Sistemi Informativi Aziendali

Appunti per il corso - Capitolo 7

Fulvio Corno Marco Torchiano

Politecnico di Torino – Dipartimento di Automatica e Informatica

Versione 0.1.1

8 gennaio 2018

INDICE

CAPITOLO 7

REQUISITI FUNZIONALI E CASI D'USO

Example is always more efficacious than precept.
Samuel Johnson

I casi d'uso sono una tecnica utilizzata per descrivere alcune tipologie di requisiti funzionali. A definirli, nella loro versione originale, fu Ivar Jacobson negli anni '90 [1]; successivamente hanno trovato una grande diffusione soprattutto nello sviluppo di software con l'approccio orientato agli oggetti.

Servono per catturare una sorta di mini-contratto tra gli stakeholder e chi sviluppa il software, che in maniera sintetica descrive come si deve comportare il sistema. I casi d'uso forniscono una descrizione *dall'esterno* ossia dal punto di vista dell'utente che utilizza il software; per questo motivo sono molto comprensibili dagli utenti e da tutti gli stakeholder.

7.1 Definizione di Caso d'Uso

*Un **Caso d'Uso** è la descrizione di una serie di interazioni,*
finalizzata al raggiungimento di un obiettivo,
tra uno o più attori esterni ed il sistema in considerazione.[2]

Gli elementi chiave, illustrati in figura 7.1, della definizione sono:

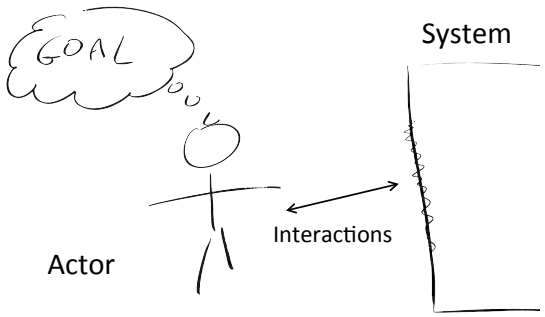

Figura 7.1: Concetti chiave di un caso d'uso

Attore è l'utente che sta usando il sistema per svolgere un determinato compito e tipicamente per raggiungere un proprio obiettivo. Non si tratta di un individuo specifico, ma indica un individuo appartenente ad un **gruppo di individui** che il sistema tratta in maniera indistinguibile, le **identifica con una categoria** che interagisce in maniera analoga con il sistema informativo.

Non si tratta necessariamente neppure di un utente umano: può rappresentare un sistema software (esterno al sistema considerato) o dei componenti hardware (es., lettore codice a barre).

È possibile (anzi, abbastanza normale) che lo stesso individuo reale possa impersonare diversi attori, in momenti diversi: tali attori rappresentano i "permessi" o "privilegi" che sono assegnati all'utente.

Obiettivo (Goal) è un qualcosa che l'attore può raggiungere tramite l'aiuto del sistema informativo; deve trattarsi di qualcosa che ha un valore specifico per l'utente.

È espresso dal punto di vista dell'attore e rappresenta le funzioni che il sistema informativo può offrire all'attore.

Sistema rappresenta quello che deve essere specificato e costruito e viene visto come una *scatola nera* di cui si osserva la superficie e come interagisce con gli attori, ma i cui dettagli interni sono lasciati non dettagliati, nascosti.

Questi sono gli elementi chiave che definiscono il caso d'uso. La descrizione di un caso d'uso si concentra sulle interazioni tra l'attore e il sistema, sulle informazioni che si scambiano e sull'ordine delle operazioni.

Il punto di partenza è un obiettivo che un utente del sistema vuole raggiungere: cosa un utente che vuole svolgere una particolare attività si può attendere dal sistema. Il caso d'uso descrive quindi come il sistema permette all'utente (attore) di raggiungere uno specifico obiettivo.

Nel mini-contratto rappresentato da un caso d'uso viene descritto come il sistema permette di raggiungere l'obiettivo proteggendo gli interessi dei vari stakeholder.

Due sono le figure prese in considerazione dai casi d'uso.

L'**attore**, che interagisce direttamente con il sistema. Ad esempio, per un bancomat (ATM) l'attore è il cliente della banca e può avere come obiettivo prelevare dei soldi dal bancomat. Un caso d'uso può essere quello in cui il cliente interagisce con il bancomat per prelevare del contante (cioè raggiungere l'obiettivo).

Lo **stakeholder** (portatore di interesse) rappresenta una persona o un gruppo di persone che hanno un interesse (diretto o indiretto) nel sistema (e nel fatto che gli attori raggiungano i propri obiettivi). Sia il cliente sia gli azionisti della banca sono degli stakeholder perché hanno degli interessi nei confronti del sistema, tali interessi sono diversi e potrebbero essere contrastanti. Di norma, uno stakeholder non interagisce direttamente con il sistema (a meno che non sia uno degli attori).

In questo capitolo si vedrà la descrizione dei casi d'uso attraverso due rappresentazioni distinte e complementari:

- il *diagramma dei casi d'uso*, utile per rappresentare l'insieme di tutti i casi d'uso offerti da un sistema, ed identificare gli attori che hanno accesso ad essi;
- la *descrizione narrativa di un caso d'uso*, che mira a descrivere nel dettaglio la sequenza delle possibili interazioni tra attore e sistema, in uno specifico caso d'uso.

A sua volta, la descrizione (o narrativa) del caso d'uso può avere livelli di dettaglio diversi:

Breve si riassume in poche frasi quelle che sono le interazioni tra attore e Sistema. Si tratta di un livello "macro" di descrizione, privo di dettagli.

Informale si racconta il caso d'uso attraverso una storia.

Dettagliato si fornisce un elenco numerato sequenzialmente delle interazioni e delle informazioni scambiate dall'attore e dal sistema.

7.1.1 Attori e obiettivi

La descrizione di un caso d'uso richiede di svolgere tre passi di analisi:

1. Identificare gli attori: le entità (esseri umani, o sistemi informativi) che stanno all'esterno del sistema e interagiscono con esso.
2. Capire i loro obiettivi: perché vogliono interagire con il sistema.
3. Definire come l'attore raggiunge il proprio obiettivo: quali interazioni sono necessarie con il sistema.

Nell'identificare gli attori è bene distinguere tra:

Attore Primario è colui che possiede un obiettivo da raggiungere, è tipicamente colui che inizia l'uso del sistema.

Deve essere sempre presente e rappresenta il fulcro del caso d'uso.

Attori Secondari (o di supporto): collaborano con l'attore primario nel raggiungimento dell'obiettivo. Sono attori che lavorano contemporaneamente all'attore primario sullo stesso sistema informativo e sullo stesso task, sono funzionali al raggiungimento dell'obiettivo dell'attore primario (sono un supporto).

Non sono necessariamente presenti, anzi sono relativamente rari in quanto i sistemi informativi moderni prevedono di essere utilizzati da una persona per volta.

Un obiettivo è un qualcosa di utile per l'attore, che può raggiungere tramite l'aiuto del sistema informativo; ovviamente deve trattarsi di qualcosa di valore per l'utente. Ad esempio, nel caso del bancomat, inserire il codice PIN non è qualcosa che abbia un valore per l'utente, mentre ritirare del contante è un obiettivo ragionevole in quanto ha un valore per l'attore.

Nel definire gli obiettivi è importante esplicitare come un tipo di attore voglia fare qualcosa per ottenere un certo valore per se stesso o per l'organizzazione. Una tecnica utile utilizza un template del tipo:

Come	*ruolo*
voglio poter	*fare qualcosa*
al fine di	*creare valore*

Con riferimento all'esempio del bancomat, si potrebbe esprimere un obiettivo come:

Come	cliente della banca
voglio poter	effettuare un prelievo
al fine di	ottenere dei contanti.

Nella definizione degli obiettivi è sempre necessario verificare che essi ricadano nell'ambito del sistema. Ci saranno obiettivi degli utenti che non sono soddisfacibili dal sistema (o dalla parte del sistema) che stiamo modellando: in tal caso, non dovranno essere rappresentati. Solo dopo aver capito quali obiettivi stanno dentro al sistema (ed avendo eliminato quelli che ne stanno fuori) si possono definire gli attori ed i relativi obiettivi.

