

# Esempio completo

Un esempio completo di

## Specifica-Analisi-Progetto-Sviluppo

Ci serve a vedere come si applicano i concetti visti

### Agenzia Turistica

Un sistema per la gestione delle prenotazioni di una agenzia  
turistica

(Per il problema vedere il testo)

PROVVISORIO  
Questo documento non è completo

# Contenuto/Metodo seguito

- Seguiremo il metodo introdotto in precedenza
  - Requisiti
    - Specifica requisiti
    - Modello di dominio
    - Casi d'uso
  - Analisi/Progetto preliminare
    - Analisi di robustezza
  - Progetto dettagliato
    - Diagrammi di sequenza
    - Responsabilità classi
  - Realizzazione

# Intermezzo:

# Specifica dei requisiti

(vedi slide)

# Analisi dei Requisiti del nostro sistema

- Scopo
  - La gestione delle prenotazioni dell'agenzia.
- Vincoli
  - sistema viene realizzato come installazione unica, per funzionare all'interno di una agenzia avente una sola sede.
  - Tutte le operazioni vengono effettuate dal solo personale dell'agenzia e non c'è necessità di distinguere i differenti impiegati.
    - equivale ad assumere che ci sia un solo attore.

# Requisiti

- [R1] Il sistema deve consentire l'inserimento di una nuova escursione nel programma dell'agenzia, in un qualunque momento.
  - [R1.1] Ogni escursione è caratterizzata da: data, tipo, descrizione, costo, numero massimo di partecipanti, optional possibili.
  - [R1.2] Le escursioni sono di 2 tipi: gite in barca e gite a cavallo. Costo e numero massimo di persone iscrivibili sono stabiliti all'atto dell'inserimento e possono variare da escursione a escursione. Pure i tipi di optional previsti da ogni escursione sono fissati all'atto dell'inserimento
  - [R1.3] I possibili tipi di optional sono 3: "Pranzo", "Merenda", "Visita". I loro costi sono fissi, indipendentemente dall'escursione cui si associano

# Requisiti

- [R2] Il sistema deve permettere in qualunque momento la modifica o l'eventuale eliminazione di ogni singola escursione. Si assume di trascurare le implicazioni e gli effetti indotti da una modifica o da una cancellazione, nel senso che non si producono avvisi per le persone registrate, non si calcolano gli eventuali rimborsi da effettuare, ecc..
- [R3] Il sistema deve permettere di registrare un partecipante ad una data escursione, consentendo, nel caso siano previsti, la scelta di eventuali optional; deve calcolare il costo dell'escursione comprensivo degli optional.

# Requisiti

- [R3.1] Un partecipante è caratterizzato attraverso il suo codice fiscale, nome cognome e indirizzo. I dati di un partecipante devono essere registrati alla prima prenotazione e non più cancellati dal sistema, anche nel caso in cui un partecipante si iscriva a una sola gita e poi si cancelli.
- [R4] Il sistema deve permettere in qualunque momento sia la cancellazione di un partecipante da una data escursione sia l'eventuale modifica del numero e del tipo di optional scelti da un dato partecipante in riferimento a una data escursione; nel caso di modifica degli optional scelti deve essere calcolato il nuovo costo risultante. Si assume di trascurare le implicazioni e gli effetti indotti da una cancellazione o da un cambio degli optional scelti (rimborsi, ecc.).

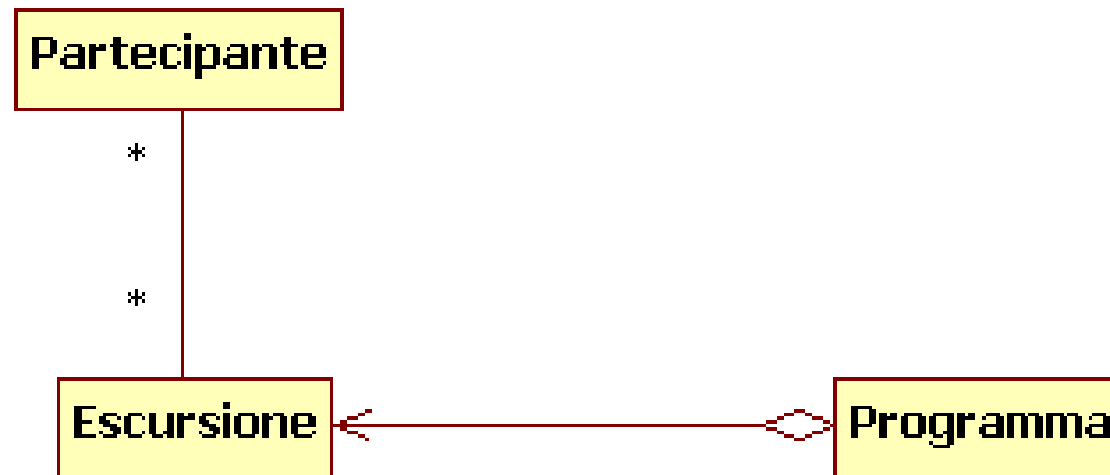
# Costruzione modello di dominio

- Si deve leggere il testo (meglio: la specifica dei requisiti) per individuare le entità del modello

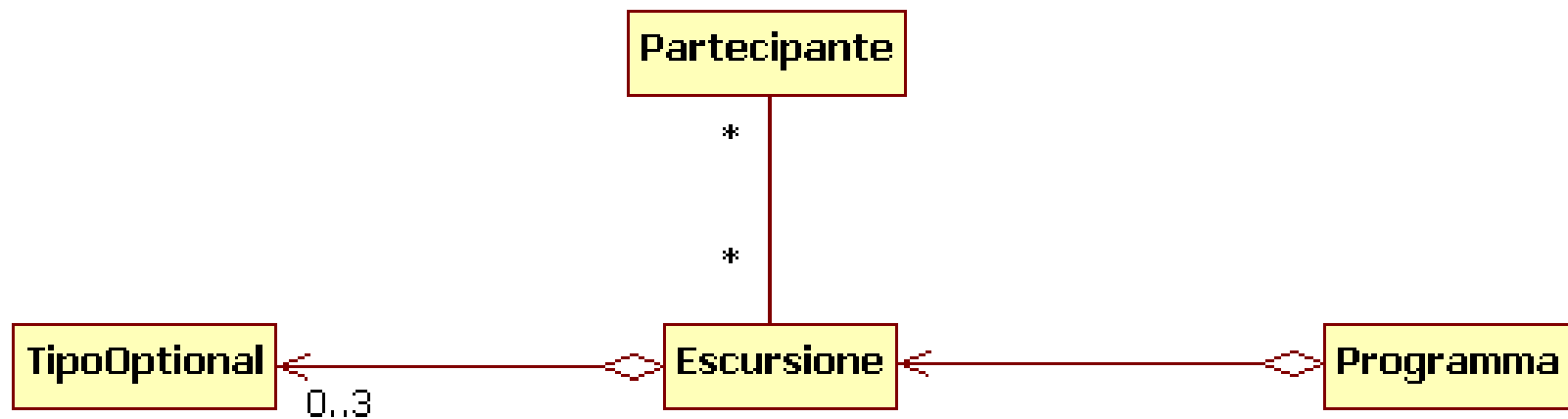




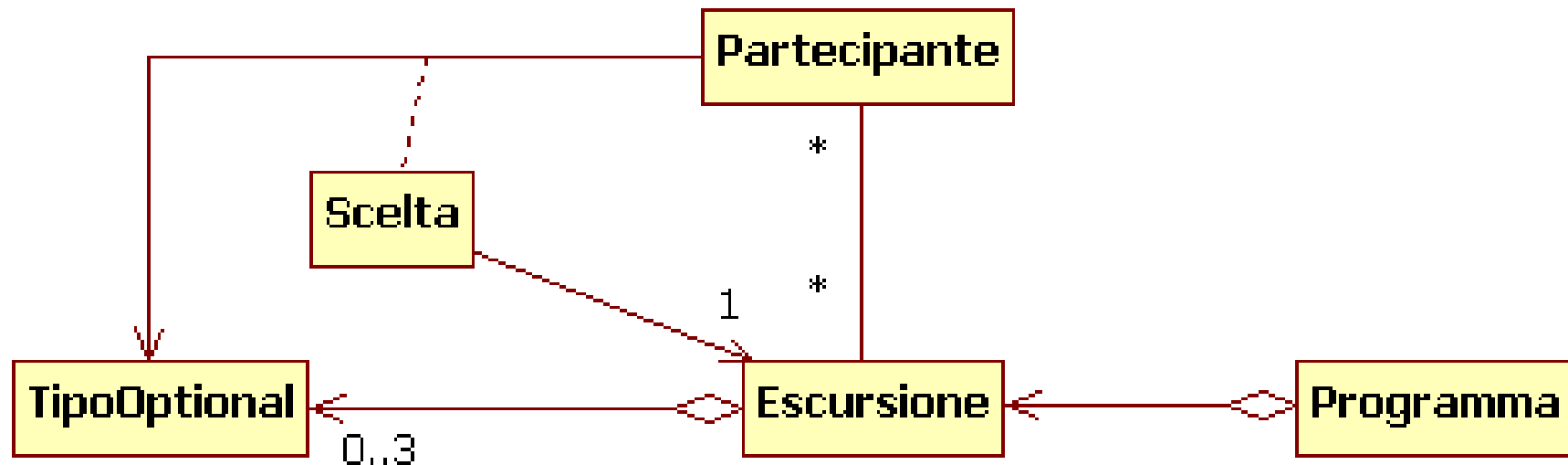
## .. Costruzione modello di dominio



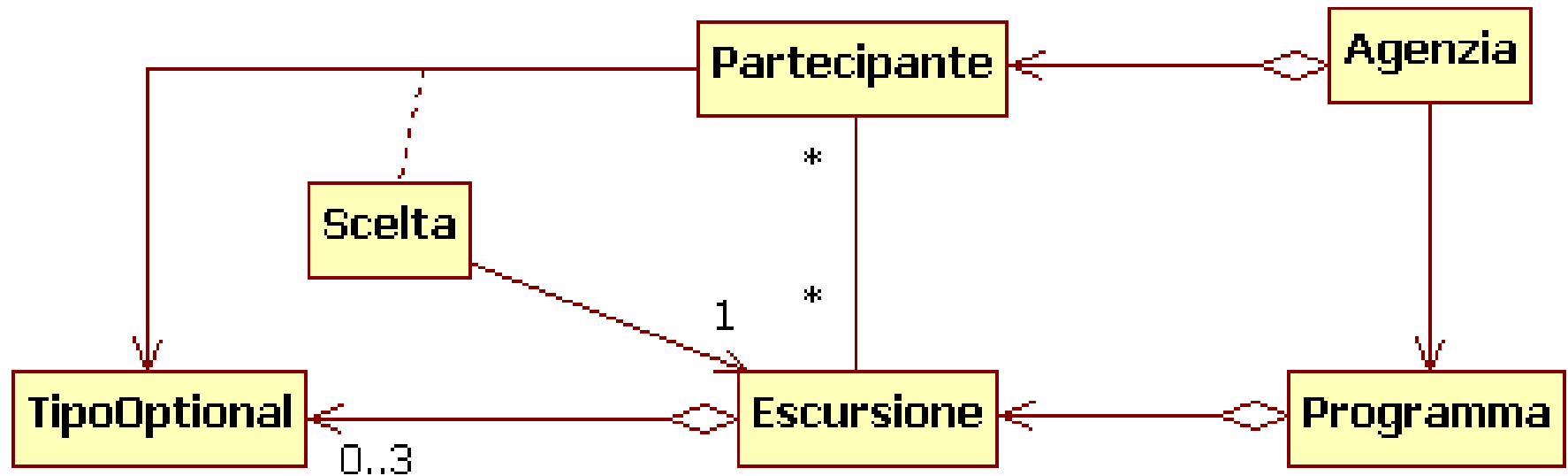
# ..Costruzione modello di dominio



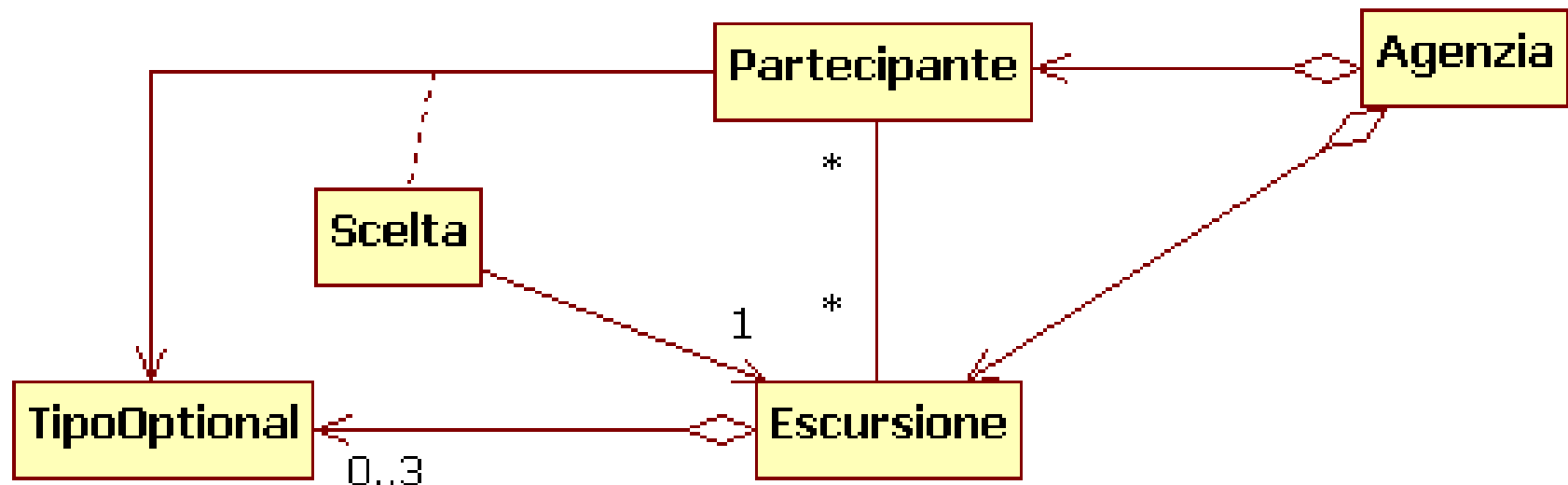
## ... Costruzione modello di dominio



# ..Costruzione modello di dominio



# Programma non serve a nulla



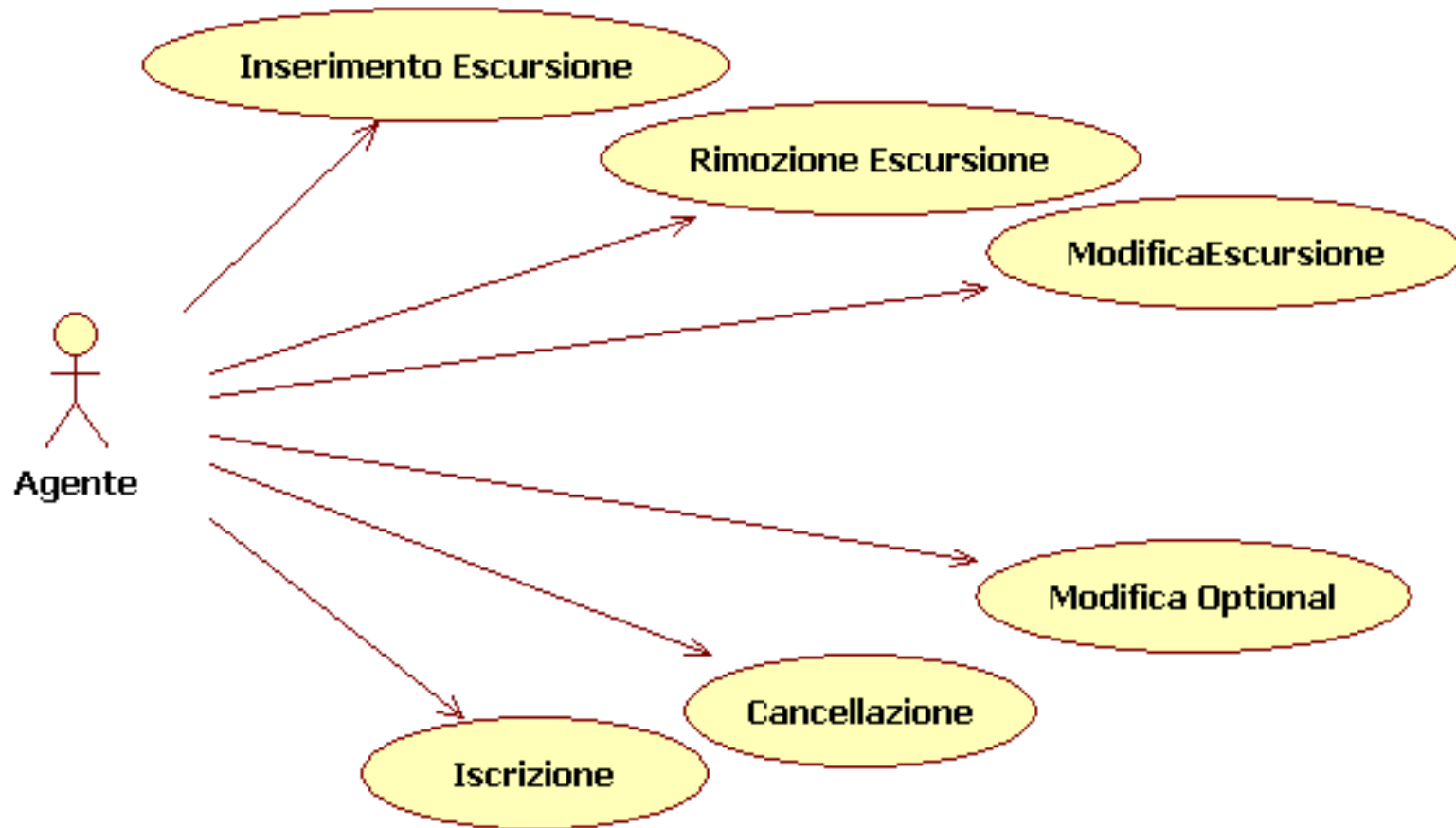
# Intermezzo:

# Analisi Casi d'Uso

(vedi slide)

# Analisi dei casi d'uso

## Diagramma (iniziale)



# CU1

CU1: Inserimento di una escursione a programma
Attore: Agente
Precondizione: Il sistema è idle e sullo schermo viene presentata la finestra con il menu principale (schermata "Menu Principale").
Sequenza eventi: <ol style="list-style-type: none"><li>1. il caso d'uso inizia quando l'attore clicca il bottone "Inserimento" sul "Menu Principale"</li><li>2. il sistema mostra una nuova schermata ("Inserimento Escursioni"), contenente vari <i>widgets</i> (campi, checkbox, listbox, pulsanti ecc.), adeguati all'inserimento dei dati richiesti per definire un'escursione.</li><li>3. l'attore agendo sugli elementi presenti sul video inserisce i dati relativi all'escursione (data, barca/cavallo, descrizione, ecc.); al termine preme il bottone "Inserisci";</li><li>4. il sistema controlla i dati immessi; se i dati inseriti sono corretti il sistema presenta la finestra di dialogo "Conferma Inserimento", con la quale chiede di confermare la scelta di aggiungere l'escursione al programma.<ol style="list-style-type: none"><li>(a) In caso affermativo il sistema aggiunge l'escursione al programma</li><li>(b) In caso contrario niente viene modificato</li></ol></li></ol> <p>indipendentemente da quale dei due passi precedenti sia stato eseguito, il sistema torna a presentare la schermata del punto 1, mostrando gli eventuali dati già inseriti a video.</p>



# ..CU1

indipendentemente da quale dei due passi precedenti sia stato eseguito, il sistema torna a presentare la schermata del punto 5.1, mostrando gli eventuali dati già inseriti a video.

Invariante:

A partire dal punto 1 il caso d'uso termina incondizionatamente in qualunque momento venga premuto il bottone Esci

Sequenza alternativa:

Se al punto 1 si verifica che i dati non sono corretti, viene presentata la finestra di dialogo "Errore di inserimento" nella quale si indica quale campo deve essere aggiornato. Quando l'operatore preme il bottone "OK" presente tale finestra il caso d'uso riprende dal punto 1 (senza apportare modifiche al contenuto dei campi già immessi).

Postcondizione:

Le eventuali escursioni inserite sono state memorizzate; sullo schermo viene rappresentato il menu principale

# A corredo di CU1

La vista ha  
un nome

Prototipo di  
schermata  
video (vista)  
per inserire le  
escursioni

Inserimento Escursioni

Per prima cosa  
selezionare una  
data

Una qualche forma di  
calendario che consenta di  
selezionare una data

Data	Escurs	Descrizione	Costo	Posti	Pranzo	Merenda	Visita
2/9/2008	Gita barca	Ischia	50	15	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

Gita in Barca

☒ Pranzo ☒ Merenda ☒ Visita

Costo  MaxIscrivibili

# CU2: Rimozione escursioni

Rimozione Escursioni

ALTRA ROBA (Calendario?)

Data	Escurs	Descrizione	Costo	Posti	Pranzo	Meren	Visita
2/9/2008	Gita barca	Ischia	50	15	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

Rimuovi Esci

Conferma Rimozione

Conferma rimozione escursione?

