Soluzione prova scritta del 04/02/2021

Tempo a disposizione: 75 minuti. È permesso consultare testi o appunti.

Si consideri il seguente scenario:

Un minimarket vuole fornire un servizio di delivery ai propri clienti. Per farlo predispone un sistema distribuito basato su applicazioni mobili e interfaccia web. Il sistema può essere usato:

- *dal cliente (per ordinare i prodotti, tramite app mobile)*
- *dal rider (per consegnare la merce, tramite app mobile)*
- *dal gestore (per caricare i prodotti disponibili e le quantità, tramite interfaccia web)*

App Cliente

Si assume che ogni cliente abbia un account e che sia loggato nel sistema. Oltre ai dati personali, del cliente è noto il suo indirizzo in forma testuale e come latitudine e longitudine. Successivamente riceverà una lista di prodotti con fotografia, titolo e prezzo. Cliccando sul prodotto potrà visualizzare il dettaglio ed inserirlo nel carrello indicando le quantità. Il pagamento avverrà esclusivamente via contanti o tramite carta di credito con il rider. Una volta completato l'ordine sarà possibile procedere al checkout. A quel punto l'app del cliente cerca un rider disponibile. Una volta che il rider ha accettato l'incarico e ha lasciato il minimarket, l'utente riceverà una notifica e da quel momento il cliente potrà inserire i dettagli per la consegna (es.: citofono, numero di telefono, ecc.).

Dopo la consegna l'utente può valutare il servizio (qualità merce, velocità consegna, cortesia rider)

In qualsiasi momento l'utente può visualizzare la lista di tutti gli ordini pregressi.

App Rider

Si assume che ogni rider abbia un account e che sia loggato nel sistema. Il rider, in qualunque momento, imposta il suo stato (disponibile per consegne, oppure non disponibile). Se disponibile potrà essere selezionato dal gestore.

Il quando un utente conclude un ordine un rider disponibile riceverà una notifica per la consegna e potrà accettare l'incarico o rifiutarlo. Una volta accettato, scansionerà il Barcode della consegna (oppure, in caso di problemi, inserendo manualmente) e visualizzerà sulla mappa il percorso per raggiungere l'indirizzo del cliente.

Una volta raggiunto il cliente, si segnalerà l'avvenuto pagamento e la fine della consegna. In caso di problemi di pagamento o di cliente assente sarà segnalato il fallimento della consegna ed il rider riporterà i prodotti in negozio.

Ogni rider deve poter visualizzare lo storico degli incarichi con il loro esito (accettato e consegnato, accettato e consegna fallita, incarico rifiutato).

Al termine della consegna il rider può valutare il cliente (cortesia, presenza in casa, ...)

Interfaccia Gestore

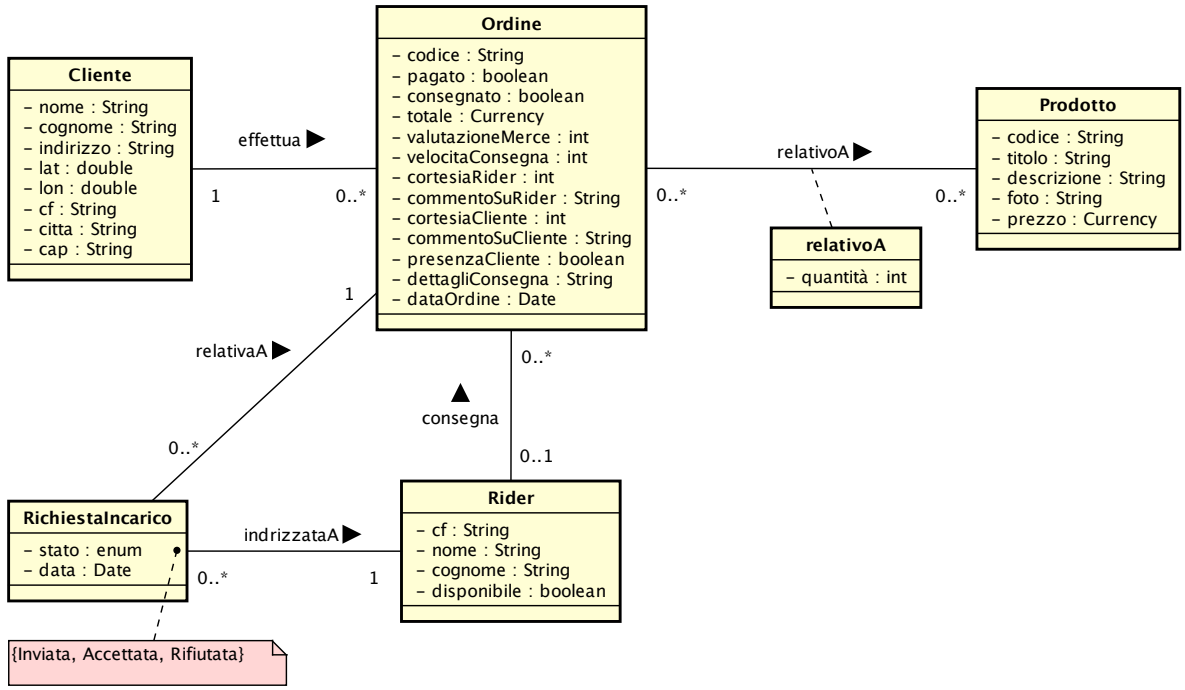
Si assume che il gestore abbia un account e che sia loggato nel sistema. Il gestore potrà caricare gli articoli disponibili. È possibile caricare una foto, un titolo, una descrizione, il prezzo. È inoltre possibile modificare i prodotti già disponibili aggiornando foto, descrizione ecc.