Inoltre, per poter guidare la pianificazione delle attività di sviluppo è importante assegnare ad ogni obiettivo una priorità, come già detto nel capitolo precedente.

Per quanto riguarda il dettaglio del caso d'uso (la *narrazione*) si tratta di una serie di scambi di informazioni (tipicamente alternati, come in un "ping-pong") tra l'attore ed il sistema che si concludono con il raggiungimento dell'obiettivo (o con il fallimento del caso d'uso).

La descrizione del caso d'uso è una sequenza di passi, corrispondenti alle singole interazioni, che avvengono tra l'attore e il sistema. Lo scenario principale descritto è quello nominale, che rappresenta il caso in cui vada tutto per il verso giusto, ossia seguendo il percorso ottimale per il raggiungimento dell'obiettivo (terminazione con successo).

In aggiunta, è necessario poi descrivere cosa succede se qualcosa non va come previsto, valutando ed esplicitando ogni possibile causa di deviazione dal percorso principale, e definendo le azioni conseguenti a tale deviazione.

7.1.2 Contesto

Per poter descrivere i casi d'uso è importante definire con precisione quale sia il contesto coperto dal sistema. Si deve descrivere l'intero sistema? solo la sua componente software? solo alcuni moduli del suo software? Se non è noto il contesto, non sarà possibile definire gli obiettivi, né capire quali obiettivi siano "dentro" al sistema.

Un modo generale per classificare l'ambito di interesse è focalizzarsi su un singolo componente o su un singolo software per volta. A questo punto tutti gli altri componenti (eventualmente presenti nel sistema completo) sono rappresentati come attori che interagiscono con il sistema (in questo caso quindi gli attori non rappresentano utenti umani, bensì sistemi informativi o porzioni di essi). In base al contesto definito, quindi, si definiranno casi d'uso diversi.

Queste informazioni possono essere rappresentate sia in forma tabellare, sia tramite diagrammi dei casi d'uso UML (descritti nella successiva sezione 7.2).

7.2 Diagrammi dei casi d'uso

L'obiettivo dei diagrammi dei casi d'uso è di fornire una visione d'insieme del sistema in termini di attori che interagiscono con esso e dei loro obiettivi. Si ottiene quindi una descrizione di alto livello, priva di dettagli, che indica quali sono gli obiettivi senza fornire ulteriori informazioni su come questi possano essere raggiunti tramite interazioni con il sistema.

Per la rappresentazione grafica si utilizzano i Diagrammi dei Casi d'Uso (Use Case Diagrame, UCD) definiti nel linguaggio UML [3] (figura 7.2).

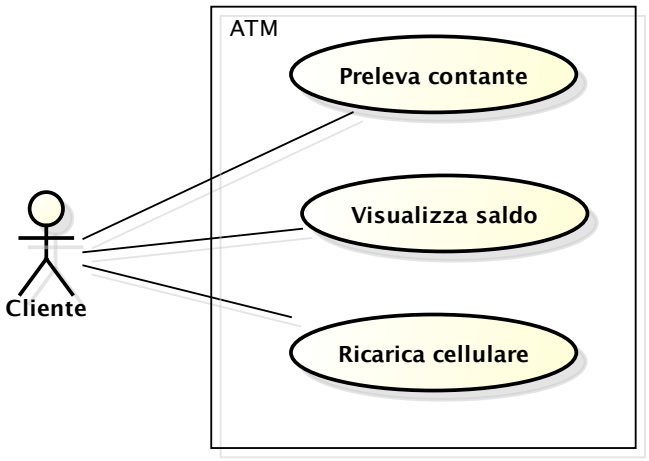

Figura 7.2: Esempio di Diagramma dei Casi d'Uso

Gli elementi grafici principali sono:

- gli attori, rappresentanti dal simbolo di essere umano stilizzato[1];
- gli obiettivi (o casi d'uso) che sono descritti con un ovale;

[1]Anche se descritto come un essere umano, è bene ricordare che è possibile che un attore sia un componente software o hardware.

- i legami tra attori e obiettivi, rappresentati come segmenti (o spezzate), che indicano quale/i attore/i partecipa/partecipano all'obiettivo (o caso d'uso);
- il sistema normalmente non viene rappresentato, si intende sottinteso; tuttavia, talvolta può essere indicato come un rettangolo che racchiude tutti i casi d'uso, in questo caso esplicitando meglio il contesto a cui ci riferiamo.

È fondamentale ricordare che gli obiettivi (o casi d'uso) non sono dei componenti del sistema software. In questo tipo di diagramma non sono mai rappresentati i componenti software[2]

7.2.1 Relazioni

Gli elementi del diagramma delle classi possono essere legati tra loro da relazioni. La più comune è quella di *partecipazione* che lega un attore al caso d'uso a cui partecipa. Tra i casi d'uso si possono definire relazioni di *inclusione* ed *estensione*. Tra gli attori può essere definita una relazione di *generalizzazione*.

Partecipazione

Il legame tra un attore ed un caso d'uso indica che l'attore prende parte al caso d'uso. La partecipazione dell'attore permette di raggiungere l'obiettivo rappresentato dal caso d'uso (figura 7.3).

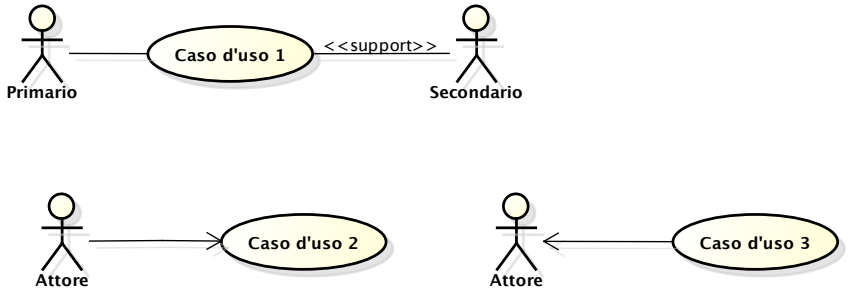

Figura 7.3: Relazioni di partecipazione

Il ruolo dell'attore può essere diverso:

- l'attore *primario* è connesso tramite un semplice segmento non annotato
- un attore *secondario* (o di *supporto*), ove presente, è collegato al caso d'uso tramite un segmento che riporta lo stereotipo «`support`» (caso d'uso 1 in figura 7.3).

Il coinvolgimento dell'attore nel caso d'uso può avvenire:

- su iniziativa dell'attore (primario), questo caso è quello implicito, ma può essere indicato esplicitamente con una freccia in direzione del caso d'uso (caso d'uso 2 in figura 7.3);
- su iniziativa del sistema, questo caso può essere esplicitato indicando un freccia in direzione dell'attore (caso d'uso 3 in figura 7.3).

[2]Un errore molto grave è quello di includere componenti software in un diagramma dei casi d'uso; ad esempio un diagramma che riporta un ovale con l'indicazione *database* non ha nessun significato, il database non è evidentemente un obiettivo da raggiungere.

Inclusione

Un possibile legame fra due casi d'uso è l'inclusione. Dire che il caso d'uso `A` include il caso d'uso `B` significa che `B` è un sotto-obiettivo di `A`. Si tratta di una sotto-funzione che tipicamente non rappresenta un vero obiettivo con un effettivo valore per l'attore.

Per rappresentare l'inclusione (figura 7.4) si utilizza una freccia tratteggiata che parte dal caso d'uso che include, verso il del caso incluso, accando al segmento viene riportato lo stereotipo «`include`».

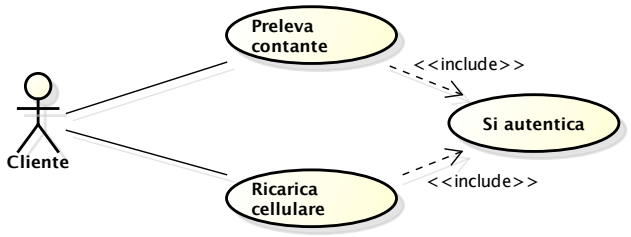

Figura 7.4: Inclusione tra casi d'uso

Normalmente il costrutto dell'inclusione si utilizza quando si vogliono specificare una serie di passaggi che sono parte del caso d'uso principale (ad esempio `Preleva Contante`) ma si possono ritrovare anche in altri casi d'uso (es. `Ricarica cellulare`). Il caso d'uso incluso (es. `Si autentica`) rappresenta solo una fase del caso d'uso principale e di conseguenza non rappresenta un obiettivo vero e proprio per l'attore.