Sì No

# CU2

CU2: Rimozione escursione
<p>Sequenza eventi:</p> <ol style="list-style-type: none"><li>1. il caso d'uso inizia quando l'attore clicca sul bottone "Rimozione" sulla finestra del "Menu Principale"</li><li>2. il sistema presenta la schermata "Rimozione escursioni", nella quale viene mostrata, la lista delle escursioni in programma.</li><li>3. l'attore seleziona l'escursione da eliminare e preme il bottone "Rimuovi";</li><li>4. il sistema presenta la finestra di dialogo "Conferma rimozione" con la quale viene chiesto di confermare la scelta di eliminare l'escursione selezionata.<ol style="list-style-type: none"><li>(a) In caso affermativo l'escursione viene eliminata</li><li>(b) In caso contrario niente viene modificato</li></ol></li></ol> <p>indipendentemente da quale dei due passi precedenti sia stato eseguito, il sistema torna a presentare la schermata del punto ??.</p>
<p>Invariante:</p> <p>A partire dal punto 1 il caso d'uso termina incondizionatamente in qualunque momento venga premuto il bottone Esci</p>
<p>Postcondizione:</p> <p>Le escursioni per cui è stata confermata la rimozione sono state eliminate; sullo schermo viene ripresentato il menu principale</p>

# CU3: Modifica Escursione

- Il caso d'uso inizia quando l'attore clicca il bottone **Modifica** del menu principale.
- Il sistema presenta una schermata contenente la lista delle escursioni e i campi per inserire i dati che definiscono le escursioni.
- L'attore seleziona una escursione dalla lista e modifica i campi; al termine clicca sul bottone **SalvaMod**. Il sistema chiede conferma e in caso affermativo aggiorna i dati relativi all'escursione selezionata.
- Il ciclo può essere iterato.
- Il caso d'uso ha fine quando viene cliccato il bottone **Esci**

# CU3: Modifica Escursioni

Modifica Escursioni

Selezionare una data

Calendario che permetta di selezionare una data

Data	Escurs	Descrizione	Costo	Posti	Pranzo	Merenda	Visita
2/9/2008	Gita barca	Ischia	50	15	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

1

Gita in Barca

Descrizione: Ischia

☒ Pranzo ☐ Merenda ☒ Visita

Costo 50

MaxIscrivibili 10

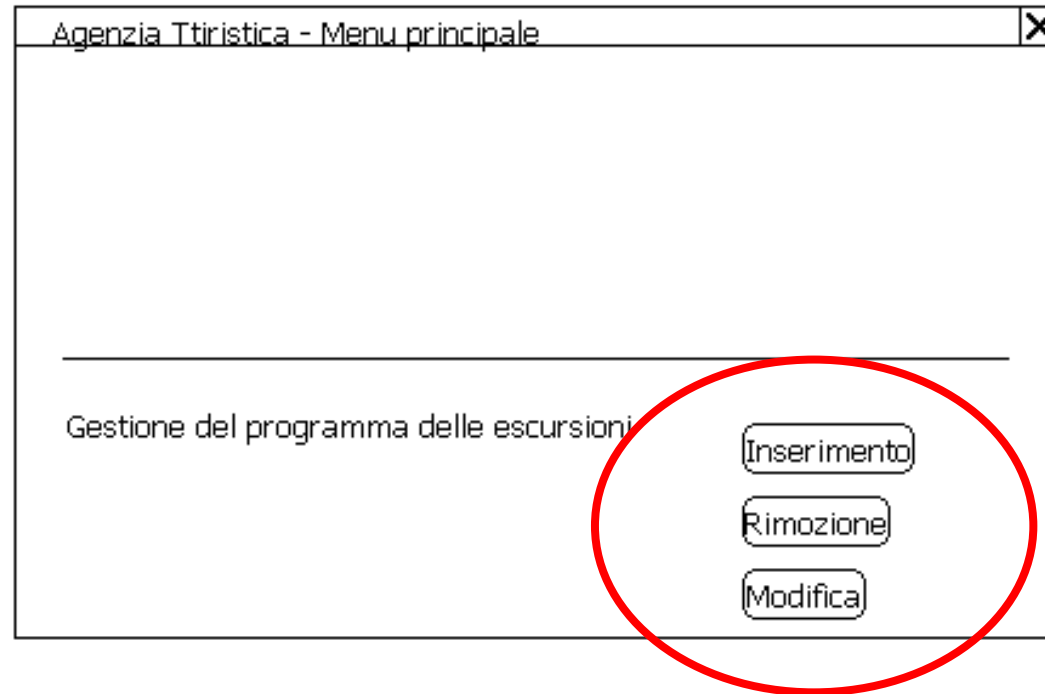
4 SalvaMod

Esci

3

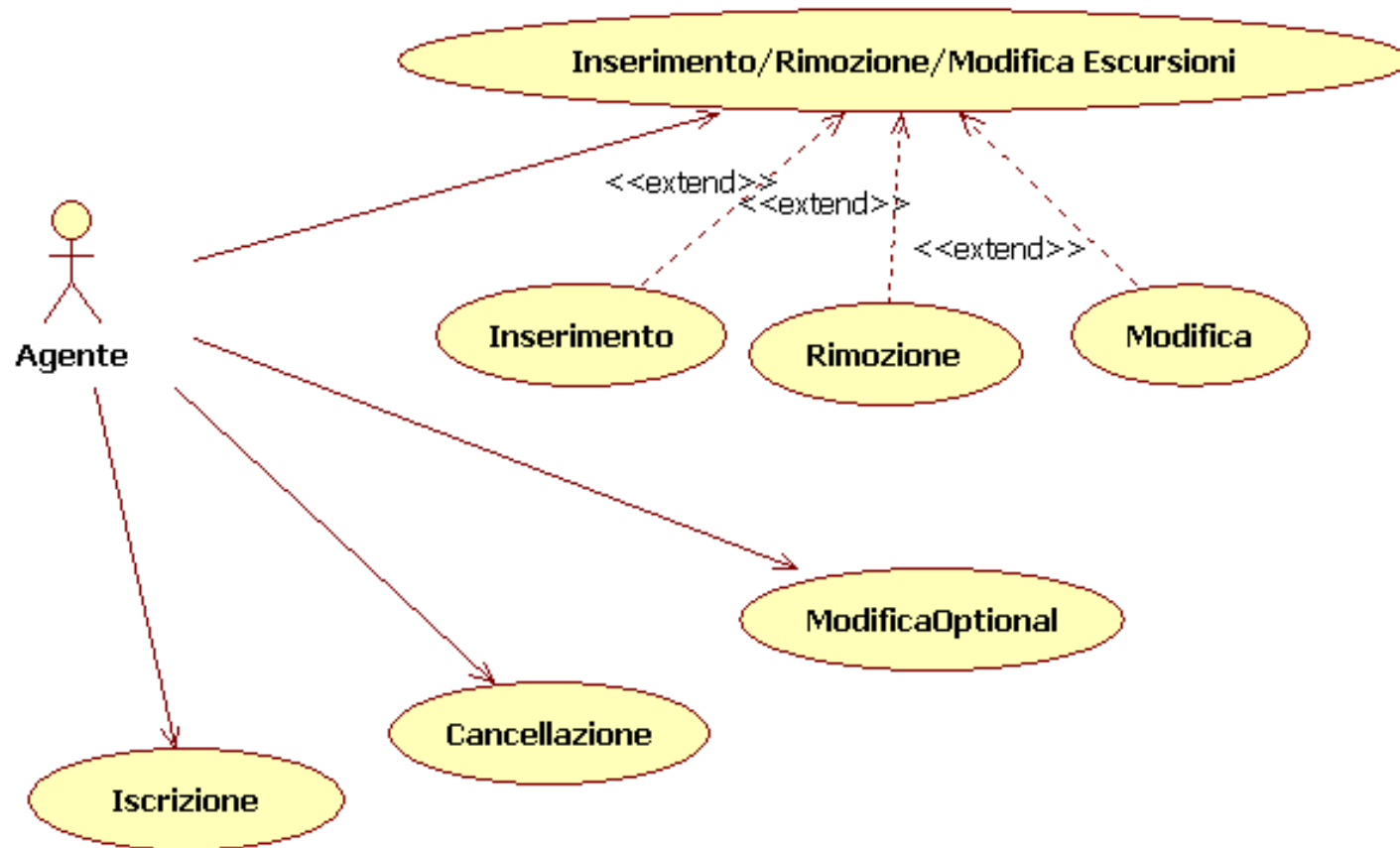
2

# Menu principale (temporaneo)



Portano sostanzialmente alla medesima schermata  
>>>> riunificare i tre casi d'uso in uno e  
prevedere tre sottocasi (estensioni)

# Rivisitazione





# ...Rivisitazione

Agenzia Turistica - Menu principale

Gestione del programma delle escursioni

Inserimenti

Inserimento/Rimozione/Modifica Escursioni

Settembre 2008

Selezionare una data

Lu	Ma	Me	Gio	Ve	Sa	Dc
1	2	3	4			
8				ecc		

Data	Escurs	Descrizione	Costo	Posti	Pranzo	Merenda	Visita
2/12/2008	Gita barca	Ischia	50	15	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

Gita in Barca

Descrizione: Ischia

☐ Pranzo ☐ Merenda ☐ Visita

Costo  MaxIscritti

Inserisci

Rimuovi

SalvaMod

Esci

Non attivi all'entrata

# ...Rivisitazione

Agency Turistica - Menu principale

Gestione del programma delle escursioni

Inserimento

**L'inserimento richiede che per prima cosa venga selezionata una data**

Inserimento/Rimozione/Modifica Escursioni

Settembre 2008

Selezionare una data

Lu	Ma	Me	Gio	Ve	Sa	Dc
1	2	3	4			
8				ecc		

Data	Escurs	Descr
2/12/2008	Gita barca	Ischia

Gita in Barca

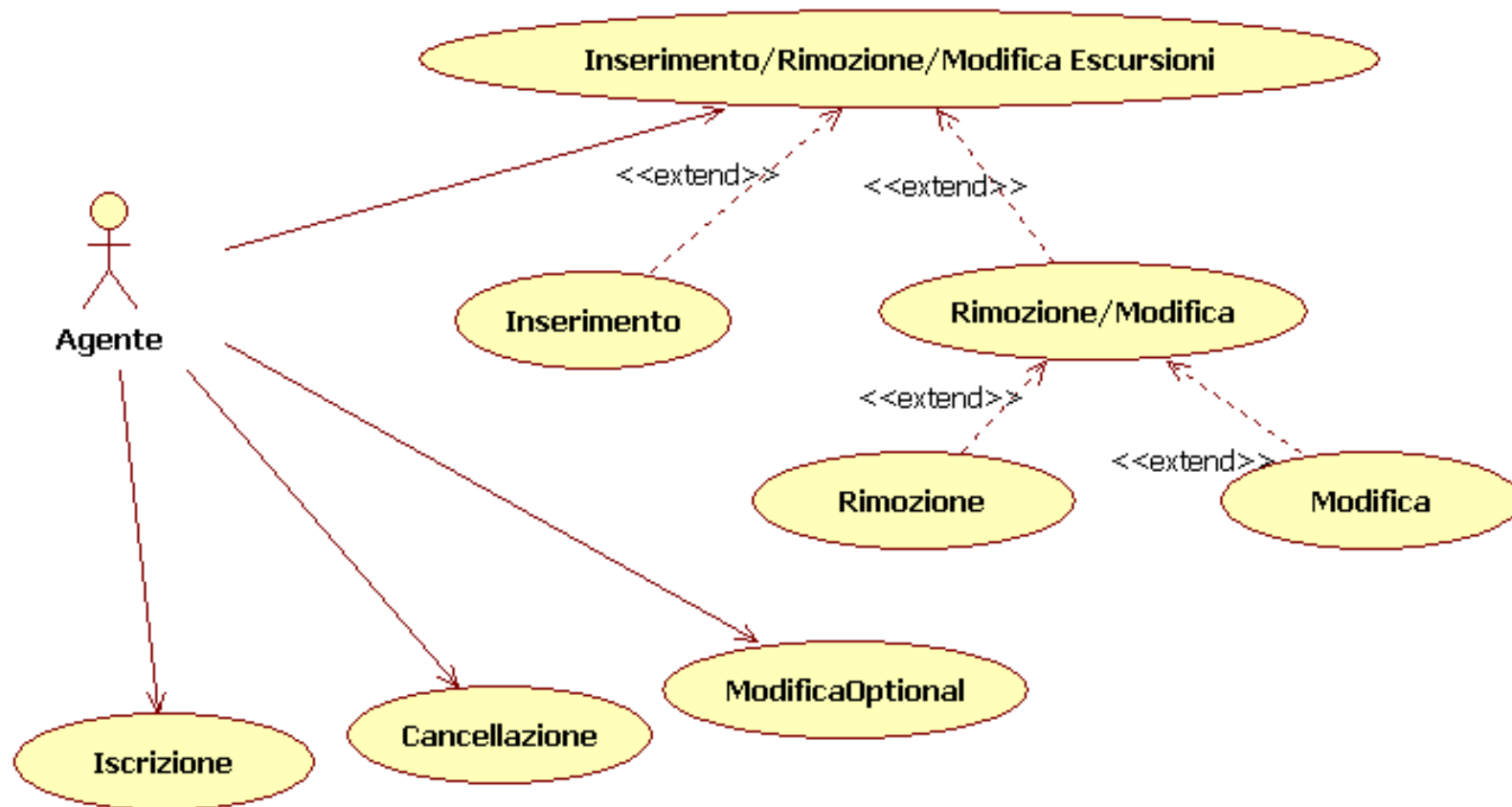
☐ Pranzo ☐ M

Costo  MaxIscritti

**Rimozione e modifica richiedono che per prima cosa venga selezionata un'escursione**

**Suggeriscono un'ulteriore rivisitazione**

# ..ulteriore rivisitazione



# Sarà meglio pensarci bene

- Potrebbe essere che il framework grafico ci consente di passare da una situazione all'altra senza imporre una scelta iniziale. Per esempio: si sceglie una data, si inizia a introdurre i dati per una escursione, poi si decide di non inserirla e si va a cercare una escursione già a programma (per modificarla/eliminarla)

L'analisi dei casi d'uso non può arrivare fino a questo punto di dettaglio: c'è il rischio di sprecare tempo.

Si tratta di dettagli da precisare in fase di implementazione, in base al tool per la costruzione della GUI

Per esempio: le Swing permettono di fare una finestra di dialogo di conferma e riconoscere la risposta un un solo statement.

The image shows a GUI window with a pink text box overlay. The text box contains three paragraphs of text. Below the text box, the GUI elements are visible: three checkboxes labeled 'Pranzo', 'Merenda', and 'Visita'; two input fields labeled 'Costo' and 'MaxIscritti'; and four buttons labeled 'Inserisci', 'Rimuovi', 'SalvaMod', and 'Esci'.

# Sarà meglio pensarci bene

- Potrebbe essere che il framework grafico ci consente di passare da una situazione all'altra senza imporre una scelta iniziale. Per esempio: si sceglie una data, si inizia a introdurre i dati per una escursione, poi si decide di non inserirla e si va a cercare una escursione già a programma (per modificarla/eliminarla)

**L'analisi deve essere rigorosa (evitare ambiguità) ma non inutilmente dettagliata. Alcuni aspetti possono essere fissati decentemente solo in fase di implementazione**

Inserisci

Data	Escurs	Descrizione	Costo	Posti	Pranzo	Merenda	Visita
2/12/2008	Gita barca	Ischia	50	15	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

Gita in Barca

Descrizione:

☐ Pranzo    ☐ Merenda    ☐ Visita

Costo     MaxIscritti

# CU: Iscrizione

Registrazione di un cliente alle escursioni X

Data	Escurs	Descrizione	Costo	Posti	Pranzo	Merenda	Visita
2/9/2008	Gita barca	Ischia	50	15	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>

☐ Pranzo    ☐ Merenda    ☐ Gita    Costo

Dati anagrafici del cliente

Codice fiscale

Nome     Cognome

# CU: Cancellaz/Mod Optional

## *Percorso principale*

- Il caso uso inizia quando viene premuto il pulsante **Cancellazione/ModificaOptional** sul menu principale.
- Il sistema presenta la schermata **Cancellazione/ModificaOptional**.
- L'attore deve anzitutto inserire il codice fiscale del partecipante e premere il pulsante **Cerca**. Se il codice immesso corrisponde effettivamente a quello di un partecipante, il sistema risponde presentando i dati del partecipante e la lista delle escursioni a cui è iscritto.
- L'attore seleziona una escursione tra quelle in lista; se ci sono optional scelti le relative checkbox della parte del display in cui si inseriscono gli optional riportano il segno di spunta. L'attore può modificare la spunta degli optional.
- Se l'attore preme il pulsante **SalvaModifica** si determina il cambiamento degli optional secondo la nuova spunta.
- Se preme il pulsante **Cancella** il partecipante viene cancellato dall'escursione.
- Il ciclo può essere iterato e ha fine al click di **Esci**. Dopo l'uscita viene ripresentato il menu principale

# .. CU: Cancellaz/mod optional

## *Percorso alternativo*

- Se non si trova il partecipante non viene presentata alcuna lista delle escursioni e non vengono attivati i pulsanti **Cancella** e **SalvaModifica**, per cui non resta che uscire.

Comporta un nuovo caso d'uso e un bottone sul menu principale

Cancellazione/modificaOptional

Codice fiscale

Nome  Cognome

Iscritto alle Escursioni

Data	Escurs	Descrizione	Costo	Posti	Pranzo	Merenda	Visita	Costo

☐ Pranzo ☐ Merenda ☐ Visita

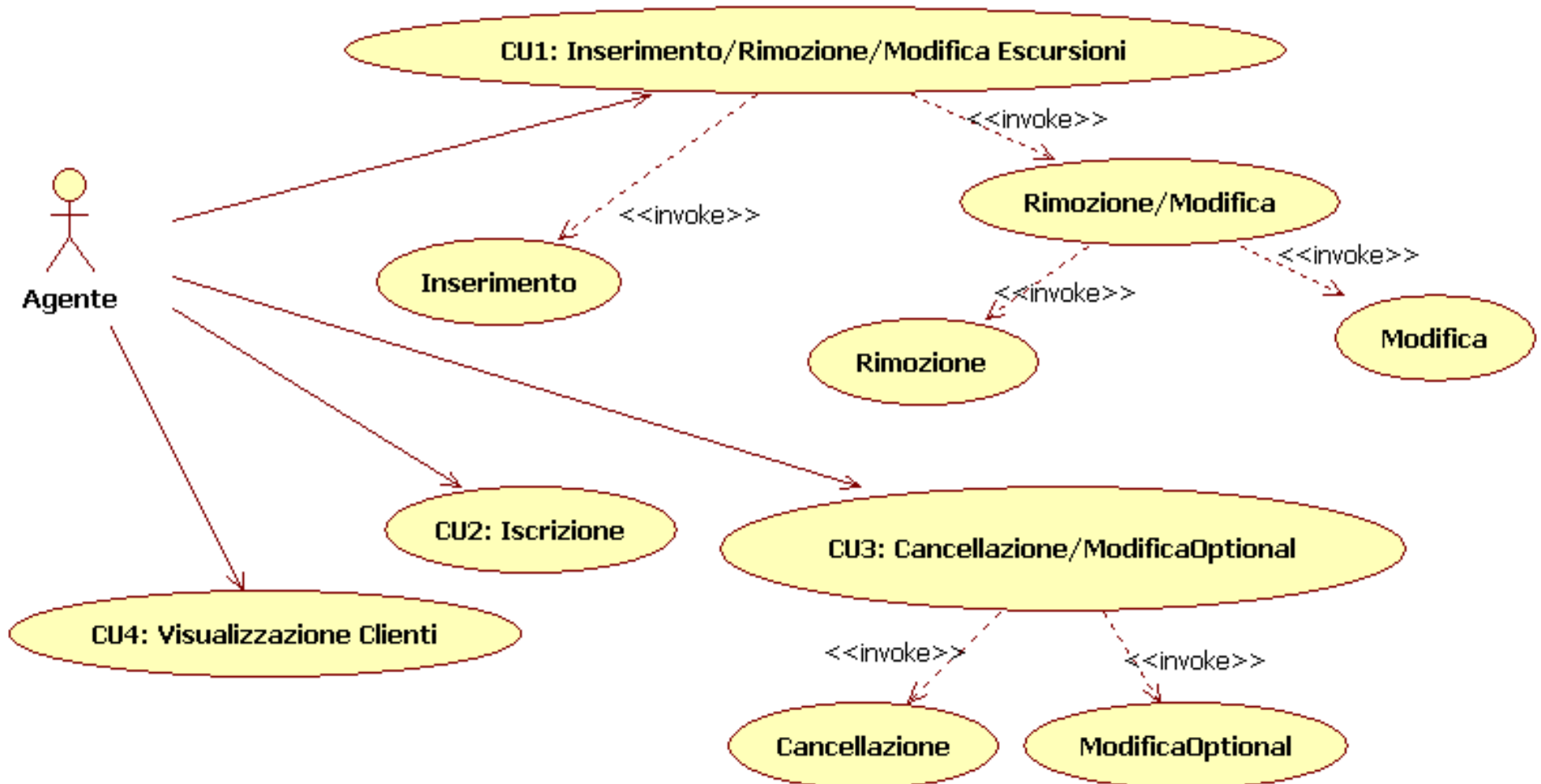
Non attivi all'entrata



# Nota

- E' stato fatto volutamente complicato:
  - Nella vista precedente si poteva partire dal nome, cercare il cliente e, se presente, mostrare il suo codice fiscale.
  - La scelta di andare per codice fiscale giustifica il caso d'uso "visualizzazione clienti"

# Casi d'uso finale



# Menu principale (finale)

Si rende necessario per vedere se un cliente è nel sistema ottenendo il suo CF ai fini della cancellazione o modifica degli optional

Agenzia Turistica - Menu principale

Gestione delle iscrizioni

Registrazione

Elenco clienti

Cancellazione Iscritto/Modifica Optional

---

Gestione del programma delle escursioni

Inserimento/Rimozione/Modifica Escursioni

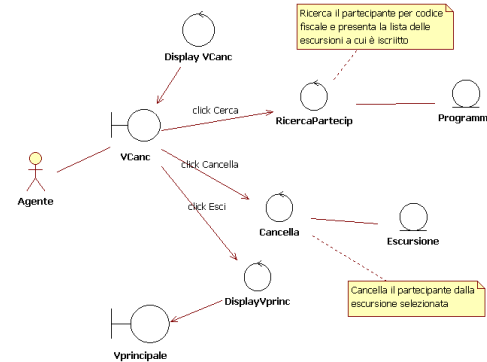
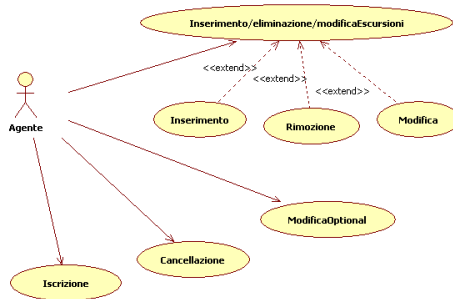
# Cosa abbiamo fatto?