Nel contesto dello scenario delineato sopra, si definiscano:

1. Il diagramma delle classi
2. Il modello del processo

→ È necessario modellare esclusivamente gli aspetti direttamente rilevanti per il sistema informativo.

1) Diagramma delle classi

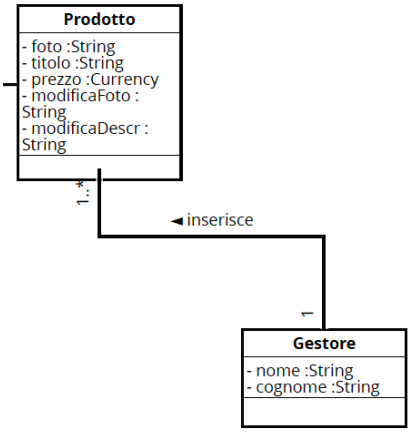


È possibile, invece della classe di associazione *relativoA* avere un classe intermedia che rappresenta la singola linea dell'ordine.

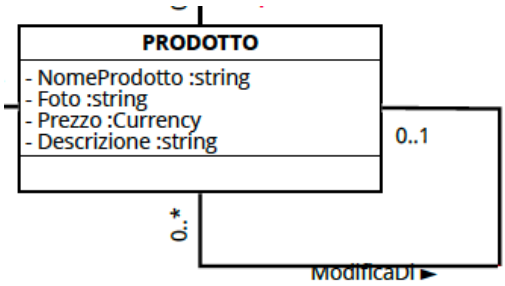
Lo stato delle richiesta potrebbe essere sostituito da due boolean: uno che indica se il rider ha dato una risposta, l'altro che indica se ha accettato.

Errori tipici

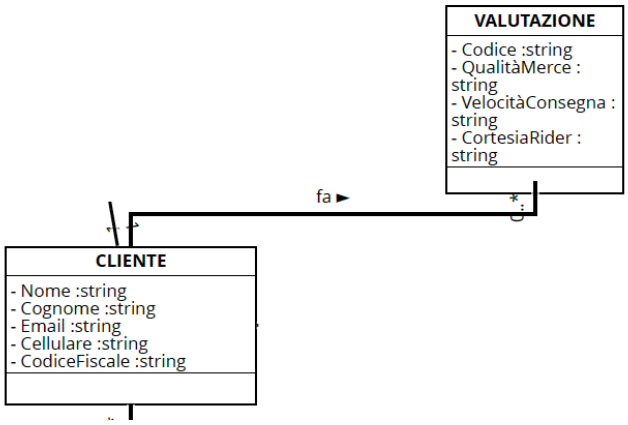
In molti hanno modellato queste due situazioni dove il gestore inserisce/carica/pubblica/possiede modifica ecc. Ciò non era richiesto. Le azioni (mostrate nel diagramma di processo) devono essere modellate quando, ad esempio in un blog, viene mostrato chi ha pubblicato o modificato un post. In questo caso specifico il gestore era uno solo. Per cui possiamo assumere che sia sempre lui a fare questo tipo di operazione.



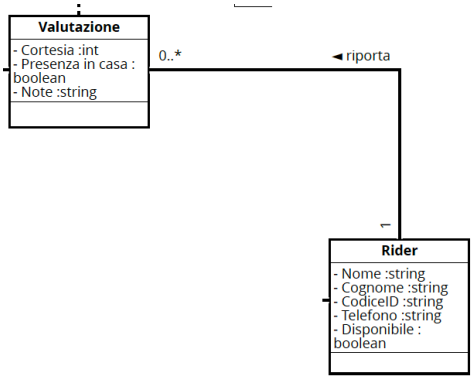
Qui è stato invece modellata la possibilità di vedere da quale prodotto fosse stato clonato il prodotto corrente. Esempio: Acqua vera, foto, 1,20€, “Cassa di acqua Vera da 6 bottiglie” e Acqua Panna, foto, 1,50€, “Cassa di acqua Panna da 6 bottiglie” ed un riferimento all’acqua vera da cui si è partiti per modificarlo. Questo da un esercizio fatto in aula (DVD) ma se nel caso dei film aveva senso il puntatore al remake qui assolutamente no.



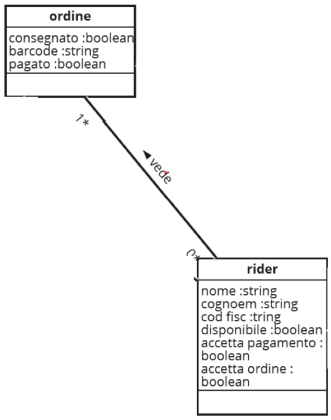
In tantissimi hanno modellato azioni nel modello concettuale. Questo è un errore molto grave. Se a volte il mettere un verbo sulla relazione è inevitabile (esempio cliente effettua ordine), solitamente bisognerebbe evitarlo per non incappare in questo tipo di errori. Infatti nonostante nell’esempio cliente effettua ordine ci sia un verbo, qui non stiamo modellando il fatto di effettuare l’ordine in quanto azione, piuttosto avere da parte del cliente la lista degli ordini effettuati e dall’altra parte per ogni ordine avere i dati del cliente che l’ha effettuato. Qui invece si sta modellando che il cliente fa una valutazione. Ma poi la valutazione è fine a se stessa e non è collegata ad un ordine. Quindi avremmo per il cliente una lista di valutazioni e ogni valutazione avrebbe un cliente. Ma non è possibile sapere l’ordine per cui si sta facendo questa valutazione



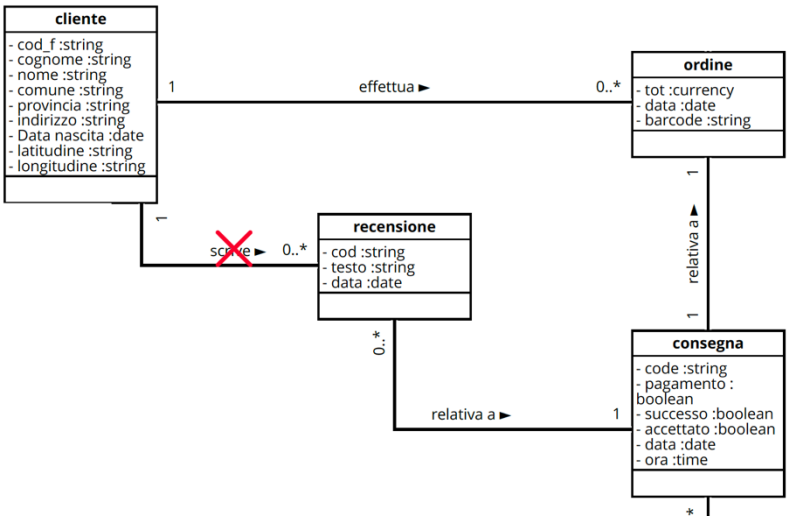
La stessa cosa succede in molti compiti per il rider.



