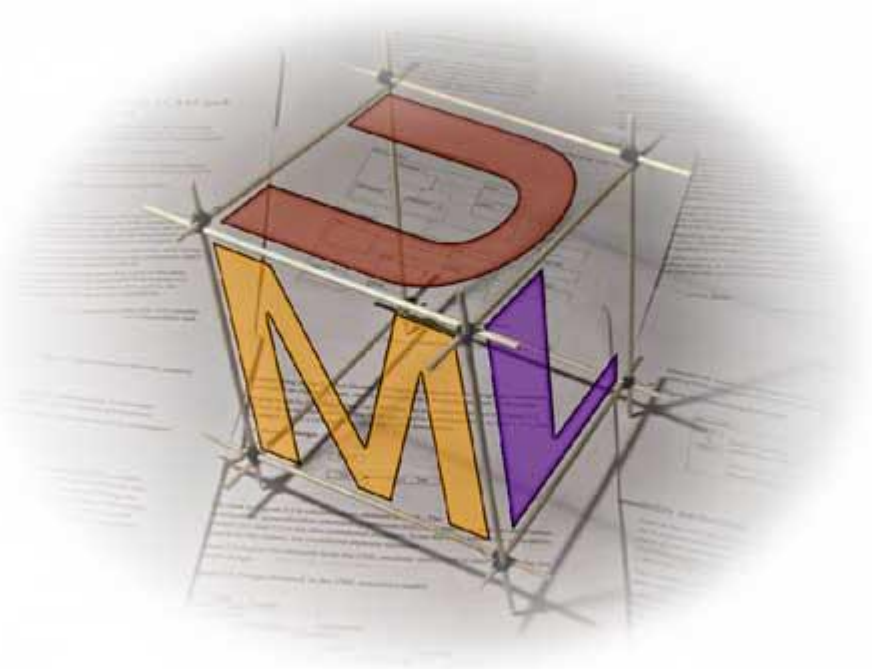
UML e i casi d'uso

Sintassi e Linee Guida

Dr. Andrea Baruzzo

andrea.baruzzo@dimi.uniud.it


UML

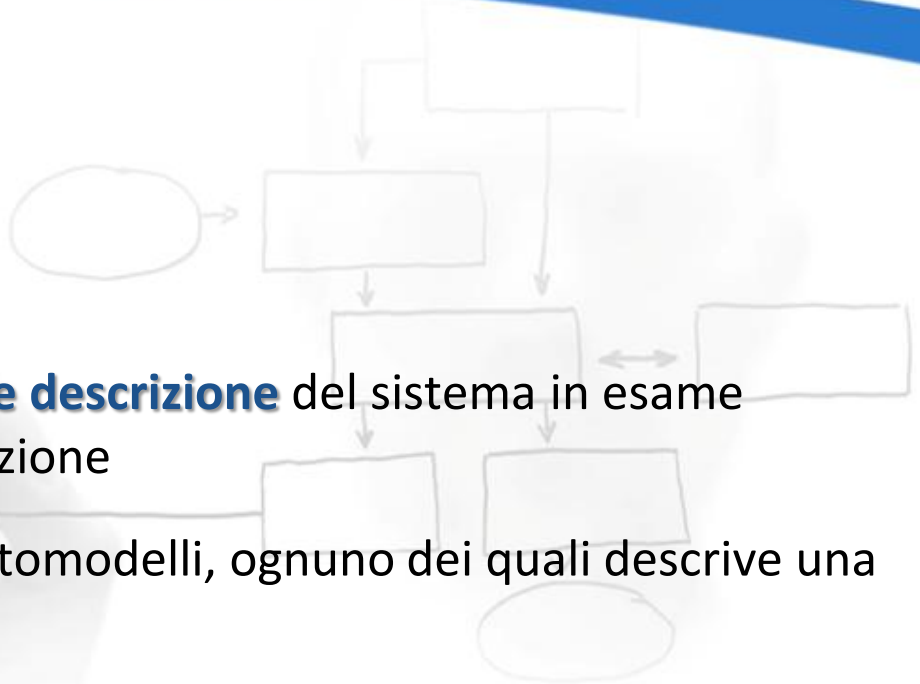
Agenda

1. Introduzione a UML: storia, approccio e motivazioni
   - Cos'è un modello (software)?
   - Perché creiamo modelli? Cos'è UML? Perché proprio UML?
   - UML non è un linguaggio di programmazione!
2. Modellare i requisiti funzionali con la tecnica dei casi d'uso
3. Gli elementi fondamentali dei casi d'uso
4. Relazioni nei diagrammi dei casi d'uso
5. Linee guida generali

UML: l'approccio…

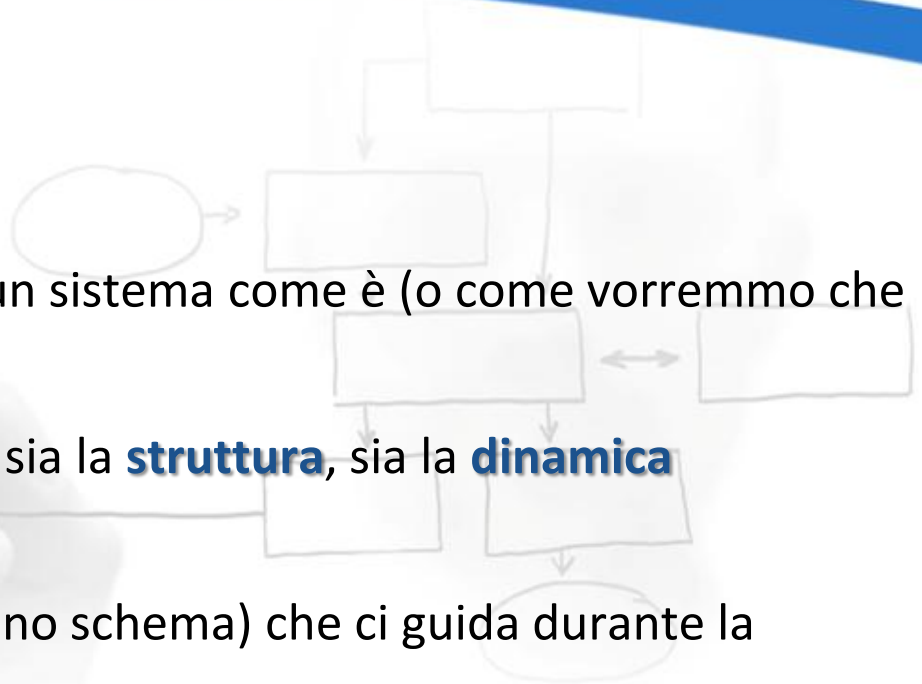
- Approccio pratico alla modellazione di sistemi software reali
- Orientamento alle tecnologie, agli strumenti e ai linguaggi largamente utilizzati in ambito industriale
- Costruzione di rappresentazioni grafiche (modelli) indipendenti dal linguaggio di implementazione
  - Beh … quasi! Si suppone un ambiente di sviluppo OO
    - Sistemi software target: sistemi object-oriented
    - Linguaggi di programmazione target: C++, C#, Java
    - Linguaggio di modellazione: UML
- Paradigma Model-Driven Development
  - Forte impatto su analisi, design e documentazione
  - Crescente impatto sulle attività di sviluppo (forward-reverse engineering) and testing

Cos'è un modello



- Una **semplificazione** della realtà
- Un'**accurata** e (possibilmente) **parziale descrizione** del sistema in esame espressa ad un qualche livello di astrazione
- Un modello è composto da diversi sottomodelli, ognuno dei quali descrive una certa **vista** (o prospettiva) del sistema
- Un modello non deve essere necessariamente completo
- Un modello è espresso in un qualche linguaggio ad un qualche livello di astrazione
- Un modello è più di una semplice descrizione
  - È una *rappresentazione analogica* degli elementi che esprime