- Abbiamo fatto il manuale utente!
  - Basta riordinare il contenuto dei casi d'uso, riscrivendoli in modo che appaiano come un manuale
- La prospettiva giusta nel fare i casi d'uso:

**Analisi casi d'uso ⇔ Stesura manuale utente**

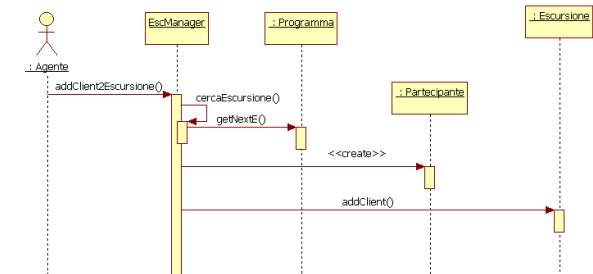
# Riempire il fossato (tra cosa e come)

COSA



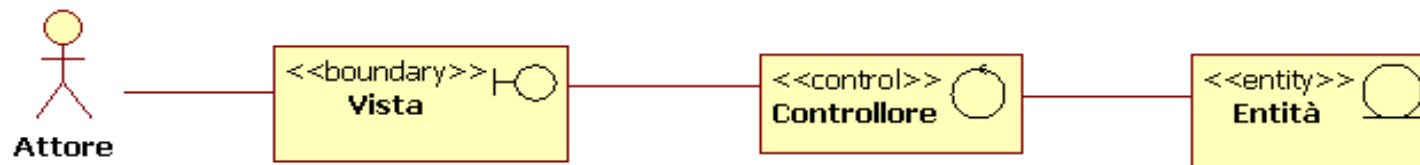
COME

Robustness  
Analysis



# Analisi (di robustezza)

- Il modello di dominio contiene i dati
- L'analisi dei casi d'uso ha messo in luce cosa succede.
  - A partire dai casi d'uso dobbiamo individuare le funzioni applicative
  - Per il momento confonderemo controllori e funzioni



# Prima però...

- Studiamo meglio la questione non trattabile in dettaglio con i casi d'uso:
  - Esaminiamo con i diagrammi delle macchine a stati il comportamento dell'interfaccia
  - Ci fornirà indicazioni per l'analisi
- Di solito le macchine a stati si usano in fase di implementazione
  - Ma in questo caso è bene approfondire aspetti non completamente chiari

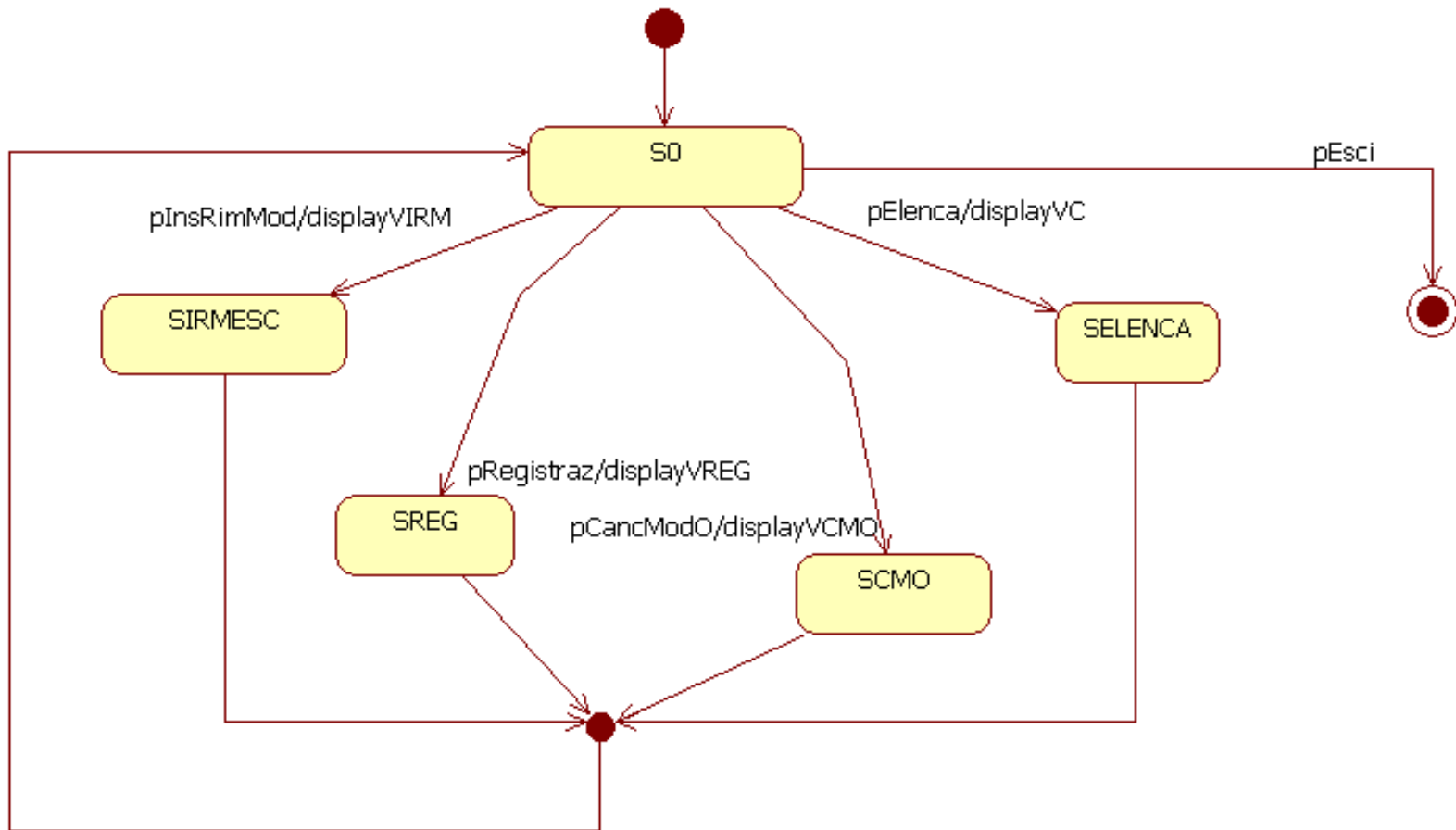
# Intermezzo:

# Le macchine a stati

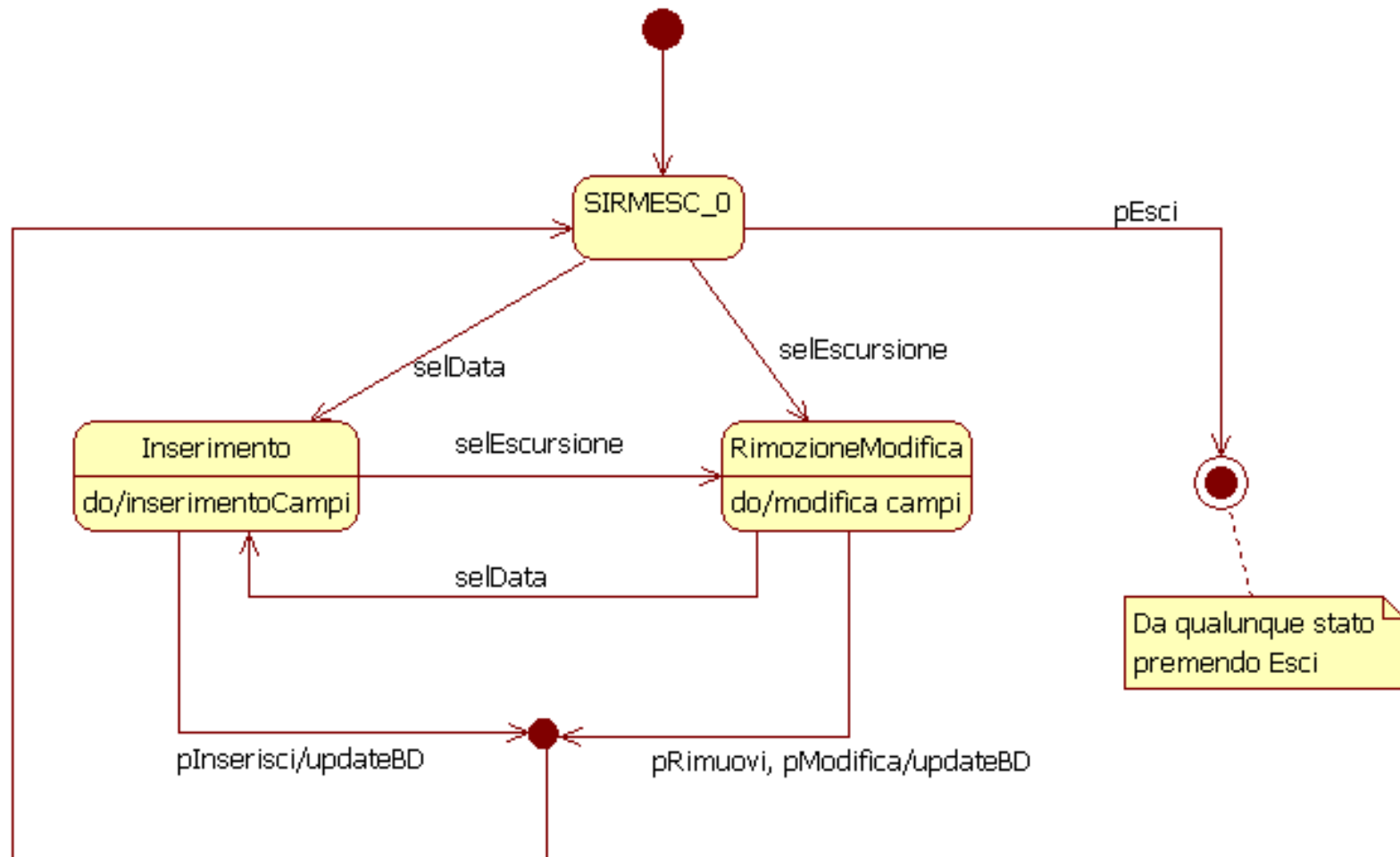
(vedi slide)



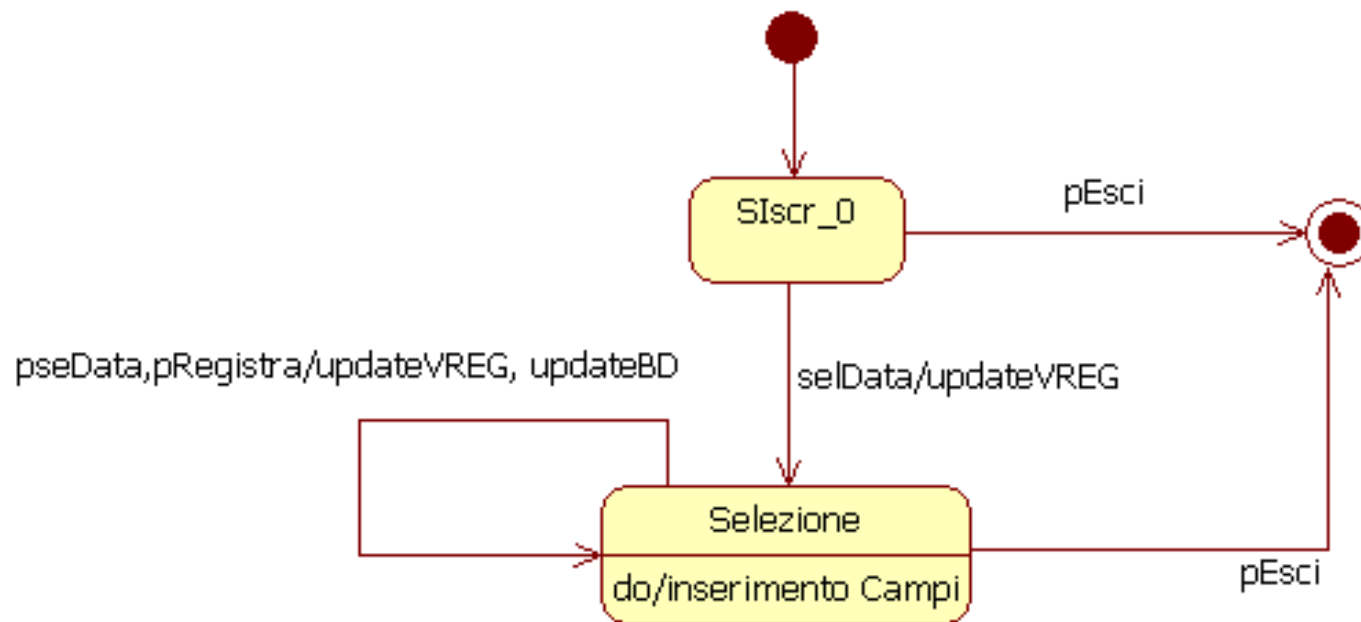
# Stato Aggregato Interfaccia



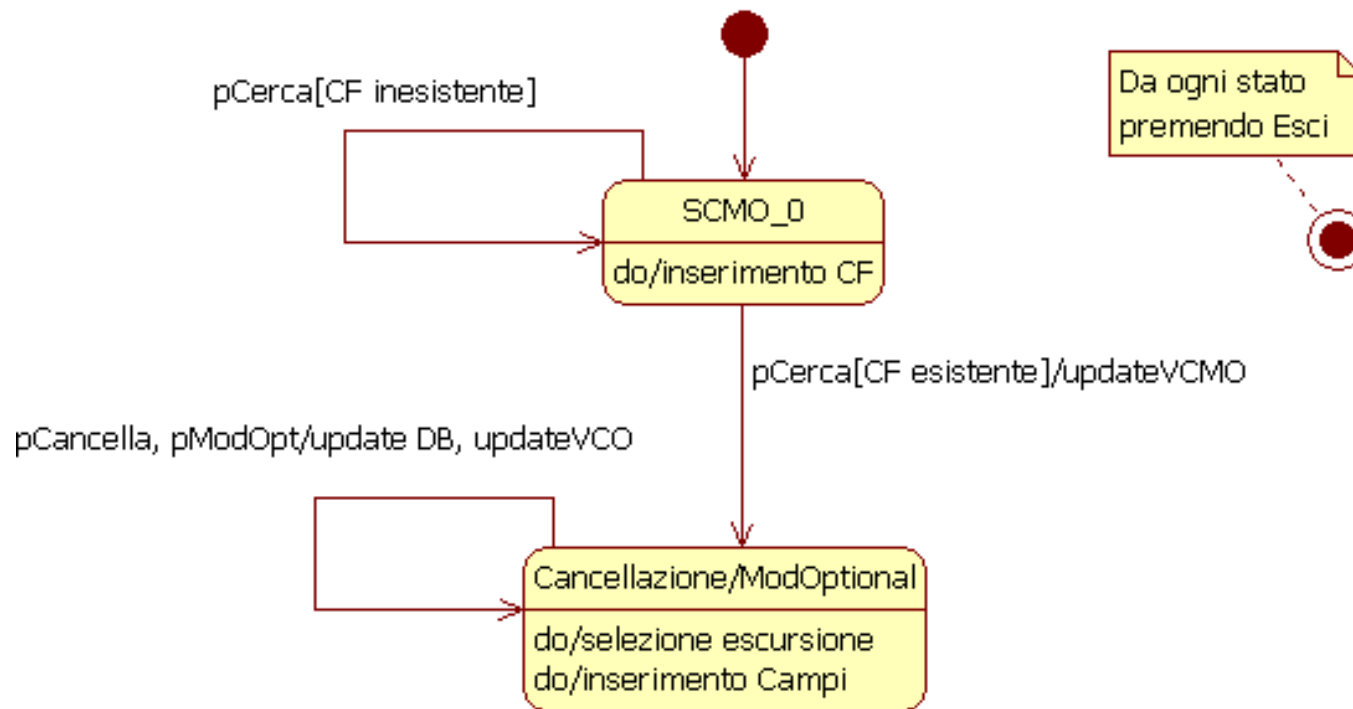
# Inserimento/Rimozione/Modifica



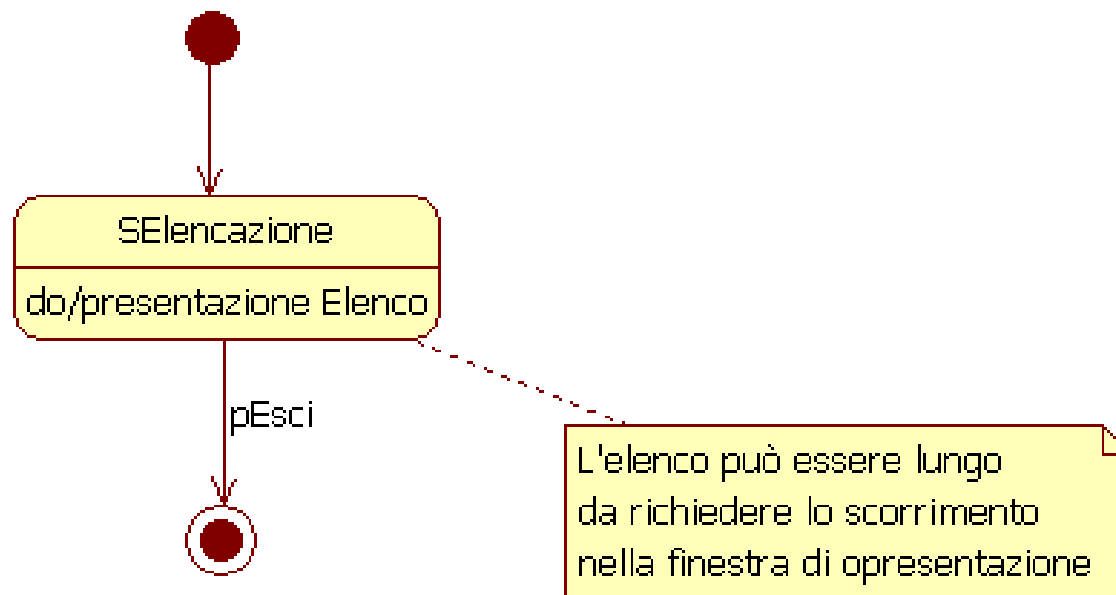
# Iscrizione



# Cancellazione/ModificaOptional



# Elencazione

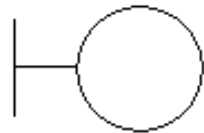


# Analisi di robustezza

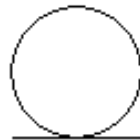
## Premessa

- Dobbiamo trovare le funzioni
  - Le rappresentiamo con lo stereotipo del controller
  - Ma resta inteso che non sono dei controller effettivi
  - In una fase successiva le funzioni verranno allocate a classi del modello e a specifici controller

# Regole (un po' rigide)



Boundary



Entity



Control

Sostantivi

Verbi

- I sostantivi possono comunicare con i verbi e viceversa
- I sostantivi non possono comunicare tra loro
- I verbi possono comunicare con i sostantivi (e viceversa) e tra di loro

Sono le regole di ICONIX

# Suggerimento

Cominciare riportando il caso d'uso in forma testuale su un lato del foglio

Sull'altro lato sviluppare il diagramma

L'attore clicca sul bottone  
Inserisci/Elimina/Modifica  
sulla finestra de Menu Principale.  
Il sistema presenta la schermata  
Inserimento/Rimozione/Modifica escursioni



# RobCU1:

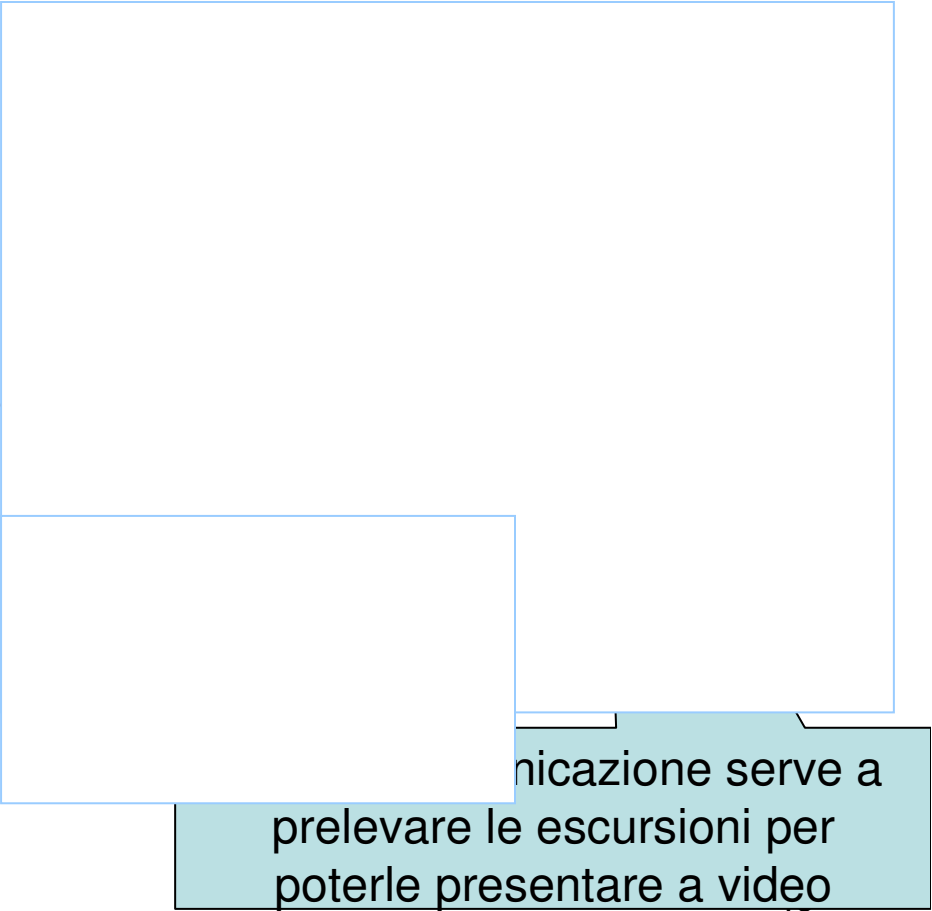
Cominciare riportando il caso d'uso in forma testuale su un lato del foglio

L'attore clicca sul bottone Inserisci/Elimina/Modifica sulla finestra de Menu Principale. Il sistema presenta la schermata Inserimento/Rimozione/Modifica escursioni

Sull'altro lato sviluppare il diagramma



Agente



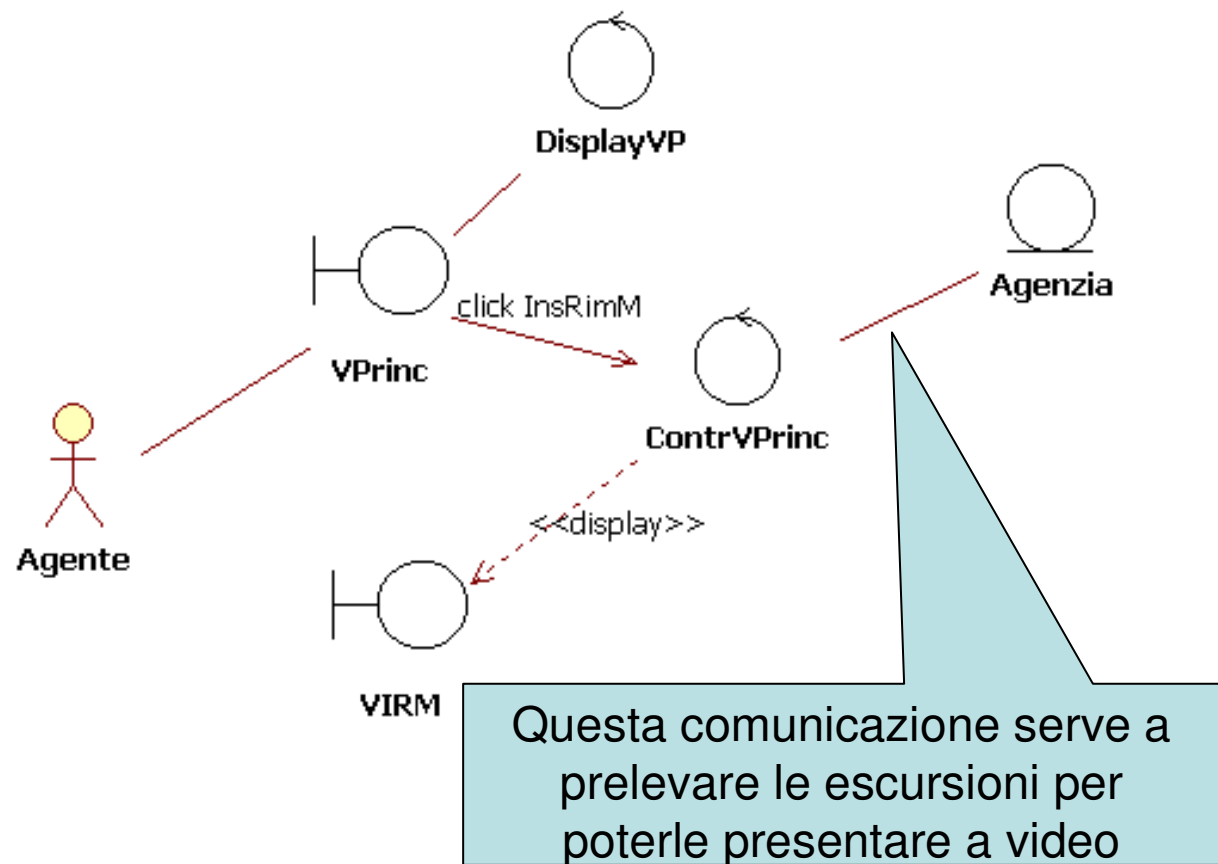
La comunicazione serve a prelevare le escursioni per poterle presentare a video

# RobCU1:

Cominciare riportando il caso d'uso in forma testuale su un lato del foglio

L'attore clicca sul bottone Inserisci/Elimina/Modifica sulla finestra de Menu Principale. Il sistema presenta la schermata Inserimento/Rimozione/Modifica escursioni

Sull'altro lato sviluppare il diagramma



# RobCU1 Ins

L'estensione viene presa quando l'attore, ~~come prima cosa,~~  
~~seleziona una data, il sistema attiva~~  
~~il pulsante Inserisci.~~  
~~(Successivamente l'attore può immettere gli altri~~  
~~dati relativi all'escursione da inserire.)~~

Quando viene cliccato il pulsante Inserisci, se i dati sono corretti, viene presentata la finestra Conferma registrazione, alla quale l'attore risponde cliccando sul pulsante Sì o No. Se viene cliccato il pulsante Sì l'escursione viene registrata e la sua descrizione inserita nella lista a video; se viene cliccato il pulsante No niente accade; in ambedue i casi viene ripresentata la schermata Inserimento/Rimozione/Modifica escursioni (con i tre pulsanti Inserisci, Rimuovi e SalvaMod disabilitati) sulla quale è possibile effettuare un nuovo inserimento, ovvero cancellare, ovvero modificare un'escursione.

