Capitolo 4

Modellazione avanzata degli Use Case

Uno dei peggiori sprechi di energie che si possa produrre è realizzare un sistema, magari impeccabile dal punto di vista tecnico, che però semplicemente non è quello di cui aveva bisogno l'utente.

Cliente: "Vorrei un mezzo di trasporto in grado di teletrasportare persone da Roma a NY in 5 minuti"…
Architetto: "OK, si può fare… Sono necessari 10^3 anni-uomo e 10^{20} dollari".

Introduzione

Nel capitolo precedente sono stati illustratati in dettaglio i diagrammi dei casi d'uso e in particolare si è visto come siano lo strumento ideale per modellare la proiezione statica dell'omonimo modello (uno dei manufatti principali dell'analisi dei requisiti utente).

Obiettivo del presente capitolo è illustrarne un utilizzo più operativo anche se, a volte, sensibilmente distante dalle direttive standard: ancora una volta il conflitto tra pratica e teoria. Modellare sistemi reali mette in luce una serie di problemi e lacune raramente trattati nei libri e nelle specifiche ufficiali dello UML.

Nel presente capitolo viene riportato un esempio completo di un modello dei casi d'uso di un sistema per il commercio elettronico. Logica conseguenza è che il capitolo risulta decisamente corposo… Il lato positivo è che non è assolutamente necessario studiare attentamente tutto l'esempio.

Il pregio che rende particolarmente attraenti gli use case probabilmente risiede nel formalismo amichevole e nella notazione grafica quasi banale: tutto viene rappresentato — a meno di stereotipizzazioni — attraverso omini stilizzati, ellissi e opportune relazioni. Il problema è che spesso tale pregio si tramuta in difetto. La relativa semplicità spinge tecnici a cimentarsi con la notazione dei casi d'uso senza possederne un'adeguata conoscenza. Il risultato è che si commettono tanti errori, molti dei quali ben noti. Esempio tipico è un utilizzo "procedurale": si utilizza il formalismo dei casi d'uso, ma con una metodologia riconducibile alla famosa analisi strutturata. In altre parole si dà luogo a una decomposizione funzionale, si realizzano use case via via più raffinati fino a giungere a dettagli implementativi: in sostanza le descrizioni dei singoli casi d'uso risultano essere delle vere e proprie procedure, i vari use case vengono connessi a catena: use case A include use case B, che include use case C e così via. La principale anomalia generata da tale approccio distorto è relativa al fatto che si disegna il sistema con un formalismo inadeguato in una fase in cui non si dispone di una sufficiente conoscenza del sistema.

La notazione dei casi d'uso, sebbene possa sembrare molto semplice o addirittura banale, richiede un adeguato studio teorico e molta sperimentazione pratica, al fine di riuscire a realizzare modelli validi di use case.

I diagrammi dei casi d'uso permettono di illustrare le singole funzionalità del sistema, conferendo particolare risalto alla percezione che ne ha l'utente.

Il problema però è che talune volte si tenta di utilizzare strumenti per usi non propri; nel caso degli use case, è diffusa l'opinione che essi siano in grado di catturare ogni dettaglio della vista dei casi d'uso. Nulla di più inesatto: infatti non possiedono alcuna caratteristica intrinseca che li renda in grado di rilevare il comportamento dinamico; è inoltre necessario ricorrere a diversi artifici per tentare di evidenziare eventuali condizioni anomale; non è possibile specificare le azioni da intraprendere per la relativa gestione, e così via.

Una prima soluzione a questi inconvenienti è l'utilizzo della tecnica dei flussi di azione, così come prescritto dalle direttive dei Tres Amigos (*cfr* Capitolo 3).

Opinione comune in molti tecnici — compreso l'autore ovviamente — è che nelle pratica quotidiana anche questo formalismo non risulta sempre adeguato e privo di lacune; per esempio non vengono menzionate pre- e postcondizioni, probabilmente non viene conferita abbastanza enfasi agli attori coinvolti nel caso d'uso, non vengono specificate le garanzie, il tempo stimato per l'esecuzione dello use case e così via.

Un'ulteriore alternativa potrebbe essere quella di ricorrere ad altri diagrammi messi a disposizione dallo UML per la modellazione del comportamento dinamico del sistema (diagrammi di sequenza, di attività, ecc). In sostanza si potrebbero utilizzare questi diagrammi, selezionando di volta in volta quello in grado di evidenziare più adeguatamente l'aspetto (il tempo, l'organizzazione il parallelismo, ecc.) ritenuto più importante per la particolare interazione attore/sistema oggetto di studio. Dovendo descrivere il comportamento dinamico di casi d'uso, il livello di astrazione dei diagrammi realizzati dovrebbe

essere necessariamente molto elevato ricordando che l'obiettivo è descrivere **cosa** il sistema dovrà fare e **non** il **come**: bisogna impegnarsi a non guardare il sistema al suo interno.

L'utilizzo dei diagrammi succitati però non solo non risolve tutte le lacune evidenziate per i flussi di azione, ma anzi aggiungerebbe altri inconvenienti, quali per esempio l'aumento significativo del tempo necessario per realizzare i diagrammi stessi, l'estrema difficoltà di manutenzione.

L'esperienza quotidiana suggerisce che, molto spesso, lo strumento più opportuno da utilizzarsi per dettagliare il comportamento dinamico dei casi d'uso rimane il buon vecchio template: semplice, veloce da realizzare e da far comprendere ai clienti, agevole da manutenere e così via.

Ai lettori "l'ardua sentenza".

Template

In questa sezione viene presentato un primo esempio di template dimostratosi particolarmente efficace nella documentazione della proiezione dinamica degli use case. La versione illustrata è frutto di una lunga genesi costituita da continui miglioramenti resisi necessari in corso d'uso, sia per saturare le lacune evidenziate dall'utilizzo delle versioni precedenti, sia per includere suggerimenti ricavati dallo studio di vari testi (quali per esempio [BIB10]).

Nel Capitolo 7 verrà presentato un modello a oggetti formale del template presentato successivamente, realizzato attraverso i diagrammi delle classi. Al termine della lettura dei paragrafi dedicati all'esposizione del template, l'esame del relativo modello a oggetti potrebbe concorrere a chiarire le idee.

In prima analisi, il modello (template) riportato in tab. 4.1 può essere suddiviso in quattro macrosezioni: l'intestazione, le informazioni generali, gli scenari (che a loro volta si dividono in: principali, alternativi e di errore) e la sezione per eventuali annotazioni.

Nell'intestazione è necessario riportare un codice simbolico univoco da utilizzarsi per riferirsi al caso d'uso, una descrizione breve (il nome), la data dell'ultima modifica e la versione.

Il codice si rivela particolarmente utile sia per evidenti questioni di catalogazione e facilità di reperimento, sia nell'utilizzo di tool per la produzione di diagrammi UML: è automatico associare al nome del diagramma il codice del caso d'uso.

Durante le primissime fasi di studio dei requisiti utente e le relative stesure dei casi d'uso è piuttosto normale che non si abbiano le idee completamente chiare, spesso anche per via di un certo smarrimento tipico di alcuni clienti in merito alle funzionalità che il caso d'uso dovrà fornire, alle relative modalità, e quindi, in ultima analisi, ai suoi reali obiettivi. Spesso vi è una certa difficoltà iniziale nel definire i confini dei casi d'uso; ciò non deve preoccupare più di tanto in quanto di solito la versione definitiva di uno use case è il risultato di un certo numero di revisioni. Anche a sistema rilasciato si può assistere a

Tabella 4.1 — *Esempio di template*

CASO D'USO: *Codice*	*Nome del caso d'uso.*	**Data:** / **Versione:** 0.00.000
Descrizione:	*Descrizione generale del caso d'uso (scope).*	
Priorità:	*Priorità attribuita al caso d'uso dagli utenti.*	
Durata:	*Ordine di grandezza stimata della durata del caso d'uso.*	
Attore primario:	*Nome*	
	Interessi nell'esecuzione del caso d'uso	
Attori secondari:	*Nome*	
	Interessi nell'esecuzione del caso d'uso	
Precondizioni:	*Descrizione.*	
Garanzie:	**Minime:** *descrizione.*	
	Successo: *descrizione.*	
Avvio:	*Evento che innesca l'avvio del caso d'uso.*	
Scenario principale.		
1.	*<STEP 1>*	
2.	*<STEP 2>*	
3.	*<STEP 3>*	
4.	*<STEP 4>*	
Primo scenario alternativo.		
1.1.	*Elenco delle azioni da eseguire come alternativa a quanto prescritto nel primo passo.*	
Secondo scenario alternativo.		
3.1.	*Elenco delle azioni da eseguire come alternativa a quanto prescritto nel terzo passo.*	
Primo scenario di errore.		
2.1.	*Elenco delle azioni da eseguire nel caso in cui si verifichi una condizione di errore durante l'esecuzione del secondo passo.*	
Annotazioni.		
4.	*Annotazioni relative al punto 4 dello scenario principale.*	

devastanti variazioni dei casi d'uso: le temutissime *change requests* (cambiamenti delle specifiche). Nella pratica si evidenzia l'esistenza di una certa proporzionalità tra il numero di versioni di un caso d'uso, le riunioni effettuate con i clienti e i personaggi presenti presso la relativa organizzazione abilitati a esternare la propria opinione sul sistema. Al risultato andrebbe poi aggiunto qualche fattore correttivo da imputarsi ai commerciali presenti presso la propria società.

La convenzione generalmente accettata per l'attribuzione del numero di versione a un caso d'uso è quella di fatto utilizzata nella gestione delle diverse versioni del codice sorgente, che prevede un formato del tipo `0.00.000`. La cifra più significativa viene aggior-

nata solo nel caso in cui, tra una versione e quella successiva, si proceda alla completa riscrittura del caso d'uso. Le due cifre centrali si modificano a seguito di variazioni significative del comportamento del caso d'uso, come per esempio introduzione di passi aggiuntivi o eliminazione di altri presenti. Infine le ultime tre cifre si variano nel caso di correzioni (*bug fixing*) e quindi per aggiornamenti di minor rilievo.

Nella sezione dedicata alle informazioni generali è necessario specificare una breve descrizione del caso d'uso, la priorità assegnata dall'utente (il che equivale a dire l'importanza), l'ordine di grandezza della durata dell'esecuzione dello stesso, gli attori divisi tra principali e secondari, le precondizioni, le garanzie e la causa innescante (il *trigger*).

Per ciò che concerne la **descrizione** è necessario riportare una breve illustrazione dei servizi offerti dal caso d'uso oggetto di studio. Probabilmente non è il caso di spendere molto tempo cercando di descrivere dettagliatamente l'intero caso d'uso: poche righe di descrizione riportanti le attività più significative, eventualmente corredate dalle relative condizioni di fallimento, nella pratica si dimostrano essere più che sufficienti. La descrizione iniziale non è la sede più opportuna per dilungarsi: per la specifica di dettaglio è prevista la sezione dedicata agli scenari.

La **priorità** permette di evidenziare l'importanza assegnata alle funzionalità che il sistema dovrà fornire, al fine sia di suddividere gli use case in un'opportuna gerarchia di importanza, sia per poter assegnare in modo mirato i vari casi d'uso alle iterazioni di sviluppo.

La priorità dovrebbe essere attribuita dal business analyst, rispecchiando le direttive del cliente. Si tratta di valori molto importanti, prodotto di input per l'attività di pianificazione delle iterazioni in cui organizzare il processo di sviluppo del software. L'assegnazione dei valori delle priorità deve essere eseguita tenendo ben presenti almeno due tipologie di fattori di rischio: quello relativo ai requisiti utente (pericolo di realizzare un sistema che non sia quello di cui l'utente ha bisogno); quello tecnologico (rischio di non disporre delle tecnologie che permettono di ottenere quanto richiesto dall'utente con tempi e costi accettabili). Pertanto, la priorità finale di un caso d'uso deve essere ottenuta da un'opportuna media tra il fattore di rischio assegnato dagli utenti e quello stabilito dal team di sviluppo (tecnologico).

Qualora si utilizzasse un processo di sviluppo di tipo iterativo e incrementale — sempre consigliato in quanto permette di controllare i fattori di rischio del progetto — è necessario tener presente che ogni iterazione dovrebbe essere guidata da un gruppo ben definito di use case, o addirittura, da un insieme prestabilito di scenario. Per esempio non è infrequente il caso in cui in una determinata iterazione, verosimilmente una di quelle iniziali, si sia interessati a realizzare unicamente il *best scenario* (scenario in cui tutto funziona bene: si trascura momentaneamente la gestione delle anomalie) di un certo numero di casi d'uso. Ciò ha una sua logica: si è interessati a mostrare al cliente, il prima possibile, una versione del sistema, o per ottenere preziosi riscontri o semplicemente per provare che l'architettura progettata è efficace. Si capisce allora che il processo di assegnazione delle priorità degli use case è un'attività molto importante e finisce per influenzare la pianificazione delle iterazioni che, a sua volta, è un'attività molto sensibile in quanto ha a che fare con i rischi e il loro controllo.

Come ormai dovrebbe essere arcinoto, un **attore** è una qualsiasi entità, persona o sottosistema, interessato al comportamento del sistema in generale e di un insieme di use case in particolare.

Tipicamente gli attori, nel contesto di uno stesso caso d'uso, sono suddivisi in primari e secondari. Le peculiarità degli attori primari, molto brevemente, sono di fornire lo stimolo iniziale che avvia l'esecuzione del caso d'uso, e di fruire del relativo servizio. Sono quindi interessati a ricevere i principali messaggi inviati dal sistema, ossia i risultati dell'esecuzione del servizio.

I servizi temporizzati costituiscono tipici casi d'uso non avviati esplicitamente da attori. L'avvio viene innescato automaticamente dal sistema stesso a un prestabilito istante di tempo. In tal caso, per rendere i casi d'uso più chiari e comprensibili, si ricorre all'artificio di considerare il tempo come un attore.

Questa soluzione è stata oggetto di discussione del capitolo precedente.

In presenza di servizi temporizzati, la personificazione del tempo come attore è più che legittima. Importante è non esagerare iniziando a visualizzare come attore anche un intero "servizio schedulatore" che, evidentemente è un processo interno al sistema. Considerando attori funzionalità interne del sistema (come per esempio uno schedulatore appunto), si corre il rischio di non analizzare opportunamente servizi che dovranno essere stimati, progettati, implementati ecc. Sempre nel caso si mostri uno scheduler come attore, non si attribuirebbe sufficiente enfasi alle funzioni di prenotazione di eventi, alle eventuali condizioni di avvio, alla relativa comunicazione, cancellazione e così via, che, viceversa, sarebbe opportuno analizzare con un certo dettaglio.

Dal punto di vista della *progettazione* del sistema (non dell'analisi dei requisiti), ciò che interessa non è molto la ripartizione degli attori in primari o secondari (forse neanche tanto gli attori stessi), né tantomeno i nomi da attribuire ad essi (sebbene ciò possa far risparmiare molto tempo nelle riunioni con i clienti), bensì gli obiettivi che questi intendono raggiungere per mezzo dell'esecuzione del caso d'uso: la funzionalità da dover realizzare. Chiaramente è sempre conveniente scegliere nomi appropriati per le varie entità del sistema; in particolare, nel caso degli attori: i nomi permettono di sintetizzare descrizione del lavoro svolto, background e skill relativi, e così via.

Una volta realizzata la funzionalità specificata da un caso d'uso, non è infrequente la situazione in cui si scopra che magicamente la funzione descritta sia oggetto di interesse per una molteplicità di attori; si consideri per esempio il servizio di *Ricarica account* descritto in fig. 4.18. Ebbene, questa dovrebbe essere eseguita dal commesso per ricaricare l'account del cliente, ma nulla vieta all'amministratore di espletare lo stesso servizio.

Infine, la quasi totalità dei sistemi reali prevede un qualche meccanismo di sicurezza. Uno dei criteri principali utilizzati da questi meccanismi per il processo di autorizzazione consiste nel suddividere la popolazione utenti in gruppi al fine di poter gestire, per ciascuno di essi, l'elenco dei servizi eseguibili. Pertanto, nel sistema finale, l'associazione attori/servizi avviene configurando opportunamente il sistema di sicurezza.

Le **precondizioni** sono i requisiti che devono essere soddisfatti per poter eseguire il caso d'uso al quale si riferiscono: è compito del sistema verificarne l'adempimento prima di avviare l'esecuzione dell'efferente caso d'uso, mentre è responsabilità dell'utilizzatore di assicurarne il soddisfacimento.

L'esempio di precondizione più classico, quasi onnipresente nelle use case view, è l'ottenimento da parte dell'utente dell'abilitazione del sistema per l'esecuzione di funzioni considerate sensibili. La celebre frase cita letteralmente "precondizione: l'utente abbia eseguito il login e sia stato riconosciuto dal sistema".

Il riscontro pratico delle precondizioni è che l'implementazione dell'efferente caso d'uso preveda una serie di controlli iniziali atti a verificare, appunto il relativo soddisfacimento.

Si consideri ancora un qualsivoglia sito di e-commerce che offra ai propri utenti la possibilità di acquistare prodotti e/o servizi stando seduti comodamente nella ultracitata poltrona di casa. Si prenda in esame il caso d'uso in grado di effettuare un ordine che contempli prodotti precedentemente inseriti nel carrello della spesa. Le precondizioni per un caso d'uso del genere potrebbero essere che:

1. "l'utente abbia eseguito con successo il login e sia quindi stato riconosciuto dal sistema";

2. "il carrello della spesa non sia vuoto".

Un errore che spesso capita di commettere è confondere condizioni che solo in particolari casi devono essere esaudite, con le precondizioni che invece devono essere sempre soddisfatte.

Per ciò che concerne le **garanzie** è necessario citare esplicitamente sia quelle minime, sia quelle di successo. Le prime, come suggerisce il nome, sono le più basilari assicurate dall'esecuzione del caso d'uso. Tipicamente si tratta delle garanzie assicurate dal sistema nel caso in cui non sia possibile fornire il servizio specificato (scopo del caso d'uso non raggiunto) in quanto l'esecuzione è viziata da condizioni di errore, o comunque da anomalie rispetto al flusso principale del caso d'uso (*best scenario*). In condizioni di fallimento nel conseguimento degli obiettivi del caso d'uso, le condizioni minime dovrebbero almeno prevedere il mantenimento di uno stato consistente: tipicamente quello immediatamente antecedente all'avvio dello use case.

Non è infrequente il caso in cui si compilino lunghe liste di condizioni minime. Un errore comune è quello di specificare le condizioni minime per ogni possibile fallimento dell'esecuzione del caso d'uso. Probabilmente non è il caso di dilungarsi e di creare problemi di sincronizzazione tra la lista delle garanzie e le possibili anomalie che possono insorgere nell'esecuzione di un caso d'uso: tutte le condizioni di eccezione hanno un insieme minimo comune di garanzie, pertanto è sufficiente e opportuno riportare unicamente tale insieme minimo.

Spesso, per semplificare il compito, invece di riportare le garanzie minime, si sostituisce la dicitura con "garanzie in caso di fallimento": entrambe sono legittime e la scelta deve ricadere sull'alternativa che fornisce un migliore grado di comprensione.

Con riferimento al caso del sito di commercio elettronico, una garanzia minima dell'esecuzione della funzione di compilazione degli ordini potrebbe essere: "l'importo dell'ordine viene addebitato solo a seguito di totale convalida dello stesso", mentre la relativa versione di garanzia in caso di fallimento potrebbe essere "l'ordine non viene preso in carico e nessun importo viene addebitato al cliente".

Specularmente a quelle minime, le garanzie di successo specificano ulteriori condizioni soddisfatte dal completamento dell'esecuzione del relativo caso d'uso, ossia dopo l'esecuzione dello scenario principale o al termine dell'esecuzione di una sua variante comunque di successo (flussi alternativi). Mentre nel caso delle precondizioni è compito dell'utilizzatore del caso d'uso assicurarle, nel caso delle garanzie è compito del sistema, qualora le prime siano soddisfatte, assicurarne il conseguimento.

Come si può notare, ciò permette di evidenziare e dividere nettamente le responsabilità tra le varie entità che partecipano a un caso d'uso (purtroppo non è sempre possibile richiedere lo stesso a talune organizzazioni…).

Le garanzie di successo implicano il soddisfacimento di quelle minime con l'introduzione di ulteriori condizioni rese possibili dal raggiungimento di almeno uno degli obiettivi del caso d'uso.

Nel caso della compilazione di un ordine di acquisto in un sito di commercio elettronico, le garanzie di successo potrebbero prevedere l'accettazione e la presa in carico, da parte dell'organizzazione, dell'ordine effettuato con addebito del relativo importo all'utente.

Per ciò che concerne il campo *Avvio* è necessario descrivere l'evento che determina l'inizio dell'esecuzione del sistema.

Per esempio considerando il caso d'uso di interazione tra l'utente e un sistema ATM (il nostrano Bancomat), lo use case viene avviato a seguito dell'introduzione da parte dell'utente della relativa carta di credito.

Prendendo in considerazione funzioni eseguite periodicamente dal sistema, l'avvio è generato dallo scadere di un determinato intervallo di tempo.

Si consideri per esempio il sito per il commercio elettronico ed in particolare il caso d'uso che permette di inviare gli ordini effettuati dal cliente al legacy system dell'organizzazione. Nel caso in esame l'avvio del caso d'uso avviene o a intervalli di tempo prestabiliti: ogni *n* ore, oppure a orari ben stabiliti: alle ore 13 e alle 24, e così via.

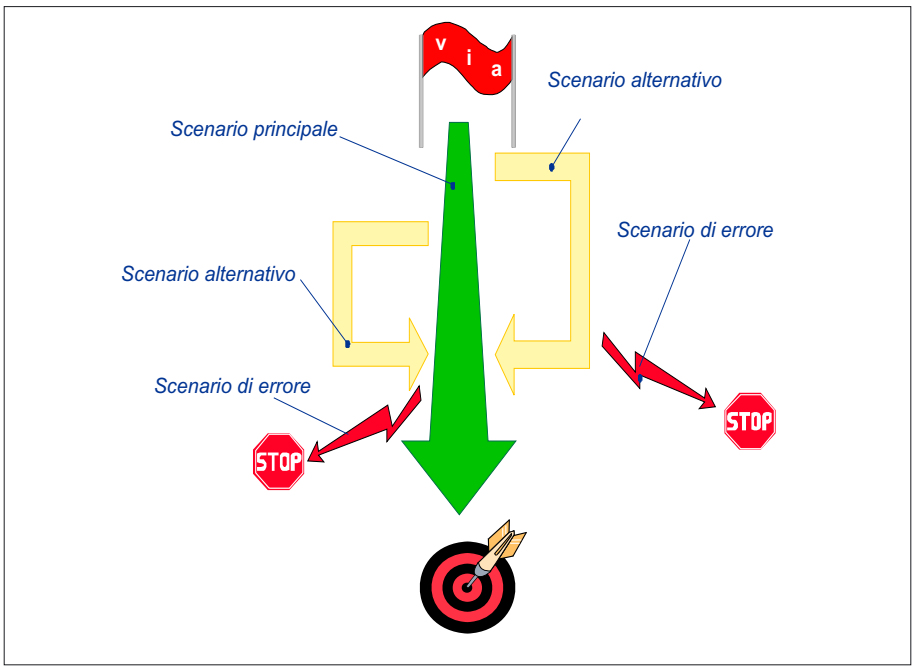
La sezione successiva del modello è dedicato agli scenari. Tipicamente si distinguono tre tipologie (fig. 4.1):

1. **scenario principale di successo** (*main success scenario*): lo scenario che produce il servizio richiesto dall'attore primario nel caso in cui tutto funzioni correttamente: dati di ingresso validi, nessuna eccezione scaturisce durante lo svolgimento, ecc.;

2. **scenari alternativi** (*alternative scenario*): si tratta di flussi di azioni che rappresentano diramazioni dello scenario principale la cui funzionalità però non sia la gestione di condizioni di errore. Si è ancora in un flusso di successo, ma l'esecuzione del servizio richiede un percorso diverso. In sostanza è una forma più elegante per evitare diramazioni nel flusso principale (il famoso costrutto `if then else`) che comunque restano legittime;

3. **scenari di fallimento o di errore** (*failure scenario*): gli scenari che specificano le azioni da intraprendere nel caso in cui nell'esecuzione delle azioni dello scenario principale o alternativi sia impossibilitata dall'insorgere di condizioni di errore. Si tratta di illustrare il comportamento da eseguire in modo del tutto simile a quello utilizzato dai linguaggi di programmazione per specificare la gestione delle eccezioni. Da tener presente che ogni qualvolta in un passo dei flussi precedenti si men-

Figura 4.1 — *Schematizzazione degli scenario di un caso d'uso.*



ziona un verbo del tipo "verifica", "controlla", "valida", ecc. verosimilmente esiste un flusso alternativo o di errore associato con la verifica negativa del test.

Talune volte capita di non riuscire a distinguere chiaramente flussi alternativi da quelli di errore. Per esempio intervengono condizioni anomale che però possono essere gestite. Una regola semplice semplice consiste nell'interrogarsi se il flusso oggetto del conteso generi o meno il fallimento dello use case. In caso di fallimento si tratta evidentemente di un flusso di errore, altrimenti di uno alternativo.

La descrizione dei flussi alternativi o di errore è molto importante e permette di evidenziare delle dinamiche che invece tenderebbero a essere rilevate solamente durante la fase di test. Qualora alcuni comportamenti non siano ben specificati o mancanti, è tendenza naturale di molti programmatori assumere comportamenti più consoni alle proprie esigenze di implementazione, di tempo, ecc.

La struttura utilizzata nel template oggetto di studio prevede di specificare inizialmente la sequenza di azioni da compiere nel caso in cui tutto funzioni correttamente, dall'avvio del caso d'uso al relativo compimento, per poi fornire le condizioni anomale che possono intervenire e le azioni da intraprendere. Vi è una stretta similitudine con la pratica quotidiana. Si consideri la situazione in cui è necessario illustrare a un'altra persona un qualche sistema o una disposizione: si inizia fornendo le istruzioni sulla sequenza "corretta" e poi si spiega come gestire eventuali casi eccezionali che potrebbero intervenire: "è necessario fare questo, quello e quell'altro ma, nel caso in cui si verifichi quest'altro ancora, allora è necessario comportarsi in tale maniera ecc.".

Spesso, piuttosto che "nominare" i flussi alternativi e di errore con numeri ordinali indicanti la sequenza temporale di apparizione, si preferisce attribuire una descrizione che precisi la condizione che ne genera l'esecuzione. Chiaramente ciò non avrebbe alcun senso per ciò che concerne il flusso principale (il nome sarebbe sempre lo stesso!). Il vantaggio offerto da tale tecnica è legato alla maggiore comprensibilità della descrizione del caso d'uso, alla minimizzazione del lavoro richiesto da eventuali aggiornamenti (per esempio, dovendo inserire un nuovo flusso o eliminarne un altro non sarebbe necessario numerare nuovamente i successivi), e all'utilizzo più proficuo di strumenti da utilizzarsi per la catalogazione e gestione dei requisiti (l'elenco dei flussi mostrerebbe una descrizione autoesplicativa piuttosto che un anonimo numero). Tale approccio è stato utilizzato nell'esempio relativo al sito per il commercio elettronico.

Nella stesura degli scenari probabilmente può risultare utile strutturarli come una successione di azioni o "transazioni" limitate.

Molto importante è evidenziare chiaramente quale entità (attore, sistema) svolge ciascuna azione. Molto spesso si utilizzano particolari versioni di template nei quali la sezione dedicata agli scenari è suddivisa orizzontalmente in un numero di colonne equivalenti alle entità che prendono parte al caso d'uso. Ciò permette di riportare in ciascuna colonna unicamente le azioni eseguite dall'entità di appartenenza.

Nella definizione di un caso d'uso, oltre a quello principale è necessario riportare tutta la sequenza di casi di errore o comportamento anomalo che potrebbero verificarsi con le relative azioni da compiere. Esistono diverse modalità per specificare gli scenari di fallimento. La più classica è quella che utilizza uno stile del tutto simile al codice scritto con linguaggi di programmazione che non supportano la gestione delle eccezioni (*C like*): dopo ogni azione che può causare un errore viene eseguito un test esplicito e quindi vengono dettagliate le azioni da compiere nel relativo blocco. Pertanto il tutto viene specificato nella stessa sequenza. Una tecnica alternativa prevede invece di utilizzare uno stile simile alla stesura del codice utilizzando il meccanismo delle eccezioni: nelle apposite sezioni vengono riportate "le eccezioni" che si intende gestire e le relative operazioni da compiere. Pertanto per ogni azione che può generare anomalie sono presenti tante sezioni "alternative", una per ciascuna tipologia di anomalia, riportanti sia la dichiarazione dell'anomalia stessa sia le azioni da compiere. Questa organizzazione è particolarmente utile

anche per l'attività di pianificazione del processo di sviluppo del software: le varie versioni (*build*) possono essere "schedulate" considerando solo specifici flussi di determinati casi d'uso, rimandando a versioni future quelli tralasciati.

L'illustrazione degli scenari alternativi effettuata attraverso opportuni template evidenzia il grande vantaggio offerto da questo formalismo rispetto a quello grafico. In quest'ultimo è tipicamente necessario disegnare un nuovo diagramma per ogni alternativa, con maggiore dispendio di tempo per la realizzazione e notevoli difficoltà nella gestione delle relative modifiche. L'ultima sezione, del tutto opzionale, prevede di riportare eventuali annotazioni, che possono essere riferite sia all'intero use case, sia a singole azioni.

Qualora si decida di ricorrere ai template, c'è da tener presente che molte persone ritengono tali modelli complicati, "fuorvianti" perché distolgono l'attenzione dai contenuti e così via. Probabilmente le stesse persone, dopo aver utilizzato un programma di video scrittura, lascerebbero tracce di bianchetto sullo schermo.

Il vantaggio dei template è quello di essere più concisi, conferire una migliore organizzazione alle informazioni, renderne più facile il reperimento, infondere maggiore coerenza alla descrizione dei casi d'uso ecc. Qualora però le idiosincrasie dovessero rimanere, si consiglia di stampare le varie descrizioni senza i bordi tipici delle tabelle. Invece, per i nostalgici delle schede perforate, i bordi vanno benissimo… Anzi, magari si potesse fare a meno anche dei grafici…

Flussi alternativi e sottoflussi (*subflow*)

Uno dei problemi storicamente più critici insito nei processi di sviluppo di sistemi informatici consiste nella adeguata comprensione delle esatte necessità dell'utente.