Estensione

Quando un caso d'uso `B` costituisce il proseguimento (opzionale) di un altro caso d'uso `A`, si dice che `B` estende `A`.

Per rappresentare l'estensione (figura 7.5) si utilizza una freccia tratteggiata che parte dal caso d'uso che estende, verso il caso che viene esteso, ed accanto al segmento viene riportato lo stereotipo «`extend`».

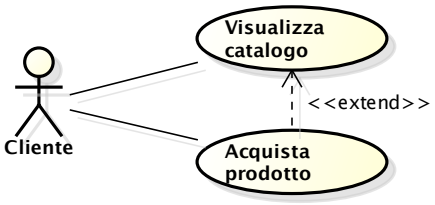

Figura 7.5: Inclusione tra casi

Quando due casi d'uso distinti sono legati da una relazione di estensione, ciascuno di essi può rappresentare un obiettivo per l'attore. L'attore può limitarsi al primo caso d'uso (nell'esempio in figura 7.5, l'utente che ha come obiettivo la ricerca del prodotto può trovarsi ad essere soddisfatto della ricerca e fermarsi) oppure voler raggiungere anche l'obiettivo rappresentato dal secondo caso d'uso (l'utente può decidere di procedere con l'acquisto di un prodotto risultato della ricerca). In questa seconda ipotesi, l'obiettivo dell'attore è potenzialmente cambiato in itinere (da ricerca ad acquisto), non si tratta di un caso d'uso diverso ma dell'estensione del caso d'uso iniziale. Tuttavia l'obiettivo poteva essere, già in partenza, il caso d'uso estensore, che di conseguenza è rappresentato da un caso d'uso a sé stante.

Generalizzazione

Talvolta due attori rappresentano due tipologie di utenti, una più generica ed una più specializzata: l'attore più specifico ha tutti gli obiettivi di quello più generale più altri a lui riservati. Tale relazione si rappresenta con la relazione di generalizzazione[3].

La generalizzazione viene rappresentata (figura 7.6) con una freccia la cui punta è un triangolo, che va va dall'attore più specifico a quello dello più generico. La notazione è la stessa utilizzata per la generalizzazione nel diagramma delle classi (si veda §**??**).

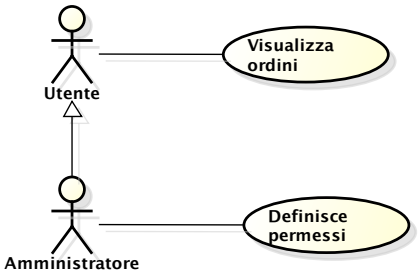

Figura 7.6: Inclusione tra casi

Si noti che tutto ciò che può fare l'attore generico (es. Utente) lo può fare anche quello specifico (es. Amministratore), ma quest'ultimo ha degli obiettivi aggiuntivi. La raffigurazione tipica vede l'attore generico in alto e quello specifico in basso, questa è in contrasto con la visualizzazione della gerarchia di un organigramma, dove i ruoli più in alto hanno maggiori prerogative e privilegi.

7.2.2 Livelli di dettaglio (Granularità)

Qual è il livello giusto di dettaglio per i casi d'uso? In generale possiamo identificare tre livelli principali:

Summary è il livello più generale e ampio, e permette di raggruppare più casi d'uso di livello user-goal in un insieme di obiettivi tra di loro correlati. Descrive una serie di azioni correlate tra loro, un po' come se fossero l'indice di tutti i casi d'uso. Servono come uno strumento di aggregazione dei casi d'uso di livello user-goal.

I casi d'uso di livello summary vengono contrassegnati dallo stereotipo «`summary`».

Per capire quando un caso d'uso è di livello summary, possiamo chiederci: cosa vogliono fare in generale gli attori? Es: gestione esami, gestione studenti, gestione anagrafica. Sono delle funzioni generali che sappiamo che poi andranno scomposte in azioni più dettagliate, ma che di per sé hanno già un obiettivo.

Normalmente, alla domanda "perché l'utente fa questo?" è possibile trovare dei sotto-obiettivi più concreti e di valore.

Quando si descrive la narrativa (§7.3) di un caso d'uso di livello summary non andremo a specificare tutti i dettagli, ma solamente quelli più generali.

User-goal sono quelli fondamentali che ci interessa descrivere. Corrispondono ad una effettiva esigenza dell'attore.

Graficamente, questi casi d'uso possono essere opzionalmente contrassegnati dallo stereotipo «`user-goal`», che può essere omesso se non esiste ambiguità sul livello di granularità.

[3] È possibile anche avere una generalizzazione tra casi d'uso, che ha una semantica complicata e per tale motivo non è trattata in questa sede

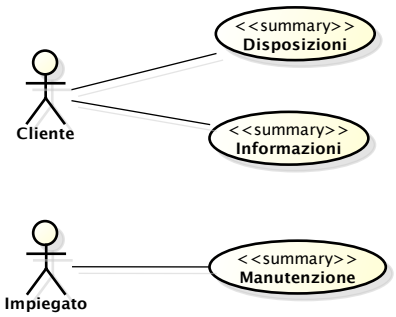

Figura 7.7: Esempio: UCD Bancomat livello summary

Per capire la giusta granularità, la regola che possiamo seguire per identificare questo tipo di livello è che normalmente questi vengono svolti da una persona in un unico luogo e in un lasso di tempo ridotto (dell'ordine dei minuti).

Se il caso d'uso è invece di pochi secondi probabilmente sarà di livello sub-function.

Per capire quando un caso d'uso è di livello user-goal, possiamo chiederci: "cosa permette di fare il sistema agli attori?". Inoltre alla domanda "perché l'utente fa questo?" la risposta dovrebbe essere "perché per lui questo ha valore".

Sub-function: questo livello fornisce supporto ad attività utilizzate nei casi d'uso di livello user goal. Qualora sia necessario entrare più nel dettaglio su come viene realizzata una certa funzione si utilizza un caso d'uso di questo livello.

I casi d'uso di livello sub-function si indicano tramite o *stereotipo* «`sub-function`».

Rappresentano i dettagli dei casi d'uso di livello user-goal. Possono essere utilizzati per indicare una volta sola una (sequenza di) sotto-attività che sono incluse in più casi d'uso di livello user-goal.

Normalmente i casi d'uso di livello sub-function vengono inclusi (attraverso la relazione «`include`») da quelli di livello user-goal.

Generalmente, se non viene fornita alcuna indicazione esplicita, si suppone che il caso d'uso sia di *livello user-goal*.

Un caso d'uso di livello summary permette di raggruppare più casi d'uso di livello user-goal in un insieme di obiettivi tra di loro correlati. Permette di distinguere fra informazioni dispositive e informazioni di visualizzazione. Per definire il summary level utilizzeremo un diagramma a parte: gli attori che compariranno saranno gli stessi, gli obiettivi saranno invece *raggruppati* e rappresentati in maniera più aggregata.

Per capire quale sia la giusta granularità, dobbiamo immaginare l'arco temporale necessario per svolgere ogni azione:

- Lo user-goal è svolto da una persona (attore principale), in un luogo determinato ad un'ora determinata, in un lasso di tempo relativamente breve (decine di secondi o alcuni minuti).
- Se il caso d'uso è estremamente piccolo (es: pochi millisecondi o pochi secondi, tipo l'autenticazione), probabilmente è una sub-function (funge da supporto all'obiettivo principale).
- Se infine l'obiettivo è molto complesso e articolato (insieme di operazioni), probabilmente si tratta di un summary level.