Percorso alternativo

Meglio non prestare attenzione. Rinviare alla fase di implementazione.

Peraltro abbiamo rivisto questo comportamento con il diagramma di stato Inserimento/Rimozione/Modifica

# RobCU1 Ins

L'estensione viene presa quando l'attore,  
~~come prima cosa,~~  
~~seleziona una data, il sistema attiva~~  
~~il pulsante Inserisci.~~  
(~~Successivamente l'attore può immettere gli altri~~  
~~dati relativi all'escursione da inserire.~~)  
Quando viene cliccato il pulsante Inserisci, se i  
dati sono corretti, ~~viene presentata la finestra~~  
~~Conferma registrazione, alla quale l'attore~~  
~~risponde cliccando sul pulsante Sì o No.~~  
~~Se viene cliccato il pulsante Sì~~  
l'escursione viene registrata e la sua  
descrizione inserita nella lista a video;  
~~se viene cliccato il pulsante No niente accade,~~  
~~in ambedue i casi viene ripresentata la schermata~~  
Inserimento/Rimozione/Modifica escursioni  
(con i tre pulsanti Inserisci, Rimuovi e SalvaMod  
disabilitati) sulla quale è possibile effettuare un  
nuovo inserimento, ovvero cancellare,  
ovvero modificare un'escursione.

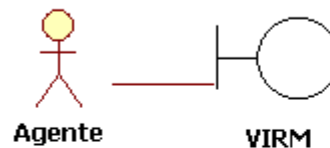
Percorso alternativo

Per le stesse ragioni

# RobCU1 Inserimento

L'estensione viene presa quando l'attore, come prima cosa, seleziona una data, il sistema attiva il pulsante Inserisci. (Successivamente l'attore può immettere gli altri dati relativi all'escursione da inserire.) Quando viene cliccato il pulsante Inserisci, se i dati sono corretti, viene presentata la finestra Conferma registrazione, alla quale l'attore risponde cliccando sul pulsante Sì o No. Se viene cliccato il pulsante Sì l'escursione viene registrata e la sua descrizione inserita nella lista a video; se viene cliccato il pulsante No niente accade, in entrambi i casi viene ripresentata la schermata Inserimento/Rimozione/Modifica escursioni (con i tre pulsanti Inserisci, Rimuovi e SalvaMod disabilitati) sulla quale è possibile effettuare un nuovo inserimento, ovvero cancellare, ovvero modificare un'escursione.

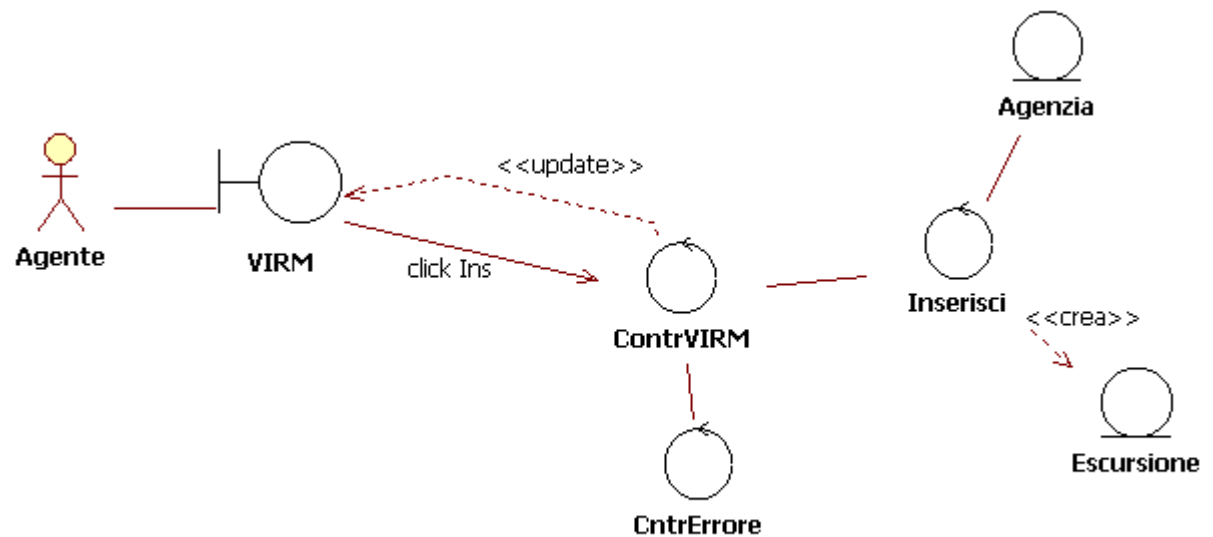
Percorso alternativo



# RobCU1 Inserimento

L'estensione viene presa quando l'attore,  
come prima cosa,  
seleziona una data, il sistema attiva  
il pulsante Inserisci.  
(Successivamente l'attore può immettere gli altri  
dati relativi all'escursione da inserire.)  
Quando viene cliccato il pulsante Inserisci, se i  
dati sono corretti, viene presentata la finestra  
Conferma registrazione, alla quale l'attore  
risponde cliccando sul pulsante Sì o No.  
Se viene cliccato il pulsante Sì  
l'escursione viene registrata e la sua  
descrizione inserita nella lista a video;  
se viene cliccato il pulsante No niente accade,  
in entrambi i casi viene ripresentata la schermata  
Inserimento/Rimozione/Modifica escursioni  
(con i tre pulsanti Inserisci, Rimuovi e SalvaMod  
disabilitati) sulla quale è possibile effettuare un  
nuovo inserimento, ovvero cancellare,  
ovvero modificare un'escursione.

Percorso alternativo

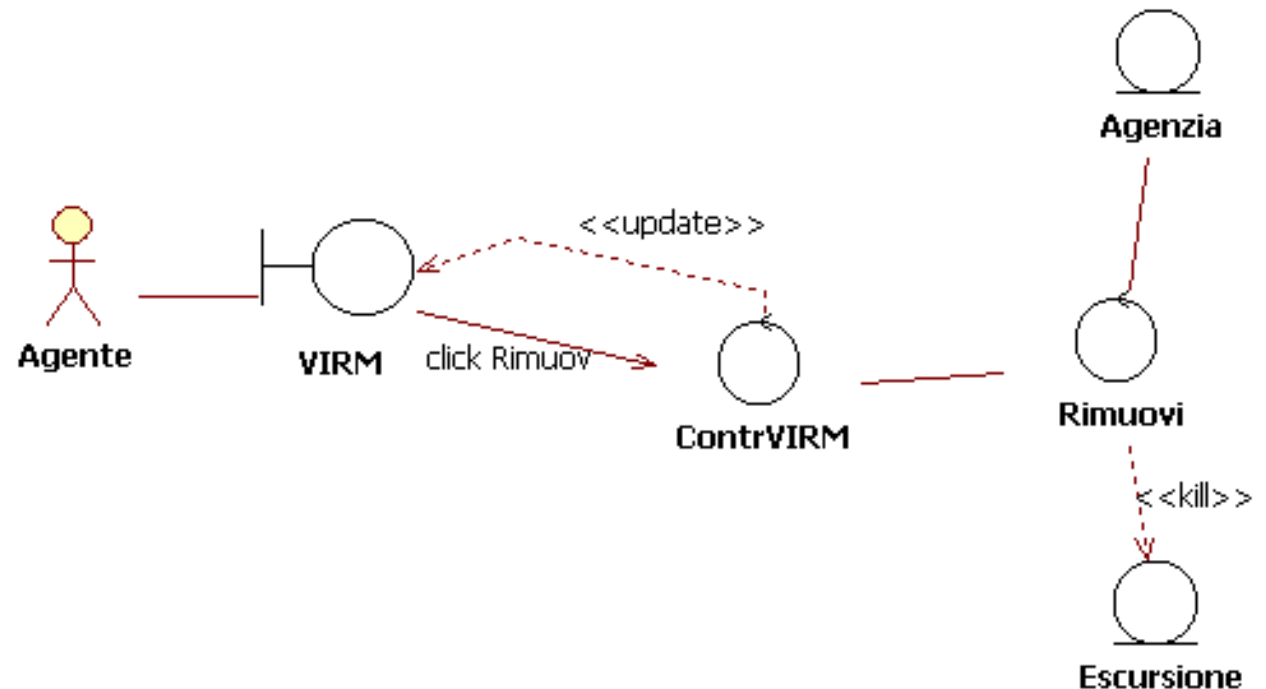


# RobCU1 Rimozione

L'estensione Rimozione/Modifica viene presa quando l'attore, come prima cosa, seleziona un'escursione dalla lista; a questo punto si attivano i pulsanti Rimuovi e SalvaMod. Se L'attore clicca il bottone Rimuovi.

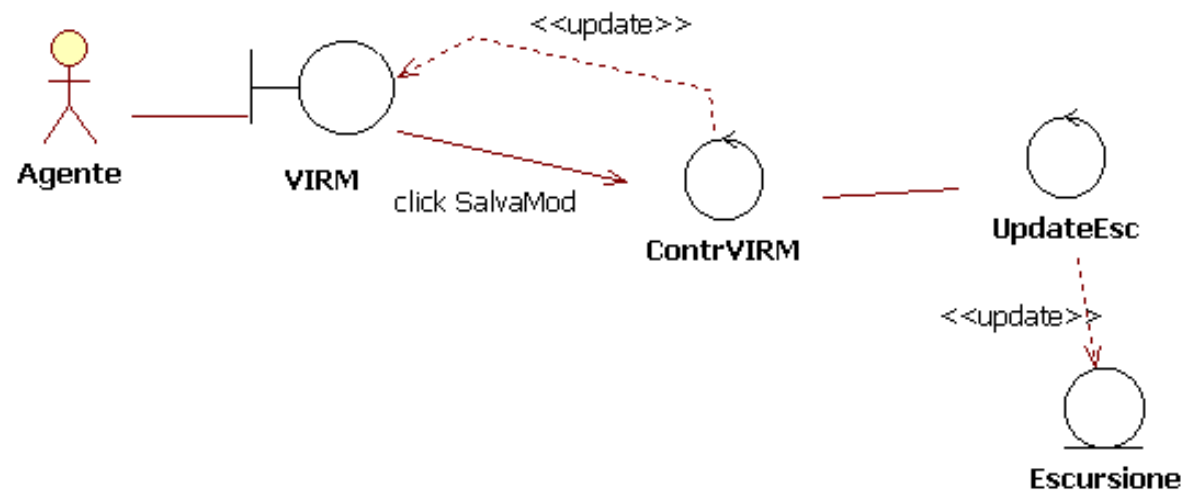
l'escursione viene rimossa, Il sistema chiede conferma. Al compimento dell'azione viene ripresentata la schermata

Inserim/Rim/Mod escursioni (con i tre pulsanti Inserisci, Rimuovi e SalvaMod disabilitati) sulla quale è possibile effettuare un nuovo inserimento, ovvero cancellare un'escursione, ovvero modificare un'escursione.



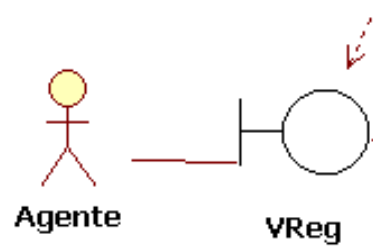
# RobCU1 Modifica

L'estensione Rimozione/Modifica viene presa quando l'attore, come prima cosa, seleziona un'escursione dalla lista; a questo punto si attivano i pulsanti Rimuovi e SalvaMod. Se L'attore clicca il bottone SalvaMod l'escursione viene modificata con i dati immessi. Il sistema chiede conferma. Al compimento dell'azione viene ripresentata la schermata Inserim/Rim/Mod escursioni (con i tre pulsanti Inserisci, Rimuovi e SalvaMod disabilitati) sulla quale è possibile effettuare un nuovo inserimento, ovvero cancellare un'escursione, ovvero modificare un'escursione.



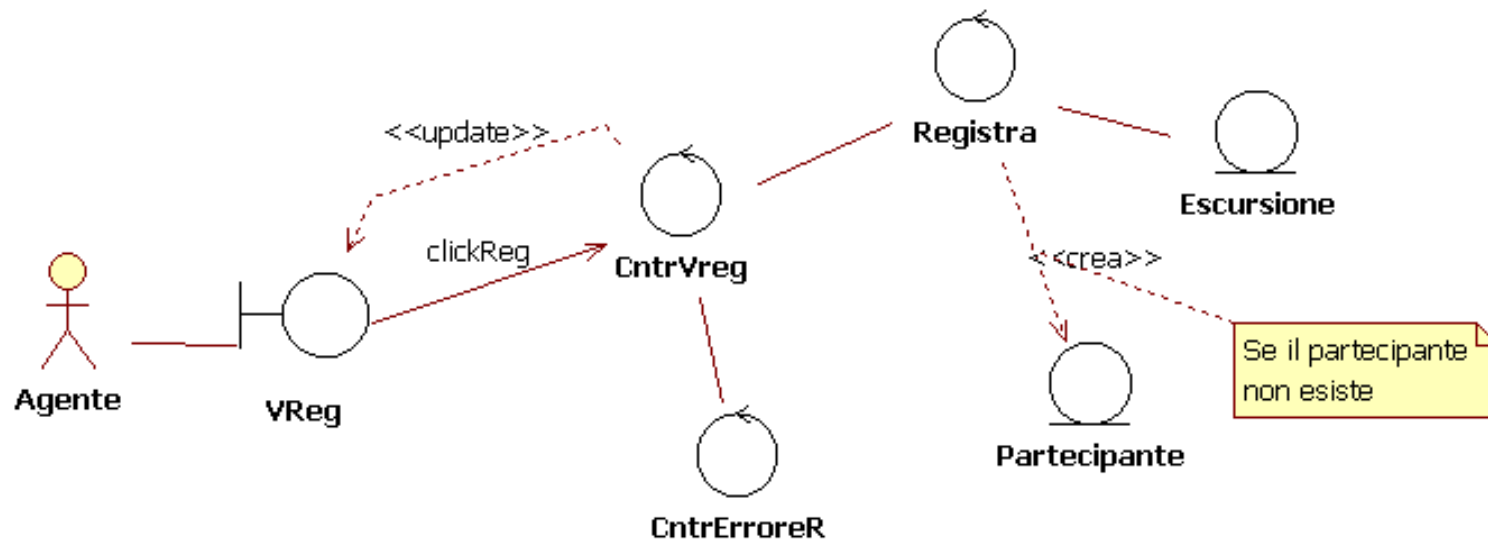


# RobCU2: Iscrizione



- I rimanenti diagrammi sono ottenibili in modo analogo

# RobCU2: Iscrizione



- I rimanenti diagrammi sono ottenibili in modo analogo

# Dove si mettono le funzioni?

- Come funziona il sistema è chiaro (anche tenendo conto di come operano le Swing):
  - Ogni schermata ha uno o più *ascoltatori di eventi* relativi agli oggetti che la compongono. Essi sono responsabili di avviare o eseguire le funzioni associate
    - Funzioni applicative da demandare a oggetti specifici
    - Funzioni ausiliare (controllo correttezza campi) eseguibili direttamente dall'ascoltatore
- Capito il concetto non vale la pena dannarsi con troppi dettagli

# ..Dove si mettono le Funzioni

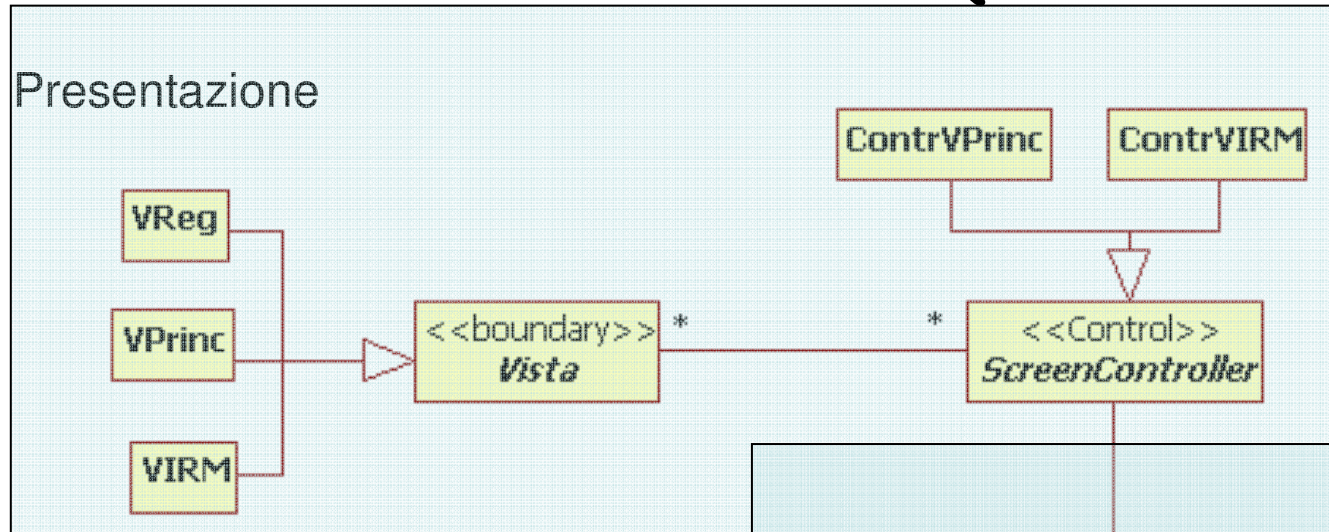
- Ci sono essenzialmente di classi di oggetti da manipolare: Escursioni e Partecipanti
  - La classe agenzia è il “contenitore”
  - I tipi di optional sono solo 3 (tre singleton)
- E' naturale prevedere due oggetti applicativi deputati rispettivamente alla gestione delle Escursioni e dei Partecipanti
  - (Essi pure da realizzare come singleton)
  - Allocare a questi due oggetti le funzionalità identificate (e quelle che verranno identificate nella fase di implementazione)

# ...Dove si mettono le Funzioni

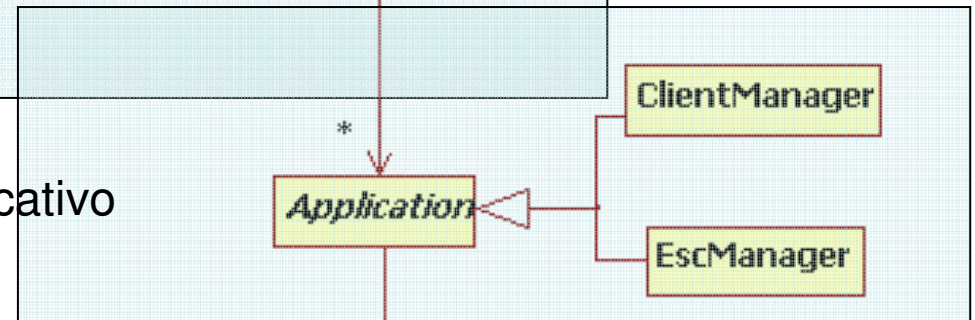
- Chiamiamo EscManager e ClientManager le classi applicative delineate
  - La funzione di modifica dei dati di una escursione sarà un metodo di ESCManager chiamato dall'ascoltatore del bottone "SalvaMod"
  - La funzione di modifica degli optional scelti da un partecipante sarà un metodo di ClientManager chiamato dall'ascoltatore del bottone "ModOpt"
- I diagrammi di sequenza servono ad allocare le funzionalità individuate, a indentificarne delle nuove, eventualmente allocandone parte alle classi che fanno parte del modello di dominio.

