una magia, ma la magia è vietata! In pratica, stai dicendo che ti farebbe comodo un oggetto CalcolatorePrezzo, ma non hai idea di come farlo. In effetti, gli stai assegnando una responsabilità (quella di calcolare il prezzo di un ordine) ma non stai dicendo come la soddisfa, ovvero come collabora con gli altri oggetti per soddisfarla. Ora, se pensi a come soddisfare questa responsabilità, ti dovrebbe essere chiaro che bisogna collaborare in primo luogo con l'ordine, e poi l'ordine deve collaborare con altri oggetti che lui conosce. Allora, per Information Expert e per Low Coupling, dovresti capire che questa responsabilità deve essere assegnata proprio all'ordine, e non ad una Pure Fabrication magica."

Pertanto va usato un diagramma di interazione che inizia come mostrato qui di seguito (il progetto dettagliato dell'operazione di interrogazione viene rimandato al momento della codifica).

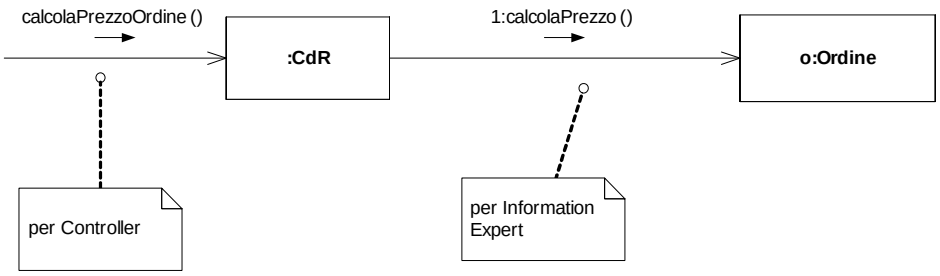

Figura 3.10: Q1: calcolo del prezzo dell'ordine (inizio, seconda versione)

3.4.5 associaOrdineCliente (nome,cognome,indirizzo:String,data:Date,ora:Time)

Se si assume che il cliente sia già noto alla Colazione di Rosa:

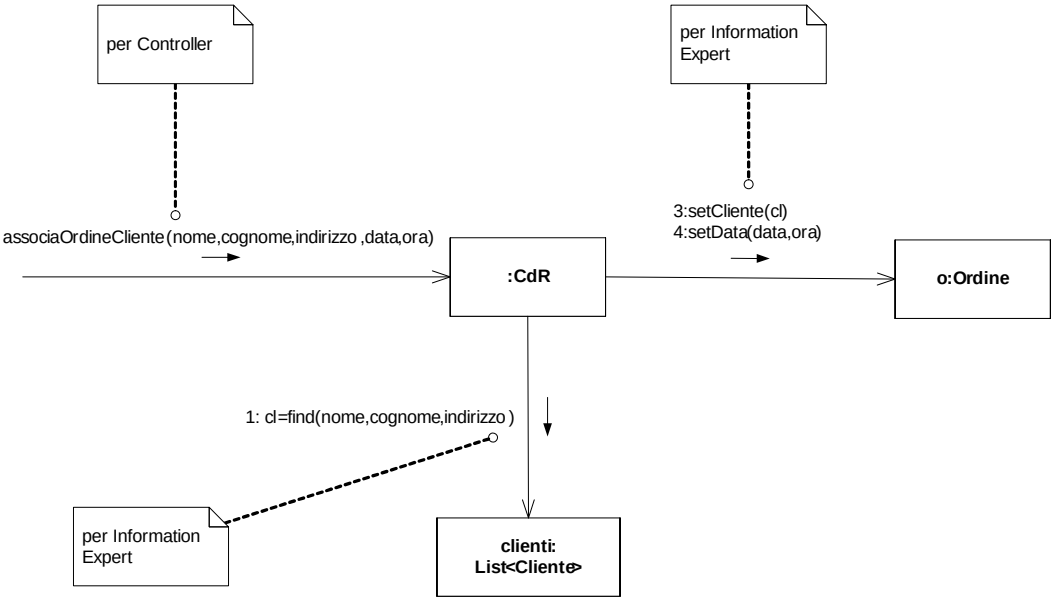

Figura 3.11: OP7: associa cliente all'ordine (cliente noto)

Se si vuole creare un nuovo cliente, nel caso la ricerca del cliente fallisca:

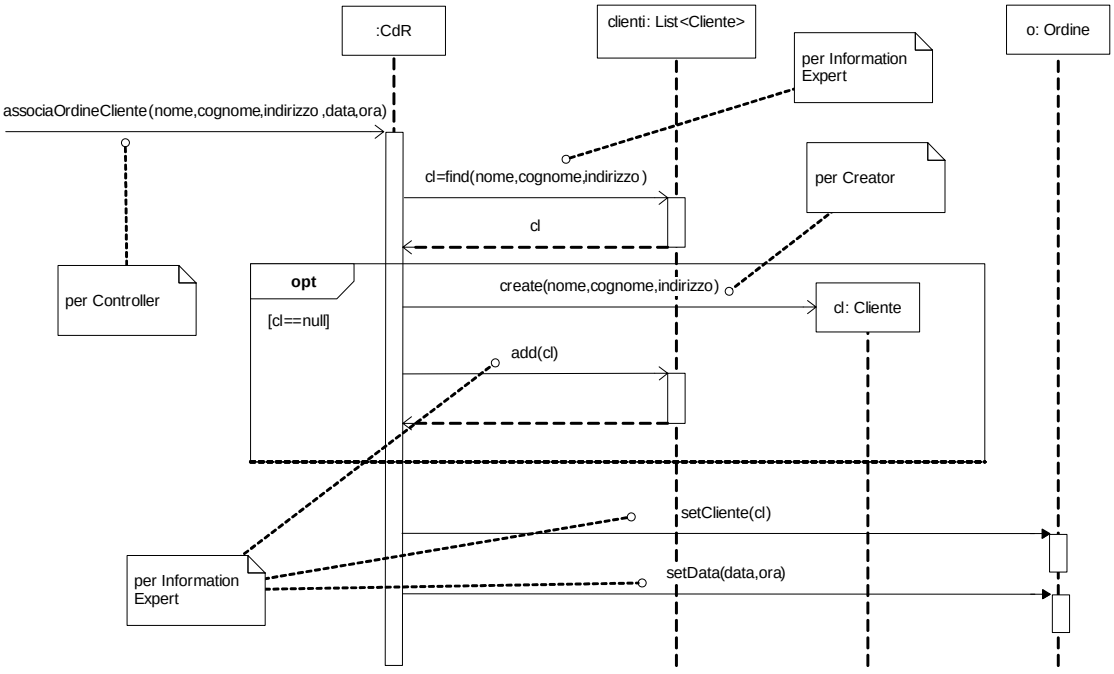

Figura 3.12: OP7: associa cliente all'ordine (con eventuale creazione del cliente)

3.4.6 confermaOrdine()

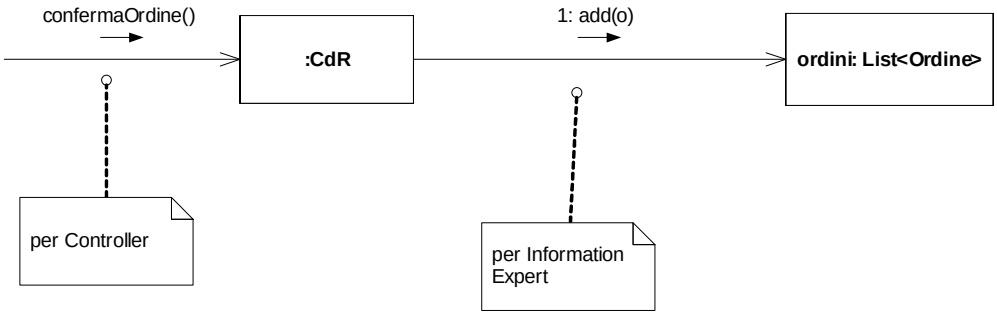

Figura 3.13: OP8: conferma dell'ordine

3.5 Caso d'uso UC2, Diagramma delle classi di progetto

Nel DCD non sono state ripetute le classi di interesse per il solo caso d'uso UC1.

