INTRODUZIONE A UML

LEZIONE 9
27/10/2025
VINCENZO RICCIO

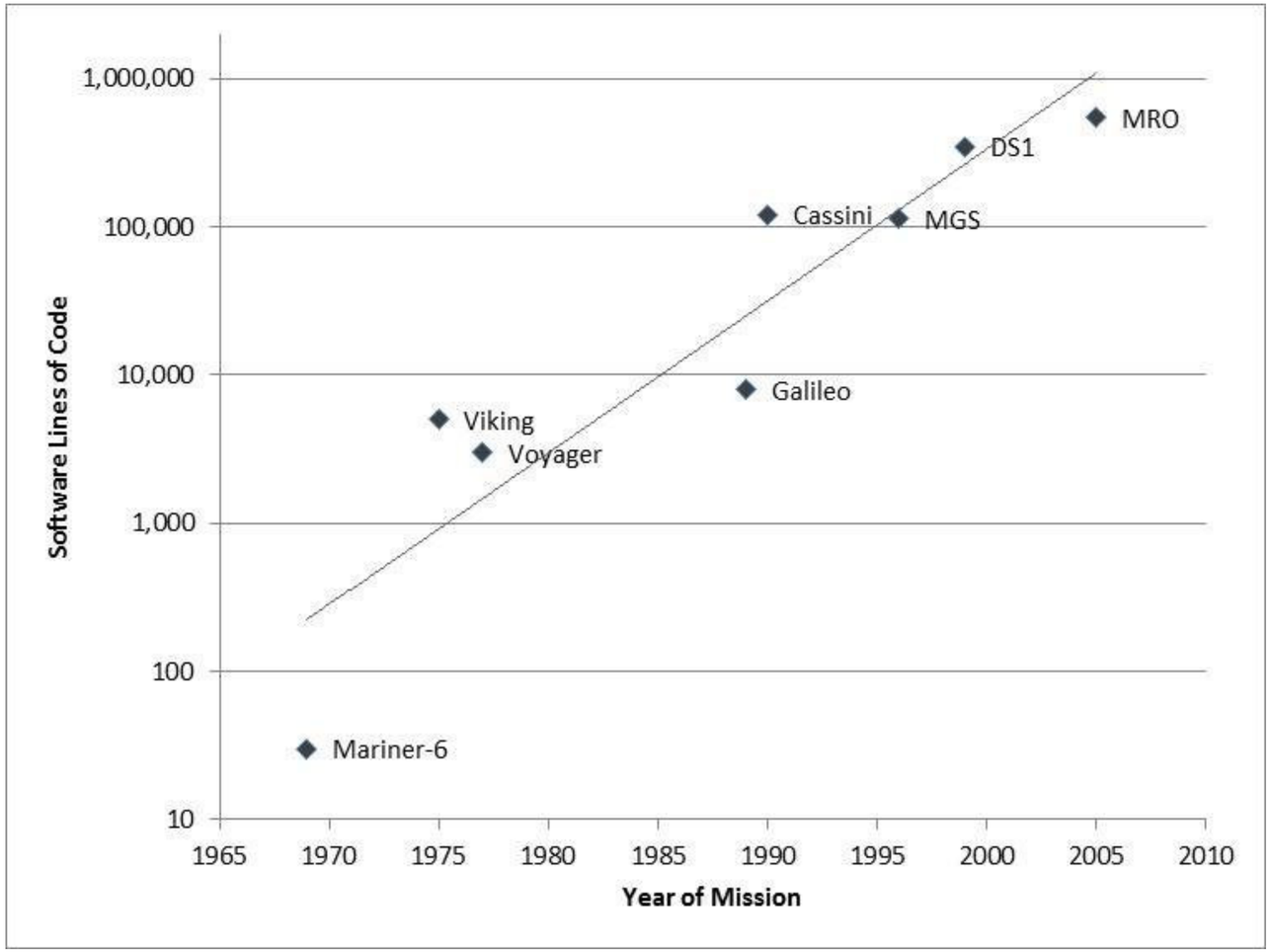
RIFERIMENTI

Vincenzo Riccio
Ingegneria del Software 2025/2026
Università degli Studi di Udine

- Fowler - Capitolo 1
- Sommerville - Capitolo 5 (intro)

MOTIVAZIONE

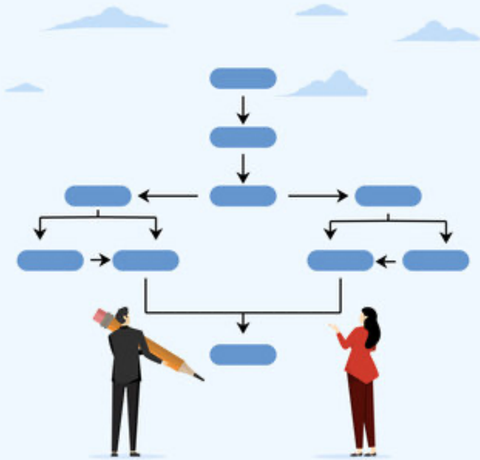
- I moderni sistemi Software tendono a crescere e a diventare sempre più complessi
- *Esempio:* i moderni sistemi operativi raggiungono dimensioni di milioni di linee di codice



MOTIVAZIONE

Vincenzo Riccio
Ingegneria del Software 2025/2026
Università degli Studi di Udine

- É impensabile comprendere sistemi software complessi direttamente dal codice
- Il codice è spesso incomprensibile (soprattutto per chi non ha partecipato al suo sviluppo)
- Necessità di forme di rappresentazione più astratte per discutere le scelte di progetto
- La modellazione è un modo per gestire la complessità del software



MODELLAZIONE DI UN SISTEMA

Vincenzo Riccio
Ingegneria del Software 2025/2026
Università degli Studi di Udine

- Processo che sviluppa modelli astratti di un sistema, non una rappresentazione alternativa
  - La **rappresentazione** di un sistema mantiene tutte le informazioni sull'entità che rappresenta
  - Un'**astrazione** semplifica deliberatamente un sistema evidenziandone le caratteristiche più salienti, eventualmente tralasciando altre caratteristiche
- Ad un sistema possono corrispondere più modelli
  - Ogni modello rappresenta una differente vista o prospettiva del sistema
- Si possono utilizzare notazioni grafiche o notazioni matematiche

TERMINOLOGIA

Vincenzo Riccio
Ingegneria del Software 2025/2026
Università degli Studi di Udine

- Un **modello** è un'astrazione che descrive un sistema o un sotto-sistema
- Una **vista (o prospettiva)** è una descrizione di aspetti specifici di un sistema da una certa prospettiva, in cui si omettono dettagli non rilevanti per tale prospettiva
- Una **notazione** è un insieme di elementi grafici o testuali e regole per rappresentare le viste
- Diverse viste/modelli di un sistema possono sovrapporsi

LINGUAGGI DI MODELLAZIONE

Vincenzo Riccio
Ingegneria del Software 2025/2026
Università degli Studi di Udine

- Per descrivere i modelli si utilizzano linguaggi di modellazione
- In passato, diversi linguaggi di modellazione erano utilizzati a supporto delle metodologie che si applicavano nelle varie fasi del processo di sviluppo del software
- Negli ultimi anni il linguaggio UML si sta affermando come linguaggio unificato che possa essere utilizzato in tutte le attività di modellazione
- Esistono anche altri linguaggi come SysML, variante di UML per la modellazione generale di sistemi complessi (ad es. IoT)

STORIA DI UML 1/2

Vincenzo Riccio
Ingegneria del Software 2025/2026
Università degli Studi di Udine

- UML (Unified Modelling Language) nasce come linguaggio grafico standard per modellare software object-oriented
- Tra la fine degli anni ‘80 e gli inizi degli anni ‘90 fecero la comparsa i primi processi di sviluppo object-oriented
- La proliferazione di metodi e notazioni diverse creavano confusione
- Due importanti metodologi, Rumbaugh e Booch, decisero di unire i loro approcci nel 1994. A loro si aggiunse Jacobson nel 1995

I “Three Amigos”

STORIA DI UML 2/2

Vincenzo Riccio
Ingegneria del Software 2025/2026
Università degli Studi di Udine