L'avvento del formalismo dei casi d'uso, la cui escogitazione si deve prevalentemente al lavoro portato avanti da Ivar Jacobson (lo svedese dei Tres Amigos) tra la fine degli anni Ottanta e gli inizi degli anni Novanta, sembrerebbe aver contribuito in modo decisivo a risolvere il problema, sebbene il formalismo da solo non basti.

L'esperienza dell'autore mostra che, benché la notazione dei casi d'uso sia piuttosto diretta e relativamente semplice, molte organizzazioni considerano il ricorso ai casi d'uso molto complicato, oneroso e difficile da capire e gestire. Dove risiederà mai il problema? La vocina interna suggerirebbe di cercare in diverse direzioni. La scarsa voglia di un aggiornamento non superficiale, ma anche il mettere tutto nero su bianco e la relativa necessità di chiarezza implica responsabilità che ben pochi si vogliono assumere; e cosa dire di coloro che fanno roccaforte dei quattro concetti da loro conosciuti?

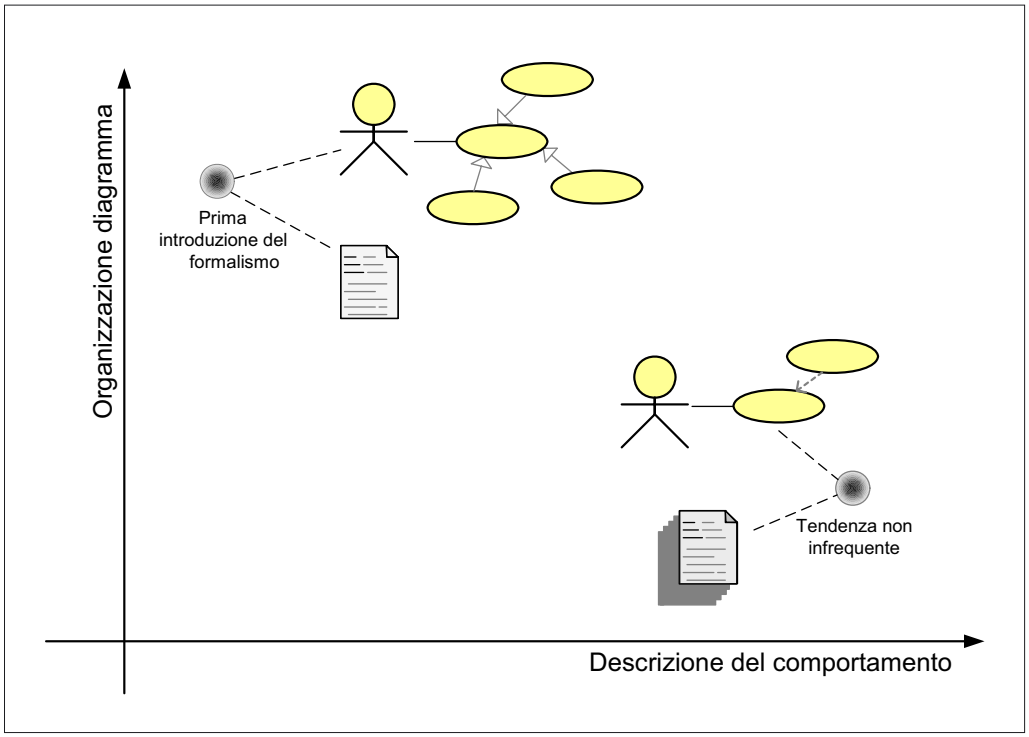
In ogni modo, la definizione iniziale del formalismo prevedeva quasi esclusivamente la notazione grafica, che, tra l'altro, contemplava un insieme piuttosto limitato di elementi per strutturare i casi d'uso. Ben presto però si capì che queste ellissi, troppo "vuote", erano eccessivamente "leggere" per specificare le funzionalità che il sistema doveva prevedere: il solo nome non aiutava granché.

Nella definizione iniziale del formalismo dei casi d'uso, dunque, il ruolo del comportamento dinamico era ridotto quasi al minimo, affidato a un testo generico.

Attualmente, grazie anche ai template provvisti da talune aziende, si assiste al problema diametralmente opposto (fig. 4.2). In molte organizzazioni si tende a ridurre eccessivamente la strutturazione del diagramma dei casi d'uso a favore della descrizione del relativo comportamento dinamico. Vi è un attore che interagisce con un caso d'uso in maniera piuttosto lineare, salvo però poi accedere alla descrizione dello stesso e trovarsi di fronte a decine di pagine di specifiche, in cui compaiono flussi alternativi, sottoflussi, e così via. In tali situazioni è legittimo interrogarsi sulla necessità di disegnare il diagramma che perde completamente, o quasi, il suo valore.

Da tener presente che la definizione di flussi alternativi, sottoflussi, ecc., all'interno della descrizione di un singolo caso d'uso non è perfettamente in linea con le direttive standard dello UML.

Figura 4.2 — *Il diagramma mostra una verosimile evoluzione dell'utilizzo tipico del formalismo dei casi d'uso, nell'arco del decennio (inizi degli anni Novanta a oggi), nelle organizzazioni informatiche.*



I flussi alternativi, se non utilizzati per mostrare in maniera più elegante un costrutto `if ... then ... else ...`, e quindi se eccedenti le 5-6 righe, dovrebbero essere evidenziati per mezzo di un ulteriore caso d'uso associato a quello di partenza tramite una relazione di *extend*.

Per ciò che riguarda i sottoflussi, essi rappresentano veri e propri use case da associarsi a quello oggetto di studio per mezzo della relazione di *include*.

A questo punto, le domande che potrebbero sorgere sono essenzialmente due:

1. perché potrebbe risultare importante non inglobare più casi d'uso in uno solo?

2. perché invece spesso si preferisce procedere con tale approccio?

Per ciò che concerne il secondo interrogativo, la risposta potrebbe essere legata alla necessità di contenere il numero dei casi d'uso e la complessità dei relativi diagrammi.

Uno dei principali motivi per cui è importante modellare risiede nella necessità di sostenere la comunicazione. Si tratta di un problema molto rilevante specie nelle fasi iniziali del ciclo di vita del software, ove gli interlocutori sono gli utenti, i quali, tipicamente, dispongono di un background profondamente diverso da quello dei tecnici.

Contenere la complessità dei diagrammi, pertanto, aiuta il processo di comunicazione. Da tenere presente che, sebbene il formalismo dei casi d'uso sia piuttosto accattivante, è comunque avvertito come estraneo dagli utenti.

Inoltre, disporre di una fitta rete di use case potrebbe generare non pochi problemi al processo mentale di ricostruzione della dinamica del servizio descritto: bisognerebbe interrompere più volte la lettura della descrizione di un caso d'uso per "saltare" a quella di un altro e così via.

Infine, molto spesso gli utenti, che per definizione devono riesaminare i modelli dei casi d'uso, sono intimoriti da elevate quantità degli stessi.

Tale abnorme proliferazione però, spesso è dovuta ad alcuni "modellatori" di casi d'uso i quali, dimenticandosi che questo formalismo non appartiene alla fase di analisi e disegno del sistema, danno luogo a vere e proprie attività di progettazione della struttura interna del sistema attraverso i casi d'uso.

Comprese le ragioni per le quali alcune volte si tenta di inglobare la descrizione dei casi d'uso in uno solo, è importante capire perché ciò non sia auspicabile, rispondendo quindi alla prima domanda.

In primo luogo si rischia di attenuare l'importanza della dichiarazione di pre- e post-*conditions*. Dovendo soddisfare un gruppo di casi d'uso, queste dovrebbero necessaria-

mente subire un processo di estrapolazione del "massimo comune divisore": ossia si correrebbe il rischio di evidenziare solo pre- e postcondizioni condivise da tutti i casi d'uso, rinunciando al necessario dettaglio.

Qualora invece si decidesse di specificarle tutte le pre- e postcondizioni attraverso apposita lista, si correrebbe il rischio di generare confusione circa l'associazione di ciascuna di esse al relativo use case di appartenenza. Ciò rappresenterebbe un problema specie per le precondizioni che, in ultima analisi, costituiscono dei controlli da dover effettuare all'avvio del relativo servizio.

Se invece si decidesse di ripetere la sezione di pre- e post-conditions per la descrizione di ciascun caso d'uso, verrebbe meno la necessità di raggrupparli in un unico documento/modello.

Inglobare descrizioni di casi d'uso, inoltre, non ne semplifica il riutilizzo e tantomeno permette di evidenziare importanti sezioni di comportamento condiviso.

Doug Rosenberg e Kendall Scott — autori dei testi *Applied Use Case-Driven Object Modeling* (Addison-Wesley, 2001) e *Use Case-Driven Object Modeling with UML* (Addison-Wesley, 2001) — asseriscono che "bisogna preoccuparsi qualora le descrizioni dei casi d'uso siano lunghe quattro pagine. Lo scenario principale dovrebbe essere lungo due-tre paragrafi al massimo. Ogni flusso alternativo dovrebbe essere di una-due frasi. Qualche volta può capitare di avere casi d'uso molto brevi, specie quando sono utilizzati come tessuto di collegamento, altre volte potrebbe esserci bisogno di use case di maggiori dimensioni. Ma è necessario utilizzare tecniche […] atte a raggruppare a fattore comune il comportamento condiviso per scrivere use case più concisi che meglio si prestano al riutilizzo".

Ancora, la priorità assegnata potrebbe non essere chiaramente scindibile tra i vari casi d'uso specificati all'interno di una stessa descrizione del comportamento dinamico, con conseguenze sul processo di decisione dei casi d'uso/scenari da incorporare in ciascuna iterazione.

Per finire, inglobare i casi d'uso può complicare sia le attività di realizzazione dei test case, sia la pianificazione del processo (stima della complessità, delle risorse da allocare, ecc.) e così via.

A questo punto cosa fare? Probabilmente la soluzione migliore consiste nel tentare di trovare "il giusto mezzo", sebbene l'autore sia un sostenitore della antiglobalizzazione dei casi d'uso, come si avrà ben modo di costatare successivamente.

Sezioni aggiuntive

Analizzando modelli presenti in varie organizzazioni informatiche, relativi alla descrizione del comportamento dinamico dei casi d'uso, è possibile imbattersi in versioni con un numero di sezioni supplementari non sempre indispensabili. In questi casi, l'inutile e noioso lavoro aggiuntivo rende possibile intuire le ragioni dei tecnici che tendono a rinnegare i processi accademici di sviluppo del software.

Una delle sezioni a cui è possibile rinunciare senza troppo sacrificio è quella dedicata alla descrizione delle relazioni che coinvolgono il caso d'uso oggetto di specifica. Non solo si tratta di un gruppo di informazioni poco proficue, quindi meri dati, (è sufficiente ispezionare visivamente il diagramma dei casi d'uso per evincerle), ma esse tendono anche a generare lavoro extra gratuito: ogni qualvolta si aggiornano delle relazioni nel diagramma (per esempio si decide che sia più conveniente visualizzare una determinata relazione per mezzo di un *extend* anziché di un *include*) è necessario rivedere le varie descrizioni del comportamento dinamico dei casi d'uso.

Un gruppo di informazioni che spesso può valere il caso riportare è quello relativo ai cosiddetti **requisiti speciali** (*special requirements*). Si tratta di descrizioni testuali che raggruppano l'insieme di requisiti non funzionali che influenzano l'implementazione dello use case oggetto di specifica. Da tener presente che per questi dati tipicamente è previsto un opportuno manufatto, documento e/o foglio elettronico, denominato NFR (*Non Functional Requirements*, requisiti non funzionali).

Riportare gli NFR nel dettaglio degli use case soggetti a tali vincoli può risultare molto comodo, ma solo se il *repository* rimane il relativo documento e negli use case c'è un link alla sezione desiderata. Ciò al fine di minimizzare le ripercussioni dovute a variazioni dei requisiti non funzionali (individuare tutti gli use case in cui sono riportati tali requisiti, aggiornarli, ecc.).

Esempi tipici possono essere relativi all'area:

- sicurezza:
  - la parola chiave deve avere una dimensione superiore ai 6 caratteri e contenere caratteri "speciali";
  - la password non deve appartenere all'insieme delle ultime 4 utilizzate;
  - l'algoritmo di criptazione prevede chiavi asimmetriche di lunghezza 128 bit;

- performance:
  - l'utente non deve attendere più di *x* secondi per l'espletamento del servizio;
  - in condizioni di regime il sistema deve essere in grado di supportare la connessione di *y* utenti contemporanei;

Un semplice processo per produrre la descrizione dei casi d'uso

Di seguito si presenta un processo essenziale, frutto dell'esperienza personale dell'autore, in grado di agevolare la produzione di appropriate descrizioni del comportamento dei casi d'uso.

In effetti, è un'attività spesso non banale e, verosimilmente, non è del tutto corretto asserire che si tratti di una mera attività di descrizione (altrimenti sarebbero necessari "tecnici" dotati unicamente di skill linguistico, magari scrittori), bensì è necessario eseguire un vero e proprio processo di investigazione, scoperta e analisi dei requisiti utenti.

Il processo, sinteticamente, si articola nei seguenti passi:

1. definizione del goal;
2. definizione delle precondizioni;
3. produzione del dettaglio dello scenario principale;
4. specificazione degli scenari alternativi;
5. inclusione delle rimanenti informazioni.

In primo luogo è consigliabile definire chiaramente quale sia lo scopo dei casi d'uso che si vuole descrivere, ossia le post-conditions (alcuni esempi sono: autorizzare l'utente ad accedere al sistema, validare il carrello della spesa, eseguire un ordine, ecc.). Qualora anche questa attività dovesse presentare incertezze, sarebbe il primo campanello d'allarme che qualcosa sia stato definito non accuratamente.

Dichiarati gli obiettivi dei caso d'uso è importante iniziare a definire le precondizioni: è importante in quanto si tratta di requisiti che devono essere soddisfatti da tutti i flussi. Qualora nel procedimento di individuazione e descrizione di flussi alternativi ci si dovesse accorgere che ciò non è vero, ci si troverebbe di fronte a due eventualità:

1. le precondizioni non sono state definite con molta precisione, e quindi è sufficiente correggerle;
2. i flussi rappresentano effettivamente casi d'uso diversi e quindi è necessario procedere con un'attività di ristrutturazione dell'organizzazione degli use case.

Il passo successivo consiste nel dettagliare lo scenario principale. In sostanza si definisce come giungere all'obiettivo sancito dal caso d'uso nell'ipotesi in cui tutto funzioni meravigliosamente bene. Per questo motivo, spesso, ci si riferisce allo scenario principale

con la definizione di *happy days scenario* (scenario dei giorni felici). Per esempio nello use case di *login*, nello scenario principale si potrebbe assumere che il profilo dell'utente esista già, la password inserita concordi con quella dell'utente, e così via.

Qualora la descrizione dello use case sia troppo prolissa o, viceversa, eccessivamente breve, potrebbe essere opportuno verificare se sia possibile scomporre ulteriormente il caso d'uso (prima evenienza) estraendo comportamento comune o ripetitivo o, (seconda evenienza) se sia possibile inglobare lo stesso in altri casi d'uso, tenendo conto anche di quanto sancito precedentemente.

In molti processi di sviluppo del software si preferisce utilizzare un approccio iterativo e incrementale anche nella costruzione del modello dei casi d'uso. Invece di definire completamente ogni singolo caso d'uso prima di passare al successivo, si preferisce realizzare una certa percentuale di ciascuno di essi (magari solo lo scenario principale) al fine di produrre rapidamente una prima versione da rivedere con gli utenti. Approcci di questo tipo sono particolarmente utili qualora il sistema da realizzare preveda un numero non trascurabile di casi d'uso.

Definito lo scenario principale, altra attività importante consiste nell'individuare i flussi alternativi o di errore. Si tenga in mente che la presenza di questi flussi è del tutto normale, mentre la loro assenza potrebbe essere imputabile alla mancata individuazione di percorsi possibili.

Qualora esistano diversi flussi alternativi (flussi comunque in grado di conseguire l'obiettivo dichiarato) dovrebbe risultare abbastanza intuitivo utilizzare quello più frequente come flusso principale… Il condizionale è d'obbligo perché capita comunque di imbattersi in situazioni opposte.

Ogniqualvolta nel flusso principale siano presenti frasi del tipo "il sistema verifica", "controlla", "valida", "si accerta", ecc. la presenza di flussi alternativi diventa obbligatoria. Spesso si commette l'errore di non evidenziare questi verbi. Per esempio invece di affermare "il sistema tenta di reperire il profilo utente", si asserisce "il sistema reperisce il profilo utente". Frasi sì congegnate tendono a rendere più difficile l'individuazione di flussi alternativi e/o di errore. In generale, un buon criterio per individuare i flussi alternativi consiste nello scorrere il flusso principale, punto per punto, e nell'interrogarsi se per ciascuno di essi sia possibile che qualcosa non funzioni come ci si aspetti, che il sistema si comporti diversamente, che l'attore agisca in modo diverso da quanto sancito e così via.

Definiti anche i flussi alternativi e di errore, le rimanenti attività sono completamente volte all'individuazione delle restanti informazioni che, a questo punto, non dovrebbero più creare eccessivi problemi.

Esempio: Internet University Booking System

In questa sezione viene fornito un esempio relativo a una parte di un sistema universitario atto a fornire a studenti e docenti universitari una serie di servizi fruibili attraverso un comune browser Internet.

Obiettivo del "progetto" è realizzare sia un sito Internet/Intranet dinamico e interattivo, sia uno strato di comunicazione (*wrapper*) tra il sito stesso e il sistema "gestionale" dell'università (un'istanza di legacy system).

In questo paragrafo, l'attenzione è focalizzata sul sito Internet e in particolare sulla funzione di prenotazione sessioni di esami. Gli utenti (studenti universitari) collegati al sito universitario, dopo essere stati opportunamente riconosciuti dal sistema (attraverso la classica coppia login + password), possono consultare le varie sessioni di appello previste per gli esami appartenenti alla propria facoltà, ed eventualmente possono prenotarsi.

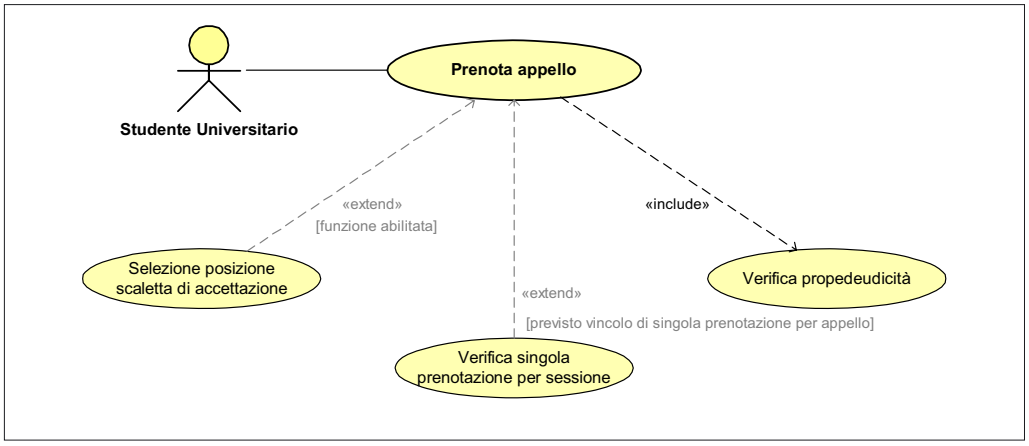
Si inizi con il considerare lo use case diagram riportato nella fig. 4.3. che illustra graficamente la proiezione statica del caso d'uso per la prenotazione Internet degli esami universitari. Dalla sua analisi è possibile evincere l'unico attore, chiaramente principale (lo `Studente Universitario`), le principali funzioni e la relativa organizzazione strutturale.

Si ricordi che probabilmente non è il caso di perdere troppo tempo nel realizzare diagrammi dei casi d'uso eccessivamente dettagliati. L'eccessivo dettaglio, oltre a incidere negativamente sul tempo necessario per produrre la versione iniziale del diagramma, ne complica inutilmente la manutenzione e genera tutta una serie di effetti indesiderati, come il tentativo di definire l'architettura del sistema con strumenti sbagliati, vincolare improduttivamente le restanti fasi del processo e così via.

I diagrammi dei casi d'uso sono assolutamente soggettivi: la struttura evidenzia le "sottofunzioni" ritenute più significative dal progettista in funzione di quelle che sono le esigenze del cliente.

Può capitare per esempio che un cliente pretenda di vedere visualizzato uno specifico use case, anche se dal punto di vista tecnico risulta assolutamente insignificante e non coerente… Come al solito bisogna mettere il padrone dove vuole l'asino (spesso è molto più razionale del contrario).

Figura 4.3 — *Use case prenotazione Internet sessioni d'esame.*



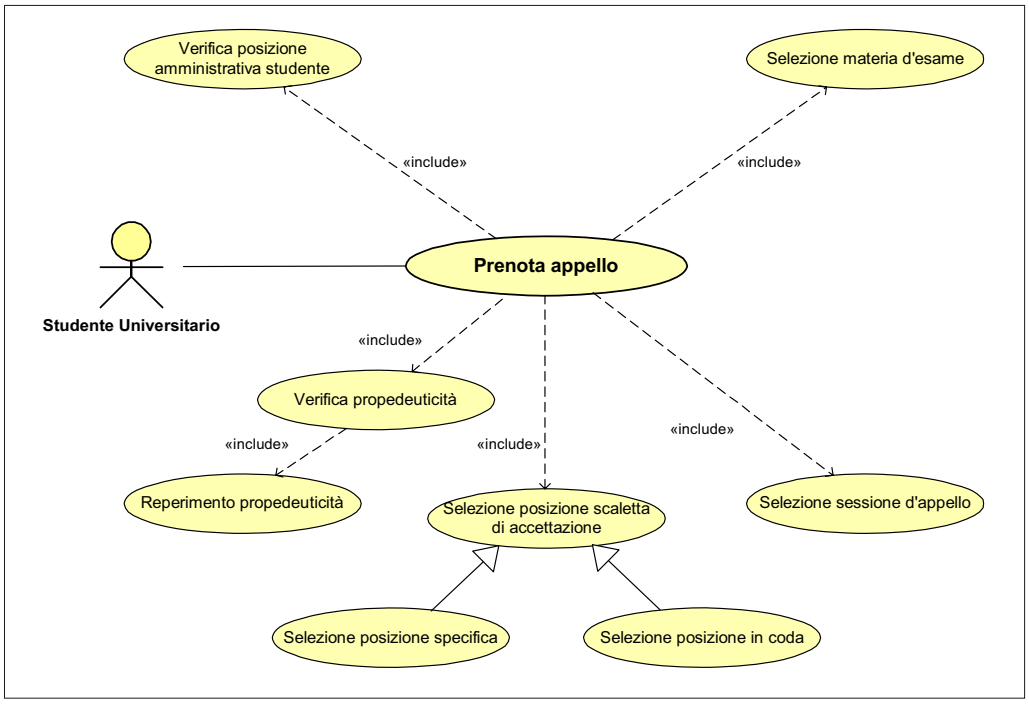
Molto probabilmente, se si fornissero esattamente le stesse specifiche a dieci tecnici diversi, si otterrebbero altrettante diverse legittime versioni. Tuttavia, verosimilmente, sarebbe possibile evidenziarne alcune, per così dire, più corrette o almeno più convenienti di altre. Per esempio nello use case `Prenota Appello` visualizzato nella fig. 4.3 si è deciso di evidenziare le funzioni di `Verifica propedeuticità`, `Selezione posizione scaletta di prenotazione` e `Verifica singola prenotazione per sessione`. In altre parole si è deciso di evidenziare che lo studente, per potersi prenotare a una determinata sessione di appello, deve aver superato gli eventuali esami propedeutici a quello a cui si vuole iscrivere e, nei limiti legati alla disponibilità, può selezionare la posizione desiderata nella scaletta di accettazione. È altresì possibile evincere che per talune sessioni di esame, a discrezione dell'insegnante, è possibile iscriversi a uno solo degli appelli previsti nella relativa sessione (per esempio quando ne sono previsti diversi a scadenze piuttosto ravvicinate).

> Un servizio così concepito correrebbe il rischio di essere eccessivamente rigido e particolarmente odiato dagli studenti. Una soluzione conveniente potrebbe consistere nel dare la possibilità agli studenti di effettuare prenotazioni con riserva. Questa tipologia verrebbe selezionata automaticamente dal sistema nel caso in cui la posizione amministrativa dello studente e/o la verifica delle propedeuticità non risultassero completamente in regola.

Nel diagramma in fig. 4.4 è illustrata la stessa funzionalità ma attraverso un diagramma dei casi d'uso decisamente più dettagliato. Sebbene esso sia a tutti gli effetti corretto, è comprensibile che il relativo livello di dettaglio sia eccessivo.

Per il proseguimento della trattazione si faccia riferimento al diagramma di fig. 4.3.

Figura 4.4 — *Use case eccessivamente dettagliato per la prenotazione automatica sessioni d'esame.*



A questo punto, definita la proiezione statica dello use case *prenotazione Internet sessioni di esami,* è necessario passare allo studio del comportamento dinamico.

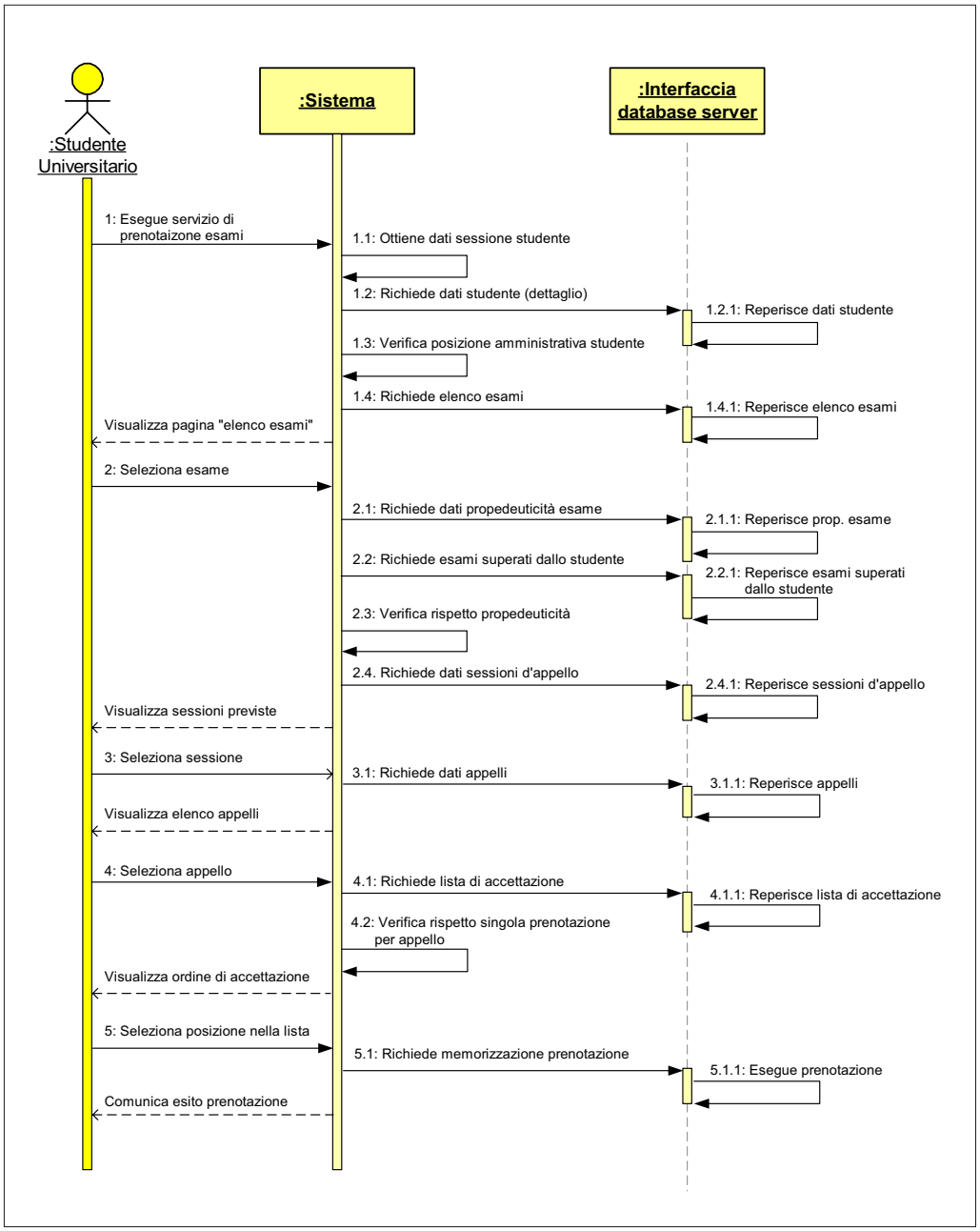
Una prima alternativa potrebbe essere quella di utilizzare i diagrammi dello UML preposti per lo studio/documentazione degli aspetti dinamici del sistema. Ovviamente il livello di astrazione dovrebbe essere elevato al fine di risultare compatibile con i principali fruitori: gli utenti.

Anche se i diagrammi mostrati di seguito non sono stati ancora oggetto di una presentazione organica (illustrati nel corso del Capitolo 9), il relativo utilizzo, almeno in questa fase, è così intuitivo da non richiedere spiegazioni di dettaglio.

In particolare in fig. 4.5 è presentato un diagramma di sequenza (sequence diagram) che illustra un ipotetico scenario di successo per la funzione di prenotazione esami.

> Molto brevemente, nella prima riga vengono riportati gli "oggetti" che partecipano alla realizzazione della funzione, l'asse verticale illustra il trascorrere del tempo e le varie frecce illustrano lo scambio di messaggi tra i vari oggetti.

Figura 4.5 — *Esempio di sequence diagram relativo unicamente allo scenario di successo della funzione di prenotazione esame.*



Il vantaggio offerto dai sequence, e dai diagrammi in generale, è l'essere particolarmente accattivanti grazie al formalismo grafico che ne facilita notevolmente comprensione e memorizzazione. Però, ahimè, a fronte di questo vantaggio esiste una serie di svantaggi che probabilmente ne sconsigliano l'utilizzo a favore di strumenti più pragmatici.

