



UNIVERSITÀ  
DEGLI STUDI DI TRIESTE



## Corso di Laurea in Tecniche di Radiologia Medica per immagini e Radioterapia Informatica Medica

2CFU – 20 ore

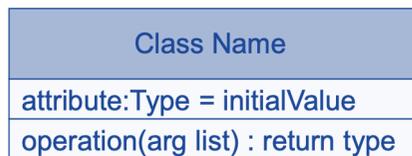
### CENNI DI UNIFIED MODELING LANGUAGE

*Prof. Sara Renata Francesca Marceglio*

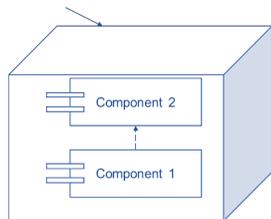
# CHE COS'È UML

“The Unified Modeling Language (UML) is a **graphical** language for **specifying, visualizing, constructing, and documenting** the artifacts of software systems, as well as for business modeling and other non-software systems”.

## STATIC VIEWS



Class Diagram



Component Diagram

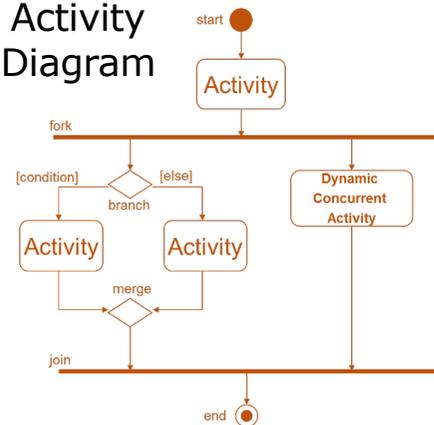
Use Case

Use-case Diagram

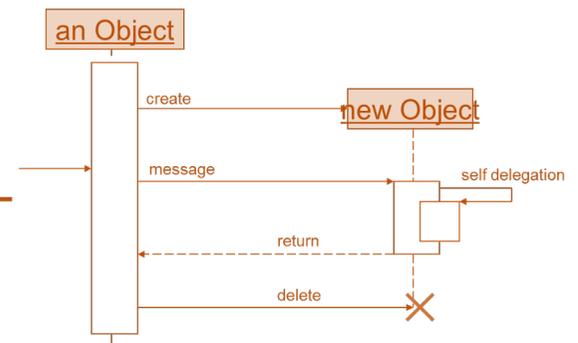


## DYNAMIC VIEWS

Activity Diagram



Sequence Diagram



# OBIETTIVI DI UML

- UML è un linguaggio e non un metodo
- UML è costituito da un insieme di linguaggi che consentono di dare descrivere e modellare diversi aspetti di un sistema:
  - Aspetti statici
  - Aspetti dinamici
  - Interazioni tra le diverse componenti
- UML facilita la **comunicazione**→
  - tra analisti e esperti di dominio
  - Tra analisti e progettisti
  - Tra progettisti e sviluppatori
- Linguaggio grafico ma formale, preciso e non ambiguo
- Dà una visione concisa del sistema da diversi punti di vista
- Non necessita di conoscenza del codice
- UML è basato sulla logica della **programmazione a oggetti**

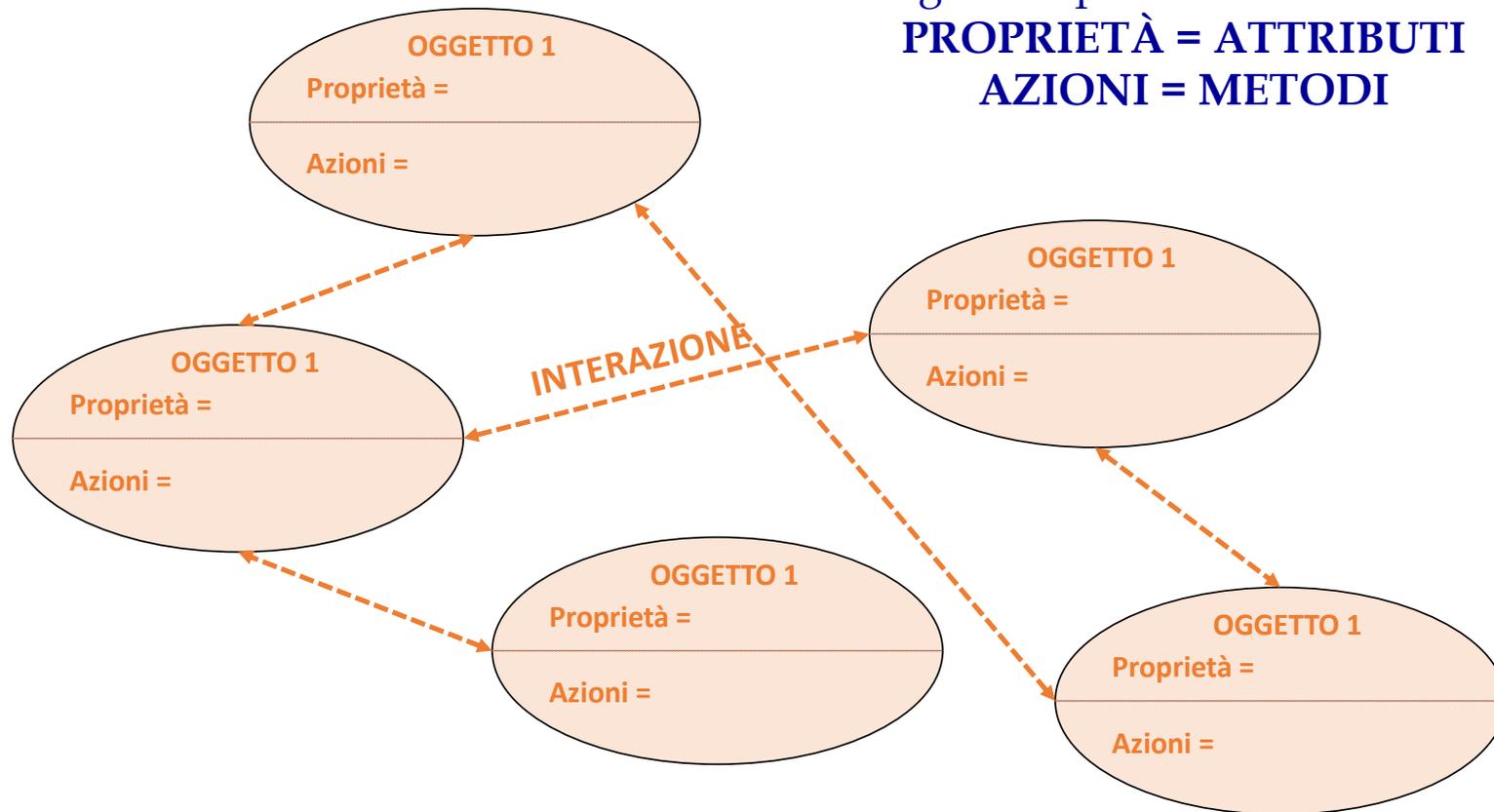
## PERCHÈ SI USA UML

- Si deve operare con esperti di dominio
- Si devono catturare gli aspetti più importanti senza perdersi nei dettagli
- La notazione di UML è flessibile → molti diagrammi rappresentano aspetti diversi dei concetti
- La riunione di tutti i diagrammi prodotti può essere facilmente utilizzata come rappresentazione del sistema intero → [scheletro del sistema](#)

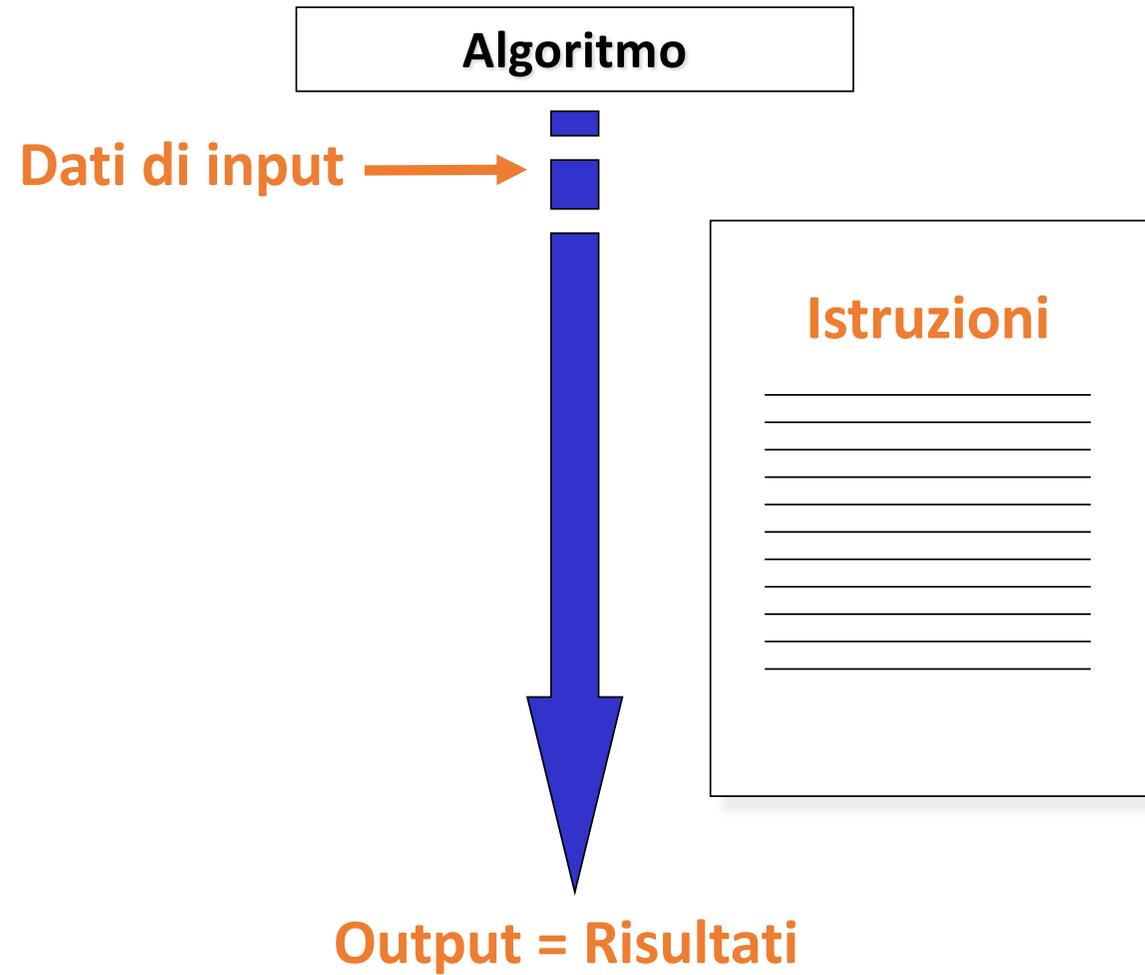
# PROGRAMMAZIONE A OGGETTI

## SISTEMA

Un sistema è un insieme di  
entità interagenti = **OGGETTI**  
in cui ogni componente è caratterizzato da  
**PROPRIETÀ = ATTRIBUTI**  
**AZIONI = METODI**

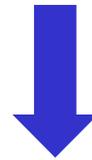


# PROGRAMMAZIONE ALGORITMICA

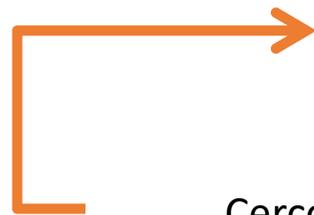


# PROGRAMMAZIONE ALGORITMICA (2)

**DATO UN VETTORE IN INGRESSO**  
[2 77 1 935 11 19 773 15 3]



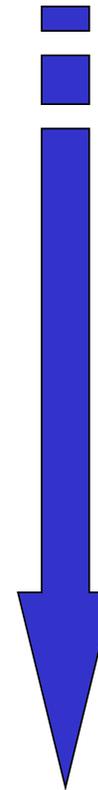
**VOGLIO IN USCITA IL VETTORE ORDINATO**



Cerco il minimo  
Lo metto da parte  
Lo elimino dal vettore  
Cerco il minimo nel vettore rimanente



**VETTORE IN USCITA**  
[1 2 3 11 15 19 77 773 935]



# ALGORITMI vs OGGETTI

## PROGRAMMAZIONE ALGORITMICA

- Sequenza di azioni
- Basato su DATI e FUNZIONI = PROGRAMMI
- Obiettivo: risolvere un PROBLEMA

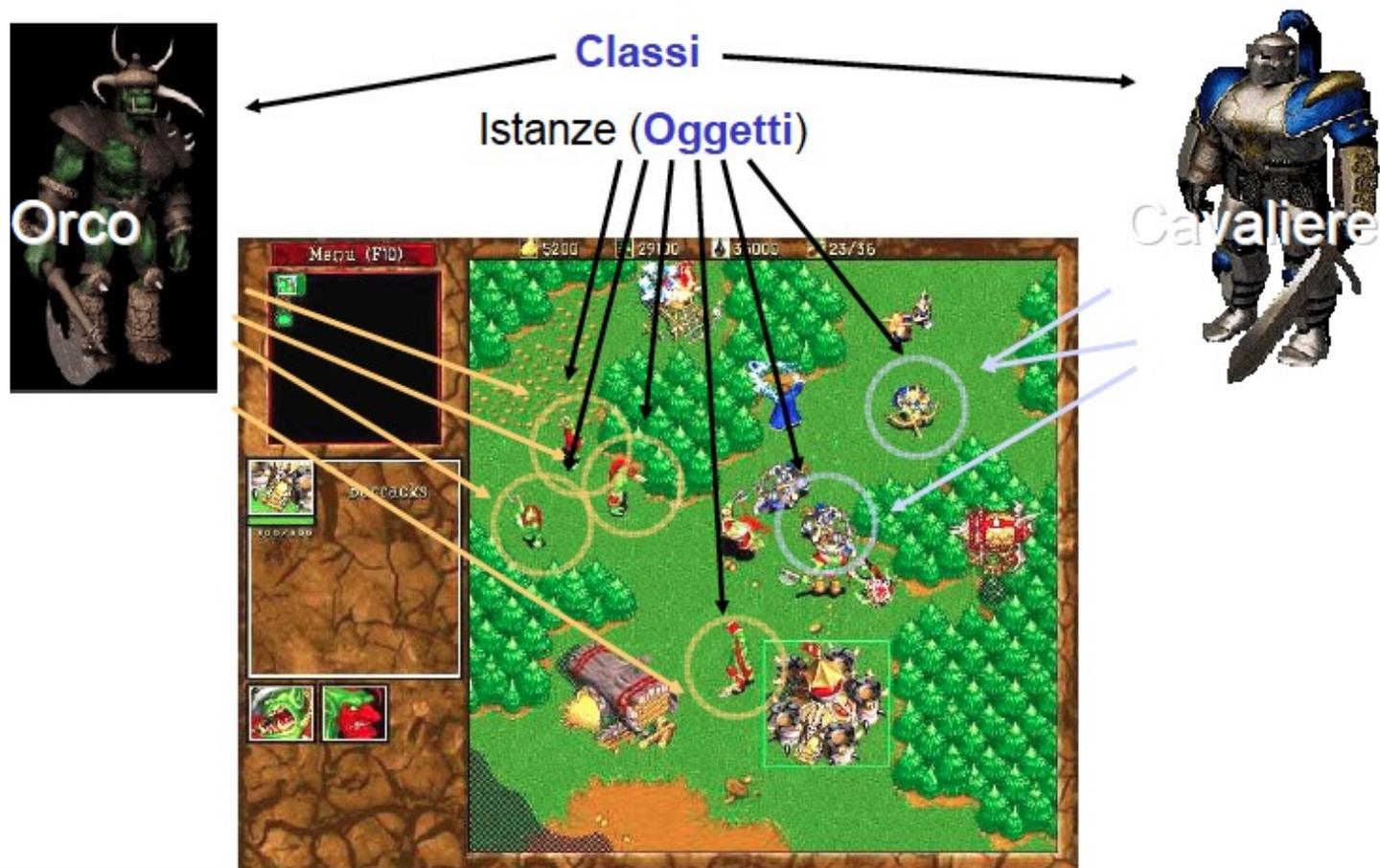
## PROGRAMMAZIONE A OGGETTI

- Sistema = Insieme di oggetti
- Basato su OGGETTI fatti da AZIONI e ATTRIBUTI
- Obiettivo: gestire un SISTEMA

# OGGETTI E CLASSI



# OGGETTI E CLASSI



# ESEMPIO

## CLASSE

ATTRIBUTI =  
Caratteristiche

METODI =  
Comportamenti

### FARMACO

Principio attivo  
Dosaggio  
Forma Farmaceutica

Prescrivi  
Cambia Dosaggio

**OGGETTO =  
ISTANZA DI UNA CLASSE**

## OGGETTI

### MOMENT

Principio Att = Ibuprofene  
Dosaggio = 200 mg  
Forma = Compresse

Prescrivi  
Cambia Dosaggio

### TACHIPIRINA

Principio Att = Paracetamolo  
Dosaggio = 1000 mg  
Forma = Compresse effervescenti

Prescrivi  
Cambia Dosaggio

## ESEMPIO: OSSERVAZIONI

### MOMENT

Principio Att = Ibuprofene  
Dosaggio = 200 mg  
Forma = Compresse

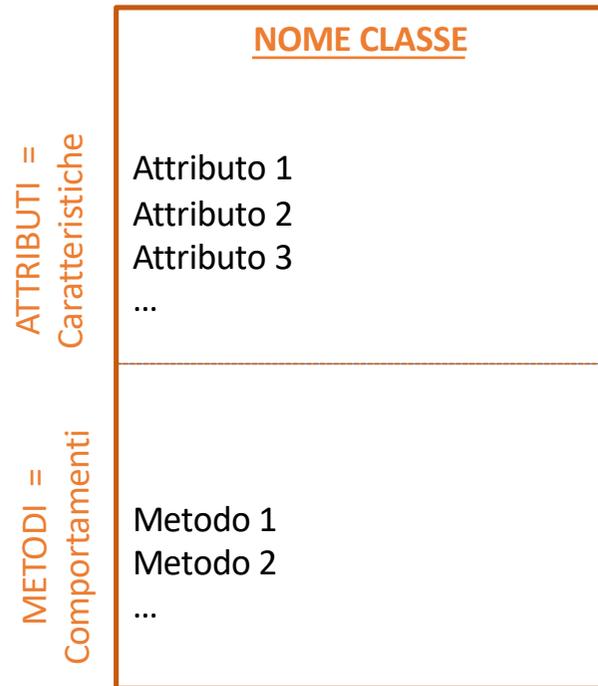
Prescrivi  
Cambia Dosaggio

I VALORI DEGLI ATTRIBUTI  
SONO SPECIFICI  
DELL'OGGETTO ISTANZIATO  
(ogni oggetto ha il suo insieme di  
valori)

LE AZIONI SONO COMUNI A  
TUTTE LE ISTANZE DELLA  
CLASSE

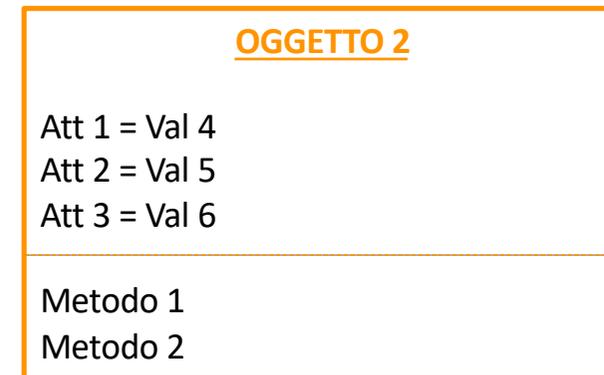
# CLASSI E OGGETTI: DEFINIZIONE

## CLASSE



**OGGETTO =  
ISTANZA DI UNA CLASSE**

## OGGETTI



# ATTRIBUTI E METODI

## ATTRIBUTI

- Descrivono le proprietà **statiche** dell'oggetto
- Nella programmazione gli attributi vengono realizzati attraverso l'uso delle **variabili** utilizzate dall'oggetto per memorizzare i dati

## METODI

- Descrivono le proprietà **dinamiche** dell'oggetto
- Nella programmazione i metodi vengono realizzati attraverso la scrittura di codice (**procedure e funzioni**) che implementano le operazioni dell'oggetto

# PROPRIETÀ DELLE CLASSI: EREDITARIETÀ

## *Classe genitrice*

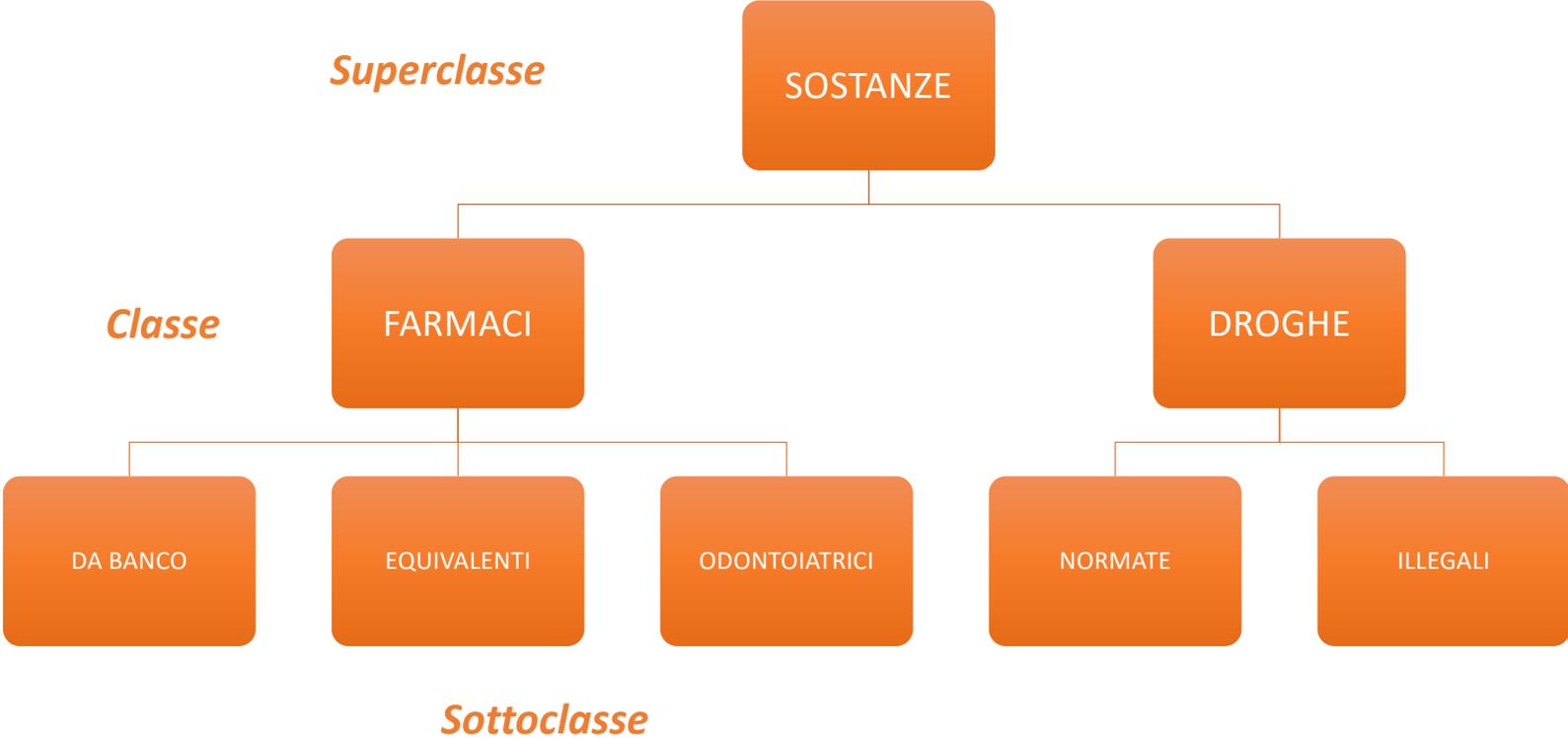
<i>farmaco</i>
Principio attivo
Dosaggio
Forma Farmaceutica
Costo SSN
Prescrivi
Cambia Dosaggio