# Con il contorno (3 livelli)

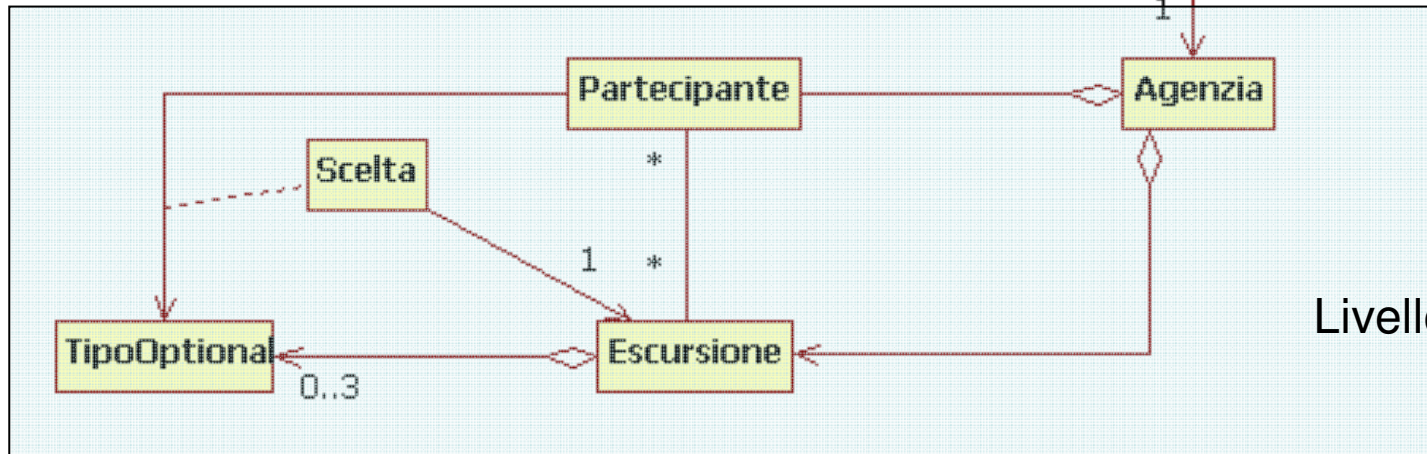
Livello Presentazione



Livello Applicativo

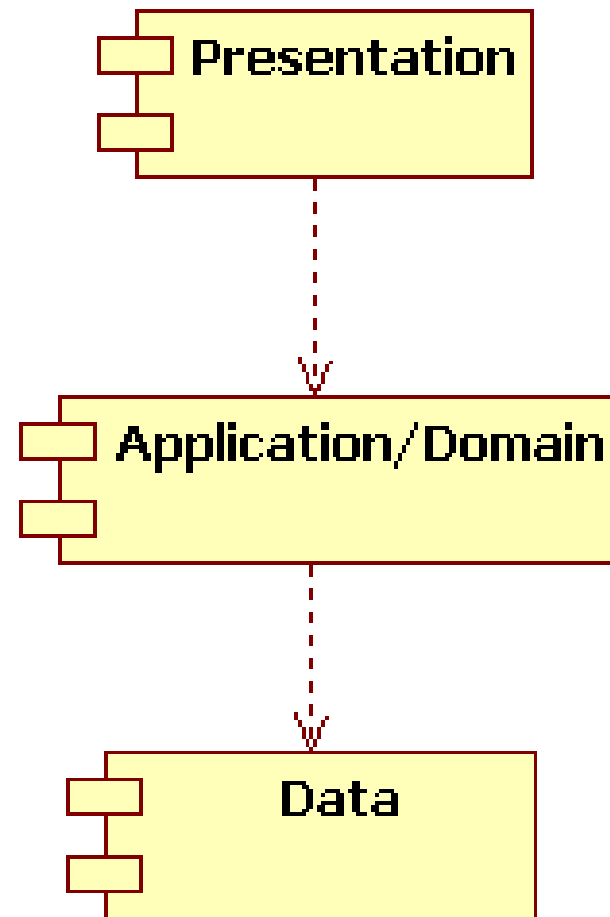


Livello Dati



# Qualunque sistema

- E' schematizzabile in questo modo
- Nei sistemi reali il livello più basso è una base di dati (relazionale)

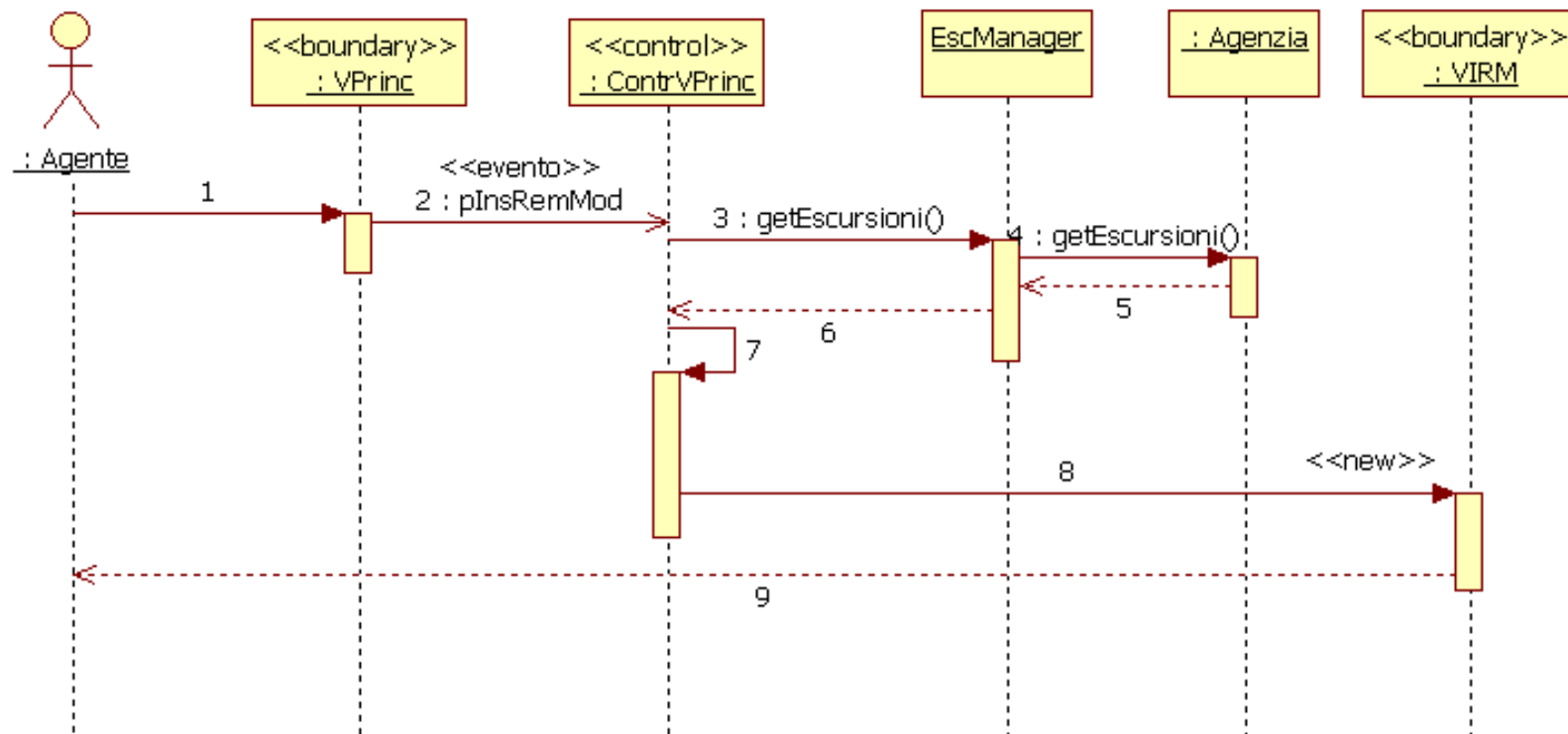


# Diagrammi di sequenza

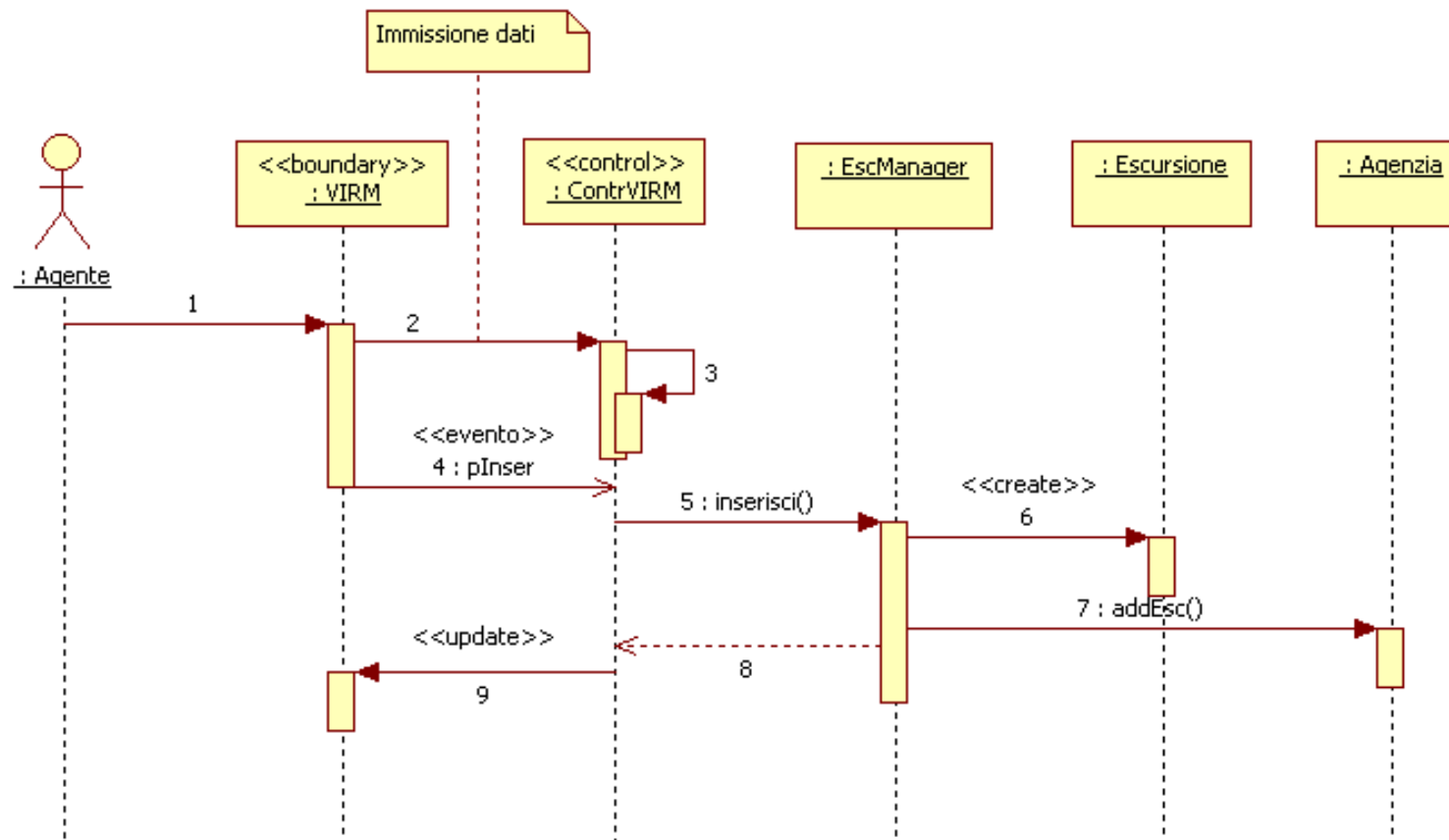
- Sono la traduzione dei casi d'uso, in termini di responsabilità delle classi
- La loro stesura è l'avvio del progetto dettagliato
  - Identificare i metodi delle classi
  - Ripercorrere i casi d'uso



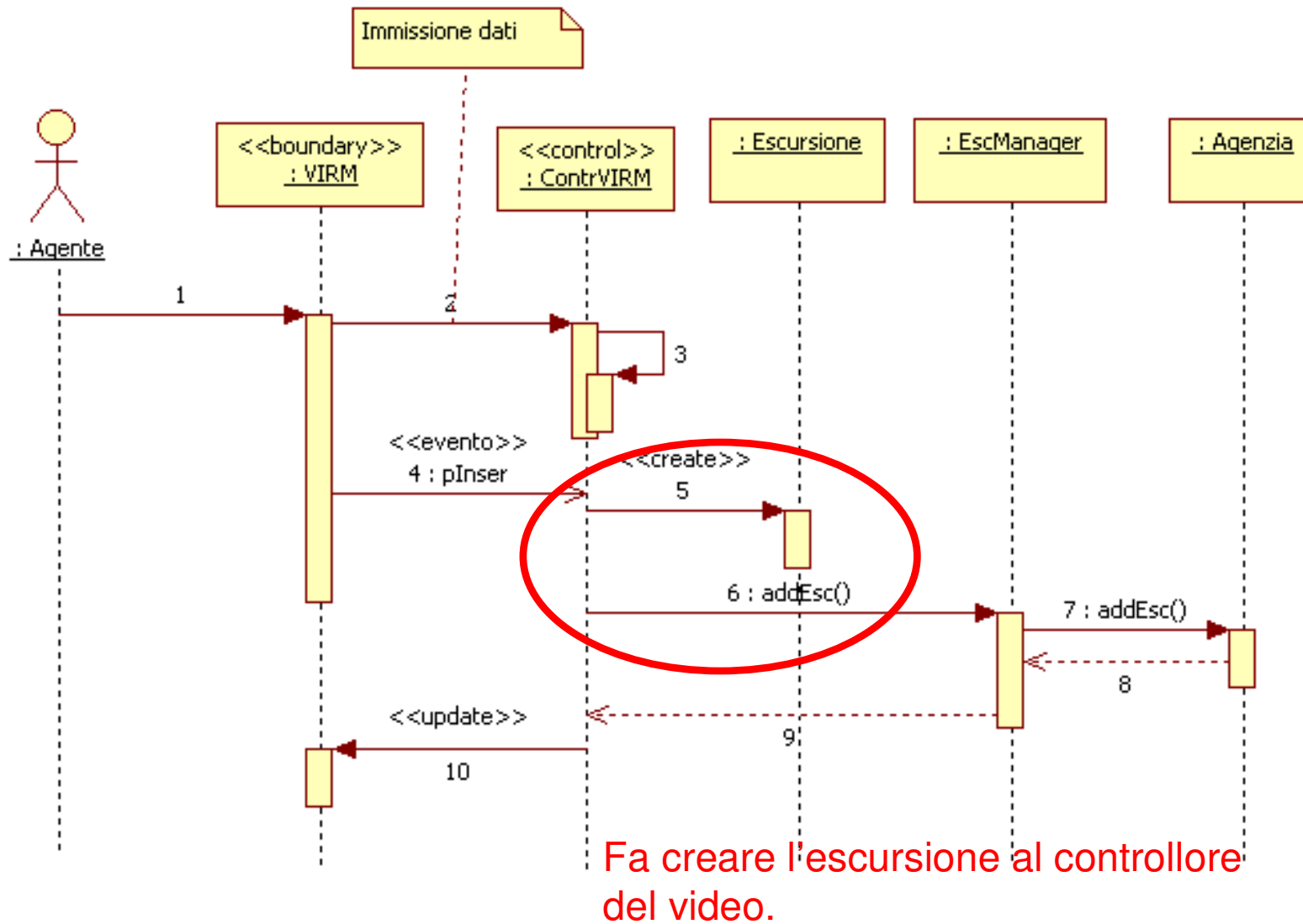
# SeqCU1 (Cfr RobCU1)



# SeqCU1Ins (Cfr RobCU1Ins)



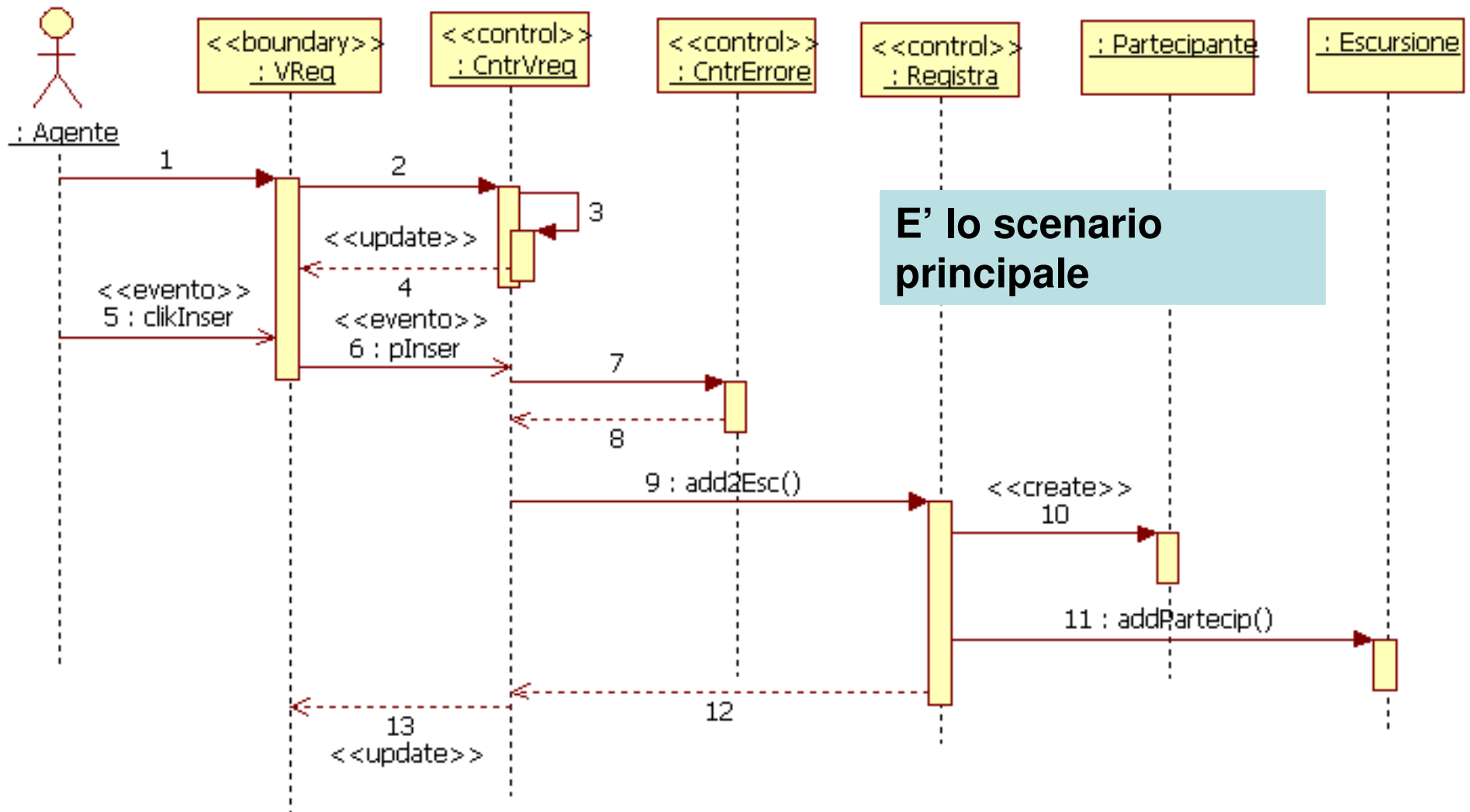
# E questa??



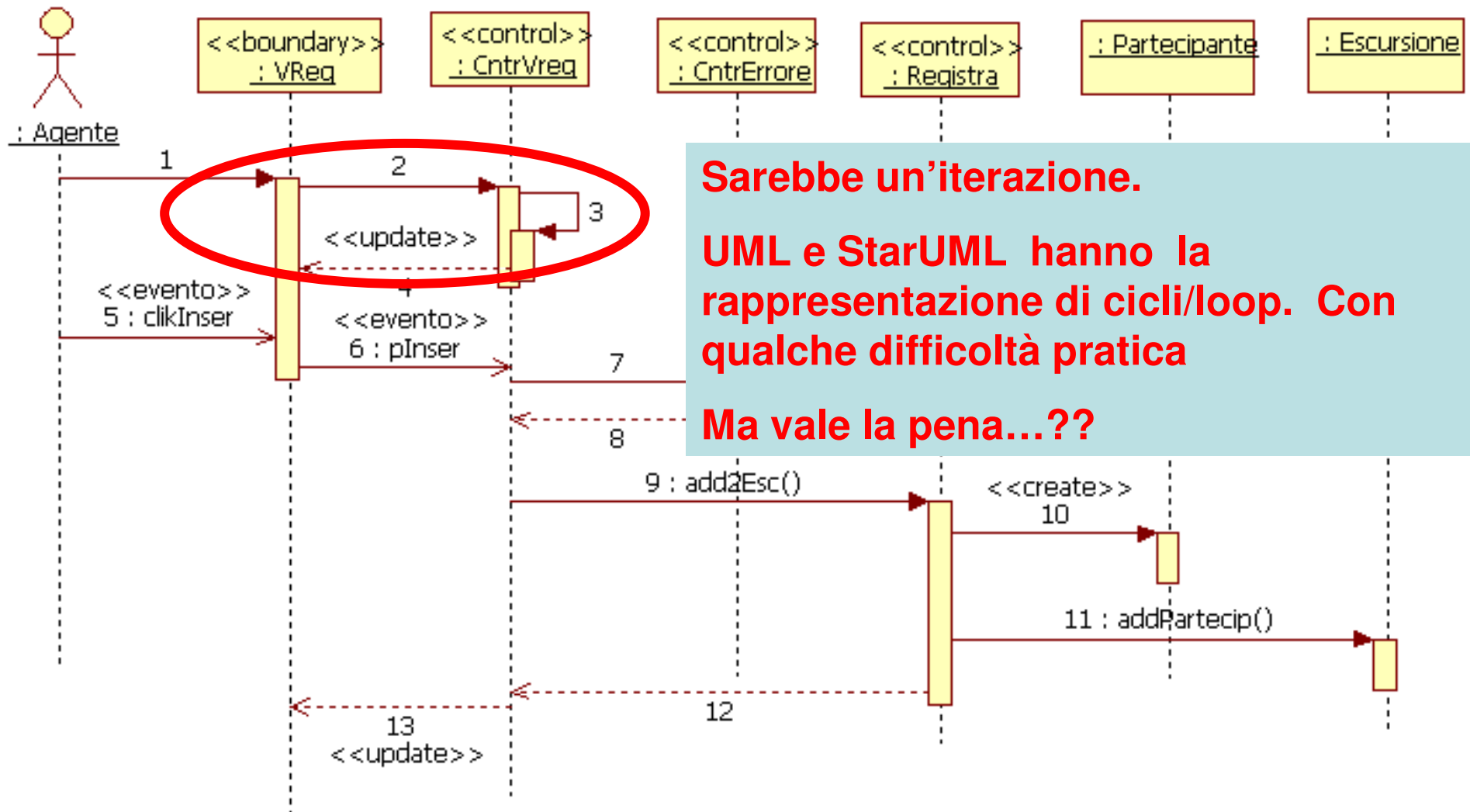
## ..segue (e questa??)

