Use case

Introduzione

Obiettivi:

- Identificare i *casi d'uso*
- Connettere i casi d'uso rispetto agli obiettivi dell'*attore principale*
- Mantenere uno stile *essenziale*, non troppo informativo, poichè potrebbe rendere maggiormente difficile l'individuazione del *goal*
- Adottare uno *sviluppo iterativo*

Use case spesso è indicato come il meccanismo per scoprire e registrare i requisiti, di carattere *funzionale*. La scrittura dei *casi d'uso* è un eccellente tecnica per capire e descrivere ciò che il *sistema* debba implementare, rispetto ad una totalità di influenze esterne che creino la situazione ideale per poter proseguire con il compimento dell'obiettivo principale da parte del *soggetto*. **UP**, ossia *Unified Process*, indica un certo modello dei casi d'uso, definito come **modello comportamentale**, per cui fortemente connesso rispetto ai *comportamenti* degli elementi trattati.

I *diagrammi* dei *casi d'uso* sono **diagrammi comportamentali** utilizzati per descrivere un insieme di azioni, o meglio *behavior*, che un sistema deve svolgere insieme ad uno o più utenti esterni al sistema. Inoltre, carattere imprescindibile di tale modellazione, prevede che ogni *use case* fornisca un **risultato di valore**, per gli utenti con cui interagiscono oppure per gli *stakeholders*. *Customers* e *final users* pongono certi obiettivi, detti *needs*, i quali richiedono che *sistemi* siano in grado di rendere chiaro il loro **connubio**.

Differenti sono gli approcci dettati, tuttavia le strategie migliori riguardano un uso semplice e famigliare degli strumenti messi a disposizione, poichè rendono più semplice la lettura del modello, contribuiscono alla ricerca dei requisiti necessari e abbassa il rischio di perdere l'obiettivo su cui fonda l'intera architettura, definito anche *missing the mark*.
Infatti il primo step ricade nella comprensione del problema assieme all'individuazione dei requisiti funzionali che comporranno i casi d'uso, affinchè siano comprensibili da parte dei portatori di interesse. Tuttavia, quest'ultimo passaggio rappresenta la maggiore difficoltà di questo modello, ossia la decisione e scoperta di cosa sia potenzialmente necessario; durante una prima fase di analisi potrebbero essere inseriti dei *layer sofistaticati-aggiuntivi*, i quali solitamente oscurano il reale proseguimento del *mark*.

Per cui, ricapitolando, il diagramma dei casi d'uso è utilizzato per specificare:

- I requisiti esterni degli stakeholders, funzionali rispetto a quanto il sistema debba implementare
- Le funzionalità offerte dal sistema per tutti i final user esterni
- Rispettare i limiti secondo cui il sistema debba sottostare, defininendo come tale ambiente debba interagire con il soggetto pur di proseguire verso il *path* voluto

Elementi del diagramma

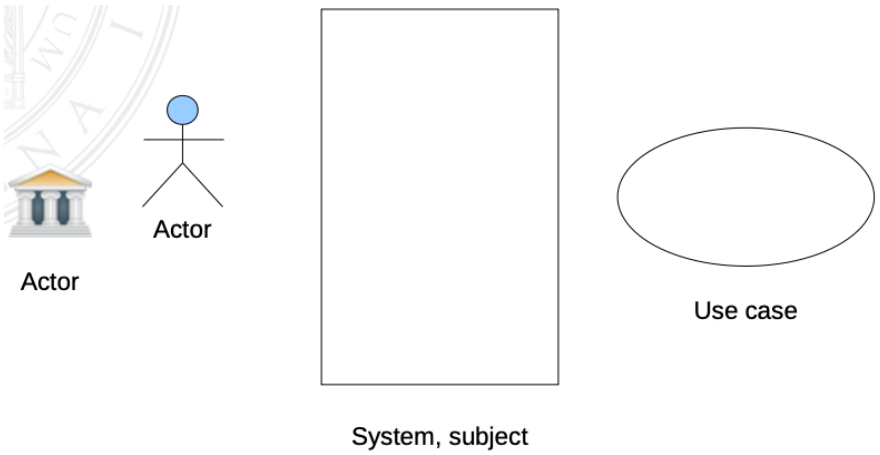
In questa sezione si analizzano gli elementi comuni ad un qualsiasi diagramma dei casi d'uso, descrivendo successivamente caratteristiche uniche rispetto alle strategie elencate.

Definizione informale
Un **attore** è qualcosa caratterizzato da un comportamento, come una persona, un computer system oppure un'organizzazione.

Per cui, descrivendone maggiormente il ruolo svolto, un **attore** è un **classificatore comportamentale**, ossia un'entità che interagisce con il sistema implementato o ideato mediante **interazioni**, quali, semplicemente, l'invio di segnali o di dati.

Definzione informale
Uno **scenario**, definita anche *istanza* del caso d'uso, indica un preciso percorso attraversando certi e specifici *use case*.

Quindi il modello comportamentale trattato è definito mediante un insieme di scenari fallimentari o di successo, in relazione al conseguimento dell'obiettivo dell'attore principale, adeguandosi ai comportamenti illustrati all'interno dei singoli casi d'uso. E' bene sottolineare come ogni caso d'uso rappresenti uno specifico requisito funzionale, posto al di fuori del diagramma, dove solamente mediante interazioni è possibile stabilire un path per cui il sistema sia in grado di raggiungere l'obiettivo.
UP definisce differenti formati e tipi di use case...



Attore

Negli UML un **attore** è un **classificatore comportamentale** che *specifica il ruolo svolto da un entità esterna che interagisce con il soggetto* (ad esempio, scambiando segnali e dati), un utente umano del sistema, altri sistemi o hardware che utilizza servizi del soggetto.

Soggetto

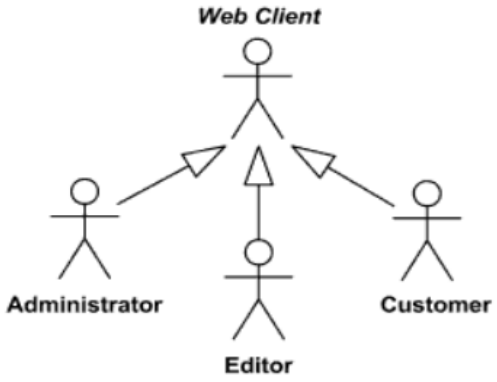
Il **soggetto** è il **sistema sotto analisi** al quale vengono applicati i diversi casi d'uso. Il soggetto può essere un business, un sistema software, un sistema fisico, o un sottosistema più piccolo con qualche comportamento. In termini di UML, il soggetto è un **classificatore di casi d'uso** che ha il ruolo di "*soggetto*".

Casi d'uso

Negli UML, un **caso d'uso** è un **classificatore comportamentale** che specifica il comportamento di un soggetto descrivendo una **sequenza di azioni** eseguite dal sistema per ottenere un **risultato osservabile** di un qualche valore per uno o più attori o stakeholders del sistema. Ogni caso d'uso quindi descrive un'unità di una funzionalità utile e finita che il soggetto fornisce agli utenti. Vengono di solito rappresentati graficamente come un **ellisse** contenente il nome del caso d'uso.

Relazioni tra attori

Possiamo definire attori astratti o reali e specializzarli usando **relazioni di generalizzazione**. La generalizzazione tra attori è rappresentata graficamente come una **linea direzionata da una grande punta** (stesso discorso per la generalizzazione tra classi).



Associazioni tra attori e casi d'uso

Ogni caso d'uso specifica delle funzionalità utili che il soggetto fornisce agli attori. Queste funzionalità devono essere **inizializzate da un attore**. Gli attori possono essere connessi ad un caso d'uso soltanto da una **associazione relazionale binaria**.