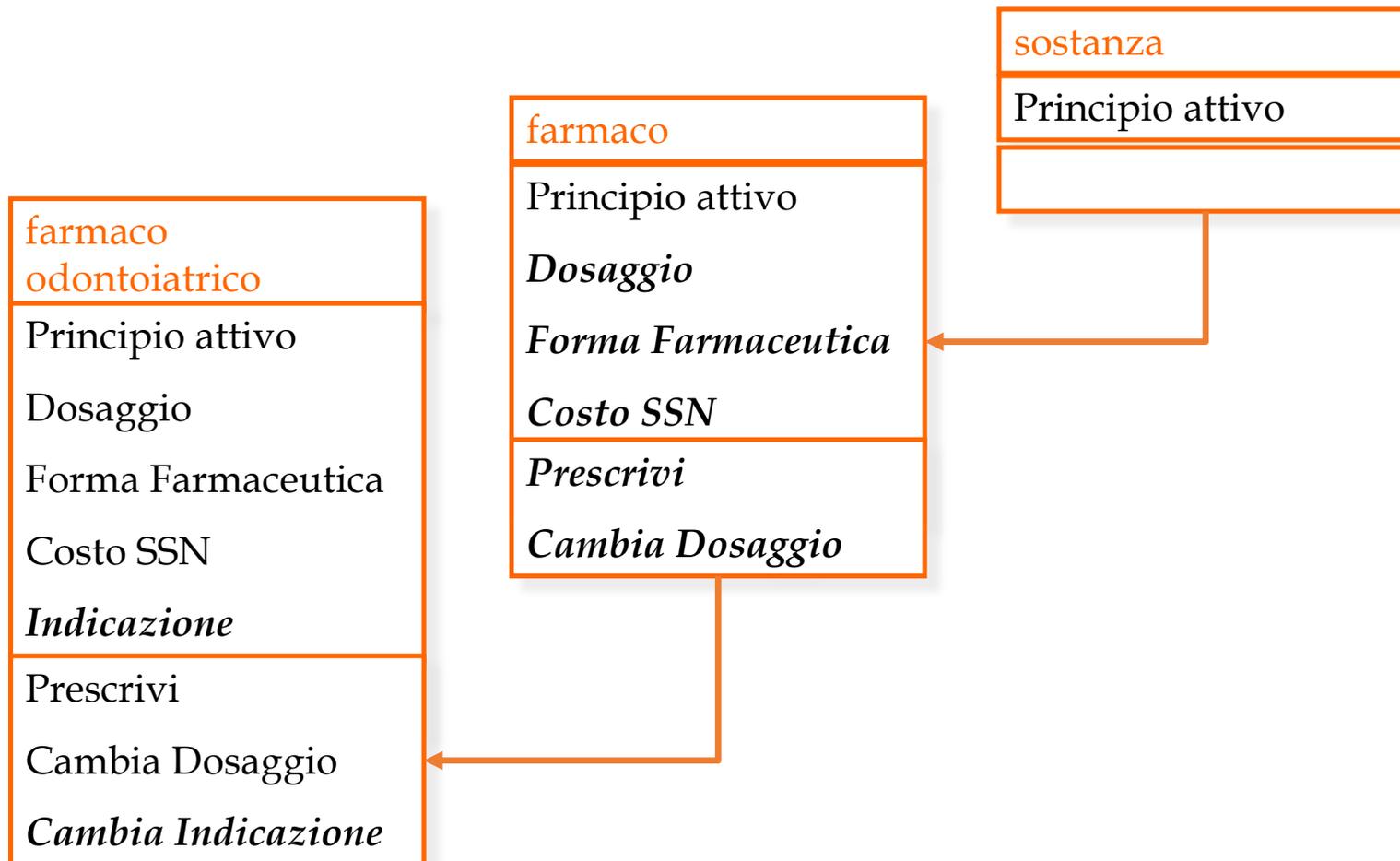
## *Nuova Classe*

<i>Farmaco di marca</i>
Principio attivo
Dosaggio
Forma Farmaceutica
<i>Nome Commerciale</i>
Costo SSN
<i>Prezzo Pubblico</i>
Prescrivi
Cambia Dosaggio
<i>Calcola costo paziente</i>

# GERARCHIE DI CLASSI

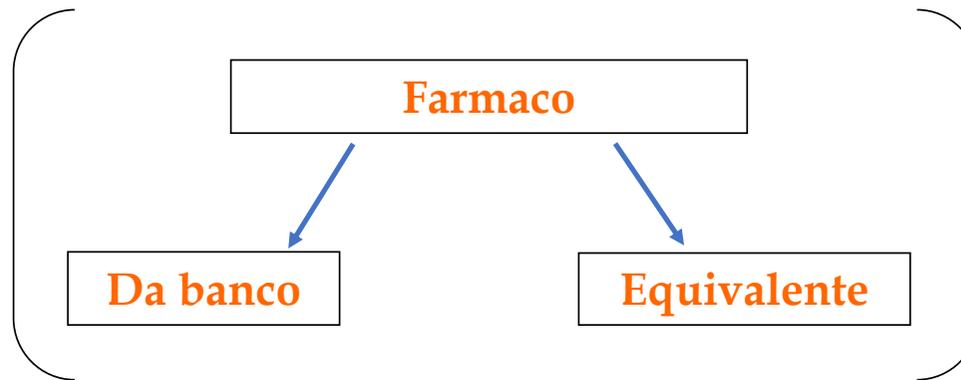


# ESEMPIO

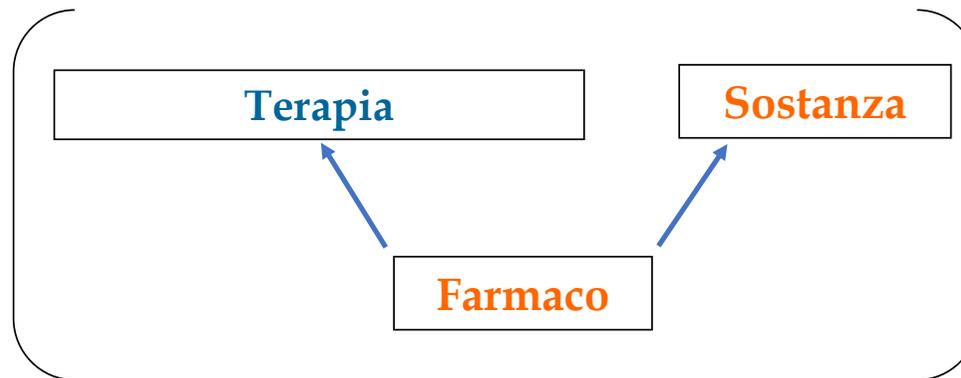


# TIPI DI EREDITARETÀ

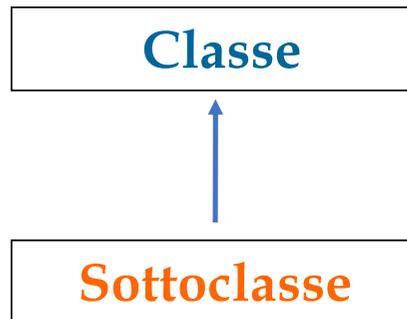
Ereditarietà  
singola



Ereditarietà  
multipla



# PROPRIETÀ DELL'EREDITARIETÀ



## ESTENSIONE

la sottoclasse  
AGGIUNGE NUOVI  
METODI/ATTRIBUTI

## RIDEFINIZIONE

la sottoclasse  
RIDEFINISCE I  
METODI

OVERRIDING = riscrittura del  
codice del metodo

# POLIMORFISMO

**OVERRIDING** = i *metodi* possono *assumere forme diverse* (cioè *implementazioni diverse*) all'interno della gerarchia delle classi

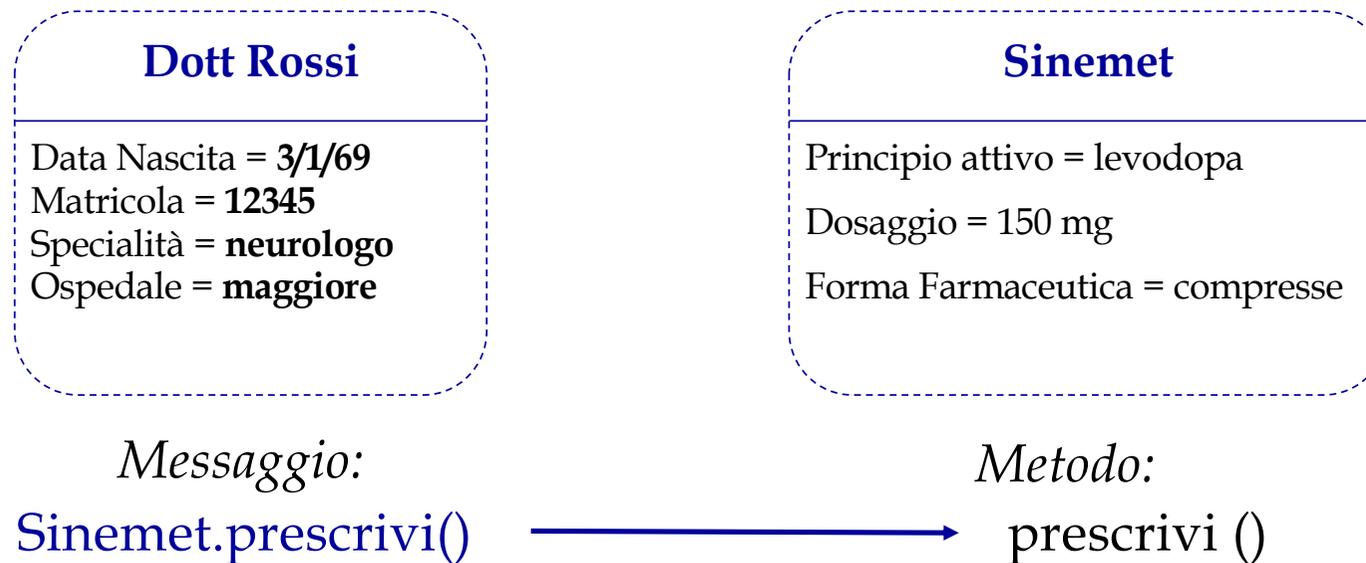
ES. IL METODO "PRESCRIVI" SARÀ IMPLEMENTATO DIVERSAMENTE NEL FARMACO DA BANCO E NEL FARMACO GENERICO

**OVERLOADING** = i *metodi* possono *assumere forme diverse* (cioè *implementazioni diverse*) all'interno della stessa classe

ES. IL METODO "CAMBIA DOSAGGIO" Può RICHIEDERE O DI CAMBIARE IL NUMERO DI ASSUNZIONI O DI CAMBIARE IL NUMERO DI DOSI PER ASSUNZIONE

# INTERAZIONE TRA OGGETTI: MESSAGGI

- Un programma ad oggetti è caratterizzato dalla presenza di tanti oggetti che **interagiscono fra loro attraverso il meccanismo dello scambio di messaggi**
- I messaggi possono:
  - Richiedere un'informazione su un oggetto
  - Modificare lo stato di un oggetto



# DIAGRAMMI UML

## VISTE STATICHE

RAPPRESENTAZIONE  
DELLE CLASSI

- Use case diagrams
- **Class diagram**
- Component diagram
- Deployment diagram
- Object diagram

## VISTE DINAMICHE

- **Sequence diagrams** RAPPRESENTAZIONE DELLO SCAMBIO DI MESSAGGI

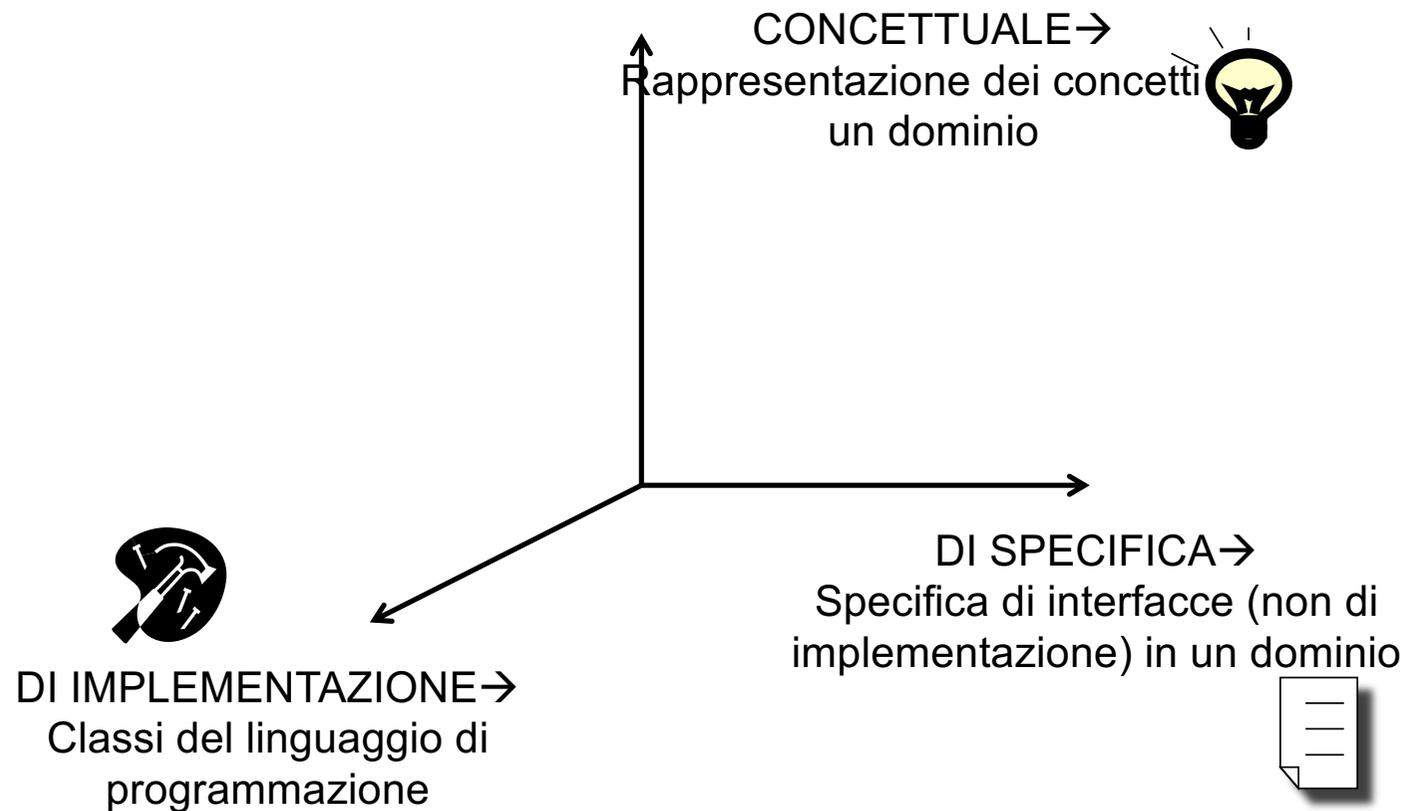
- Collaboration diagrams

- State chart diagrams

- **Activity diagrams** RAPPRESENTAZIONE DELLE SEQUENZE DI AZIONI

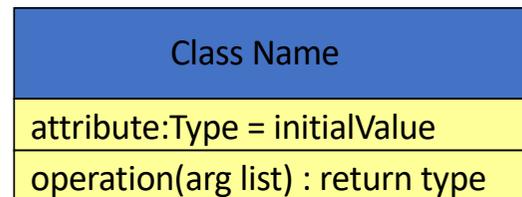
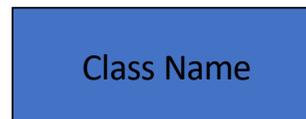
# CLASS DIAGRAMS

- Le classi rappresentano OGGETTI e CATEGORIE con significato diverso



# CLASS DIAGRAM: NOTAZIONE GRAFICA

- Ogni classe ha un'intestazione, degli attributi e delle operazioni
- Una classe viene rappresentata graficamente da un rettangolo contenente il nome della classe e, opzionalmente, l'elenco degli attributi e delle operazioni.

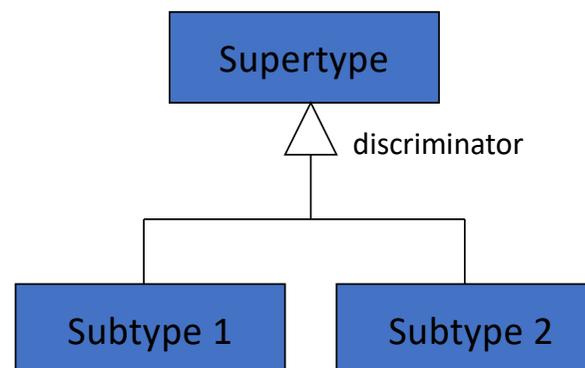


# CLASS DIAGRAM: NOTAZIONE GRAFICA

## Association

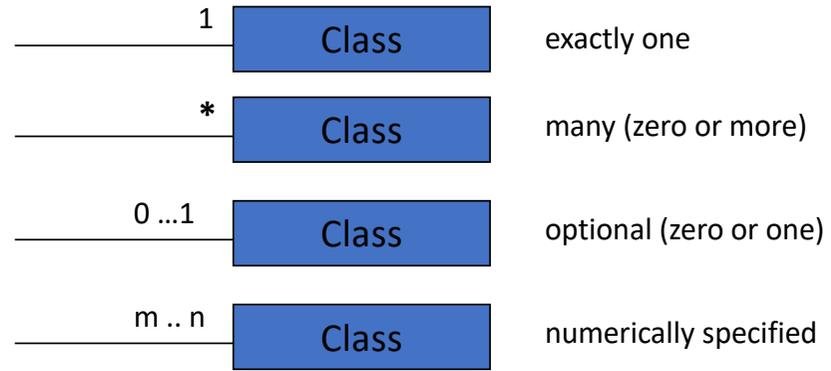


## Generalization

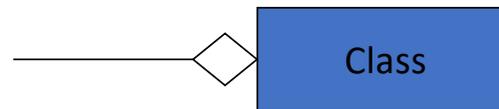


# CLASS DIAGRAM: NOTAZIONE GRAFICA

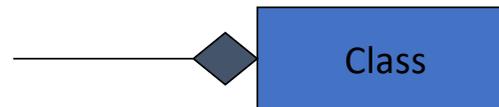
## Multiplicities



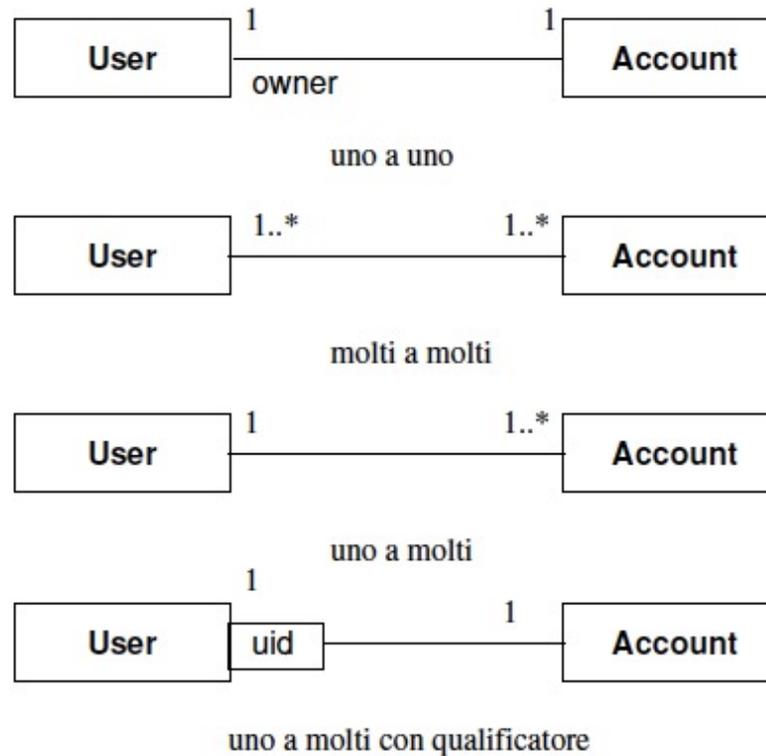
## Aggregation



## Composition



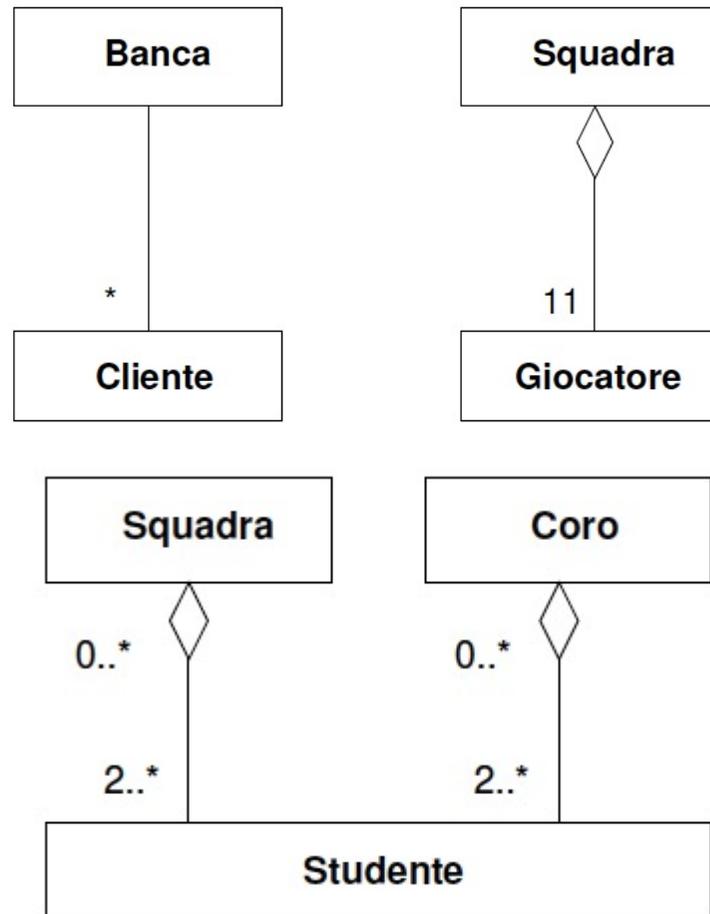
# ASSOCIAZIONI E MOLTEPLICITÀ



Uid è il qualificatore dell'account che rende la relazione 1..1 → uno user può avere più account ma se considero un certo account con un certo uid, avrà uno ed un solo utente associato

