# 04-UML

Con uml si possono realizzare diversi diagrammi:

* Class diagrams → concettuale
* Activity diagram → processi
* Use Case diagram → funzionale

## CLASS DIAGRAM:

* **Immagine che contiene testo, Carattere, schermata, numero

  Descrizione generata automaticamenteClasse**: descrive un insieme di oggetti (nome singolare)
  + Definisce attività comuni
  + In una classe vive il concetto di esistenza: l’istanza di una classe è un oggetto di un tipo rappresentato da quella classe
* **Immagine che contiene testo, schermata, Carattere, linea

  Descrizione generata automaticamenteIstanza**/**oggetto**: modello di un oggetto (studenti, esame, finestra)
  + Identità
  + Attributi
  + Operazione che possono essere svolte su oggetto
  + Messaggio che può ricevere

*l’istanza di un oggetto vive da sola*

* **Immagine che contiene testo, Carattere, strumento

  Descrizione generata automaticamenteAttributo**: proprietà elementari della classe
  + Nome
  + Tipo
* **Operazione**
* **Associazione** → logica tra due oggetti

Posso usare il class diagram:

* Per fare il glossario
* Per fare il design del sistema (HW+SW)
* Definire le classi SW che andranno ad implementare ciò che stiamo definendo (SW design)

Posso avere delle entità corrispondenti nell’applicazione sw, a **diversi livelli**:

* **Strato dato**: impiegato in DB
* **Business logic**: impiegato in classe Java
* **Strato** **presentazione**: form per insierire i dati dell’impiegato, form per mostrare i dati

*Dunque, prima di fare il class diagram devo specificare per quale livello lo sto facendo*

**Nel class diagram posso definire**:

* Entità fisiche (persona, animale)
* Ruoli (professore)
* Entità sociali, organizzative (Università)
* Eventi (quando avviene la vendita: ordine, richiesta)
* Intervalli di tempo (periodo di autonoleggio, prenotazione albergo)
* Entità geografiche (strada, regione)
* Reports (previsioni del tempo)

**SW design**: ho degli aspetti più specifici

* Collezioni
* String, Integer, Float
* Glasse GUI

**Immagine che contiene testo, schermata, Carattere, linea

Descrizione generata automaticamenteCollegamenti**: collegano le varie entità, modellando le proprietà tra gli oggetti

Immagine che contiene testo, schermata, Carattere, linea

Descrizione generata automaticamente

**Associazioni**: insieme di collegamenti tra insiemi di classi diverse**Immagine che contiene testo, schermata, diagramma, Carattere

Descrizione generata automaticamente**. Caratterizzate da:

* **>** indica la direzione in cui va letta (usando quella forma verbale)
* Immagine che contiene testo, schermata, diagramma, linea

  Descrizione generata automaticamenteMolteplicità: **min .. max**
  + \* → 0+
  + 2 → esattamente 2
  + 0..2 → 0, 1 o 2
* **Nome** di un ruolo dell’associazione

Es: ricorsiva Immagine che contiene testo, Carattere, schermata, linea

Descrizione generata automaticamente

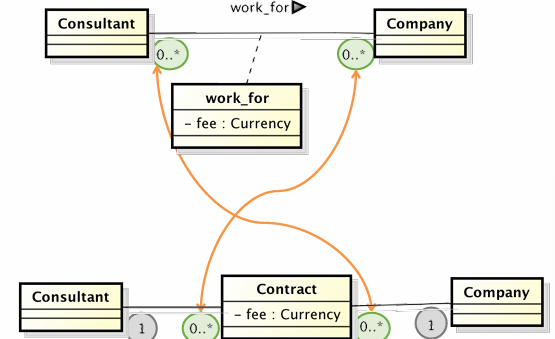
**Stile** Associazioni:

* **CamelCase**
* Classe sempre al singolare
* Attributo → va scritto il nome (eventualmente tipo se serve)

**Immagine che contiene diagramma, linea, Parallelo, Disegno tecnico

Descrizione generata automaticamente3 casi** particolari di associazione:

* **Aggregazione**:
  + A possiede B
  + Immagine che contiene linea, schermata, diagramma, Parallelo