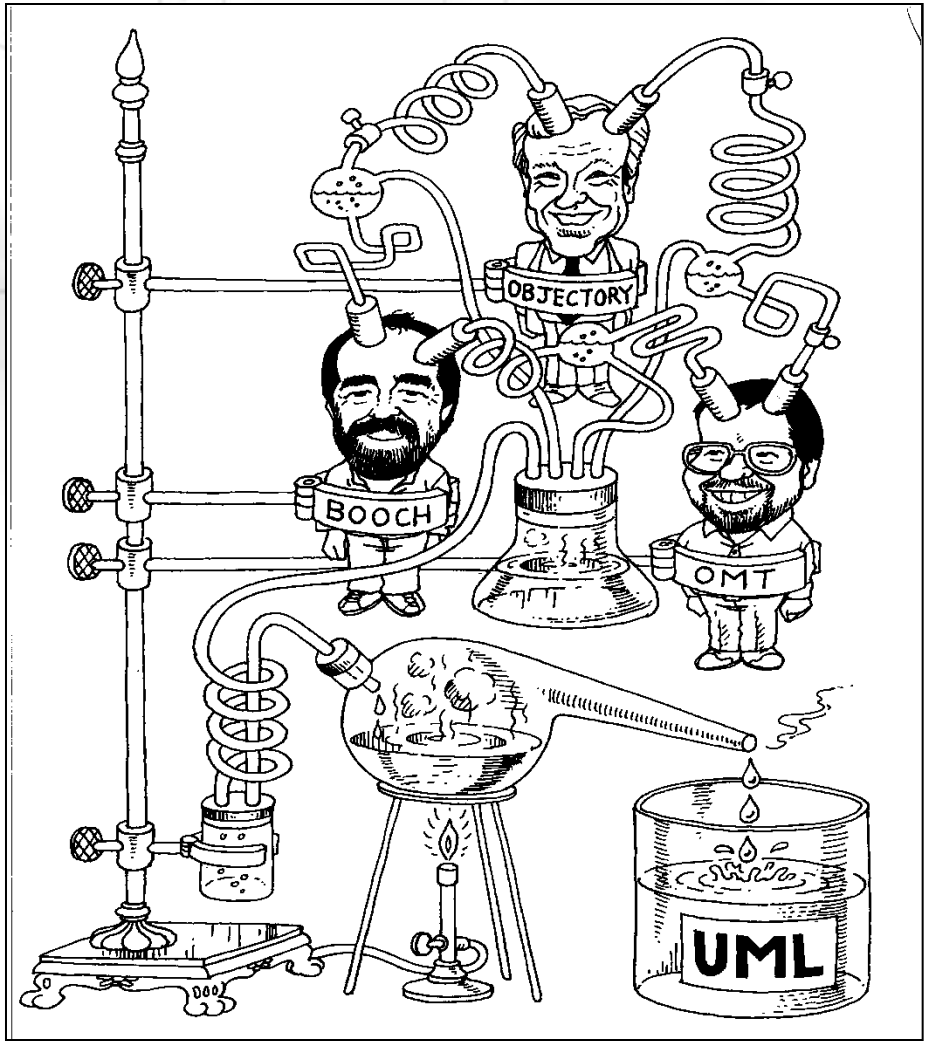
Perché creiamo modelli?



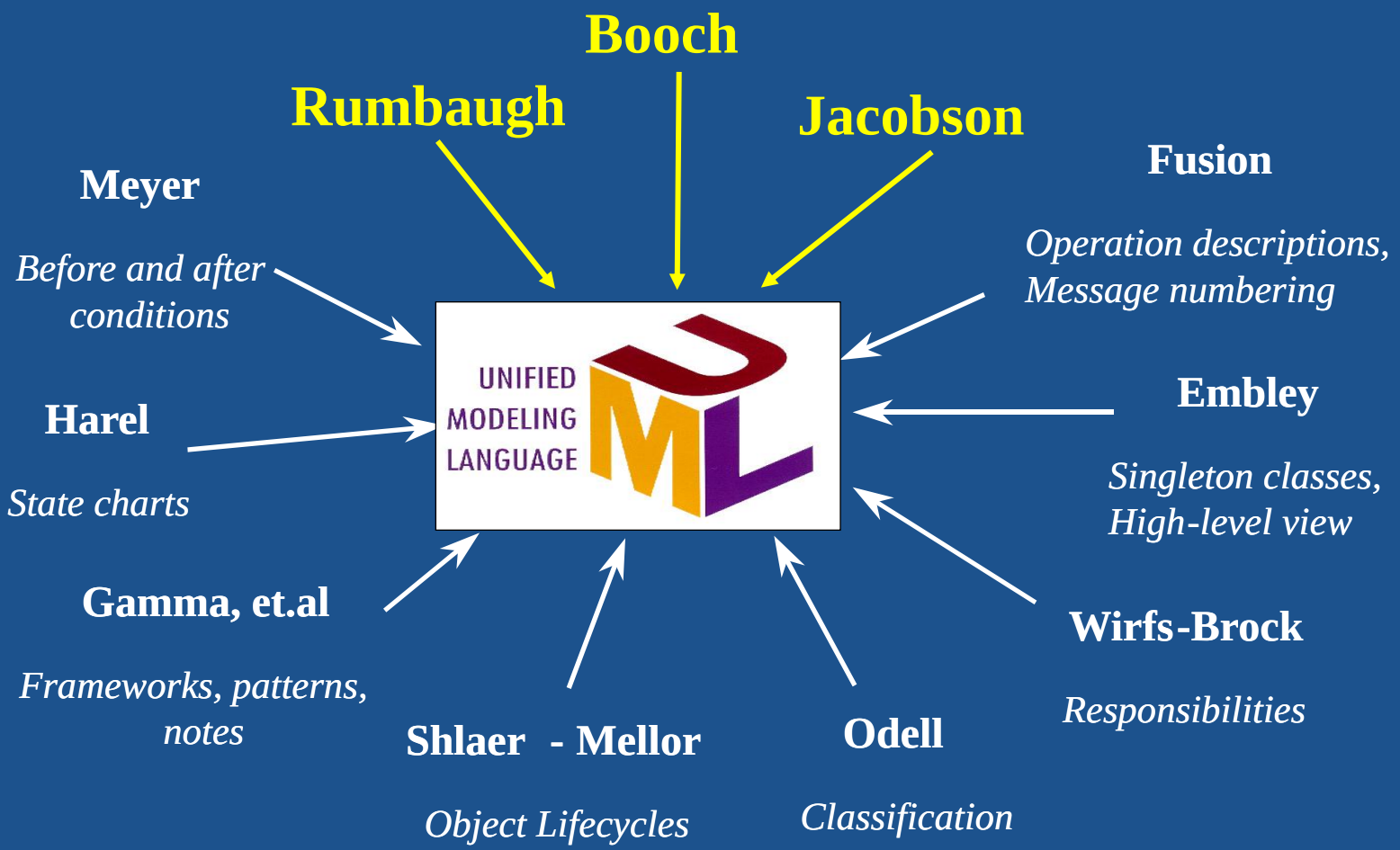
- Un modello permette di **visualizzare** un sistema come è (o come vorremmo che fosse)
- Un modello ci permette di specificare sia la **struttura**, sia la **dinamica** (comportamento) di un sistema
- Un modello ci fornisce un **template** (uno schema) che ci guida durante la costruzione del sistema
- I modelli **documentano** le decisioni che prendiamo

Cos'è UML

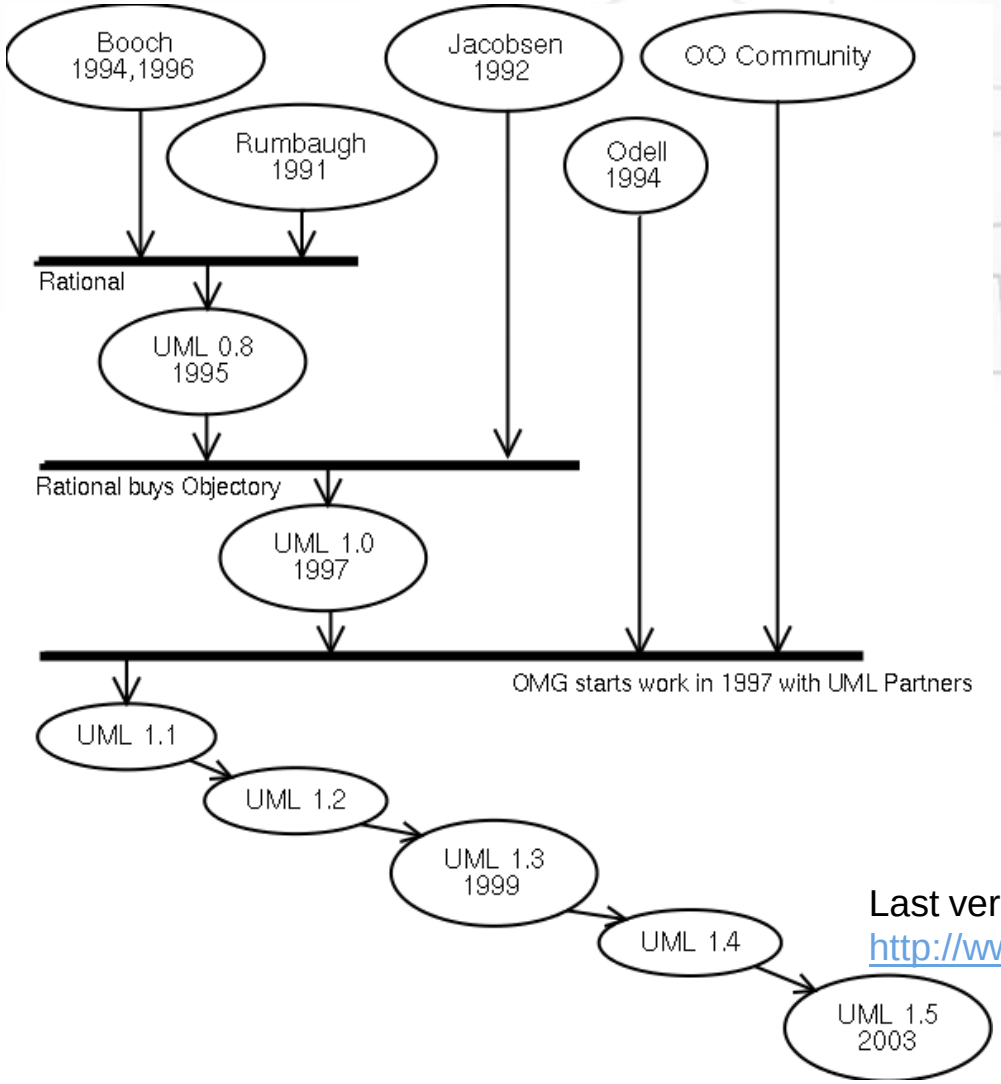
- Vari predecessori ("methods war")
  - OMT, Booch, Meyer, Martin, …
- UML è un linguaggio di modellazione che unifica tre precedenti metodi: Booch, Objectory (Jacobson), OMT (Rumbaugh)
- UML is a language for
  - Visualize,
  - Specify,
  - Build,
  - Document…
- … software artifacts
- Standard, unificato, …
  - Linguaggio per Analisi e Progetto