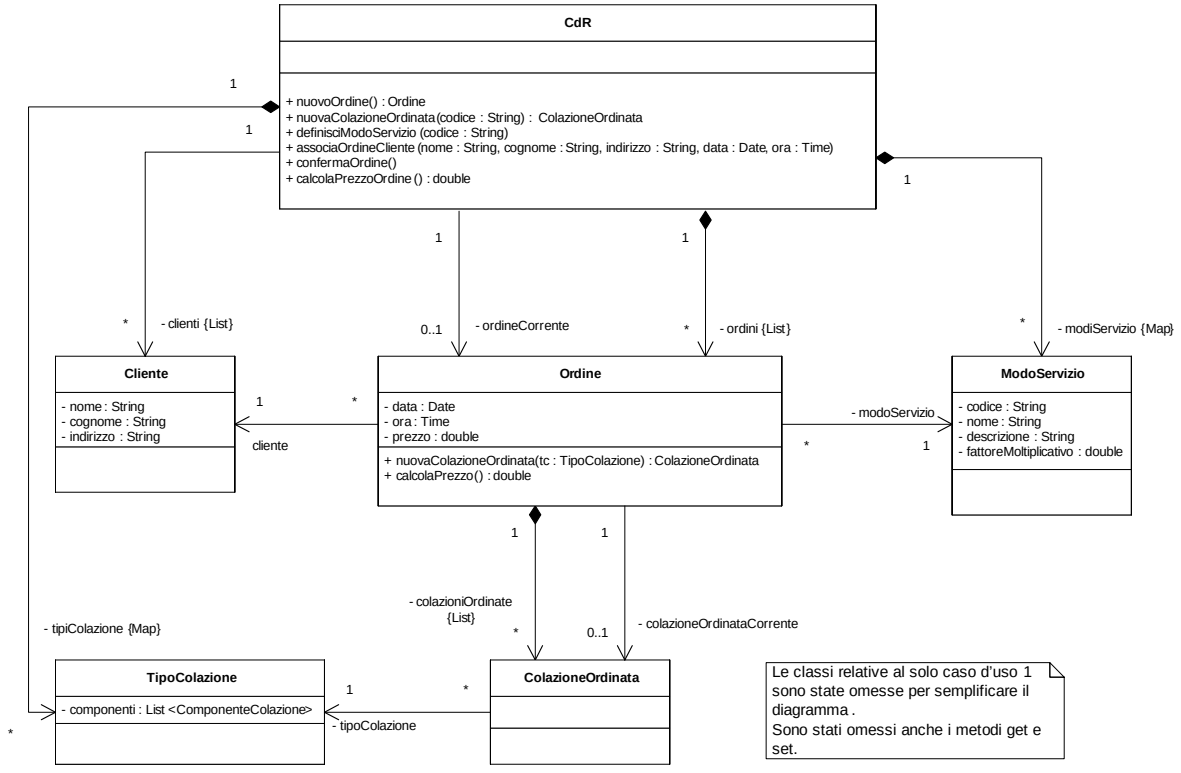

Figura 3.14: DCD

Capitolo 4

Iterazione 1, Refactoring

4.1 Introduzione

Dopo aver scritto il codice per il progetto fatto, Bob si è reso conto che la classe controller CdR ha responsabilità eccessive, che possono essere risolte riducendo il suo carico di lavoro. La scelta più semplice è introdurre una classe Menu e assegnarli la responsabilità di conoscere i tipi di colazione, i componenti delle colazioni nonché i modi di servizio, ovvero le informazioni che ci si aspetta di trovare in un menu cartaceo.

Questa attività viene svolta nel codice, mediante una successione di refactoring che cambiano l'organizzazione del codice senza cambiarne il comportamento complessivo. Nello svolgere questa attività, viene anche scelto di usare il design pattern Singleton (GoF) per il controller, la classe CdR.

Queste attività, volte a "ripulire" il codice, potrebbero essere parte delle fasi finali dell'iterazione 1, oppure parte delle fasi iniziali dell'iterazione 2.

Dal codice così ottenuto, usando la funzionalità di reverse engineering di uno strumento CASE, sono stati prodotti i diagrammi che mostrano il progetto effettivo allo stato attuale. Questo capitolo illustra alcuni di questi diagrammi.

4.2 Caso d'uso UC1

L'aggiunta di una classe Menu, che gestisce in particolare le associazioni verso le classi TipoColazione, Descrizione Componente e ModoServizio, come descritto nei modelli di dominio mostrati nelle figure 2.1 e 2.5, modifica tutte le interazioni relative al caso d'uso UC1, che pertanto verranno ripetute in questa sezione.

4.2.1 nuovoTipoColazione(codice : String, nome : String)

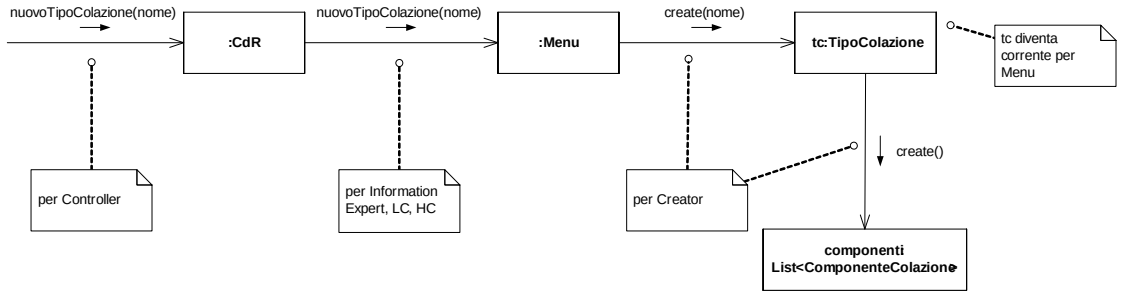

Figura 4.1: OP1: creazione nuovo tipo di colazione

4.2.2 aggiungiComponenteColazione(codice : String, quantità : Integer)

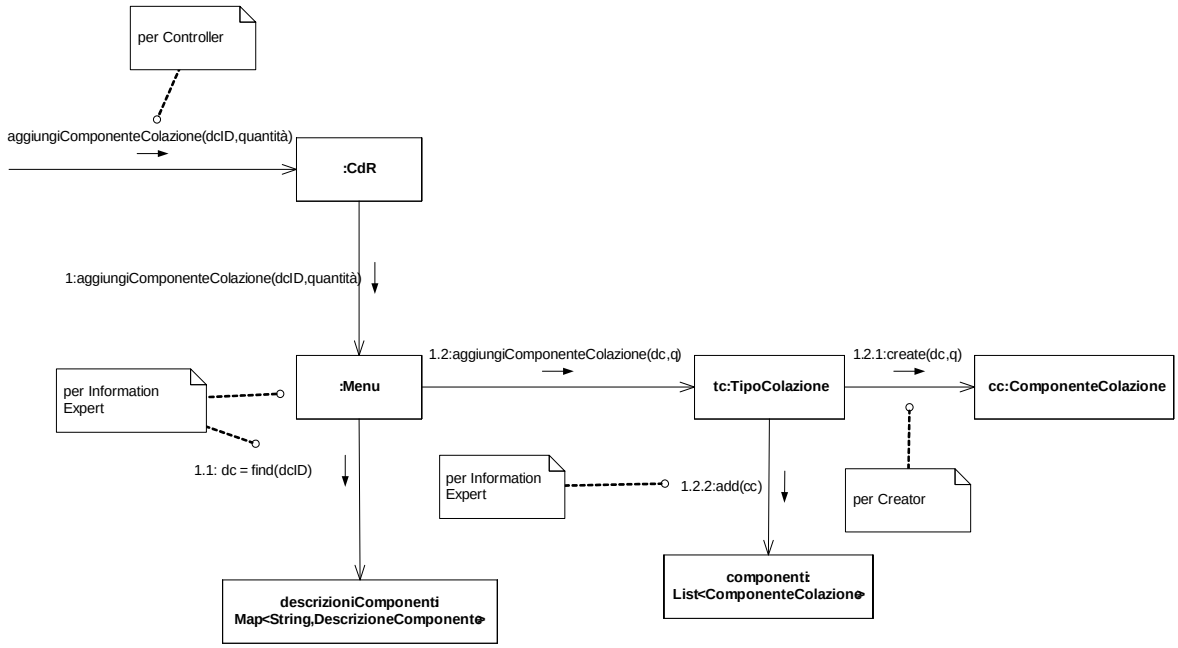

Figura 4.2: OP2: aggiungi componente al tipo di colazione

4.2.3 definisciPrezzo(prezzo:Double)

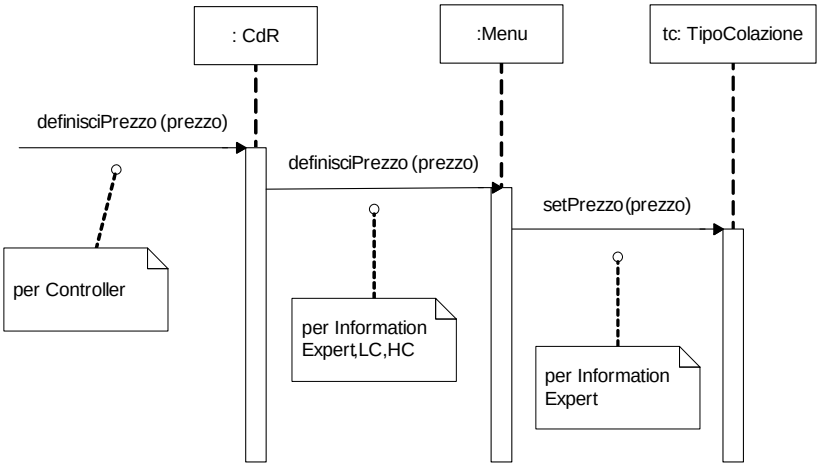

Figura 4.3: OP3: definisci il prezzo del tipo di colazione

4.2.4 confermaTipoColazione()

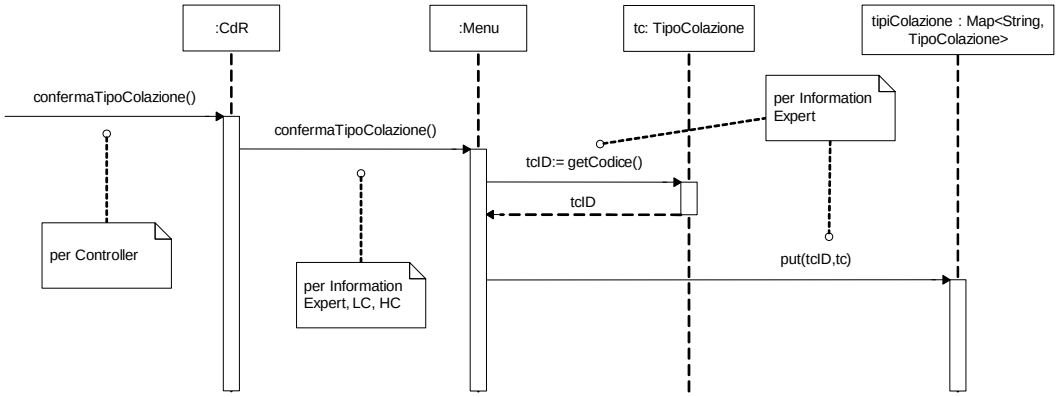

Figura 4.4: OP4: conferma il tipo di colazione

4.2.5 Caso d'uso UC1, Diagramma delle classi di progetto

Poiché Menu viene acceduto come Singleton, la relazione tra CdR e Menu è mostrata come una dipendenza e non come associazione.

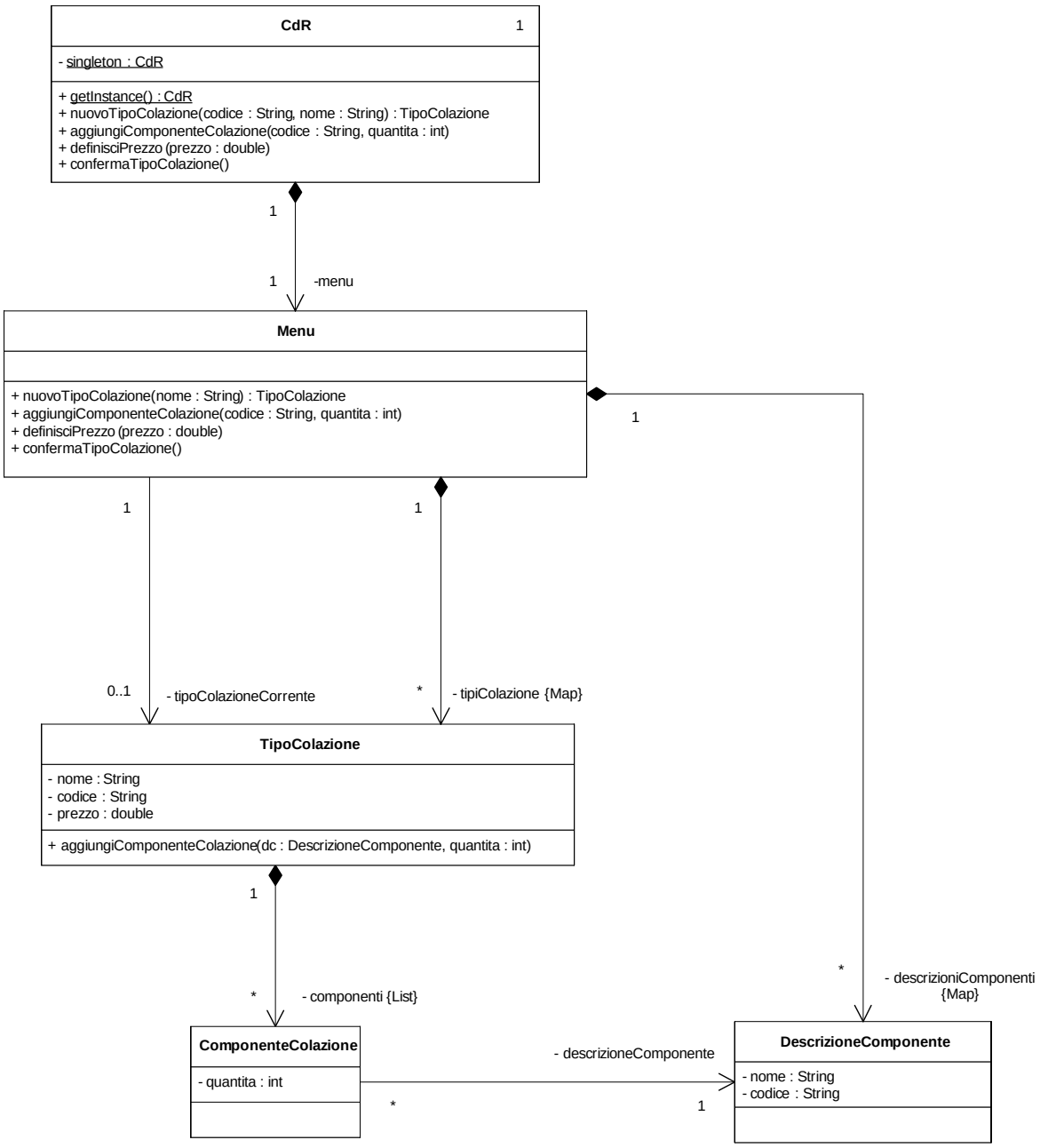

Figura 4.5: DCD

4.3 Caso d'uso UC2

L'aggiunta di una classe Menu ha un impatto decisamente minore sulle operazioni di sistema relative al caso d'uso UC2, molte delle quali rimangono immutate. Cambia invece l'accesso agli oggetti TipoColazione e ModoServizio, che avverrà tramite la classe Menu. Pertanto, i soli diagrammi di interazione coinvolti sono quelli relativi alle operazioni di sistema *nuovaColazioneOrdinata* e *definisciModoServizio*.