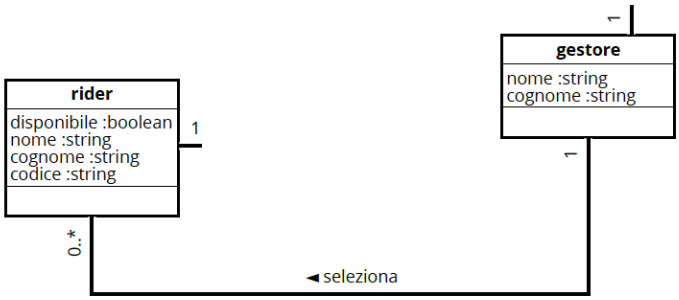
Questo è un altro esempio di azione modellata nel posto sbagliato. Non ha senso salvarsi che il rider vede l’ordine. La sua visualizzazione non dovrebbe essere tracciata ogni volta



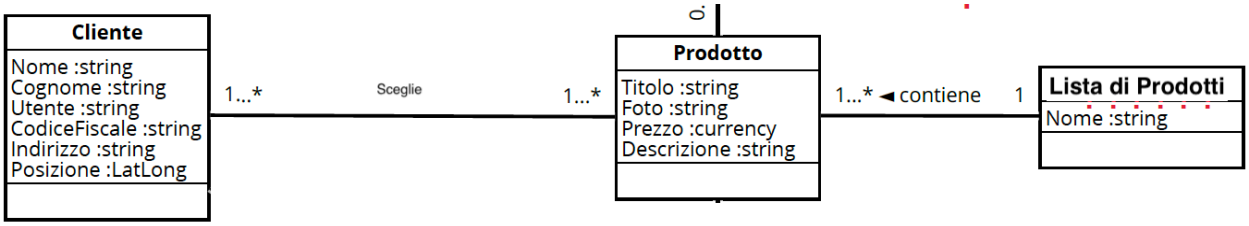
Qui invece abbiamo una situazione intermedia che però dovrebbe aiutarci a capire quando scegliere di collegare o meno due classi tra di loro. La recensione del cliente è stata in questo caso collegata correttamente con la consegna. Dopo però è stata collegata anche con il cliente con la relazione scrive. Questa è una azione. Nel modello concettuale non devo modellare l'azione di scrittura della recensione. Ammettendo di non aver saputo scrivere un'altra etichetta, quale vantaggio mi dà il sapere che cliente mario rossi ha scritto proprio quella recensione? La consegna di quell'ordine non può essere solo e soltanto per Mario Rossi? La risposta e si e per questo motivo viene meno il bisogno di relazionare cliente e recensione. La stessa cosa vale per il rider. per cui entrambe le valutazioni sarebbero potute essere rappresentate come semplici attributi di consegna o ordine oppure come classi ma collegate **SOLO** con ordine o consegna. In questo esempio si è deciso di spezzarle ma se notate c'è una molteplicità 1 1. Quando è così in teoria le due classi potrebbero anche essere una sola. Il decidere di separarle (perché rappresentano due concetti diversi) è una possibilità valida.