Cos'è UML (continua)



UML: la storia… (continua)



Last version:
http://www.omg.org/spec/UML/2.1.2/

Perché UML?

- **Qualità** del software!
- La **globalizzazione** sta cambiando il modo in cui il software viene progettato
- La singola parola che meglio descrive I benefici derivanti dall'utilizzo di UML è **comunicazione**
  - A failure to communicate during the development process can lead to disaster, and a great deal of money and time may be wasted
- UML è indipendente da metodologie e linguaggi di programmazione

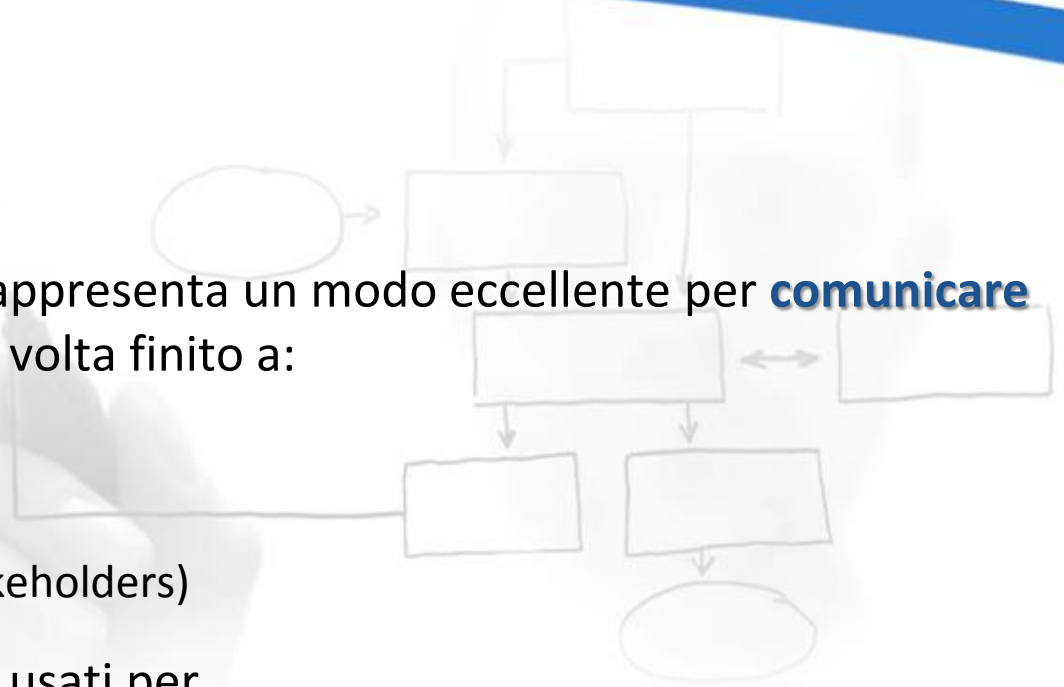
UML non è un linguaggio di programmazione

- Uno degli obiettivi di UML è **astrarre** dalla macchina fisica, cioé:
  - pensare in termini del **problema** e non della soluzione
  - (memoria, registri, CPU… livello di astrazione troppo basso per risolvere I problemi di grande complessità)
- UML serve per **"far parlare" il problema e il progetto**,
  - mostrando solo l'*informazione essenziale*
  - in base allo *scopo* del singolo diagramma
- UML fornisce **viste multiple** di uno stesso artefatto
  - adattando il livello di dettaglio in base al compito affrontato dal progettista

Agenda

1. Introduzione a UML: storia, approccio e motivazioni
2. Modellare i requisiti funzionali con la tecnica dei casi d'uso
3. Gli elementi fondamentali dei casi d'uso
4. Relazioni nei diagrammi dei casi d'uso
5. Linee guida generali

Diagrammi dei casi d'uso



- Un **diagramma dei casi d'uso** rappresenta un modo eccellente per **comunicare** ciò che il sistema deve fare una volta finito a:
  - Management
  - Clienti e utenti
  - Altro personale non tecnico (stakeholders)
- I diagrammi dei casi d'uso sono usati per …
  - Modellare il *contesto* di un sistema
  - Modellare I *requisiti* (funzionali) di un sistema
- I casi d'uso descrivono il sistema visto dalla **prospettiva dell'utente** che lo deve utilizzare
  - Che funzioni ci sono?
  - Quale valore all'utente fornisce ciascuna funzione?

I concetti fondamentali nei diagrammi dei casi d'uso

- Quattro concetti chiave:
- **Caso d'uso** – rappresenta una specifica interazione tra un attore e il sistema. In UML rappresentato mediante un'ellisse etichettata col nome del caso d'uso.
- **Attore** – rappresenta un ruolo che caratterizza le interazioni tra utente e sistema. L'utente non è necessariamente umano; In UML rappresentato mediante un omino.
- **Scenario** – sequenza di azioni che definisce una particolare interazione tra attore e sistema (casi d'uso).
- **Descrizione testuale (specifica del caso d'uso)** – testo che descrive lo scenario, l'ordine temporale della sequenza di azioni, l'eventuale trattamento degli errori, gli attori coinvolti.