4.3.1 nuovaColazioneOrdinata(codice:String)

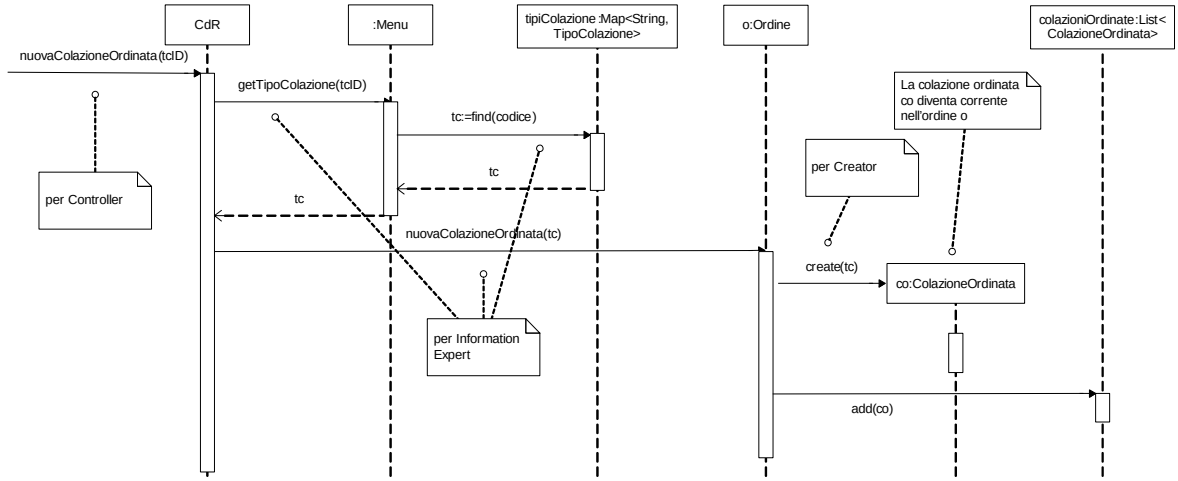

Figura 4.6: OP2: ordinazione di una nuova colazione

4.3.2 definisciModoServizio(codice:String)

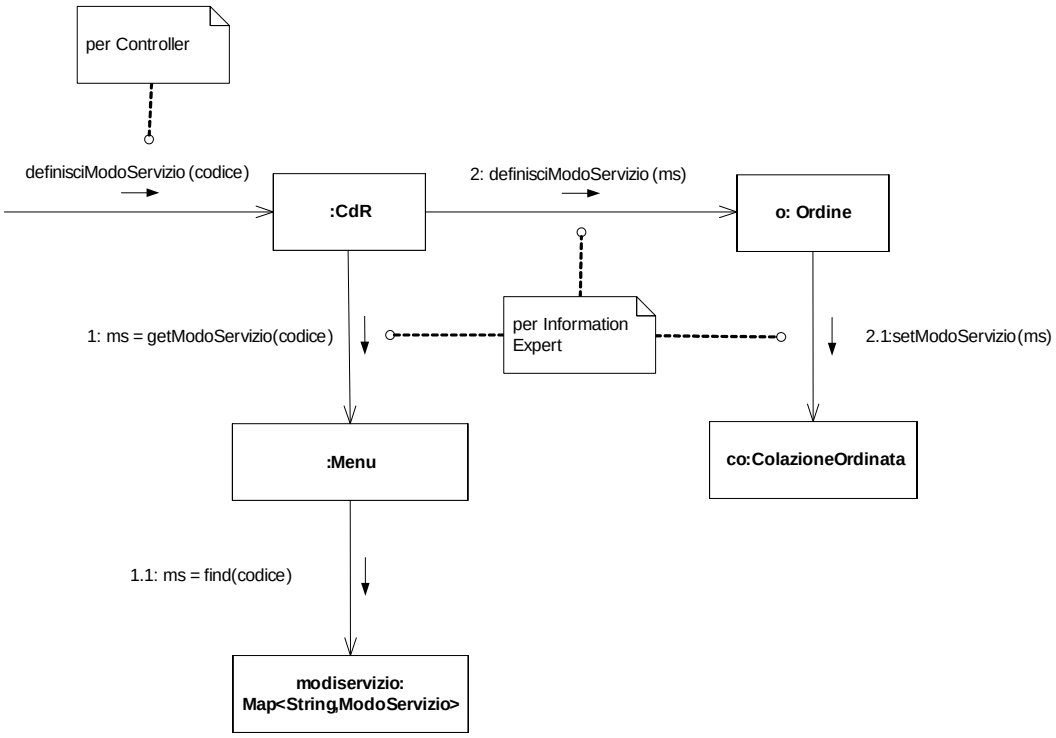

Figura 4.7: OP6: definizione della modalità di servizio

4.3.3 Caso d'uso UC2, Diagramma delle classi di progetto

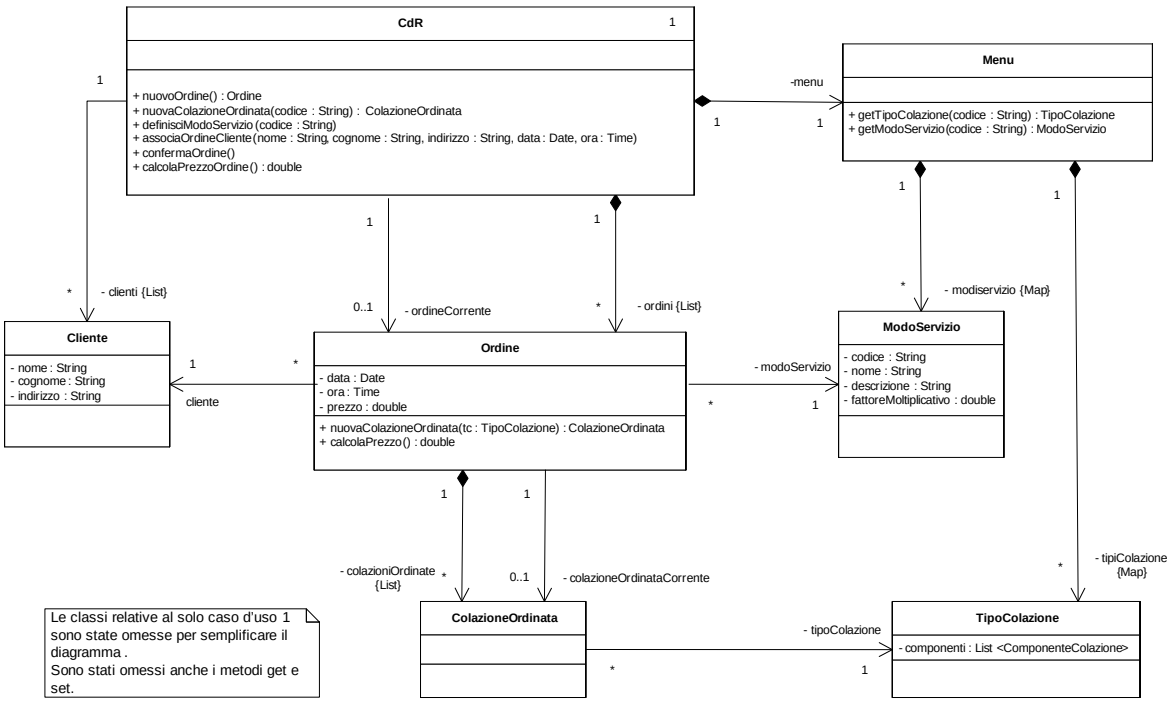

Figura 4.8: DCD

Capitolo 5

Iterazione 2, Analisi

5.1 Introduzione

Per l'iterazione 2, sono stati scelti i seguenti requisiti:

- fare riferimento solo a dati in memoria principale (scelta solo di natura didattica), trascurando ancora il problema della persistenza di oggetti;
- lo scenario principale di successo del caso d'uso UC1 (Inserimento nuovo tipo di colazione) è stato completato nell'iterazione 1, e non si desiderano implementare scenari alternativi;
- si vuole completare lo scenario principale di successo del caso d'uso UC2 (Inserimento nuovo ordine), considerando anche la possibilità di ordinare varianti delle colazioni ordinate;
- nello sviluppo dello scenario principale di successo semplificato per il caso d'uso UC2 (Inserimento nuovo ordine), è stato rilevato un errore di comprensione dei requisiti, che si vuole correggere già in questa iterazione; il caso d'uso deve prevedere che le varie colazioni di uno stesso ordine possano avere modo di servizio diverso (Rosa ci ha raccontato che "la signorina Connie ordina sempre la sua colazione nel modo superiore, ma la colazione per il suo gatto la ordina nel modo normale"); il caso d'uso corretto è riportato più avanti;
- il calcolo del prezzo di un ordine è basato sulle regole di dominio A, B e C.