In primo luogo è necessario più tempo per produrli e il loro processo di manutenzione è piuttosto oneroso. Il diagramma presentato in figura illustra unicamente lo scenario di successo; oltre ad esso sarebbe necessario produrne diversi per illustrarne gli scenari alternativi. Volendo, sarebbe anche possibile specificare i vari flussi alternativi nello stesso diagramma: sebbene ciò da un lato limiterebbe il numero di diversi diagrammi da dover realizzare, dall'altro renderebbe il diagramma decisamente più complicato e quindi più difficilmente interpretabile. Questo approccio è consigliabile qualora sia presente un numero limitato di flussi alternativi; eventualità che però non corrisponde al caso in questione. Tipicamente si preferisce realizzare lo scenario di successo e tanti altri diagrammi quanti sono gli scenari alternativi. In questo caso si capisce bene che una modifica allo scenario principale potrebbe ripercuotersi in tutti i rimanenti diagrammi.

Un altro inconveniente è legato al fatto che tali diagrammi mostrano il comportamento dinamico di oggetti del sistema, quindi, anche se molto ma molto superficialmente, è necessario dare una prima "sbirciata" all'interno dello stesso.

Probabilmente l'inconveniente principale rimane l'impossibilità di specificare ulteriori informazioni molto importanti quali precondizioni, garanzie, durata, ecc. Ovviamente nulla vieta di inserirle come note aggiuntive ma il risultato sarebbe comunque meno formale e meno "standardizzato" e quindi completamente demandato alla buona volontà dei singoli tecnici.

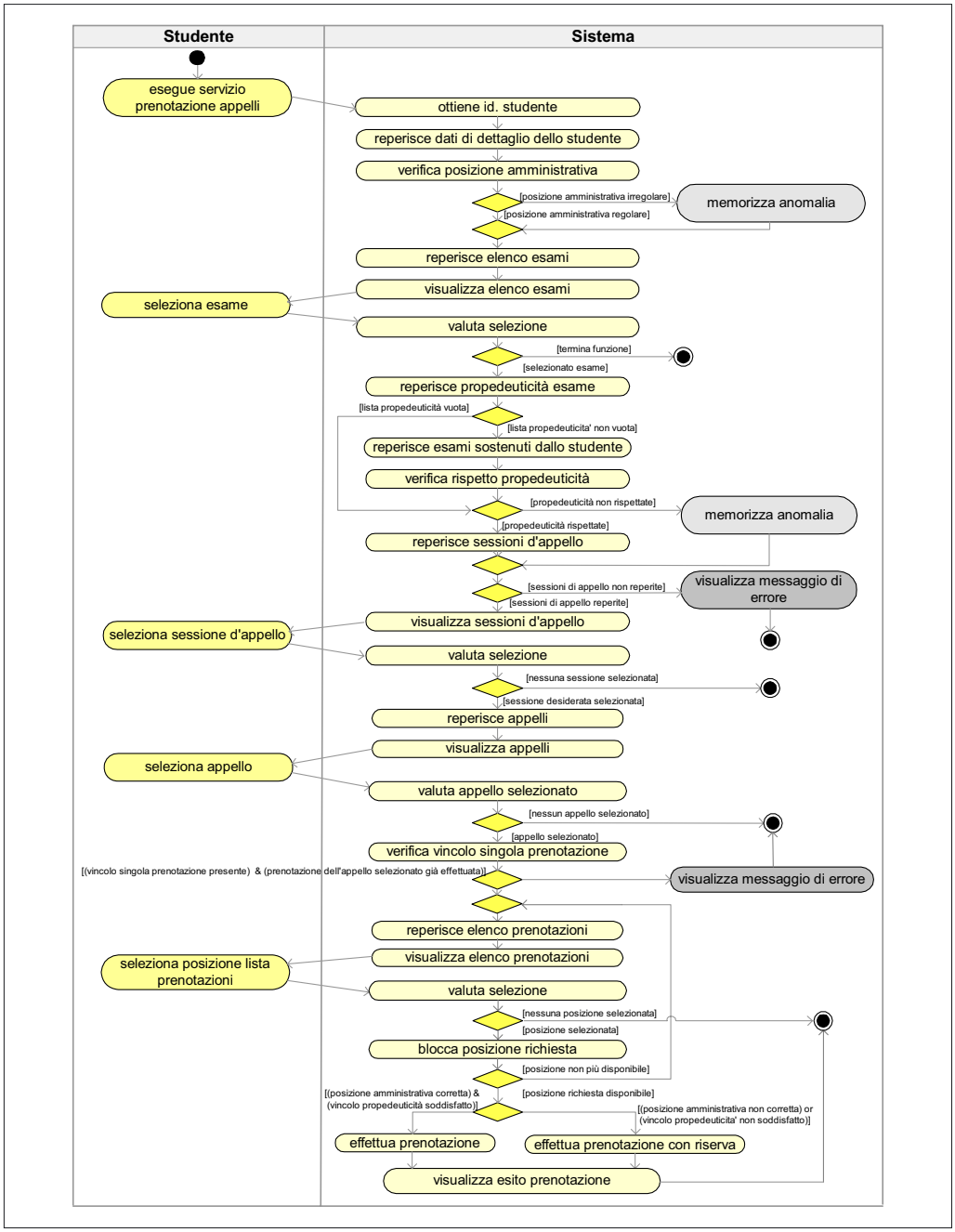
Per via di questi inconvenienti, nella pratica si preferisce illustrare il comportamento dinamico dei casi d'uso attraverso strumenti più vicini al testo.

Come già menzionato si potrebbe ricorrere al flusso di azioni così come da direttive standard UML, sebbene però sia più agevole e frequente l'utilizzo di opportuni template.

L'analisi del sequence diagram in fig. 4.5 potrebbe originare nei lettori più attenti un'obiezione degna di rilievo: se il sistema interagisce con un server, quest'ultimo allora dovrebbe essere un attore nel caso d'uso in questione.

In uno "scenario" logico la risposta è sicuramente affermativa. Chiaramente nessuno vorrebbe replicare i dati di ciascuno studente in più sistemi benché, tipicamente, i clienti risultino abbastanza restii nel concedere l'accesso diretto ai propri database. Sistemi del genere, progettati in maniera accorta (!?), dovrebbero presentare diverse soluzioni di accesso (server socket, CORBA, EJB, COM+, ecc.). In casi — sempre frequenti — di progettazione non completamente lungimirante può capitare di dover interagire con sistemi legacy (per esempio attraverso emulatore 3270) simulando la modalità dell'operatore (*screen scrapling*): si sottopone un opportuno comando al sistema, si attende la schermata di risposta, la si analizza e, se i dati forniti sono completi, si termina la transazione, altrimenti si prosegue con il successivo comando, si rianalizza la schermata di risposta e così via.

Figura 4.6 — *Esempio di activity diagram della funzione di prenotazione esame.*



Nel caso in questione invece si suppone di avere un database ridondante a disposizione del solo sistema Internet — non è assolutamente un caso improbabile — e quindi, essendo parte integrante del sistema, il povero DB server non ha "dignità" di assurgere al ruolo di attore.

Nella fig. 4.6 è riportato un esempio di activity diagram utilizzato per descrivere la funzione prenotazione esami.

Si tenga presente che nella fase del processo di sviluppo relativa alla cattura dei requisiti utente spesso si assiste a una sorta di "guerra" tra gli utenti e il team tecnico e quindi tutti i mezzi che permettono di instaurare una valida piattaforma di dialogo sono benvenuti.

> Nella pratica — come sarà dettagliato nel Capitolo 5 — molto spesso si rivela provvidenziale utilizzare i diagrammi delle attività per descrivere il comportamento dinamico dei casi d'uso per tutta una serie di motivi, tra i quali
>
> 1. trattandosi di un formalismo grafico ne condivide i vantaggi: la mente umana tende a comprendere e memorizzare una maggiore percentuale del contenuto informativo laddove presentato attraverso opportuno formalismo grafico;
> 2. visualizza in maniera inequivocabile la suddivisione di responsabilità tra le varie entità e le operazioni che eventualmente possono essere compiute in parallelo;
> 3. nelle fasi di "cattura" dei requisiti li si utilizza con un livello di astrazione molto elevato che ne evidenzia le affinità con il progenitore flowchart e anche il cliente più ignorante — informaticamente parlando, *of course* — ha familiarità con tale notazione: chi non ha mai realizzato un flowchart in vita sua?
>
> A fronte dei succitati vantaggi, rimangono le condizioni sfavorevoli comuni alle varie notazioni grafiche
>
> 1. all'aumentare della complessità, i diagrammi tendono a diventare inevitabilmente confusi, nonostante il grande dispendio di tempo ed energia profuso dai realizzatori nel vano tentativo di rendere il tutto più lineare possibile;
> 2. problemi pratici di "visualizzazione" che vanno tenuti presenti: diagrammi di dimensioni medio-grandi sono difficili da stampare (spesso richiedono modelli A3 non facilmente reperibili);
> 3. tipicamente richiedono un periodo di tempo maggiore e talune volte decisamente spropositato per la manutenzione, tempo peraltro amplificato poi dall'aumentare della complessità degli stessi.

Nella pratica il formalismo che alla fine risulta più conveniente è decisamente quello dei template, di cui nelle pagine seguenti viene riportata l'applicazione al caso in esame.

Dall'esame dello use case si può notare che in svariati punti la fruizione del servizio può fallire per problemi di carattere tecnico e non per questioni relative all'applicazione delle regole di business. Per esempio il sistema può fallire nell'ottenere una sessione di connessione al database necessaria per ottenere i dati relativi alla situazione amministra-

Tabella 4.2

CASO D'USO: UC_IUBS03	**Prenota appello**	**Data:** 10/01/2001 **Versione:** 0.01.002
Descrizione:	Consente agli studenti universitari, preventivamente autorizzati, di visualizzare le sessioni di esami previste per la materia di interesse ed eventualmente prenotarsi. L'accettazione incondizionata della prenotazione è vincolata al rispetto di eventuali vincoli di propedeuticità previsti dai singoli esami e dalla regolarità della posizione amministrativa.	
Priorità:	Elevata.	
Durata:	Minuti.	
Attore primario:	Studente universitario.	
	Ha interesse a ricevere informazioni relative alle sessioni e agli esami del proprio corso di laurea ed, eventualmente, ha interesse ad iscriversi a singoli appelli di specifici esami	
Attori secondari:	Amministratore del sistema	
	Viene informato qualora si verifichino gravi condizioni di anomalie del sistema.	
Precondizioni:	Lo studente deve essere stato preventivamente riconosciuto dal sistema nel corso della sessione. È disponibile il profilo dello stesso (amministrativo e studi).	
Garanzie:	Minime: La sessione avviata dello studente rimane valida.	
	Successo: Lo studente prenota la sessione di appello desiderata.	
Avvio:	Lo studente richiede esplicitamente l'esecuzione del servizio di prenotazione sessioni di esame.	

Scenario principale.	
1.	Sistema: Reperisce dati studente.
2.	Sistema: Verifica posizione amministrativa studente. *(Cfr. Business Rule STD_VER_POS_AMM)*
3.	Sistema: Reperisce le materie relative alla facoltà/corso a cui lo studente risulta iscritto.
4.	Sistema: Visualizza l'elenco delle materie precedentemente reperite.
5.	Studente: Selezione la materia di interesse.
6.	*Include(Verifica propedeuticità).*
7.	Sistema: Reperisce l'elenco delle sessioni di appello previste per la materia selezionata.
8.	Sistema: Visualizza l'elenco delle sessioni di appello reperite.
9.	Studente: Selezione la sessione ritenuta soddisfacente.
10.	Sistema: Visualizza i diversi appelli previsti per la sessione.
11.	Studente: Selezione l'appello desiderato.
12.	P*unto di estensione: Verifica singola prenotazione per sessione.* *Condizione: L'appello prevede il vincolo di singola prenotazione all'interno di una stessa sessione*
13.	Sistema: Reperisce l'elenco delle prenotazioni per la sessione prescelta.
14.	Sistema: Visualizza l'ordine di accettazione visualizzando unicamente quali "posti" sono disponibili e quali no.
15.	*Punto di estensione: Selezione ordine di accettazione.* *Condizione: Funzione abilitata*
16.	Sistema: Effettua prenotazione.
17.	Sistema: Comunica avvenuta prenotazione.

Tabella 4.3

I scenario alternativo.	
2.1.	Sistema: La posizione amministrativa dello studente non risulta in regola.
2.2.	Sistema: Memorizza condizione di riserva.
II scenario alternativo.	
6.1.	Sistema: La propedeuticità prevista dall'esame non è rispettata dagli esami sostenuti dallo studente.
6.2.	Sistema: Memorizza condizione di riserva.
III scenario alternativo.	
17.1.	Sistema: Presenti condizioni di riserva nella sessione.
17.2.	Sistema: Effettua prenotazione con riserva.
17.3.	Sistema: Comunica allo studente la tipologia della prenotazione e le cause.
IV scenario alternativo.	
17.1.	Sistema: Fallita prenotazione per conflitto di posizione prescelta.
17.2.	Sistema: Comunica situazione anomala.
17.3.	Sistema: Riprende dal punto.
I scenario di errore. Punti: 5, 9, 11, 15.	
5.1.	Studente: Seleziona la terminazione dell'esecuzione del caso d'uso.
5.2.	Sistema: Termina l'esecuzione del caso d'uso con insuccesso.
II scenario di errore.	
7.1.	Sistema: Sessioni di appello per la materia selezionata non disponibili.
7.2.	Sistema: Comunica allo studente l'assenza delle sessioni di appello.
7.3.	Sistema: Termina l'esecuzione del caso d'uso con insuccesso.
III scenario di errore.	
12.1.	Sistema: (Prevista singola prenotazione per appello nell'ambito della stessa sessione) e (lo studente risulta già prenotato ad un appello per la materia nella sessione corrente).
12.2.	Sistema: Comunica l'inibizione della funzionalità allo studente.
12.3.	Sistema: Propone la schermata di cui al punto 4.
Annotazioni.	
14.	Per questioni di tutela della privacy, la funzione di visualizzazione della prenotazione dell'ordine di accettazione dovrebbe visualizzare unicamente le posizioni ancora disponibili e quelle già riservate senza fornire informazioni relative all'identità degli studenti prenotati.

tiva dello studente, nel reperire l'elenco degli esami previsti, la stessa connessione al database potrebbe non funzionare, ecc. In breve possono verificarsi eccezioni per così dire di sistema.

La domanda è se sia il caso o meno di inserire eccezioni di sistema nella descrizione dei casi d'uso. La risposta dell'autore è: probabilmente no. Finirebbe unicamente per rendere i casi d'uso più complessi, pesanti da leggere, ecc. senza peraltro fornire alcuna informazione. Il consiglio pertanto è di non inserire questi casi (a meno di situazioni veramente particolari da trattare in maniera "non standard") nei casi d'uso e prevedere un documento incluso nel modello dei requisiti utente in cui specificare il trattamento di questa tipologia di eccezione. Tale documento si presterebbe a essere rielaborato nelle fasi più tecniche con una sezione relativa alle direttive per il programmatore.

Come si può notare, gran parte della descrizione del comportamento dinamico del caso d'uso è stata assorbita dalla descrizione dell'utilizzo della GUI.

Diversi autori consigliano di non perdere troppo tempo in dettagli di questo tipo e di demandarli al relativo modello/prototipo, qualora presente. Chiaramente è sempre consigliato realizzare qualche forma di navigazione della GUI (da quella completamente automatica detta anche prototipo a quella rappresentata per mezzo di tabelle di un qualsivoglia editor grafico) poiché semplifica il processo di individuazione dei casi d'uso e di flussi alternativi. Altro processo che risente positivamente del prototipo della GUI è quello di verifica: qualora la descrizione di uno use case non dovesse essere consistente con la descrizione della relativa GUI, probabilmente qualche problema potrebbe sussistere.

L'autore è dell'opinione che la specifica delle azioni che un utente tipico esegue interagendo con le schermate del terminale aiuti ad accrescere il livello di comprensione della percezione del punto di vista dell'utente. Non è opportuno tuttavia riferirsi a dettagli, per così dire, "cosmetici" della GUI: l'utente seleziona la materia dalla relativa lista video, preme quindi il pulsante per la visualizzazione della finestra relativa alla descrizione di dettaglio, ecc.

Di questo avviso non sono tutti coloro che, non riuscendo mai a dire un loro parere, visto il relativo livello di "professionalità", si prodigano in guerre di religione, degne delle crociate, sulle interfacce utente (colore dell'etichetta, dimensione del tasto, ecc.).

In sintesi, la descrizione dell'interazione utente/GUI semplifica la comprensione della prospettiva del sistema posseduta dallo stesso. Volendo incorporare i cosiddetti requisiti "speciali", si potrebbe riportare una tabella come la tab. 4.4.

Tabella 4.4

Requisiti speciali.	
1.	In condizioni di massimo utilizzo, il servizio di prenotazione dovrebbe essere fruito, contemporaneamente, da 80/100 studenti. In condizioni di regime il numero si dovrebbe ridurre a 20/30 unità.
2.	Il numero di conflitti, dovuti a studenti impegnati a realizzare una prenotazione per lo stesso appello relativo ad una specifica materia di esame, non dovrebbe mai eccedere le 10 unità.

Dichiarazione di *include* ed *extend* nel template

Si vedranno ora le convenzioni di notazione utilizzate all'interno del template dei casi d'uso per evidenziare dichiarazioni di inclusione e di estensione. Tali convenzioni possono essere estese a qualsiasi altra tipologia di documento atta a descrivere il comportamento dinamico dei casi d'uso.

In primo luogo, l'esecuzione di entrambe le operazioni avviene sotto il controllo del sistema — non avrebbe molto senso evidenziare inclusioni ed estensioni del comportamento degli attori — pertanto è inutile riportare la dichiarazione che è il sistema a eseguire il punto (`Sistema:`).

Per quanto concerne le clausole di inclusione c'è ben poco da dire: è sufficiente riportare nel punto desiderato la dichiarazione `include`, magari in corsivo per conferire più enfasi, con il caso d'uso incluso racchiuso tra parentesi tonde:

```
include (<nome caso d'uso>)
```

Per ciò che attiene la relazione di estensione, la situazione è più complessa per tutta una serie di motivi, quali:

- l'estensione avviene sotto il controllo di una condizione;
- un caso d'uso estendente può estenderne svariati altri in diversi punti;
- un singolo caso d'uso può essere esteso negli stessi punti da diversi casi d'uso.

Le ultime due argomentazioni possono essere sintetizzate riportando che esiste una relazione *n* a *n* tra i punti che possono essere estesi di un caso d'uso esteso e i segmenti che ne specificano il comportamento nei casi d'uso estendenti.

La prima considerazione genera il problema di quale use case (esteso o estendente) debba ospitare la condizione di estensione. La risposta corretta sarebbe: nessuno dei due.

Si tratta infatti di una condizione relativa alla relazione stessa e non ai singoli casi d'uso. Pragmaticamente però, nella maggior pare dei casi è conveniente specificarla nel caso d'uso esteso, ciò perché uno stesso caso d'uso estendente può estenderne svariati in funzione di diverse condizioni.

> Specificare nello use case esteso i casi d'uso estendenti, da un punto di vista formale, costituirebbe un problema: lo use case esteso non ha conoscenza di quanti e quali casi d'uso lo estendono… Però, in questo caso, si tratta esclusivamente di una convenzione che agevola produzione e manutenzione della documentazione. Per esempio, la dichiarazione dei casi d'uso estendenti all'interno di quelli estesi consente, tool permettendo, di associare tra loro i vari documenti per mezzo di opportuni collegamenti (link) e quindi poterli scorrere naturalmente in fase di lettura.

Inserire tale clausola nello use case esteso può dare luogo a ripetizioni (più use case estesi da uno stesso in funzione della medesima condizione) e può dar luogo a noiosi elenchi di condizioni (più use case ne estendono uno nello stesso punto, quindi per ogni relazione di estensione va riportata la clausola), tuttavia si tratta di una tecnica consistente. La circostanza in cui uno stesso use case ne estende altri in diversi punti si risolve dichiarando:

1. all'interno dei casi d'uso estesi, uno specifico nome (magari riportato ancora in carattere corsivo per evidenziarlo) per ogni punto il cui comportamento può essere esteso. Chiaramente tale nome deve essere univoco all'interno di ogni caso d'uso. Inoltre, come illustrato poc'anzi, è opportuno riportare anche la condizione:

```
Punto di estensione:   <nome punto di estensione>
Condizione:            <definizione condizione>
```

2. all'interno dei casi d'uso estendenti, per ogni segmento definito, il punto dello use case esteso:

```
Definizione punto di estensione: <nome punto di estensione>
```

Quindi, se la condizione risulta soddisfatta, si può immaginare che nello use case esteso vengano sostituiti tutti i punti di estensione dichiarati, con i segmenti corrispondenti nel caso d'uso estendente (l'abbinamento avviene attraverso il parametro <nome punto di estensione>).

Per quanto concerne la relazione di generalizzazione il discorso è abbastanza semplice. Lo use case che specializza quello base deve menzionare tutti i punti in cui il comportamento "base" va specializzato, eventuali punti aggiuntivi, punti da non eseguire, ecc.

Esempio: sistema di banking

Di seguito viene preso in esame lo studio di un'infrastruttura per il supporto dell'area finanziaria (*finance*) di una banca (*investment bank and market*).

> L'autore si scusa se le varie parti presentate in questo paragrafo appaiono inquadrate in maniera non completa in un contesto difficilmente riconoscibile. La ragione di certe reticenze esiste: come al solito incombono vitto e alloggio a spese dello Stato…

Prima di addentrarsi nell'esame dei vari casi d'uso selezionati si ritiene opportuno presentarne brevemente l'ambiente.

Tipicamente le strutture informatiche di sistemi complessi come quelli bancari/di investimento prevedono una serie di sottosistemi, ognuno specializzato nella erogazione di un insieme ben definito di servizi, che cooperano (scambiandosi messaggi) al fine di raggiungere l'obiettivo comune di fornire determinati servizi (situazione del sistema di sistemi esaminata nel Capitolo 2 *UML: struttura, organizzazione, utilizzo*).

Esempi tipici di sottosistemi (fig. 4.7) che si possono trovare in un sistema bancario sono quello Internet (il famoso `.com`), il sottosistema specializzato per la gestione del front office (sportello), quello demandato all'accounting, altri finalizzati alla gestione degli investimenti e relativi controlli, quello dedicato alla sicurezza, quello atto a controllare che il sistemi funzioni correttamente (health monitoring), altri ancora dedicati alla gestione in tempo reale dei dati delle quotazioni di mercato dei prodotti finanziari e così via.

L'interconnessione tra i vari sistemi è ottenuta di solito attraverso opportuni middleware indicati con l'acronimo *MOM* (Messaging Oriented Middleware, sistemi orientati alla gestione della messaggistica). L'obiettivo è disaccoppiare il più possibile i vari sistemi: una volta il problema era a livello di classi o al più di package, ma oggigiorno il livello di astrazione è in crescita continua.

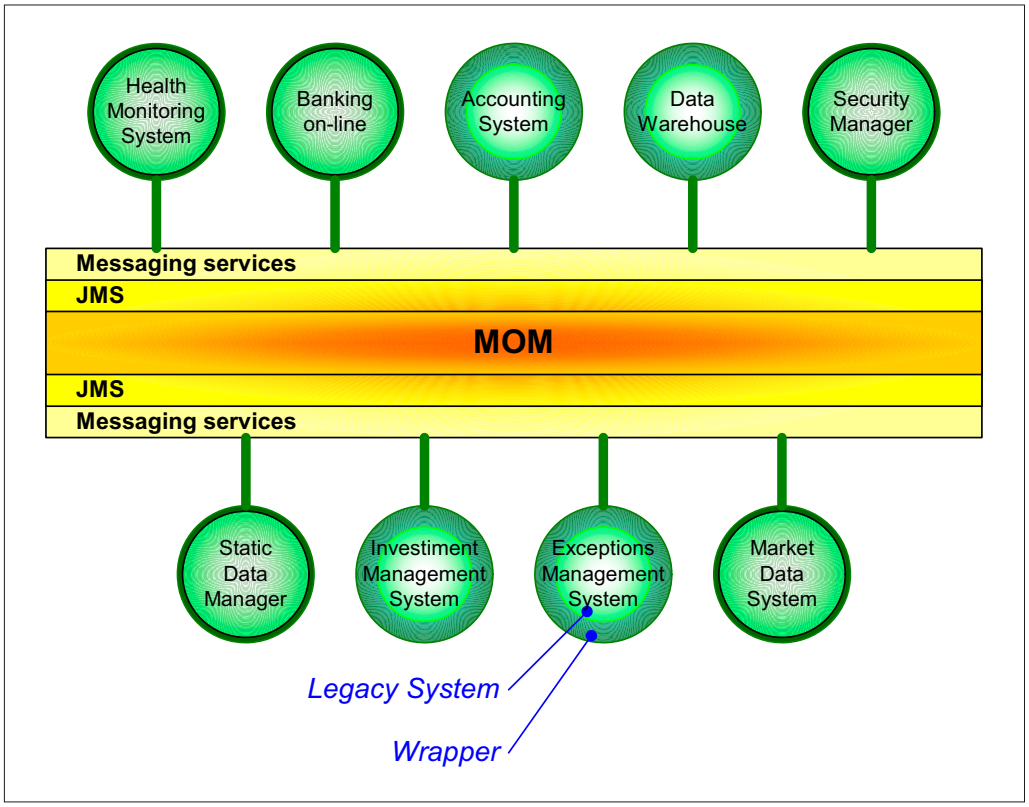
Al momento in cui questo libro viene redatto si assiste sempre più frequentemente all'utilizzo di MOM di nuova generazione basati su architetture JMS come da specifiche Sun (Java Messaging System, sistema di messaggistica Java), in grado di funzionare sia in modalità *point to point* (PTP, punto a punto), sia in *publish and subscribe* (Pub&Sub, pubblica e sottoscrivi).

Secondo questo scenario, il bus di comunicazione (MOM) viene ripartito in una serie di canali logici in funzione della tipologia di messaggi da scambiare (dati dinamici, dati statici, dati necessari per fini statistici, dati riportanti segnalazioni di anomalie, ecc.).

Un sistema per emettere/ricevere messaggi deve necessariamente e preventivamente registrarsi (*subscribe*) al o ai canali sui quali intende, rispettivamente, trasmettere (*publish*) e ricevere.

Come evidenziato in figura, l'attuazione di architetture di questo tipo prevede la necessità di adattare legacy system di vecchia concezione ad architetture più moderne. Il problema viene affrontato cercando di incapsulare il sistema stesso in un apposito layer

Figura 4.7 — *Frammento di un esempio di architettura di un sistema bancario. Nello scenario mostrato, alcuni sistemi risultano ex-novo (Static Data Manager, Market Data System, Banking on-line) e quindi sono realizzati direttamente per dialogare via MOM con altri sistemi. Gli altri, essendo di vecchia generazione (Investment Management System, Exceptions Management System, Accounting System, Data Warehouse), necessitano di un apposito strato di wrapping.*



(wrapper, ancora una volta) in grado di comunicare con i due mondi: MOM da una parte e legacy system dall'altra.

Altri sistemi invece (come per esempio quello di banking online) essendo di nuova generazione, dovrebbero venir progettati per poter interagire naturalmente con middleware di comunicazione e quindi non hanno bisogno di particolari strutture di wrapping.

Analizzando gli attori che interagiscono con i vari sottosistemi, potrebbe nascere un dilemma. Per ciò che concerne gli utenti veri e propri, non ci sarebbe nulla di nuovo, ma per quanto riguarda l'interazione con altri sottosistemi, potrebbe nascere il dubbio se

visualizzare unicamente il MOM come attore "tecnologico" oppure, di volta, in volta menzionare i vari sottosistemi.

L'alternativa ritenuta più corretta dall'autore è la seconda. In primo luogo perché l'attore MOM sarebbe troppo legato allo spazio delle soluzioni e poco a quello del problema: si tratta di un meccanismo di comunicazione e probabilmente sarebbe come dire che il modem o il telefono potrebbero, a loro volta, assurgere al ruolo di attori.

Trattandosi poi di un attore unico "scudo" di ciascun sottosistema, quindi assolutamente privo di "volontà e necessità proprie", renderebbe più difficile inquadrare i servizi in un contesto organico di cooperazione tra sottosistemi.

Un'altra argomentazione contraria al MOM come attore è relativa al fatto che l'utente medio, che, per definizione, non dispone di conoscenze informatiche, tende a non comprendere dettagli di soluzioni tecnico-architetturali e quindi ad aver problemi nell'identificare il MOM come attore.

Wrapper sistema di gestione dei trade

Si cominci con l'analizzare un frammento di un dispositivo di wrapper atto a gestire la manutenzione dei dati statici nel sistema di gestione dei trade (Trade Management System, TMS).

Responsabilità della sezione del wrapper preso in esame è ricevere i dati statici e incorporare gli aggiornamenti nel TMS. Analizzando lo schema dell'architettura (figura 4.7), è possibile evidenziare un sistema denominato *Static Data Manager* (Gestore dei Dati Statici), il cui compito è assicurare l'uniformità dei dati, definiti tecnicamente statici, utilizzati dall'intero sistema. Esempi di dati statici sono le informazioni relative alle "controparti" (entità che prendono parte in un trade, acquistano e/o vendono prodotti finanziari), alle valute (Euro, Sterlina, ecc.), alle istruzioni per i "settlement", ai prodotti finanziari, ecc. In particolare, lo Static Data Manager, prevede tipicamente tre macro componenti:

1. un'interfaccia utente (tipicamente web-based), atta a consentire all'utente la gestione dei dati statici (presentation layer);

2. un middle-tier di integrazione con il database. Espone una serie di servizi atti a reperire i dati da mostrare nell'interfaccia utente e conseguentemente per memorizzare gli eventuali aggiornamenti apportati dall'utente;

3. un sistema di messaggistica atto a pubblicare i dati, attraverso opportuni messaggi immessi sul MOM, in risposta agli aggiornamenti effettuati.