- Il confronto tra le due soluzioni precedenti suggerisce che certi dettagli vanno lasciati alla fase di implementazione
  - Dipende da come il tool dell'interfaccia supporta i componenti che vengono presentati a video
    - Quanta conoscenza ha l'interfaccia circa le entità per poterle rappresentare
- Criterio generale: favorire il disaccoppiamento tra i tre livelli. Nel caso specifico portare in EscManager la creazione dell'escursione

# SeqCU2 (Cfr RobCU2)



# SeqCU2 (Cfr RobCU2)

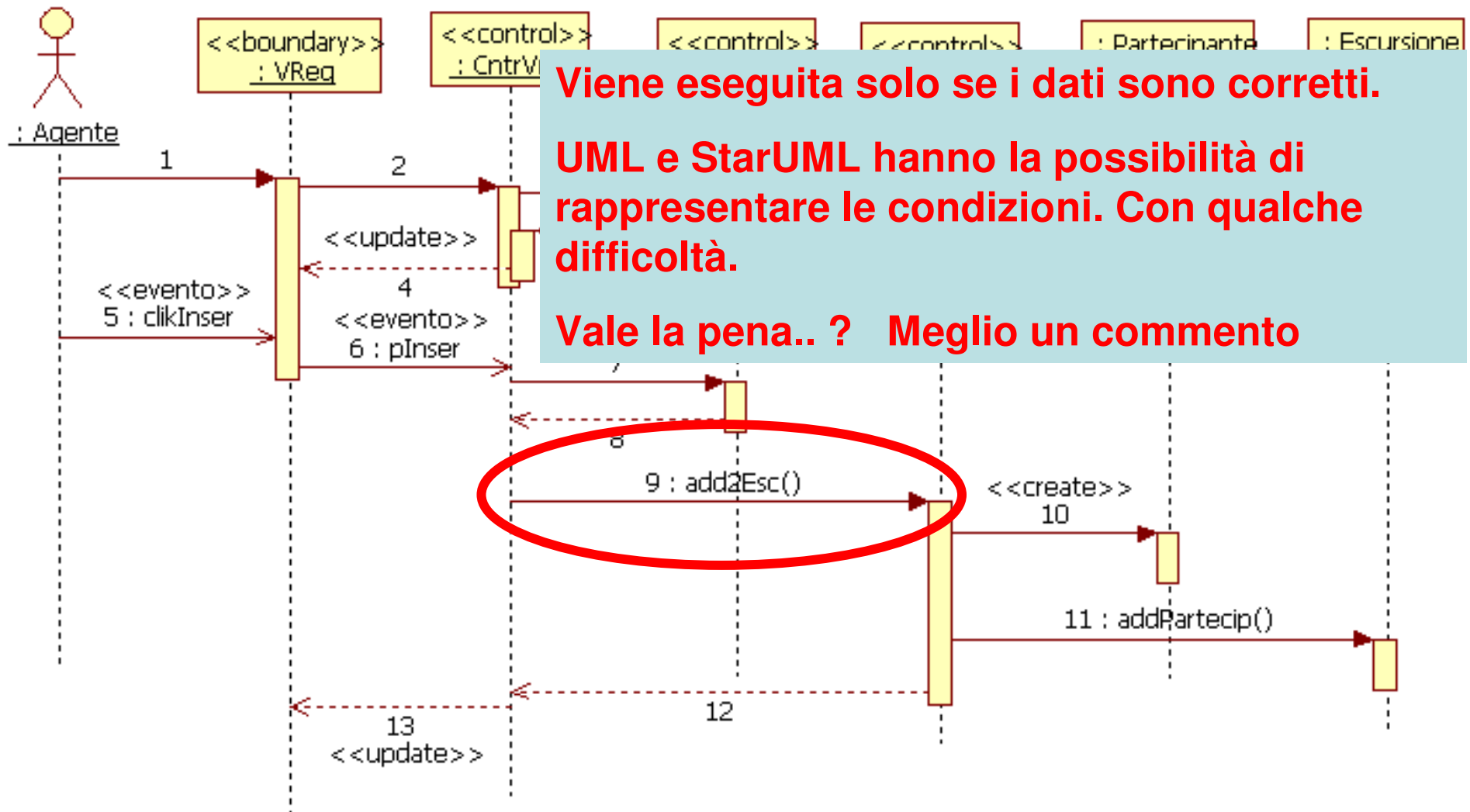


**Sarebbe un'iterazione.**

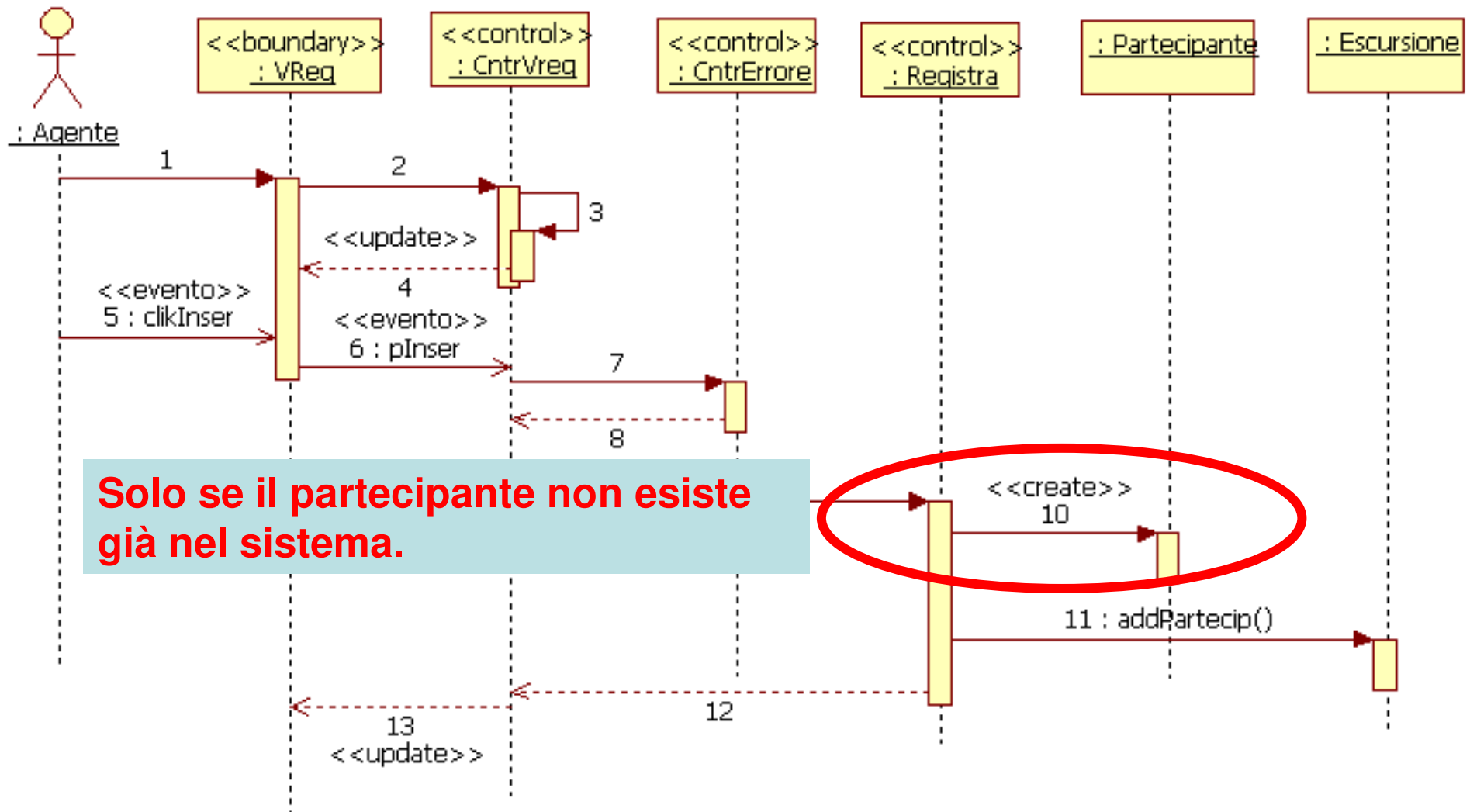
**UML e StarUML hanno la rappresentazione di cicli/loop. Con qualche difficoltà pratica**

**Ma vale la pena...??**

# SeqCU2 (Cfr RobCU2)

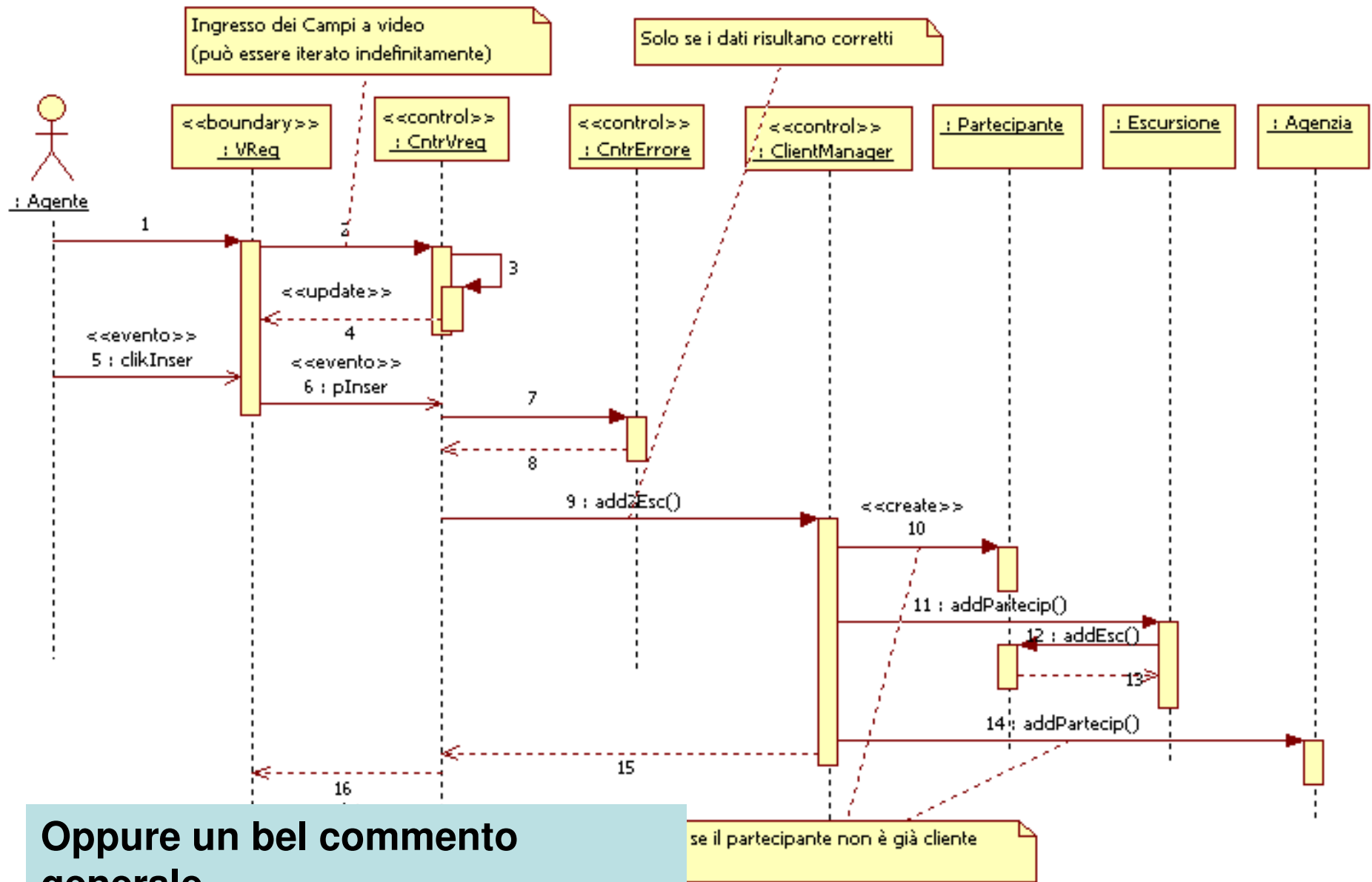


# SeqCU2 (Cfr RobCU2)

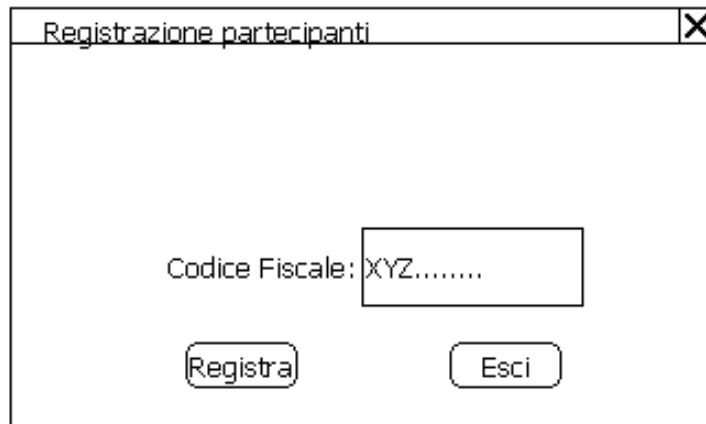




# SeqCU2 commentato



# ..Quanto andare a fondo



Registrazione partecipanti

Codice Fiscale: XYZ.....

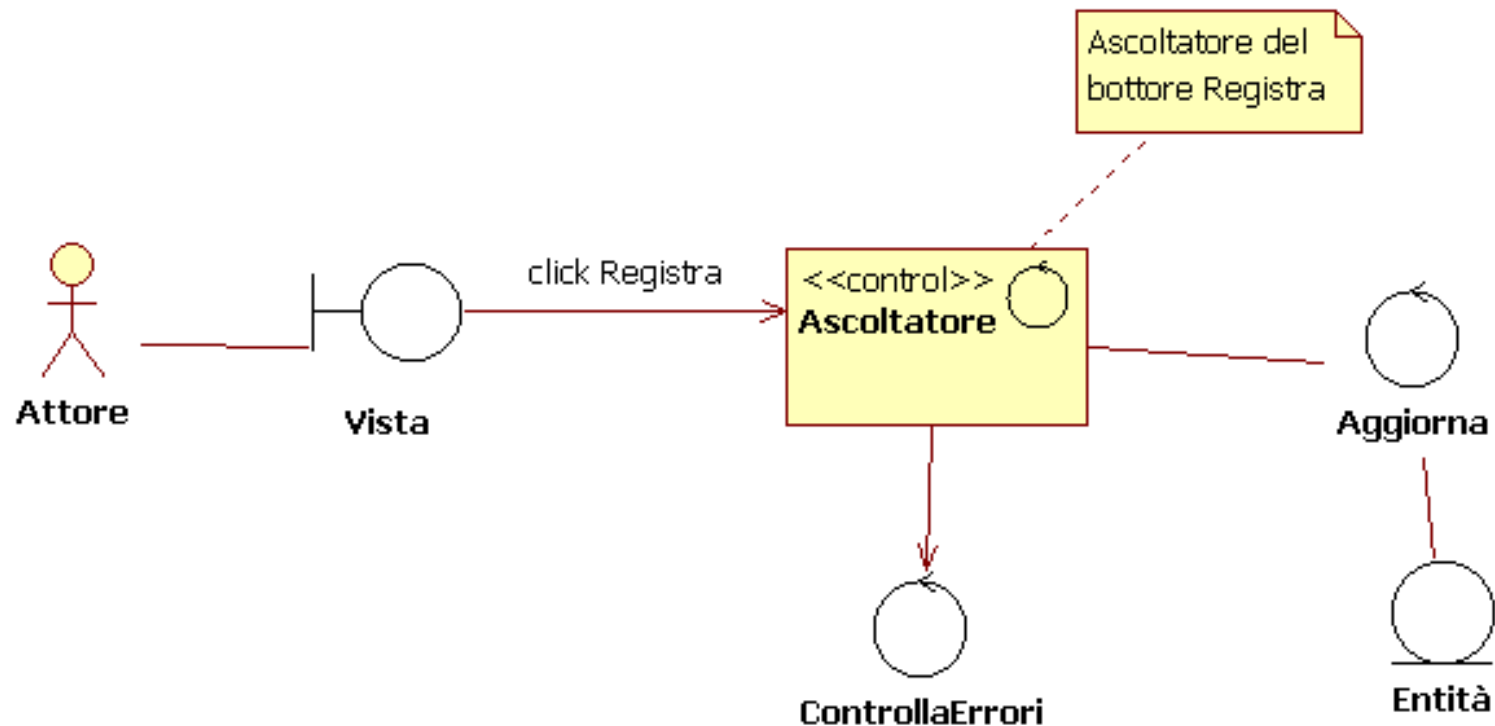
Registra Esci

## Semplificato

Nota:  
Sulla finestra ci saranno anche altri campi  
per selezionare/inserire informazioni

- Associare un ascoltatore al bottone Registra
- L'ascoltatore deve fare tutte le funzionalità relative alla registrazione oppure delegarne parte ad altri oggetti?

## ... Quanto andare a fondo



Piuttosto che come oggetti indipendenti riguardiamo  
“Aggiorna” e “ControllaErrori” come funzionalità.

Chi le svolge?

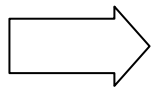
## ...Quanto andare a fondo

- Aggiorna:
  - Ragionevolmente è una funzione svolta da un oggetto a parte (p.e.: il gestore dei partecipanti)
- ControllaErrori
  - Ragionevolmente è una funzione da svolgere entro lo stesso ascoltatore
    - Validazione formale dei campi immessi
    - Eventuale presentazione di finestre con segnalazioni sugli errori
      - Questi controlli vengono praticamente a costo zero con i toolkit/librerie di interfaccia

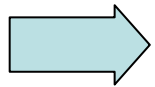
## ...esempio

```
r2Btn = new JButton("Registra");  
r2Btn.addActionListener(new ActionListener() {  
    public void actionPerformed(ActionEvent e) {
```

Verifica se è stata  
selezionata una riga  
di una tabella



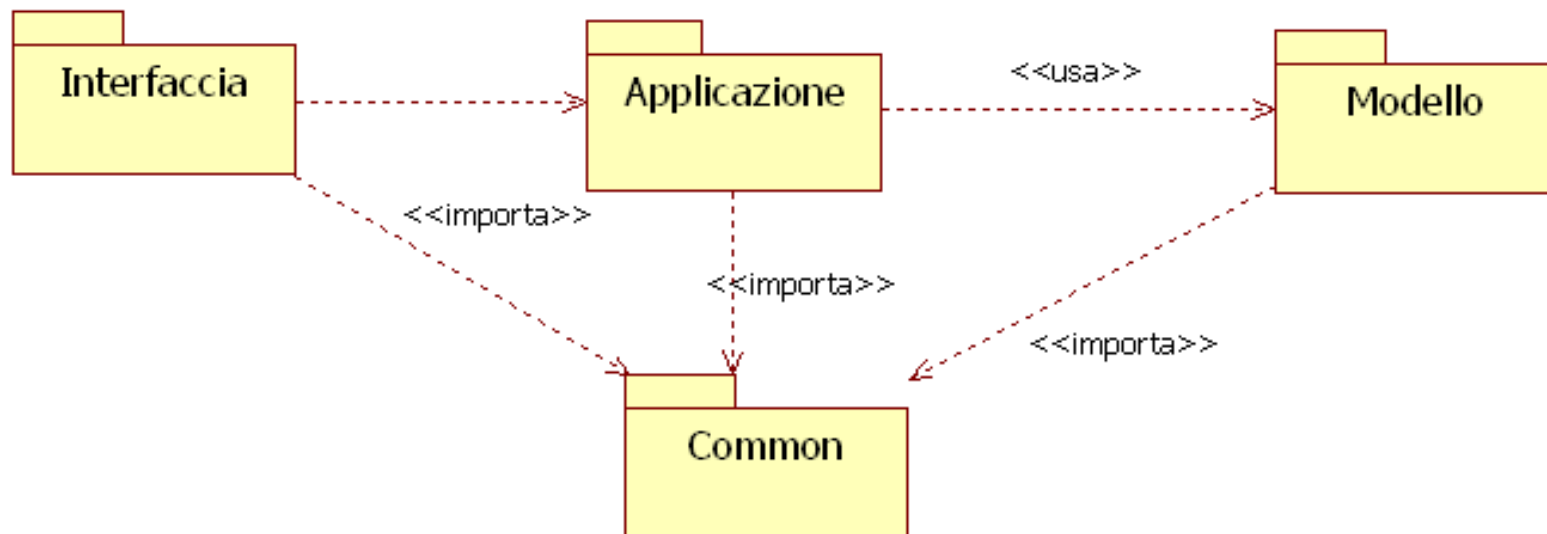
```
        //controllo ingressi, p.e.:  
        if(...getEscursioniTable().getSelectedRow() == 1)  
            ....showMessageDialog(..., "Selezionare un'escursione");
```



```
        //altre azioni tra cui:  
        clientManager.addNewPartecipante(partecipante);  
        escManager.addCliente2Esc(escursione, partecipante);  
    }
```

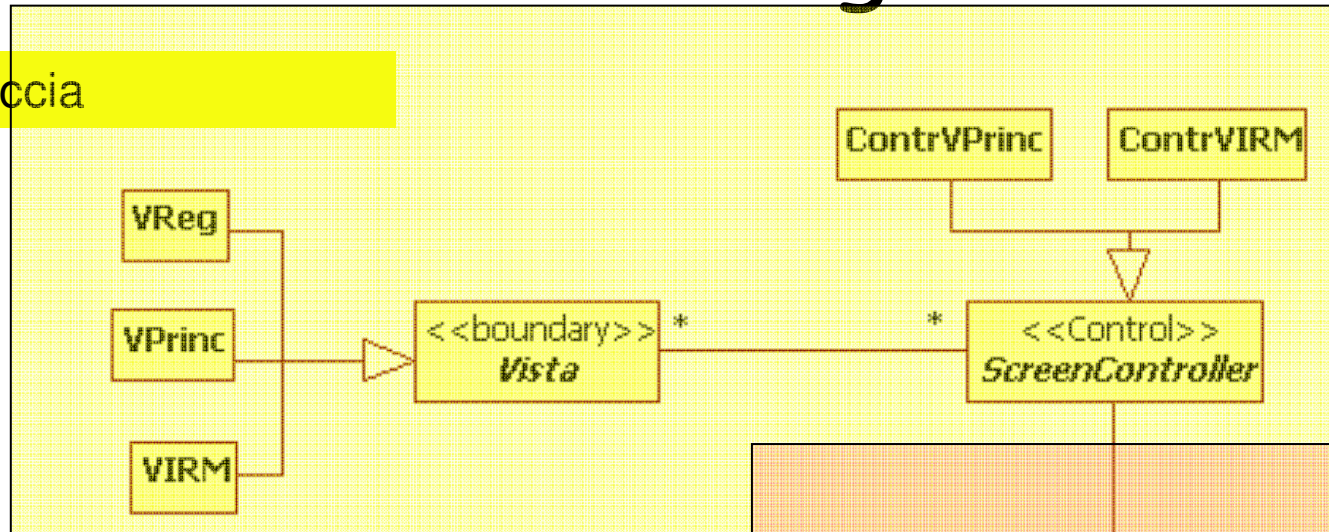
Funzionalità assegnate ad altri  
controllori