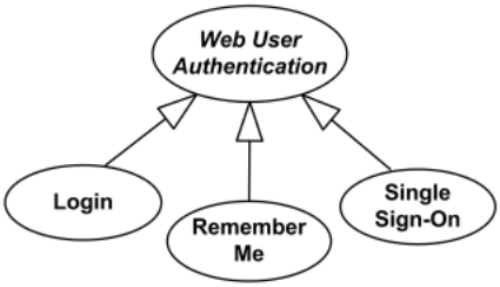
Relazioni tra casi d'uso

I casi d'uso possono essere organizzati usando le seguenti relazioni:

- **Generalizzazione**
- **Extend**
- **Include**
- (*Associazione*)

Generalizzazione

La generalizzazione viene rappresentata graficamente come una **linea direzionata da una grande punta** (stesso discorso per la generalizzazione tra classi), dove la freccia va **dal caso più specifico verso quello più generico**.

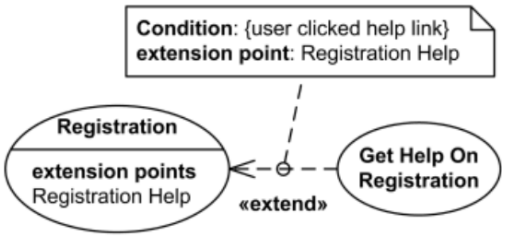


In foto possiamo osservare diversi modi per eseguire il "Web User Authentication". Ogni modo porta allo stesso obiettivo, ma con **percorsi diversi**.

Extend

L'**extend** è una relazione diretta che specifica come e quando il comportamento definito in maniera opzionale, *che estende il caso d'uso*, **può essere** inserito nel comportamento definito nel caso d'uso esteso. L'estensione prende luogo in uno o più punti di estensione definiti nel caso d'uso esteso. Le relazioni *extend* vengono rappresentate graficamente come una **freccia tratteggiata con una punta aperta**, dove la freccia va **dal caso esteso verso il caso base**. La freccia viene inoltre rinominata con la **parola chiave "extend"**.

Un **punto di estensione** è una funzionalità di un caso d'uso che **identifica un punto nel comportamento del caso d'uso dove il comportamento può essere esteso da altri casi d'uso**, come già specificato dalla relazione di estensione. I punti di estensione possono essere rappresentati graficamente in **uno spazio del simbolo ovale dei casi d'uso**. Ogni punto di estensione deve avere un nome, unico all'interno del caso d'uso.