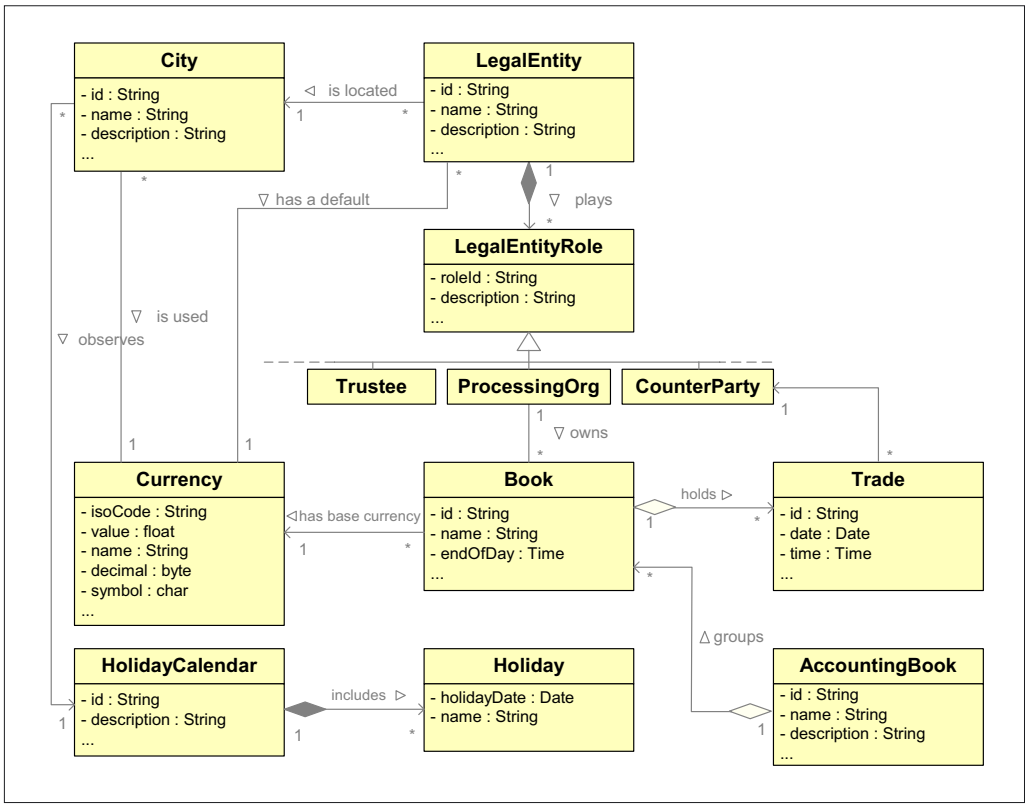
Per lo scambio dei messaggi si preferisce ricorrere a formati non proprietari, quali per esempio XML (al momento in cui viene scritto il libro è a tutti gli effetti lo standard *de facto*) per

consentire a sistemi utilizzanti tecnologie diverse di comunicare tra loro. Ciò implica la necessità però di effettuare il parsing e la conversione dello stesso in una forma più vicina al sistema (oggetti opportunamente relazionati).

A questo punto si consideri la porzione del wrapper che consente di aggiornare i dati relativi ai libri (*book*) del sistema finanziario (fig. 4.8).

Brevemente i *book* sono le più piccole unità di un'organizzazione finanziaria, utilizzate per organizzare i *trade*. Ognuno di essi appartiene a una sottoorganizzazione (*Processing Organisation,* specializzazione di una *LegalEntity*) dell'istituto finanziario, sita in una città ben definita. Questa associazione è molto importante perché permette di stabilire molte informazioni, quali l'orario della chiusura delle attività finanziare della città di riferimento e quindi l'orario di avvio di opportune procedure batch (come per esempio rivalutazioni di fine giornata, *end of*

Figura 4.8 — *Frazione del modello a oggetti del dominio relativo al book.*



day revaluation), il calendario delle festività (*Holiday Calendar*) la cui conoscenza permette di evitare che specifiche date (maturazione trade) cadano in giorni festivi, ecc. Da tener presente che non sempre le *Processing Organisation* utilizzano la valuta della città/nazione in cui sono ubicate e che il calendario delle festività di una determinata città non coincide con quello della nazione di appartenenza (si considerino, per esempio, le feste patronali). Per ciascun *Book* è anche necessario poi specificare la valuta base di riferimento. Brevissimamente, un *Trade* è un contratto tra una *Processing Organisation* e un'altra entità (*CounterParty*) negoziato per lo scambio di determinati prodotti finanziari.

Sebbene il formalismo del diagramma delle classi mostrato non sia stato ancora illustrato in dettaglio, il suo significato dovrebbe essere sufficientemente chiaro.

Si tratta di un frammento (quello relativo ai *book* ovviamente) di un *modello a oggetti del dominio*.

I modelli ad oggetti sono trattati nel corso dei Capitoli 7 e 8. Per ora basti sapere che il modello del dominio è una rappresentazione concettuale delle entità coinvolte nell'area business oggetto di studio. Si tratta di un manufatto di estrema importanza. Fornisce una prima versione utile per il modello di disegno, per l'organizzazione in componenti del sistema, per la progettazione del database e dell'interfaccia utente, ecc.

Nel contesto dei casi d'uso si consiglia vivamente di riferirsi ai diagrammi a "oggetti" del dominio. La presenza questi manufatti permette di avere una chiara e completa visione dei requisiti utente: i casi d'uso sono focalizzati sui servizi da erogare, mentre il modello a oggetti del dominio mostra l'organizzazione dei dati coinvolti. Lo sviluppo congiunto dei diagrammi favorisce una migliore comprensione dell'area oggetto di studio, permette di bilanciare gli stessi diagrammi dei casi d'uso, favorisce il conferimento della giusta enfasi ai dati principali di eventuali interfacce, aiuta a evidenziare la necessità di pianificare specifiche funzioni, e così via. Con il termine "bilanciare", si intende dire che, una volta catturati i requisiti del sistema, si vuole disporre di diagrammi dei casi d'uso e del dominio (o business) corretti, completi e consistenti.

Un modo per bilanciare il modello dei casi d'uso con quello ad oggetti del dominio consiste nel verificare che:

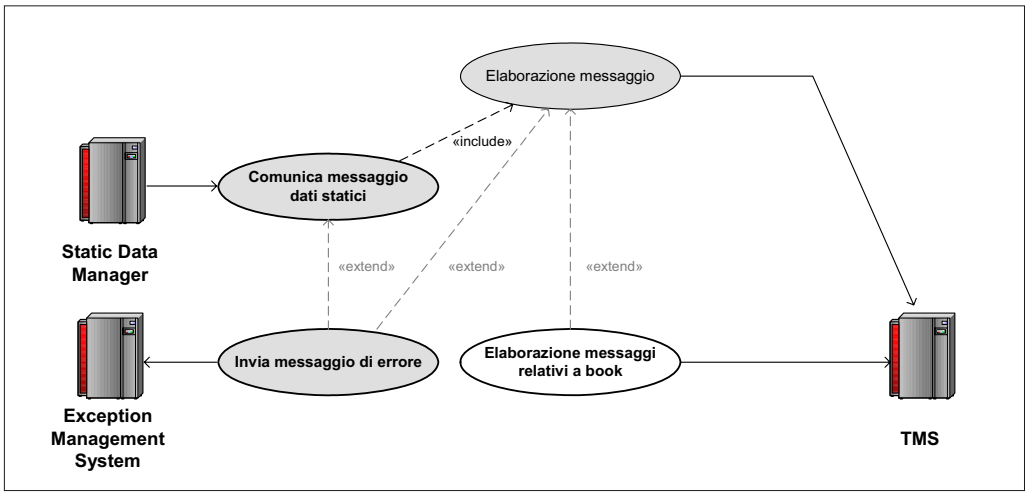
1. tutti gli oggetti menzionati nei casi d'uso siano presenti nel modello a oggetti del dominio, opportunamente relazionati (bilanciamento del modello del dominio rispetto ai casi d'uso);

2. tutte le classi presenti nel modello del dominio, siano, in qualche misura, riferite nei diagrammi dei casi d'uso, con particolare riferimento alle funzionalità di cui hanno bisogno (inserimento, aggiornamento, ecc.) e/o che sono in grado di provvedere al bilanciamento dei casi d'uso rispetto al modello a oggetti del dominio.

Per esempio, nel caso in cui non fosse ben chiara la relazione tra le entità *Book* e *Trade*, analizzando le funzioni necessarie per l'elaborazione di un messaggio relativo a un *Book*, e in particolare relativo alla eliminazione dello stesso, si potrebbe correre il rischio di non considerare alcune casistiche (flussi) di una certa importanza e, conseguentemente, lasciare inesplorate diverse aree. Potrebbe ad esempio sfuggire la necessità di prevedere una funzione di reperimento dei *Trade* associati a un *Book*. Questa funzione è necessaria per assicurare che, dovendo eliminare un *Book,* questo non abbia *Trade* associati, in quanto in caso contrario, procedendo con l'eliminazione, si perderebbero tutti i dati dei *Trade* ad esso associati (magari qualche milione di Euro…). A tal fine sarebbe necessaria una semplice operazione di check (*Trade* presenti o meno); invece si ricorre a un reperimento in quanto, qualora presenti dei *Trade* associati al book da eliminare, i relativi identificativi andrebbero comunicati al sistema di risoluzione delle eccezioni. Ciò al fine di far precedere alla procedura di eliminazione di un *Book* quella di "spostamento" dei *Trade* dal *Book* da rimuovere a un altro.

Gli use case colorati in grigio di fig. 4.9 mostrano le funzionalità comuni eseguite conseguentemente alla ricezione di un messaggio, indipendentemente dalla natura dello stes-

Figura 4.9 — *Diagramma dei casi d'uso relativo all'elaborazione del messaggio inerente aggiornamenti relativi all'entità "book".*



so. Verosimilmente, con riferimento alla descrizione dell'architettura riportata in fig. 4.7, i primi quattro use case rappresentano lo strato ribattezzato *Messaging services* utilizzabile dai vari sistemi di wrapper.

Tabella 4.5

CASO D'USO: UC_BNK_FRM_01	**Comunica messaggio dati statici**	**Data:** 13/02/2001 **Versione:** 0.01.000
Descrizione:	Il sottosistema SDM notifica, al sistema *wrapper*, attraverso opportuno messaggio pubblicato sul MOM, l'avvenuta variazione del proprio stato (variazione di dati statici) al fine di consentire agli altri sottosistemi di sincronizzarsi di conseguenza.	
Priorità:	Media. *(Nelle prime versioni del sistema questa funzionalità può essere simulata attraverso apposite infrastrutture).*	
Durata:	Secondi.	
Attore primario:	SDM (Static Data Manager).	
	Ha la responsabilità di pubblicare messaggi contenenti informazioni relative ad aggiornamenti eseguiti sui dati statici.	
Precondizioni:	Il sistema si sia preventivamente sottoscritto per la ricezione della tipologia del messaggio.	
Garanzie:	**Minime:** Il messaggio rimane disponibile.	
	Successo: Il sistema prende correttamente in carico il messaggio e riflette le relative operazioni nella propria base dati	
Avvio:	Viene pubblicato un messaggio appartenente ad una delle tipologie per la quale il sistema si è sottoscritto.	

Scenario principale.	
1.	SDM: Pubblica un nuovo messaggio relativo a variazioni dei dati statici.
2.	Sistema: Acquisisce il messaggio.
3.	Sistema: *Include(Elaborazione messaggio)*
Primo scenario di errore.	
2.1.	Il sistema fallisce nell'acquisizione del messaggio
2.2.	*Punto di estensione: Notifica condizione di errore.* *Condizione: Verificatosi problema nell'acquisizione del messaggio*
2.3.	Termina lo use case in maniera anomala
Annotazioni.	
	Il presente caso d'uso appartiene al modello relativo al sistema wrapper, e quindi va analizzato dall'ottica di tale sotto-sistema. Per quanto attiene i casi d'uso del sistema di gestione dei dati statici (SDM), sono disponibili nel relativo modello.

Prima di descrivere il comportamento dinamico è inevitabile una breve discussione sul livello di dettaglio dei casi d'uso descritti.

> Come si potrà notare sia il linguaggio utilizzato, sia la particolare area di cui sono analizzati i requisiti, presentano un elevato livello tecnico. Secondo quanto riportato nel capitolo precedente, si tratta di casi d'uso appartenenti al modello di sistema (*System use case*). Ciò è facilmente ravvisabile dal fatto che i casi d'uso sono calati nell'organizzazione architetturale del sistema (per esempio alcuni attori sono specifici sottosistemi). Con le peculiarità testé evidenziate, si tratta comunque di casi d'uso che necessitano di essere specificati esplicitamente: le relative funzionalità devono essere implementate e verificate. Il caso d'uso veramente rilevante, quello contenente importanti regole business, è `Elaborazione messaggi relativi a book`, mentre i restanti hanno prevalentemente una funzione di "collante". Il comportamento di questi ultimi, per via delle peculiarità succitate, non si prestano ad essere specificati dagli utenti.

Tabella 4.6

CASO D'USO: UC_BNK_FRM_03	**Invia messaggi di errore**	**Data:** 14/02/2001 **Versione:** 0.00.001
Descrizione:	Pubblica un apposito messaggio al fine di notificare all'Exception Management System il verificarsi di un'anomalia. La ricezione di questi messaggi innesca un specifico processo per la relativa gestione	
Priorità:	Media.	
Durata:	Secondi.	
Attore primario:	Exception Management System.	
	Riceve il messaggio di errore emesso dal sistema.	
Precondizioni:	Il sistema disponga dei dati atti a descrivere l'errore verificatosi.	
Garanzie:	**Minime:** I dati efferenti l'errore restano inalterati.	
	Successo: Il messaggio viene correttamente pubblicato.	
Scenario principale.		
1.	*Definizione punto di estensione: Notifica condizione di errore.*	
2.	Sistema: Struttura secondo il formato previsto il messaggio di errore.	
3.	Sistema: Pubblica il messaggio.	
Primo scenario di errore.		
3.1.	Il sistema fallisce l'invio del messaggio.	
3.2.	Registra l'evento nel file di log.	
3.3.	Termina lo use case in maniera anomala.	
Annotazioni.		
3.1.	**PUNTO DA DISCUTERE:** Cosa fare quando si fallisce la pubblicazione di un errore? Sufficiente registrare l'anomalia in un opportuno file di log?	

Tabella 4.7

CASO D'USO: UC_BNK_FRM_04	**Elaborazione messaggio**	**Data:** 16/02/2001 **Versione:** 0.00.001
Descrizione:	Elabora il messaggio ricevuto in base alla relativa tipologia e alla funzione da eseguire (inserimento, aggiornamento e cancellazione).	
Priorità:	Media.	
Durata:	Secondi.	
Attore primario:	Trade Management System (TMS)	
	Il sistema di gestione dei trade ha la responsabilità di eseguire la transizione specificata nel messaggio al fine di mantenere i propri dati statici sincronizzati con quelli presenti presso il SDM.	
Precondizioni:	Sono disponibili i dati relativi al messaggio ricevuto.	
Garanzie:	**Minime:** I dati relativi al messaggio non vengono alterati.	
	Successo: Il messaggio viene elaborato correttamente in funzione alla sua tipologia e la transazione specificata viene eseguita correttamente dal TMS.	

Scenario principale.	
1.	Sistema: Acquisisce le regole per la trasformazione del messaggio.
2.	Sistema: Trasforma il contenuto del messaggio in una rappresentazione OO consona al sistema.
3.	*Punto di estensione: Elabora il messaggio.* *Condizione: Tipologia messaggio = book* *Esteso dallo use case "Elabora messaggi relativi ai book"* *Tipologia messaggio = trade* *Esteso dallo use case "Elabora messaggi relativi a Trade"* *...*
4.	Sistema: Richiede al sistema TMS di eseguire la transazione scaturita dall'elaborazione del messaggio al punto 3.
5.	TMS: Riceve ed esegue la transazione.
6.	TMS: Comunica esito transazione.
7.	Sistema: Verifica che la transazione sia stata eseguita correttamente
Primo scenario di errore.	
2.1.	Sistema: Fallisce la trasformazione del messaggio.
2.2.	*Punto di estensione: Notifica condizione di errore.* *Condizione: Trasformazione del messaggio fallita*
2.3.	Sistema: Termina lo use case in maniera anomala
Secondo scenario di errore.	
3.1.	Sistema: Fallisce il processo di elaborazione del messaggio.
3.2.	*Punto di estensione: Notifica condizione di errore.* *Condizione: Trasformazione del messaggio fallita*
3.3.	Sistema: Termina lo use case in maniera anomala
Terzo scenario di errore.	
7.1.	Sistema: Verifica fallimento transazione TMS
7.2.	*Punto di estensione: Notifica condizione di errore.* *Condizione: Trasformazione del messaggio fallita*
7.3.	Sistema: Termina lo use case in maniera anomala.

Tabella 4.8

CASO D'USO: UC_BNK_BKM_02	**Elaborazione messaggi relativi a book**	**Data:** 16/02/2001 **Versione:** 0.00.004
Descrizione:	Elabora il messaggio inerente il *Book* specificato. In particolare, in funzione del contenuto del messaggio, il sistema deve eseguire una delle seguenti transazioni sui dati del *Book* specificato: - inserimento nuova istanza; - aggiornamento di una istanza; - eliminazione di una istanza.	
Priorità:	Media.	
Durata:	Secondo.	
Attore primario:	TMS	
	Comunica i dati relativi al *Book* richiesto.	
Precondizioni:	Il messaggio ricevuto è relativo ad un *Book*. La versione OO del messaggio è disponibile nel sistema.	
Garanzie:	**Minime:** I dati relativi al messaggio non vengono alterati.	
	Successo: Il messaggio viene elaborato correttamente, e viene preparata la transazione da eseguire.	

Scenario principale.	
1.	*Definizione punto di estensione: Elabora il messaggio.*
2.	Sistema: Acquisisce il codice del *Book* e il codice della transazione da eseguire.
3.	Sistema: Richiede al sistema TMS le informazione relative al *Book* oggetto del messaggio.
4.	TMS: Comunica le informazioni relative al *Book* specificato.
5.	Sistema: Memorizza i dati ricevuti dal sistema TMS.
6.	Sistema: Verifica che sia richiesta un'operazione di aggiornamento.
7.	Sistema: Verifica che i dati del *book* reperiti dal sistema TMS siano dello stesso *book* da modificare.
8.	Sistema: Organizza i dati da aggiornare in una transazione compatibile con il sistema TMS.
9.	Sistema: Termina con successo.
Primo scenario alternativo.	
6.1.	Sistema: Verifica che sia richiesta un'operazione di inserimento.
6.2.	Sistema: Verifica che non sia stato reperito nel sistema TMS alcun *book* (il *book* non esiste).
6.3.	Sistema: Organizza i dati da inserire in una transazione compatibile con il sistema TMS.
6.4.	Sistema: Termina con successo.
Secondo scenario alternativo.	
6.1.	Sistema: Verifica che sia richiesta un'operazione di eliminazione.
6.2.	Sistema: Verifica che i dati del *book* forniti dal sistema TMS siano dello stesso *book* da eliminare.
6.3.	Sistema: Organizza i dati da eliminare in una transazione compatibile con il sistema TMS.
6.4.	Sistema: Termina con successo.
Primo scenario di errore.	
7.1.	Sistema: Operazione di aggiornamento e *book* non reperito.
7.2.	*Punto di estensione: Notifica condizione di errore.* *Condizione: Trasformazione del messaggio fallita*
7.3.	Sistema: Termina lo use case in maniera anomala
Secondo scenario di errore.	
7.2.1.	Sistema: Operazione di inserimento e *book* reperito. (Il sistema già dispone di un'istanza del book da inserire)
7.2.2.	*Punto di estensione: Notifica condizione di errore.* *Condizione: Trasformazione del messaggio fallita*
7.2.3.	Sistema: Termina lo use case in maniera anomala
Secondo scenario di errore.	
7.2.1.	Sistema: Operazione di eliminazione e *book* non reperito..
7.2.2.	*Punto di estensione: Notifica condizione di errore.* *Condizione: Trasformazione del messaggio fallita*
7.2.3.	Sistema: Termina lo use case in maniera anomala

Qualora si abbia a che fare con sistemi di messaggistica è utile riportare uno o più specifici use case per la comunicazione di messaggi in quanto semplificano il processo di individuazione e bilanciamento dei messaggi presenti nel sistema.

Casi d'uso di tipo CRUD

L'acronimo CRUD (*Create*, *Read*, *Update*, *Delete* – crea, leggi, aggiorna, cancella) è utilizzato dalla comunità informatica per indicare i servizi che permettono di gestire completamente opportuni gruppi di dati (per esempio dati pazienti, dati medicinali, dati ospedali, ecc.).

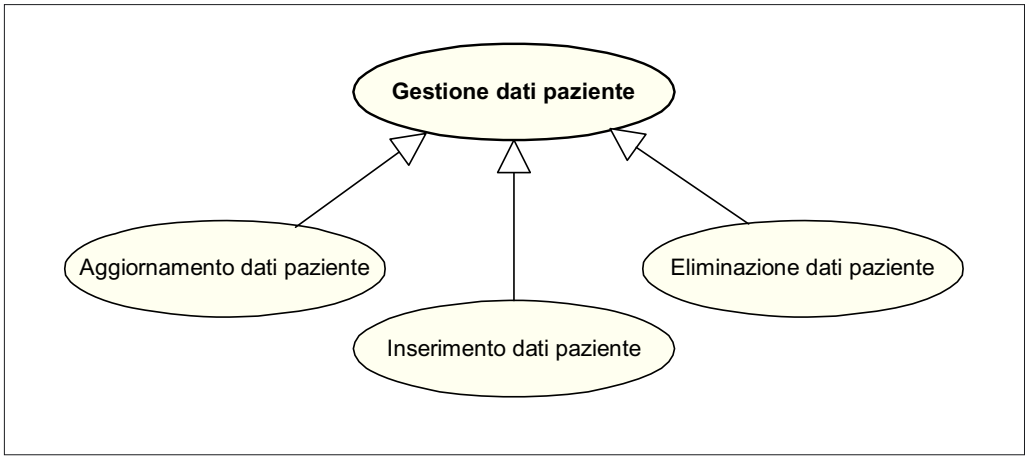
Si tratta di funzionalità particolarmente ricorrenti nei sistemi informatici che quindi devono essere realizzati come specificato dai relativi requisiti utenti e documentati per mezzo di opportuni casi d'uso. Questi casi d'uso sono caratterizzati dal possedere una struttura ben definita e, tipicamente, un numero piuttosto contenuto di regole business. Ciò è particolarmente vero qualora si disponga di un modello a oggetti del dominio: dalla relativa analisi è possibile desumere molte regole. Per esempio dall'analisi del diagramma di fig. 4.8 è possibile desumere regole quali: non è possibile inserire un'istanza `Book` se non è presente l'istanza della valuta base, ciascun `book` deve essere necessariamente assegnato ad una `LegalEntity`, ecc.

A questo punto viene di interrogarsi circa la modalità migliore di rappresentare casi d'uso di tipo CRUD. Da un punto di vista strettamente teorico non ci dovrebbero essere alternative: ogni servizio rappresenta un caso d'uso distinto che estende quello base di gestione della tipologia dei dati (cfr. fig. 4.10). Un esempio è mostrato nei casi d'uso di gestione del carrello della spesa mostrati nell'esempio conclusivo del capitolo.

A favore di questa soluzione esistono diverse argomentazioni:

- ogni caso d'uso "estendente" ha diversi obiettivi (garanzie di successo o post-condition): aggiornamento dei dati, inserimento di una nuova istanza, cancellazione dati;
- ciascun caso d'uso ha diverse precondizioni;
- i singoli casi d'uso sono più lineari;
- la visualizzazione esplicita dei diversi casi d'uso fornisce migliori informazioni sia al processo di organizzazione del sistema in componenti, sia al workflow di gestione del progetto;
- eventualmente è possibile evidenziare attori abilitati o meno ad eseguire determinati servizi (per esempio si potrebbe distinguere tra ruoli abilitati a reperire dati ed altri autorizzati ad aggiornarli).

Figura 4.10 — *Struttura casi d'uso della tipologia CRUD.*



Ciò nonostante, considerata la "tediosa" ripetitività dei casi d'uso CRUD, spesso si preferisce utilizzare un approccio diverso, più pragmatico, che consiste nel collassare i vari casi d'uso specifici in quello base, strutturando appropriatamente i vari *scenario*. Questa tecnica è stata utilizzata per il caso d'uso `Elaborazione messaggi relativi a book` (*cfr* fig. 4.9). Ciò diviene naturale in tutte quelle situazioni in cui i casi d'uso specializzati presentano una sequenza di azioni piuttosto contenuta.

A favore di questa alternativa esistono alcuni punti su cui può valere la pena riflettere. In primo luogo può capitare di dover realizzare sistemi con tante funzionalità di tipo CRUD molto simili tra loro. In questo caso, al fine di velocizzare l'analisi dei requisiti e contenere "l'esplosione" dei casi d'uso, è possibile ricorrere alla versione collassata. Un'altra considerazione è relativa alla percezione che gli utenti hanno dei casi d'uso astratti. Tipicamente, sono più difficili da far comprendere e la stessa fruizione non è lineare (generalmente è necessario).

Sintetizzando, da un punto di vista strettamente teorico è opportuno realizzare un caso d'uso per ogni servizio mentre, in talune circostanze, utilizzando un approccio più pragmatico, è del tutto accettabile collassare i vari casi d'uso in uno solo.

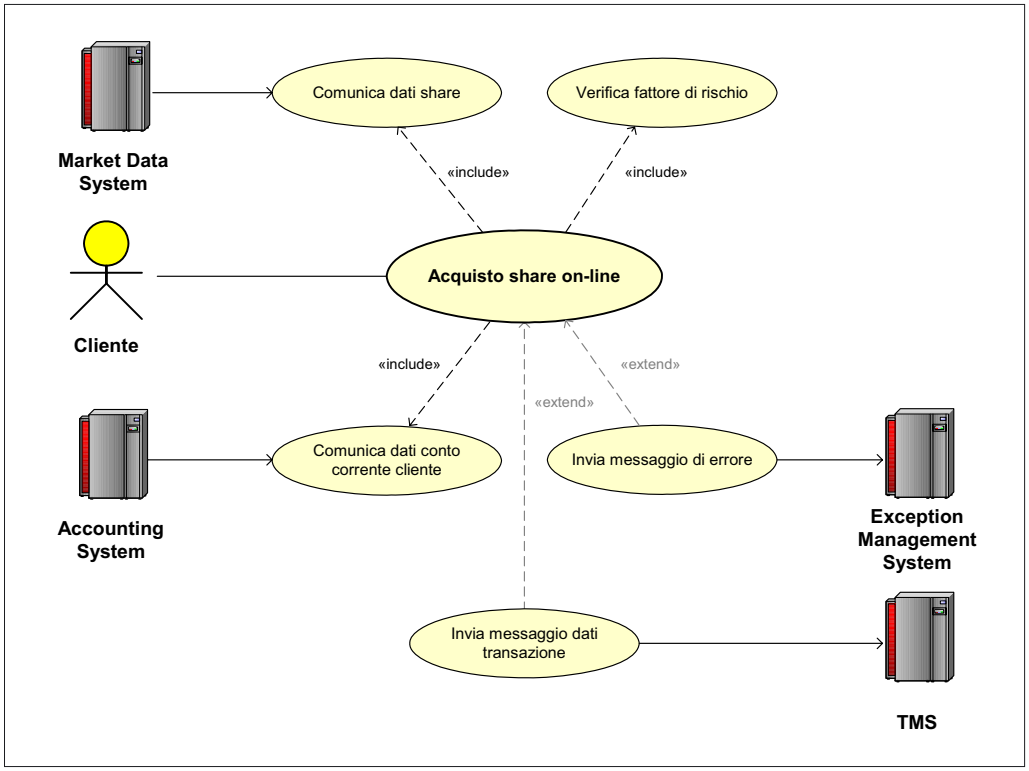
Sistema di banking online

Si illustrerà uno dei servizi offerti dal sistema di banking online; in particolare viene presentata la funzione che consente ai clienti accreditati di acquistare azioni (*shares*) online. I clienti devono possedere un conto corrente presso la banca e aver ottenuto conferma alla relativa richiesta di fruizione del servizi di trading online. Si tratta del solito ozioso utente condannato a starsene seduto comodamente a casa sua e a navigare in Internet.

Questo servizio, sebbene, per ovvi motivi, presentato in una versione semplificata, non è stato selezionato casualmente, ma in virtù degli spunti che è in grado di offrire. In particolare viene evidenziato come gli use case appartenenti al modello di sistema, oltre a inglobare indicazioni provenienti dall'architettura dello stesso, incorporano sezioni, o meglio suggerimenti, derivanti dai requisiti non funzionali.

Una struttura online, destinata a operare secondo le specifiche del protocollo sincrono HTTP, difficilmente potrebbe funzionare utilizzando unicamente i meccanismi offerti da sistemi MOM. In particolare, il sistema dovrebbe soddisfare anche precisi requisiti non funzionali, come per esempio stringenti tempi di risposta non garantiti dai sistemi di messaggistica. Questi, tipicamente, si fanno carico di una serie di assicurazioni (i messaggi vengono consegnati correttamente e solo una volta), ma non forniscono alcuna certezza sulle prestazioni, sebbene poi nella stragrande maggioranza dei casi riescano a effettuare la consegna nell'arco dei secondi. Ulteriore considerazione è relativa al tempo necessario

Figura 4.11 — *Use Case relativo all'acquisto on-line di share.*



per costruire i vari messaggi in formato XML (in spedizione) e per trasformarli nuovamente in un apposito grafo di oggetti (in ricezione). Tempi che non sempre sono trascurabili.

Il vincolo di ricevere informazioni molto rapidamente è una restrizione decisamente pressante non solo per motivi legati alla fruizione del servizio in Internet, ma anche per la natura del servizio stesso. Per esempio, dovendo visualizzare le quotazioni in tempo reale delle azioni, non si può di certo tollerare di ricevere i dati con una latenza tale da rendere gli stessi non più attendibili.

A questo punto si prenda in considerazione il diagramma dei casi d'uso riportato nella figura seguente.

Gli attori previsti dallo use case sono:

- il cliente (l'investitore);
- `Market Data System`, utilizzato per reperire, in tempo reale, i dati relativi alle quotazioni di mercato di determinati prodotti finanziari. Tipicamente si tratta di un sistema interno alla banca alimentato da dati erogati da terze parti;
- `Accounting System`, necessario per reperire la situazione finanziaria (i dati contabili) dell'investitore che richiede di acquistare le varie azioni;
- `Exception Management System`, al quale vengono inviati eventuali messaggi relativi a gravi anomalie occorse. Ogni tipologia di messaggio innesca in tale sistema un opportuno workflow (processo) di gestione;
- `Trade Management System`, al quale vengono affidate, attraverso opportuni messaggi, le transazioni effettuate online.