# Packages

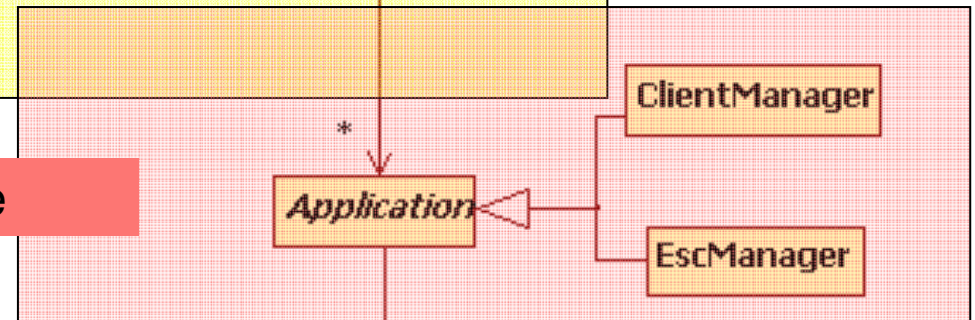


# Packages

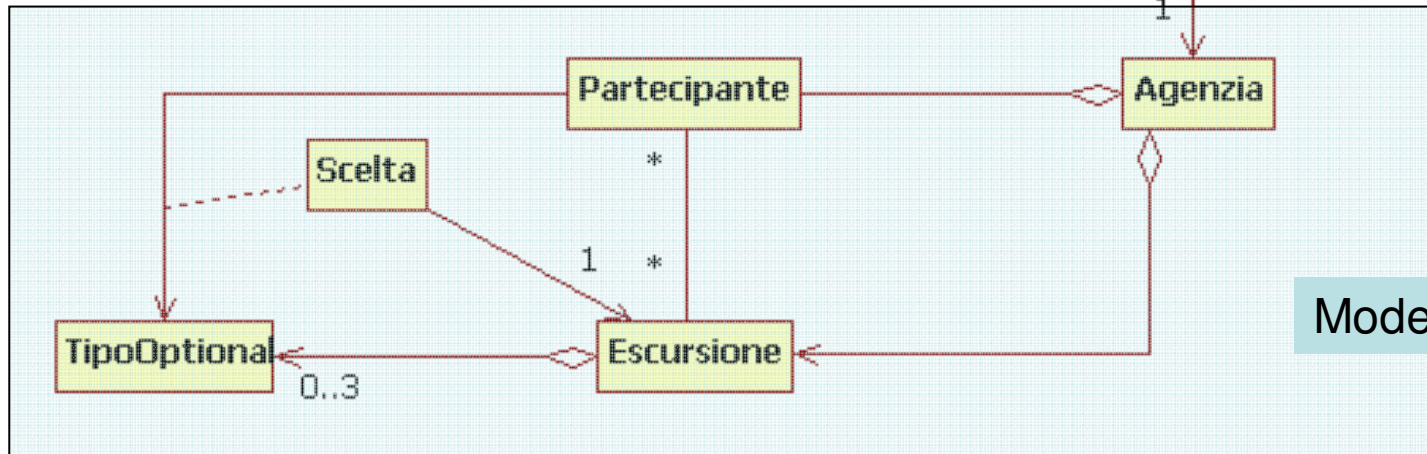
Interfaccia



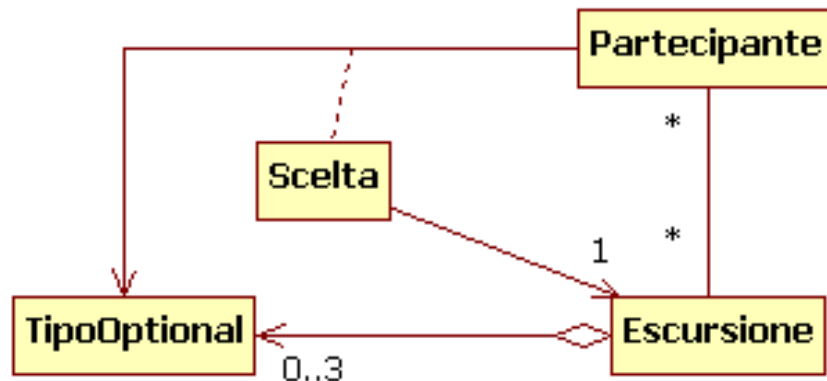
Applicazione



Modello

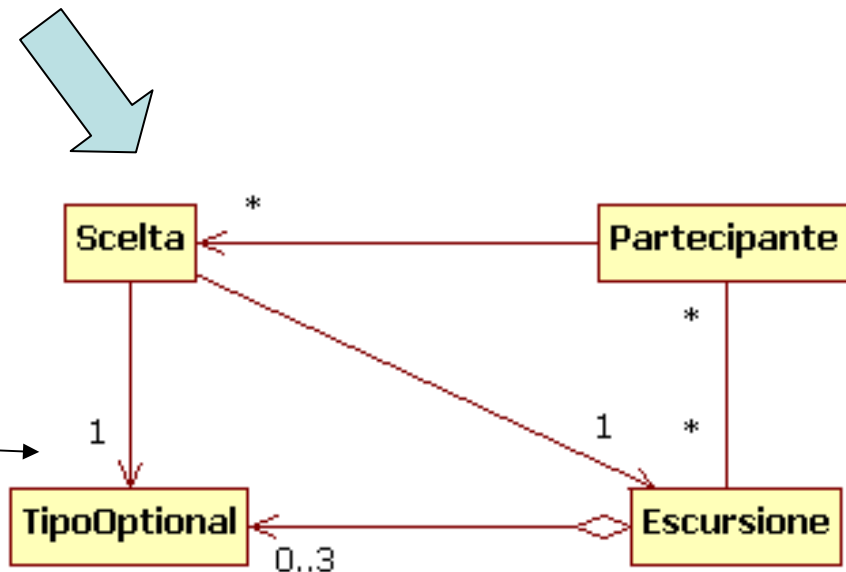


# Approfondiamo il modello



Come si realizza la scelta?

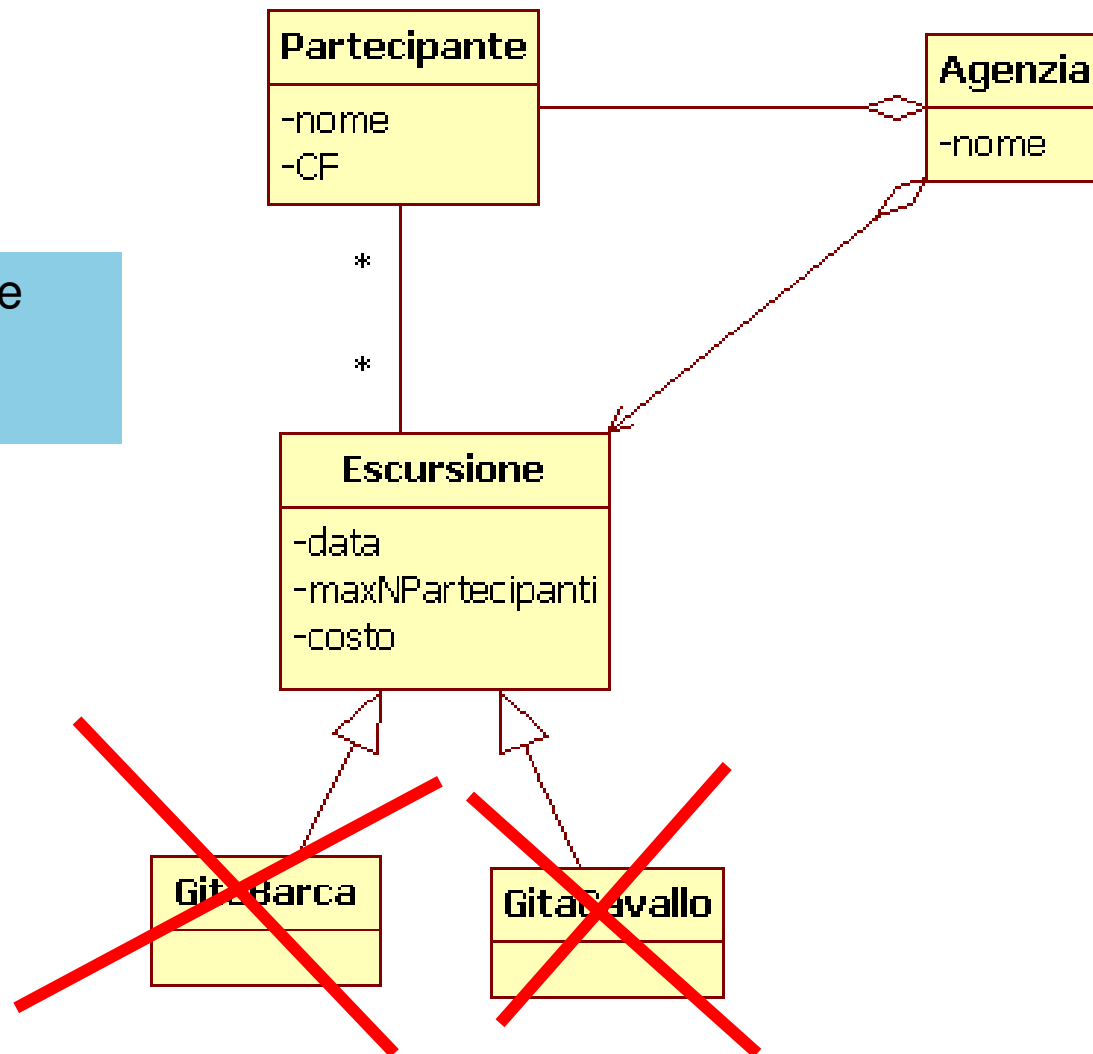
Oppure: .0..3 qui

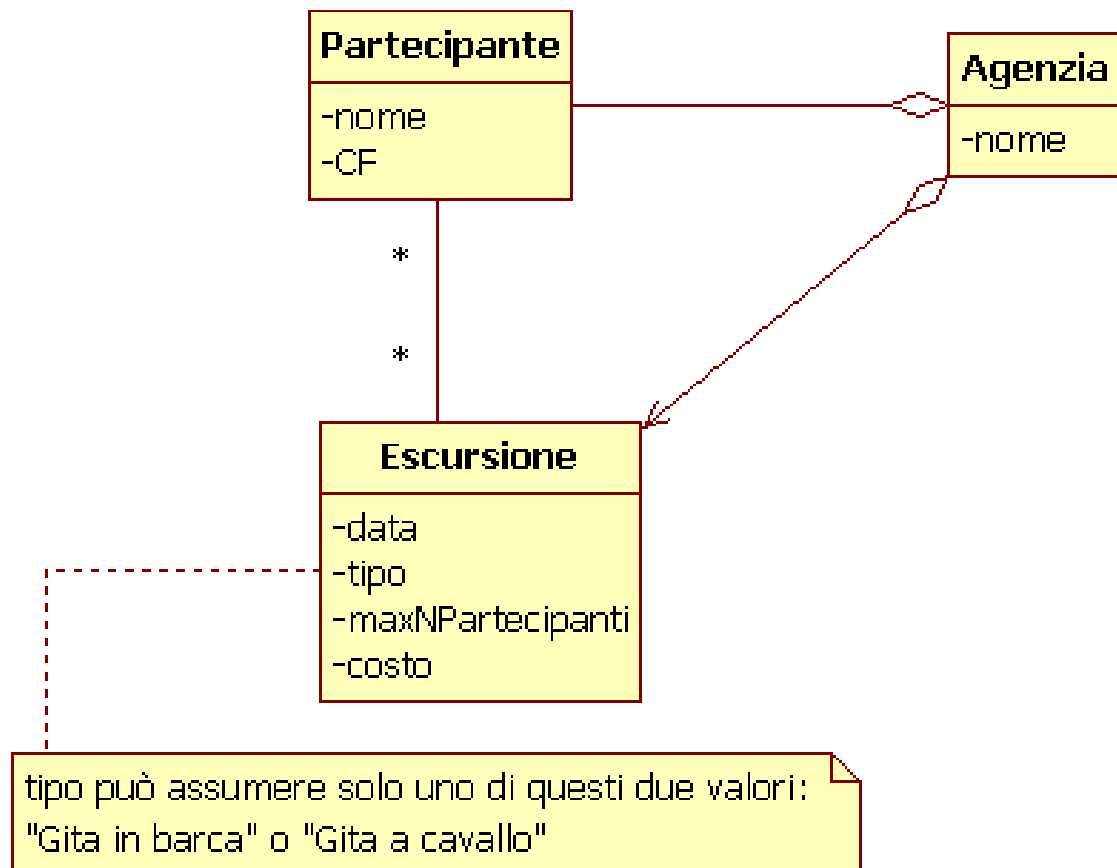




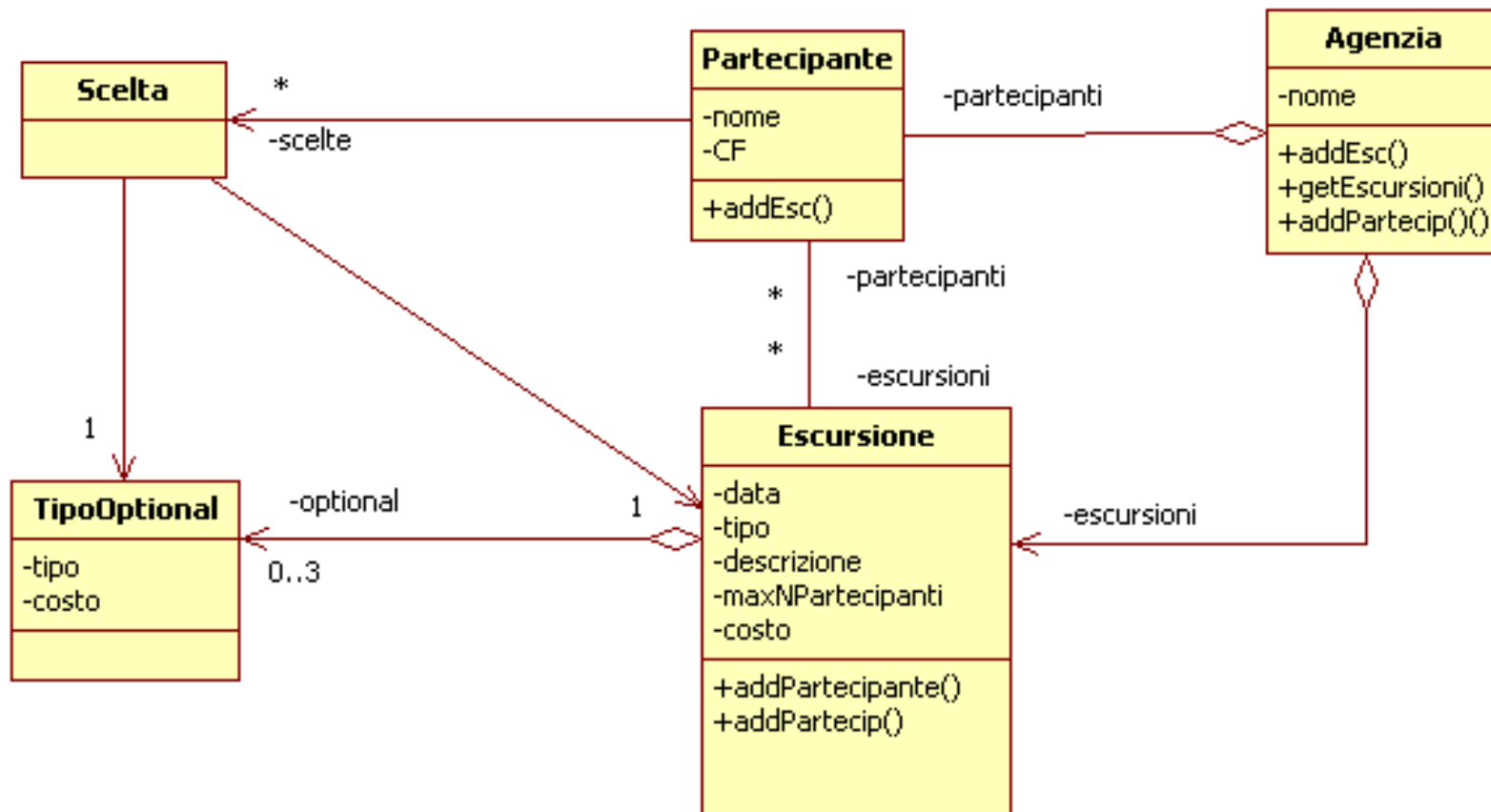
# Approfondiamo il modello

Ha senso derivare le due sottoclassi di Escursione?



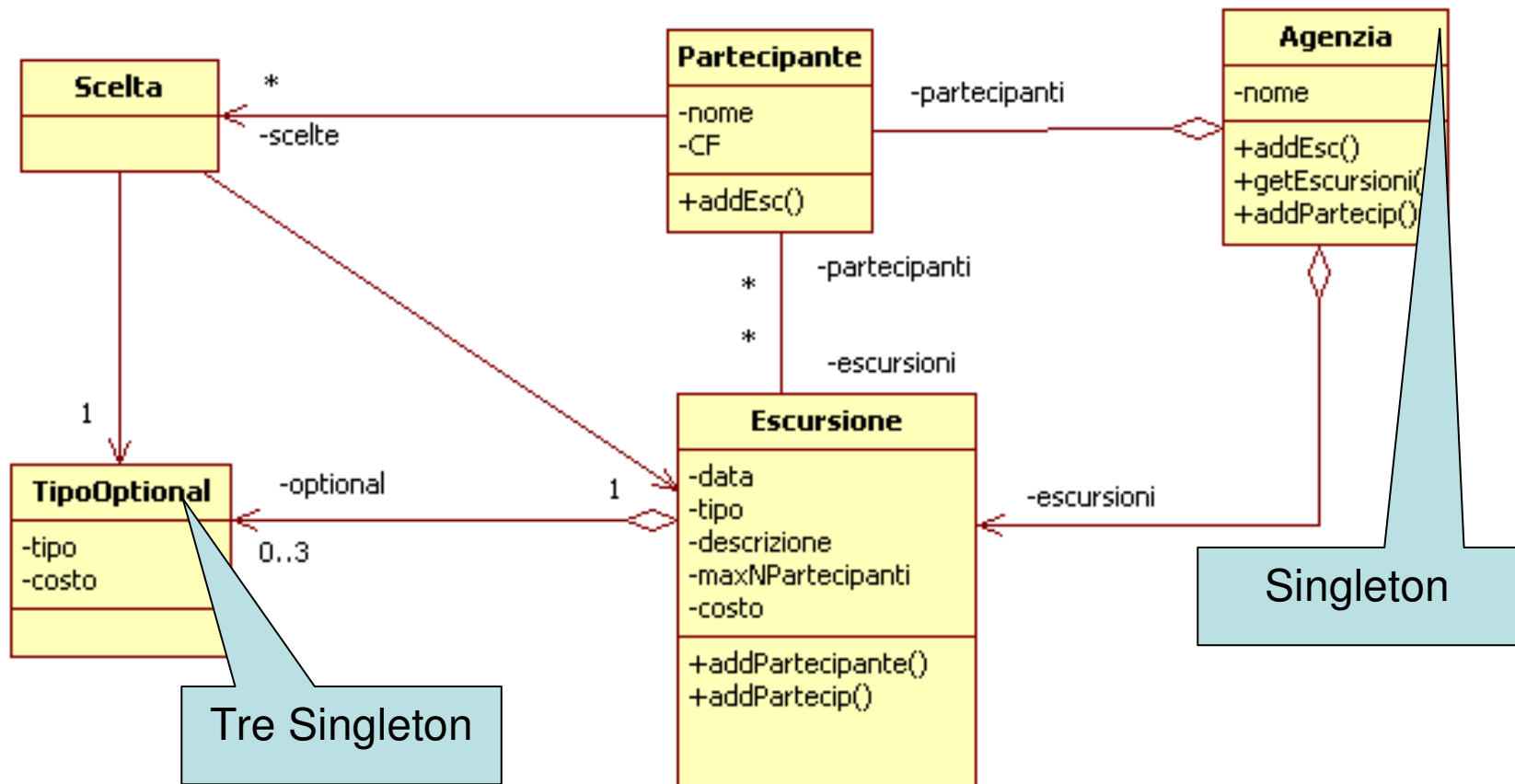


# Attributi ..e metodi (qui solo quelli dai Diagr Seq)



**NB:** l'attributo **descrizione** ci serve ad individuare univocamente gli oggetti della classe Escursione

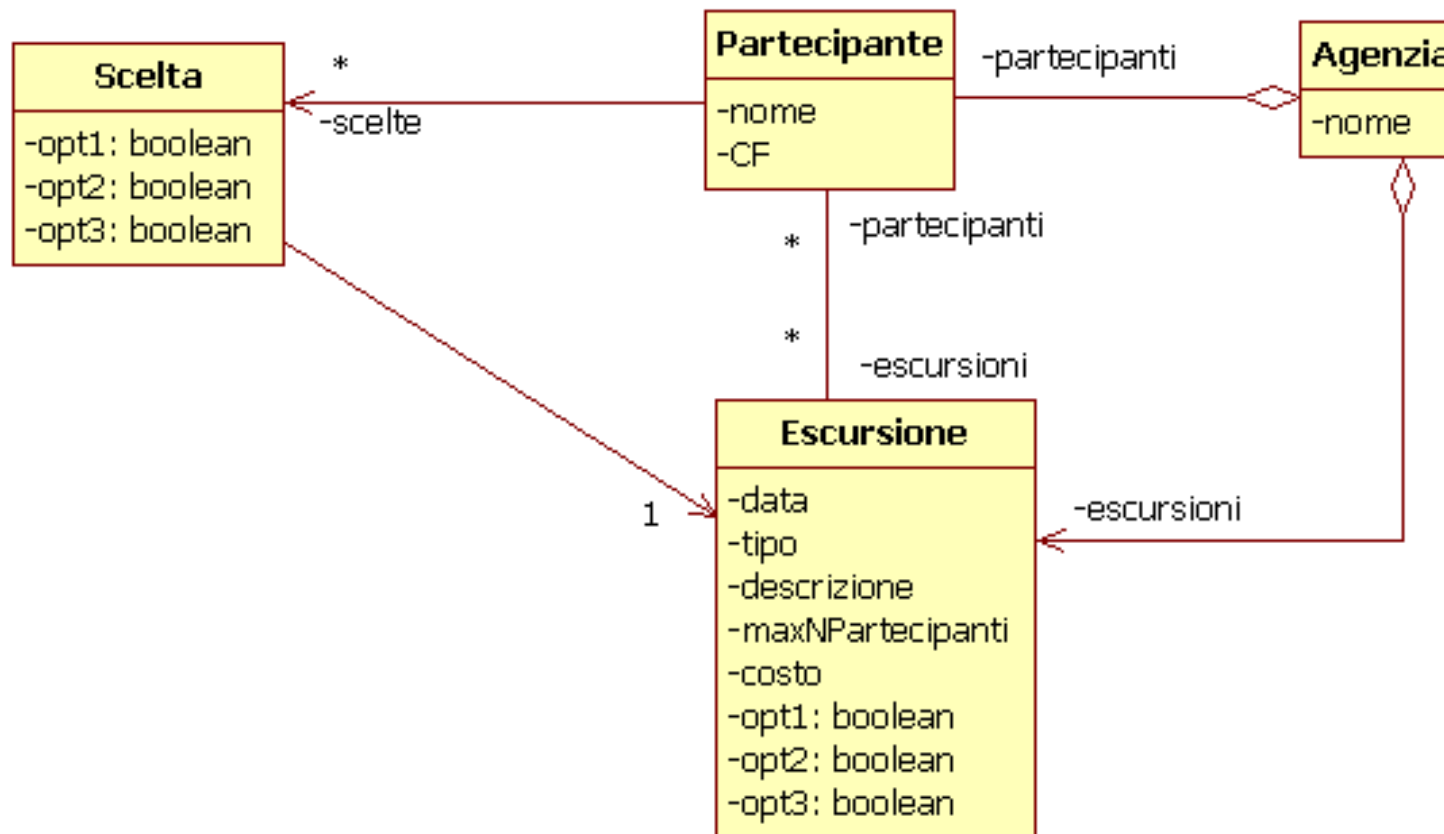
# Attributi ..e metodi (qui solo quelli dai Diagr Seq)