Consideriamo la seguente situazione:

> *Giovanna si trova di fronte al bancomat, la sera. È buio, lei ha appena inserito il PIN e sta cercando il bottone Invio.*

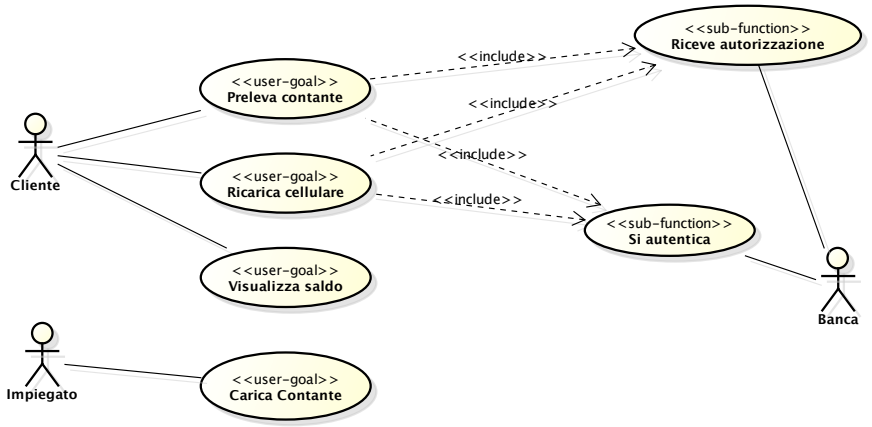

Figura 7.8: Esempio: UCD Bancomat

Con riferimento al sistema Bancomat, possiamo identificare tre casi d'uso di diverso livello:

1. Livello summary: Giovanna vuole usare il bancomat
2. Livello user-goal: Giovanna vuole ottenere contante dal bancomat
3. Livello sub-function: Giovanna vuole effettuare l'autenticazione.

Un ulteriore livello di dettaglio potrebbe essere quello di trovare il tasto di invio. Chiaramente questo è troppo specifico e riguarda un dettaglio specifico dell'interfaccia che prevede un pulsante con l'etichetta "Invio". Tale livello di dettaglio è sicuramente fuori dallo scopo dei casi d'uso.

D'altro lato, un livello di dettaglio ancora più ampio potrebbe essere che Giovanna vuole andare a cena con gli amici. Questo è un livello errato perchè fa riferimento ad obiettivi che non sono direttamente perseguibili tramite il sistema che si sta descrivendo.

Esempio

Consideriamo l'esempio del Bancomat per capire come poter definire un diagramma dei casi d'uso.

Supponiamo di avere identificato due utenti che dovranno interagire con il sistema: il `Cliente` che utilizzerà il bancomat e l'`Impiegato` che è responsabile della sua operatività.

Occorre chiedersi quali siano gli obiettivi di ciascun attore. Il cliente può avere la scelta fra ricaricare il cellulare, visualizzare il saldo o prelevare del denaro. L'impiegato invece si occupa di ricaricare il bancomat. Tuttavia prima di ogni operazione il cliente si deve autenticare, questa operazione, che di per se non è un obiettivo per il cliente, rappresenta tuttavia un passaggio fisso per poter raggiungere gli obiettivi principali. L'autenticazione è un sotto-obiettivo. Un discorso analogo riguarda l'autorizzazione ad effettuare il prelievo o a ricaricare il cellulare.

Sia l'autenticazione che l'autorizzazione non sono legati al cliente, in quanto non rappresentano obiettivi che possono essere perseguiti dall'attore indipendentemente, hanno senso solo se interpretati come fasi all'interno di altri casi d'uso di livello superiore.

Queste due sotto-funzioni avvengono appoggiandosi alla banca, che è l'ultima responsabile per le azioni. Di fatto si tratta del sistema informativo centrale della banca che effettuerà l'autenticazione e l'autorizzazione.

L'autenticazione, come spesso succede per le sub-function, è un obiettivo che non è legato a nessun attore, non c'è nessuno che lo persegue! Sarà invece uno dei passaggi obbligati per l'attore Cliente quanto eseguirà uno dei casi d'uso a livello user-goal.

Consideriamo altresì che l'autenticazione non può essere compiuta in autonomia dal bancomat, ma è necessaria la conferma da parte dell'istituto bancario. Quindi inseriamo un attore che si occupi di ciò: la banca (attore di tipo sistema informativo, e non di tipo umano). Quest'ultima si occupa inoltre di autorizzare l'operazione di ricarica o quella di prelievo.

La banca, in questo caso, è un attore secondario (di supporto). Essa non persegue davvero un obiettivo, ma piuttosto aiuta il cliente a raggiungere i suoi tramite le operazioni di autenticazione e autorizzazione.

Il sistema è un'entità onnipresente sullo sfondo, non è mai un attore. Gli attori sono per definizione enti esterni al sistema: commettere questo errore, è estremamente grave. Talvolta il sistema può essere rappresentato come un rettangolo che racchiude l'insieme di tutti i casi d'uso.

7.3 Narrativa di un Caso d'Uso

Quando osserviamo un diagramma dei casi d'uso, per ciascun caso d'uso sappiamo che vi è un'interazione con un dato obiettivo, ma non sappiamo come avviene.

È necessario fare un ulteriore passo e descrivere come l'interazione ha luogo, ovvero descrivere la *narrativa* del caso d'uso.

Per fare questo utilizziamo una struttura ben definita: il Template SWEED, che è una versione leggermente semplificata del template "fully dressed" definito da Cockburn [2].

7.3.1 Template

Il template è una struttura puramente testuale costituito da una serie di sezioni (quelle contrassegnate da '(*)' sono opzionali) tra le quali una include la sequenza numerata delle interazioni tra attore e sistema.

Use case (Nome del caso d'uso): corrisponde all'etichetta del caso d'uso utilizzata nel diagramma dei casi d'uso. Di fatto esprime sinteticamente l'obiettivo.

Scope (Contesto): molto spesso il contesto è il sistema che stiamo descrivendo, od un suo componente (nel caso di sistemi più grandi).

Level (Livello): summary, user-goal, o sub-function. Sottinteso: user-goal.

Intention in context (Intenzione dell'attore): descrive nel dettaglio qual'è l'intenzione dell'attore primario nell'usare il sistema.

Primary actor (Attore primario): l'attore primario, colui che vuole raggiungere un obiettivo.

Support actors (*) (Attori di supporto): gli eventuali attori secondari che possono coadiuvare l'attore primario nel raggiungimento del suo obiettivo.

Stakeholders' interests (*) (Interessi degli stakeholder): esplicita quali interessi (relativi ad uno o più stakeholder aggiuntivi rispetto all'attore primario) il sistema deve salvaguardare nel raggiungere l'obiettivo.

Es.: per il caso bancomat, lo stakeholder *amministratore della banca* ha interesse che la somma erogata non superi il saldo attuale sul conto del cliente.

Preconditions (*) (Pre-condizioni): condizioni che devono essere verificate perché possa svolgersi questo caso d'uso; è una guardia o blocco all'ingresso. Se le precondizioni non sono verificate, questo caso d'uso non potrà essere eseguite. Viceversa, all'interno del caso d'uso, si può avere la certezza assoluta che tutte le precondizioni sono verificate.

Es.: per la *segreteria studenti*, lo studente può sostenere l'esame solo se è presente nella lista dei prenotati.

In particolare, vi sono due tipologie di pre-condizioni che sono sempre presenti, e pertanto possiamo considerare come implicite:

1. le condizioni di dipendenza già espresse nell'activity diagram. Se l'activity diagram specifica che l'azione A precede l'azione B, allora il caso d'uso relativo a B avrà come precondizione il completamento di A. Queste precondizioni sono sempre presenti dal punto di vista logico, ma non è necessario rappresentarle nel caso d'uso.
2. l'identità dell'attore. Esiste sempre la pre-condizione per cui l'utente connesso al sistema sia riconosciuto. Consideriamo questa pre-condizione come implicita nel fatto che il caso d'uso è connesso ad un attore ben definito, per cui diamo per scontato che il sistema informativo abbia già verificato che l'utente ha i privilegi corrispondenti. Di conseguenza, azioni come "l'utente fa il login" o simili non saranno mai inserite nel caso d'uso, in quando presumiamo che siano già state compiute in precedenza[4].

Minimum guarantees (*) (Garanzie minime): condizioni che sono sempre vere in qualsiasi circostanza, sia che il caso d'uso termini con un successo sia che fallisca.

Success guarantees (*) (Garanzie di successo): condizioni che sono garantite se il caso d'uso si completa con successo.

Trigger (*) (Causa scatenante): rappresenta una condizione o evento esterno che conduce all'attivazione del caso d'uso. Se è presente il caso d'uso si attiva al verificarsi della condizione, invece che su iniziativa di un attore.