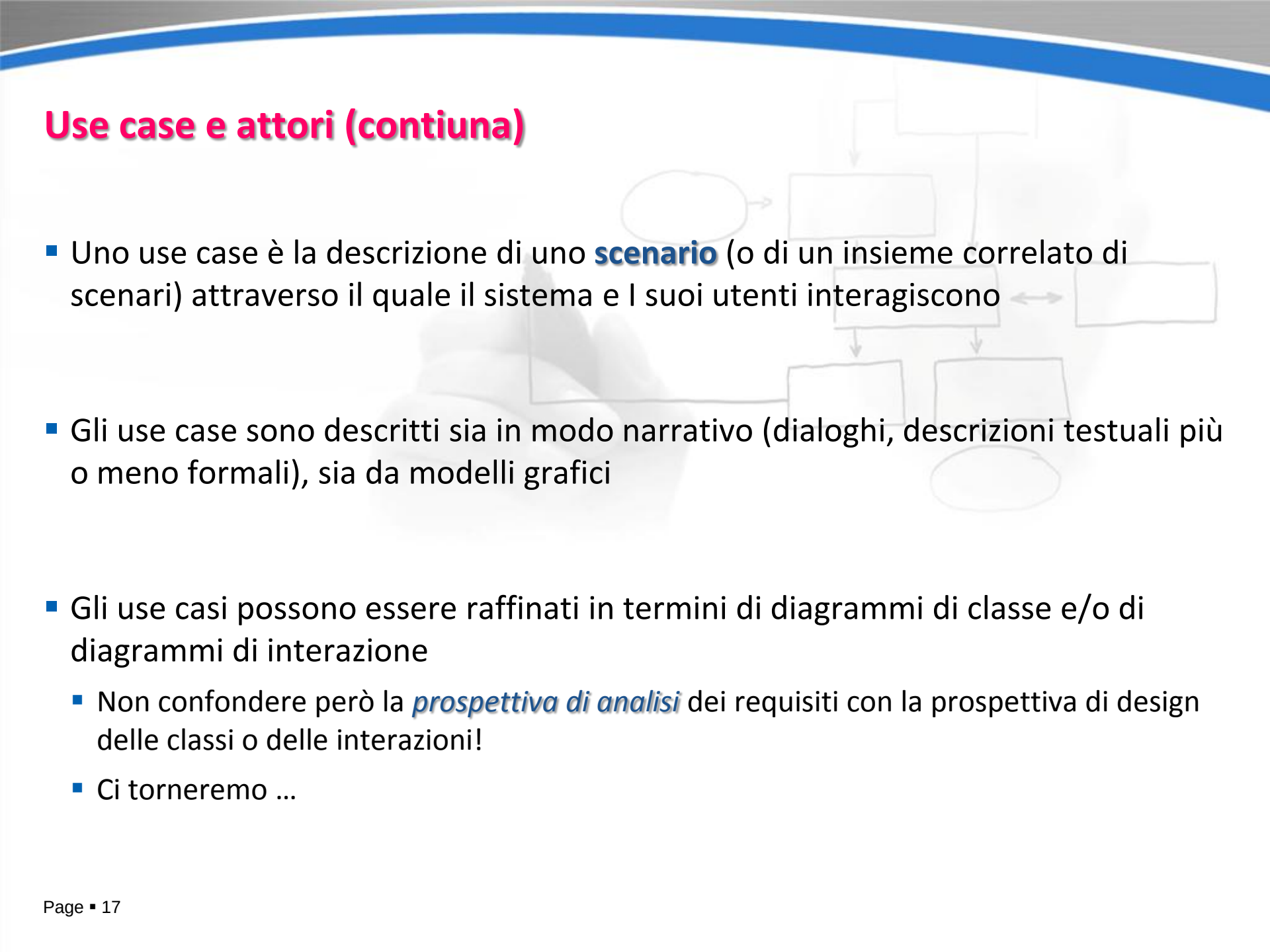
Agenda

1. Introduzione a UML: storia, approccio e motivazioni
2. Modellare i requisiti funzionali con la tecnica dei casi d'uso
3. Gli elementi fondamentali dei casi d'uso
4. Relazioni nei diagrammi dei casi d'uso
5. Linee guida generali

Use case e attori

- Uno **use case** è la descrizione di una o più sequenze di azioni e varianti che un sistema compie per produrre un risultato osservabile da un attore esterno
  - L'attore esterno tra vantaggio dallo use case
  - Lo use case ha *valore* per l'attore poiché risolve un problema o fornisce comunque un beneficio tangibile
- Uno use case descrive il sistema in termini di ciò che esso fa (**what**), non di come lo fa (how)
- Uno use case di solito rappresenta una **parte sostanziale di una funzionalità** percepita dall'utente finale

Use case e attori (contiuna)



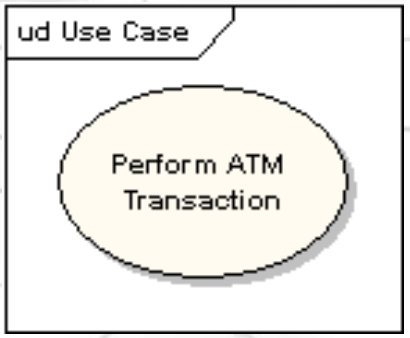
- Uno use case è la descrizione di uno **scenario** (o di un insieme correlato di scenari) attraverso il quale il sistema e I suoi utenti interagiscono

- Gli use case sono descritti sia in modo narrativo (dialoghi, descrizioni testuali più o meno formali), sia da modelli grafici

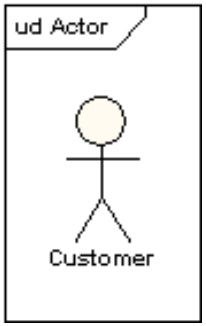
- Gli use casi possono essere raffinati in termini di diagrammi di classe e/o di diagrammi di interazione
  - Non confondere però la *prospettiva di analisi* dei requisiti con la prospettiva di design delle classi o delle interazioni!
  - Ci torneremo …

Use case e attori (contiuna)

- Un **attore** è qualsiasi persona o sistema esterno che deve interagire con il sistema in esame
- Gli attori NON sono parte del sistema da sviluppare

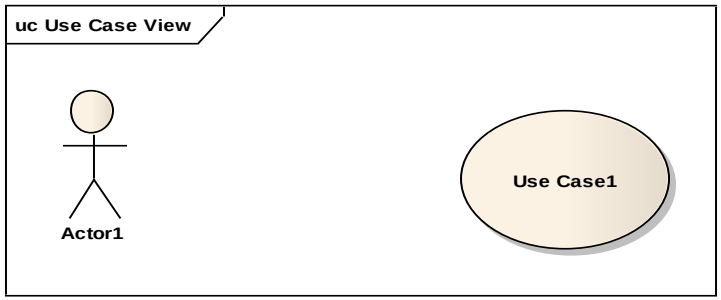


- In UML, uno use case è graficamente rappresentato da un **ovale**, mentre un attore viene rappresentato da un **omino** stilizzato



Associazioni tra casi d'uso e attori

- Una sola relazione: l'associazione
- Un'associazione tra un attore e uno use case indica che l'attore e lo use case comunicano tra loro, ognuno che può inviare e ricevere messaggi
  - Indica che un attore, svolgendo un particolare ruolo, interagisce con il sistema
- L'interazione è descritta dal caso d'uso associato (parte testuale)
- Non indica un flusso di dati, non è un data-flow diagram!



Agenda

1. Introduzione a UML: storia, approccio e motivazioni
2. Modellare i requisiti funzionali con la tecnica dei casi d'uso
3. Gli elementi fondamentali dei casi d'uso
4. Relazioni nei diagrammi dei casi d'uso
5. Linee guida generali

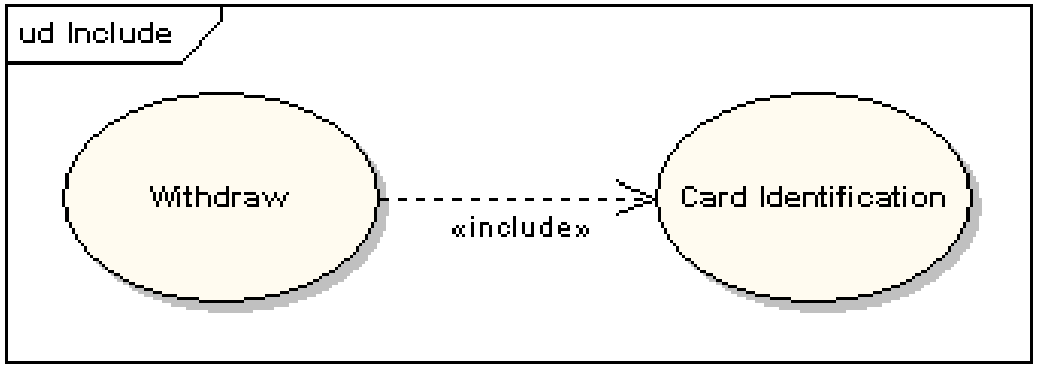
Relazioni tra casi d'uso

- Poche relazioni: modello semplice, senza grossa complessità sintattica
- Relazione di **inclusione** «include»
- Relazione di **generalizzazione**
- Relazione di **estensione** «extend»

- Cerchiamo di capire bene però queste poche relazioni

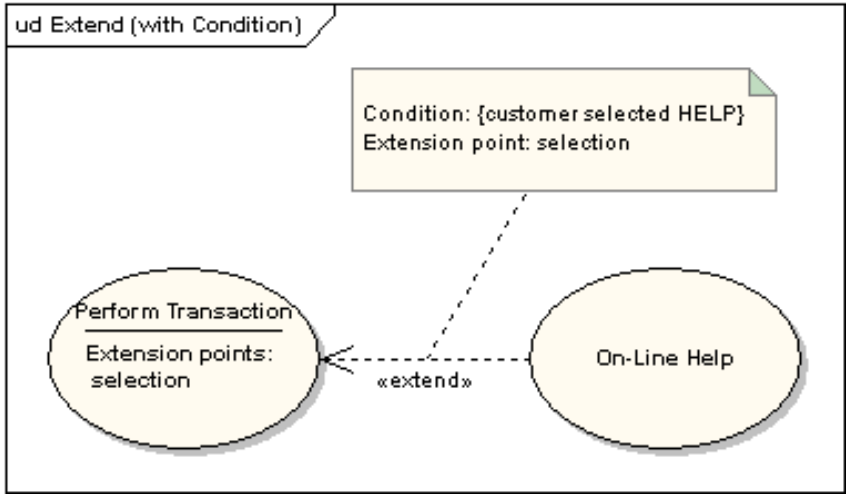
Relazioni di dependenza tra casi d'uso

- Inclusione **«include»**
  - Indica che il caso d'uso principale incorpora esplicitamente il comportamento di un altro caso d'uso subordinato
    - Il caso d'uso principale indica l'esatto punto in cui il caso d'uso subordinato viene incluso
    - Al termine dell'esecuzione del caso d'uso subordinato, il caso d'uso principale riprende dal punto in cui è stato sospeso



