

---

# *Analisi e progettazione Object Oriented con UML*

Esercizi

---

# Esercizio 0

---

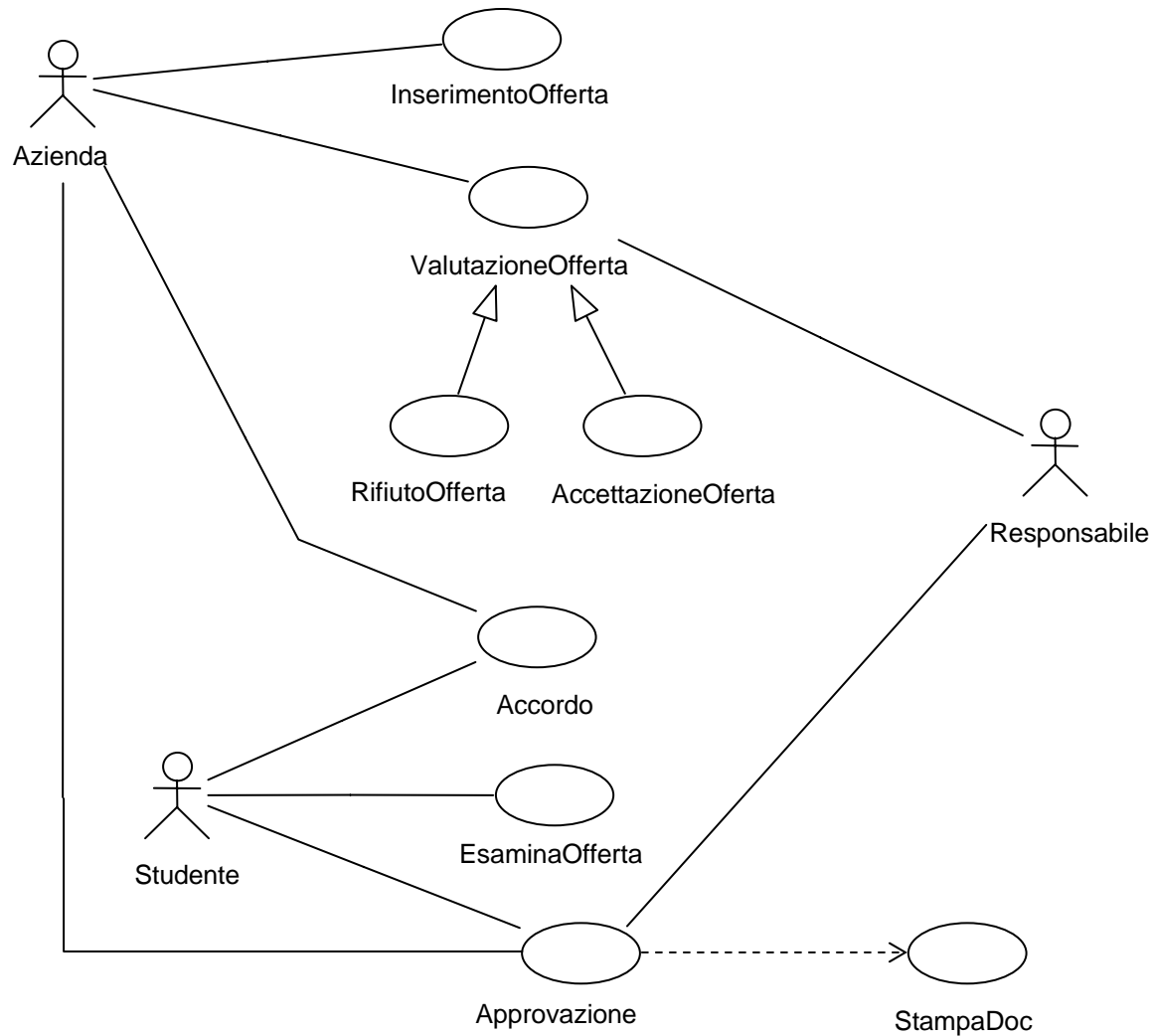
- Modelliamo il sistema di gestione dei tirocini di una università
  - Siamo interessati a rappresentare i requisiti utente, quindi usiamo gli use case diagrams.
-

# Gestione tirocini

---

- Le aziende interessate producono delle offerte di tirocinio.
  - Il responsabile dei tirocini approva o rifiuta le offerte.
    - ▶ I rifiuti sono notificati all'azienda proponente.
    - ▶ Le offerte accettate diventano visibili agli studenti.
  - Gli studenti visualizzano le offerte. In conseguenza di ciò possono accordarsi con l'azienda proponente.
  - Le aziende assegnano gli studenti graditi ai tirocini offerti.
  - Il responsabile dei tirocini approva o rifiuta gli accoppiamenti studente-tirocinio.
    - ▶ In caso di accettazione si stampa l'accordo che dovrà essere firmato da tutte le parti in causa.
-

# Use case diagram

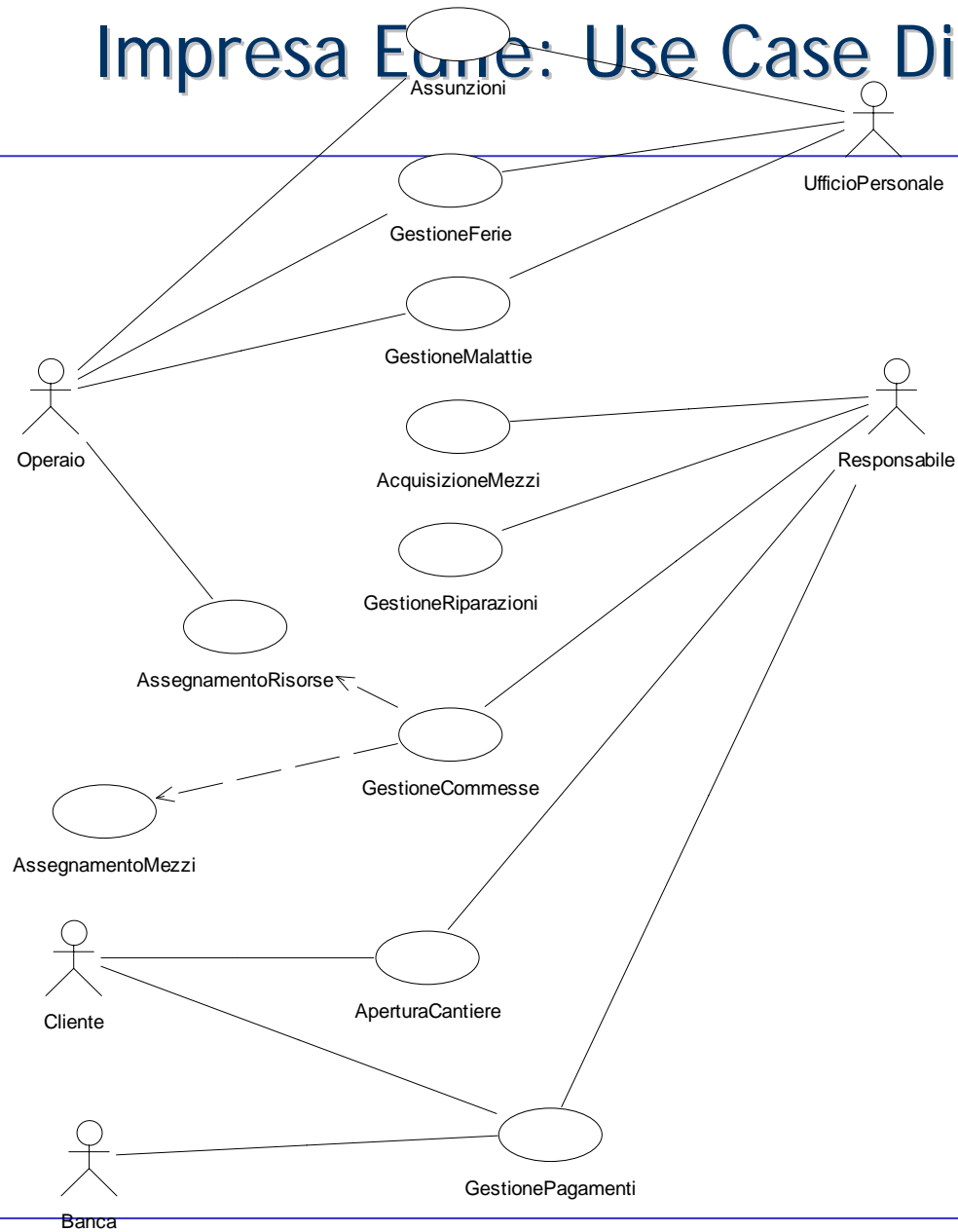


# Esercizio 1: Impresa edile

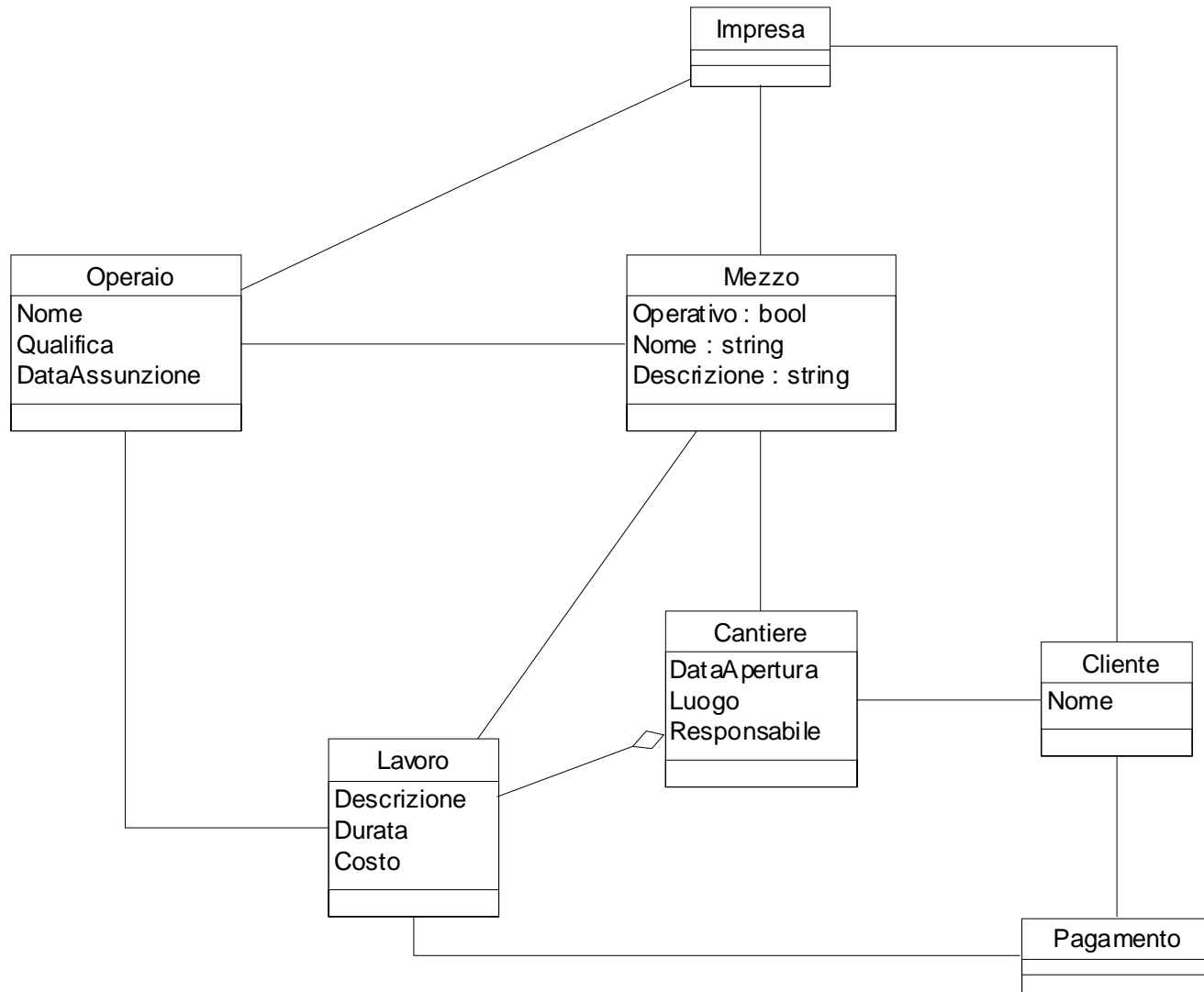
---

- Si definisca un semplice Use Case Diagram per modellare il sistema informativo di una impresa edile. Il sistema deve gestire gli operai, i cantieri, il parco mezzi e i clienti.

# Impresa Eune: Use Case Diagram



# Impresa Edile: Class Diagram



# Esercizio 2:

## Compagnia di Assicurazioni

---

- Descriviamo una compagnia di assicurazione
- Siamo interessati a descrivere il dominio del problema, con particolare riferimento agli aspetti di modellazione statica.
  - ▶ Produciamo un diagramma delle classi



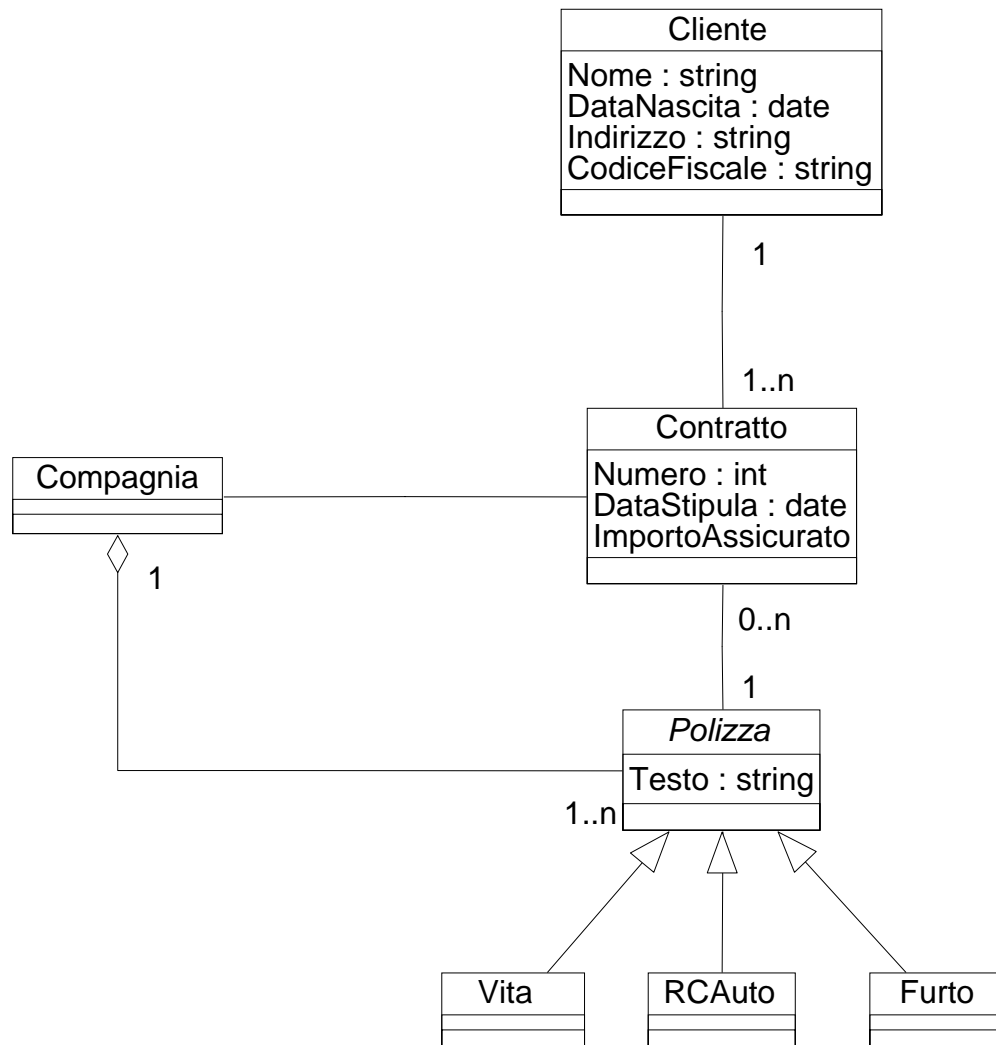
# La compagnia di assicurazioni

---

- La compagnia di assicurazioni stipula diversi tipi di polizze (RC auto, vita, rischi diversi).
  - La compagnia ha diversi clienti, ciascuno dei quali può sottoscrivere più contratti.
-

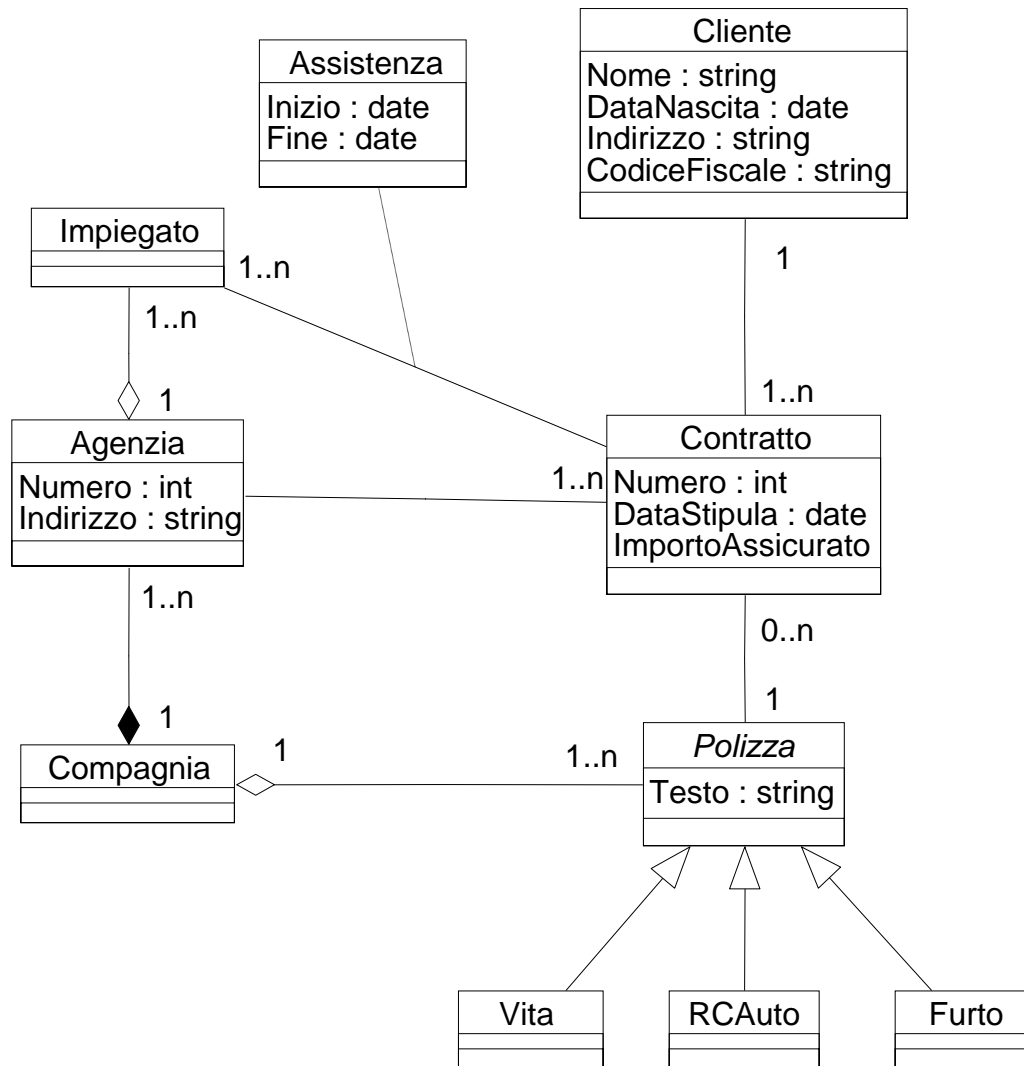
# Compagnia di Assicurazioni: Class diagram (1)

---



# Compagnia di Assicurazioni:

## Class diagram (2)



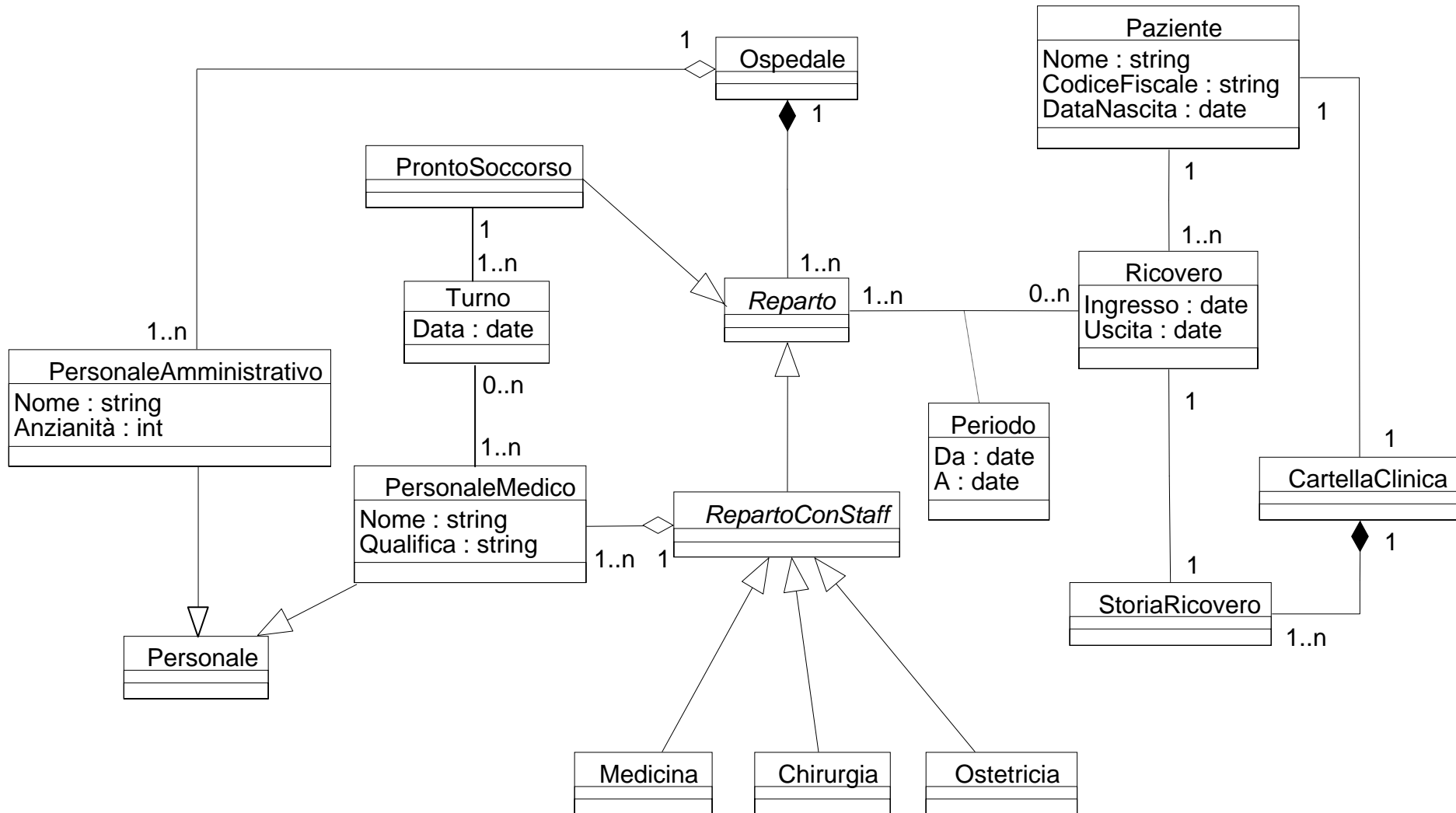
## Esercizio 3:

# Struttura di un Ospedale

---

- Descriviamo la struttura (statica) di un ospedale con un diagramma delle classi
  - L'ospedale è organizzato in reparti, ciascuno dei quali ha uno staff e dei pazienti.
  - I pazienti hanno una cartella clinica associata.
  - Ciascun paziente può essere ricoverato più volte.
  - Si vuole tenere traccia dei ricoveri e delle dimissioni.
  - I reparti hanno uno staff medico fisso. Il pronto soccorso no: tutto lo staff, secondo opportuni turni, vi lavora.
-

# Ospedale: class diagram



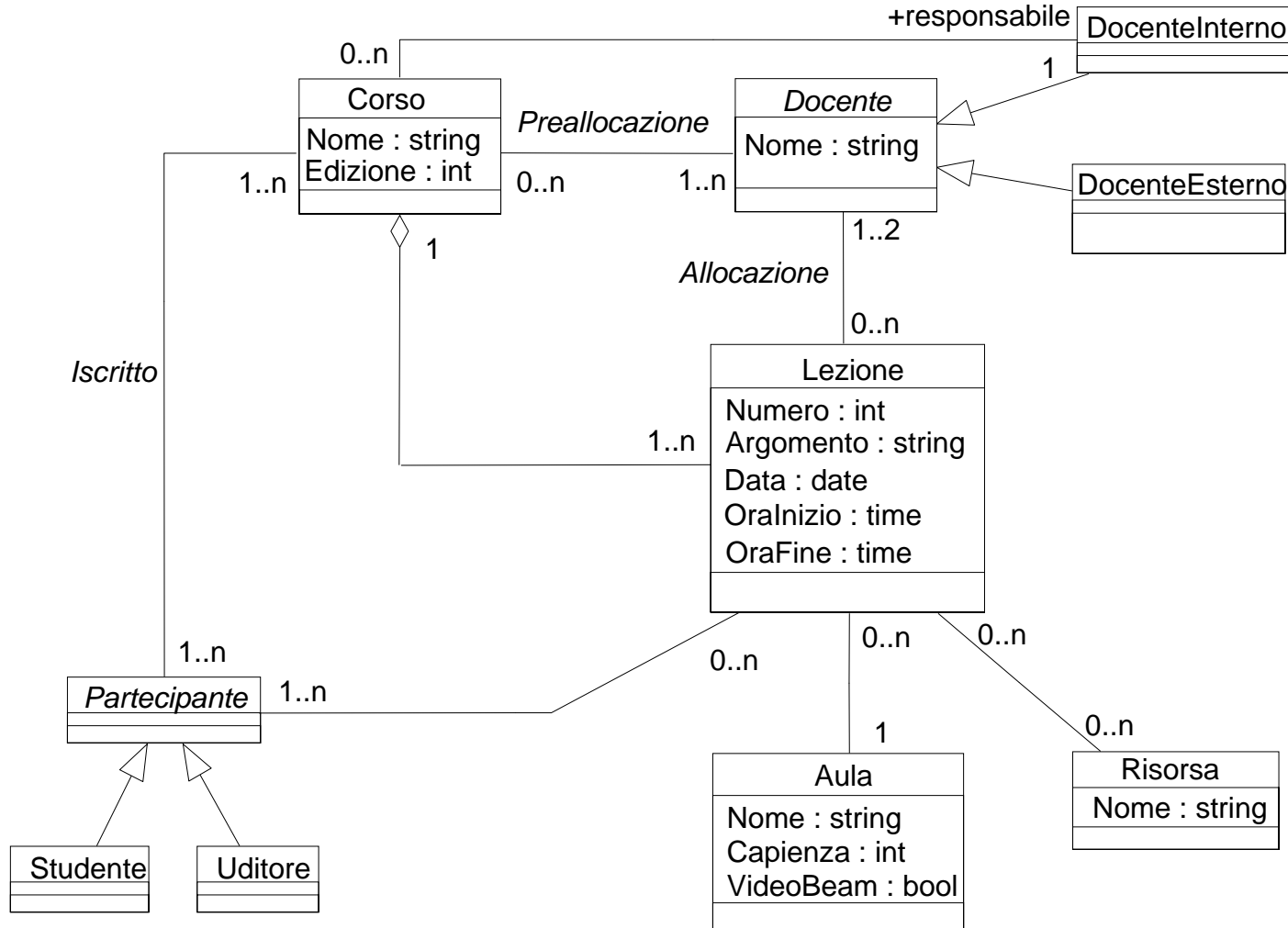
## Esercizio 4:

### Sistema di gestione corsi (1)

---

- Si vuole realizzare un sistema per la gestione amministrativa dei corsi di Master
    - ▶ Ciascun corso può essere svolto in più edizioni
    - ▶ Ogni edizione si articola in un certo numero di lezioni e ogni lezione può essere tenuta da più docenti. Possono esistere docenti assegnati al corso che, per vari motivi, non tengono alcuna lezione (ad esempio vengono “preallocati” per possibili sostituzioni). Ogni lezione ha una data, un’orario di inizio, una durata e un’aula assegnata
    - ▶ Ogni lezione può richiedere l’utilizzo di risorse aggiuntive (es. VideoBeam, PC, etc.)
    - ▶ I docenti si distinguono fra docenti interni ed esterni. Ciascun corso ha come responsabile un docente interno
    - ▶ Ogni edizione di un corso ha un certo numero di partecipanti, iscritti a quella specifica edizione, distinti fra studenti di Master e uditori esterni
-

# Sistema di gestione corsi: class diagram



## Esercizio 5: Palestra

---

- Si progetti un sistema informativo per la gestione di una palestra.
  - Ad ogni cliente, la palestra chiede nome, cognome, età, indirizzo e professione, nonché la tipologia di abbonamento e i servizi richiesti.
  - La palestra offre abbonamenti mensili e annuali (13 mesi).
  - I clienti possono usufruire della sala pesi, della piscina, della parete da arrampicata, della sauna e del bagno turco. Sala pesi, sauna e bagno turco sono compresi nell'abbonamento base; gli altri servizi sono venduti in aggiunta alla quota base.
-



## Palestra (2)

---

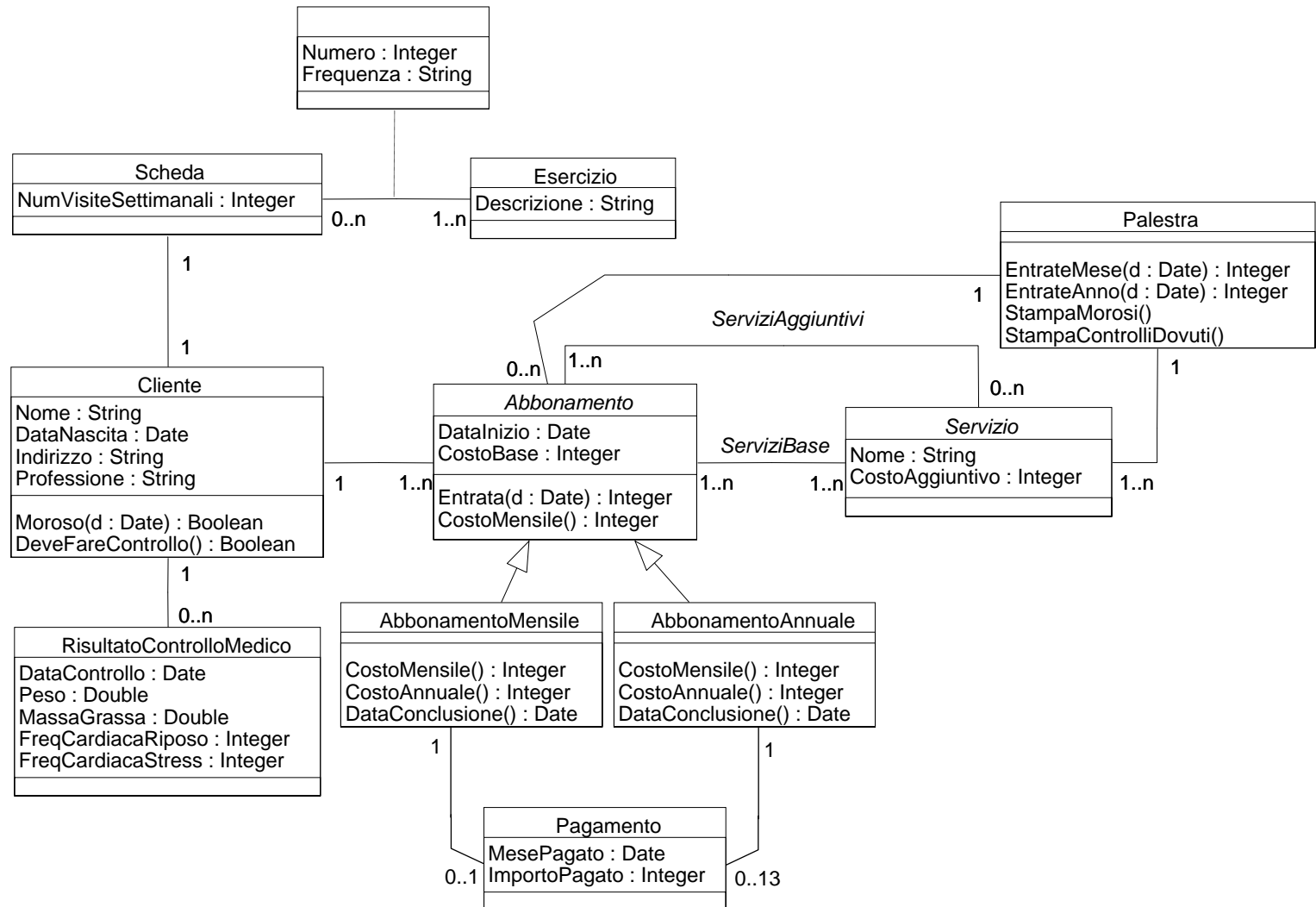
- Ad ogni cliente è associata una scheda - per la sala pesi - che definisce gli esercizi da compiere, il numero di ripetizioni e la frequenza. La scheda può essere organizzata su una visita alla palestra a settimana oppure su due. Nel caso i clienti siano abbonati a servizi aggiuntivi (ad esempio, la piscina), la scheda può considerare anche questi servizi e, di conseguenza, il numero di visite aumenta.
-

## Palestra (3)

---

- La palestra offre, a chi fosse interessato, uno staff medico per assistere i clienti, stabilirne il grado di forma e definire diete opportune. Chi richiede questo servizio, deve sottoporsi a controlli mensili e i risultati vengono opportunamente archiviati: peso, massa grassa, elettrocardiogramma, battiti del cuore a riposo e sotto sforzo.
  - Il sistema deve essere in grado di elaborare il bilancio complessivo (entrate dagli abbonati) della palestra sia mese per mese, che alla fine di ogni anno, e deve sollecitare eventuali clienti morosi o che devono sottoporsi al controllo mensile.
-

# Palestra: Diagramma delle classi



## Esercizio 6: Bookmaker

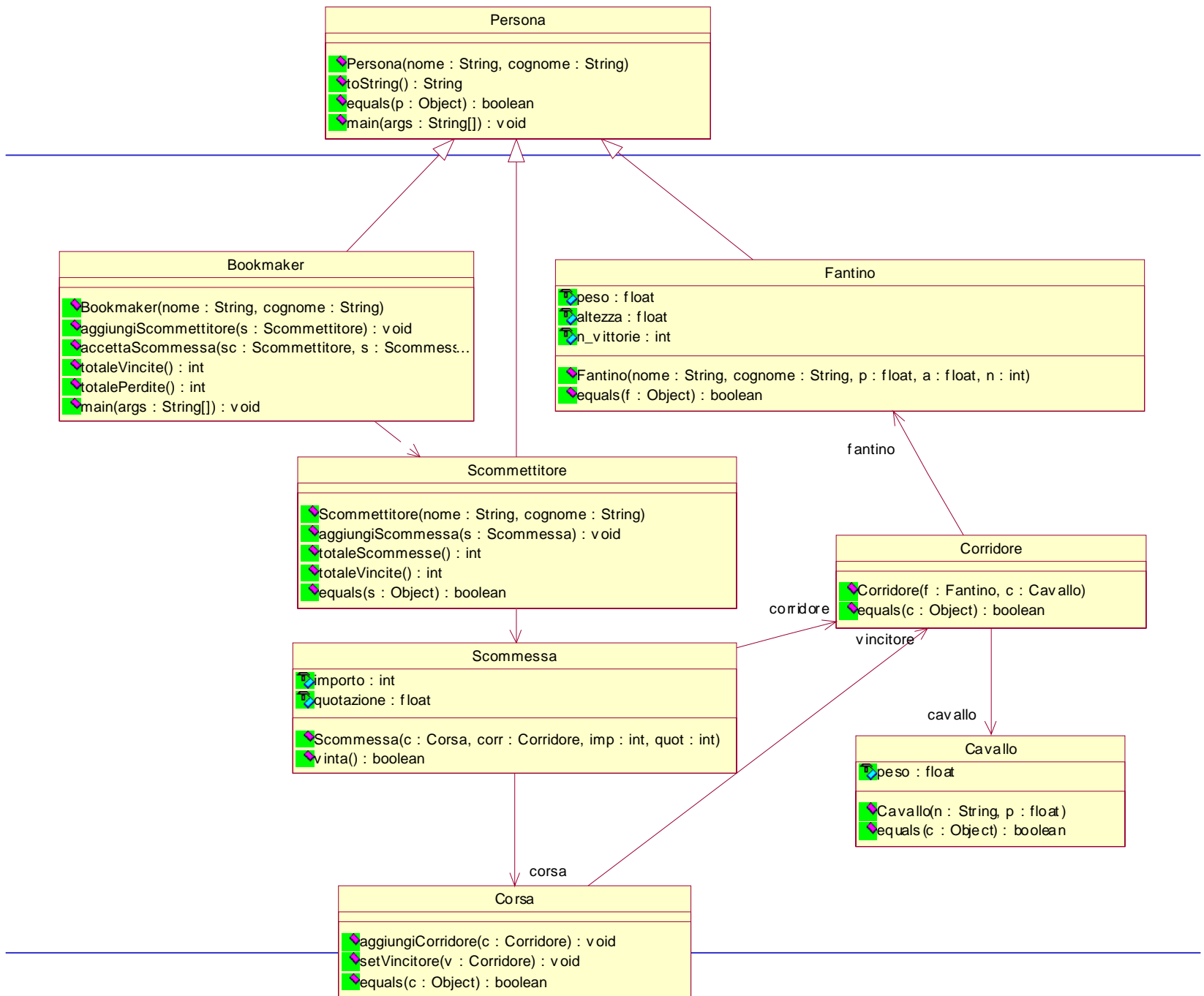
---

- Un Bookmaker vuole realizzare una applicazione per la gestione di una lista di scommettitori sulle corse dei cavalli.
  - Una corsa è caratterizzata da un certo numero di corridori e da un vincitore, che deve essere un corridore partecipante alla corsa
  - Un corridore è un cavallo guidato da un fantino
  - Ciascuno scommettitore può effettuare un numero qualsiasi di scommesse
  - Ciascuna scommessa è riferita ad una singola corsa, ad un particolare corridore presente nella corsa ed è caratterizzata da un importo e da una quotazione
-

## Bookmaker (2)

---

- Il sistema deve consentire
    - ▶ la gestione di un elenco di scommettitori
    - ▶ l'accettazione di scommesse da parte del bookmaker per conto di scommettitori (già conosciuti o nuovi)
    - ▶ la determinazione dell'importo totale delle vincite e delle perdite per il bookmaker al termine delle corse (quando cioè sono noti i vincitori di tutte le corse sulle quali sono state accettate scommesse)
-



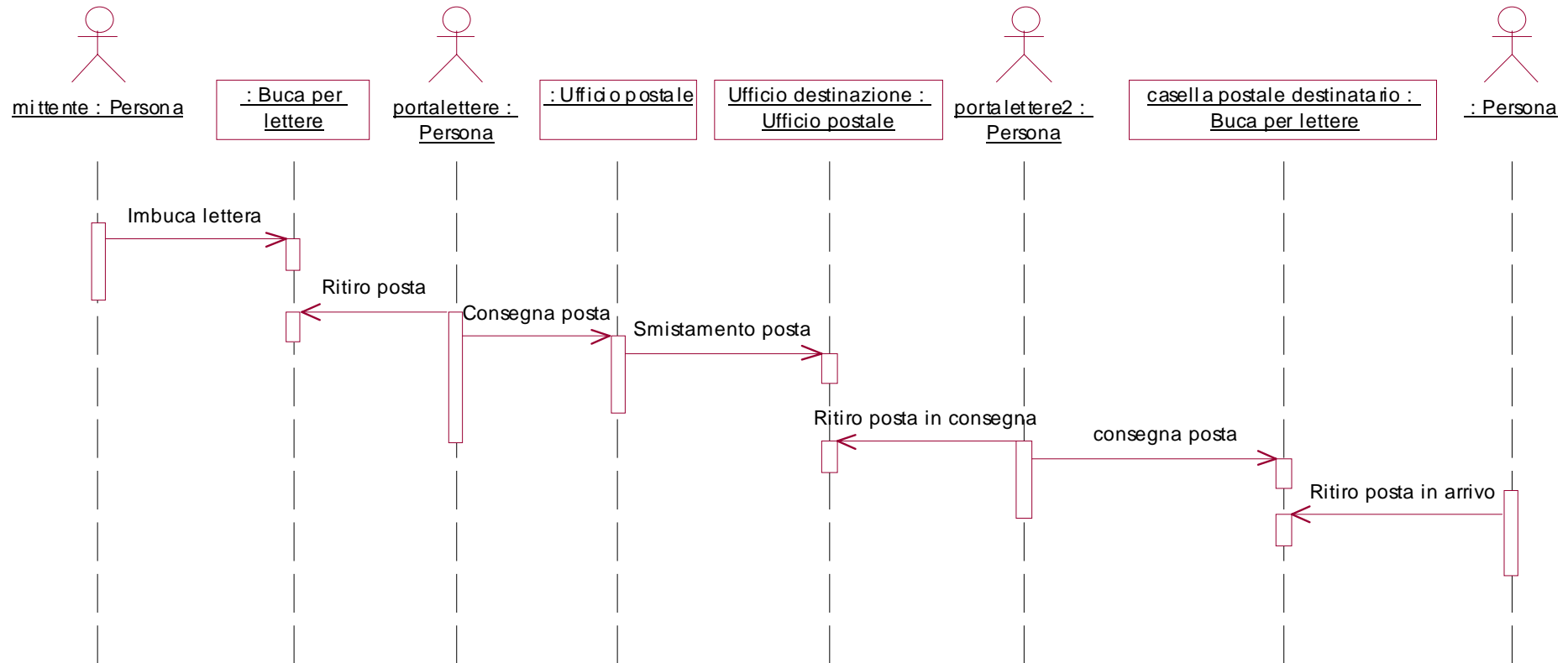
## Esercizio 7:

### Spedizione posta ordinaria

---

- Descrivere con un sequence diagram le operazioni che caratterizzano l'invio di una lettera per posta ordinaria dal momento della spedizione fino a quello della ricezione da parte del destinatario.
  - Introdurre gli attori, le classi e gli oggetti ritenuti utili per una descrizione sufficientemente completa del problema.
    - ▶ Eventualmente descrivere diversi scenari alternativi.
-

# Spedizione posta

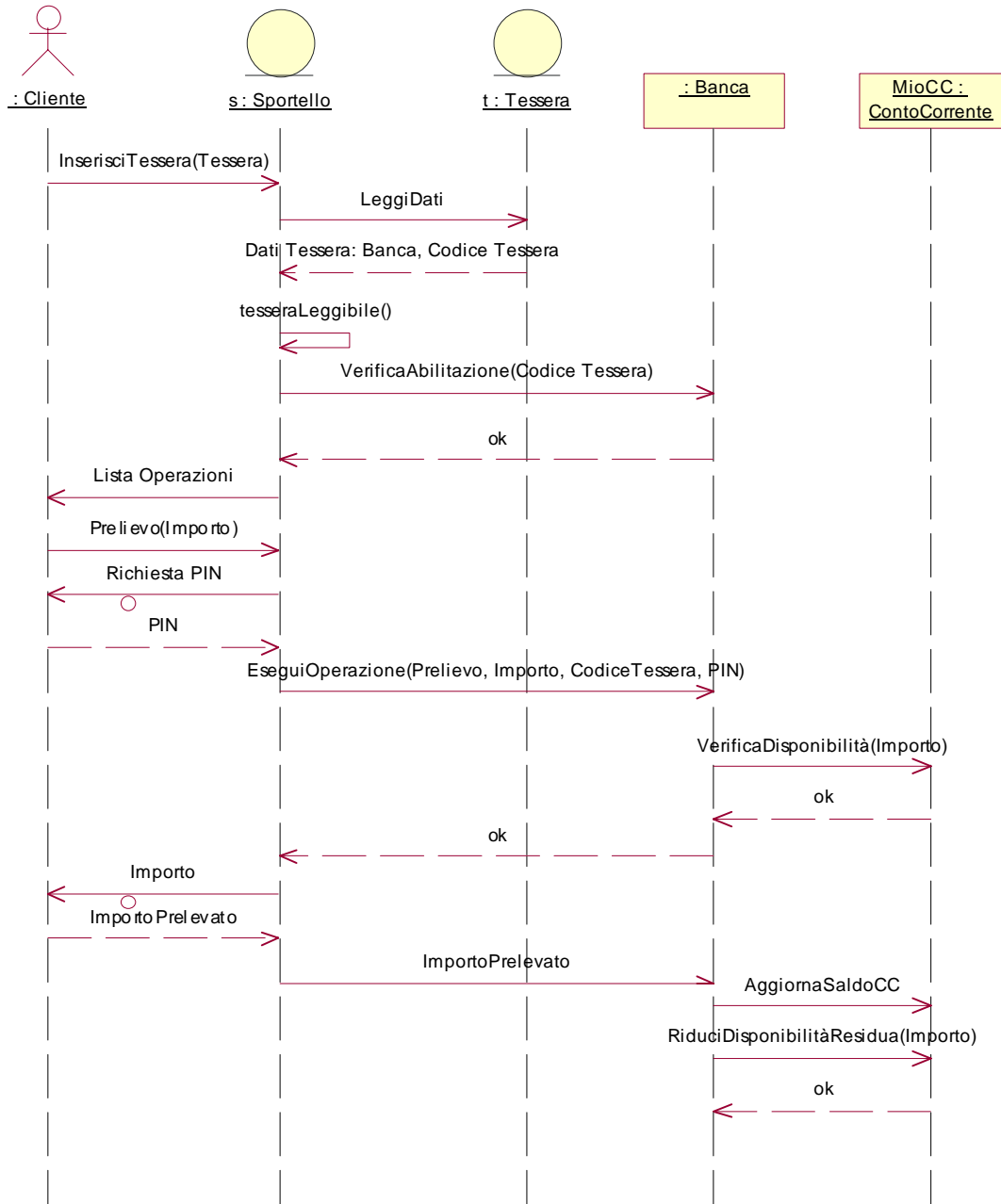




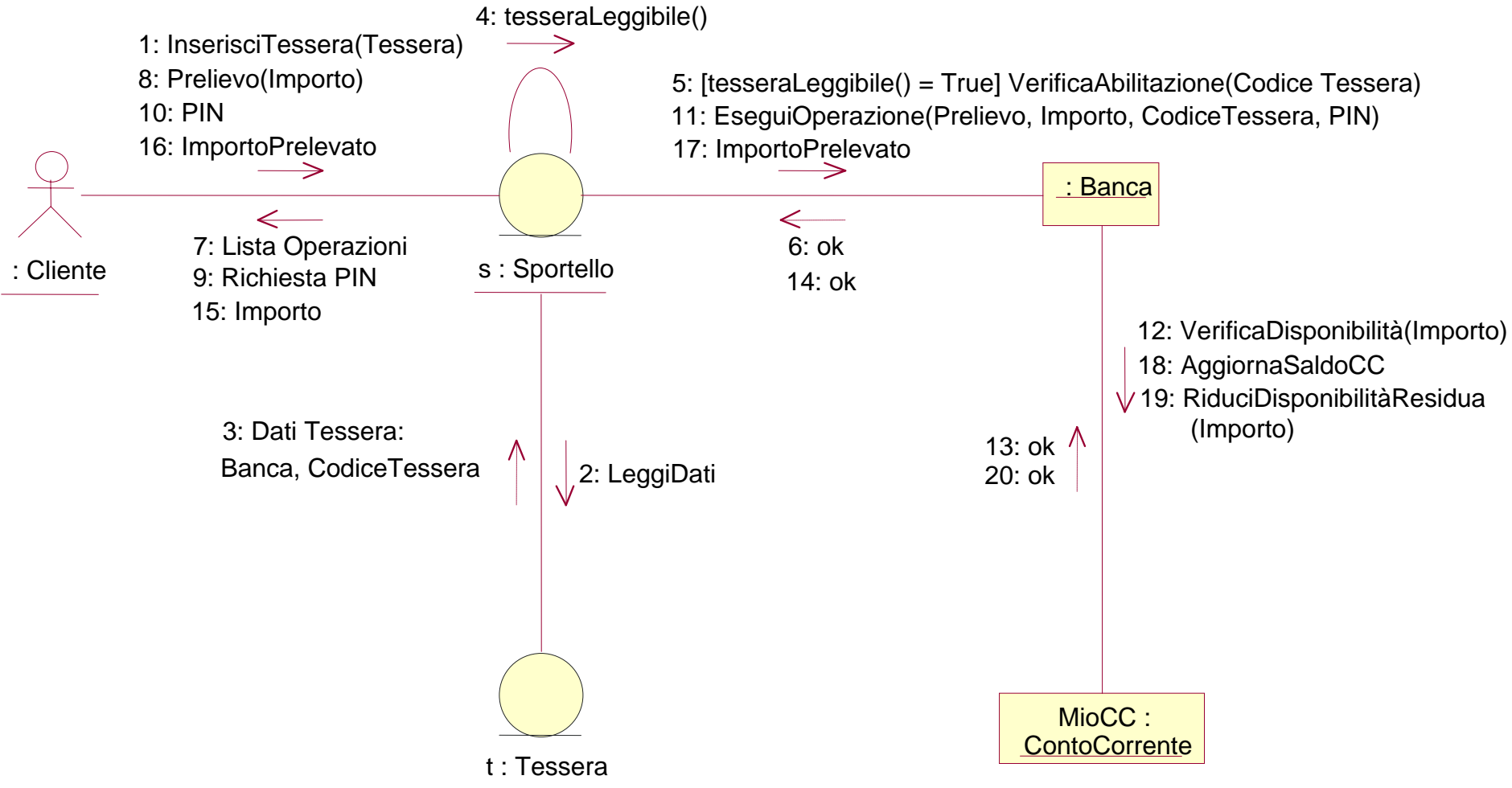
## Esercizio 8: Prelievo Bancomat

---

- Si definisca un semplice Sequence Diagram per modellare il prelievo di denaro da uno sportello Bancomat



# Collaboration diagram equivalente



## Esercizio 9:

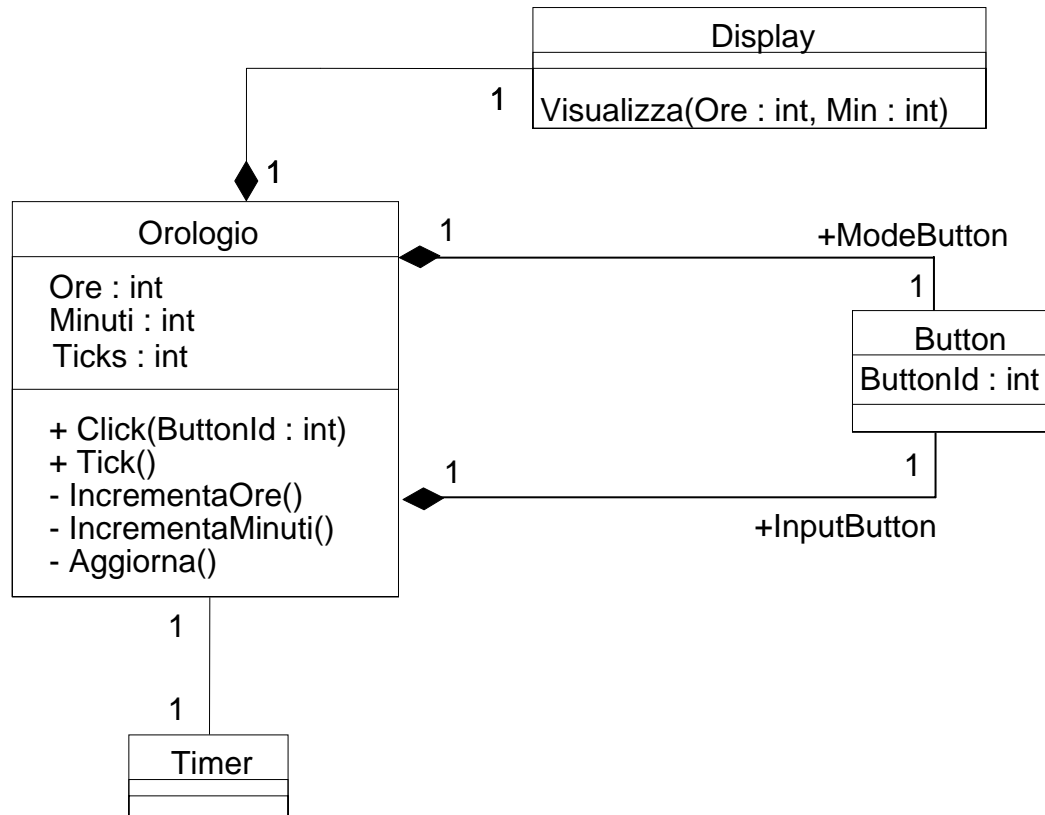
# Orologio digitale

---

- Descrivere con un diagramma a stati il comportamento di un orologio digitale dotato di display per la visualizzazione e di due bottoni per l'impostazione dell'orario.
    - ▶ Il primo bottone definisce la modalità di visualizzazione (normale, impostazione ora, impostazione minuti), mentre il secondo viene utilizzato in modalità impostazione per incrementare l'ora o i minuti correnti.
  - Dalla modalità di visualizzazione normale (nella quale viene visualizzato l'orario corrente)
    - ▶ premendo una prima volta il bottone che seleziona la modalità di visualizzazione si passa alla modalità di impostazione dell'ora
    - ▶ premendo una seconda volta si passa all'impostazione dei minuti
    - ▶ premendolo una terza volta si torna alla modalità di visualizzazione normale.
-

# Orologio digitale: class diagram

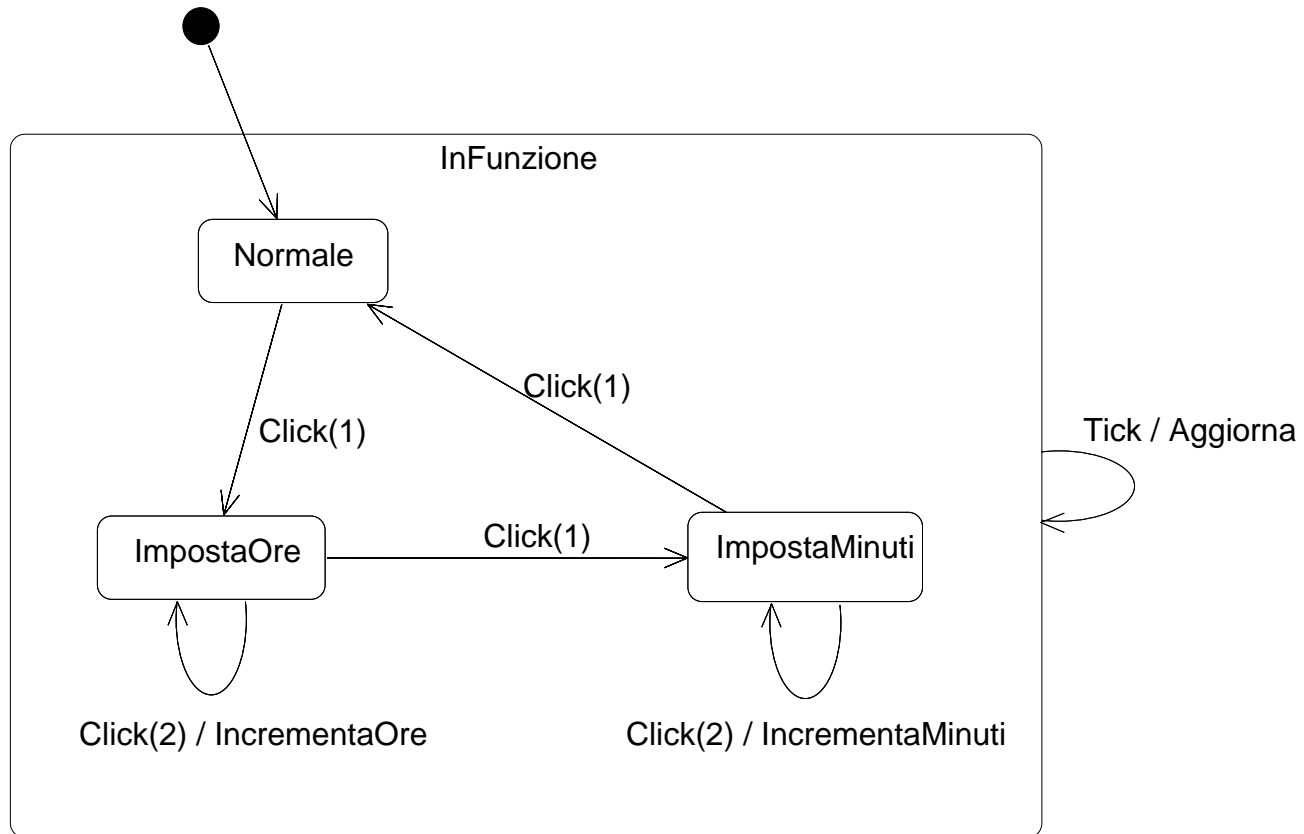
---



# Orologio digitale

## classe orologio: state diagram

---

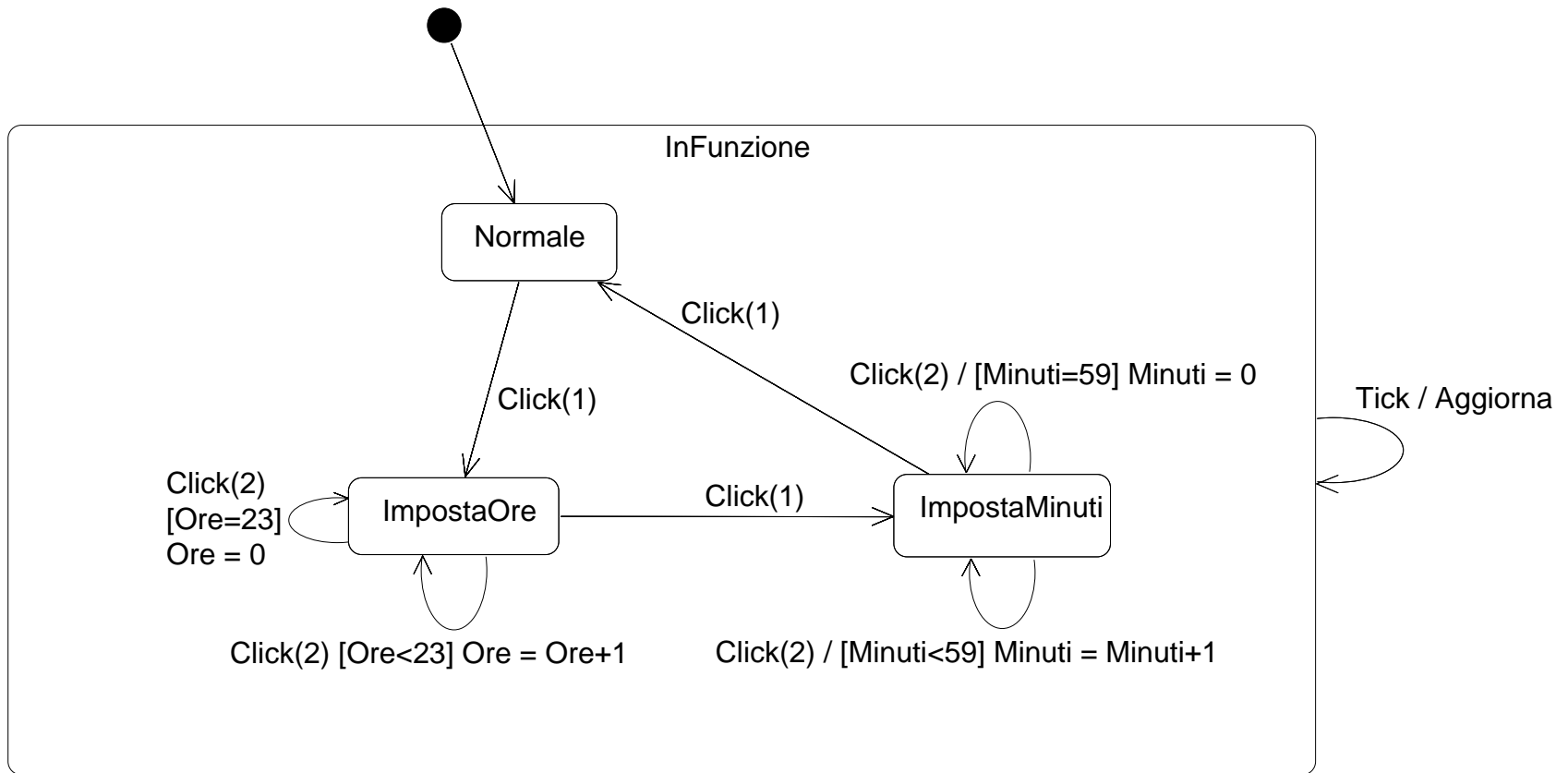


- Nota: in genere è preferibile specificare il significato delle operazioni mediante OCL.
  - Comunque entro certi limiti si può fare anche negli state diagram.
-

# Orologio digitale

## classe orologio: state diagram

---



---

Caso di Studio:  
Sistema di controllo per un ascensore

---



# Contesto

---

- Sistema di controllo degli ascensori in un palazzo di  $m$  piani
  - Il sistema di controllo deve gestire il movimento degli ascensori tra i diversi piani in accordo con i seguenti vincoli:
    - ▶ Ogni ascensore ha un insieme di bottoni corrispondenti ai diversi piani. Questi bottoni si illuminano quando vengono premuti indicando la richiesta di arresto dell'ascensore al raggiungimento del piano corrispondente
-

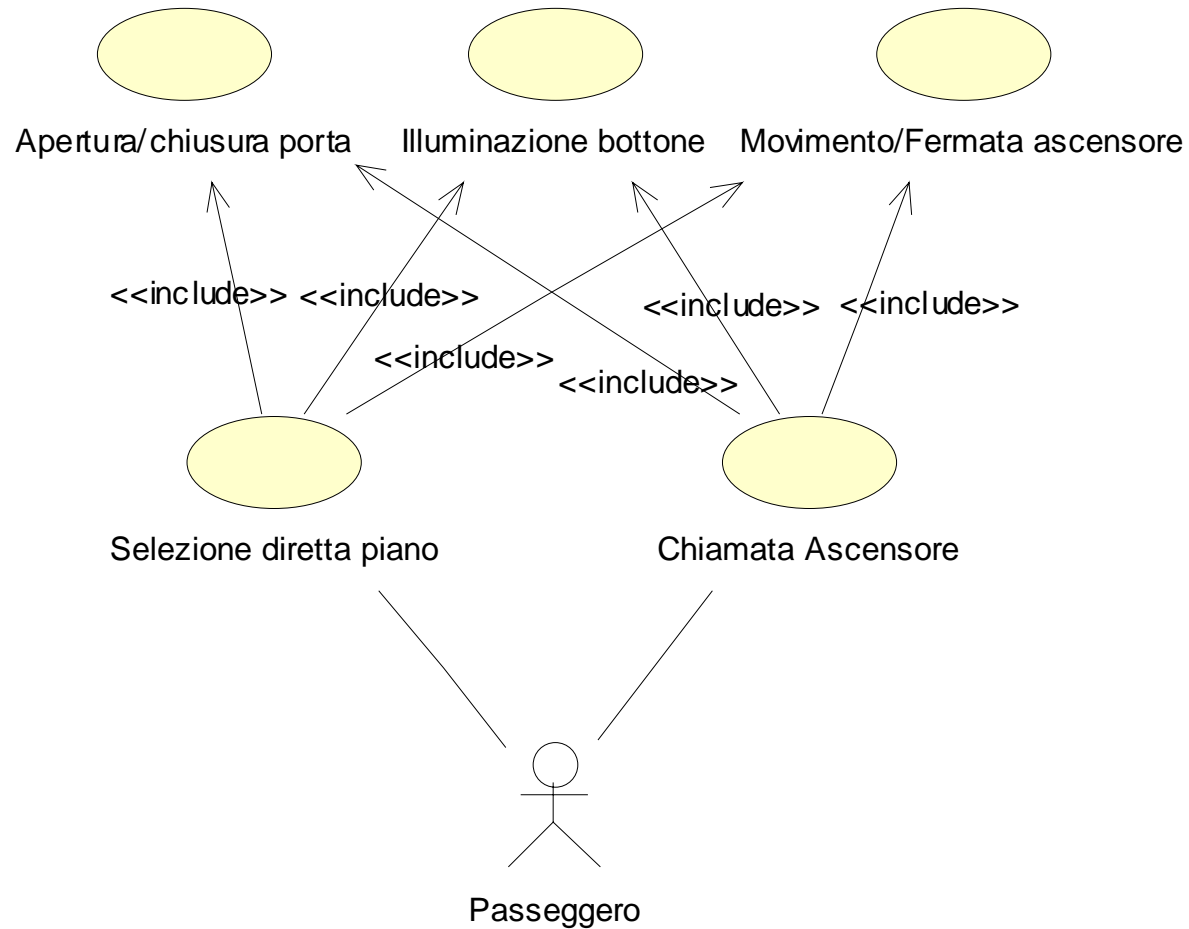
# Contesto

---

- Vincoli (segue):
    - ▶ In ogni piano, ad eccezione del primo e dell'ultimo, esistono due bottoni per chiamare l'ascensore specificando se si vuole salire o scendere. I bottoni si illuminano quando vengono premuti. L'illuminazione si spegne quando l'ascensore arriva al piano muovendosi nella direzione corrispondente al bottone illuminato
-

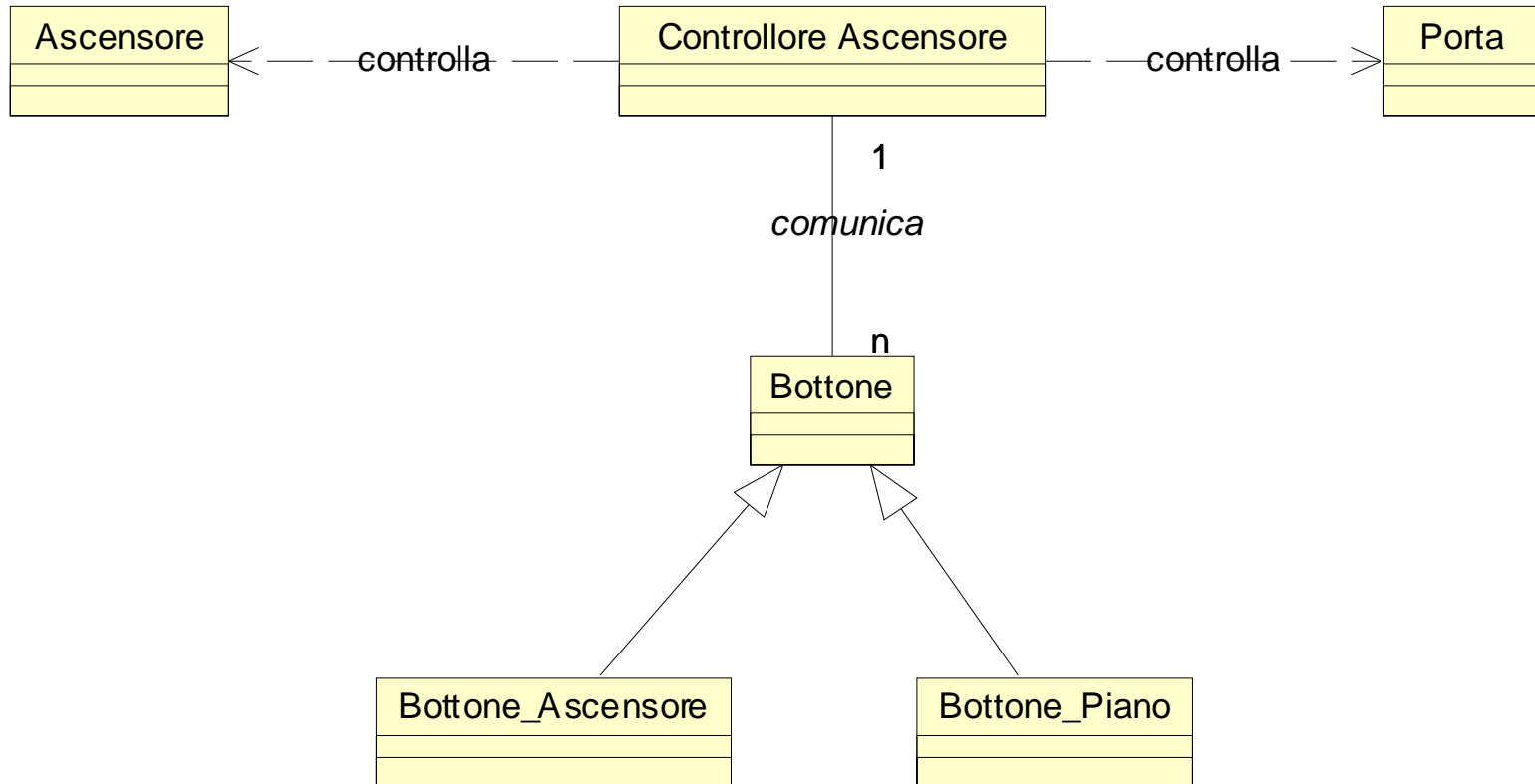
# Identificazione dei requisiti: use case

---

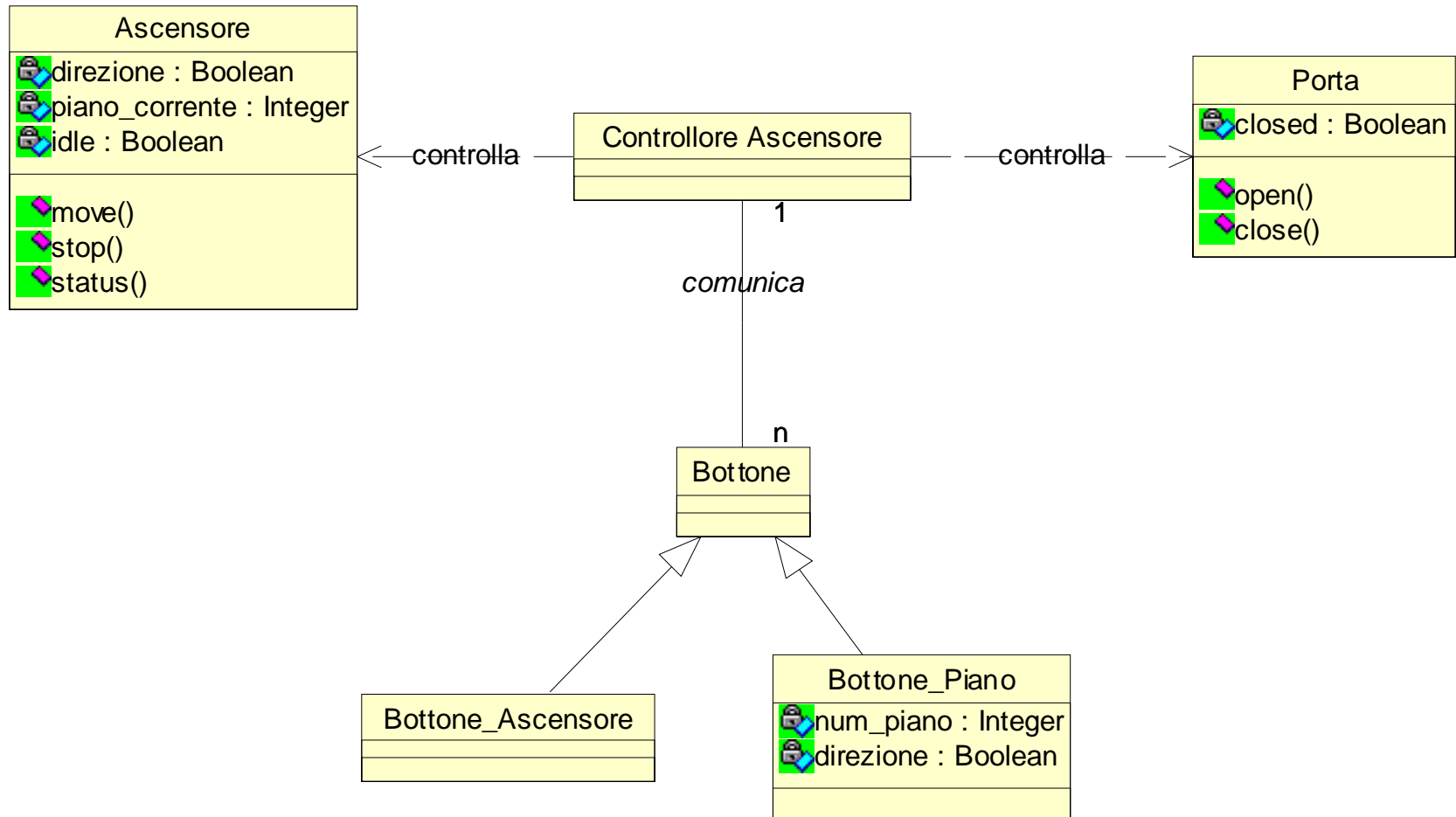


# Diagramma delle classi iniziale

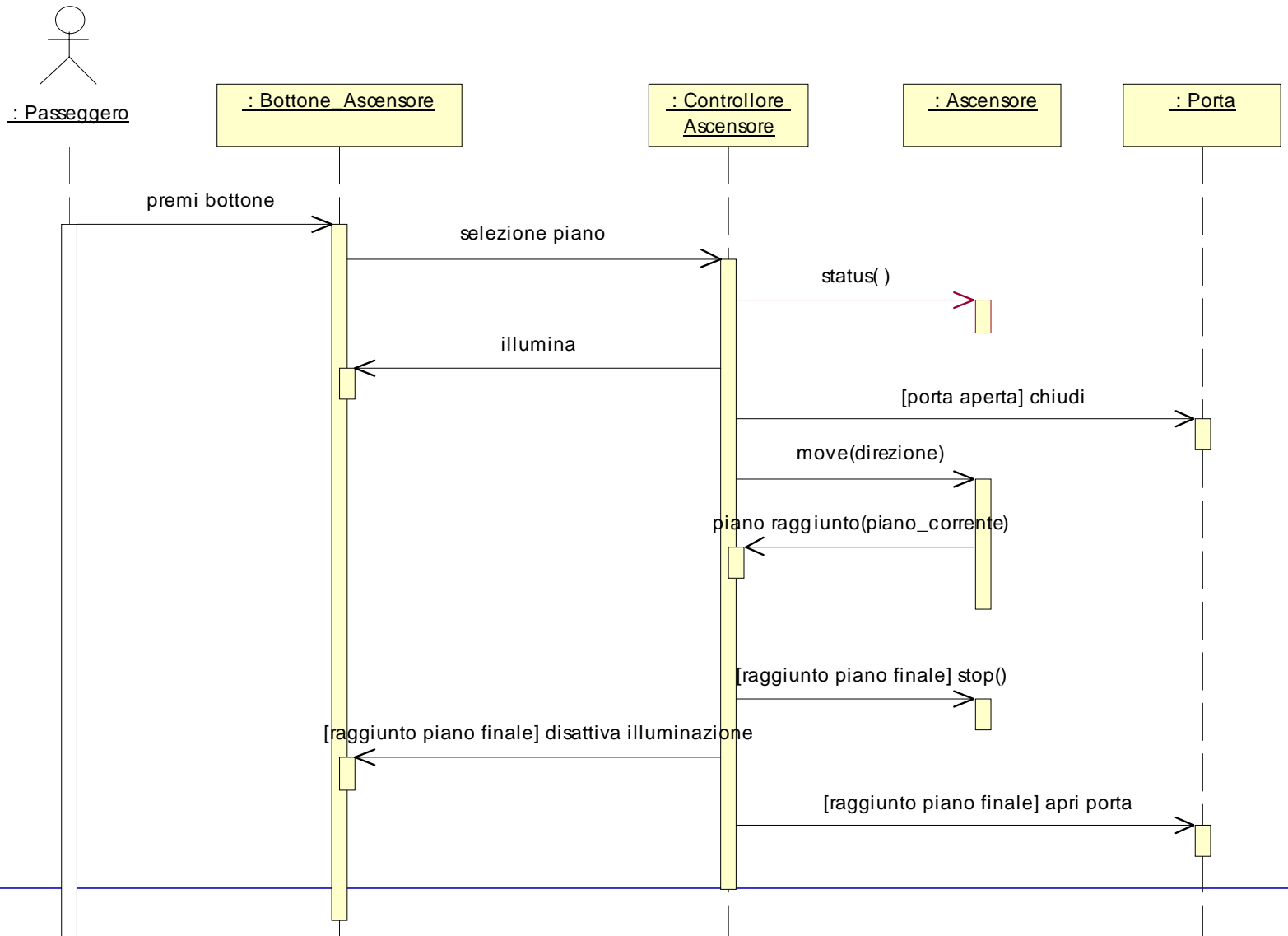
---



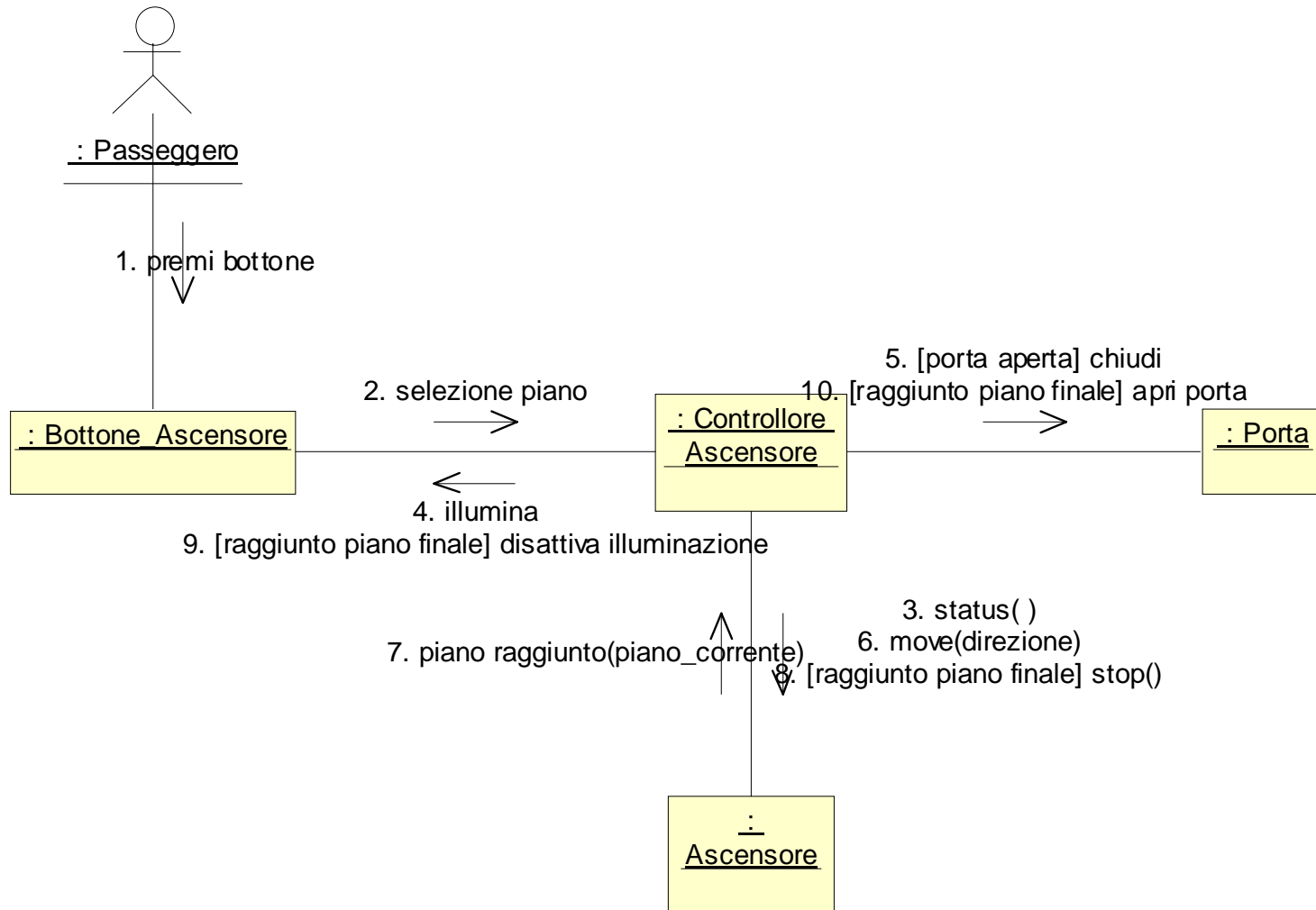
# Diagramma delle classi con attributi



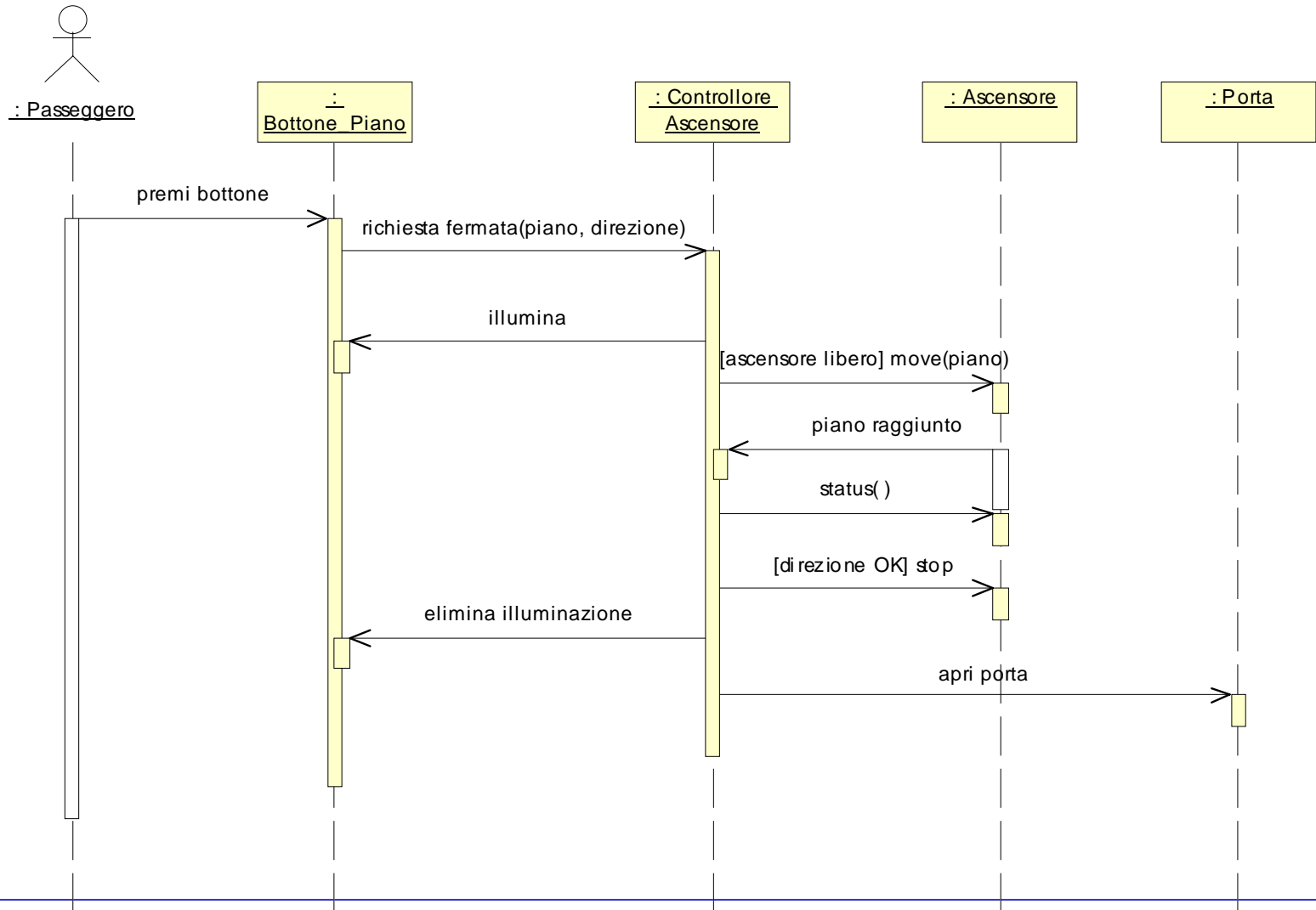
# Sequence diagram: Selezione diretta piano



# Collaboration diagram: Selezione diretta piano



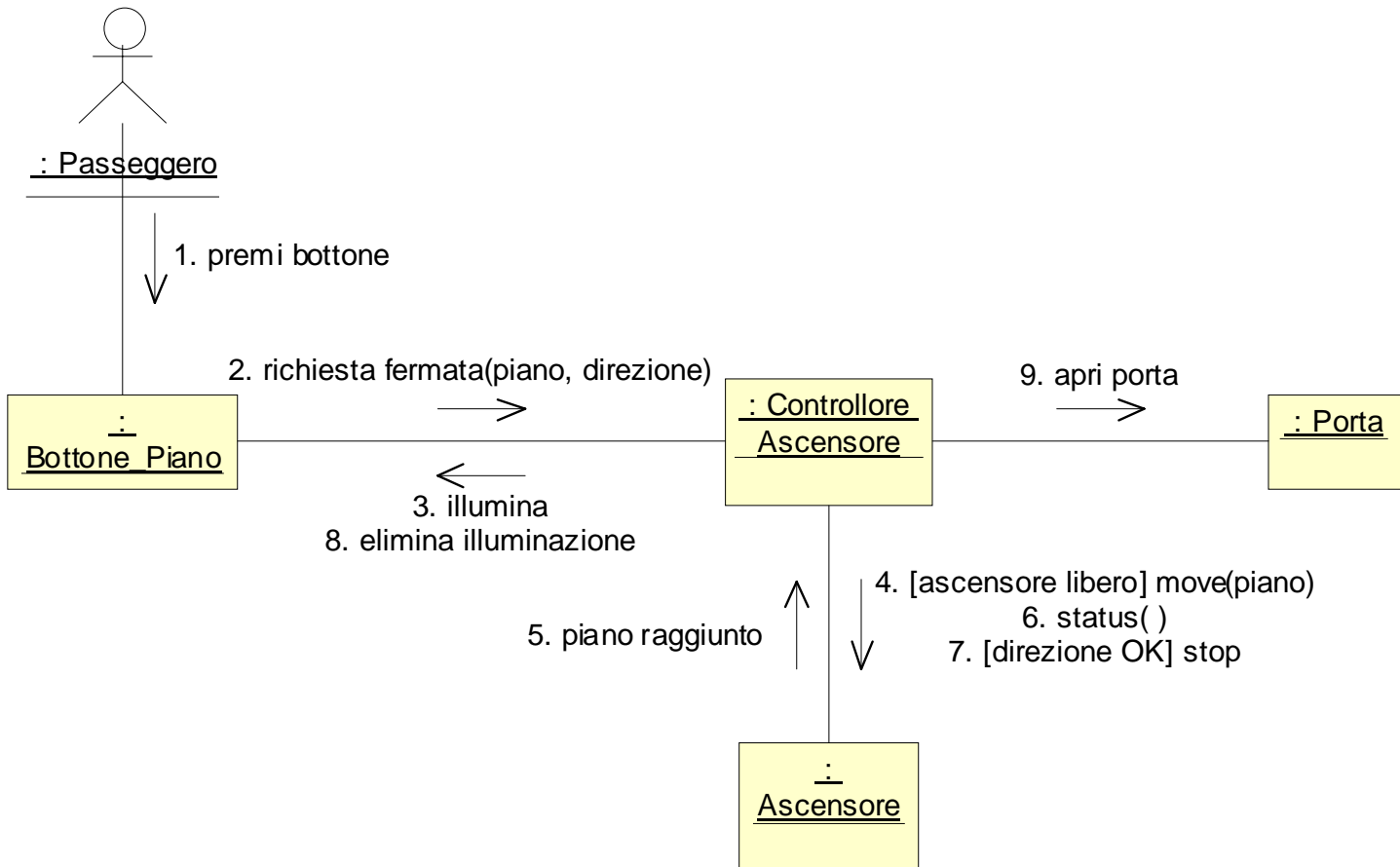
# Sequence diagram: chiamata ascensore





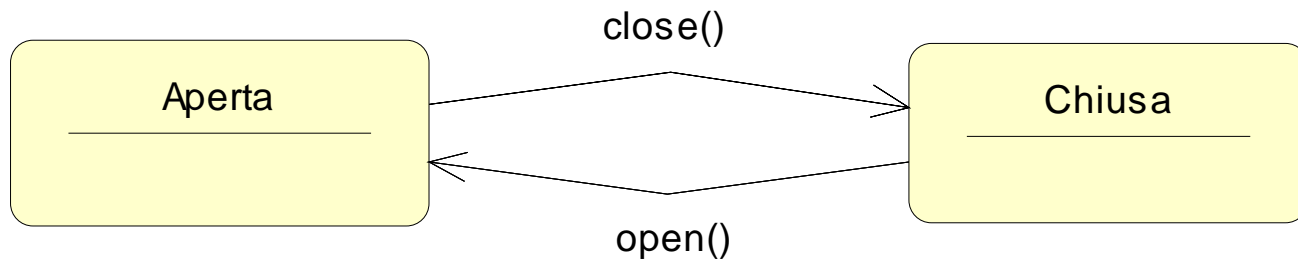
# Collaboration diagram: chiamata ascensore

---



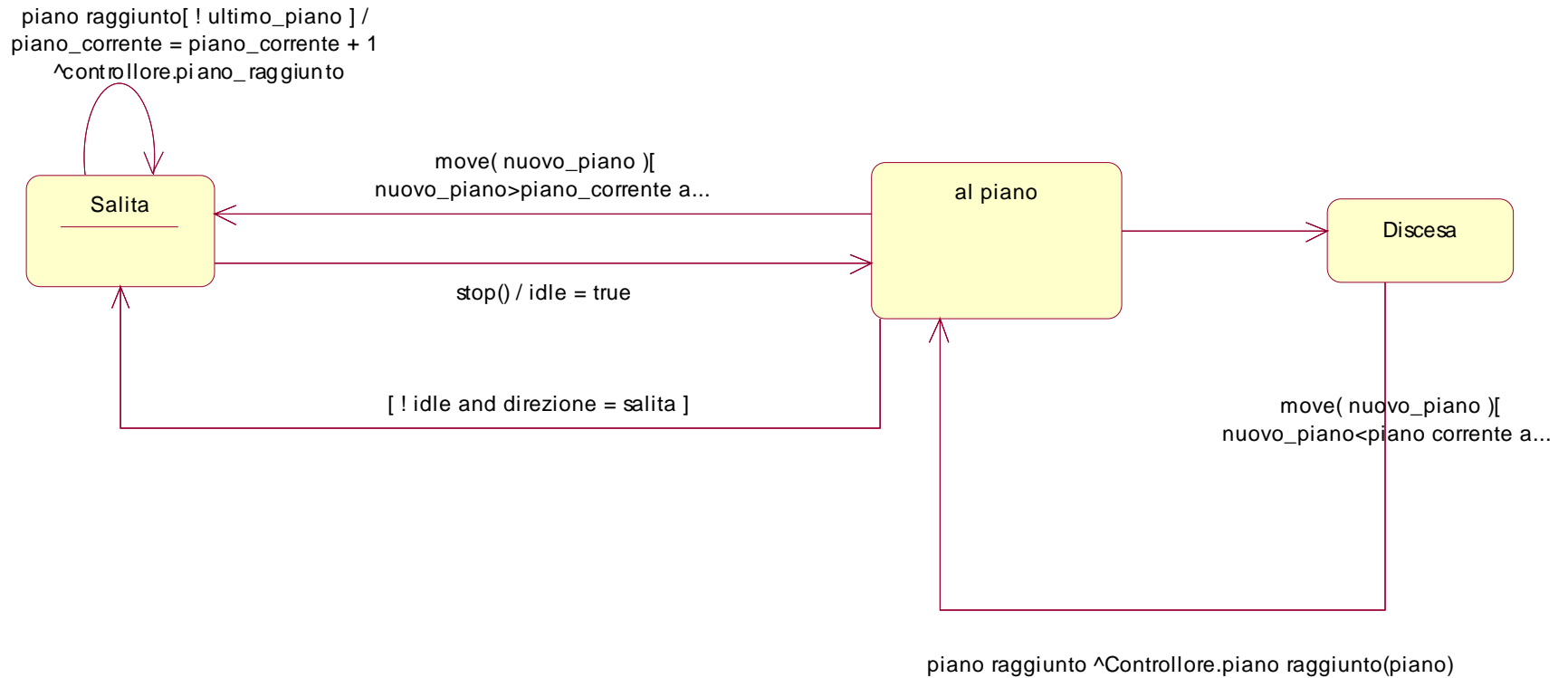
# Porta: diagramma degli stati

---



# Ascensore: esempio di diagramma degli stati

---



---

**Caso di Studio:**  
**Sistema di gestione posta elettronica**

---

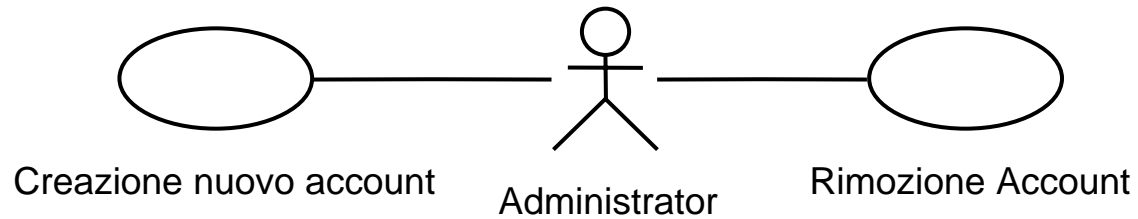
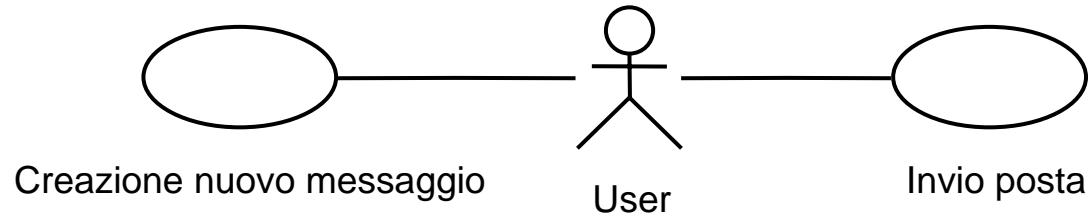
# Contesto

---

- In un sistema di posta elettronica ogni utente è caratterizzato dal suo username.
  - Per ogni utente esistono tre mailbox (inbox, outbox, draft) che, per semplicità, consideriamo insieme non ordinati di messaggi.
  - Si specifichino le operazioni di creazione di una mail e di sua spedizione:
    - ▶ La prima operazione crea una nuova mail nella outbox dell'utente.
    - ▶ La seconda operazione preleva i messaggi in outbox e li deposita nelle inbox dei destinatari.
  - Precisazione:
    - ▶ Solo indicazioni di massima, si trascurano molti dettagli
-