Main success scenario (Scenario di successo principale): è costituito da un elenco numerato di passi che descrivono le interazioni tra il sistema e l'attore – o gli attori se ne esistono di supporto –. Lo scenario principale descrive il caso nominale in cui l'attore primario inizia l'interazione con un obiettivo che raggiunge tramite l'aiuto del sistema. Rappresenta una tra le possibili serie di interazioni (*scenario*) che permettono all'attore di raggiungere il proprio obiettivo (*success*), in particolare è quello più significativo o importante (*main*).

Extensions (Estensioni): rappresentano le possibili alternative al main success scenario, che descrivono come raggiungere l'obiettivo in altro modo e come gestire errori ed anomalie.

7.3.2 Main Success Scenario

Il fulcro della narrativa di un caso d'uso è il *Main Success Scenario* (MSS), che ha le seguenti caratteristiche essenziali:

- i passi descrivono le interazioni tra un attore e il sistema;
- un'interazione consiste in un flusso di informazioni o comandi da un lato verso l'altro;
- gli scambi di informazione avvengono, alternativamente, su iniziativa di un attore o del sistema;
- il primo passo è solitamente[5] compiuto dell'attore che attiva il caso d'uso;
- i passi sono numerati in maniera progressiva.

[4] Fanno eccezione quei casi d'uso che permettono di "cambiare" i privilegi dell'utente, come Login, Logout, Diventa Amministratore, ecc.

[5] a meno che il caso d'uso prenda avvio per un trigger

È ragionevole pensare che il sistema esegua più operazioni di quelle descritte nel caso d'uso, ma per il MSS è rilevante solo cio' che si manifesta come interazione con l'attore. Si escludono quindi tutte le operazioni di ricerca e calcolo che il sistema fa senza dare riscontro all'utente.

Le azioni svolte dal sistema sono di due tipi:

1. Esegue delle elaborazioni e mostra all'utente i risultati
2. Analizza i dati e ne conferma la correttezza

I singoli passi di interazioni consistono in descrizioni semplici, strutturate come:

```
<Soggetto> <Verbo> <Complemento oggetto>
```

È importante esplicitare, nell'interazione, chi "*sta colpendo la palla*" nel ping pong tra sistema e attore. Ovviamente nessuno dei due può colpire la palla 2 volte di seguito[6]. I passi sono sempre alternati: l'attore fa A e il sistema risponde B.

In genere, è meglio mantenere un certo livello di astrazione. Non si indica la pressione di un pulsante, se mai il significato dell'azione svolta (completo, procedo...) Lo stile da utilizzare è quello che *mostra l'intento, non il movimento*.

Infine, le azioni devono essere presenti in numero limitato (come indicazione generale, da 3 a 9).

Esempio di Main Success Scenario

> *Mary, taking her two daughters to the day care on the way to work, drives up to the ATM, runs her card across the card reader, enters her PIN code, selects FAST CASH, and enters $35 as the amount. The ATM issues a $20 and three $5 bills, plus a receipt showing her account balance after the $35 is debited. The ATM resets its screens after each transaction with FAST CASH, so that Mary can drive away and not worry that the next driver will have access to her account.*

Nell'esercizio ci sono due alternative a disposizione dell'utente per prelevare i soldi:

1. Fast cash
2. Full cash

Fino a quando l'utente non selezione, il sistema informativo non sa cosa vuole l'attore. I primi passi saranno uguali per i diversi casi d'uso. Una possibile descrizione dell'interazione potrebbe essere la seguente:

1. L'utente passa la carta sul lettore
2. Il SI (bancomat) legge il codice e verifica la validità (senza questo passaggio non si potrebbe andare avanti)
3. Il cliente inserisce il codice PIN
4. Il SI verifica il codice e lo convalida
5. L'utente seleziona la modalità di prelievo *Fast cash* e l'importo deve essere un multiplo di 5 euro
6. Il SI notifica alla banca centrale del cliente l'ammontare del prelievo
7. Il cliente ritira i soldi

[6] Salvo eccezioni, dovute ad esempio all'inattività (time-out) dell'utente

7.3.3 Extensions

Mentre il main success scenario descrive il caso ideale, nella realtà è possibile che le cose non vadano sempre come ci si aspetta, generalmente a causa di errori dell'utente, malfunzionamenti, o percorsi alternativi.

Dato un obiettivo, è possibile raggiungerlo (success) oppure no (failure). Il sucesso è raggiungibile tramite lo scenario principale, oppure tramite uno più percorsi alternativi che comunque portano a raggiungere l'obiettivo.

I singoli passi dello scenario possono essere visti come una serie di sotto-obiettivi, come indicato schematicamente in figura 7.9: se tutti vanno a buon fine si realizza il main succcess scenario, se uno o più falliscono è possibile comunque ottenere il successo per vie alternative, oppure il fallimento.

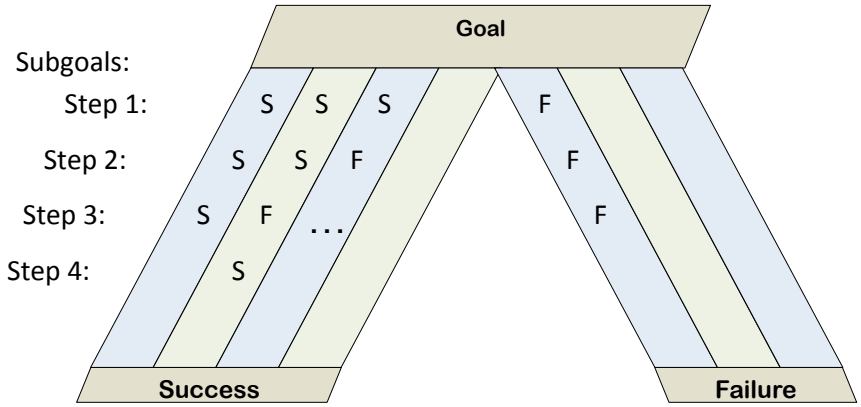

Figura 7.9: Possibili scenari alternativi in un caso d'uso

In questo contesto, il main success scenario costituisce una delle tante "strisce" a sinistra. Le estensioni servono per completare il template e descrivere le alternative (tutte le altre "strisce" a destra e a sinistra).

Le azioni delle estensioni sono specificate come:

```
<Condizione> :  <Azione>
```

- Si descrivono solo condizioni che sono osservabili dal sistema. Non siamo interessati alla causa prima, ma all'anomalia osservabile dal sistema in base alle informazioni in suo possesso.

  Ad esempio, nell'istante in cui il Bancomat richiede inserimento di PIN possiamo immaginare due modi di scrivere le condizioni:

  - "Si raggiunge tempo di timeout sull'inserimento del PIN": questa condizione è corretta perchè può essere verificata direttamente dal sistema.
  - "L'utente ha dimenticato il PIN": questa condizione è errata perchè non si riferisce a fatti o informazioni conoscibili dal sistema.

- Le azioni definiscono cosa deve fare il sistema informativo in caso di fallimento del passo del percorso principale. Il sistema deve essere in grado di gestire anche le anomalie.

- Le estensioni fanno sempre riferimento ad un passo del main success scenario. In particolare, possono essere in alternativa al percorso principale (1.a: faccio 1.a anziché fare 1) oppure in aggiunta ad esso (1+, faccio 1+ dopo avere fatto 1).

Per identificare le consizioni, è necessario domandarsi cosa può succedere che porti a successo tramite vie alternative oppure a fallimento. Si noti che tutto è fatto dal punto di vista del sistema informativo.

Per identificare le condizioni è possibile utilizzare la seguente checklist:

- Percorsi alternativi di successo
- Comportamento errato dell'attore primario
- Inattività dell'attore primario
- Esito negativo di una convalida
- Malfunzionamento interno (prevedibile)
- Malfunzionamento inatteso o anormale
- Malfunzionamento prestazionale critico.

Vi sono alcuni fallimenti che possono essere prevedibili a priori. Ad esempio, il bancomat deve stampare una ricevuta. Il fallimento perché finisce carta oppure inchiostro è del tutto prevedibile. Se finisce l'inchiostro oppure la carta, si prevede uno scenario alternativo che porti al successo.

Vi sono poi alcuni fallimenti non prevedibili, che saranno allora di tipo recuperabile oppure irrecuperabile.

Altra tipologia di fallimento che consideriamo è quella di performance critica.