Relazioni di dependenza tra casi d'uso - Estensione

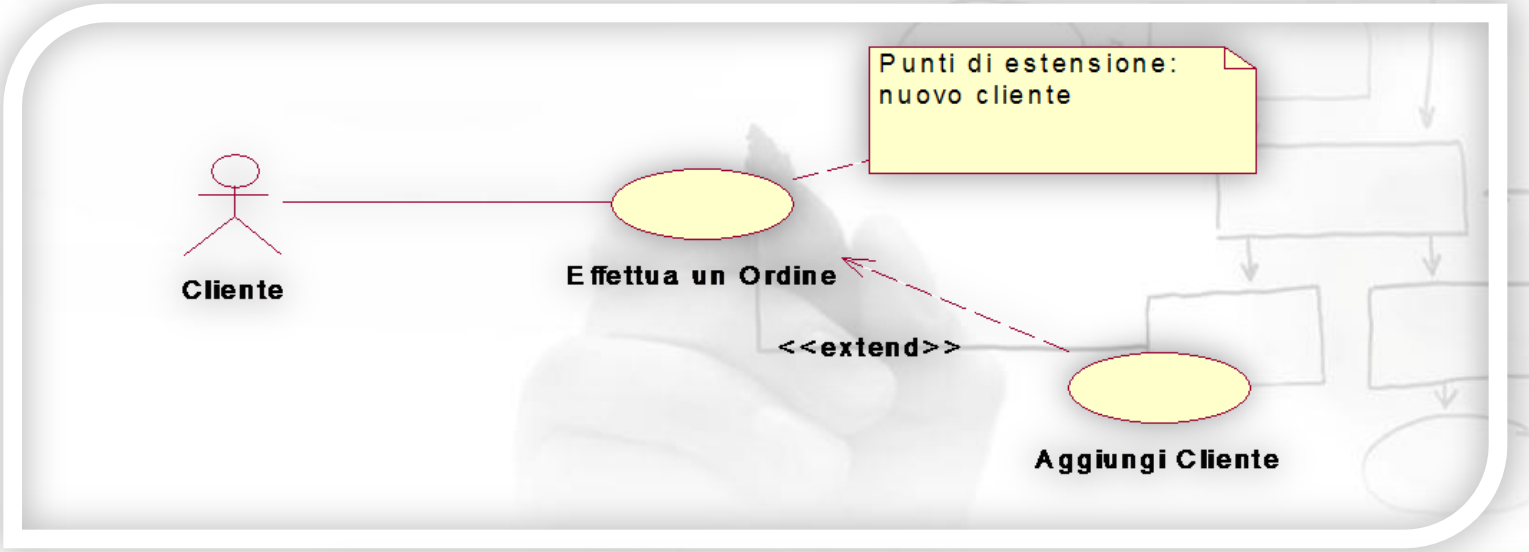
- Estensione **«extend»**
  - Indica che il caso d'uso subordinato estende il comportamento del caso d'uso principale, aggiungendo la logica necessaria per gestire eccezioni, flussi di lavoro alternativi, ecc.
    - Il caso d'uso principale indica l'esatto punto in cui il caso d'uso subordinato viene incluso (detto "punto di estensione")
    - Al termine dell'esecuzione del caso d'uso subordinato, il caso d'uso principale riprende dal punto in cui è stato sospeso
  - **Punti di estensione**
    - Condizioni di attivazione

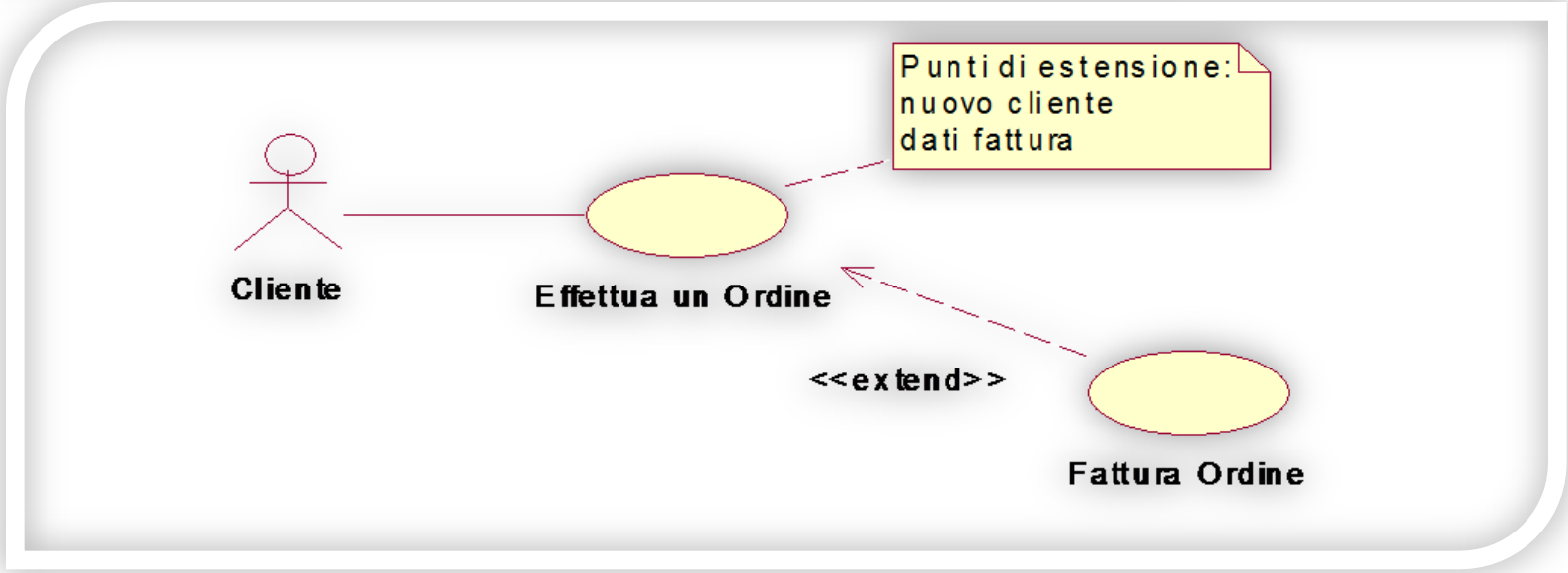


Relazioni di dependenza tra casi d'uso - Estensione

- Da non confondere con l'ereditarietà (estensione in Java)
- Il caso d'uso estensione continua il comportamento del caso d'uso di base inserendovi delle azioni
- Il caso d'uso di base dichiara *tutti* i possibili punti di estensione, anche a livello grafico, generalmente con una nota o con una descrizione testuale all'interno dell'ovale del caso d'uso base
- Simile alla gestione degli interrupt hardware (gestione eccezioni)
- È una "vera" estensione solo se serve a completare il (una parte del) caso d'uso principale
  - L'estensione "vive" all'interno del caso d'uso principale; in altre parole, l'esecuzione dell'estensione è ancora parte dell'esecuzione del caso d'uso principale

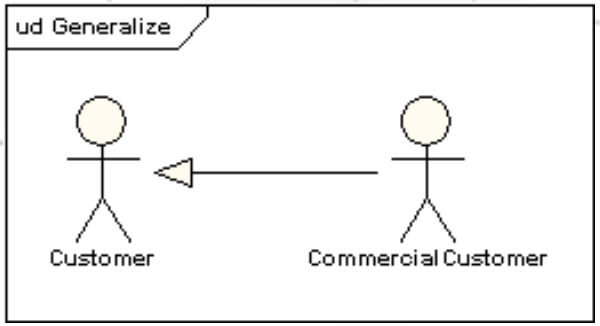
Altri esempi di estensioni

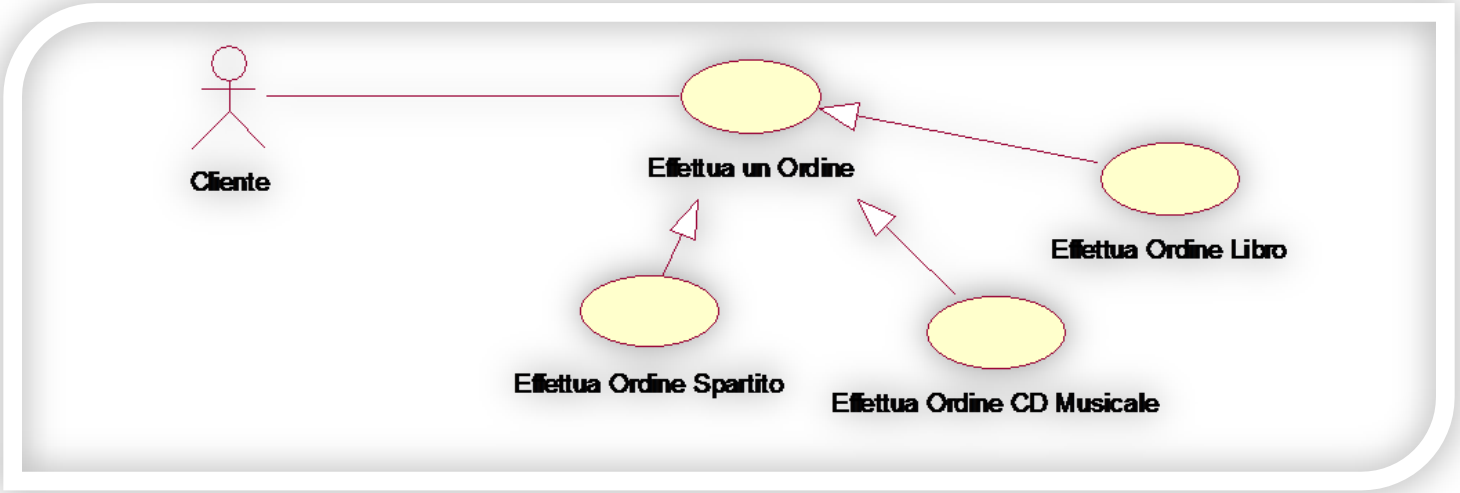




Relazioni di dependenza tra casi d'uso

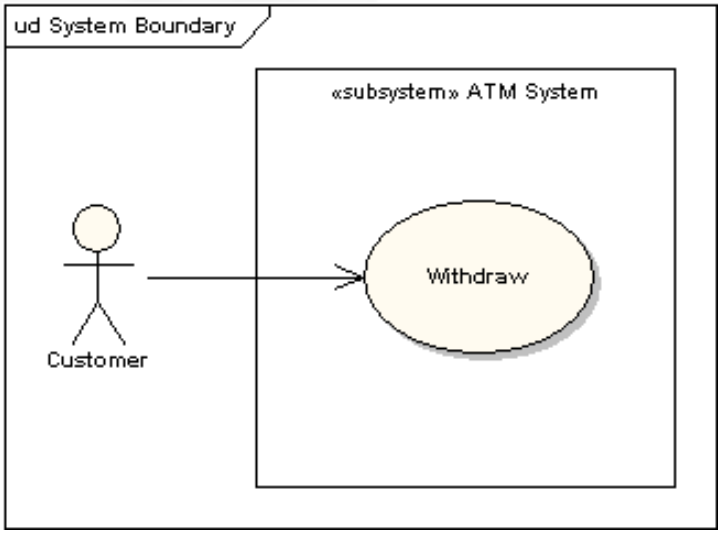
- Generalizzazione
  - Specifica gerarchie (categorie) di attori e/o casi d'uso
    - (concettualmente, è la classica relazione di ereditarietà)





System boundary

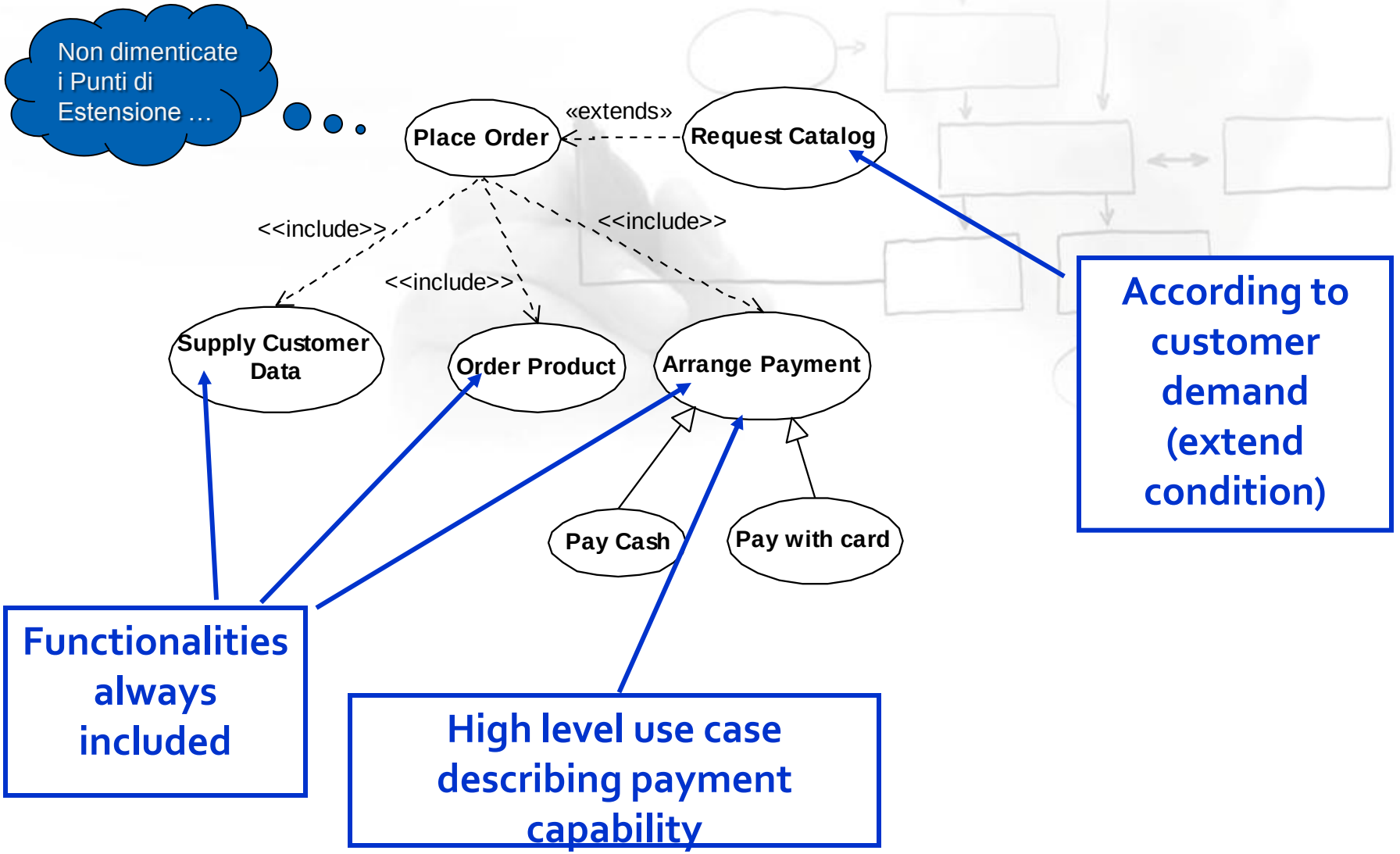
- E’ buona pratica visualizzare gli use case come se fossero racchiusi dentro ad una scatola (system boundary) che descrive il confine tra il sistema da sviluppare e il resto del mondo
  - Gli attori che interagiscono con il sistema vengono rappresentati all’esterno della scatola