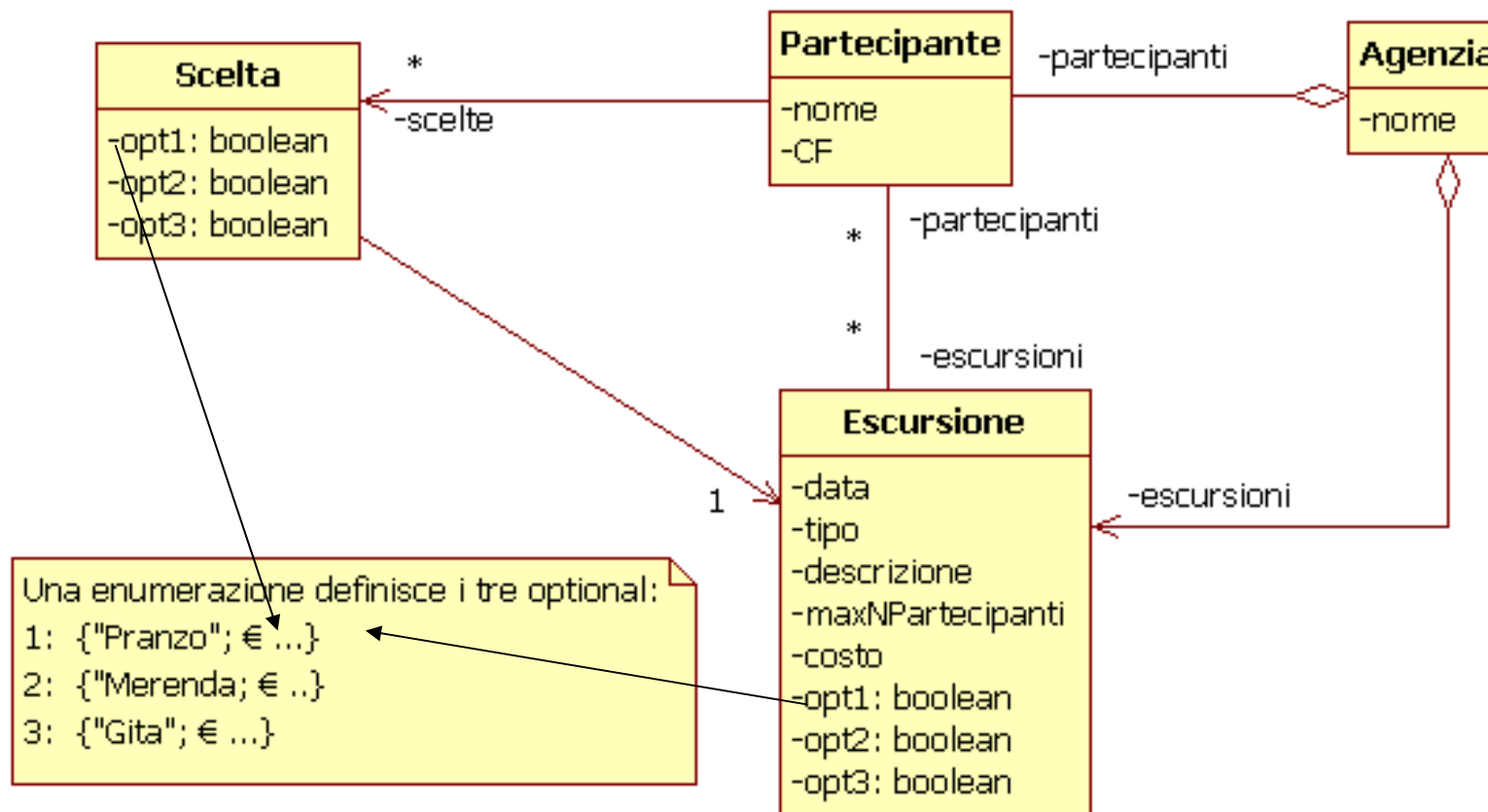
E' buona regola prevedere metodi get e set di ogni attributo.

P.e: `getDescrizione()`, `getTipo()`, ecc. per **Escursione**

# Perché non così?



# Perché non così?

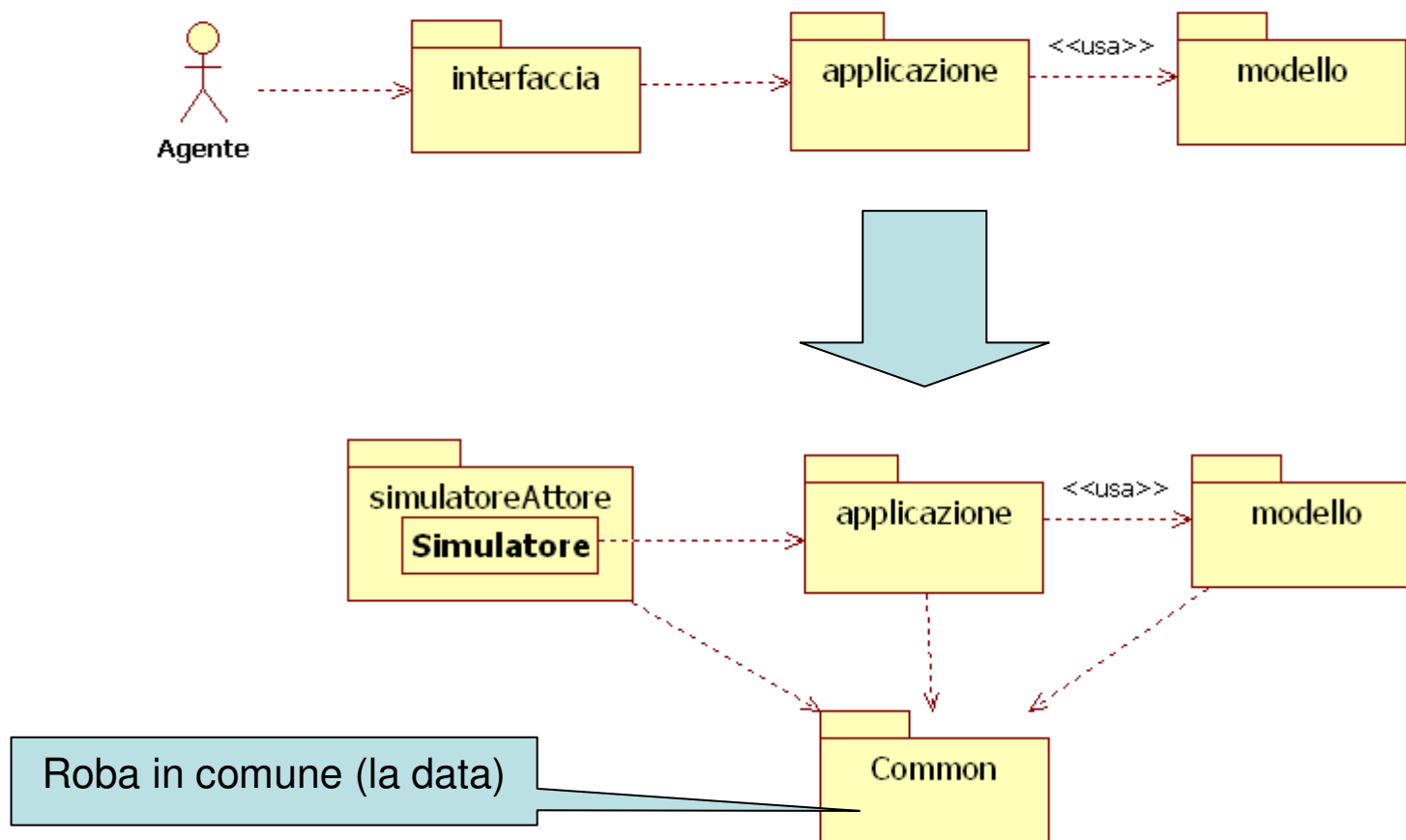


# Una prima implementazione

- Realizzata in stile “Esercizio di esame”
  - Senza GUI
  - Simulando GUI e attore (chiamando direttamente i metodi delle classi applicative)
  - Il programma di simulazione può essere il **main**
- Esempio

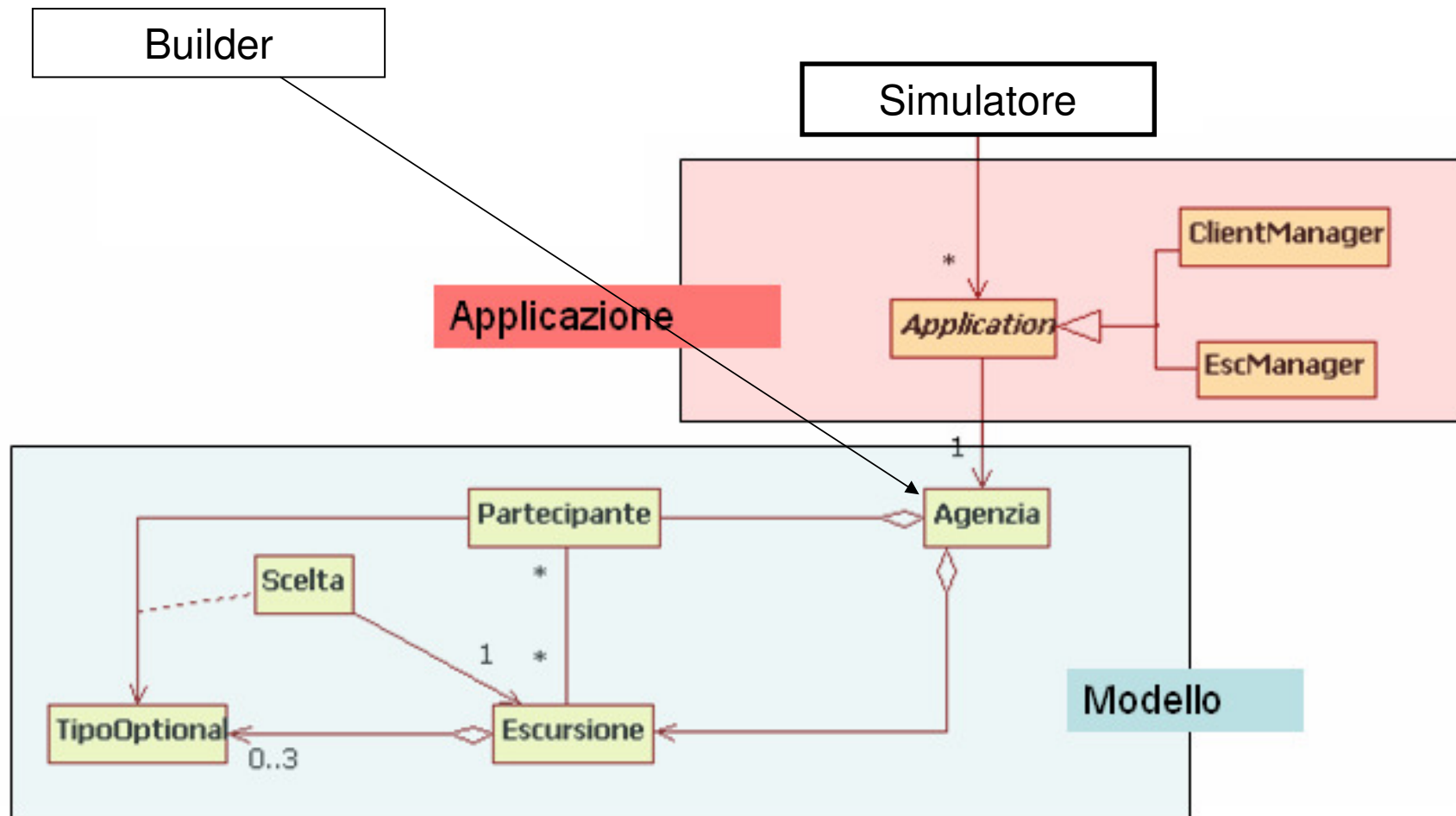
```
//cM è il ClientManager  
cM.addPartecipante2Agenzia(new  
    Partecipante("AMDALD82...", "Aldo"));
```

# Cosa cambia





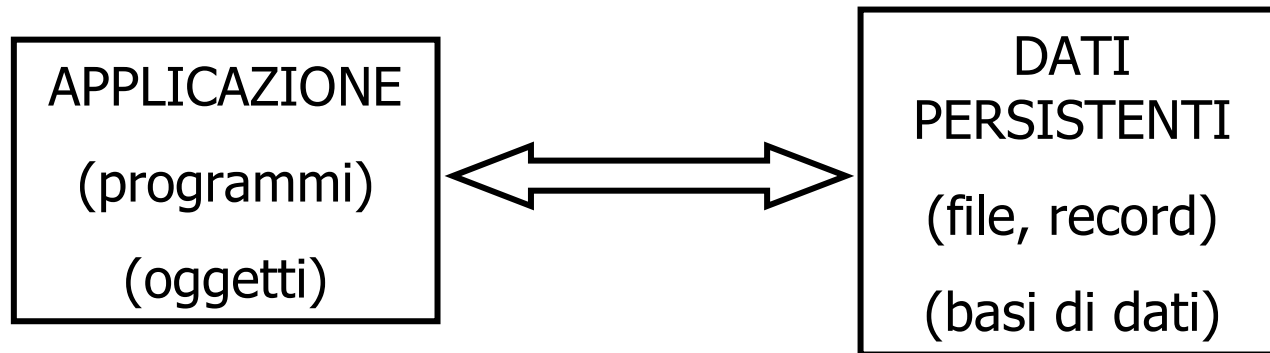
# Come diventa



# Come si procede

- Si segue il processo descritto in precedenza
  - Modello di dominio
  - Casi d'uso, analisi robustezza, diagrammi di sequenza
  - (si generano gli stub delle classi tramite StarUML)
  - Si passa allo sviluppo
- Per lo sviluppo
  - TDD possibilmente
    - Top-down, bottom-up ??
  - Procedere in modo incrementale
  - (vedere l'esempio)

# La persistenza



- I programmi trattano oggetti
  - (Stato, comportamento, associazioni)
- La base di dati è fatta di tabelle
  - (Valori, relazioni )

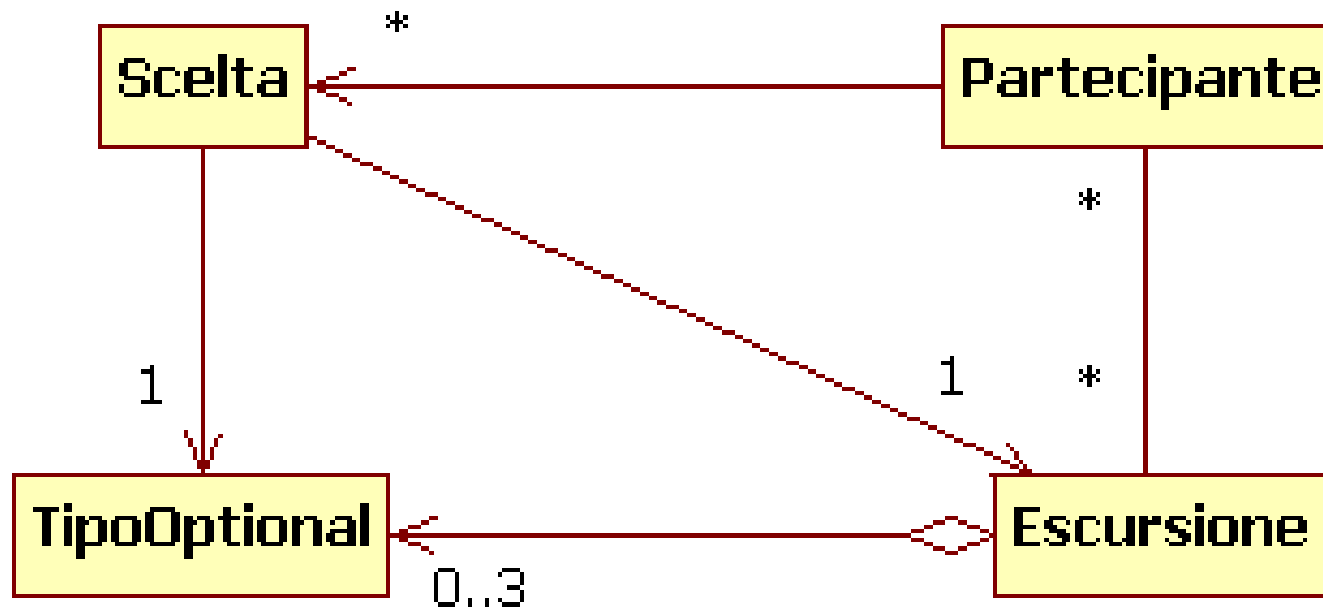
Vedi  
Persistenza.ppt

# Torniamo all'Agenzia

- Se c'è la base di dati il contenitore degli oggetti di dominio (l'Agenzia) non serve più
  - Gli oggetti sono permanenti nella forma di dati in tabelle
  - Quando un oggetto viene istanziato viene presa la riga che lo rappresenta in tabella e i valori dei campi diventano i valori degli attributi dell'oggetto
    - Nasce un oggetto transiente
  - In memoria solo gli oggetti che interessano in un dato momento
  - L'interfaccia al contenitore degli oggetti *permanenti* è il DBMS

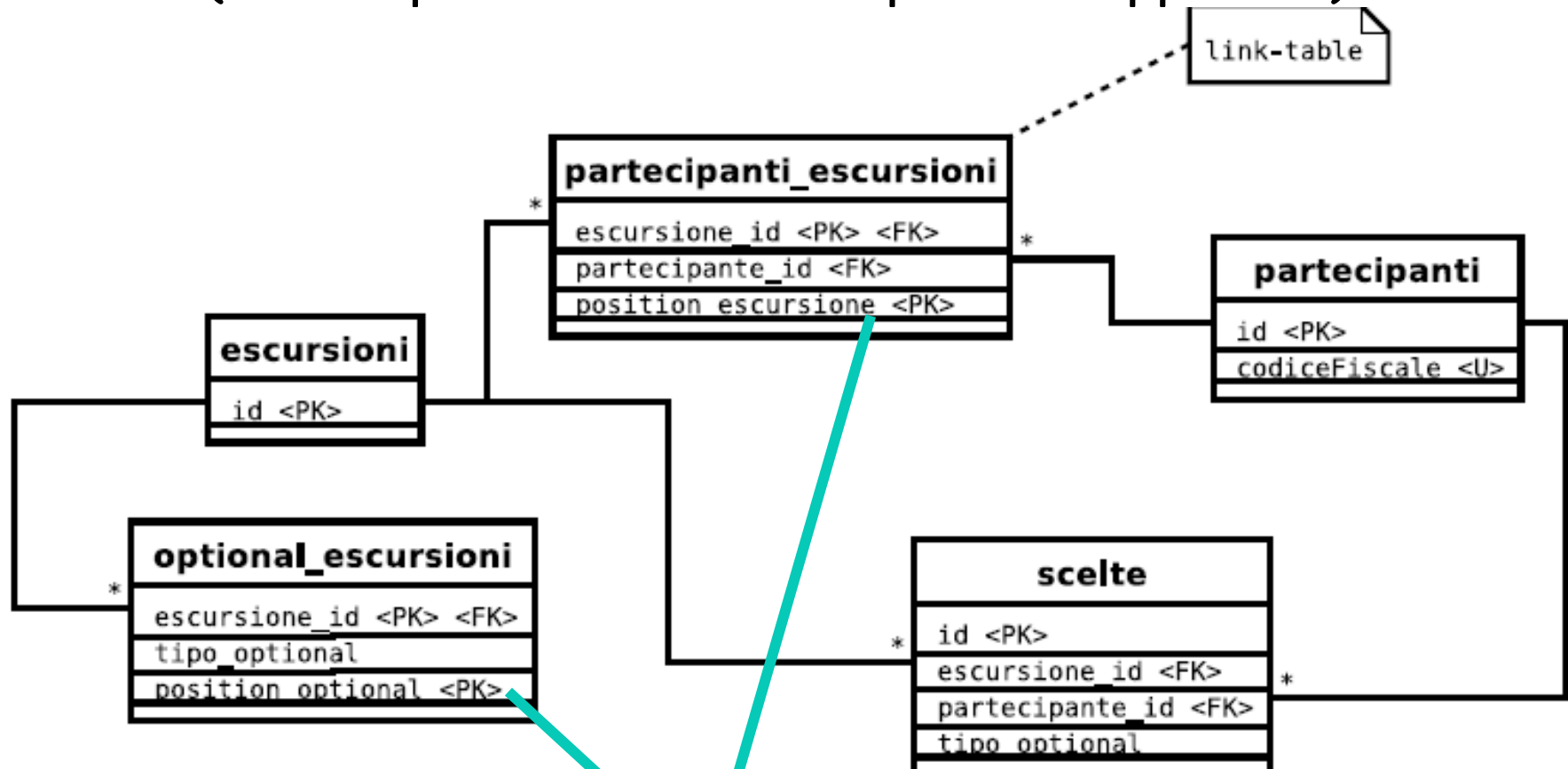
# Il modello finale (con DB)

- Agenzia non serve



# La base di dati (con Hibernate)

(solo le parti di interesse per la mappatura)



Per come Hibernate gestisce le relazioni 1:n o m:n

# DAO Escursioni

```
public interface EscursioneDAO extends
    GenericDAO<Escursione> {
    public List<Escursione> getByPartecipante(Partecipante
        p) ;
    public List<Escursione> findByDate(Data d) ;
}
```



## ... DAO Escursioni

```
public class EscursioneDAOImpl extends
    GenericAbstractHibernateDAO<Escursione>
    implements EscursioneDAO {

    public List<Escursione> getByPartecipante(
        Partecipante p) {
        return find("from Escursione e where :partecipante
            in elements(e.iscrittiLista)", "partecipante", p);
    }

    public List<Escursione> findByDate(Data d) {
        return find("from Escursione where data = :data",
            "data", d);
    }
}
```

# In EscManager

```
public class EscManager {  
    private EscursioneDAO eDAO;  
  
    public List<Escursione> getEscursioni (Partecipante p) {  
        return eDAO.getByPartecipante(p);  
    }  
  
    public List<Escursione> getListaEscursioni (int day,  
                                                int month, int year) {  
        return eDAO.findByDate (new Data (day, month, year));  
    }  
  
    public long getNumEscursioni () {  
        return eDAO.getCount ();  
    }  
}
```