# AGGREGAZIONI

**AGGREGAZIONE =**  
associazione che  
lega un'entità  
complessa  
(aggregato) alle  
proprie parti  
componenti.

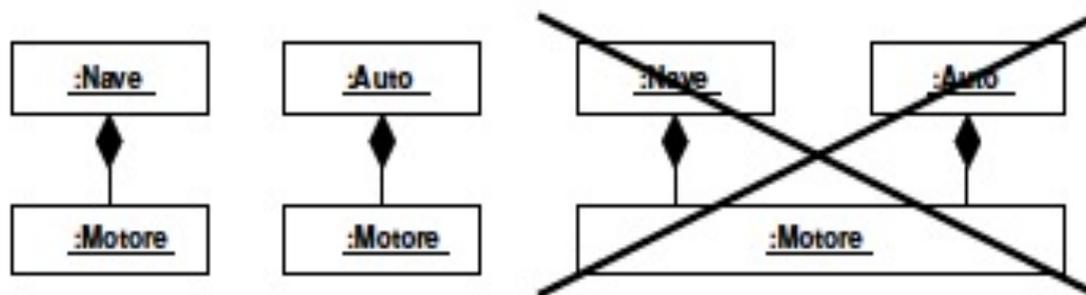
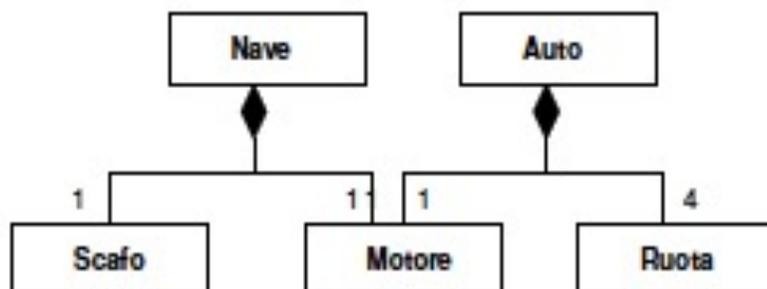


I clienti non fanno parte della loro banca, mentre i giocatori fanno parte della squadra

Uno studente può far parte sia del coro (che è formato da studenti) che della squadra (che è formata da studenti)

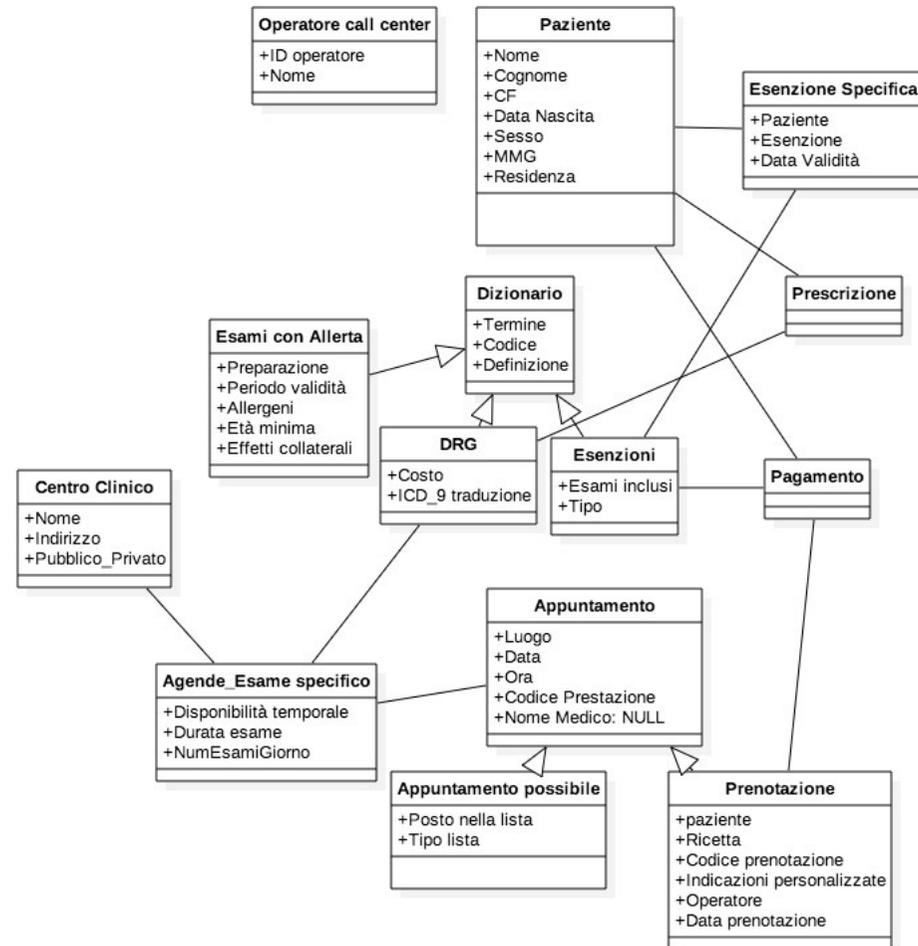
# COMPOSIZIONE

**COMPOSIZIONE = subordinazione strutturale rigida in cui l'aggregato ha il completo e unico controllo delle parti componenti.**

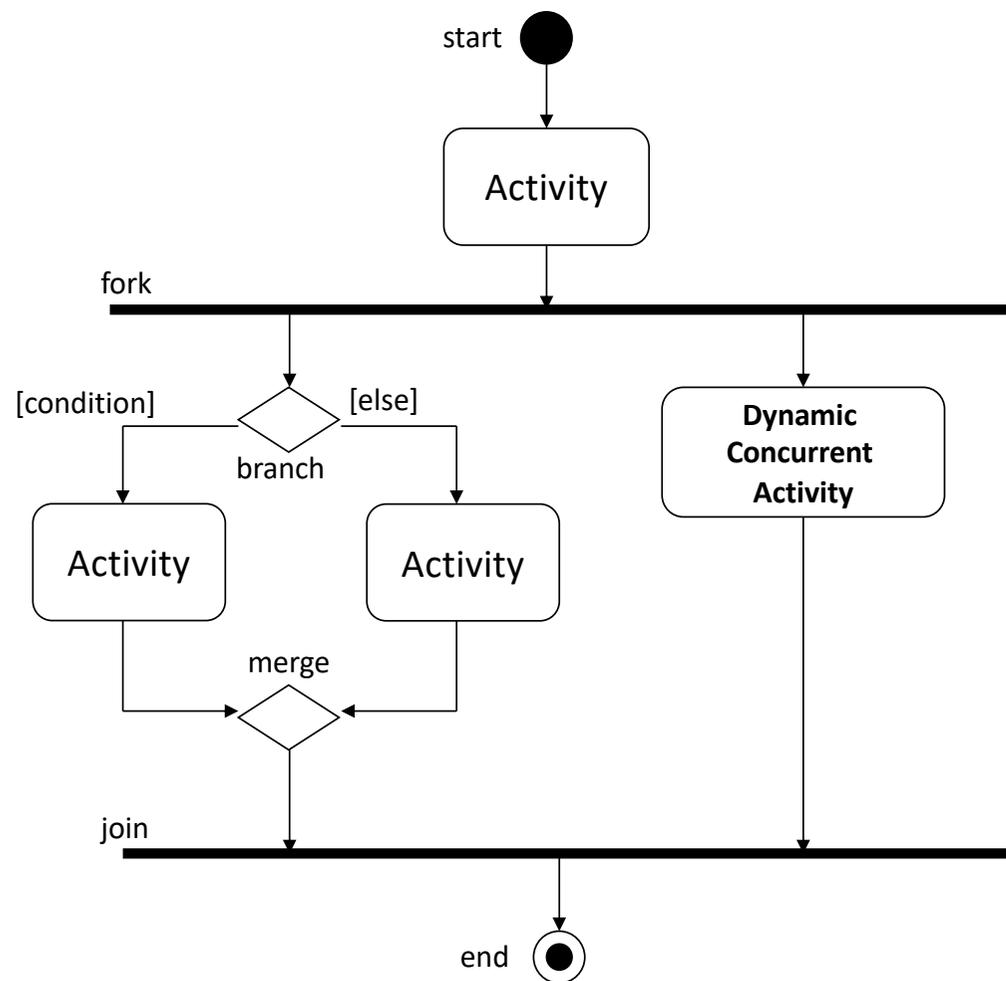


- Una classe può appartenere come componente a più di una composizione
- Un'istanza di una classe componente può appartenere ad una sola istanza di una classe composta