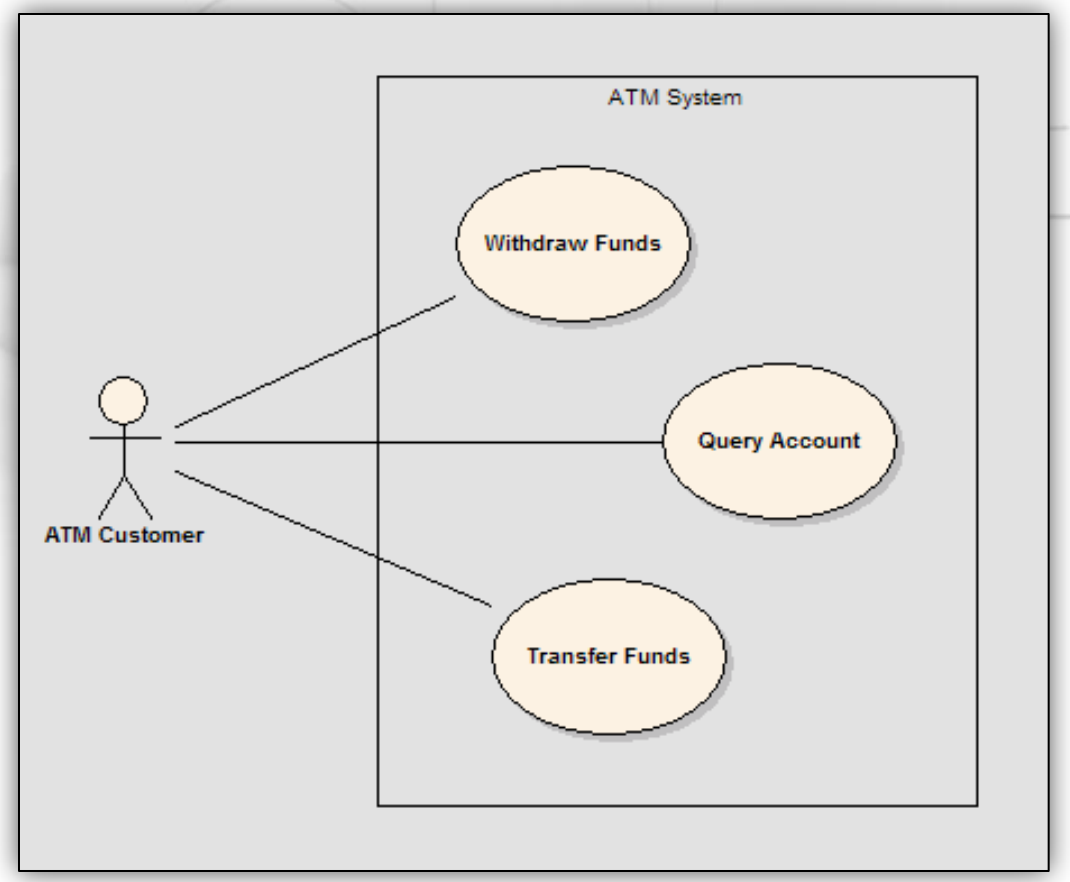
Riassumendo …

- Perché è importante modellare i casi d'uso?
  - Primo momento nel ciclo di vita del software in cui costruiamo delle rappresentazioni (talvolta semi-formali) del software
  - Esplicitare e comunicare a tutti gli stakeholder i requisiti funzionali del sistema– In generale, analizzare, identificare, descrivere gli usi tipici del sistema da parte dei suoi utilizzatori
  - Considerare anche tutta l'eventuale logica derivante dalla gestione di errori, eccezioni, flussi alternativi
  - Validare i requisiti utente – Essendo il modello dei casi d'uso piuttosto semplice (pochi costrutti, semantica precisa, ma intuitiva), diventa un valido strumento per assicurarci di aver sviluppato un sistema software corrispondente alle vere necessità dell'utente.

Un esempio di diagramma dei casi d'uso con relazioni multiple



Esempio di scenario: Prelievo di denaro dal Bancomat (Withdraw funds)



Nome caso d'uso: Withdraw Funds.

Scope: ATM system

Goal (summary): L'utente dell'ATM preleva una specifica somma di denaro da un conto corrente bancario valido.

Dipendenze con altri casi d'uso: nessuna

Attori: Cliente del terminale ATM.

Template per la specifica di un caso d'uso

TEMPLATE DI BASE

- *Nome caso d'uso* – Ogni caso d'uso deve avere un nome; il nome esprime il goal dell'utente nell'utilizzo del sistema.
- *Goal (summary description)* – descrizione della funzionalità fornita dal sistema e che soddisfa una necessità dell'utente, ossia che è percepita dallo stesso utente come "valore".
- *Attori* – persona, dispositivo o altra entità esterna al sistema che interagisce con il sistema. Per ogni caso d'uso esiste sempre un attore primario che è colui che inizia il caso d'uso stesso.
- *Precondizioni* – Condizioni che devono essere soddisfatte all'inizio del caso d'uso. Rappresentano le "garanzie minime" che devono essere soddisfatte per poter attivare lo scenario di utilizzo del sistema (Garanzie fornite dagli attori al sistema).
- *Trigger* – evento trigger che attiva il caso d'uso
- *Descrizione (main success scenario o scenario principale)* – descrizione della sequenza di interazioni più comune tra gli attori e il sistema. In particolare viene descritta la sequenza principale che porta alla conclusione del caso d'uso con successo. La descrizione è definita in termini di input forniti dall'attore e di risposta del sistema. Il sistema è trattato secondo un modello di tipo black-box, concentrandosi su cosa esso fa in risposta agli input, e non su come internamente queste risposte vengano prodotte.
- *Alternative (estensioni)* – descrizioni delle variazioni dalla sequenza di passi tipica del main success scenario. Tali alternative estendono lo scenario principale. La gestione delle eccezioni è un esempio tipico di tali estensioni. Non tutte le alternative portano necessariamente ad un fallimento del caso d'uso.

- *Postcondizioni* – Condizioni sempre soddisfatte al termine del caso d'uso (Garanzie fornite dal sistema agli attori)

Scenario descritto mediante template testuale

Trigger: inserimento di una carta di credito nel lettore del terminale ATM

Precondizioni: Il terminale dell'ATM è in attesa (idle) e visualizza un messaggio di "Benvenuto".

Descrizione (main success scenario o scenario principale):

1. Il cliente inserisce nel lettore del terminale ATM la sua carta di credito.
2. Il sistema riconosce la carta e ne legge il numero (card number).
3. Il sistema visualizza a terminale la richiesta di inserimento del PIN utente.
4. Il cliente inserisce il suo PIN.
5. Il sistema verifica che la carta non è scaduta, né è stata rubata o risulta smarrita.
6. Il sistema verifica che il PIN inserito dal cliente corrisponda con quello della carta.
7. Il sistema controlla che il conto corrente sia accessibile mediante l'utilizzo della carta.
8. Il sistema visualizza il conto corrente del cliente e visualizza le possibili operazioni che il cliente può effettuare (Prelievo, Ultimi Movimenti, Saldo Disponibile, Altri Servizi).
9. Il cliente seleziona il conto corrente desiderato (nel caso il cliente ne abbia più di uno) e seleziona l'operazione Prelievo.
10. Il sistema visualizza la richiesta dell'ammontare da prelevare.
11. Il cliente inserisce l'ammontare da prelevare.
12. Il sistema verifica che il conto corrente contenga sufficienti fondi per consentire la conclusione dell'operazione e che non sia stato superato il limite massimo giornaliero per il prelievo.
13. Il sistema autorizza il prelievo.
14. Il sistema emette il denaro, registrando la transazione sul conto corrente.
15. Il sistema stampa la ricevuta mostrando il numero della transazione, il tipo di operazione effettuata, la quantità di denaro prelevata e il saldo del conto corrente.
16. Il sistema espelle la carta.
17. Il sistema ritorna nello stato iniziale di attesa (idle), visualizzando il messaggio di "Benvenuto"

Scenario descritto mediante template testuale (continua)

Alternative (estensioni):

2a. Se il sistema non riconosce la carta dell'utente, quest'ultima viene espulsa dal lettore.

5a. Se la carta risulta scaduta, il sistema la confisca.
5b. Se la carta risulta smarrita, il sistema la confisca.
5c. Se la carta risulta rubata, il sistema la confisca.

6a. Se il cliente ha inserito un PIN che non corrisponde con quello della carta, il sistema visualizza una nuova richiesta di inserimento del PIN. Se il cliente inserisce per tre volte un PIN errato, il sistema confisca la carta.

7a. Se il sistema verifica che il numero di conto non è valido, viene visualizzato un messaggio di errore e la carta viene espulsa dal lettore.

12a. Se non ci sono sufficienti fondi nel conto corrente, il sistema visualizza un messaggio di errore ed espelle la carta. Il terminale ATM viene riportato allo stato di attesa (idle) e viene visualizzato il messaggio di "Benvenuto".
12b. Se il limite giornaliero massimo di prelievo è stato già raggiunto, il sistema visualizza un messaggio di errore ed espelle la carta. Il terminale ATM viene riportato allo stato di attesa (idle) e viene visualizzato il messaggio di "Benvenuto".

14a. Se lil terminale ATM non ha sufficienti fondi per completare la transazione, il sistema visualizza un messaggio di errore, espelle la carta, il terminale ATM viene riportato allo stato di attesa (idle) e viene visualizzato il messaggio di "Benvenuto".
14b. Se il cliente seleziona da terminale il pulsante Cancel, il sistema cancella la transazione ed espelle la carta. Il terminale ATM viene riportato allo stato di attesa (idle) e viene visualizzato il messaggio di "Benvenuto".