Questo capitolo descrive i nuovi requisiti per l'iterazione 2, nonché l'aggiornamento dell'analisi relativa al caso d'uso UC2.

5.2 Casi d'uso

Ecco il caso d'uso UC2 aggiornato.

Caso d'uso UC2:
Inserimento nuovo ordine

Attore primario: un Addetto del sistema.

1. Il Cliente telefona a *La Colazione di Rosa* per ordinare delle colazioni.
2. L'Addetto inizia un nuovo ordine.
3. Il Cliente dice quale tipo di colazione vuole ordinare, e l'Addetto ne inserisce il codice identificativo. Il Sistema registra la colazione richiesta e mostra la composizione della colazione scelta.

Il passo 4 sarà ripetuto finché serve.

4. Il Cliente specifica un componente della colazione richiesta da modificare o rimuovere, indicando la quantità desiderata. L'Addetto inserisce il codice identificativo del componente e la nuova quantità richiesta (zero per cancellare). Il Sistema registra la modifica alla colazione richiesta.

Il passo 5 sarà ripetuto finché serve.

5. Il Cliente specifica un componente (che non è previsto nella colazione richiesta) da aggiungere, indicando la quantità desiderata. L'Addetto inserisce il codice identificativo del componente e la quantità richiesta. Il Sistema registra la modifica alla colazione richiesta.
6. Il Cliente specifica il modo con cui deve essere servita la colazione richiesta. L'Addetto inserisce il codice identificativo del modo di servizio richiesto. Il Sistema registra il modo di servizio richiesto.

I passi da 3 a 6 vengono ripetuti finché serve, per richiedere altre colazioni nell'ambito dello stesso ordine.

7. Il Sistema calcola e mostra il prezzo delle colazioni ordinate (applicando l'algoritmo opportuno, vedi le *Regole di dominio*). L'Addetto comunica il prezzo al Cliente e ottiene conferma dell'ordine.
8. Il Cliente comunica il proprio nome e cognome, nonché la data, l'ora e l'indirizzo della consegna per le colazioni richieste. L'Addetto inserisce queste informazioni nel Sistema. Il Sistema registra le informazioni sul cliente e mostra il riepilogo delle informazioni sull'ordine.
9. L'Addetto conferma l'ordine. Il Sistema registra l'ordine. L'addetto saluta il Cliente e chiude la telefonata.

5.3 Modello di dominio

Ecco il modello di dominio aggiornato, che tiene in considerazione sia il caso d'uso UC1 che la nuova versione del caso d'uso UC2.

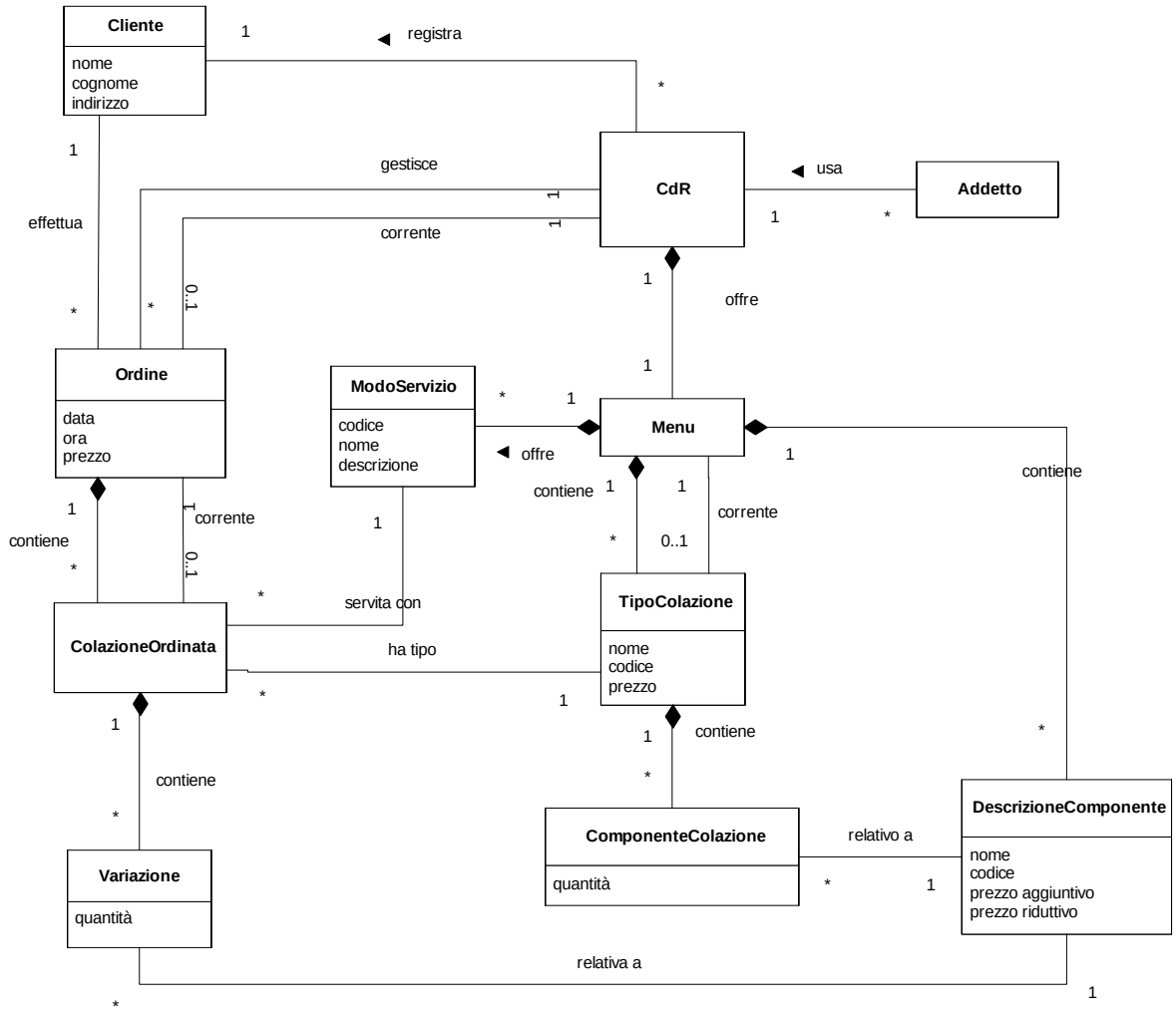

Figura 5.1: Modello di dominio

La seguente figura mostra un ordine relativo ad una colazione semplice e una colazione francese. La colazione semplice (normalmente un cappuccino e due cornetti) è stata richiesta per avere un cappuccino, un solo cornetto ed anche un caffè. La colazione semplice (con variazioni) è stata richiesta nel modo di servizio normale, quella francese nel modo superiore.

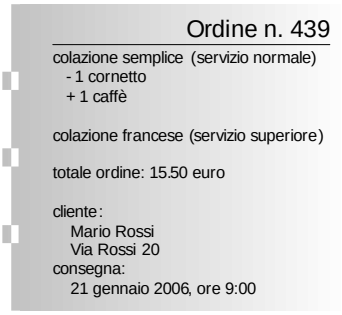

Figura 5.2: Un ordine (con variazioni)

Il seguente modello degli oggetti di dominio mostra la rappresentazione ad oggetti di quest'ordine. Gli oggetti *Variazione* sono usati per rappresentare le variazioni rispetto alle composizioni standard. Si noti anche come gli oggetti *ModoServizio* siano associati alle singole colazioni ordinate, e non più all'intero ordine.

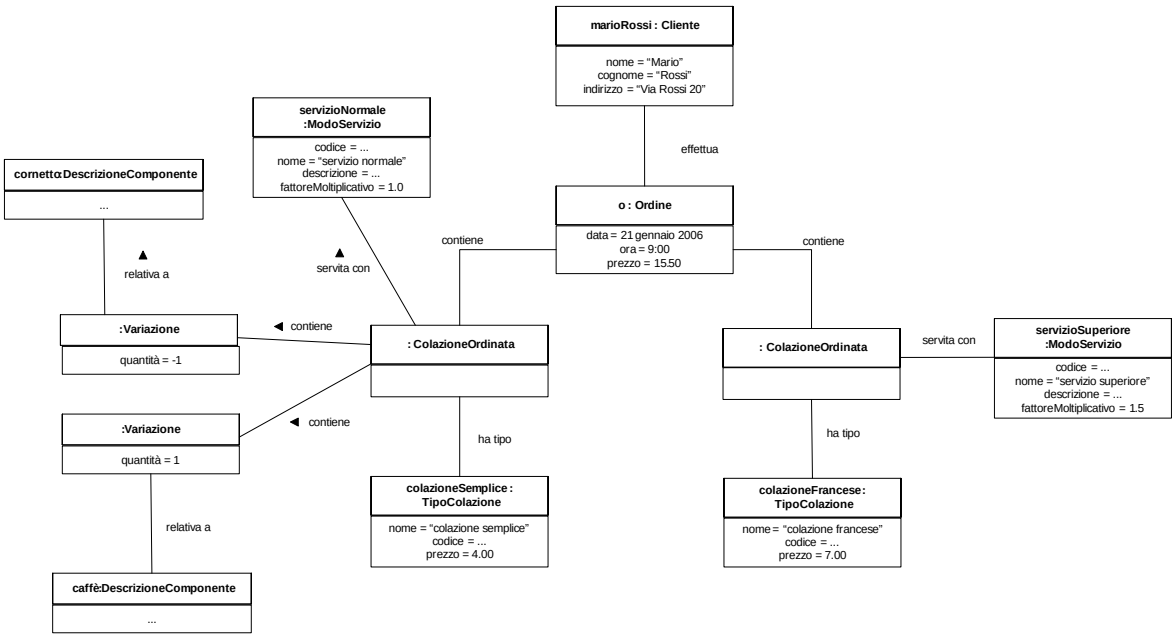

Figura 5.3: Oggetti di dominio per un ordine con variazioni

5.4 Diagrammi di sequenza di sistema

Il diagramma di sequenza di sistema per il caso d'uso UC1 rimane inalterato. Ecco l'aggiornamento del diagramma di sequenza di sistema per il caso d'uso UC2.

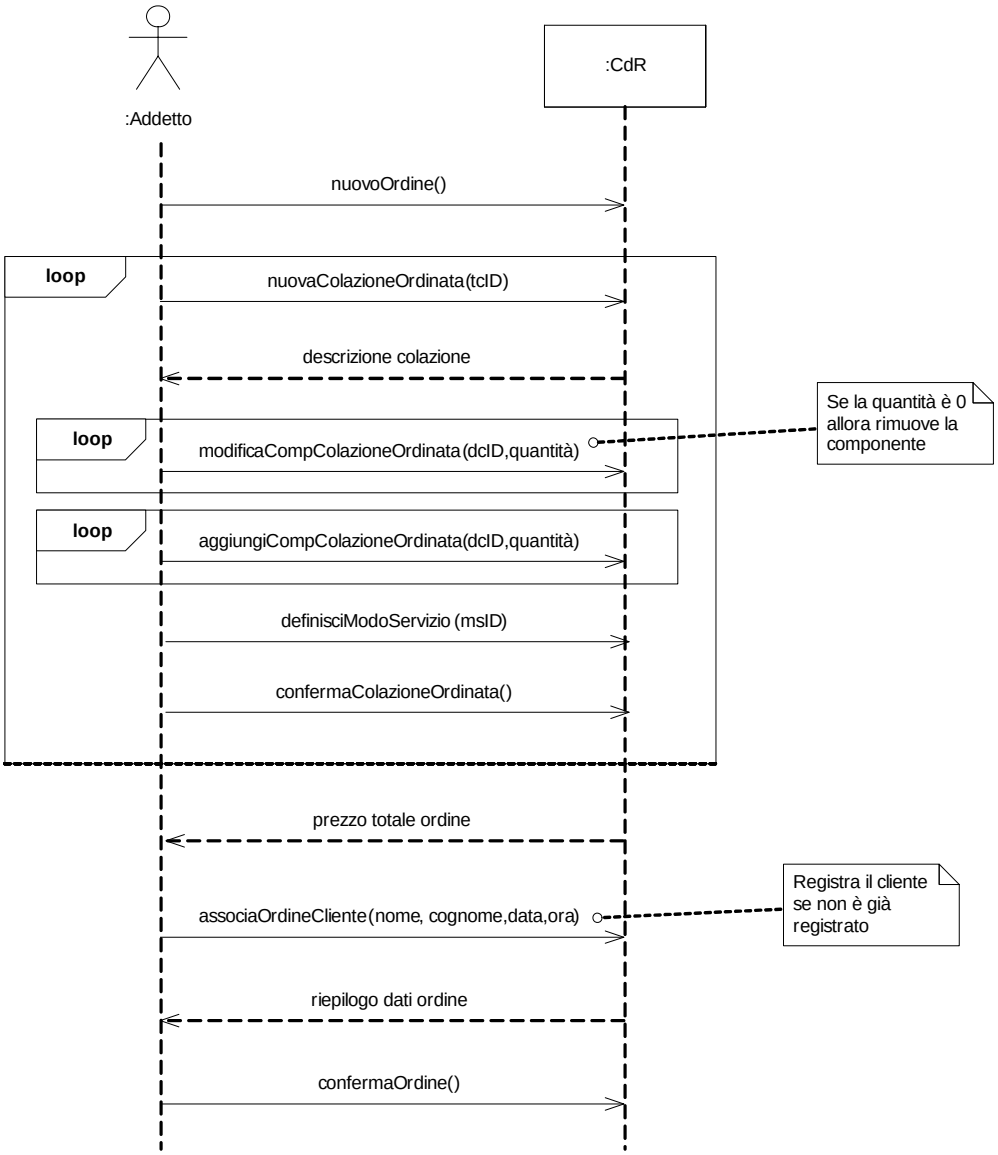

Figura 5.4: UC2: diagramma di sequenza di sistema

5.5 Contratti delle operazioni

Anche in questo caso, nessun aggiornamento va fatto relativamente ai contratti per le operazioni di sistema del caso d'uso UC1.

Relativamente al caso d'uso UC2, invece, ci sono delle nuove operazioni di sistema (*modificaCompColazioneOrdinata* e *aggiungiCompColazioneOrdinata*) ed altre sono cambiate (*nuovaColazioneOrdinata* e *definisciModoServizio*). Di seguito vengono riportati i contratti per tali operazioni di sistema. I contratti delle altre operazioni (*nuovoOrdine*, *associaOrdineCliente* e *confermaOrdine*) rimangono invece inalterati.

Segue l'aggiornamento dei contratti delle operazioni. Sono stati riportati i contratti relativi alle sole nuove operazioni di sistema, poiché che quelli scritti in precedenza sono ancora validi (ad eccezione di quello relativo alle operazioni di sistema *nuovaColazione* e *definisciModoServizio*).

5.5.1 Nuova Colazione Ordinata

L'operazione di sistema *nuovaColazioneOrdinata* consente di aggiungere una nuova colazione all'ordine corrente.

Operazione	nuovaColazioneOrdinata(codice:String)
Riferimenti	Caso d'uso: *Inserimento nuovo ordine*
Pre-condizioni	- è in corso l'inserimento di un Ordine *o*
Post-condizioni	- è stata creata una nuova istanza *co* di ColazioneOrdinata; - gli attributi di *co* sono stati inizializzati; - la colazione *co* è stata associata a un TipoColazione *tc*, sulla base del codice, tramite l'associazione "ha tipo"; - la ColazioneOrdinata *co* è stata associata all'Ordine corrente *o* tramite l'associazione "contiene"; - la ColazioneOrdinata *co* è stata associata all'Ordine corrente *o* tramite l'associazione "corrente".

5.5.2 Aggiungi componente a colazione ordinata

L'operazione di sistema *aggiungiCompColazioneOrdinata* consente di aggiungere un nuovo componente alla colazione ordinata corrente dell'ordine corrente.

Operazione	aggiungiCompColazioneOrdinata(codice:String, quantità:Integer)
Riferimenti	Caso d'uso: *Inserimento nuovo ordine*
Pre-condizioni	- è in corso l'inserimento di una ColazioneOrdinata *co* nell'ambito dell'Ordine *o* corrente
Post-condizioni	- è stata creata una nuova istanza *v* di Variazione; - l'attributo *quantità* di *v* è divenuto uguale a *quantità*; - *v* è stata associata alla colazione ordinata corrente *co* tramite l'associazione "contiene"; - *v* è stata associata a una DescrizioneComponente *dc* sulla base del codice, tramite l'associazione "relativa a".

5.5.3 Modifica componente a colazione ordinata

L'operazione di sistema *modificaCompColazioneOrdinata* consente di modificare la quantità di una componente nella colazione corrente dell'ordine corrente.

Questa operazione di sistema differisce dall'operazione *aggiungiCompColazioneOrdinata* in quanto la quantità della variazione è data dalla differenza tra il parametro quantità e la quantità base nella colazione scelta.

Operazione	modificaCompColazioneOrdinata(codice:String, quantità:Integer)
Riferimenti	Caso d'uso: *Inserimento nuovo ordine*
Pre-condizioni	- è in corso l'inserimento di una ColazioneOrdinata *co* nell'ambito dell'Ordine *o* corrente
Post-condizioni	- è stata creata una nuova istanza *v* di Variazione; - l'attributo *quantità* di *v* è divenuto uguale a *quantità* meno la quantità della descrizione componente scelta nella colazione scelta; - *v* è stata associata alla colazione ordinata corrente *co* tramite l'associazione "contiene"; - *v* è stata associata a una DescrizioneComponente *dc* sulla base del codice, tramite l'associazione "relativa a".

5.5.4 Definisci modo servizio

L'operazione di sistema *definisciModoServizio* consente di associare un modo di servizio alla colazione ordinata corrente (nell'ambito di un ordine, il modo di servizio può variare da colazione a colazione).

Operazione	definisciModoServizio(codice:String)
Riferimenti	Caso d'uso: *Inserimento nuovo ordine*
Pre-condizioni	- è in corso l'inserimento di una ColazioneOrdinata *co* nell'ambito dell'Ordine *o* corrente
Post-condizioni	- è stata associata l'istanza *co* di ColazioneOrdinata corrente a *ms* tramite l'associazione "servita con".

Capitolo 6

Iterazione 2, Progettazione

6.1 Introduzione

Nessuna modifica è richiesta al progetto relativamente al caso d'uso UC1. Viceversa, per il caso d'uso UC2 è necessario progettare per le nuove operazioni di sistema (*modificaCompColazioneOrdinata* e *aggiungiCompColazioneOrdinata*) e per le operazioni di sistema modificate (*nuovaColazioneOrdinata* e *definisciModoServizio*). Nessuna modifica è invece richiesta per le altre operazioni di sistema (*nuovoOrdine*, *associaOrdineCliente* e *confermaOrdine*).

6.2 Diagrammi di interazione

6.2.1 nuovaColazioneOrdinata(tcID:String)

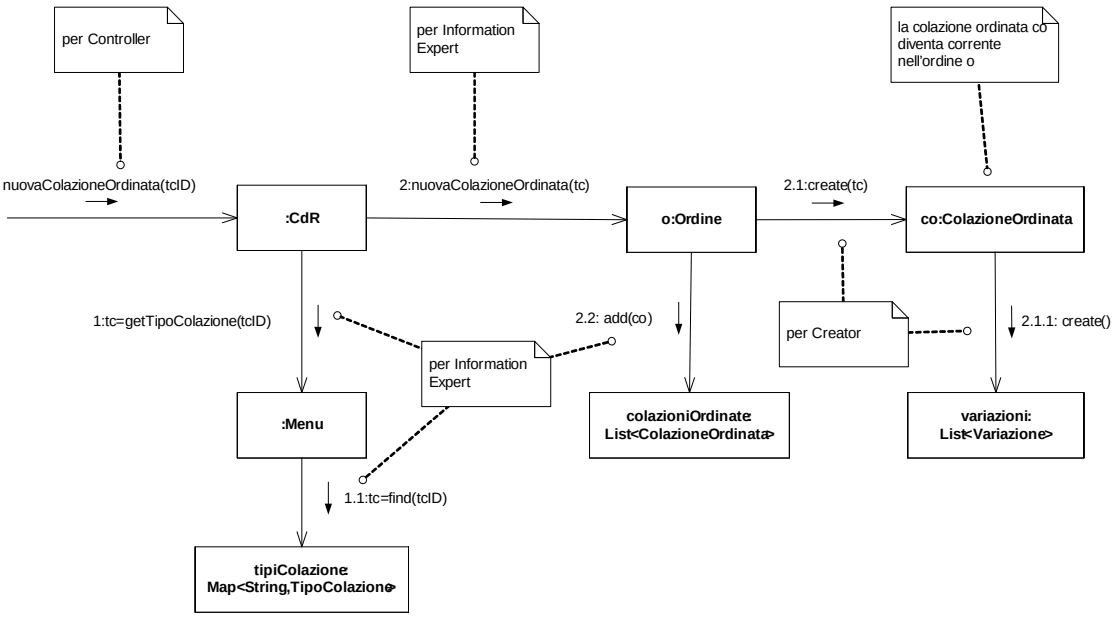

Figura 6.1: OP2: ordinazione di una nuova colazione

6.2.2 aggiungiCompColazioneOrdinata (dcID:String, quantità:Integer)

Per realizzare questa operazione bisogna creare una Variazione associata alla Descrizione-Componente selezionata, la cui quantità è uguale al parametro quantità.

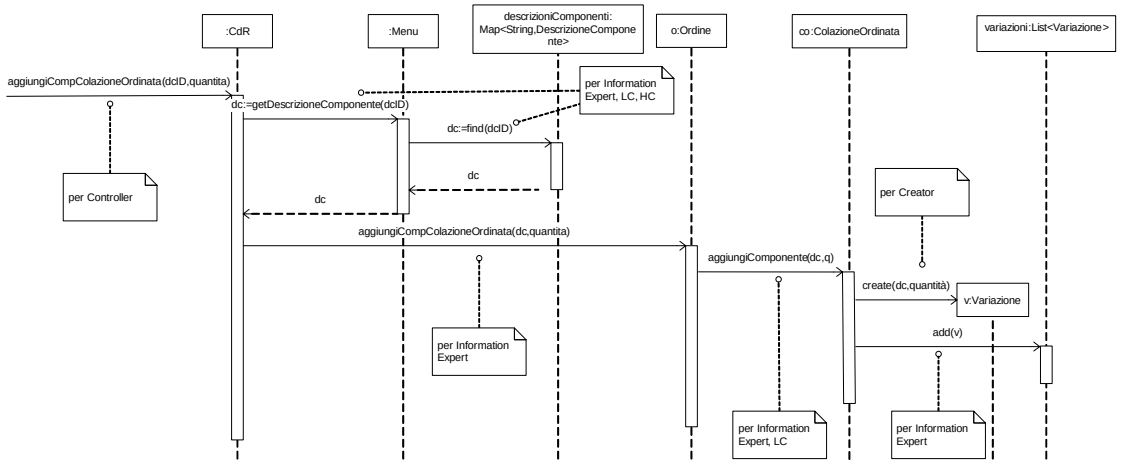

Figura 6.2: OP4: aggiunta di una componente di una colazione ordinata

6.2.3 modificaCompColazioneOrdinata (dcID:String, quantità:Integer)

Per realizzare questa operazione bisogna creare una Variazione associata alla DescrizioneComponente selezionata, la cui quantità è uguale alla differenza tra il parametro quantità e la quantità della DescrizioneComponente nel TipoColazione.