> Per quanto concerne la descrizione del comportamento dinamico del precedente caso d'uso, il *template* adottato è sensibilmente differente da quello utilizzato in precedenza. In questo caso la parte dedicata allo scenario principale è ripartita in tante sezioni quante sono le entità coinvolte (gli attori più il sistema stesso).
>
> La peculiarità della nuova versione è di evidenziare chiaramente le azioni svolte dalle varie entità e quindi le efferenti responsabilità. Per molti versi ricorda molto gli activity diagram (*cfr* Capitolo 10).
>
> A fronte di questo vantaggio, presenta lo svantaggio di occupare molto più spazio. Spesso, il non riuscire a visualizzare tutto un flusso in una pagina può creare dei problemi di fruizione e quindi comprensione dello stesso.

Nella sezione relativa alle precondizioni del *template* di tab. 4.9 (`Acquisto share on-line`) è riportato il vincolo secondo il quale il cliente, per poter usufruire del servi-

Tabella 4.9

CASO D'USO: UC_BKL_BKS_01	**Acquisto share on line**	**Data:** 03/12/2000 **Versione:** 0.09.003
Descrizione:	Fornisce ai clienti abilitati il servizio on-line di acquisto titoli azionari. L'utente, previa opportuna autorizzazione, fruendo del servizio attraverso un comune browser Internet, è in grado di ricevere informazioni ed eventualmente acquistare prodotti finanziari.	
Durata:	Minuti.	
Priorità:	Elevata.	
Attore primario:	Cliente della banca.	
	Ha interesse a ricevere informazioni relative a particolari prodotti finanziari ed eventualmente ad acquistarli.	
Precondizioni:	▪ Il cliente dispone di almeno un conto corrente presso la banca; ▪ è abilitato alla fruizione del servizio (tramite opportune procedure amministrative bancarie); ▪ è stato autorizzato alla fruizione del servizio dal meccanismo di sicurezza (eseguiti correttamente i servizi di autenticazione e autorizzazione).	
Garanzie minime:	Lo situazione azionaria ed il conto corrente del cliente non vengono modificati : permangono nello stato presente prima dell'esecuzione.	
Garanzie successo:	Il cliente effettua una transazione di acquisto di titoli azionari. Le informazioni relative alle transazioni vengono inviate, per mezzo di opportuno messaggio, al sistema di back office.	
Avvio:	Il cliente richiede esplicitamente l'esecuzione del servizio di acquisto titoli azionari.	

Scenario principale.

	SISTEMA	CLIENTE
1.	Visualizza le specializzazioni in cui si suddividono le azioni acquistabili on-line.	
2.		Seleziona la specializzazione di interesse.
3.	Visualizza l'insieme delle azioni appartenenti alla specializzazione selezionata.	
4.		Seleziona il titolo di interesse.
5.	*Include(Comunica dati share)*	
	Visualizza i dati aggiornati relativi al titolo selezionato.	
6.		Imposta la quantità desiderata.
7.	*Include(Verifica fattore di rischio)*	
8.	*Include(Comunica dati conto correnti cliente).*	
9.	Visualizza dati conto correnti intestati al cliente	
10.		Seleziona il conto corrente da utilizzarsi per la transazione.
11.	*Include(Comunica dati share)*	
12.	Visualizza i dettagli economici ed in particolare l'ultimo prezzo disponibile per il titolo oggetto di acquisto.	
14.	Chiede conferma a procedere con la transazione.	
15.		Accetta la transazione

Tabella 4.10

I scenario alternativo. *Punti: 5 e 11*		
5.1.	Sistema: Il titolo selezionato non è acquistabile (per esempio il titolo è stato sospeso)	
5.2.	Sistema: Comunica apposito messaggio all'utente.	
5.3.	Sistema: Torna ad eseguire dal punto 1.	
II scenario alternativo.		
15.1.	Utente: Non conferma la transazione nei limiti di tempo prestabiliti.	
15.2.	Sistema: Comunica apposito messaggio all'utente.	
15.3.	Sistema: Torna ad eseguire dal punto 11.	
I scenario di errore. *Punti: 2, 4, 6, 10, 15.*		
2.1.	Utente: Non imposta i dati attesi e richiede la terminazione della funzione.	
2.2.	Sistema: Termina lo use case con insuccesso.	
II scenario di errore.		
7.1.	Sistema: Fattore di rischio dell'operazione richiesta superiore ai limiti previsti.	
7.2.	Sistema: Visualizza apposito messaggio all'utente.	
7.3	Sistema: Termina lo use case con insuccesso.	
III scenario di errore.		
5.1.	Sistema: Errore nel reperimento dati share.	
5.2.	Sistema: Comunica momentanea non disponibilità del sistema.	
5.3.	*Punto di estensione: Invio messaggio di errore*	
5.4.	Sistema: Termina l'esecuzione della funzione con insuccesso.	
IV scenario di errore.		
8.1.	Sistema: Errore nel reperimento dati conti correnti utente.	*(Prosegue come dal punto 5.2)*
V scenario di errore.		
11.1.	Sistema: Errore nel reperimento dati share.	*(Prosegue come dal punto 5.2)*
VI scenario di errore.		
16.1.	Sistema: Fallisce invio messaggio dati transazione.	*(Prosegue come dal punto 5.2)*

zio, deve essere precedentemente abilitato alla fruizione. L'ottenimento di tale permesso, tipicamente, richiede che il cliente compili specifici moduli e li consegni ad un apposito "sportello" della banca. Gli impiegati di quest'ultima, dopo aver eseguito gli accertamenti del caso, si incaricano di inserire i dati di tale abilitazione in un apposito sistema, e quindi l'utente viene fornito della coppia (login e password). Ebbene, questo processo è esterno al sistema oggetto di studio e prevede diverse procedure manuali. Si tratta quindi di processi appartenenti al sistema informativo e non a quello informatico che, come riportato nel Capitolo 2, sono esempi di caso d'uso al livello di business.

La lettura del punto 1 nel template di tab. 4.12 dovrebbe far capire l'importanza di produrre il modello ad oggetti del dominio durante le fasi di analisi dei requisiti. Questo, oltre ad agevolare fin dalle prime fasi la produzione di importanti manufatti, come per esempio la specifica dettagliata dei messaggi che i sotto-sistemi devono scambiarsi

(interfaccia), permette di semplificare notevolmente la stessa analisi dei requisiti, la produzione dei primi modelli della GUI, ecc.

L'analisi degli use case presentati in questo paragrafo mette in evidenza come, quantunque il modello dei casi d'uso sia uno strumento imprescindibile per l'analisi dei requisiti utente, da solo non è assolutamente in grado di catturarne tutti gli aspetti. Inoltre, focalizzando l'attenzione esclusivamente sulle funzionalità, non favorisce l'analisi dei requisiti da altre prospettive (quali per esempio quella dell'organizzazione dei dati), che permettono di enfatizzare ulteriori aspetti molto importanti. Per questo motivo è di fondamentale importanza utilizzare completare l'analisi dei requisiti con altri modelli, come per esempio le business rule (presentate nel Capitolo 5), il glossario dei termini, il modello a oggetti del dominio, il modello della GUI, ecc.

Iterazioni nella costruzione del modello dei casi d'uso

Indipendentemente dalla modalità stabilita per descrivere il comportamento dinamico dei casi d'uso è importante considerare che difficilmente un buon modello si ottiene con un'unica iterazione. Nella pratica è necessario effettuarne diverse, per dar modo agli stessi utenti di chiarirsi le idee, per riuscire a considerare situazioni meno frequenti, per raggiungere un livello qualitativo omogeneo ed elevato (tipicamente la qualità dei primi casi d'uso è pessima), ecc. Per questi motivi, è buona norma organizzare la fase di analisi dei requisiti secondo un processo iterativo e incrementale.

> La validità o meno di ricorrere a un processo così articolato anche per la costruzione del modello dei casi d'uso è stato argomento di dibattito tra l'autore e Frank Armour (coautore del testo [BIB13]) nel corso di amichevoli discussioni.

Nel loro libro, Frank Armour e Granville Miller illustrano un processo ben definito per realizzare il modello dei casi d'uso basato su tre macroiterazioni (*cfr* fig. 4.12):

Figura 4.12 — *Schematizzazione del processo di analisi dei requisiti proposto nel testo [BIB13].*

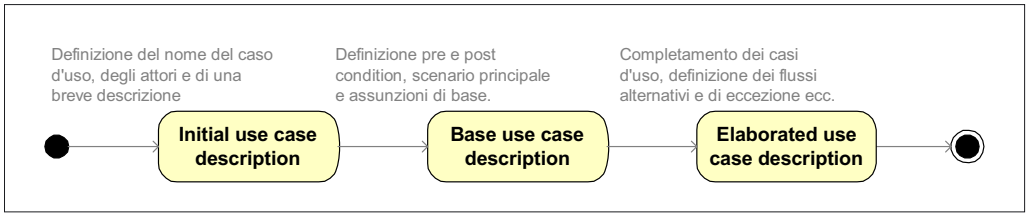


Tabella 4.11

CASO D'USO: UC_BKL_BKS_02	**Comunica dati share**	**Data:** 03/12/2000 **Versione:** 0.01.002
Descrizione:	Permette di reperire i dati in tempo reale di specifici prodotti finanziari richiesti. I dati cui si è interessati sono, essenzialmente, quotazioni di mercato real time, sospensione del titolo, ecc. Per il dettaglio cfr. business rule DatiSh_TmpRl.	
Durata:	Secondo.	
Priorità:	Elevata.	
Attore primario:	Market Data System (MDS).	
	Fornisce le quotazioni di mercato relative ai prodotti specificati.	
Precondizioni:	Sia disponibile il codice del prodotto da reperire.	
Garanzie minime:	?.	
Garanzie successo:	Vengono fornite le informazioni richieste.	

Scenario principale.

	SISTEMA	Market Data System
1.	Reperisce il codice del prodotto del quale si vuole reperire le quotazioni.	
2.	Richiede il servizio fornitura quotazioni di mercato in tempo reale delle azioni.	
3.		Riceve la richiesta di esecuzione del servizio.
4.		Estrae il codice del prodotto del quale si intende reperire la quotazione.
5.		Reperisce i dati della quotazione
6.		Restituisce i dati di quotazione del prodotto reperiti.
7.	Memorizza i dati ricevuti.	

I scenario di errore.	
2.1.	Sistema: Fallisce nell'emettere la richiesta di erogazione del servizio. Restituisce un messaggio di errore.
2.2.	Sistema: Termina l'esecuzione della funzione con insuccesso.
II scenario di errore.	
6.1.	MDS: Fallisce il reperimento dei dati relativi al prodotto specificato (codice non riconosciuto, problemi di connessione, ecc). Restituisce un messaggio di errore.
6.2.	Sistema: Termina l'esecuzione della funzione con insuccesso.
III scenario di errore.	
7.1.	Sistema: Scade il time-out associato alla fruizione del servizio.
7.2.	Sistema: Termina l'esecuzione della funzione con insuccesso.

Tabella 4.12

CASO D'USO: UC_BKL_BKS_03	**Invia messaggio dati transazione**	**Data:** 03/12/2000 **Versione:** 0.01.002
Descrizione:	Invia al sistema di gestione trade un messaggio relativo alla transazione effettuata on-line dall'utente. (N.B. Ogni transazione è considerata effettiva solo a seguito della relativa elaborazione effettuata nel TMS).	
Durata:	Secondi.	
Priorità:	Elevata.	
Attore primario:	TMS.	
	Riceve il messaggio con i dati relativi alla transazione on-line eseguita dall'utente.	
Precondizioni:	L'utente abbia impostato e confermato una transazione, i relativi dati siano disponibili ed il sistema abbia eseguito correttamente tutti i controlli.	
Garanzie minime:	I dati della transazione non subiscono alcuna variazione.	
Garanzie successo:	I dati della transazione vengono organizzati in un opportuno messaggio, e lo stesso viene inviato al TMS.	

Scenario principale.		
	SISTEMA	TMS
	Definizione punto di estensione: *Invio messaggio dati transazione*	
1.	Reperisce i dati della transazione, tra cui: codice cliente; codice prodotto acquistato; quantità e quotazione in tempo reale del prodotto; data e orario della conferma della transazione; *(Cfr business rule: BR_TRADE_DT_TR)*	
2.	Organizza i dati secondo il formato previsto (per esempio in base ad uno schema XML). *(Cfr modello CD_TRADE_DT_TR)*	
3.	Pubblica il messaggio	
4.		Prende il carico il messaggio
I scenario di errore.		
3.1.	Sistema:Fallisce la pubblicazione del messaggio.	
3.2.	Sistema:Termina l'esecuzione della funzione con insuccesso.	

Tabella 4.13

CASO D'USO: UC_BKL_BKS_04	**Reperisce dati conto corrente cliente**	**Data:** 03/12/2000 **Versione:** 0.01.002
Descrizione:	Permette di reperire le informazioni relative ai conti correnti intestati all'utente.	
Durata:	Secondi.	
Priorità:	Elevata.	
Attore primario:	Accounting System (AS).	
	Fornisce i dati relativi ai conti correnti intestati all'utente.	
Precondizioni:	Sia disponibile il codice del cliente e questi disponga di almeno un conto corrente gestito dalla banca.	
Garanzie minime:	La sessione avviata dal cliente rimane valida.	
Garanzie successo:	Vengono reperiti i dati relativi ai conti correnti del cliente compresi i relativi saldi.	

Scenario principale.		
	SISTEMA	ACCOUNTING SYSTEM (AS)
1.	Reperisce il codice dell'utente.	
2.	Richiede il servizio fornitura dati conto corrente utente.	
3.		Riceve la richiesta di esecuzione del servizio.
4.		Estrae il codice del cliente.
5.		Reperisce i dati relativi ai conti correnti corredati dai rispettivi saldi.
6.		Restituisce i dati relativi ai conti correnti del cliente.
7.	Memorizza i dati ricevuti.	
I scenario di errore.		
2.1.	Sistema: Fallisce l'emissione della richiesta di erogazione del servizio.	
2.2.	Sistema: Termina l'esecuzione della funzione con insuccesso.	
II scenario di errore.		
6.1.	AS: Fallisce il reperimento dei dati relativi ai conti correnti intestati all'utente (codice cliente non riconosciuto, problemi di connessione, ecc). Restituisce un messaggio di errore.	
6.2.	Sistema: Termina l'esecuzione della funzione con insuccesso.	
III scenario di errore.		
7.1.	Sistema: Scade il time-out associato alla fruizione del servizio.	
7.2.	Sistema: Termina l'esecuzione della funzione con insuccesso.	

Tabella 4.14

CASO D'USO: UC_BKL_BKS_05	**Invia messaggio di errore**	**Data:** 03/12/2000 **Versione:** 0.01.002
Descrizione:	Invia al sistema di gestione delle eccezioni un messaggio con i dati relativi all'anomalia verificatasi.	
Durata:	Secondi.	
Priorità:	Elevata.	
Attore primario:	Exception Management System (EMS).	
	Riceve il messaggio con i dati relativi all'anomalia verificatasi.	
Precondizioni:	Nel sistema si sia verificata una anomalia grave e i dati identificanti la stessa siano disponibili.	
Garanzie minime:	I dati efferenti restano disponibili nel sistema.	
Garanzie successo:	I dati della anomalia vengono organizzati in un opportuno messaggio, secondo il formato previsto, e lo stesso viene inviato al relativo sistema di gestione.	

Scenario principale.		
	SISTEMA	EMS
	Definizione punte di estensione: Invio messaggio di errore	
1.	Reperisce i dati relativi all'anomalia verificatasi.	
2.	Organizza i dati secondo il formato previsto (per esempio in base ad uno schema XML). *(Cfr modello CD_MSG_ERR)*	
3.	Pubblica il messaggio.	
4.		Prende il carico il messaggio
I scenario di errore.		
3.1.	Sistema:Fallisce la pubblicazione del messaggio.	
3.2.	Sistema:Termina l'esecuzione della funzione con insuccesso.	

1. *initial use case description* (descrizione iniziale dei casi d'uso), denominata così in quanto prevede la definizione, per ogni use case, di pochi basilari elementi, come il nome, gli attori e la descrizione. Si tratta chiaramente di una fase iniziale da avviare congiuntamente all'analisi dei requisiti;

2. *base use case description* (descrizione base dei casi d'uso). La versione precedente viene estesa focalizzando l'attenzione sullo scenario principale, ossia si considera il percorso ideale, trascurando per il momento flussi alternativi e di eccezione;

3. *elaborated use case description* (descrizione elaborata dei casi d'uso). Si tratta dell'ultima fase nella quale vengono aggiunti tutti i restanti dettagli che permettono di terminare (almeno fino alla prossima variazione dei requisiti) la descrizione dei casi d'uso. Quindi vengono inglobate la descrizione e gestione delle situazioni di errore, eventuali flussi alternativi, ecc.

In questo processo, i veri use case (i diagrammi) risultano opzionali, da utilizzarsi per una prospettiva ad alto livello.

Tabella 4.15

CASO D'USO: UC_BKL_BKS_07	**Verifica fattore di rischio**	**Data:** 03/12/2000 **Versione:** 0.01.002
Descrizione:	Verifica che il fattore di rischio associato con la transazione richiesta dal cliente rientri nei limiti previsti dal relativo profilo.	
Durata:	Secondi.	
Priorità:	Elevata.	
Precondizioni:	Nel sistema siano disponibili i dati relativi alla transazione richiesta dall'utente ed il relativo fattore di rischio calcolato.	
Garanzie minime:	I dati relativi al rischio restano disponibili nel sistema.	
Garanzie successo:	Viene eseguita la valutazione del fattore di rischio.	
Scenario principale.		
	SISTEMA	
1.	Reperisce i dati relativi al profilo dell'utente.	
2.	Calcola il valore del rischio connesso con la transazione richiesta dall'utente. (Cfr: business rule: BR_CLC_RISK)	
3.	Verifica la compatibilità del fattore di rischio calcolato con quello impostato nel profilo utente.	
4.	Restituisce la conferma del rispetto del fattore di rischio	
I scenario alternativo.		
4.1.	Sistema:Fattore di rischio della transazione fuori dai limiti di tolleranza.	
4.2.	Sistema:Termina l'esecuzione restituendo esito negativo della verifica.	
Annotazioni.		
3.	Per le specifiche formali relative al calcolo dei fattori di rischio si consulti il documento delle business rule sezione 3.2.	

Il ricorso a un processo di questo tipo sarebbe piuttosto opinabile in sistemi di dimensioni contenute, mentre agevola moltissimo la vita in sistemi di dimensioni medio-grandi. In particolare, permette di dare maggior tempo agli utenti per riflettere sul funzionamento dei casi d'uso ed eventualmente cambiare idea prima che risme di documentazione siano realizzate: non è necessario attendere la completa realizzazione dei caso d'uso, ma è possibile riesaminarli già dalla seconda versione (base). Lo stesso team di management può iniziare in anticipo la pianificazione dell'intero processo con particolare riferimento alle varie iterazioni (con la speranza di mettere numeri non casuali nelle caselline del GAANT). Prima di definire completamente una funzionalità è possibile avere un quadro più chiaro del funzionamento dell'intero sistema. Ciò permette sia di aumentare il livello di consistenza delle varie funzionalità, sia di fornire abbastanza materiale al team degli architetti per avviare la progettazione del sistema.

A fronte di questi vantaggi esistono alcuni inconvenienti. Per esempio dover ritornare più volte su questioni già affrontate con i clienti (generalmente questa cosa tende a infastidirli, soprattutto quando si parla con i manager… Ah, novelli Paganini…), e quindi "sprecare" tempo per fare mente locale, riorganizzare workshop, ricontrollare il materiale, ecc. Poi, sebbene il processo realizzato in modo iterativo sia più flessibile e aiuti ad aumentare il livello qualitativo, è anche vero che apparentemente tende a richiedere più tempo, che non sempre si ha a disposizione.

Dopo lunghe argomentazioni, qualora si utilizzi un approccio basato sui casi d'uso a mo' di ariete da sfondamento (nel capitolo successivo ne viene presentato uno diametralmente opposto basato sugli activity diagram), si è convenuto che un buon approccio potrebbe essere:

1. iniziare il processo realizzando un modello basato quasi esclusivamente sui diagrammi dei casi d'uso con eventualmente una descrizione molto sintetica di due, tre righe (dice nulla questo inizio?). Nell'evenienza in cui si tratti di clienti con difficoltà a comprendere i meccanismi della relazione di estensione, è saggio avvalersi di un approccio pragmatico che consideri l'utilizzo della sola relazione di inclusione;

2. realizzare una prima descrizione che contempli, ovviamente il *main scenario*, gli attori (già evidenziati nei precedenti diagrammi), le precondizioni, le varie garanzie e i flussi alternativi e di eccezioni più semplici ed evidenti. Man mano che si procede con la definizione delle dinamiche del sistema, possono rendersi necessa-

rie attività di revisione dei diagrammi dei casi d'uso: mantenere i modelli consistenti è un'attività di fondamentale importanza;

3. concludere il tutto incorporando suggerimenti provenienti dal team degli architetti, dalla revisione con i clienti, ecc.

Dopo la fase iniziale, qualora si verifichi la necessità, è necessario ristrutturare i diagrammi dei casi d'uso.

Nel caso in cui si decida di ricorrere a processi iterativi e incrementali anche per l'analisi dei requisiti, è utile prevedere una serie di informazioni supplementari, da associare ai vari manufatti prodotti, necessarie per tener traccia della relativa evoluzione. Queste informazioni dovrebbero riportare le varie versioni dei manufatti (in questo caso use case), gli autori, la data di prevista o avvenuta consegna, la descrizione e così via. Chiaramente non c'è alcuna necessità di specificarle per ogni singolo caso d'uso (singola ellisse), ma è sufficiente riportarle solo per le macrofunzionalità (in altre parole al livello di use case diagram).

Con riferimento all'esempio del sistema di banking online, sarebbe utile visualizzare l'evoluzione dello use case diagram *acquisto share on-line*, mentre sarebbero di scarso interesse per tutti i singoli casi d'uso coinvolti *verifica fattore di rischio*, *Invia messaggio di errore*, e così via. Ovviamente le informazioni specificate per lo use case generale si riferiscono anche a tutti i casi d'uso di cui è composto.

Come suggerito anche dal processo della *Rational Software Corporation* (RUP Rational Unified Process) è consigliabile riportare, per ogni use case diagram, una tabella 4.come la seguente:

Probabilmente potrebbe risultare altrettanto conveniente disporre di un'altra tabella, in qualche modo speculare alla prima, collocata alla fine della descrizione del comportamento dinamico di ogni use case diagram. Lo scopo è di ospitare, in un sol punto, eventuali considerazioni del personale coinvolto nello sviluppo e nel controllo della qualità del modello.

Sebbene sia possibile specificare eventuali considerazioni anche nel modulo utilizzato per descrivere il comportamento dinamico di ogni singolo caso d'uso, probabilmente è comodo disporre anche di uno più generale. Ciò consente di raggruppare le varie note in un solo punto evitando di doverle cercare per tutto il progetto. Un altro vantaggio consiste nella semplificazione della gestione dello stato di avanzamento delle osservazioni. Si supponga che durante il processo di revisione dei casi d'uso nascano delle perplessità o suggerimenti: le relative descrizioni andrebbero riportate nella tabella 4.conclusiva di ogni use case diagram. Pertanto è possibile verificare se ci siano o meno quesiti in attesa di risposta, accedendo direttamente a queste tabelle.

Tabella 4.16

n. Iterazione	Autore	Consegna	Descrizione
1A	LVT	10/II/2001	Initial draft – Diagramma e descrizione
1A	RV	12/II/2001	Base – Definizione del main scenario
1B	LVT	20/II/2001	Elaborated – Inclusione degli scenari alternativi e di errore.
1B	AR	22/II/2001	Revisione – Aggiunta feedback utente

Tabella 4.17

n. Iterazione	Autore	Consegna	Descrizione
1°	LVT	10/II/2001	Initial draft

La sicurezza nel modello dei casi d'uso

Prima di procedere con l'esempio conclusivo del capitolo, si è ritenuto utile esaminare l'argomento relativo all'incorporazione della sicurezza nel modello dei casi d'uso. In primo luogo va ricordato che si tratta di requisiti non funzionali che, come tali, costituiscono una parte, anche molto importante, dell'apposito manufatto (presentato al capitolo successivo). Inoltre si tratta di un vincolo che, tipicamente, si applica a tutti i sistemi e quindi alla stragrande maggioranza di casi d'uso.

Ora si potrebbe utilizzare l'approccio più tradizionale, ossia includere i relativi casi d'uso in tutti gli use case soggetti ai vincoli di sicurezza. Esempi di casi d'uso contenenti politiche di sicurezza potrebbero essere *autenticazione utente* (serve per accertarsi che l'utente sia effettivamente quello che dichiara di essere), *autorizzazione a fruire i servizi selezionati*, *autorizzazione a trattare i dati richiesti*, *scrittura di appositi log* per tener traccia della navigazione dell'utente, ecc. Per ciò che concerne l'autenticazione, è da notare che il problema si pone anche per lo scambio di messaggio con altri sistemi.

Per quanto attiene alle autorizzazioni è evidente che se un utente viene autorizzato a espletare un servizio (per esempio *Visualizzazione estratto conto*) ciò non significa che lo

potrà richiedere per qualsiasi insieme di dati (per esempio estratto conto di un'altra persona). Questo approccio, sebbene molto semplice, pone una serie di svantaggi:

- altera la logica business mostrata dai vari diagrammi dei casi d'uso. Includendo in un apposito use case quelli relativi all'autenticazione e/o autorizzazione, le relazioni con gli altri inclusi di carattere business tendono a diventare automaticamente degli extend: vengono eseguiti solo se si ottiene l'autorizzazione;

- la descrizione e gli stessi diagrammi dei casi d'uso viene inutilmente complicata con dettagli già conosciuti al livello dell'intero sistema.

Il consiglio pertanto consiste nel ricorrere alla seguente tecnica:

1. riportare gli use case della sicurezza "isolati" dagli altri. Ciò è molto utile per specificare particolari comportamenti/requisiti. Per esempio, dopo che l'utente è stato riconosciuto, si potrebbe verificare se il suo profilo sia stato bloccato, oppure se la relativa password sia scaduta, ecc.
2. redigere la sezione del manufatto dei requisiti non funzionali (o eventualmente un apposito documento) descrivendo dettagliatamente anche il comportamento da applicare ai diversi casi d'uso.
3. eventualmente nelle precondizioni dei vari use case citare le clausole standard: "l'utente è stato riconosciuto (autenticato) e autorizzato a eseguire il servizio" (da notare che la seconda clausola implica la prima) e nella descrizione del comportamento dinamico, far riferimento alla specifiche sezioni menzionate nel punto precedente.

Per esempio in un sistema Internet/Intranet di un'organizzazione potrebbe essere presente il vincolo che gli utenti possano accedere unicamente ai dati relativi al proprio dipartimento. Questa restrizione si potrebbe suddividere in due sezioni:

- reperimento: i dati da mostrare all'utente devono essere filtrati in base al relativo dipartimento di appartenenza;

- inserimento: tutti i dati devono riportare l'informazione sul dipartimento di appartenenza.

Nel documento NFR potrebbe essere opportuno riportare queste due sezioni (ovviamente con maggiori dettagli) e quindi riferirle nei casi d'uso in ogni punto in cui si effettui una ricerca o si aggiorni il database.

e-commerce: un esempio

In questo paragrafo viene presentata un'ampia sezione del modello dei casi d'uso di un sistema del tutto originale: sito e-commerce. Visto che è stato citato abbondantemente negli esempi, si è deciso di presentarlo finalmente in maniera più organica.

> In questo paragrafo è fornita una versione iniziale, uno scheletro facilmente espandibile per la descrizione di scenari reali (la realtà è sempre più complessa di quanto riportato nei libri). Per esempio non sono presi in considerazione servizi di supervisione, di risoluzioni anomalie, statistiche, ecc.

Tipicamente, sistemi di questo tipo possono essere scomposti *almeno* in tre sottosistemi: il sito Internet ovviamente, il legacy system, e il layer di interfacciamento (wrapper) necessario per consentire ai due precedenti sistemi di interagire.

> Molto raramente accade che un'azienda decida di "lanciarsi" nel commercio elettronico partendo da zero, ossia avendo contestualmente la necessità di allestire il sito Internet e la propria infrastruttura informatica gestionale. Queste rarissime occasioni costituiscono il sogno — sì, proprio di sogno si deve parlare — dei tecnici in quanto consentono di realizzare il tutto attraverso tecnologie omogenee e moderne — ossia permettono di far giocare i tecnici con nuovi giocattoli — e soprattutto evitano tanti problemi di connessione che si generano tra sistemi costruiti con tecnologie eterogenee.
>
> Spesso i sistemi gestionali presenti presso i clienti (legacy system) risultano privi di meccanismi di interfacciamento con altri sistemi automatizzati, oppure risultano dotati di incomprensibili modalità di funzionamento con le quali, ahimè, è necessario interagire. Per esempio può capitare il caso di utilizzare sistemi che prevedano la quotazione degli articoli inseriti in un ordine solo dopo la conferma dell'ordine stesso. Come dire prima si effettua l'ordine e poi si viene a conoscenza del relativo importo.
>
> Il caso decisamente più frequente è rappresentato da aziende di dimensioni medio-grandi, già dotate di ben collaudati sistemi di gestione dei propri processi interni (si tratta, tipicamente, di sistemi funzionanti su architetture hardware del tipo Mainframe, IBM AS400, IBM AIX, HP UX, Bull, ecc.), che decidono di sfruttare i vantaggi offerti dal commercio elettronico.
>
> In tutti questi casi è necessario realizzare, oltre al sito Internet, anche il famoso sistema wrapper in grado di realizzare la comunicazione con il legacy system. Chiaramente è lungi dalla volontà delle organizzazioni introdurre nuovi sistemi con la necessità di replicare i processi manuali di manutenzione dati. Per esempio è ovvio che si cerchi di evitare l'inserimento dei dati efferenti nuovi prodotti dapprima nel legacy system e poi nel sistema per il commercio elettronico, così come non si desidera reinserire manualmente nel sistema di legacy ordini effettuati on-line. Si esige chiaramente che i dati presenti in un sistema siano replicati automaticamente negli altri.
>
> I sistemi di wrapper offrono un insieme di servizi prestabilito, ottenibili essenzialmente da quelli disponibili nel legacy system, presentandoli al mondo esterno attraverso opportune

"interfacce" sfruttanti tecnologie moderne (CORBA, RMI, server socket, ecc.). Si tratta di un layer proxy in grado di offrire una percezione diversa e decisamente più moderna di un sistema di legacy. In altre parole sono sistemi paragonabili ad alcuni politici del Belpaese: facce nuove che celano vecchie logiche…

Nel caso presentato si è considerata unicamente l'ipotesi semplificata di sistemi che necessitino solo di uno scambio asincrono di informazioni. Nella realtà, spesso si verifica contestualmente la necessità di uno scambio sincrono e asincrono di informazioni tra sistemi. Si supponga per esempio di dover realizzare il sito Internet del legacy system in grado di quotare gli ordini solo dopo averne ricevuto conferma. In tal caso il sistema Internet dovrebbe inviare i dati dell'ordine e quindi attendere la risposta dell'elaborazione avvenuta nel legacy system al fine di visualizzare all'utente l'importo dell'ordine effettuato (Sembra assurdo vero? Non si sente la vocina interna dell'esperienza sbraitare?).