    Descrizione generata automaticamenteB è una parte di A
* **Composizione**:
  + Aggregazione stringente
    - **Ciclo di vita di B è strettamente legato ad A**
* **Classe di Associazione**: aggiungere proprietà tra oggetti al nostro SW
  + in alcuni casi potrebbe essere limitante in quanto   
    non permette di fare alcune ass. ternarie
    - *Nel caso in esempio (in alto) un consultant può avere per ogni company un solo work\_for,   
      di conseguenza un solo fee*
    - *Se volessi avere più fee per una relazione consultant-company, dovrei aggiungere una* ***classe intermedia*** *vera e propria (come in basso)*

**Specializzazione/Generalizzazione**

Immagine che contiene testo, linea, diagramma, schermata

Descrizione generata automaticamenteImmagine che contiene linea, Rettangolo, schermata

Descrizione generata automaticamenteClassi figlie specializzano il comportamento di una classe generica

* + **B (figlia) specializza A**
  + A è una generalizzazione di B

*Classe figlia eredita metodi, attributi e associazioni della classe padre*

*Immagine che contiene testo, linea, Carattere, schermata

Descrizione generata automaticamente*

**NOTA**:

* Immagine che contiene testo, Carattere, schermata, linea

  Descrizione generata automaticamenteClassi devono avere nome singolare
* Attributi non devono essere composti, se mi serve composto → devo aggiungere una classe e quindi una relazione (indirizzo)
* Immagine che contiene linea, testo, schermata, Carattere

  Descrizione generata automaticamenteAttributo deve rappresentare un singolo oggetto
* Non usare relazioni per situazioni che possono cambiare
  + Immagine che contiene testo, schermata, linea, Carattere

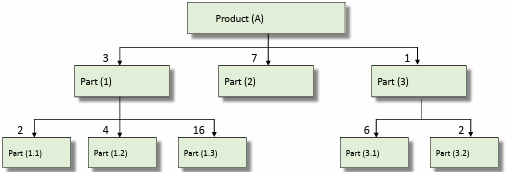
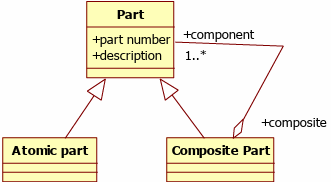
    Descrizione generata automaticamenteEs: entra, esce → corrispondono ad eventi → evitare
* Non ripetere gli attributi (no chiavi esterne)
* Usare loop solo quando effettivamente servono
  + In questo caso utile perché riesco ad avere una molteplicità 1 verso ufficio
    - Immagine che contiene testo, linea, Carattere, Rettangolo

      Descrizione generata automaticamenteSe non ottenessi ciò, sarebbe inutile loop
* Immagine che contiene testo, linea, Carattere, schermata

  Descrizione generata automaticamenteIn alcuni casi è meglio usare i ruoli e non le sottoclassi
  + Se un utente di una sottoclasse potrebbe diventare di un’altra classe

→ non devo usare le sottoclassi ma i ruoli

* + - Se un capitano può diventare copilota



**BOM: Bill Of Material**

### **DEPLOIMENT DIAGRAM – 20/03/2024**

Definisce quale artifact gira su ciascun nodo.

**Immagine che contiene linea, Rettangolo, Carattere, schermata

Descrizione generata automaticamenteNODO**: entità fisica o sw capace di processare Immagine che contiene Rettangolo, testo, Carattere, schermata

Descrizione generata automaticamente

**Immagine che contiene testo, schermata, Rettangolo, diagramma

Descrizione generata automaticamenteImmagine che contiene testo, schermata, linea, Carattere

Descrizione generata automaticamente**

**ASSOCIATION**: link fisico

Può essere annidato

**Immagine che contiene testo, Carattere, Rettangolo, schermata

Descrizione generata automaticamenteArtifact**: qualcosa che definsce un sw (file sorgente, file eseguibile, libreria, tabella db)

Immagine che contiene testo, schermata, Carattere, linea

Descrizione generata automaticamenteEs:

Immagine che contiene testo, schermata, Carattere, Rettangolo

Descrizione generata automaticamente

se integrato nel server: