UML

CLASS DIAGRAM

prospettive diverse: ambiti diversi

→ prospettiva **concettuale**
- → analista
- → elementi da modellare
- → classe UML: concetto del dominio

→ prospettiva **software**
- → sviluppatore
- → design di un sw (moduli da implementare)
- → classe UML: classe effettiva di un linguaggio OO

Classe UML

→ pensata tipo OO (con attributi e operazioni)

→ rappresentata come rettangolo
- → **nome** e opzionalmente attributi e operazioni (alto livello)
- → **attributi**:= visibilità nome: tipo [molteplicità] = default {proprietà}
  - → visibilità public, private, package, …
  - → nome attributo unica parte obbligatoria
  - → molteplicità: quantitativo dell'attributo (numerico, Kleene-like, …)
    - → insiemistico: attributi di default non ordinati, no ripetizioni
- → **operazioni**: visibilità nome (lista_params): tipo_return {proprietà}
  - → lista_params: direzione nome: tipo = default
    - → direzione: in, out, inout … (C#-like); default in
  - → solo dichiarazione, no implementazione/behaviour

→ **datatype**: stereotipo <<datatype>>
- → valori e non oggetti
  - → 2 istanze di oggetti sono diverse, 2 istanze di datatype sono la stessa cosa
- → ad es. due date istanziate uguali dovrebbero essere la stessa cosa
- → specializzazione: ad es. <<enumeration>> è un caso speciale di <<datatype>>

→ **associazioni**: relazione tra classi UML (chiamata a operazioni / attributi di tipo associata)
- → predicato verbale completo di ▸ a indicare la direzione di lettura (nome associaz.)
  - → alternativa dicitura di ruolo in sostantivo vicino alla classe che lo assume
- → freccia su linea invece che su nome associazione: significato diverso **!**
  - → navigazione di reperimento informazioni all'interno del diagramma
- → **molteplicità**: stile ER (notazione numerica $n..m$ con $n, m \in \mathbb{N} \cup \{" * "\}$)
- → riflessive: di classe con sé stessa
- → bidirezionali: doppie frecce ↔; complesse da implementare (sincronizzazione)
- → **aggregazioni**: rappresentazione a rombo; relazione "tutto-parte"
- → **composizioni**: aggregazioni forti; rombo pieno; "composto da": tolto padre, dipendenti senza senso
- → **generalizzazione**: concettuale, relazione "is a": istanza di classe è anche istanza di sua superclasse
- → **ereditarietà**: implementativa: meccanismo con il quale gli specializzati ereditano
  - → rappresentazione grafica UML con freccina completa di △

→ note: tipo commenti esplicativi, rappresentati con rettangolino tagliato

→ **convenzione**:
- dato primitivo → attributo
- dato di tipo classe → associazione

→ **staticità**:
→ di attributi e/o operazioni (standard OO)
→ attributi: condivisione tra istanze
→ operazioni: metodi generali

→ **operazioni**: dichiarazione di procedura/funzione
→ **metodo**: corpo effettivo implementativo della funzione (behaviour)

→ elementi **astratti**: rappresentati in italico oppure con stereotipo <<abstract>>

→ **dipendenze**: linea tratteggiata
→ più deboli e quindi più presenti: non necessario rappresentarle tutte

---

OBJECT DIAGRAM
→ rappresentazione degli oggetti istanziati e non delle classi
→ nomi sottolineati
→ associazioni link
→ rappresentazione grafica nome: classe
→ dopo class diagram: esempî e specializzazioni istanziali
→ possibilità di integrare class e object diagram insieme
→ linea tratteggiata in class diagram (dipendenza, più debole dell'associazione)