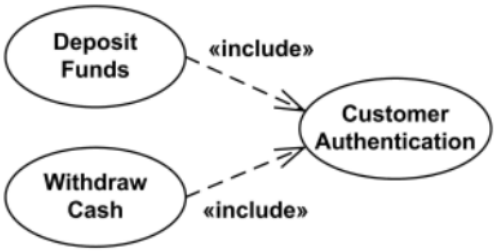


Include

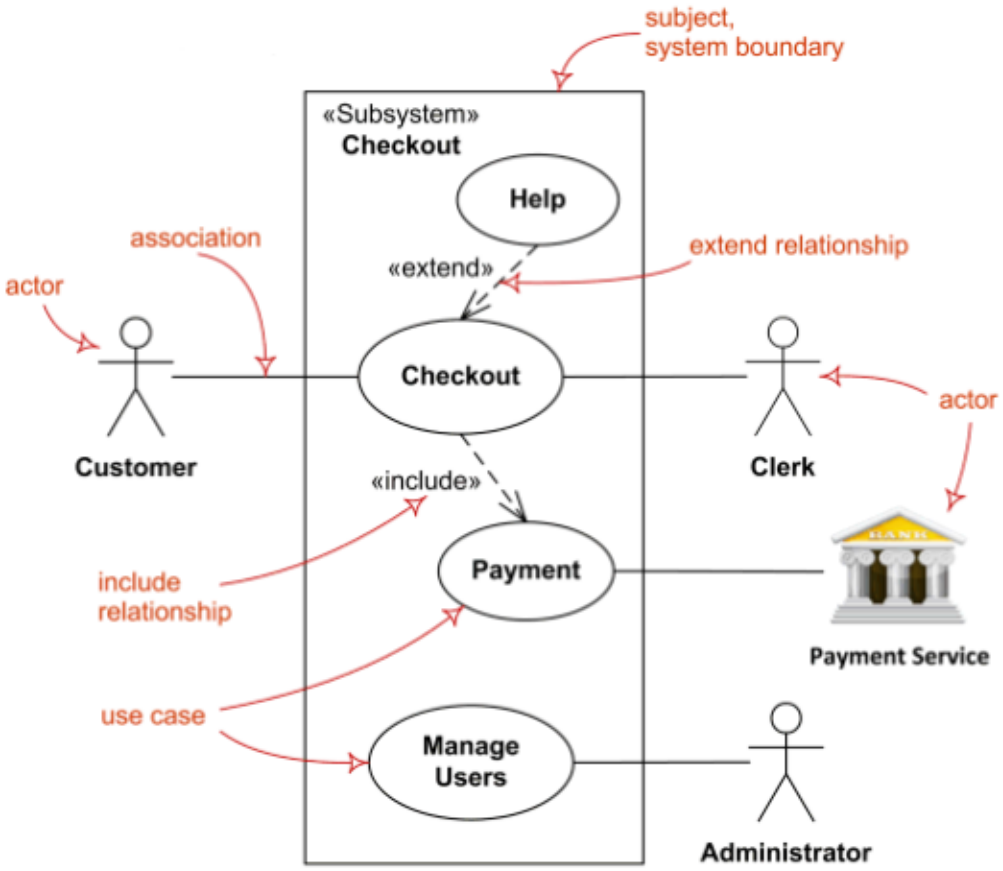
Una **relazione di inclusione** è una relazione diretta tra due o più casi d'uso utilizzata quando un comportamento richiesto, **non opzionale**, è **inserito nei comportamenti inclusi nel caso d'uso base**. Questo tipo di relazione può essere usato quando:

- ci sono parti comportamentali comuni tra due o più casi d'uso
- per semplificare grandi casi d'uso dividendolo in più casi d'uso

Una relazione di inclusione può essere rappresentata graficamente come una **freccia tratteggiata con una punta aperta**, dove la freccia va **dal caso d'uso base verso il caso d'uso da includere** (comune per più casi d'uso). La freccia viene inoltre rinominata con la **parola chiave "include"**.



Esempio di diagramma dei casi d'uso



Nota Bene: è consuetudine, se è presente ambiguità su quale attore sia l'inizializzatore, che ad inizializzare sia l'attore di sinistra.

Casi d'uso di sistema vs casi d'uso business

L'obiettivo primario è di avere sempre un modello completo e preciso. Tuttavia, ci sono delle differenze tra i casi d'uso di sistema e i casi d'uso business. Il primo ha come soggetto il sistema, mentre il secondo un organizzazione. Inoltre, per i casi d'uso business, gli attori possono essere anche interni all'organizzazione.

Portata del caso d'uso di sistema

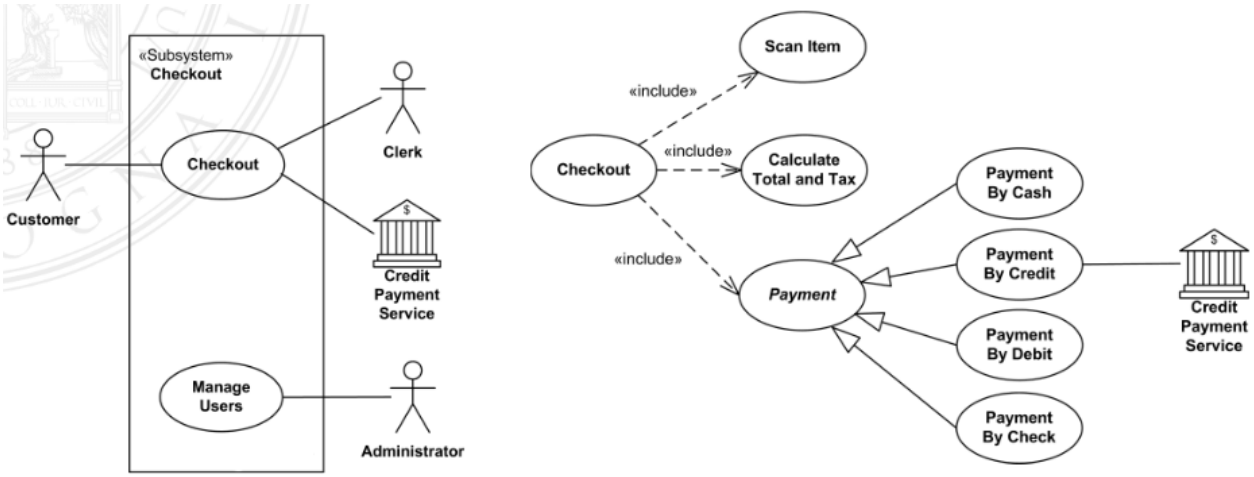
Per quanto riguarda la creazione di un caso d'uso, si cerca di creare il **più piccolo insieme di attività** che offre un risultato significativo per l'utente, definito da obiettivi che possono essere completati in una **sessione**. Di solito si tratta di una dozzina di passi. Quando un caso d'uso coinvolge più attori con obiettivi diversi si divide il caso d'uso in più casi d'uso.

Cosa fare e cosa non fare con i casi d'uso

- Sommario visuale delle diverse funzionalità

- Evitare di specificare troppo i passaggi tramite interfaccia grafica (può essere modificata durante lo sviluppo)
- Non decomporre per altri motivi diversi dal riutilizzo
- Avere una comprensione accurata delle differenze tra generalizzazione, inclusione e estensione
- Non creare nuove relazione stereotipate
- Il tempo può essere un attore
- Gli attori sono esterni al sistema (per casi d'uso di sistema)

Vediamo ora un esempio di due casi d'uso rappresentati graficamente. In quello di sinistra abbiamo una rappresentazione meno dettagliata, ma comunque efficace. A destra risulta essere molto informativa (anche troppo).



Modello di casi d'uso e diagramma dei casi d'uso

Un diagramma UML per un caso d'uso **NON** è un modello di caso d'uso! Il diagramma può essere visto come un **sommario**, non avendo al suo interno aspetti fondamentali come i dettagli sulle fasi di interazione e quando il caso d'uso viene applicato, le condizioni pre/post, ecc. Inoltre, **non esiste** una notazione standard per rappresentare un modello di caso d'uso, ma esistono molti **template** utilizzati in base al caso specifico. I template possono essere **molto dettagliati o più semplici**, in base ai differenti casi e dal team di sviluppo (è fondamentale rimanere costanti). In ogni template sono presenti diverse **sequenze**, che ci indicano i diversi casi possibili durante una ipotetica esecuzione. Spesso, oltre alla sequenza che illustra l'esecuzione ottimale (*Happy path*), vengono indicate sequenze alternative che portano comunque all'obiettivo finale seguendo un percorso diverso. Può anche esserci una sequenza (*Exceptions*) dove non viene raggiunto l'obiettivo.

Sequenze

Il concetto di sequenza è presente in tutti i template per i modelli di caso d'uso. Vediamo un esempio di sequenza (vengono tralasciate le cose più ovvie, come le precondizioni):

MAIN SUCCESS SEQUENCE (a.k.a. Happy path):

1. L'utente seleziona l'opzione per entrare in un nuovo trasferimento
2. Il sistema mostra il modulo per effettuare il trasferimento (conto destinatario, dettagli, importo, data)
3. L'utente inserisce i dati nel modulo ed effettua l'ordine
4. Il sistema mostra il messaggio di accettazione dell'ordine

EXTENSIONS:
3a. Importo non disponibile:
3a1. Il sistema mostra un messaggio di errore e torna allo step 2

Vogliamo capire soltanto il caso d'uso, non come viene eseguito o come deve essere svolto ogni passaggio. Ci chiediamo quindi solo **cosa** vuole fare, e non **come** deve farlo!

Scenari

Uno scenario può essere visto come una instanza di un caso d'uso. Descrive una possibile interazione tra attori ed il sistema. Vengono utilizzati per documentazione e per creare casi di test.

Usare uno scenario è molto utile per i casi di testing, dove possiamo spiegare gli ipotetici n scenari possibili. Un altro motivo è che cerchiamo di catturare i requisiti funzionali del sistema. Usando uno scenario come test verifichiamo il funzionamento del caso d'uso.

Stati di un modello di caso d'uso

Un modello di casi d'uso è tendenzialmente soggetto a perfezionamenti iterativi (possiamo vederlo come una sorta di ciclo di vita del modello):

- Obiettivo stabilito
- Struttura della storia compresa
- Storia più semplice completata
- Storia sufficiente completata
- Tutte le storie completate

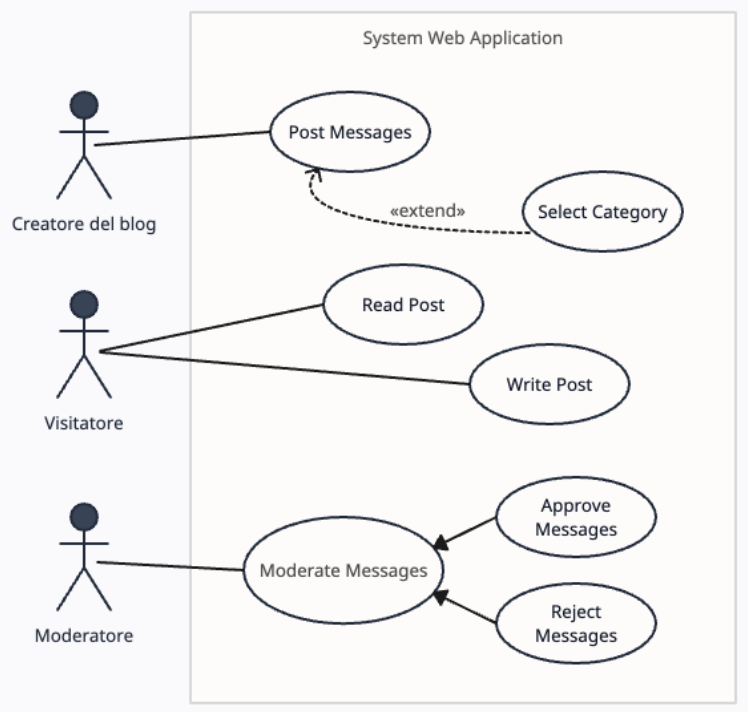
Esercizio

Un *blog* è una *web application* che contiene una serie di messaggi (*post*) datati su diversi argomenti. I messaggi vengono postati dal creatore del blog che li mette online. L'autore può associare ad ogni messaggio una o più categorie (tramite delle parole chiave).
I visitatori del blog possono commentare i messaggi; i commenti, se approvati dal moderatore (di solito è il creatore del blog), appaiono in una sezione specifica sotto il messaggio originale.

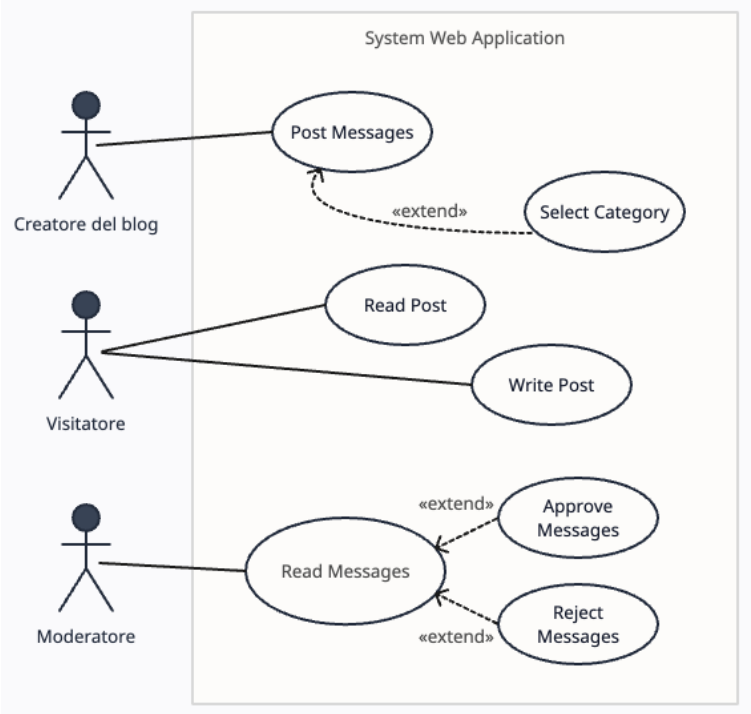
Partiamo capendo quale sia il soggetto, ovvero la nostra *web application*. Procediamo poi trovando gli attori:

- Creatore del blog
- Visitatore
- Moderatore

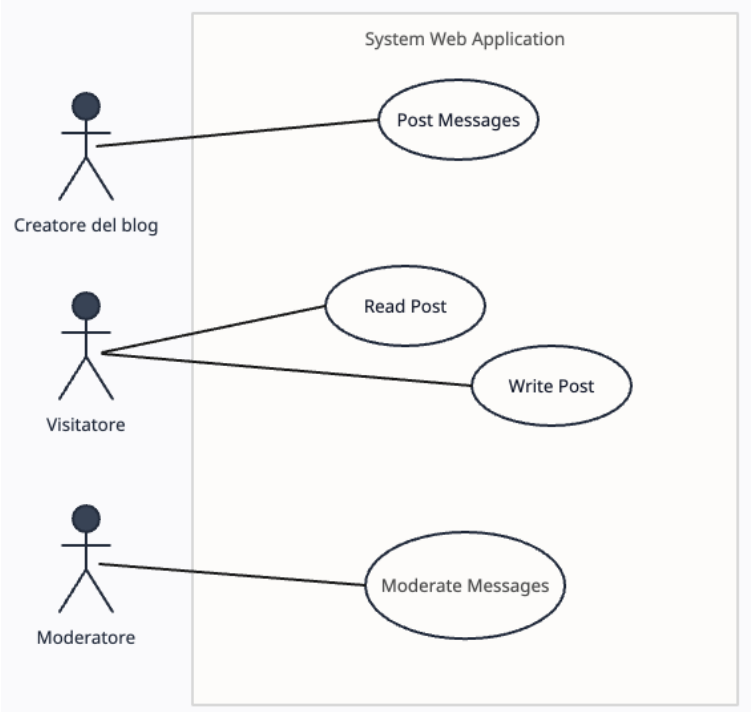
L'obiettivo del creatore del blog è quello di **pubblicare i messaggi**.
Vediamo ora un modello di casi d'uso d'esempio per questo caso:



Come possiamo osservare, abbiamo i nostri tre attori (creatore del blog, visitatore e moderatore) ed i loro obiettivi. Un'altra possibile soluzione poteva essere la seguente:



In questo caso dovremmo specificare in qualche modo che non possono essere eseguite entrambe le operazioni (approve e reject), ma soltanto una per commento (magari tramite l'utilizzo di XOR). Tra i due approcci è preferibile il primo, non dovendo specificare l'esclusività e per un approccio meno confusionario rispetto al secondo. Se volessimo mantenere un modello funzionale più semplice:



Questa soluzione è buona, anche se meno dettagliata. In questo caso però il più completo è

migliore, perchè non risulta essere troppo confusionario. Ripensando al ciclo di vita di un modello di casi d'uso, questa potrebbe essere la fase iniziale della modellazione.

Se avessimo voluto usare un template di caso d'uso semplice, avremmo ottenuto (stiamo considerando l'obiettivo *write comment*):

- **UC1**: write comment
- **Actors**: visitor
- **Pre-condition**: utente visualizza il messaggio
- **Main Sequence**:
  (a) utente scrive il commento
  (b) utente invia il commento
  (c) sistema conferma la ricezione del commento
- **Alternative Sequence**: il messaggio contiene linguaggio non appropriato
  (c1) Il sistema cancella il commento
- **Post-condition**: il messaggio viene aggiunto alla lista dei messaggi da moderare