Prima di procedere con l'illustrazione dei vari diagrammi, potrebbe essere utile riportare l'elenco di tutti gli attori previsti con le relative descrizioni, come illustrato nel capitolo precedente (*modello per la documentazione degli attori*). Si tratta di un semplice esercizio demandato ai lettori. Da notare che nell'esempio riportato sono presi in considerazione esclusivamente alcuni servizi forniti al clienti, altri peculiari per diverse tipologie di attori (come per esempio amministratori) sono state volutamente omesse.

Non è infrequente anche l'illustrazione di un primo use case diagram con i vari macroservizi offerti dal sistema. L'obiettivo è appunto mostrare le principali funzioni, senza però fornirne alcun dettaglio: una sorta di indice grafico. Volendo è possibile para-

Figura 4.13 — *Schema di un sistema di wrapping.*

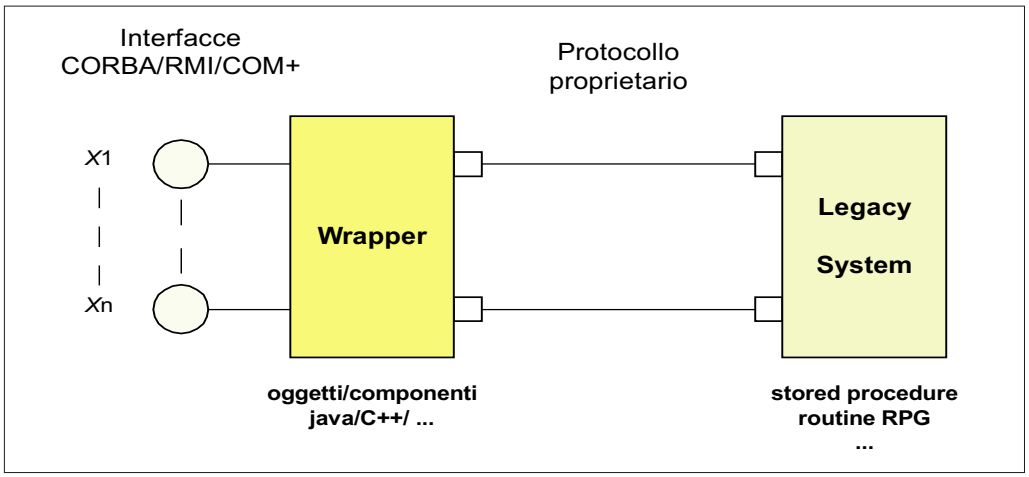
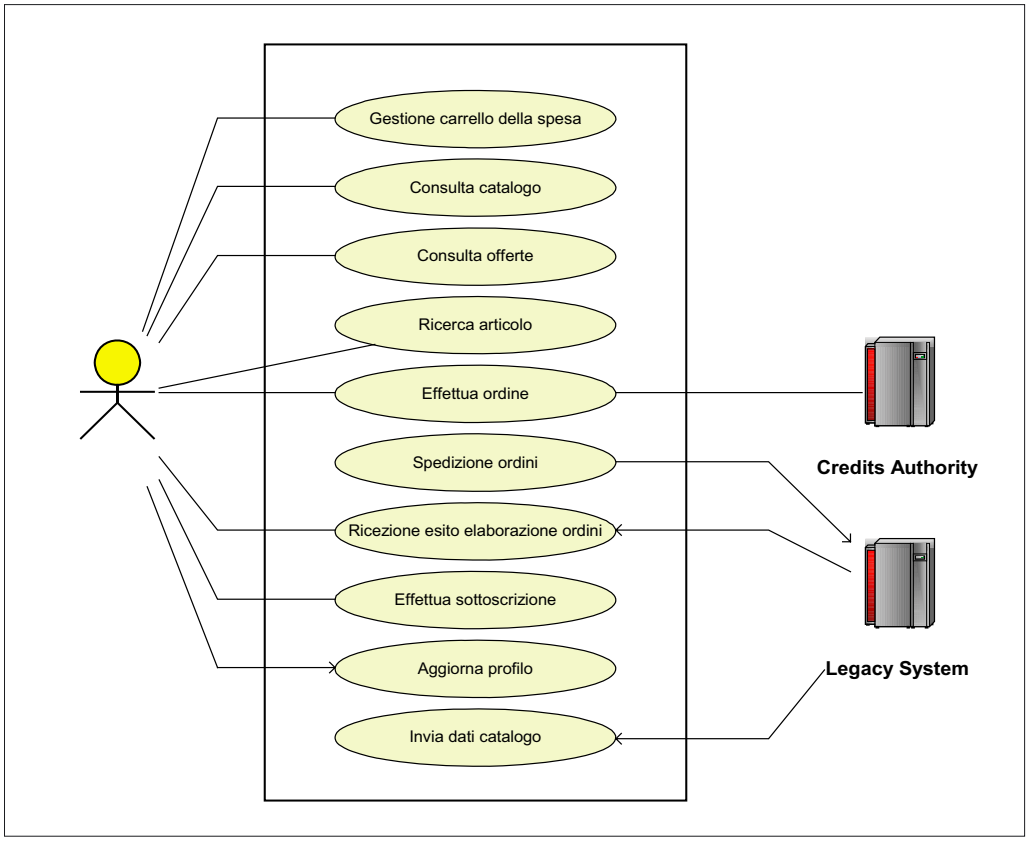


Figura 4.14 — *Macro funzioni del sistema oggetto di studio.*



gonarlo al diagramma Top Level previsto dalla metodologia DFD. Ovviamente si tratta di un solo livello di decomposizione: guai altrimenti; si rischierebbe di cadere nella trappola della decomposizione funzionale.

Nel sistema oggetto di studio la relativa struttura dovrebbe risultare simile a quella riportata nella fig. 4.14.

Si cominci con il considerare le prime due funzionalità basilari fornite dal sito Internet: registrazione di un nuovo utente (*effettua sottoscrizione*) e aggiornamento del relativo profilo (*aggiorna profilo*).

Un primo elemento di riflessione potrebbe essere relativo alla necessità di dettagliare nei casi d'uso i campi appartenenti al profilo utente. Ebbene nella pratica, purtroppo, risulta fondamentale stabilire anche dettagli di questo tipo. Questi dovrebbero essere definiti nel modello del business attraverso opportuno diagramma delle classi e nel mo-

dello della GUI. Eventualmente, queste informazioni andrebbero inserite nel template illustrato nel capitolo successivo (*Modello per le interfacce*), nel quale è possibile specificare quali dati sono obbligatori e quali no.

Riportare questi dettagli comunque permette di evitare continue richieste di variazione del disegno della pagina HTML dipendenti dal personaggio dell'organizzazione del cliente che la analizzi.

Tabella 4.18

n.	Osservazione	Autore	Stato
1	Come comportarsi nel caso x,y,z	R.V.	Risolto. Soluzione incorporata nel punto a.

Figura 4.15 — *Servizi di sottoscrizione e aggiornamento profilo utente.*

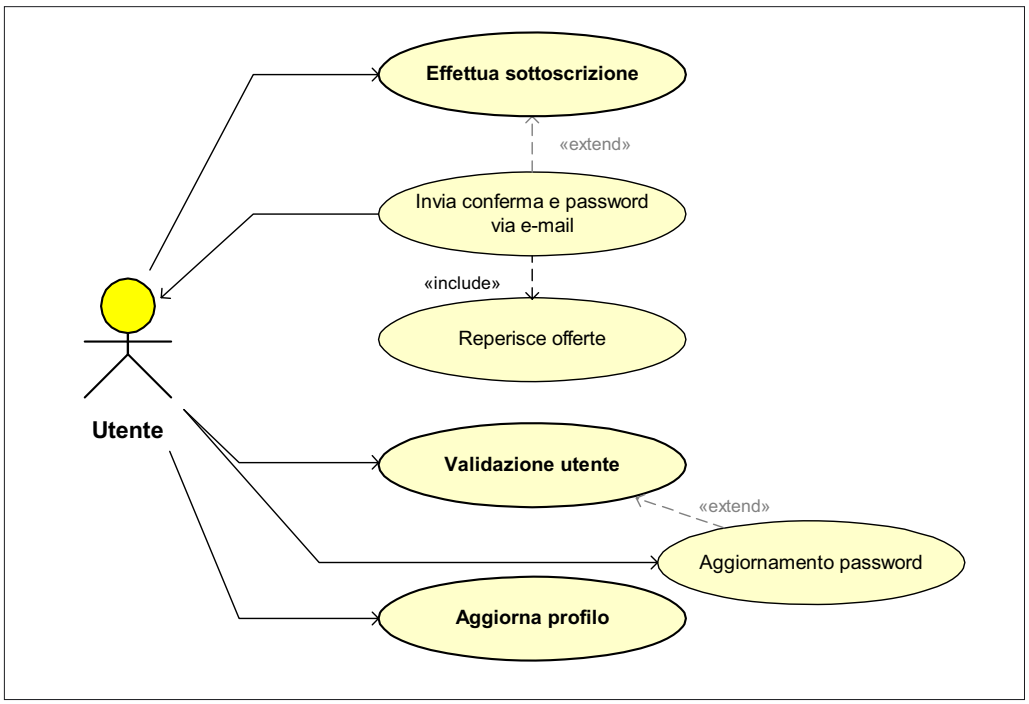


Tabella 4.19

CASO D'USO: UC_ECMMRC01	**Effettua sottoscrizione.**	**Data:** 01/II/2001 **Versione:** 0.00.002
Descrizione:	Consente agli utenti di registrarsi al servizio al fine di poter ottenere l'abilitazione ad effettuare ordini di acquisto on-line presso il sito.	
Durata:	Minuti.	
Priorità:	Elevata.	
Attore primario:	Utente.	
	Ha interesse di registrarsi presso il sito per poter fruire dei servizi offerti dal sistema.	
Precondizioni:	L'utente dispone di una connessione Internet.	
Garanzie di fallimento:	L'utente non viene registrato. Il sistema visualizza un appropriato messaggio di errore.	
Garanzie di successo:	L'utente viene registrato correttamente ed il sistema provvede ad inviargli l'e-mail di conferma sottoscrizione.	
Avvio:	L'utente richiede esplicitamente di registrarsi.	

Scenario principale.		
	UTENTE	SISTEMA
1.	Esegue esplicitamente la funzione di registrazione presso il sito.	
2.		Visualizza gli accordi commerciali (il contratto) e chiede conferma a procedere. (Cfr. Business Rule: BR_SOTT_ACC_CMM1)
3.	Conferma la propria volontà a procedere.	
4.		Visualizza la pagina con la richiesta dati utente (profilo). (Cfr. GUI: GUI_DT_UT_STT2.1)
5.	Imposta i proprio dati e quindi conferma.	
6.		Verifica i dati dell'utente.
7.		Verifica che l'utente non sia presente nella black-list.
8.		Genera password iniziale secondo specifico algoritmo. (Cfr. Business Rule: BR_GEN_PASS)
9.		*Punto di estensione:* *Invia conferma e password via e-mail* *Condizione* *L'utente confermi la registrazione*
10.		Memorizza permanentemente i dati utente.
11.		Visualizza messaggio di avvenuta registrazione.
Scenario alternativo: I dati impostati dall'utente non risultano corretti.		
6.1.	Sistema:Visualizza un messaggio indicante i dati ritenuti non corretti.	
6.2.	Sistema:Riprende dal punto 4.	
Scenario di errore: L'utente richiede di terminare anticipamene il servizio. Punti: 3, 5.		
3.1.	Sistema:Visualizza apposito messaggio.	
3.2.	Sistema:Termina l'esecuzione dello use case con insuccesso.	
Scenario di errore: Il nominativo utente è presente nella black-list		
8.1	Sistema:Comunica rifiuto iscrizione dell'utente.	
8.2.	Sistema:Termina l'esecuzione dello use case con insuccesso.	
Annotazioni.		
	I dati relativi la registrazione di un utente restano confinati nel database del sito fino all'effettuazione del primo ordine. Nel relativo invio al sistema di legacy, i dati dell'utenti vengono aggiunti a quelli dell'ordine effettuato. Solo a quel punto l'utente verrà memorizzato anche nel sistema di back office operante presso l'organizzazione e quindi assume, a tutti gli effetti, lo status di cliente.	

Poiché però è comunque diritto del cliente variare le specifiche e quindi anche i campi da visualizzare, nell'eventualità che ciò accada è possibile dimostrare, prove alla mano, che si tratta di un vero e proprio change request e come tale implica specifiche conseguenze come per esempio la dilatazione dei tempi di consegna.

Un'altra riflessione potrebbe scaturire dall'analisi del comportamento del sistema in caso di gravi anomalie quali per esempio impossibilità di inviare e-mail o di memorizzare i dati del nuovo cliente. Nella pratica una buona tecnica consiste nel definire in un apposito documento la politica di gestione delle gravi anomalie tecniche (*system exception*) e di evitarne la citazione in ogni caso d'uso. Ciò a meno di requisiti specifici per la gestione di particolari anomalie. Comunque, la gestione di malfunzionamenti dovuti a problemi tecnici dovrebbe prevedere l'aggiornamento di un apposito file di log e l'invio di messaggi a un operatore umano (quale per esempio webmaster) al fine di avvisarlo dell'anomalia.

La considerazione finale è relativa alla comunicazione descritta tra l'utente e il sistema. Come si può facilmente notare sia nel precedente diagramma, sia in quelli mostrati di seguito, si tratta di scambi di messaggi interattivi e prolungati nel tempo.

In base alle direttive formali dello UML le comunicazioni tra attori e sistema dovrebbero essere modellate, quanto più possibile, in modo asincrono. Logica conseguenza sarebbe che ogni diversa tipologia di messaggio originata da un attore dovrebbe avviare uno specifico caso d'uso. In questo contesto si è deciso di non ricorrere a tale approccio perché avrebbe finito per complicare notevolmente la fruizione dei requisiti senza apportare alcun reale vantaggio.

Considerando la descrizione nella tab. 4.19 e volendo aderire il più possibile alle direttive formali, sarebbe stato necessario riportare un caso d'uso atto ad inviare all'utente le informazioni relative gli accordi commerciali (punto 2), un altro per ricevere il relativo feedback (punto 3), un altro ancora per la richiesta dei dati profilo utente (punto 4), un altro per l'acquisizione dei relativi dati (punto 6) e così via. Come al solito il premio per il diagramma più formale non è ancora stato istituito, mentre lo stress generato da problemi di comunicazione con il cliente è più che reale.

Questo use case appartiene alla categoria di quelli che, dal punto di vista del comportamento dinamico, non dovrebbero assolutamente essere modellati in maniera indipendente: sarebbe sufficiente citare un paio di righe nella descrizione del caso d'uso invocante. Durante la realizzazione del modello dei casi d'uso, capita frequentemente che, dettagliando il comportamento dinamico di un caso d'uso, ci si accorga che non abbia più senso presentarlo in maniera autonoma.

Tabella 4.20

CASO D'USO: UC_ECMMRC02	**Invia conferma e password via e-mail**	**Data:** 01/II/2001 **Versione:** 0.00.001
Descrizione:	Invia all'utente la conferma di avvenuta registrazione corredata da: dati personali dello stesso, password iniziale ed eventuali offerte commerciali (i consigli per gli acquisti) presenti nel sito stesso.	
Priorità:	Media.	
Durata:	Secondi.	
Attore primario:	Utente.	
	Riceve l'e-mail di avvenuta registrazione.	
Precondizioni:	L'utente ha eseguito la funzione di registrazione nel sito, ha impostato i propri dati ed il sistema ha generato una password temporanea.	
Garanzie di fallimento:	L'e-mail non viene inviata e la registrazione dell'utente non viene conclusa.	
Garanzie di successo:	L'e-mail viene correttamente spedita all'utente.	
Scenario principale.		
	SISTEMA	
	Definizione punto di estensione: *Invia conferma e password via e-mail.*	
1.	Acquisisce i dati relativi al nuovo cliente compresa la password temporanea.	
2.	Formatta opportunamente il messaggio. (Cfr: GUI GUI_MSG_NV_CLN)	
3.	*Include(Reperisci offerte).*	
4.	Struttura opportunamente le eventualiinformazioni reperite (Cfr: GUI GUI_MSG_OFF_CM)	
5.	Ottiene l'indirizzo dell'utente.	
6	Invia l'e-mail.	

Tabella 4.21

CASO D'USO: UC_ECMMRC03	**Reperisce offerte**	**Data:** 01/II/2001 **Versione:** 0.00.001
Descrizione:	Reperisce l'eventuale elenco delle offerte commerciali disponibili nel sito.	
Durata:	Secondo.	
Priorità:	Bassa.	
Precondizioni:		
Garanzie di fallimento:	Nessuna offerte viene reperita.	
Garanzie di successo:	Vengono reperite le offerte commerciali disponibili nel sito.	
Scenario principale.		
	SISTEMA	
1.	Reperisce le offerte valide presenti nel sistema. (Cfr: Modello ad oggetti del dominio, diagramma offerte)	
Scenario alternativo: Nel sistema non sono presenti offerte valide.		
1.1.	Sistema:Termina l'esecuzione senza caricare alcuna offerta.	

Nel dubbio se evidenziare o meno alcuni casi d'uso, bisogna prestare attenzione a non commettere l'errore di pensare in termini implementativi. Ciò che dal punto di vista tecnico potrebbe essere poco rilevante, perché magari richiederebbe solo alcune linee di codice, dal punto di vista del cliente potrebbe avere un'importanza elevata. Questo è il caso dello use case in questione. Il cliente è interessato a evidenziare che l'utente, all'atto della registrazione, venga informato di eventuali offerte promozionali. Bisogna sempre tendere al giusto compromesso. Se da un lato mostrare l'organizzazione di alcune funzioni nei diagrammi dei casi d'uso ne può facilitare la comprensione, allo stesso tempo, eccedendo nel dettaglio si corre unicamente il rischio di generare diagrammi eccessivamente complicati e di non facile comprensione. Ancora più grave, si corre il rischio di iniziare a disegnare il sistema (per decomposizione funzionale) con gli strumenti errati e senza tener presenti aspetti molto importanti non ancora analizzati/disponibili.

Dall'esame della descrizione dello use case è possibile constatare che non si menziona mai esplicitamente il numero massimo di fallimenti consecutivi consentiti all'utente. Si tratta di un buon espediente sia per non vincolare la funzione stessa, sia per limitare le correzioni da svolgere in caso di variazioni. Il dettaglio di queste informazioni andrebbe riportato nel documento delle regole di business.

Come si può notare, molte parti importanti dei casi d'uso fanno riferimento a opportune regole di business. Come si vedrà nel prossimo capitolo, si tratta di un'ottima tecnica per centralizzare le regole del business, per evitare ripetizioni e disallineamenti delle stesse, ecc. Il problema però è che si rischia di spogliare i casi d'uso della loro importanza: si rischia, paradossalmente, di non trovarvi più la descrizione dettagliata del servizio da implementare. Pertanto la tecnica ottimale consistere nel continuare a mantenere un unico repository per tutte le business rule, nel continuare ad inserire nei casi d'uso opportuni hyperlink alle sezioni delle business rule desiderate, ma anche utilizzare degli strumenti che permettano di stampare la descrizione dei casi d'uso includendo le regole del business referenziate.

Nel caso d'uso riportato nel template di tab. 4.23, la sezione `BR_VAL_PASW` delle business rule potrebbe prevedere le seguenti verifiche: la password originaria digitata sia quella dell'utente, le due ripetizioni della nuova password coincidano, la nuova password sia più lunga di 7 caratteri, la nuova password contenga almeno un numero ed un carattere speciale, la nuova password non sia una delle recenti 7 utilizzate dall'utente, ecc.

Tabella 4.22

CASO D'USO: UC_ECMMRC04	**Validazione utente**	**Data:** 01/II/2001 **Versione:** 0.00.001
Descrizione:	Si occupa di eseguire le funzioni necessarie per l'autenticazione dell'utente connesso.	
Durata:	Secondi.	
Priorità:	Elevata.	
Precondizioni:		
Garanzie di fallimento:	L'utente non è riconosciuto dal sistema.	
Garanzie di successo:	L'utente viene riconosciuto dal sistema.	
Avvio:	L'utente esegue la procedura di log-in.	
Scenario principale.		
	UTENTE	SISTEMA
1.		Visualizza il modulo autenticazione utente.
2.	Imposta login e password.	
3.		Verifica che la coppia di valori (login, password) individui un cliente nel database del sistema.
4.		Verifica che l'utente non sia stato bloccato.
5.		Verifica che la password dell'utente non sia scaduta e che non sia una temporanea. (Cfrç BR_CICLO_VT_PSW)
6.		Reperisce il profilo dell'utente.
7.		Trasferisce in memoria i dati del cliente.
Scenario alternativo: Verifica login e password fallita e numero di tentativi consecutivi falliti minore del massimo consentito (Cfr: BR_SIC_MAX_TNT).		
3.1.	Registra nel file di log l'evento.	
3.2.	Visualizza opportuno messaggio.	
3.3.	Aumenta il contatore di tentativi di autenticazione falliti.	
3.4.	Riprende dal punto 1.	
Scenario alternativo: Utente autorizzato e password scaduta o temporanea.		
5.1.	Registra nel file di log l'evento.	
5.2.	*Punto di estensione:* *Aggiorna password*	
Scenario di errore: L'utente decide di non impostare login e password.		
2.1.	Termina l'esecuzione dello use case con insuccesso.	
Scenario di errore: Login fallito e numero massimo di tentativi falliti previsto raggiunto (Cfr: BR_SIC_MAX_TNT).		
3.1.	Registra nel file di log l'evento.	
3.2.	Visualizza opportuno messaggio.	
3.3.	Blocca l'utente (stato utente = BLOCCATO, Cfr: BR_CICLO_VT_UT)	
3.4.	Termina l'esecuzione del caso d'uso con insuccesso.	
Scenario di errore: Utente bloccato.		
4.1.	Registra nel file di log l'evento.	
4.2.	Visualizza opportuno messaggio.	
4.3.	Sistema:Termina l'esecuzione dello use case con insuccesso.	
Annotazioni.		
5.	Le password temporanee, come mostrato nel diagramma degli stati relativi al ciclo di vita delle password (Cfrç BR_CICLO_VT_PSW) sono quelle generate dal sistema (primissima password fornita all'utente, nuova password richiesta dall'utente, ecc.)	

Dall'analisi della descrizione del comportamento dinamico del presente caso d'uso emerge chiaramente che esiste una sezione comune con lo use case di registrazione utente. Ciò potrebbe spingere a cercare di strutturare ulteriormente i casi d'uso per mezzo di relazioni di generalizzazione. Benché possibile, probabilmente, in questo caso non è opportuno. Si ricordi che l'obiettivo di questa fase è catturare i requisiti del cliente, capire cosa dovrà fare il sistema e non disegnare lo stesso. Eventuali mancanze di astrazione presenti in questa fase non incidono assolutamente sulle restanti e tantomeno sul disegno del sistema.

Tabella 4.23

CASO D'USO: UC_ECMMRC05	**Aggiornamento password**	**Data:** 01/II/2001 **Versione:** 0.00.001
Descrizione:	Permette all'utente di cambiare la propria parola chiave	
Durata:	Secondi.	
Priorità:	Media.	
Precondizioni:	L'utente sia stato precedentemente autenticato.	
Garanzie di fallimento:	La password originaria dell'utente non viene modificata.	
Garanzie di successo:	L'utente aggiorna la propria password.	
Avvio:	L'utente esegue la procedura di cambiamento password, o il sistema forza l'utente ad aggiornare la password.	
Scenario principale.		
	UTENTE	SISTEMA
2.		Visualizza il modulo aggiornamento password.
3.	Imposta la password originaria e, due volte, la nuova password.	
4.		Verifica che i dati impostati siano corretti (Cfr: BR_VAL_PASW)
5.		Memorizza la nuova password.
6.		Aggiorna lo stato ed il periodo di validità della nuova password. (Cfr: BR_PSW_EXP)
7.		Termina il caso d'uso con successo.
Scenario alternativo: Verifica dati impostati fallita.		
4.1.	Registra nel file di log l'evento.	
4.2.	Visualizza opportuno messaggio.	
4.4.	Riprende dal punto 1.	
Scenario alternativo: Utente decide di terminare l'aggiornamento della password.		
3.1.	Registra nel file di log l'evento.	
3.2.	Termina il caso d'uso con insuccesso	

Tabella 4.24

CASO D'USO: UC_ECMMRC06	**Aggiorna profilo**	**Data:** 01/II/2001 **Versione:** 0.00.001
Descrizione:	Permette ad un utente precedentemente registrato di modificare i dati efferenti il proprio profilo.	
Durata:	Minuti.	
Priorità:	Bassa.	
Attore primario:	Utente. Ha interesse ad aggiornare i dati efferenti il proprio profilo.	
Precondizioni:	L'utente sia registrato nel sistema (esista il relativo profilo) e sia stato autorizzato ad eseguire il servizio.	
Garanzie di fallimento:	Il profilo dell'utente non viene aggiornato.	
Garanzie di successo:	Le modifiche apportate dall'utente al proprio profilo vengono memorizzate permanentemente nel sistema.	
Avvio:	L'utente richiede di eseguire la funzione di aggiornamento del proprio profilo.	

Scenario principale.		
	UTENTE	SISTEMA
1.	L'utente richiede esplicitamente di eseguire la funzione di aggiornamento profilo.	
2.		Reperisce i dati dettagliati dell'utente. (Cfr: Modello casi d'uso, dati utente)
3.		Visualizzazione della pagina profilo utente con i relativi dati impostati (Cfr: GUI_AGG_PRF_UT)
4.	Modifica i dati del proprio profilo e conferma.	
5.		Verifica i dati impostati dall'utente. (Cfr: BR_VAL_DT_PRF)
6.		Memorizza gli aggiornamenti.
Scenario alternativo: Dati impostati non corretti.		
5.1.	Sistema:Visualizza un messaggio con indicati i dati ritenuti non corretti.	
5.2.	Sistema:Riprende l'esecuzione dal punto 3.	
Scenario di errore: Utente annulla la funzione di aggiornamento del proprio profilo.		
4.1.	Sistema:Riprende l'esecuzione dal punto 3.	

Tabella 4.25

n.	Osservazione	Autore	Stato
1	Confermare che qualora la password dell'utente sia scaduta è sufficiente forzare l'utente a cambiarla.	RV	Pending

Compresa la funzione delle tabelle come la precedente, per motivi pratici, non è più indispensabile riportarle per i casi d'uso successivi.

Da notare che, sebbene non specificato esplicitamente, la modalità con cui i casi d'uso sono raggruppati e presentati, ne riflette l'organizzazione in package.

Un altro insieme di use case presi in esame è quello relativo al reperimento delle informazioni relative agli articoli venduti nel sito per il commercio elettronico.

Come evidenziato dal diagramma riportato nella descrizione dei casi d'uso, le funzioni di consultazione sono ritenute non sensibili per il sistema e quindi fruibili da tutti gli utenti, anche da quelli non registrati. Chiaramente, qualora l'utente manifesti interesse per uno specifico articolo e desideri aggiungerlo nel proprio carrello della spesa, si entrerebbe nell'insieme delle funzionalità ad accesso limitato ai soli utenti registrati. Questo passo si renderebbe necessario anche per reperire la specifica istanza (quella appartenente al cliente) del carrello della spesa in cui inserire l'articolo. Si tratta ovviamente di regole richieste dal committente del sito, ma nulla vieta di averne diverse.

Nel diagramma di fig. 4.16, il caso d'uso `Visualizza dettagli articolo` è stato collegato ai casi d'uso `Consulta catalogo`, `Consulta offerte` e `Ricerca articolo` per mezzo della relazione di estensione. Quantunque ciò sia perfettamente legittimo, in molti casi simili al precedente, cioè in cui si ha un caso d'uso con funzionalità ben definite ed atomiche, utilizzato da diversi altri casi d'uso, si preferisce ricorrere alla relazione di inclusione. Ciò semplifica la fruizione, i diagrammi, e la descrizione del loro comportamento dinamico.

Figura 4.16 — *Use case per il reperimento delle informazioni.*

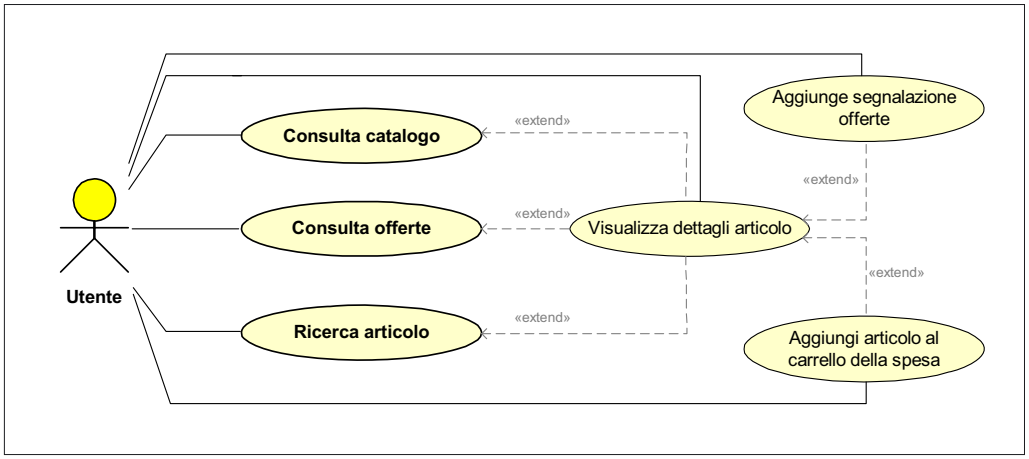


Tabella 4.26

CASO D'USO: UC_ECMMRC11	**Consulta catalogo**	**Data:** 05/II/2001 **Versione:** 0.00.001
Descrizione:	Permette ad un utente di consultare i prodotti venduti nel sito con modalità sia sequenziale, simile a quella con cui si sfogliano i cataloghi cartacei, sia ipertestuale (navigazione basata sui link presenti nelle varie pagine HTML).	
Durata:	Minuti.	
Priorità:	Elevata.	
Attore primario:	Utente.	
	Ha interesse nel ricevere informazioni relative al catalogo degli articoli.	
Precondizioni:	Disponibilità del catalogo degli articoli.	
Garanzie di fallimento:	Nessun dato viene alterato.	
Garanzie di successo:	L'utente naviga all'interno del catalogo e reperisce tutte le informazioni di interesse relative ai prodotti venduti nel sito.	
Avvio:	L'utente richiede di consultare il catalogo degli articoli.	