Esempio di Extension

Con riferimento all'esempio del fast-cash (il cui main success scenario è riportato nella sezione 7.3.2), in ciascuno dei questi punti del main success scenario ci possono essere delle anomalie come ad esempio, la carta che non è valida, l'impossibilità di leggere il codice, il cliente che non si ricorda più il pin..etc.

I cammini con cui si può raggiungere l'obiettivo sono diversi. Per raggiungere l' obiettivo non è necessario che tutte le azioni avvengano con successo, è anche possibile giungere all'obiettivo quando un'azione magari fallisce ma il fallimento di cui è vittima è recuperabile.

La costruzione di cammini alternativi vede per ciascun passo del cammino principale la valutazione delle possibili alternative. Si descrive solo ciò che è osservabile dall'interazione e quindi rilevabile dal sistema informativo.

Successivamente si decide il comportamento che il sistema deve avere quando si verificano queste condizioni modalità di gestione

Facendo riferimento all'esercizio sopra avremo ad esempio:

Anomalia passo 4 :

4a. Il sistema informativo non convalida il PIN: torna al passo 3 (dove il SI chiede nuovamente di inserire il PIN) dando la possibilità al cliente di reinserire il PIN

Anomalia passo 6 :

6a. il cliente non ha abbastanza credito: torna al punto 5

6b. l'ammontare non è multiplo di 5 euro:

6b.1. arrotonda al multiplo più vicino (correzione interna dell'anomalia)

6b.2. torna al passo 6

Anomalia passo 3 :

3a. il cliente non inserisce il PIN nel tempo permesso:

3a.1. Il SI restituisce la carta

3b.1. il caso d'uso termina con un fallimento

7.3.4 Trigger

Si tratta di una condizione (ad esempio un evento temporale o un evento esterno) che "scatena" il caso d'uso. Ad esempio, quando nel caso di un'autoradio a bordo di un'automobile durante l'ascolto della musica arriva una telefonata.

I trigger sono delle condizioni che, non appena si verificano, danno il via al caso d'uso. Un esempio classico di trigger è la scadenza dopo 5 minuti, o la ricezione di una telefonata.

In questo caso (rappresentato nel diagramma dei casi d'uso con una freccia orientata che va dal caso d'uso verso l'attore), il caso d'uso viene iniziato dal sistema, e l'attore verrà coinvolto "forzatamente" a continuare l'interazione. Il passo 1. dello scenario principale, in presenza di trigger, sta quindi al sistema informativo (e non all'utente, come avviene in assenza di trigger).

7.3.5 Processo di scrittura di un caso d'uso

In sintesi, per definire un caso d'uso, è necessario:

1. Definire lo scope
2. Definire gli attori
3. Definire i loro obiettivi (goal), che saranno i diversi casi d'uso

   Per ogni goal è poi necessario:
4. Scrivere un Brief (da 3 a 9 frasi), oppure
5. Scrivere il Main success scenario
6. Identificare i possibili fallimenti
7. Gestire tutti i possibili fallimenti

Errori tipici nella definizione di casi d'uso

- Cercare a definire caso d'uso senza avere chiaro il confine del sistema (o scope), oppure cambiarlo in corso d'opera.
- Usare il punto di vista di un attore anziché quello del sistema (rappresentando ad esempio azioni compiute dall'attore ma non visibili al sistema), oppure di identificare troppi attori.
- Definizione di uno scenario principale troppo lungo. Il caso d'uso nasce come strumento conciso e puntuale, se lo rendiamo troppo lungo stiamo perdendo di vista la ragione principale per la quale scriviamo casi d'uso. Per questo motivo i casi d'uso vanno bene per descrivere un'interazione, non un algoritmo. Una regola prevede di avere un numero di passi compreso tra 3 e 9.
- Considerare un attore nel caso d'uso, quando però l'attore non è autorizzato ad interagire o a svolgere determinate operazioni.
- Il caso d'uso inoltre deve portare a qualcosa, non può non terminare.
- Inserire troppi dettagli, che renderanno fragile il caso d'uso (appena qualcosa cambia nel SI il caso d'uso non sarà più valido).

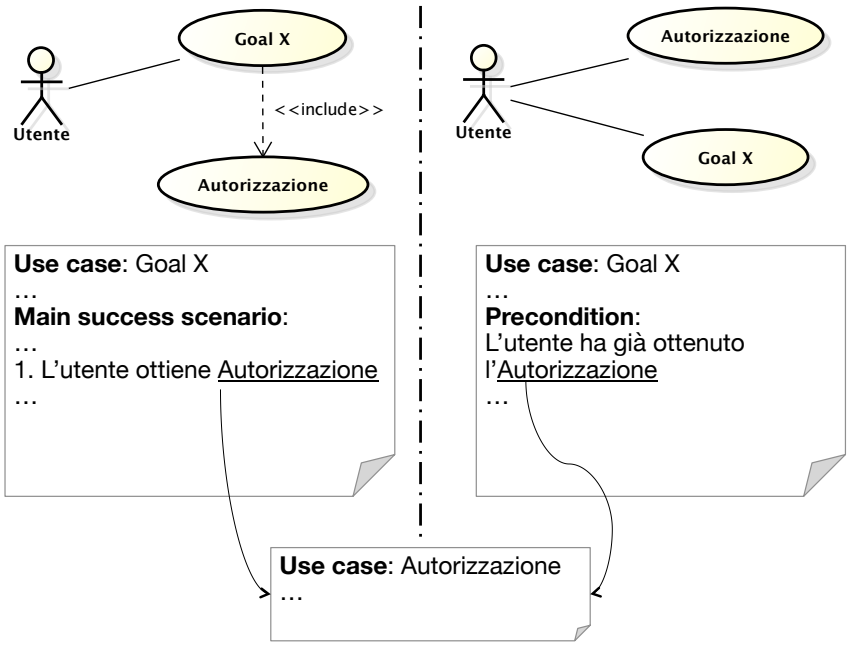

Figura 7.10: Precondizioni vs. Inclusione

Inclusione vs. Precondizione

La relazione di inclusione tra casi d'uso e le precondizioni possono avere un uso simile. Esistono tuttavia delle sottili differenze.

Consideriamo il caso mostrato in figure 7.10: prima di poter procedere con il caso d'uso relativo al proprio obiettivo *X* l'utente deve ottenere un'autorizzazione. Questa condizione può essere descritta nei due modi illustrati.

1. Il caso d'uso *Goal X* include il caso d'uso *Autorizzazione* (diagramma a sinistra), quindi nei primi passi del primo caso d'uso compare un riferimento al secondo.

   In questo caso la dipendenza è esplicitata già nel diagramma dei casi d'uso e significa che ogni volta che un l'utente vuole raggiungere il proprio obiettivo deve richiedere un'autorizzazione.

2. Il caso d'uso *Goal X* è indipendente dal caso d'uso *Autorizzazione* (diagramma a destra), tuttavia nelle pre-condizioni del primo compare l'obbligo di avere già richiesto l'autorizzazione.

   In questo caso la dipendenza non è visibile a livello di diagramma dei casi d'uso, inoltre questo implica che l'autorizzazione può essere chiesta una sola volta anche per più esecuzioni del caso d'uso *Goal X*.

   Questo secondo caso d'uso è compatibile con un diagramma di attività in cui un'azione *Autorizzazione* precede un'azione *X*, in tal caso la precondizione corrisponde al vincolo di precedenza presente nel diagramma di attività.

Autenticazione

Virtualmente in ogni sistema informativo è necessario che gli utenti si autentichino prima di poter accedere a qualunque funzionalità.

Consideriamo un esempio (illustrato in figura 7.11) in cui tre attori (Amministratore, Manager e Impiegato) possono utilizzare il sistema per scopi diversi.

Tuttavia, qualsiasi operazione che essi vogliano svolgere necessita prima di un'autenticazione. Ma, prima dell'autenticazione, il sistema non può stabilire quale tipo di utente

sia quello collegato. Perciò è necessario introdurre un ulteriore attore generico, "Utente", che ha come solo obiettivo di autenticarsi per poi diventare uno degli attori noti al sistema. Quindi Amministratore, Manager e Impiegato sono una specializzazione di Utente.

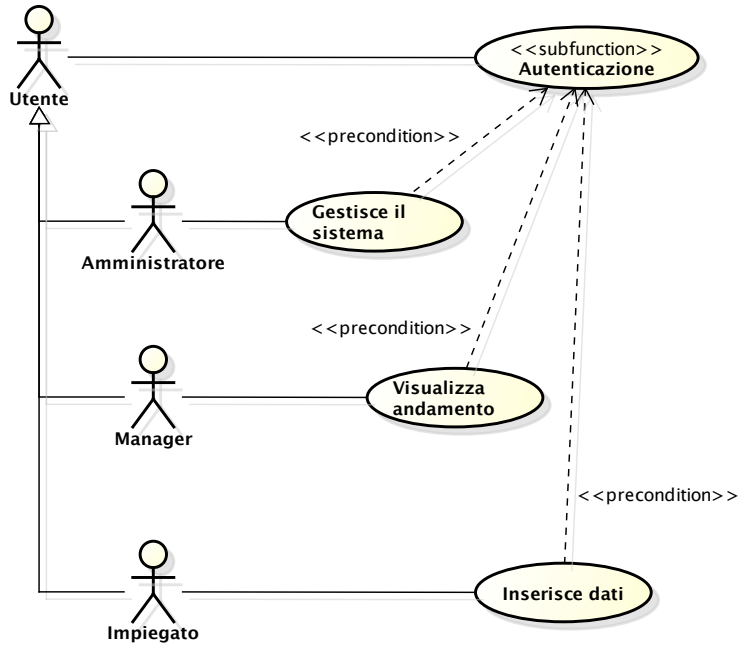

Figura 7.11: UCD completo per l'autenticazione

Questo tipo di configurazione ha diversi svantaggi: in ogni sistema è necessario introdurre questo nuovo attore che non ha alcun vero obiettivo, per di più l'unico obiettivo di tale attore non è un caso d'uso di livello user-goal, come dovrebbe essere, ma bensì solamente un caso d'uso di livello sub-function.

Per semplicità quindi si assume che *implicitamente* tutti gli attori siano già stati autenticate e che non sia necessario indicare esplicitamente l'autenticazione.

7.4 Esempio - Lista d'attesa al ristorante

Si consideri il seguente caso di studio:

> ***Gestione lista di attesa di un ristorante***
> *Un noto ristorante all'interno di un centro commerciale, utilizza un sistema per la gestione della lista di attesa.*
> *Quando un cliente arriva, il cameriere, una volta noto il numero di persone nella comitiva, verifica sul sistema il tempo di attesa stimato per avere un tavolo. Quindi il cliente fornisce nome e numero di telefono cellulare, il sistema invia un SMS di conferma al telefono del cliente. A questo punto il cliente ha il tavolo prenotato.*
> *Il cliente può quindi andare via, ad esempio a visitare i negozi del centro commerciale. Ogni cinque minuti il sistema invierà un SMS con il tempo residuo di attesa stimato.*
> *Quando il tavolo prenotato per il cliente si rende disponibile, un cameriere lo comunica al sistema che invia un SMS al cliente chiedendo di recarsi al ristorante non appena possibile.*
> *Quando il cliente si ripresenta al ristorante viene fatto accomodare al tavolo prenotato. Se il cliente non si presenta entro un tempo prefissato, il sistema richiama il prossimo cliente nella lista di attesa.*

7.4.1 Identificazione degli attori

Quali sono gli utilizzatori principali del sistema? Nel caso specifico si tratta del *Cameriere*, che costituisce quindi l'attore principale.

Occorre capire se esistono altri attori che interagiscono con il sistema. Il cliente non interagisce direttamente con il sistema ma bensì con il cameriere. Quindi il cliente non deve essere annoverato tra gli attori.

Ci sono tuttavia un paio di passaggi in cui il cliente potrebbe sembrare interagire con il sistema, ad esempio quando il cliente riceve il SMS di tempo residuo, quello che lo prega di recarsi al ristorante, o ancora il timeout. Si tratta tuttavia di un'interazione sui generis, il cliente riceve dei messaggi ma, in base al testo descrittivo, non risponde al sistema.

Generalmente, per inviare messaggi SMS automatici si possono usare due metodi:

1. Collego una SIM al sistema
2. Mi appoggio ad un provider per inviare SMS automatici tramite un SMS Gateway

Poiché il sistema in questione è per un noto ristorante, è plausibile supporre che si serva di un SMS Gateway (es. Vodafone, Tim, 3...) per inviare i messaggi. Quindi SMS Gateway sarà un attore di supporto per tutte le operazioni di invio SMS.

In conclusione avremo due attori:

- Il Cameriere, attore principale;
- L'SMS Gateway, attore di supporto.

7.4.2 Diagramma dei casi d'uso

Il contesto del sistema può essere descritto tramite un diagramma dei casi d'uso (figura 7.12) in cui compaiono gli attori ed i loro obiettivi.

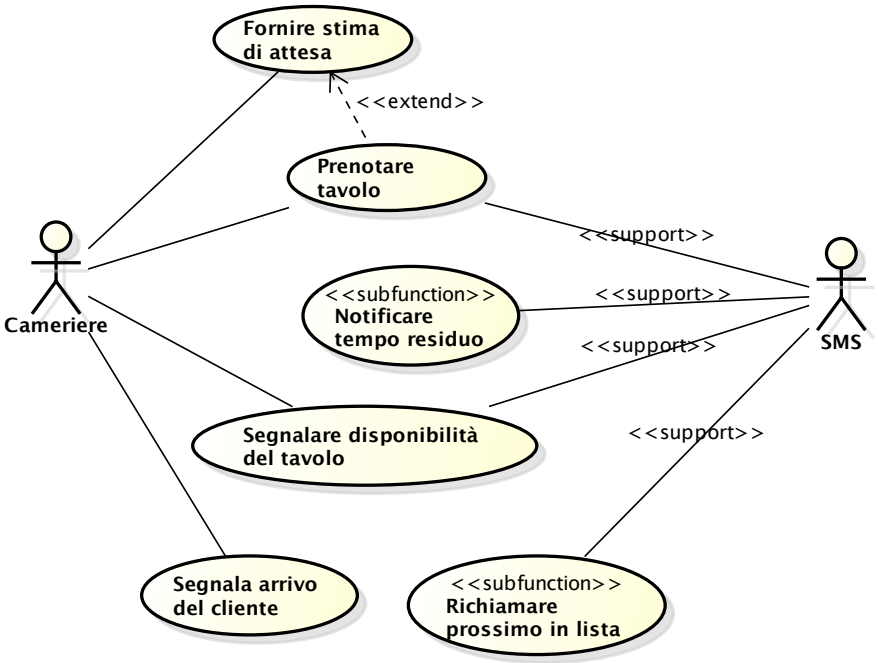

Figura 7.12: Gestione della lista d'attesa: UCD.

Sono obiettivi di livello user-goal per l'attore Cameriere:

- fornire una stima del tempo di attesa,

- prenotare un tavolo,
- segnalare che un tavolo è libero (e il conseguente richiamo di un cliente prenotato),
- segnalare l'arrivo di un cliente prenotato (e la conseguente occupazione del tavolo prenotato).

Esistono altre attività del sistema che richiedono un'interazione (sebbene limitata come il semplice invio di un SMS):

- notificare il tempo residuo di attesa ai clienti,
- segnalare la disponibilità del tavolo prenotato,
- richiamare il prossimo cliente se uno precedentemente richiamato non si presenta,

Questi obiettivi non coinvolgono l'attore principale e non sono chiaramente di valore, sono dei sotto-obiettivi necessari a poter gestire la lista delle prenotazioni. Perciò li rappresentiamo come use-case con lo stereotipo subfunction.