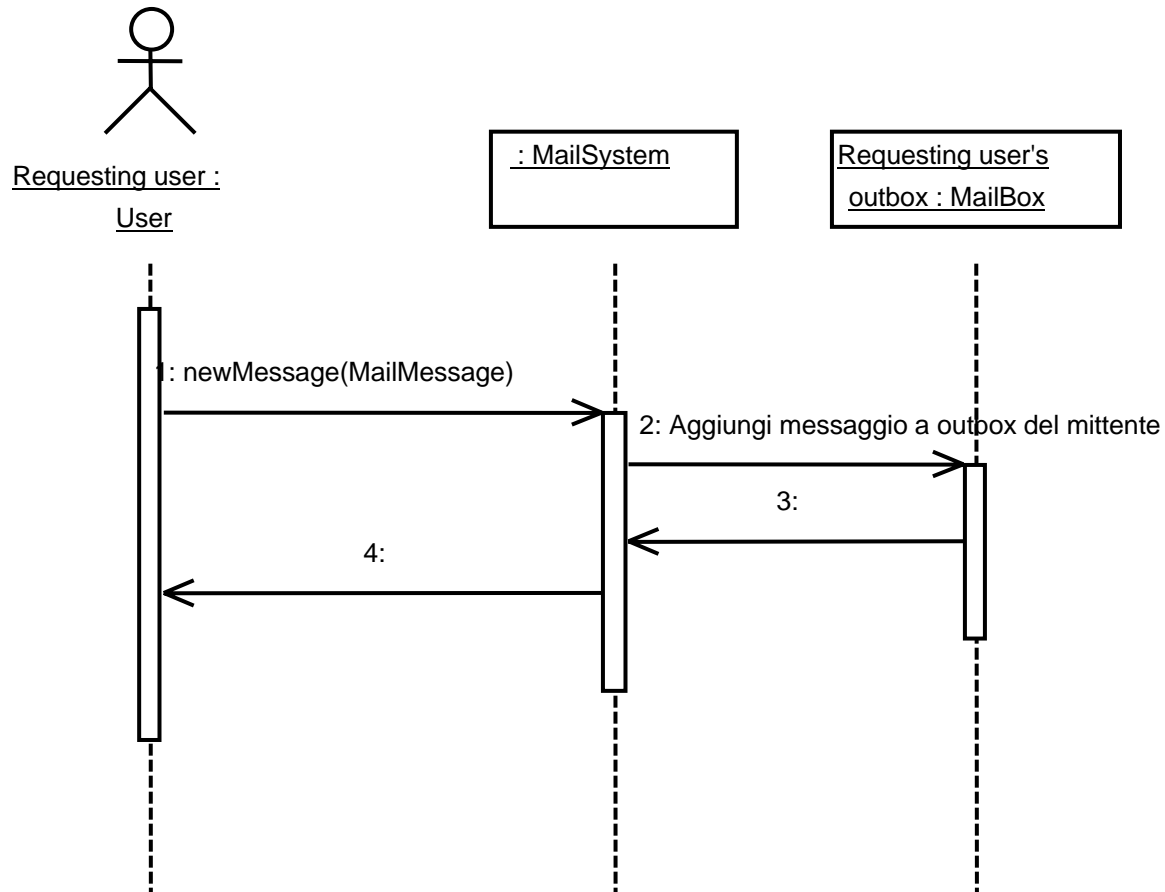
# Use cases

---

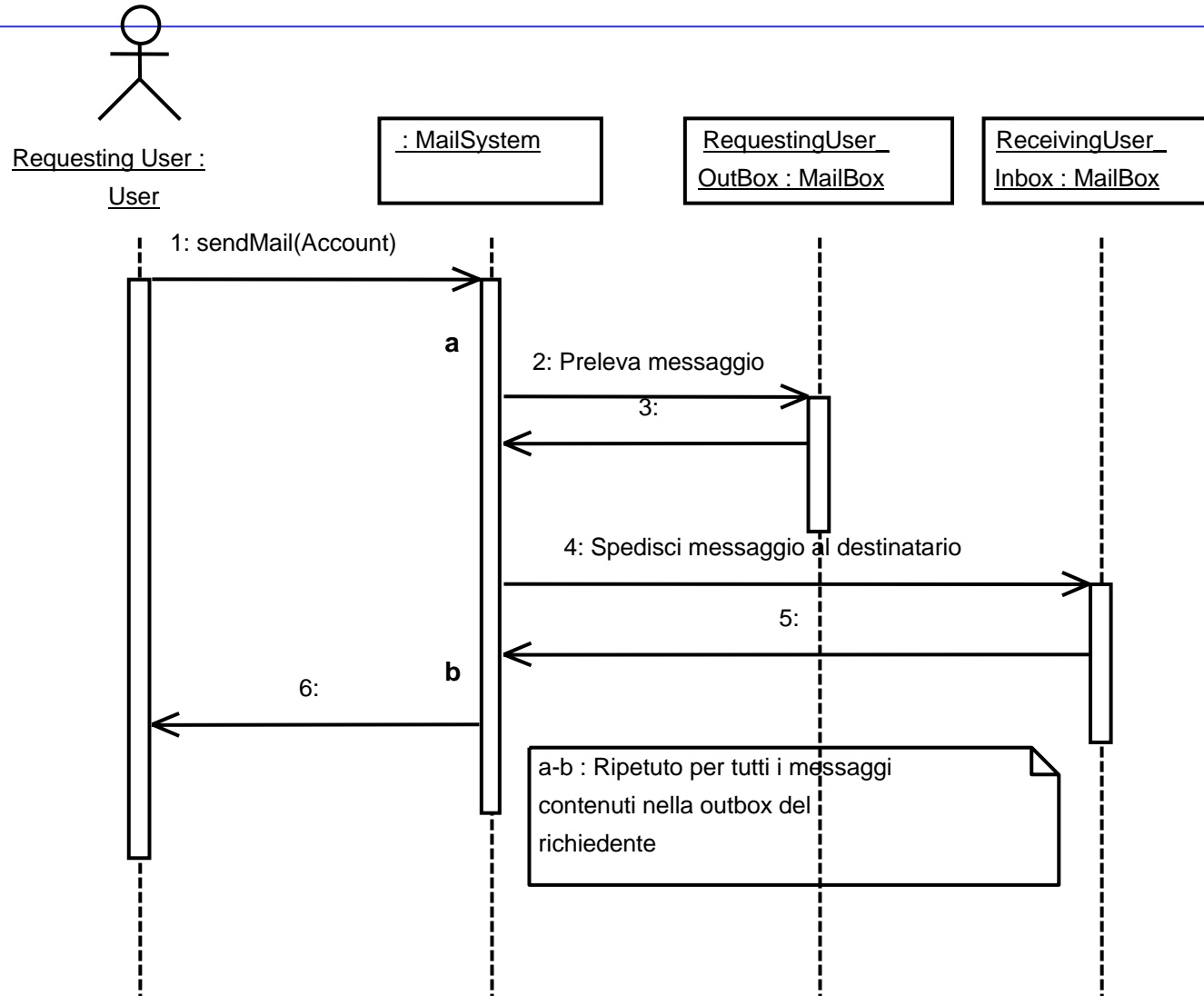


# Creazione nuovo messaggio: Scenario base

---

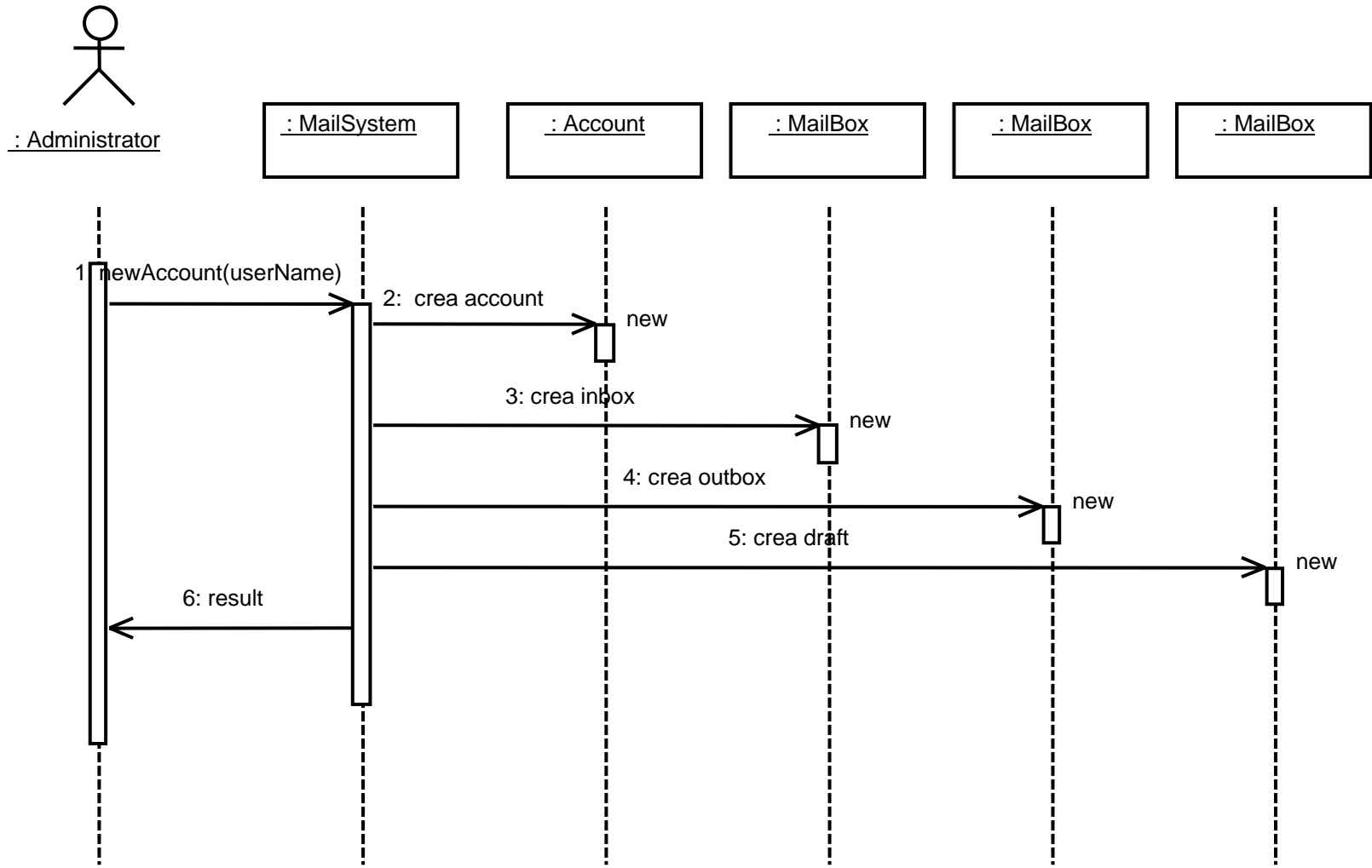


# Invio posta: Scenario base

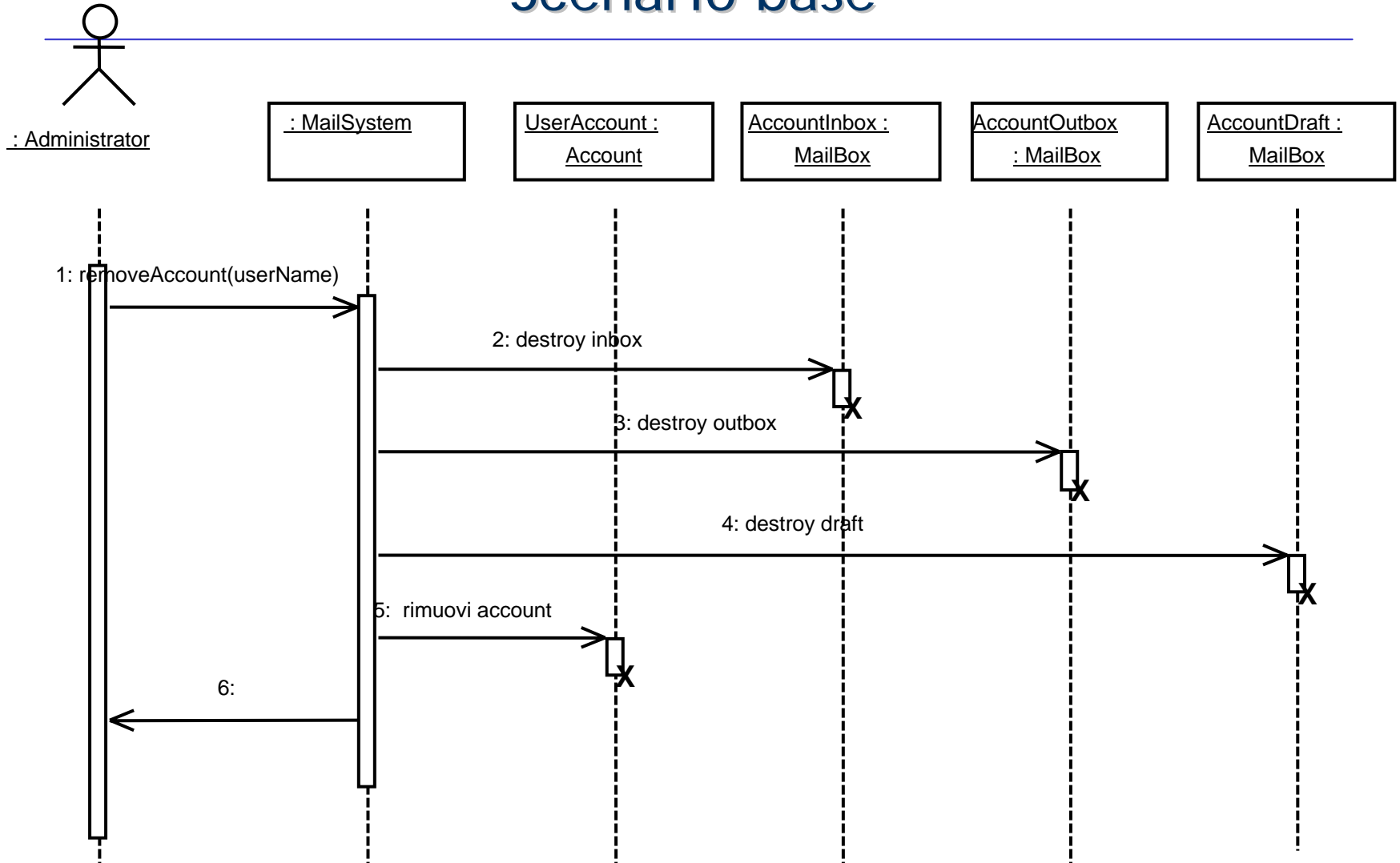




# Creazione nuovo account: scenario base



# Rimozione account: Scenario base

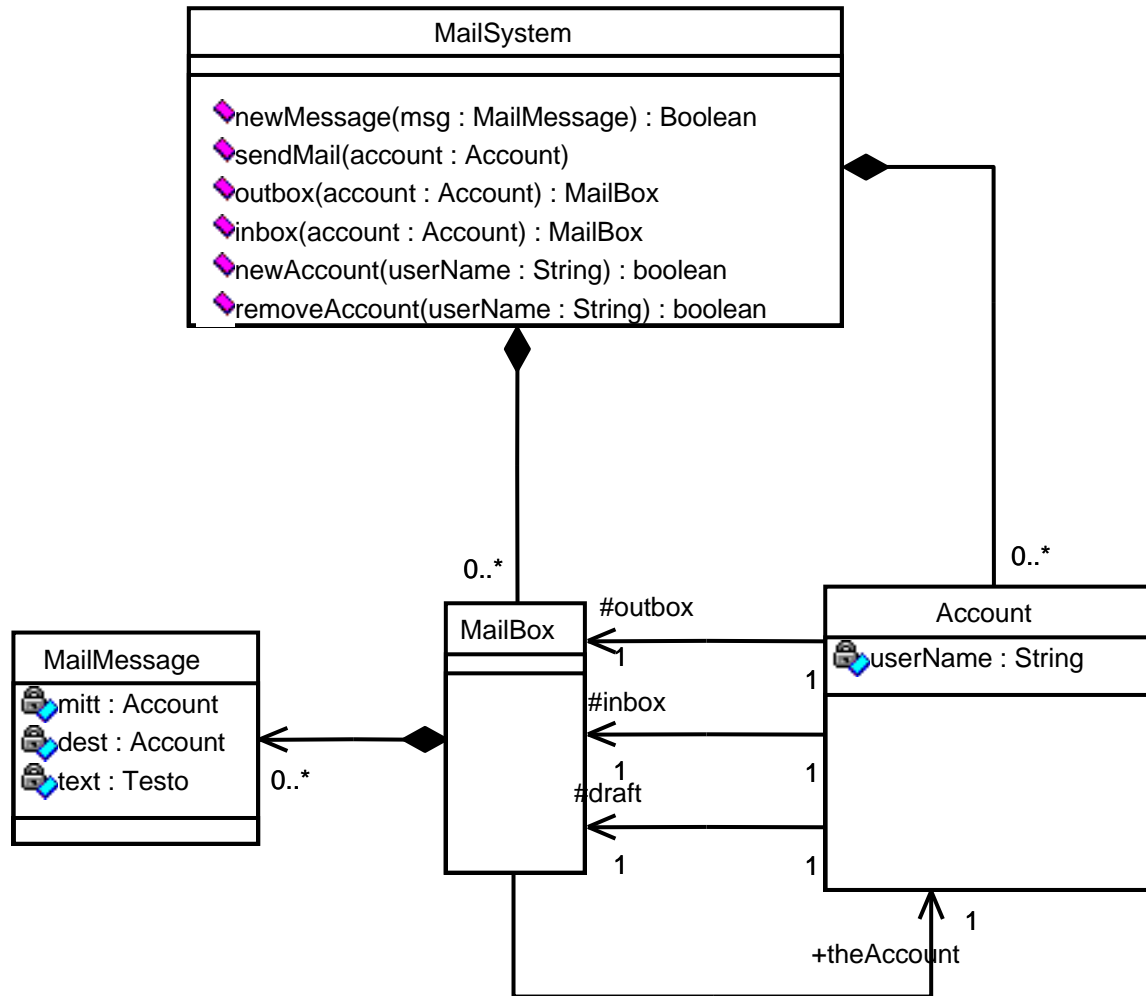


# Diagramma delle classi

---

- Struttura statica del sistema
    - ▶ Classi
    - ▶ Proprietà delle classi
    - ▶ Relazioni fra classi
  - Costruzione
    - ▶ Identificazione delle entità rilevanti (classi)
    - ▶ Identificazione delle relazioni fra classi
    - ▶ Identificazione degli attributi
    - ▶ Identificazione delle relazioni di generalizzazione/specializzazione
    - ▶ Eventuale suddivisione in package
-

# Diagramma delle classi



# Diagramma delle classi (Alternativo)