Scenario principale.		
	UTENTE	SISTEMA
1.	Richiede esplicitamente di eseguire la funzione di consultazione articoli.	
2.		Visualizza l'indice del catalogo suddiviso per categoria prodotti. (Cfr: GUI_IND_CAT)
3.	Seleziona la pagina da esaminare.	
4.		Visualizza la pagina richiesta.
5.	Prosegue nella navigazione scorrendo (avanti e indietro) le varie pagine, seguendo i link presenti nella stessa, tornando all'indice generale, oppure terminando la consultazione.	
6.		Visualizza la pagina richiesta dall'utente

Scenario alternativo: Utente seleziona un articolo specifico.		
	UTENTE	SISTEMA
5.1.	Seleziona l'opzione visualizzazione dei dettagli dell'articolo.	
5.2.		*Punto di estensione:* *visualizza dettagli articolo.* *Condizione:* *l'utente seleziona link per la visualizzazione dei dettagli dell'articolo*
5.3	Seleziona il link di ritorno all'indice.	
		Continua l'esecuzione dal punto 2

Scenario di errore: L'utente decide di abbondare la fruizione del catalogo. *Punti: 3, 5, 5.1, 5.3*	
	Il sistema termina la fruizione del catalogo

Tabella 4.27

CASO D'USO: UC_ECMMRC12	**Consulta offerte**	**Data:** 06/II/2001 **Versione:** 0.00.001
Descrizione:	Permette all'utente di consultare le eventuali offerte commerciali proposte nel sito.	
Durata:	Minuti.	
Priorità:	Media.	
Attore primario:	Utente.	
	Ha interesse nel fruire delle informazioni relative a eventuali prodotti in offerta.	
Precondizioni:	Siano presenti i dati relativi a promozioni commerciali.	
Garanzie di fallimento:	Il dati non vengono alterati.	
Garanzie di successo:	L'utente fruisce della informazioni relative agli articoli in offerta.	
Avvio:	L'utente richiede di consultare la lista degli articoli in offerta.	

Scenario principale.		
	UTENTE	SISTEMA
1.	Richiede esplicitamente di consultare le offerte commerciali disponibili.	
2.		Reperisce la lista degli articoli in offerta
3.		Visualizza la pagina con le offerte commerciali valide. (Cfr: GUI_LST_ART_OFF)
4.	Naviga le pagine delle offerte, eventualmente segue i link presenti e/o seleziona determinati articoli, oppure decidere di terminare la consultazione.	

Scenario alternativo: Offerte commerciali non disponibili.	
	SISTEMA
2.1.	Non sono presenti offerte commerciali (oppure quelle presenti non soddisfano i criteri di applicabilità, come per esempio periodo scaduto).
2.2.	Visualizza apposito messaggio e conclude la funzione.
2.3.	Termina l'esecuzione dello use case.

Scenario alternativo: Utente seleziona un articolo specifico.		
	UTENTE	SISTEMA
4.1	Seleziona l'opzione visualizzazione dei dettagli dell'articolo.	
4.2		*Punto di estensione:* *visualizza dettagli articolo.* *Condizione:* *l'utente seleziona link per la visualizzazione dei dettagli dell'articolo*
4.3	Seleziona il link di ritorno all'indice.	
4.4		Continua l'esecuzione dal punto 2

Scenario di errore: L'utente decide di abbondare la fruizione del catalogo. *Punti: 4, 4.1, 4.3*
Il sistema termina la fruizione dei prodotti in offerta

Tabella 4.28

CASO D'USO: UC_ECMMRC13	**Ricerca articolo**	**Data:** 06/II/2001 **Versione:** 0.00.001
Descrizione:	Permette all'utente di reperire le informazioni relative uno specifico prodotto.	
Durata:	Secondi.	
Priorità:	Elevata.	
Attore primario:	Utente.	
	Ha interesse nel reperire informazioni relative a specifici articoli.	
Precondizioni:	Il catalogo degli articoli non sia vuoto.	
Garanzie di fallimento:	Nessun dato viene modificato.	
Garanzie di successo:	L'utente reperisce le informazioni relative all'articolo desiderato.	
Avvio:	L'utente richiede di eseguire la funzione di reperimento articoli.	

Scenario principale.		
	UTENTE	SISTEMA
1.	Richiede esplicitamente l'esecuzione della funzione di reperimento articoli	
2.		Visualizza la pagina per l'impostazione dei criteri da utilizzare per la ricerca. (Cfr: GUI_PAG_RIC_PRD)
3.	Imposta i criteri di ricerca desiderati (codice, descrizione, categoria, limiti di prezzo, ecc.).	
4.		Reperisce la lista degli articoli le cui caratteristiche soddisfano i criteri di ricerca.
5.		Visualizza l'elenco degli articoli reperiti, suddivisi per categoria.
6.	L'utente termina la fruizione del servizio.	
7.		Termina l'esecuzione del caso d'uso.

Scenario alternativo: Nessun articolo presente nel sito soddisfa i criteri impostati dall'utente.	
	SISTEMA
4.1.	Non sono presenti nel sito articoli corrispondenti ai criteri di ricerca impostati.
4.2.	Visualizza apposito messaggio.
4.3.	Torna al punto 3.

Scenario alternativo: Utente seleziona un articolo specifico.		
	UTENTE	SISTEMA
6.1	Seleziona l'opzione visualizzazione dei dettagli di un articolo.	
6.2		*Punto di estensione:* *visualizza dettagli articolo.* *Condizione:* *l'utente seleziona link per la visualizzazione dei dettagli dell'articolo*
6.3	Seleziona il link di ritorno alla pag. di ricerca.	
6.4		Continua l'esecuzione dal punto 3

Scenario di errore: L'utente decide di abbondare la fruizione del catalogo. *Punti: 3, 6.1, 6.3*
Il sistema termina la fruizione dei prodotti in offerta

Tabella 4.29

CASO D'USO: UC_ECMMRC14	**Visualizza dettagli articolo**	**Data:** 06/II/2001 **Versione:** 0.00.001
Descrizione:	Permette all'utente di navigare le informazioni relative ad uno specifico articolo ed eventualmente selezionare uno dei seguenti servizi: aggiunta dell'articolo nel carrello della spesa, impostazione del servizio di segnalazione offerte per lo specifico articolo.	
Durata:	Secondi.	
Priorità:	Elevata.	
Attore primario:	Utente.	
	Ha interesse nel fruire delle informazioni di dettaglio dello specifico articolo.	
Precondizioni:	Il codice dell'articolo da visualizzare sia stato precedentemente individuato per mezzo di un altro servizio.	
Garanzie di fallimento:	Nessun dato viene modificato.	
Garanzie di successo:	L'utente fruisce delle informazioni relative all'articolo desiderato.	
Avvio:	L'utente seleziona l'opzione di visualizzazione di uno specifico articolo durante l'esecuzione di un altro servizio.	

Scenario principale.		
Definizione punto di estensione: Visualizza dettagli articolo.		
	UTENTE	SISTEMA
1.		Richiede tutte le informazioni relative all'articolo selezionato (descrizione dettagliata, peso, forma, ecc.). (Cfr: Modello ad oggetti del dominio, dettagli articolo)
2.		Visualizza la pagina con le informazioni di dettaglio dell'articolo. (Cfr: GUI_PAG_DTT_PRD)
3.	Naviga all'interno delle pagina delle informazioni.	
4.	Seleziona la fine della fruizione del servizio.	
5.		Termina l'esecuzione del caso d'uso.

Scenario alternativo: Utente seleziona l'opzione aggiungi articolo al carrello della spesa.	
	SISTEMA
3.1.	*Punto di estensione:* *aggiunta al carrello della spesa.* *Condizione:* Utente seleziona l'opzione aggiungi articolo al carrello della spesa.

Scenario alternativo: Utente seleziona l'opzione aggiungi segnalazione offerte.	
	SISTEMA
3.1.	*Punto di estensione:* *aggiunta segnalazione offerte.* *Condizione:* Utente seleziona l'opzione aggiungi segnalazione offerte.

Tabella 4.30

CASO D'USO: UC_ECMMRC15	**Aggiungi articolo al carrello della spesa**	**Data:** 06/II/2001 **Versione:** 0.00.003
Descrizione:	Permette di aggiungere l'articolo selezionato al carrello della spesa.	
Durata:	Secondi.	
Priorità:	Elevata.	
Precondizioni:	L'utente sia stato precedentemente autenticato dal sistema, e abbia selezionato l'articolo che intende aggiungere al relativo carrello della spesa.	
Garanzie di fallimento:	Nessuna modifica viene apportata al carrello della spesa.	
Garanzie di successo:	L'articolo selezionato dall'utente viene correttamente inserito nel relativo carrello della spesa.	
Avvio:	L'utente richiede esplicitamente di inserire l'articolo selezionato nel relativo carrello della spesa.	
Scenario principale.		
	Definizione punto di estensione: aggiunta al carrello della spesa.	
	UTENTE	SISTEMA
1.		Reperisce il carrello della spesa dell'utente.
2.		Reperisce i dati dell'articolo
3.		Chiede conferma a proseguire.
4.	Conferma l'aggiunta dell'articolo nel carrello della spesa	
5.		Inserisce i dati dell'articolo nel carrello della spesa.
6.		Visualizza messaggio di avvenuto inserimento dell'articolo nel carrello della spesa.
Scenario alternativo: Il cliente non dispone di un carrello della spesa.		
1.1.	Il sistema genera una nuova istanza vuota del carrello.	
Scenario di errore: Il cliente non conferma l'aggiunta dell'articolo nel carrello.		
4.1.	Il sistema mostra opportuno messaggio.	
4.2	Termina l'esecuzione del caso d'uso con insuccesso.	

Tabella 4.31

CASO D'USO: UC_ECMMRC15	**Aggiungi segnalazione offerte**	**Data:** 06/11/2001 **Versione:** 0.00.005
Descrizione:	Permette di impostare una segnalazione utente (allarme), qualora l'articolo selezionato sia oggetto di un'offerta commerciale.	
Durata:	Secondi.	
Priorità:	Bassa.	
Precondizioni:	L'utente sia stato precedentemente autenticato dal sistema, e abbia selezionato l'articolo che intende aggiungere al servizio notifiche offerte.	
Garanzie di fallimento:	Nessuna modifica viene apportata all'elenco notifiche offerte.	
Garanzie di successo:	L'articolo selezionato dall'utente viene correttamente inserito nel relativo elenco notifiche offerte.	
Avvio:	L'utente richiede esplicitamente di inserire l'articolo selezionato nel relativo servizio notifiche offerte.	

Scenario principale.		
	Definizione punto di estensione: aggiunta segnalazione offerte.	
	UTENTE	SISTEMA
1.		Reperisce l'elenco delle segnalazioni offerte dell'utente.
2.		Reperisce i dati dell'articolo
3.		Visualizza la pagina in cui impostare il periodo di validità della segnalazione e le condizioni. (Cfr: GUI_IMP_SGN_OFF) (Cfr: Modello ad oggetti del dominio, servizio notifiche)
4.	Imposta i criteri della segnalazione (come per esempio prezzo inferiore di una certo valore, offerta 3x2, ecc.)	
5.		Verifica che i dati siano impostati correttamente. (Cfr: Business rule BR_VAL_SEG_OFF)
6.		Inserisce la nuova istanza di segnalazione offerte.
7.		Visualizza messaggio di avvenuto inserimento.
Scenario alternativo: Il cliente non dispone di un elenco segnalazioni.		
1.1.	Il sistema genera una nuova istanza vuota dell'elenco segnalazioni.	
Scenario alternativo: I dati impostati dal cliente non sono corretti.		
5.1.	Il sistema visualizza opportuno messaggio di errore.	
5.2.	Torna al punto 3.	
Scenario di errore: Il cliente richiede la terminazione del servizio.		
4.1.	Il sistema mostra opportuno messaggio.	
4.2	Termina l'esecuzione del caso d'uso con insuccesso.	

Tabella 4.32

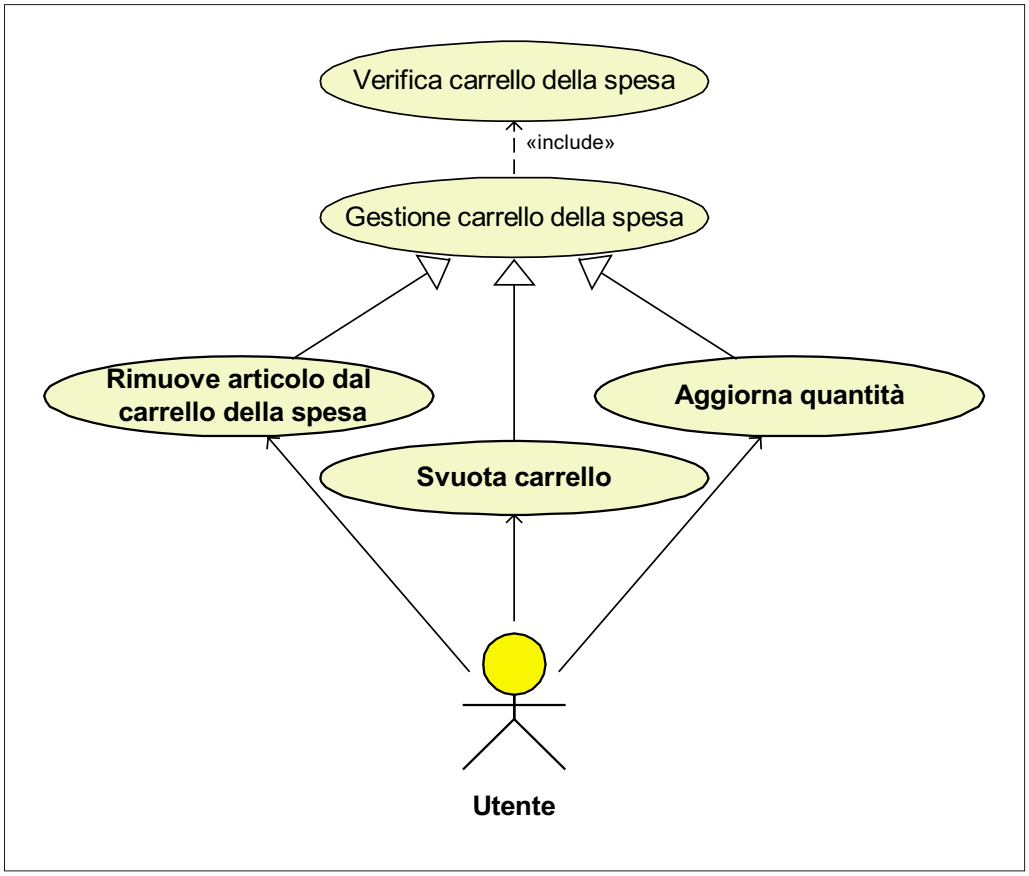
CASO D'USO: UC_ECMMRC17	**Gestione carrello della Spesa**	**Data:** 08/II/2001 **Versione:** 0.00.003
Descrizione:	Permette all'utente di gestire l'elenco degli articoli inseriti nel relativo carrello della spesa, rimuovere alcuni, modificare le quantità di altri, e così via.	
Durata:	Minuti.	
Priorità:	Media.	
Attore primario:	Utente.	
	Ha interesse nel gestire le informazioni relative al proprio carrello della spesa.	
Precondizioni:	L'utente sia stato autorizzato ad eseguire il servizio e disponga di un'istanza del carrello della spesa.	
Garanzie di fallimento:	I dati relativi al carrello della spesa utente non vengono modificati.	
Garanzie di successo:	L'utente apporta le modifiche desiderate ai dati relativi agli articoli presenti nel relativo carrello della spesa.	
Avvio:	L'utente richiede di eseguire la funzione di gestione del carrello della spesa.	

Scenario principale.		
	UTENTE	SISTEMA
1.	L'utente richiede esplicitamente l'esecuzione del servizio di gestione del carrello della spesa.	
2.		Reperisce il carrello della spesa relativo all'utente.
3.		*Include(verifica carrello della spesa)*
4.		Visualizza la pagina con i dati del carrello della spesa. (Cfr: GUI: GUI_VIS_CAR_SPS)
5.	*Punto astratto:* *Modifica dati carrello della spesa*	
6.		Termina l'esecuzione del caso d'uso
Scenario alternativo: Carrello della spesa dell'utente vuoto.		
2.1.	Visualizza apposito messaggio e conclude la funzione.	
2.2.	Termina l'esecuzione dello use case.	

Tabella 4.33

CASO D'USO: UC_ECMMRC18	**Verifica carrello della spesa**	**Data:** 08/II/2001 **Versione:** 0.00.003
Descrizione:	Esamina i dati relativi al carrello della spesa al fine di verificare se vi siano variazioni sostanziali nelle informazioni memorizzate (prezzi articoli variati, articoli non più disponibili, offerte scadute, nuove offerte, ecc.).	
Durata:	Minuti.	
Priorità:	Media.	
Precondizioni:	Il carrello della spesa del cliente è disponibile e vi sono degli articoli inseriti.	
Garanzie di fallimento:	Il carrello della spesa del cliente non viene modificato.	
Garanzie di successo:	Il carrello della spesa viene analizzato ed eventuali incoerenze vengono opportunamente segnalate.	
Scenario principale.		
	SISTEMA	
1.	Preleva l'elenco degli articoli inseriti nel carrello della spesa. (Cfr: Modello ad oggetti del dominio, carrello della spesa)	
2.	Verifica che tutti gli articoli siano ancora disponibili nel catalogo.	
3.	Reperisce i dati di tutti gli articoli ancora disponibili.	
4.	Verifica che il prezzo degli articoli non sia variato nell'intervallo di tempo che va dal momento in cui l'articolo è stato inserito nel carrello della spesa al momento attuale.	
5.	Verifica che le eventuali offerte presenti non siano scadute.	
6.	Verifica che per gli articoli inseriti non siano disponibili eventuali offerte non considerate.	
7.	Termina lo use case con successo.	
Scenario alternativo: Presenti articoli nel carrello della spesa rimossi dal catalogo.		
2.1.	Nell'istanza della riga del carrello della spesa relativa ad ognuno di essi viene impostato il flag di articolo non più disponibile. (Cfr: Modello ad oggetti del dominio, RigaArticolo).	
Scenario alternativo: Il prezzo di alcuni articoli è variato.		
4.1.	Nell'istanza della riga del carrello della spesa relativa ad ognuno di essi viene aggiornato il prezzo e, contestualmente, viene impostato il flag di prezzo variato. (Cfr: Modello ad oggetti del dominio, RigaArticolo).	
Scenario alternativo: Alcune offerte presenti all'atto dell'inserimento dell'articolo nel carrello della spesa sono cessate.		
5.1.	Nell'istanza della riga del carrello della spesa relativa ad ognuno di essi viene impostato il flag di offerta non più disponibile. (Cfr: Modello ad oggetti del dominio, RigaArticolo).	
Scenario alternativo: Per alcuni articoli è disponibile un'offerta non disponibile all'atto dell'inserimento dell'articolo nel carrello della spesa.		
6.1.	Nell'istanza della riga del carrello della spesa relativa ad ognuno di essi viene impostato il flag di offerta disponibile e, contestualmente, viene aggiunta la referenza all'offerta. (Cfr: Modello ad oggetti del dominio, RigaArticolo).	

Figura 4.17 — *Use case gestione carrello della spesa.*



Qualora ce ne fosse ancora bisogno, il caso d'uso riportato nella tab. 4.33 evidenzia quanto sia utile riferirsi ai modelli ad oggetti del dominio durante la specifica dei requisiti utente e quanto sia importante mantenere i due manufatti consistenti.

Il caso d'uso riportato nel template di tab. 4.34 e i successivi dispongono di una descrizione del comportamento dinamico piuttosto ridotta. Ciò potrebbe portare alla conclusione che probabilmente si è scelto un eccessivo livello di granularità. Si tratta di un'opinione del tutto plausibile; in questo contesto si è deciso di riportare comunque questi use case per poter visualizzare, direttamente dall'analisi visiva dello use case diagram, quali funzioni siano disponibili nel menù di gestione del carrello della spesa. Inoltre si è utilizzato il pretesto per mostrare l'approccio formale per la gestione di casi d'uso di tipo CRUD.

Tabella 4.34

CASO D'USO: UC_ECMMRC19	**Rimuove articolo dal carrello della spesa**	**Data:** 08/II/2001 **Versione:** 0.00.001
Descrizione:	Permette all'utente di rimuovere articoli precedentemente inseriti nel carrello della spesa.	
Durata:	Minuti.	
Priorità:	Bassa.	
Precondizioni:	Il carrello della spesa del cliente è disponibile e vi sono degli articoli inseriti.	
Garanzie di fallimento:	Il carrello della spesa del cliente non viene modificato.	
Garanzie di successo:	Gli articoli selezionati dall'utente vengono correttamente rimossi dal carrello della spesa.	
Avvio:	L'utente richiede esplicitamente di rimuovere alcuni articoli presenti nel carrello della spesa.	

Scenario principale.		
	UTENTE	SISTEMA
1.	Seleziona uno o più articoli presenti nel carrello della spesa.	
2.	Preme il tasto di eliminazione degli articoli selezionati.	
3.		Chiede conferma a procedere.
4.	Conferma l'eliminazione degli articoli	
5.		Elimina gli articoli selezionati
Scenario di errore: L'utente non conferma l'eliminazione degli articoli.		
4.1.	Termina lo use case con insuccesso.	

Tabella 4.35

CASO D'USO: UC_ECMMRC20	**Aggiorna quantità**	**Data:** 08/II/2001 **Versione:** 0.00.001
Descrizione:	Permette all'utente di variare la quantità richiesta di un articolo inserito nel carrello della spesa.	
Durata:	Minuti.	
Priorità:	Bassa.	
Precondizioni:	Il carrello della spesa del cliente è disponibile e vi sono degli articoli inseriti.	
Garanzie di fallimento:	I dati del carrello della spesa del cliente non vengono modificati.	
Garanzie di successo:	Viene variata la quantità di alcuni articoli presenti nel carrello della spesa.	
Avvio:	L'utente varia la quantità di uno specifico articolo presente nel carrello della spesa.	

Scenario principale.		
	UTENTE	SISTEMA
1.	Aggiorna il campo quantità riportato a fianco degli articoli presenti nella lista.	
2.	Preme il tasto di aggiornamento delle quantità	
3.		Memorizza le nuove quantità
4.		Ricalcala i vari totali

Tabella 4.36

CASO D'USO: UC_ECMMRC21	**Svuota carrello**	**Data:** 08/II/2001 **Versione:** 0.00.001
Descrizione:	Permette all'utente di eliminare tutti gli articoli presenti nel carrello della spesa.	
Durata:	Secondi.	
Priorità:	Bassa.	
Precondizioni:	Il carrello della spesa del cliente è disponibile e vi sono degli articoli inseriti.	
Garanzie di fallimento:	I dati relativi al carrello della spesa del cliente non vengono modificati.	
Garanzie di successo:	Tutti le informazioni presenti nel carrello dell'utente vengono rimosse.	
Avvio:	L'utente richiede esplicitamente di svuotare il carrello della spesa.	

Scenario principale.		
	UTENTE	SISTEMA
1.	Preme il tasto di eliminazione degli articoli presenti nel carrello.	
2.		Chiede conferma a procedere.
3.	Conferma l'eliminazione degli articoli	
4.		Elimina gli tutti gli articoli dal carrello.
Scenario di errore: L'utente non conferma l'eliminazione degli articoli.		
3.1.	Termina lo use case con insuccesso.	

Figura 4.18 — *Use case effettua ordine.*

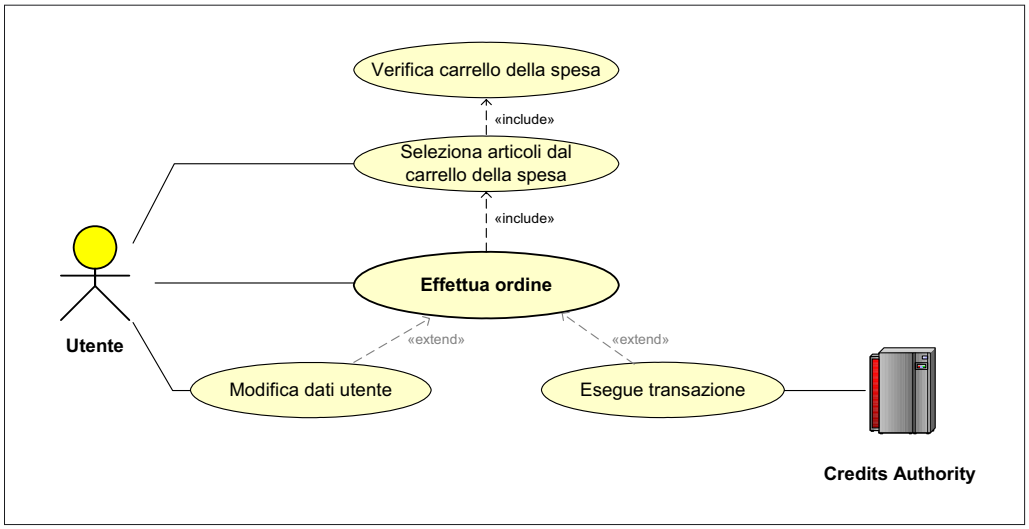


Tabella 4.37

CASO D'USO: UC_ECMMRC22	**Effettua ordine**	**Data:** 10/II/2001 **Versione:** 0.01.001
Descrizione:	Permette all'utente di effettuare un ordine selezionando gli articoli precedentemente inseriti nel carrello della spesa.	
Durata:	Minuti.	
Priorità:	Elevata.	
Attore primario:	Utente.	
	Ha interesse nell'effettuare un ordine di acquisto on-line.	
Precondizioni:	L'utente è stato autorizzato ad eseguire il servizio ed il carrello della spesa dell'utente non è vuoto.	
Garanzie di fallimento:	L'ordine non viene compilato e nessun importo viene addebitato al cliente.	
Garanzie di successo:	L'ordine viene compilato e confermato ed il relativo importo viene debitamente ascritto al cliente.	
Avvio:	L'utente richiede di eseguire la funzione di compilazione ordini.	

Scenario principale.		
	UTENTE	SISTEMA
1.	Richiede esplicitamente di effettuare un nuovo ordine.	
2.		Reperisce il carrello della spesa relativo all'utente.
3.		*Include(seleziona articoli dal carrello della spesa)*
4.		Reperisce i dati relativi al profilo dell'utente.
5.		Visualizza la pagina con i dati dell'ordine impostati. (Cfr: GUI_DTT_ORD)
6.	Verifica i dati ed eventualmente decide di modificare i propri.	
7.		*Punto di estensione:* *modifica dati utente.* *Condizione:* *L'utente intende modificare i propri dati.*
8.		Richiede conferma a procedere.
9.	Conferma dati ordine.	
10.		*Punto di estensione:* *esegue transazione* *condizione:* *l'utente conferma l'ordine*
11.		Memorizza l'ordine.
12.		Svuota il carrello della spesa.
Scenario di errore: L'utente richiede di terminare anticipatamente il servizio. *Punti 6,9.*		
6.1.	Visualizza apposito messaggio.	
6.2.	Termina lo use case con insuccesso.	
Scenario di errore: La selezione degli articoli presenti nel carrello della spesa termina con errore.		
3.1.	Visualizza apposito messaggio.	
3.2.	Termina lo use case con insuccesso.	

Tabella 4.38

CASO D'USO: UC_ECMMRC23	**Seleziona articoli dal carrello della spesa**	**Data:** 10/II/2001 **Versione:** 0.00.003
Descrizione:	Permette all'utente di selezionare gli articoli presenti nel carrello della spesa da incorporare nell'ordine di acquisto, eventualmente modificandone la quantità.	
Durata:	Minuti.	
Priorità:	Elevata.	
Attore primario:	Utente.	
	Ha interesse nel selezionare gli articoli da inserire nell'ordine di acquisto.	
Precondizioni:	Il carrello della spesa dell'utente è disponibile e non è vuoto.	
Garanzie di fallimento:	I dati relativi al carrello della spesa non vengono modificati.	
Garanzie di successo:	L'utente seleziona la lista degli articoli da impostare nell'ordine, eventualmente variandone la quantità.	

Scenario principale.		
	UTENTE	SISTEMA
1.		*Include*(verifica carrello della spesa)
2.		Visualizza l'elenco degli articoli presenti nel carrello della spesa con eventuali segnalazioni scaturite dal controllo eseguito nel punto precedente. (Cfr: GUI_CRR_SPS_DTT)
3.	Seleziona gli articoli che intende far confluire nell'ordine eventualmente variandone la quantità.	
4.	Preme il tasto di conferma.	
5.		Genera la lista con gli articoli selezionati dall'utente.
Scenario di errore: L'utente richiede di terminare anticipatamente la funzione. Punti 3, 4		
3.1.	Termina lo use case ritornando la lista degli articoli vuota.	

L'attore identificato con il nome generico *Credits Authority* rappresenta uno di quei sistemi che offrono servizi automatici di controllo ed esecuzione di transazioni economiche digitali effettuate tramite carta di credito. Utenti di tali servizi sono prevalentemente organizzazioni commerciali.

Il ciclo di vita tipico degli ordini non prevede l'immediato invio presso il legacy system: essi vengono invece parcheggiati in una cartella dedicata ove soggiornano attendendo che un apposito servizio temporizzato li prelevi, li organizzi in un messaggio e quindi, finalmente, li spedisca al legacy system.

Tabella 4.39

CASO D'USO: UC_ECMMRC24	**Modifica dati utente**	**Data:** 10/II/2001 **Versione:** 0.00.003
Descrizione:	Permette all'utente di modificare i propri dati e di impostarne altri di carattere fiscale (codice della carta di credito, scadenza, ecc.).	
Durata:	Minuti.	
Priorità:	Media.	
Attore primario:	Utente.	
	Ha interesse nel modificare i propri dati.	
Precondizioni:	È disponibile il profilo utente.	
Garanzie di fallimento:	I dati dell'utente non subisco alcuna variazione.	
Garanzie di successo:	L'utente imposta i dati fiscali ed eventualmente modifica altri (quali per esempio indirizzo di spedizione, fascia oraria, ecc.)	