7.4.3 Narrative dei casi d'uso

Use case: Fornire stima di attesa

Scope: Sistema di gestione liste d'attesa – Sottosistema di Sala

Level: user-goal

Intention in context: fornire al cliente una stima del tempo di attesa

Primary actor: Cameriere

Support actor:

Stakeholders' interests:
Cliente: ottenere una stima quanto più corretta del tempo di attesa

Preconditions: -

Minimum guarantees:

Success guarantees:

Main success scenario:
1. il cameriere richiede una stima
2. il sistema chiede il numero di persone
3. il cameriere inserisce il numero di persone
4. il sistema restituisce una stima del tempo di attesa

Extensions:
4.a. non esistono tavoli disponibili:
    4.a.1. il sistema segnala il problema
    4.a.2. il caso d'uso termina con un fallimento
5. il cameriere richiede la prenotazione: il caso d'uso continua con il caso d'uso *Esegue Prenotazione*

Use case: Esegue Prenotazione

Scope: Sistema di gestione liste d'attesa – Sottosistema di Sala

Level: user-goal

Intention in context: inserire il cliente in lista d'attesa

Primary actor: cameriere

Support actor: SMS Gateway

Stakeholders' interests:
Proprietario: ottimizzare l'occupazione dei tavoli (eventualmente a scapito dei tempi di attesa dei clienti)
Cliente: ottenere un tavolo nel minor tempo possibile

Minimum guarantees: -

Success guarantees: -

Main success scenario:
1. il cameriere richiede di inserire la prenotazione
2. il sistema richiede il numero di telefono del cliente
3. il cameriere lo inserisce
4. il sistema convalida e invia il messaggio all'SMS Gateway
5. l'SMS Gateway notifica la ricezione e spedizione del messaggio
6. il sistema richiede la conferma
7. il cameriere conferma l'avvenuta ricezione
8. il sistema inserisce il cliente in lista d'attesa

Extensions:
3.a: inattività del cameriere:
    3.a.1 il sistema annulla la richiesta di prenotazione
    il caso d'uso termina con un fallimento
7.a: inattività del cameriere:
    7.a.1: il sistema richiama l'attenzione del cameriere
    7.a.2: il caso d'uso prosegue al passo 7
4.b: esito negativo della convalida (numero non valido):
    4.b.1. il sistema segnala l'errore
    il caso d'uso prosegue al passo 3
7.c.: il cameriere chiede il reinvio del messaggio:
    7.c.1. il caso d'uso prosegue al passo 4
7.d.: il cameriere chiede di modificare il numero:
7.d.1. il sistema richiede di reinserire il numero
7.d.2. il caso d'uso prosegue la passo 3

Use case: Notificare tempo residuo

Scope: Sistema di gestione liste d'attesa

Level: sub-function

Intention in context: inviare notifica periodica del tempo residuo di attesa tramite SMS Gateway

Primary actor: -

Support actor: SMS Gateway

Stakeholders' interests: -

Minimum guarantees: -

Success guarantees: -

Trigger: ogni cinque minuti / passati cinque minuti dall'ultimo SMS inviato

Main success scenario:
1. il sistema calcola il tempo residuo di attesa ed invia il messaggio all'SMS Gateway
3. l'SMS Gateway notifica la ricezione e spedizione del messaggio

Extensions:

Use case: Segnalare disponibilità del tavolo

Scope: Sistema di gestione liste d'attesa – Sottosistema di Sala

Level: user-goal

Intention in context: rendere noto che un tavolo è stato liberato e quindi è libero per il prossimo cliente in lista di attesa

Primary actor: Cameriere

Support actor: SMS Gateway

Stakeholders' interests:
Proprietario: ottimizzare l'occupazione dei tavoli (eventualmente a scapito dei tempi di attesa dei clienti)
Cliente: ottenere un tavolo nel minor tempo possibile

Minimum guarantees: -

Success guarantees: -

Main success scenario:
1. il cameriere notifica che si è liberato un tavolo
2. il sistema richiede il codice del tavolo
3. il cameriere lo inserisce
4. il sistema convalida, individua il cliente da richiamare e invia il messaggio all'SMS Gateway
5. l'SMS Gateway notifica la ricezione e spedizione del messaggio
6. il sistema conferma l'acquisizione della comunicazione

Extensions:
3.a: inattività del cameriere:
    3.a.1 il sistema annulla la notifica
    il caso d'uso termina con un fallimento
4.a: esito negativo della convalida (codice non valido):
    4.b.1. il sistema segnala l'errore
    il caso d'uso prosegue al passo 3
5.a.: inattività da parte dell'SMS Gateway & numero fallimenti < 3:
    5.a.1. il sistema incrementa il numero fallimenti e re-invia il messaggio all'SMS Gateway
    5.a.2. il caso d'uso torna al passo 5
5.b.: inattività da parte dell'SMS Gateway & numero fallimenti = 3:
    5.b.1. il sistema segnala il malfunzionamento dell'SMS Gateway
    il caso d'uso termina con un fallimento

Use case: Richiamare il prossimo in lista

Scope: Sistema di gestione liste d'attesa

Level: sub-function

Intention in context: richiamare il prossimo cliente se quello precedentemente richiamato non è arrivato in tempo

Primary actor:

Support actor: SMS Gateway

Stakeholders' interests:
Proprietario: ottimizzare l'occupazione dei tavoli (eventualmente a scapito dei tempi di attesa dei clienti)
Cliente: ottenere un tavolo nel minor tempo possibile

Minimum guarantees: -

Success guarantees: -

Trigger: Il tempo trascorso dall'invio del messaggio di richiamo di un cliente è superiore alla soglia prefissata ed il cliente non è arrivato

Main success scenario:
1. il sistema il cliente da richiamare e invia il messaggio all'SMS Gateway
2. l'SMS Gateway notifica la ricezione e spedizione del messaggio

Extensions:
2.a.: inattività da parte dell'SMS Gateway & numero fallimenti < 3:
    2.a.1. il sistema incrementa il numero fallimenti e re-invia il messaggio all'SMS Gateway
    2.a.2. il caso d'uso torna al passo 2
2.b.: inattività da parte dell'SMS Gateway & numero fallimenti = 3:
    2.b.1. il sistema segnala il malfunzionamento dell'SMS Gateway
    il caso d'uso termina con un fallimento

Use case: Segnalare l'arrivo di un cliente

Scope: Sistema di gestione liste d'attesa – Sottosistema di Sala

Level: user-goal

Intention in context: rendere noto che un cliente precedentemente richiamato è arrivato e viene fatto accomodare al tavolo

Primary actor: Cameriere

Support actor:

Stakeholders' interests: -

Minimum guarantees: -

Success guarantees: il tavolo precedentemente liberato viene occupato dal cliente

Main success scenario:
1. il cameriere notifica che è arrivato un cliente richiamato a un tavolo
2. il sistema richiede il nome del cliente
3. il cameriere lo inserisce
4. il sistema convalida e segnala il tavolo assegnato
5. il cameriere conferma
6. il sistema considera il cliente accomodato ed il tavolo occupato

Extensions:
3.a: inattività del cameriere:
    3.a.1 il sistema annulla la notifica
    il caso d'uso termina con un fallimento
4.a: esito negativo della convalida (nome non valido):
    4.b.1. il sistema segnala l'errore
    il caso d'uso prosegue al passo 3
5.a: inattività del cameriere:
    5.a.1 il sistema richiama l'attenzione del cameriere e chiede conferma
    il caso d'uso torna al punto 5
5.b: il cameriere non conferma:
    5.a.1 il sistema considera il cliente come non arrivato
    5.a.2 il sistema il prossimo cliente da richiamare e invia il messaggio all'SMS Gateway
    5.a.3 l'SMS Gateway notifica la ricezione e spedizione del messaggio
    il caso d'uso termina con un fallimento

BIBLIOGRAFIA

[1] I. Jacobson, M. Christerson, P. Jonsson, and G. Overgaard, *Object Oriented Software Engineering: A Use Case Driven Approach*. Addison-Wesley Professional, 1992.

[2] A. Cockburn, *Writing effective use cases*. The crystal collection for software professionals, Addison-Wesley Professional Reading, 2000.

[3] M. Fowler, *UML Distilled: Guida rapida al lingugaggio di modellazione standard, 4a edizione*. Addison-Wesley, 2010.

Licenza e Colophon

Questo volume è stato redatto con il sistema di composizione LaTeX[7] utilizzando il modello di stile `memoir`[8].

Il contenuto del testo è rilasciato con la licenza Creative Commons Attribuzione - Non commerciale - Condividi allo stesso modo 2.5 Italia (CC BY-NC-SA 2.5)[9].

[7] `http://www.latex-project.org/`
[8] `http://www.ctan.org/tex-archive/macros/latex/contrib/memoir/`
[9] `http://creativecommons.org/licenses/by-nc-sa/2.5/it/`