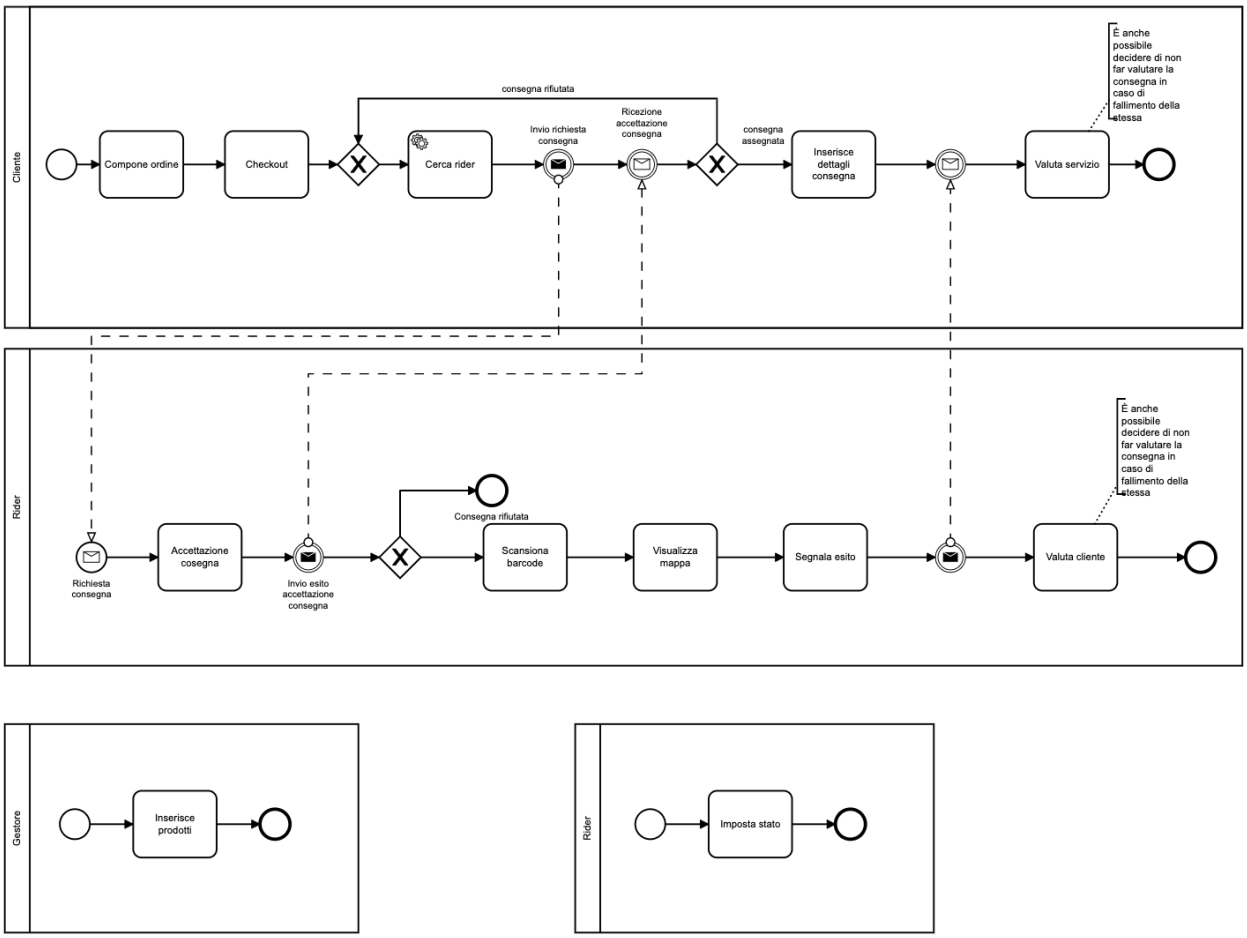
Sempre nel contesto del gestore, che è uno solo e quindi se non per getire la sua utenza non dovrebbe nemmeno apparire nel modello concettuale di questo esercizio, in molti hanno modellato la selezione del rider. Qui è chiaramente modellata una azione dove il gestore clicca da qualche parte per assegnare il rider. La selezione poi è sempre collegata tra gestore e rider e completamente scollegata dalla consegna o dall'ordine. Quindi un gestore avrebbe l'elenco di rider che ha assegnato ma non saprebbe legarli ad una consegna specifica.



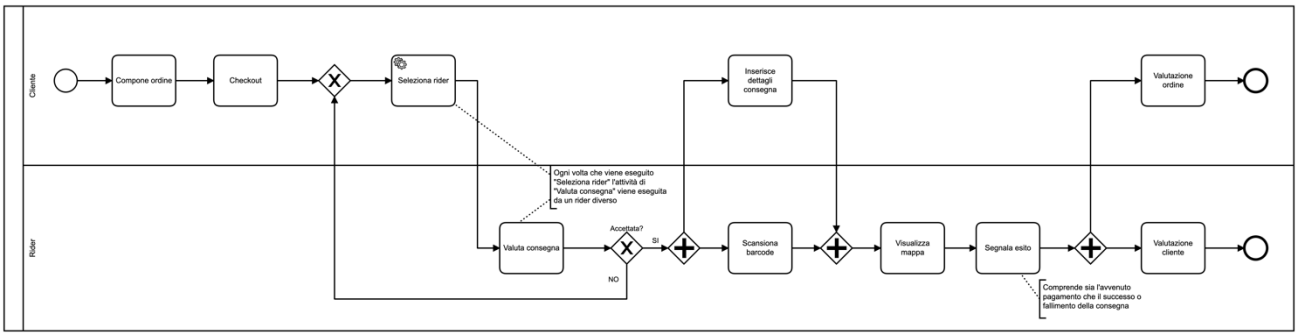
Questo è un errore grave. Qui il cliente avrebbe una lista disordinata di prodotti (quindi la somma di tutti quelli che ha ordinato da quando compra online) e poi la classe Lista di Prodotti non è il modo corretto di descrivere questi aspetti con la modellazione concettuale. È la molteplicità che indice se si ha un prodotto o una lista di prodotti. La classe deve essere sempre un nome singolare.



2) Modello del processo



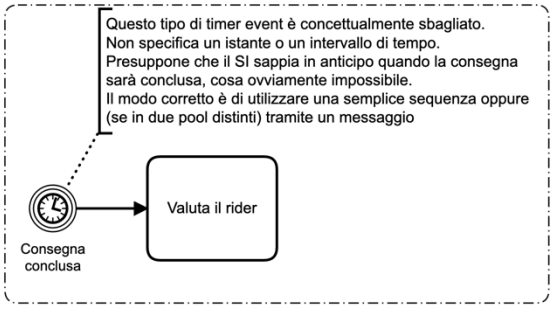
È accettabile anche una soluzione con un solo pool, in tale case non c'è la necessità di scambio messaggi perché i messaggi non possono essere scambiati all'interno di uno stesso pool e la sincronizzazione avviene tramite il flusso di esecuzione.

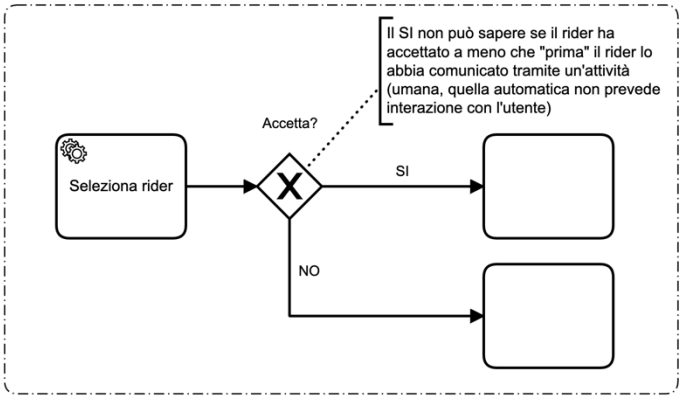


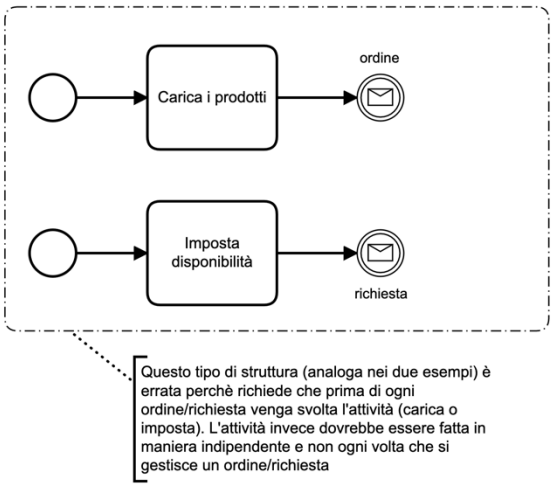
Il testo poteva essere ambiguo relativamente alla ricerca del Rider: nella soluzione fornita la ricerca è effettuata automaticamente del sistema, in alternativa era possibile prevedere un'attività, svolta dal gestore, di selezione, in tale caso era necessaria una swimlane (o un pool) aggiuntiva per questo attore.

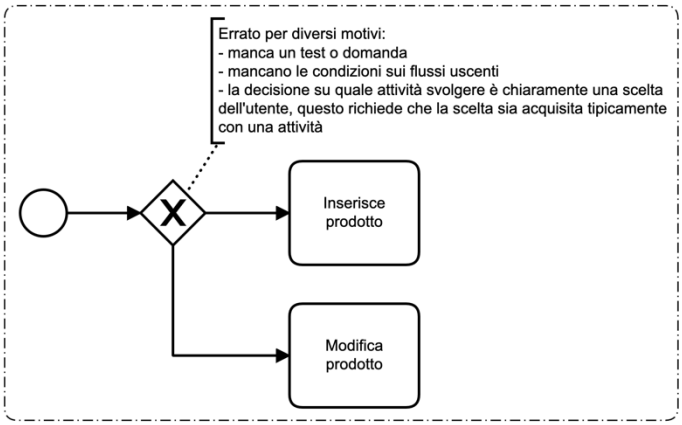
La soluzione fornita prevede la possibilità di non inserire un commento in caso di fallimento. Tale possibilità poteva essere resa in maniera esplicita con un gateway di scelta prima dell'ultima attività che in base all'esito inserito dal Rider (e inviato con messaggio al processo del cliente) possa far terminare il processo senza la valutazione.

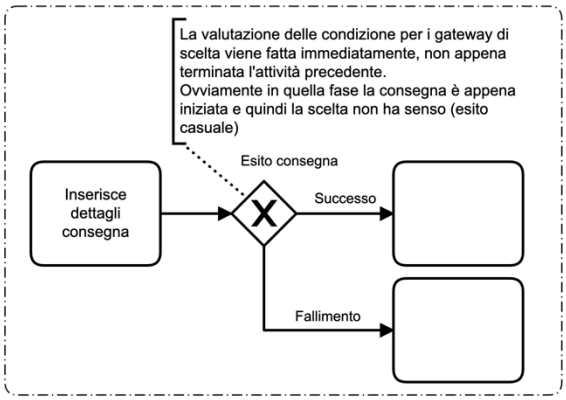
Alcuni errori tipici di uso della notazione BPMN:











Frequentemente sono state assegnate attività di pagamento al cliente ma il testo dice chiaramente che il pagamento avviene tramite il rider che è ovviamente responsabile di segnalarlo al sistema.

Il processo non è corretto se in caso di rifiuto da parte di un rider non ne viene cercato un altro. Ovviamente non basta scriverlo (in un commento, come etichetta di un evento finale o in un'attività) ma occorre prevedere la selezione del prossimo rider da interpellare.

3) Narrativa caso d'uso della gestione della consegna

Use case	**Gestione consegna**
Scope	App Rider
Level	User-goal
Intention in context	Consegnare l'ordine al cliente e ottenere una buona valutazione
Primary actor	Rider
Support actor	-
Stakeholders' interests	Cliente: ottenere velocemente la merce Gestore: far avere i propri prodotti al cliente
Precondition	Ordine esistente
Minimum guarantees	-
Success Guarantees	L'ordine è consegnato al cliente
Trigger	Notifica di richiesta di incarico
Main Success Scenario	1. Il sistema mostra la richiesta di incarico 2. Il rider accetta la consegna 3. Il sistema chiede la scansione del Barcode 4. Il rider scansiona il Barcode 5. Il sistema conferma e mostra la mappa 6. Il rider chiede di concludere la consegna 7. Il sistema richiede l'esito della consegna 8. Il rider inserisce esito positivo 9. Il sistema salva l'esito e richiede la valutazione della consegna 10. Il rider inserisce la valutazione 11. Il sistema salva i dati Il caso d'uso termina con successo
Extensions	1b il rider rifiuta, il caso d'uso termina con un fallimento 3b Il rider inserisce manualmente il codice: il caso d'uso riprende dal punto 4 7b il rider inserisce esito negativo 7b1 Il sistema chiede la valutazione della consegna 7b1 Il sistema salva i dati ed il caso d'uso termina con un fallimento

4) Calcolo dell'effort

Use case	**Inserimento prodotti**
Scope	Sistema informativo web
Level	User-goal
Intention in context	Inserimento prodotti disponibili per la vendita
Primary actor	Gestore
Support actor	-
Stakeholders' interests	Cliente: acquistare online i prodotti Rider: avere prodotti da consegnare
Precondition	
Minimum guarantees	-
Success Guarantees	Il prodotto è inserito a sistema
Trigger	-
Main Success Scenario	1. Il gestore chiede di inserire un nuovo prodotto 2. Il sistema chiede i dati del prodotto 3. Il gestore inserisce i dati del prodotto 4. Il sistema chiede di inserire la foto del prodotto 5. Il gestore inserisce carica la foto del prodotto 6. Il sistema chiede il prezzo 7. Il gestore inserisce il prezzo 8. Il sistema salva i dati Il caso d'uso termina con successo
Extensions	3a. Il gestore annulla: il caso d'uso termina con un fallimento 3b. il gestore torna indietro: il caso d'uso riprende dal punto 2 5a. il gestore annulla: il caso d'uso termina con un fallimento 5b. il gestore torna indietro: il caso d'uso riprende dal punto 4 7a. il gestore annulla: il caso d'uso termina con un fallimento 7b. il gestore torna indietro: il caso d'uso riprende dal punto 6 8b. c'è un errore nel salvataggio dei dati: il caso d'uso riprende dal punto 7