Postcondizioni: L'utente ha ottenuto la somma di denaro che aveva richiesto di prelevare.

Questioni aperte: nessuna.

Agenda

1. Introduzione a UML: storia, approccio e motivazioni
2. Modellare i requisiti funzionali con la tecnica dei casi d'uso
3. Gli elementi fondamentali dei casi d'uso
4. Relazioni nei diagrammi dei casi d'uso
5. Linee guida generali

Linee guida: Scegliere il nome del caso d'uso

- Iniziare il nome di un caso d'uso usando un verbo fortemente descrittivo
- Il caso d'uso deve essere espresso da un verbo (si tratta di un'azione!), non un entità (tipico nome riservato ai dati!!!)
- Evitare verbi troppo generici come "fare", "processare", "eseguire"
- Usare il vocabolario del dominio del problema perché è user-oriented (vs. technical-oriented)

Linee guida: Quando includere, quando estendere

- Usare «include» per indicare che un caso d'uso complesso chiama (include il comportamento di) un casi d'uso più semplice
- Nel testo specificare l'ordine con cui i casi d'uso più semplici vengono eseguiti per completare il caso d'uso complesso
- Usare «extend» per introdurre una logica alternativa (gestione eccezioni, patch) all'interno di un altro caso d'uso

- *Inclusione*:
  - Aumenta il comportamento di un caso d'uso base
  - Il caso d'uso incluso è *sempre* utilizzato per aumentare il comportamento del caso d'uso base

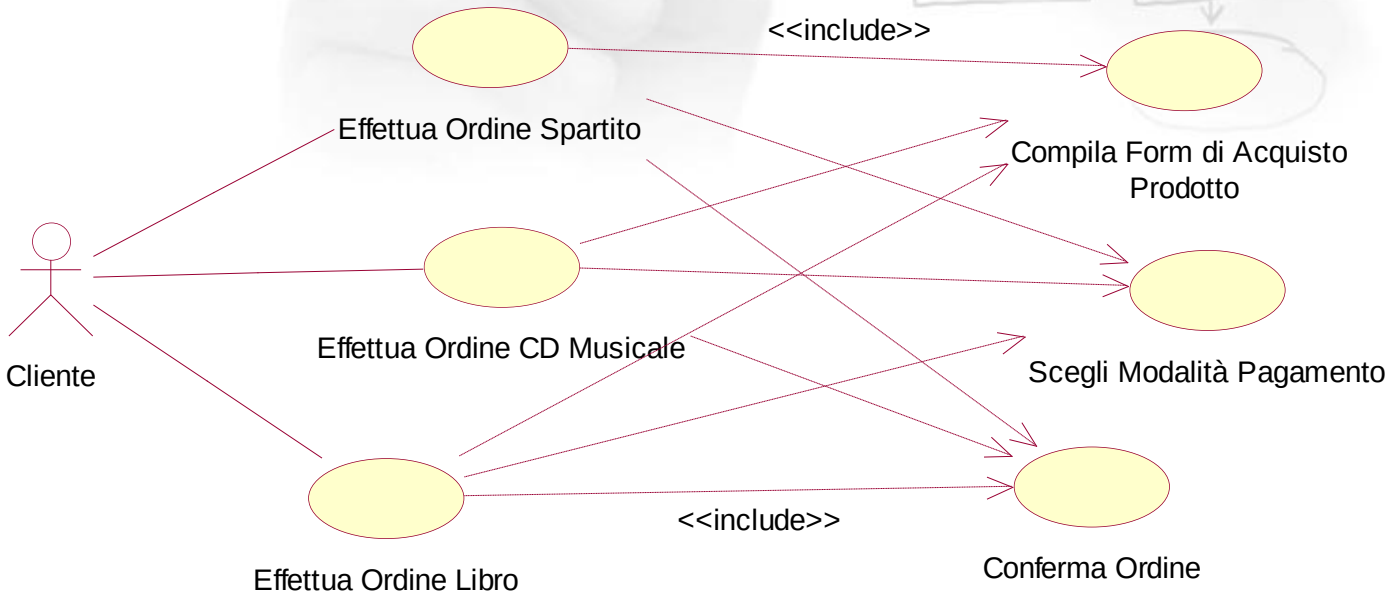
- *Estensione*
  - Aumenta il comportamento di un caso d'uso base
  - Il caso d'uso estensione *può* essere utilizzato per aumentare il comportamento del caso d'uso base

Linee guida: Quando includere, quando estendere (cont.)

- *Inclusione*:
  - Il caso d'uso base decide quando invocare il caso d'uso incluso. Il caso d'uso incluso non è a conoscenza del caso d'uso base
  - La dipendenza rappresentata da una linea tratteggiata adornata dallo stereotipo «include» è orientata dal caso d'uso base a quello incluso, richiamando l'analogia di un programma che invoca una procedura
- *Estensione*
  - Il caso d'uso estensione decide quando inserire il proprio comportamento nel caso d'uso base. Il caso d'uso base non è a conoscenza del caso d'uso estensione
  - La dipendenza rappresentata da una linea tratteggiata adornata dallo stereotipo «extend» è orientata dal caso d'uso estensione a quello base, richiamando l'analogia di un interrupt software o di una callback function

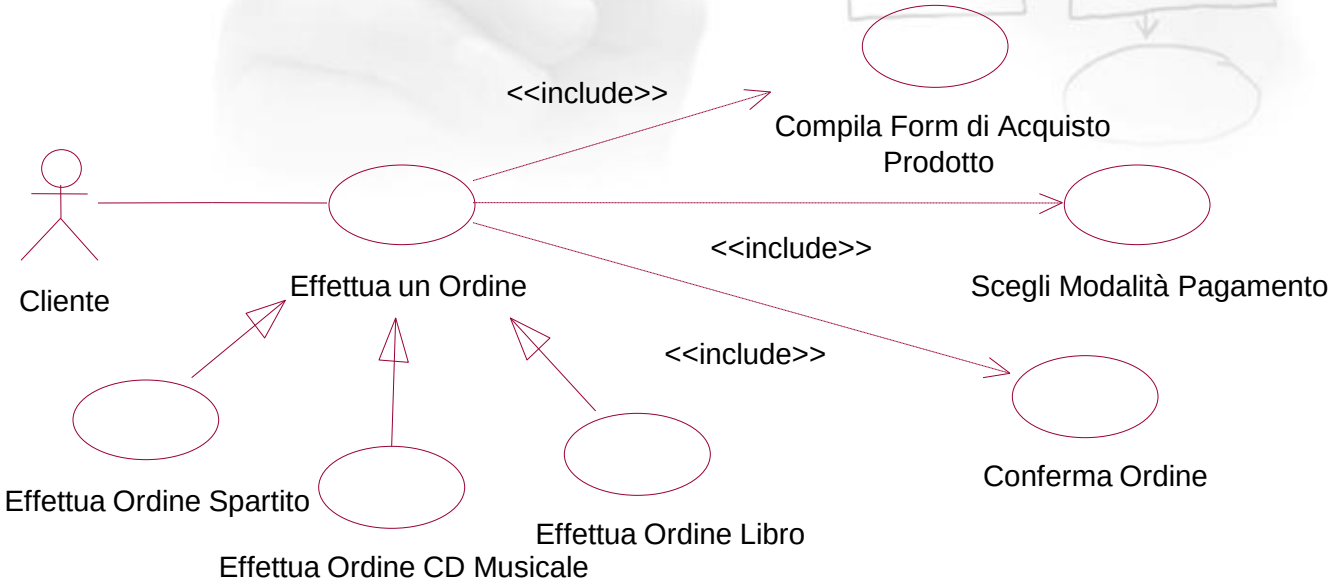
Linee guida: Quando generalizzare

- Usare la generalizzazione quando si vuole raggruppare più attività simili in una attività generica, semplificando il diagramma



Linee guida: Quando generalizzare (cont.)

- Compilare la form d'acquisto, scegliere la modalità di pagamento e confermare l'ordine sono attività valide per *ogni* tipo di ordine!



Bibliografia

- [Booch et al., 2005] Grady Booch et al. "*The Unified Modeling Language User Guide 2/E*", Addison-Wesley, 2005
- [Rumbaugh et al., 2004] J. Rumbaugh et al. "*The Unified Modeling Language Reference Manual 2/E*", Addison-Wesley, 2004
- [Fowler, 2003] Martin Fowler. "*UML Distilled 3/E*", Addison-Wesley, 2003
- [Larman, 2004] C. Larman. "*Applying UML and Patterns*", Addison-Wesley, 2004
- [Pender, 2003] Tom Pender. "*UML Bible*", Wiley&Sons, 2003

Domande?
Commenti?
Dubbi?