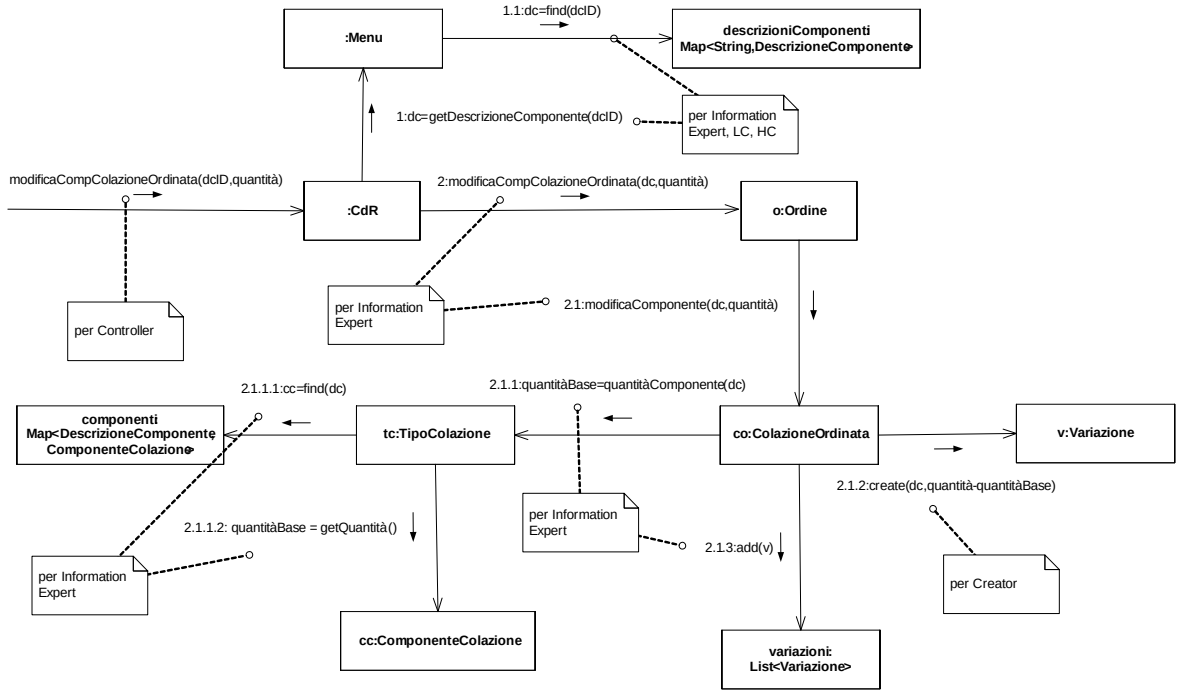

Figura 6.3: OP3: modifica di una componente di una colazione ordinata

6.2.4 definisciModoServizio(msID:String)

Il ModoServizio non va più associato all'Ordine, ma piuttosto alla ColazioneOrdinata.

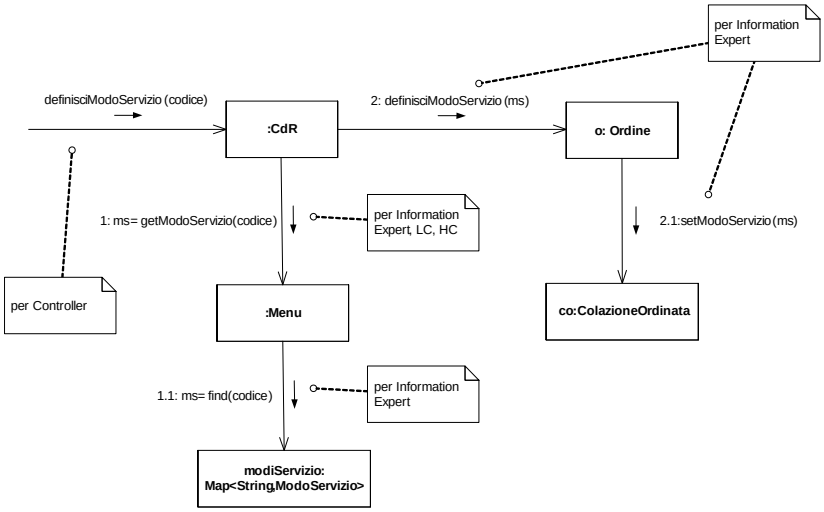

Figura 6.4: OP5: definizione della modalità di servizio

6.3 Diagramma delle classi di progetto

Ecco il diagramma delle classi di progetto per l'iterazione 2, relativo sia al caso d'uso UC1 che al caso d'uso UC2.

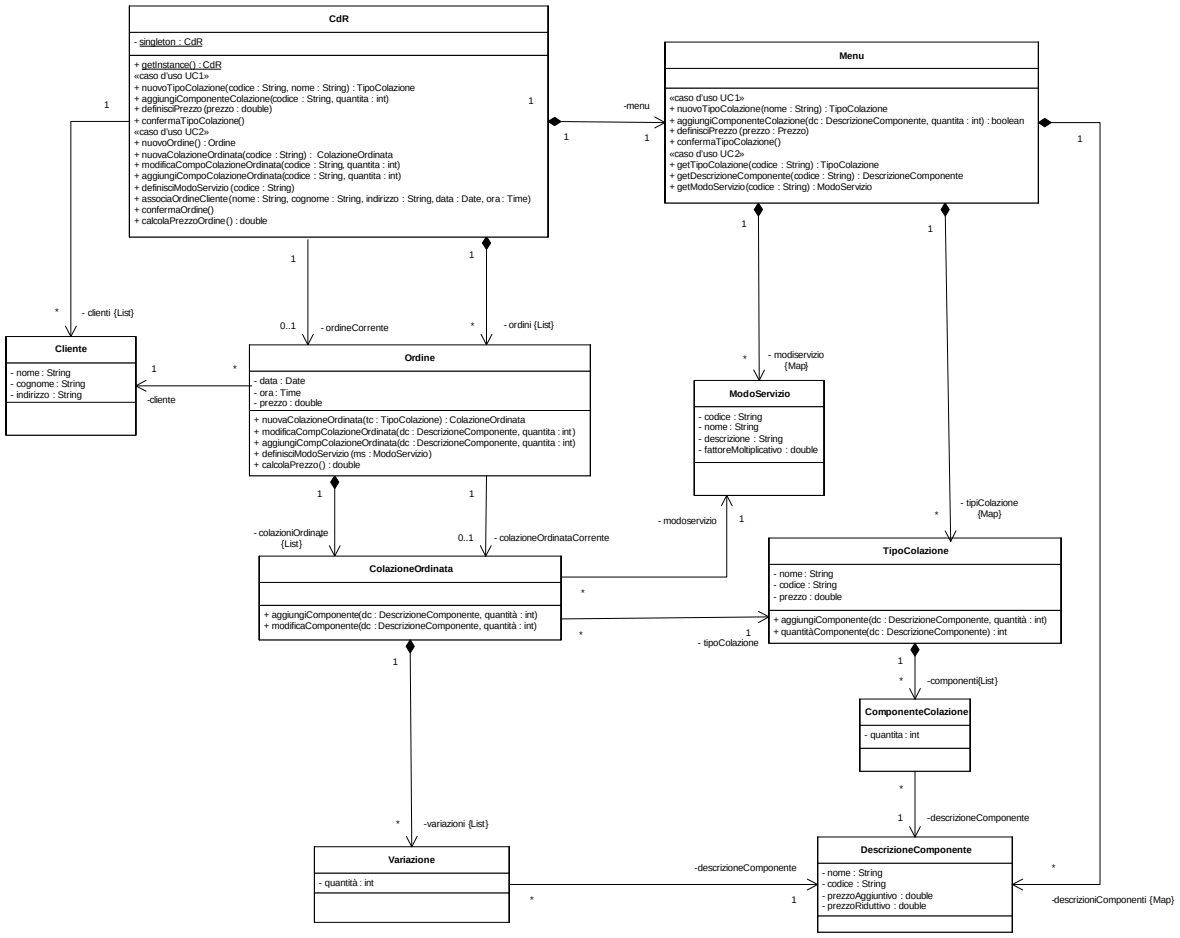

Figura 6.5: DCD