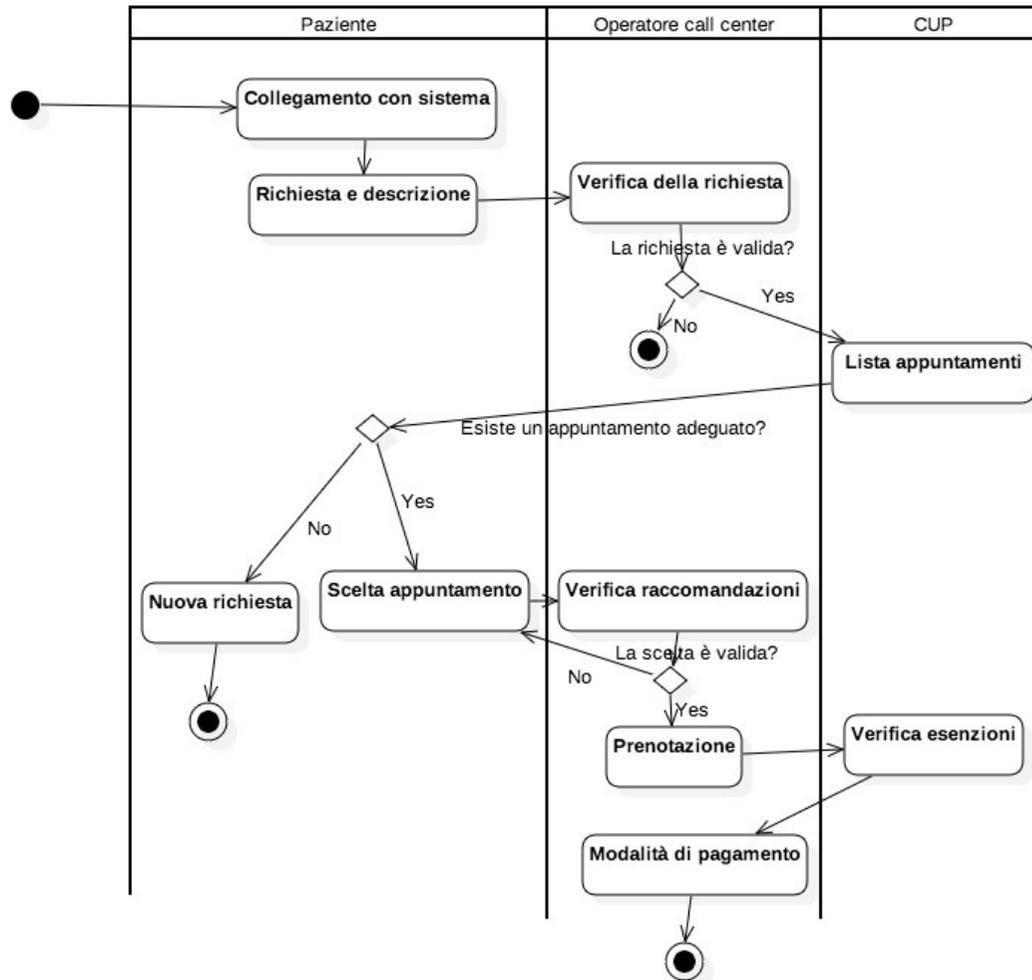
# CLASS DIAGRAM ESEMPIO



# ACTIVITY DIAGRAM: NOTAZIONE GRAFICA



# ACTIVITY DIAGRAM ESEMPIO

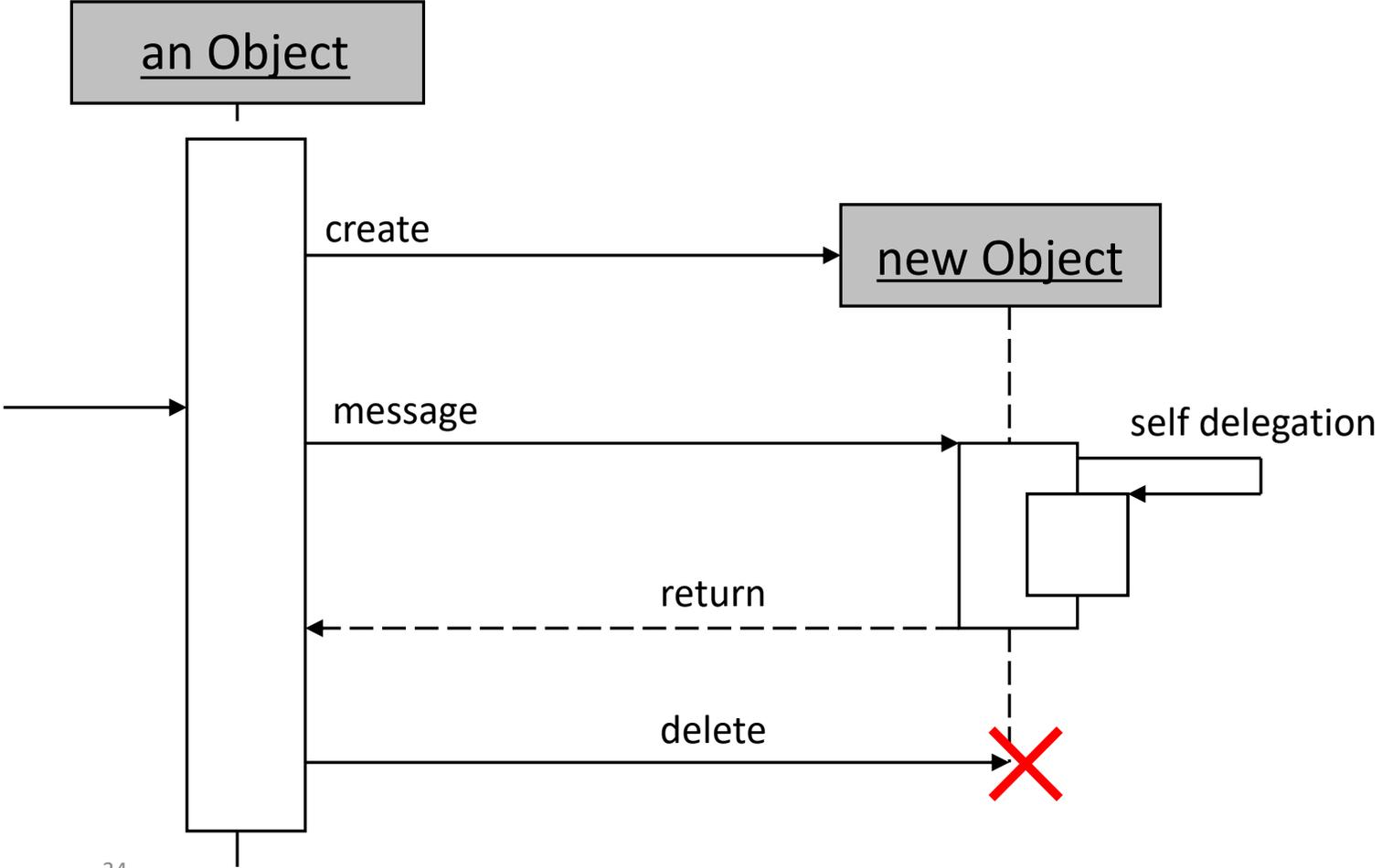


**SWIMLANE** = definizione delle attività che ricadono sotto uno specifico attore

# SEQUENCE DIAGRAMS

- Diagrammi di interazione
- Rappresentano i messaggi che gli oggetti si scambiano in determinate situazioni
- Oggetti → rettangoli posti all'inizio di linee verticali tratteggiate
- Linea della vita dell'oggetto →
  - Linea verticale tratteggiata
  - Rettangolo verticale sulla linea della vita: momento in cui l'oggetto è attivo e scambia messaggi
- Messaggi →
  - Frecce tra le linee della vita di due oggetti
  - L'ordine va dall'alto verso il basso
  - Selfcall: messaggio che l'oggetto scambia con se stesso

# SEQUENCE DIAGRAM: NOTAZIONE GRAFICA



# SEQUENCE DIAGRAM: ESEMPIO