L’azienda decide di affidare lo sviluppo ad un team basato in Turchia. Si decide di utilizzare l’inglese come lingua di riferimento ed il Lead Analyst è un esperto del settore di madrelingua turca con un livello di inglese medio. Il team lavora solo a questo progetto ma hanno un livello di esperienza in generale molto elevato per quanto riguarda la programmazione ad oggetti, ma non hanno mai sviluppato applicazioni web/mobili distribuite. Il team ha un costo giornaliero medio di 250 euro. Il cliente richiede uno sviluppo modulare che gli consenta di aggiungere funzionalità nel medio termine e deve gestire dati sensibili degli utenti quali gli indirizzi di residenza.

Metodo degli Use-Case Points:

UC	Actors Weight	Transactions	UC Weight	UUCP
Inserimento prodotti	3	5	10	13

Technical factors (in giallo sono riportati i valori già forniti in partenza):

Factor	Description	Weight	Rating (0-5)	TF (W*R)
T1	Distributed System	2	3	6
T2	Response time	2	2	4
T3	End User Efficiency	1	0	0
T4	Complex Internal Processing	1	2	2
T5	Reusable Code	1	5	5
T6	Easy to install	0,5	5	2,5
T7	Easy to use	0,5	5	2,5
T8	Cross-platform support	2	5	10
T9	Easy to change	1	5	5
T10	Concurrent	1	3	3
T11	Includes Security Features	1	4	4
T12	Provides Access for 3rd parties	1	4	4
T13	User Training Required	1	0	0

Environmental factors

	Description	Weight	Rating (0-5)	EF (W*R)	
F1	Familiarity With The Project	1.5	0	0	Si tratta di un nuovo progetto
F2	Application Experience	0.5	0	0	Mai sviluppato sistemi di questo tipo
F3	Object Oriented Experience	1	5	5	Elevata esperienza object-oriented
F4	Lead Analyst Capability	0.5	3	1.5	Lead analyst esperto ma non madrelingua
F5	Motivation	1	4	4	Lavorano solo a questo progetto, problemi per la lingua?
F6	Stable requirements	2	4	8	Livello medio di inglese e non previste variazioni a breve
F7	Part Time Workers	-1	0	0	Lavorano solo a questo progetto
F8	Difficulty of programming language	-1	2	-2	Esperienza elevato ma nuovi a questo tipo di applicazioni

Software Size	
UUCP	13

Context factors	
T Factor	48
TCF	**1,08**
E Factor	16.5
ECF	**0,91**

Productivity norms		
		PN1 10
		PN2 14
n1	2	PN3 18
n2	0	
n1+n2	1	
Productivity	10	PH / UCP

UCP	**12,71**	
Effort	**127,06**	person hours
	equivalente a	
	15,88	person days
Person cost	**250**	€/day
Total cost	**€ 3970,69**	

5) Teoria

Quando ci si riferisce all'organizzazione quale tra questi aspetti non si deve includere:

- ~~persone che operano all'interno dell'organizzazione~~
- ~~struttura~~
- **strategie di marketing**
- ~~funzioni aziendali~~
- ~~processi~~

In BPMN quando si deve modellare un task utente che viene eseguito al di fuori del sistema informativo di cui è importante il risultato:

- ~~Generic task~~
- **Manual task**
- ~~User task~~
- ~~Service task~~
- ~~Non va modellato~~

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

- ~~Definire le tecnologie che lo implementano~~
- **Identificare gli attori**
- **Capire i loro obiettivi**
- ~~Stabilire quali requisiti non funzionali sono essenziali per un corretto utilizzo del sistema~~
- **Definire come l'attore raggiunge il proprio obiettivo**