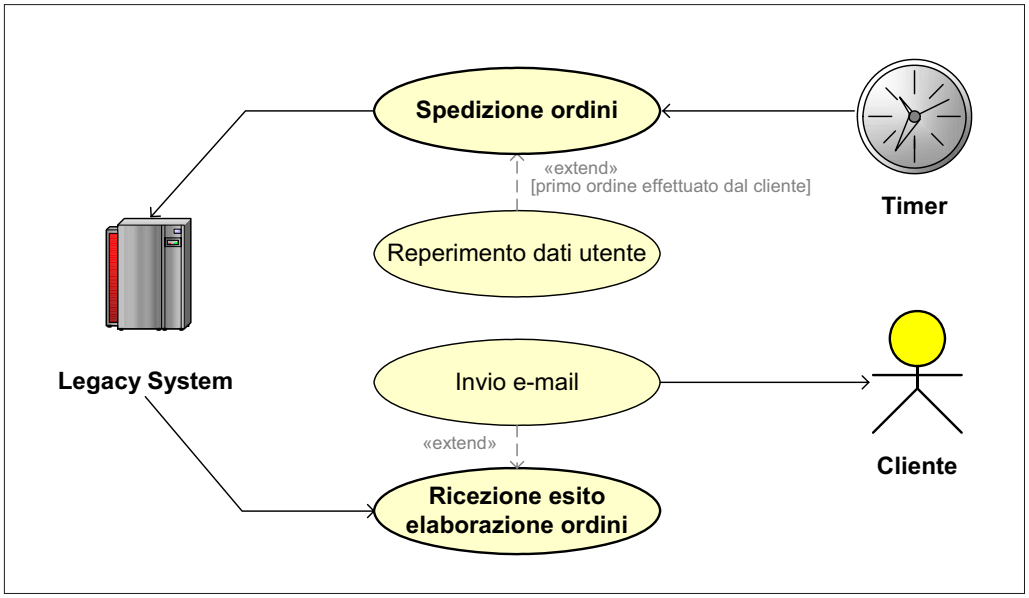
Scenario principale.		
	UTENTE	SISTEMA
1.		Visualizza la pagina con i dati dell'utente presenti nel sistema. (Cfr: GUI_DT_CL)
2.	Conferma i dati visualizzati.	
3.		Richiede di inserire i dati fiscali. (Cfr: GUI_DT_FSC)
4.	Imposta i dati fiscali e conferma.	
5.		Verifica che i dati impostati siano corretti.
6.		Richiede di selezionare l'indirizzo di spedizione. (Cfr: GUI_ADD_SPED, Modello ad oggetti del dominio, dati utente)
6.	Seleziona uno degli indirizzi presenti.	
7.		Memorizza i dati impostati
Scenario alternativo: L'utente modifica i dati del proprio profilo.		
	UTENTE	SISTEMA
2.1.		Visualizza la pagina con i dati dell'utente editabili. (Cfr: GUI_UP_DT_UT)
2.2.	Modifica i dati visualizzati e conferma.	
2.3.		Verifica che i dati impostati siano corretti.
2.4.		Memorizza i dati impostati
Scenario alternativo: Dati utente impostati errati		
2.3.1.	Visualizza apposito messaggio e torna al punto 2.1.	
Scenario alternativo: Dati fiscali impostati errati		
4.1.	Visualizza apposito messaggio e torna al punto 3.	
Scenario alternativo: L'utente inserisce un nuovo indirizzo.		
	UTENTE	SISTEMA
6.1.		Visualizza la pagina con i dati indirizzo editabili. (Cfr: GUI_UP_IND_UT)
6.2.	Modifica i dati visualizzati e conferma.	
6.3.		Verifica che i dati impostati siano corretti.
6.4.		Memorizza i dati impostati
Scenario alternativo: Dati indirizzo impostati errati		
6.3.1.	Visualizza apposito messaggio e torna al punto 6.1	
Scenario di errore: L'utente richiede di terminare anticipatamente il servizio. *Punti 2,4,6, 2.2, 6.2*		
3.1.	Visualizza apposito messaggio e termina lo use case con insuccesso	

Tabella 4.40

CASO D'USO: UC_ECMMRC26	**Esegue transazione**	**Data:** 10/II/2001 **Versione:** 0.00.003
Descrizione:	Il sistema richiede l'autorizzazione ad eseguire una determinata transazione commerciale per mezzo di carta di credito (i cui dati sono stati precedentemente impostati), al fine di accreditare l'importo dell'ordine al cliente.	
Durata:	Secondi.	
Priorità:	Elevata.	
Attore primario:	Credits Authority.	
	Ha la responsabilità di autorizzare o rifiutare le transazioni di addebito tramite carta di credito.	
Precondizioni:	I dati fiscali dell'ordine sono disponibili.	
Garanzie di fallimento:	Non viene eseguito alcun addebito al cliente.	
Garanzie di successo:	La transazione viene accettata e quindi l'importo dell'ordine viene accreditato sulla carta di credito del cliente.	

Scenario principale.		
	CREDITS AUTHORITY	SISTEMA
1.		Preleva i dati necessari per la transazione.
2.		Organizza la richiesta secondo le specifiche della Credits Authority. (Cfr: Business rule BR_AUT_TRAN)
3.		Effettua la richiesta.
4.	Riceve la richiesta.	
5.	Esegue le verifiche del caso.	
6.	Invia esito.	
7.		Verifica autorizzazione transazione.
Scenario alternativo: Fallisce la richiesta del servizio un numero di volte inferiore a quello massimo previsto.		
4.1	Esegue procedura di notifica situazione anomala.	
4.2.	Visualizza apposito messaggio.	
4.3.	Incrementa il contatore di tentativi.	
4.4.	Torna al punto 3	
Scenario di errore: Fallisce la richiesta del servizio un numero di volte uguale a quello massimo previsto.		
4.2	Esegue procedura di notifica situazione anomala.	
4.3.	Visualizza apposito messaggio.	
4.4.	Termina lo use case con insuccesso.	
Scenario di errore: Scaduto time-out .		
7.1.	La risposta della credits authority non perviene entro l'intervallo di tempo stabilito.	
7.2.	Esegue procedura di notifica situazione anomala.	
7.3.	Visualizza apposito messaggio e conclude la funzione con insuccesso.	
Scenario di errore: Transazione rifiutata.		
8.1.	Comunica opportuno messaggio all'utente.	
8.2.	Termina lo use case con insuccesso.	
Annotazioni.		
7.1.	Nel caso di time-out è necessario investigare le misure da attuare. Reiterare la richiesta potrebbe portare ad un addebitamento ripetuto.	

Figura 4.19 — *Use case spedizione ordini.*



Da notare che non viene eseguita alcuna verifica in merito alla presenza di articoli nel carrello della spesa, in quanto ciò rappresenta una precondizione.

Lo use case illustrato in fig. 4.19 illustra il processo che permette al sistema di comunicare al *Legacy System* gli ordini effettuati on-line. Il *Legacy System*, dal canto suo, origina messaggi relativi allo stato di avanzamento degli ordini (in lavorazione, spedizione merce, ecc.) con le efferenti informazioni. Si è deciso di visualizzare come attore il *Legacy system* anziché il sistema di wrapper in quanto si suppone che il software presente presso l'organizzazione del cliente sia predisposto per l'acquisizione/invio di messaggi tipicamente effettuati per mezzo di protocolli FTP.

Durante lo svolgimento del processo dell'analisi dei requisiti, spesso risulta importante avvalersi di ulteriori formalismi al fine di formalizzare chiaramente determinati requisiti utente. A tal proposito non è infrequente il ricorso alla notazione dei diagrammi delle carte di stato utilizzate per descrivere il ciclo di vita di opportuni oggetti, eventualmente composti, particolarmente importanti nel dominio oggetto di studio business. In questo contesto, un esempio è costituito dagli ordini. In particolare, in fig. 4.20 sono mostrati gli stati nei quali può transitare e gli eventi che ne determinano la transizione da uno stato all'altro.

Tabella 4.41

CASO D'USO: UC_ECMMRC30	**Spedizione ordini**	**Data:** 10/II/2001 **Versione:** 0.00.002
Descrizione:	Consente al sistema di inviare al legacy system gli ordini effettuati residenti nel sito.	
Durata:	Secondi.	
Priorità:	Elevata.	
Attore primario:	Legacy system Riceve un messaggio con gli ordini effettuati on-line.	
Precondizioni:	Siano disponibili degli ordini non ancora inviati al legacy system.	
Garanzie di fallimento:	Nessun ordine viene inviato al legacy system e gli stessi permangono nell'apposita directory del sistema.	
Garanzie di successo:	Tutti gli ordini in attesa di essere inviati vengono opportunamente organizzati in un messaggio e quindi trasferiti al legacy system	
Avvio:	Attivazione temporizzato del processo di invio ordini.	

Scenario principale.		
	LEGACY SYSTEM	SISTEMA
1.		Verifica che ci siano ordini da trasferire al legacy system.
2.		Reperisce tutti gli ordini da inviare
3.		Organizza i dati secondo il formato previsto.
4.		*Punto di estensione:* *Reperimento dati utente.* *Condizione:* *Si tratta del primo ordine del cliente.*
5.		Trasferisce (tramite protocollo FTP) il file al sistema di legacy
6.	Riceve il file con l'elenco dei recenti ordini effettuati dagli utenti del sito.	
7.		Aggiorna lo stato degli ordini trasferiti (stato = inviati). (Cfr: Modello ad oggetti del dominio, Ordini)

Scenario alternativo: Non sono presenti nel sito ordini da spedire al Legacy System.	
1.1	Esegue procedura di notifica situazione anomala.
1.2.	Termina il caso d'uso.

Scenario alternativo: Fallisce il trasferimento file un numero di volte inferiore a quello massimo previsto.	
4.1	Esegue procedura di notifica situazione anomala.
4.2.	Visualizza apposito messaggio.
4.3.	Incrementa il contatore di tentativi.
4.4.	Torna al punto 5.

Scenario di errore: Fallisce il trasferimento file un numero di volte uguale a quello massimo previsto.	
4.2	Esegue procedura di notifica situazione anomala.
4.3.	Visualizza apposito messaggio.
4.4.	Termina lo use case con insuccesso.

Tabella 4.42

CASO D'USO: UC_ECMMRC31	**Reperimento dati utente**	**Data:** 10/II/2001 **Versione:** 0.00.001
Descrizione:	Reperisce i dati efferenti uno specifico utente.	
Durata:	Secondo.	
Priorità:	Elevata.	
Precondizioni:	È disponibile il codice dell'utente del quale si desiderano reperire i dati.	
Garanzie di fallimento:	Nessun informazione viene reperita.	
Garanzie di successo:	Vengono reperiti i dati relativi all'utente specificato.	
Scenario principale.		
	Definizione punto di estensione: reperimento dati utente.	
	SISTEMA	
1.	Preleva il codice dell'utente del quale si desidera reperire le relative informazioni.	
2.	Organizza i dati utente secondo la struttura del messaggio.	
3.	Reperisce le informazioni dell'utente.	

Figura 4.20 — *Ciclo di vita di un ordine.*

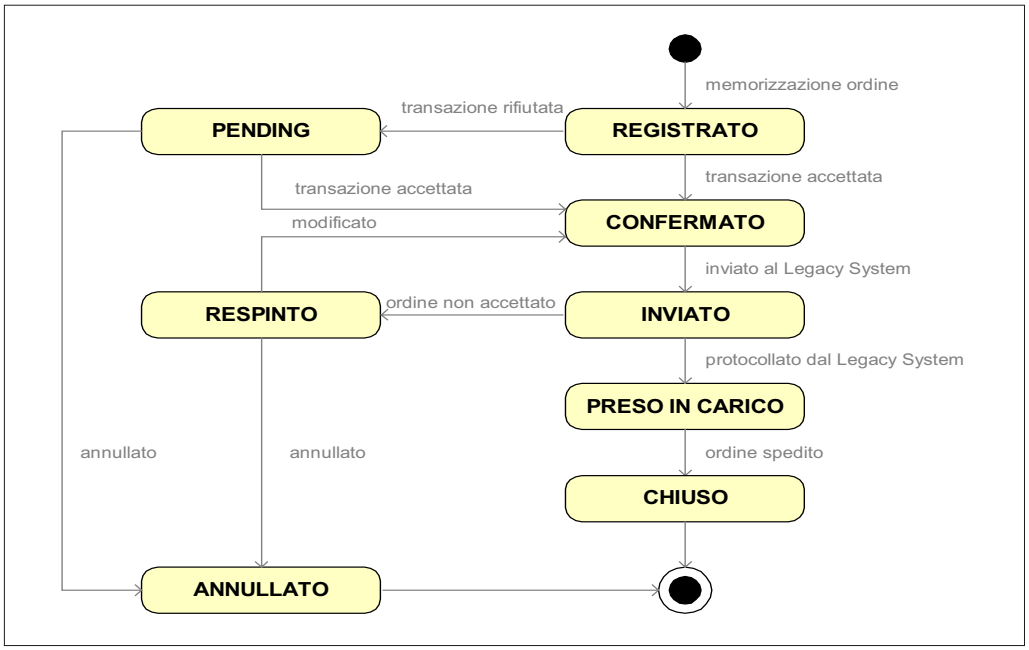


Tabella 4.43

CASO D'USO: UC_ECMMRC32	**Ricezione esito elaborazione ordini**	**Data:** 10/II/2001 **Versione:** 0.00.002
Descrizione:	Consente al sistema di ricevere il file proveniente dal legacy system con l'elenco aggiornato dello stato di elaborazione degli ordini inviati in precedenza. Per ognuno di essi è prevista apposita comunicazione all'utente effettuata via e-mail.	
Durata:	Secondi.	
Priorità:	Elevata.	
Attore primario:	Legacy system	
	Invia un messaggio con lo stato di avanzamento degli ordini effettuati on-line.	
Precondizioni:	Diversi ordini effettuati on-line siano stati precedentemente inviati al Legacy system.	
Garanzie di fallimento:	Nessun ordine viene aggiornato.	
Garanzie di successo:	Lo stato degli ordini presenti nel file ricevuto viene riportato nella base di dati del sito ed opportune comunicazione via e-mail vengono inviate agli utenti che hanno effettuato gli ordini.	
Avvio:	Il legacy system invia il file degli esiti dell'elaborazione degli ordini ricevuti.	

Scenario principale.		
	LEGACY SYSTEM	SISTEMA
1.	Invia il file con gli esiti degli ordini inviati in precedenza	
2.		Acquisisce il file inviato dal legacy system
3.		Scorre tutto uno per uno tutti i record del file. Per ogni record letto:
3.1.		- reperisce il relativo ordine dal db;
3.2.		- aggiorna lo stato dell'ordine letto in funzione a quanto specificato nel record del file;
3.3.		*Punto di estensione:* *Invio notifica al cliente.* *Condizione:* *Aggiornamento ordine effettuato.*
Scenario alternativo: Il sistema fallisce il reperimento dell'ordine dal proprio database.		
3.1.1.	Esegue la procedura di comunicazione situazione anomala.	
3.1.2.	Continua lo use case esaminando il record successivo.	
Scenario alternativo: Il sistema fallisce l'invio dell'e-mail al cliente.		
3.3.1.	Esegue la procedura di comunicazione situazione anomala.	
3.3.2.	Continua lo use case esaminando il record successivo..	
Scenario di errore: Il file ricevuto risulta vuoto.		
2.1.	Esegue la procedura di comunicazione situazione anomala.	
2.2.	Termina con insuccesso lo use case.	

Si tratta di un diagramma molto semplice e di facile comprensione: un ordine dopo essere stato memorizzato localmente nel sito si trova nella stato di *registrato*. Da questo stato transita in quello di *confermato* a seguito del ricevimento dell'autorizzazione della transazione da parte del *Credits Authority*. Una volta inviato al *Legacy System*, l'ordine passa allo stato di *inviato* permanendovi fino alla ricezione della risposta dell'elaborazione da parte del *Legacy System*. Ciò genera la transizione allo stato di *preso in carico*. Il ciclo di vita di un ordine termina quando l'organizzazione invia al cliente, la mercanzia specificata nell'ordine stesso. Il diagramma visualizza l'esistenza di due stati di errore:

- *pending* ossia il sistema non riesce a effettuare la transazione;
- *respinto* quando il *Legacy System* per qualche motivo ha respinto l'ordine.

Ogni qualvolta si ha a che fare con entità del business degne di nota, il cui ciclo di vita è limitato a un insieme ben definito di stati (in altre parole è descrivibile per mezzo di un automa a stati finiti), è opportuno tracciarne lo state chart diagram. Questo offre il vantaggio di chiarire specifici requisiti (stati in cui può transitare, eventi che ne generano il cambiamento di stato, ecc.) in maniera chiara e molto intuitiva. Questi diagrammi permettono inoltre di verificare "percorsi" che altrimenti potrebbero rimanere celati. In genere il diagramma di stato del ciclo di vita di un oggetto vale 1000 parole e fornisce documentazione molto valida anche per le fasi di costruzione del sistema.

La funzione *Invia dati catalogo* permette, come suggerito dal nome, di aggiornare il catalogo presente presso il sistema per il commercio elettronico con i dati inviati dal *Legacy System*. Gli aggiornamenti possono essere relativi all'inserimento di nuovi articoli, all'eliminazione di altri, alla modifica di alcune informazioni come il prezzo di un articolo o la relativa immagine e così via.

Figura 4.21 — *Use case ricezione dati catalogo.*

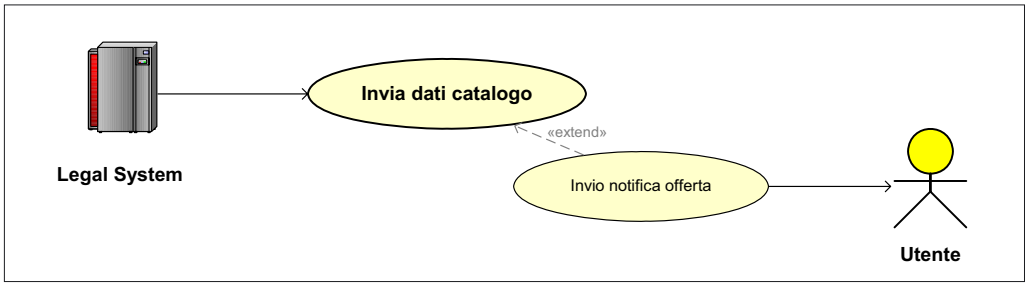


Tabella 4.44

CASO D'USO: UC_ECMMRC33	**Invia dati catalogo**	**Data:** 10/II/2001 **Versione:** 0.02.001
Descrizione:	Consente al sistema di ricevere il file con gli aggiornamenti da apportare al catalogo dei prodotti. Nello stesso file potrebbero essere presenti anche dati relativi ad offerte promozionali inerenti specifici prodotti. Qualora per i prodotti oggetti di offerta siano presenti richieste di notifica, e queste risultino soddisfatte, il sistema si occupa di inviare un'e-mail agli utenti che hanno impostato la notifica offerta.	
Durata:	Secondi.	
Priorità:	Elevata.	
Attore primario:	Legacy system	
	Invia un messaggio con i dati relativi agli aggiornamenti da apportare al catalogo prodotti presente nel sito.	
Precondizioni:	Il sistema disponga di un catalogo articoli.	
Garanzie di fallimento:	Non viene modificato alcun prodotto.	
Garanzie di successo:	Gli aggiornamenti relativi al catalogo vengono correttamente inglobati nel sistema.	
Avvio:	Il legacy system invia il file con gli aggiornamenti del catalogo.	

Scenario principale.		
	LEGACY SYSTEM	SISTEMA
1.	Invia il file con gli aggiornamenti del catalogo	
2.		Acquisisce il file inviato dal legacy system.
3.		Scorre uno per uno tutti i record del file. Per ogni record letto:
3.1.		- reperisce il relativo articolo dal db;
3.2.		- aggiorna l'articolo letto in funzione a quanto specificato nel record del file;
3.3.		*Punto di estensione:* *invia notifica offerta* *Condizione:* *siano presenti dati relativi ad un offerta commerciale per il prodotto corrente.*
Scenario alternativo: Il sistema fallisce il reperimento dell'articolo dal proprio database.		
3.1.1.	Esegue la procedura di comunicazione situazione anomala.	
3.1.2.	Continua lo use case esaminando il record successivo.	
Scenario di errore: Il file ricevuto risulta vuoto.		
2.1.	Esegue la procedura di comunicazione situazione anomala.	
2.2.	Termina con insuccesso lo use case.	

Tabella 4.45

CASO D'USO: UC_ECMMRC34	**Invia notifica offerta**	**Data:** 10/II/2001 **Versione:** 0.02.001
Descrizione:	Qualora venga inserita nel sistema un'offerta commerciale relativa ad uno specifico prodotto per il quale siano presenti richieste di notifica, il sistema verifica quali di queste siano soddisfatte e quindi provvede ad inviare un e-mail agli utenti che le hanno impostate.	
Durata:	Secondi.	
Priorità:	Elevata.	
Attore primario:	Utente	
	Ha interesse nel ricevere segnalazioni di determinati prodotti in offerta	
Precondizioni:	Un determinato articolo sia oggetto di un'offerta commerciale.	
Garanzie di fallimento:	Non viene inviata alcuna notifica.	
Garanzie di successo:	Per tutte le richieste di notifica soddisfatte relative il prodotto in questione, il sistema invia un'apposita e-mail di segnalazione dell'offerta.	
Avvio:	Viene inserita un'offerta commerciale relativa ad un determinato articolo.	

Scenario principale.		
	UTENTE	SISTEMA
1.		Reperisce l'elenco delle richieste di offerta efferenti l'articolo selezionato.
2.		Scorre tutto l'elenco delle offerte. Per ogni record letto:
2.1.		- Verifica il soddisfacimento della richiesta di notifica (periodo di validità non scaduto, prezzo inferiore ad un certo valore, ecc); (Cfr: Business rule BR_VER_VAL_OFF)
2.2.		- Se la regola è valida, reperisce i dati dell'utente che l'ha impostata e quindi invia apposita e-mail;
3.	Riceve l'e-mail con la segnalazione dell'offerta commerciale.	
Scenario di errore: Il file ricevuto risulta vuoto.		
1.1.	Non ci sono richieste di notifica relative all'articolo aggiornato.	
1.2.	Termina con insuccesso lo use case.	

Ricapitolando…

Nel presente capitolo vengono illustrate una serie di tecniche di modellazione dei casi d'uso, di orientamento specificamente operativo anche se, a volte, sensibilmente distante dalle direttive standard. Modellare sistemi reali mette in luce una serie di problemi e lacune raramente trattati nei libri e nelle specifiche ufficiali dello UML.

La prima sezione è dedicata alla presentazione di un modello atto a descrivere il comportamento dinamico dei casi d'uso. Tale modello, in prima analisi, può essere suddiviso in quattro macrosezioni: l'**intestazione**, le **informazioni generali**, gli **scenari** che a loro volta si dividono in *principali*, *alternativi* e *di errore*, e la sezione per eventuali **annotazioni**.

Nell'intestazione viene impostato un codice simbolico univoco per il caso d'uso, il nome, la data dell'ultima modifica e la versione.

La sezione dedicata alle informazioni generali ospita la descrizione del caso d'uso (non più di qualche riga), la priorità, vincoli temporali relativi all'esecuzione dello stesso, gli attori divisi tra principale e secondari, le precondizioni, le garanzie e l'evento innescante (il trigger).

Per ciò che concerne la descrizione è necessario riportare una breve illustrazione dei servizi offerti dal caso d'uso oggetto di studio senza però dilungarsi troppo: la descrizione completa viene specificata nei vari scenari.

La priorità permette di evidenziare l'importanza assegnata alle funzionalità che il sistema dovrà fornire, al fine sia di suddividere gli use case in un'opportuna gerarchia di importanza, sia per poter assegnare in maniera più opportuna i vari casi d'uso alle varie iterazioni di cui si compone tipicamente un processo di sviluppo del software.

Il processo di assegnazione delle priorità degli use case è un'attività molto importante e sensibile per l'esito dell'intero progetto software in quanto si ripercuote sulla pianificazione delle iterazioni dello stesso.

Gli attori, come già indagato precedentemente, sono entità, persone o sottosistemi, interessati al comportamento del sistema.

Tipicamente, nel contesto di uno stesso caso d'uso, gli attori vengono suddivisi in primari e secondari. I primi solitamente forniscono lo stimolo iniziale che avvia l'esecuzione del caso d'uso e fruiscono del relativo servizio, i secondi invece, in genere, ricevono comunicazioni, dati, ecc. generati dallo stesso.

Le precondizioni sono i requisiti che devono essere soddisfatti per poter eseguire il caso d'uso al quale si riferiscono: è compito del sistema verificarne l'adempimento prima di avviare l'esecuzione dell'efferente caso d'uso, mentre è responsabilità dell'utilizzatore di assicurarne il soddisfacimento.

Il riscontro pratico delle precondizioni è che l'implementazione dell'efferente caso d'uso preveda una serie di controlli iniziali atti a verificare appunto il relativo soddisfacimento.

Per ciò che concerne le garanzie è necessario citare esplicitamente, oltre a quelle minime anche quelle di successo. Le prime, come suggerito dal nome, sono le più basilari assicurate dall'esecuzione del caso d'uso. Tipicamente si tratta delle garanzie assicurate dal sistema nel caso in cui non sia possibile fornire

il servizio specificato in quanto l'esecuzione è viziata da condizioni di errore o comunque da anomalie rispetto al flusso principale del caso d'uso (*best scenario*).

Specularmente alle garanzie minime, quelle di successo specificano ulteriori condizioni soddisfatte dal completamento dell'esecuzione del relativo caso d'uso, ossia dopo l'esecuzione dello scenario principale o al termine dell'esecuzione di una sua variante comunque di successo.

Mentre è compito dell'utilizzatore del caso d'uso assicurare il soddisfacimento delle precondizioni, è responsabilità del sistema adempiere alle garanzie, qualora le prime siano soddisfatte.

Per ciò che concerne il campo *Avvio* è necessario descrivere l'evento che determina l'inizio dell'esecuzione del sistema.

La sezione successiva del modello è dedicata agli scenario. Tipicamente si distinguono tre tipologie:

1. scenario principale di successo (*main success scenario*): lo scenario che produce il servizio richiesto dall'attore primario in cui tutto funziona perfettamente;

2. scenari alternativi (*alternative scenario*) si tratta di flussi di azioni che rappresentano diramazioni dello scenario principale la cui funzionalità però non sia la gestione di condizioni di errore.

3. scenari di fallimento o di errore (*failure scenario*): gli scenari che specificano le azioni da intraprendere nel caso in cui l'esecuzione di azioni dello scenario principale sia impossibilitata dal verificarsi di condizioni di errore.

La struttura utilizzata nel template oggetto di studio prevede di specificare inizialmente la sequenza di azioni da compiere nel caso in cui tutto funzioni correttamente, dall'avvio del caso d'uso al relativo compimento, per poi fornire le condizioni anomale che possono intervenire e le azioni da intraprendere.

Molto importante è evidenziare chiaramente quale entità (*Attore*, *Sistema*) svolge ciascuna azione. Spesso si utilizzano particolari versioni di template nei quali la sezione dedicata agli scenari viene suddivisa orizzontalmente in un numero di colonne equivalenti alle entità che prendono parte al caso d'uso. Ciò permette di riportare in ciascuna colonna unicamente le azioni eseguite dall'entità di appartenenza.

L'illustrazione degli scenari alternativi effettuata attraverso opportuni template evidenzia il grande vantaggio offerto da questo formalismo rispetto a quello grafico, ove, tipicamente, è necessario disegnare un nuovo diagramma per ogni alternativa, il che richiede una quantità decisamente superiore di tempo per la realizzazione e rende difficoltosa la gestione delle relative modifiche.

L'ultima sezione del modello prevede l'impostazione di eventuali annotazioni, che possono sia essere riferite all'intero use case, sia a singole azioni.

L'utilizzo del modello descritto viene illustrato per mezzo di un primo esempio: sottosistema universitario atto a fornire a studenti e a docenti universitari una serie di servizi fruibili attraverso un browser Internet. In questa sede vengono considerati anche gli altri strumenti utilizzabili per modellare il comportamento dinamico dei casi d'uso, quali i diagrammi di sequenza e quelli di attività.

Sebbene i formalismi grafici offrano tutta una serie di vantaggi (forma più intuitiva ed accattivante, superiore grado di memorizzazione ecc.), presentano anche una serie di svantaggi tali da renderli non preferibili al modello descritto in precedenza. In particolare, all'aumentare della complessità i diagrammi tendono a diventare inevitabilmente confusi, richiedendo una quantità di tempo maggiore e talune volte decisamente spropositata per realizzazione e successiva manutenzione.

Il caso oggetto di studio successivo prevede l'analisi di alcuni componenti di un sistema di banking. Questo esempio offre tutta una serie di spunti importanti tra i quali quelli relativi al processo di sviluppo del software. In particolare si evidenzia come i diagrammi dei casi d'uso della fase di analisi debbano sia dettagliare i corrispondenti al livello dei requisiti, sia incorporare direttive provenienti dai requisiti non funzionali (NFR, *non functional requirements*) e dall'architettura, la quale a sua volta dipende dagli NFR, nonché dagli use case stessi.

Indipendentemente dalla modalità stabilita per descrivere il comportamento dinamico dei casi d'uso, qualora si decida di ricorrere a processi di sviluppo iterativi e incrementali, è utile prevedere una serie di informazioni supplementari, da associare ai vari manufatti prodotti, necessarie per tener traccia della relativa evoluzione. Queste informazioni dovrebbero riportare le varie versioni del manufatto, gli autori, la data di prevista o avvenuta consegna, la descrizione e così via.

Chiaramente non c'è alcuna necessità di specificarle per ogni singolo caso d'uso, è sufficiente riportarle solo per le macrofunzionalità (in altre parole al livello di use case diagram).

Probabilmente potrebbe risultare altrettanto conveniente disporre di un'altra tabella, in qualche modo speculare alla prima, collocata alla fine della descrizione del comportamento dinamico di ogni use case diagram. Lo scopo è di ospitare in un sol punto eventuali considerazioni del personale coinvolto nello sviluppo e nel controllo della qualità del modello.

Il terzo caso di studio che conclude il capitolo prevede un'ampia sezione della use case view di un sistema del tutto originale: sito e-commerce. Si tratta di un sistema molto citato in tutto il libro, per cui si è deciso di presentarlo finalmente in maniera più organica.