- Nel 1997 il consorzio di società Object Management Group (OMG) cominciò il processo di standardizzazione di UML
- L'obiettivo iniziale era quello di definire standard che permettessero l'interoperabilità tra sistemi orientati agli oggetti
- Nel Luglio 2005 è stata rilasciata la prima release di UML 2

UML: NOTAZIONI + META-MODELLO

Vincenzo Riccio
Ingegneria del Software 2025/2026
Università degli Studi di Udine

- UML è una famiglia di **notazioni grafiche** che si basano su un singolo **meta-modello**
- Un meta-modello è un modello che definisce i concetti stessi del linguaggio di modellazione, indica secondo quali regole sia possibile costruire modelli UML
- UML non è una metodologia: il suo obiettivo è di fornire un supporto al processo di sviluppo software (sia per la descrizione che per il progetto)
- Può essere usato all'interno dei processi di sviluppo che adottano le proprie metodologie

REGOLE PRESCRITTIVE E DESCRITTIVE

- Le regole UML possono essere considerate sia prescrittive che descrittive
  - Le regole **prescrittive** sono regole stabilite da organismi standardizzanti che definiscono precisamente lessico, sintassi e semantica del linguaggio. Fondamentali quando UML è usato come linguaggio di programmazione
  - Le regole **descrittive**, invece sono stabilite per convenzione comune, possono essere meglio comprese guardando come l'UML viene usato nella pratica da un'organizzazione
- Bisogna conoscere sempre le convenzioni particolari della specifica organizzazione e del singolo progetto, anche se al di fuori dello standard

UML

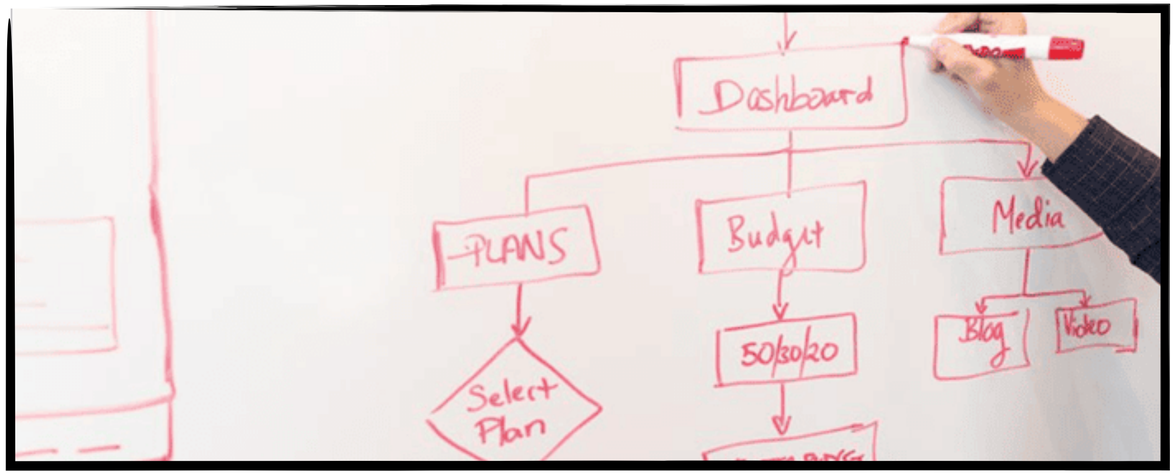
- Secondo Fowler e Mellor, UML può essere usato:
  A. Come **bozza (sketch)**, cioè per tracciare un modello informale di un sistema da realizzare (*forward engineering*) o per descrivere un sistema esistente (*reverse engineering*)
  B. Come **progetto dettagliato (blueprint)**, cioè per realizzare un modello completo della soluzione architetturale del sistema
  C. Come **linguaggio di programmazione** in grado di modellare in maniera completa e precisa il sistema software

- **Forward Engineering:** modello prima di scrivere codice
- **Reverse Engineering:** diagramma per comprendere meglio un progetto esistente

BOZZE: ESPRESSIVITÀ > COMPLETEZZA

Vincenzo Riccio
Ingegneria del Software 2025/2026
Università degli Studi di Udine

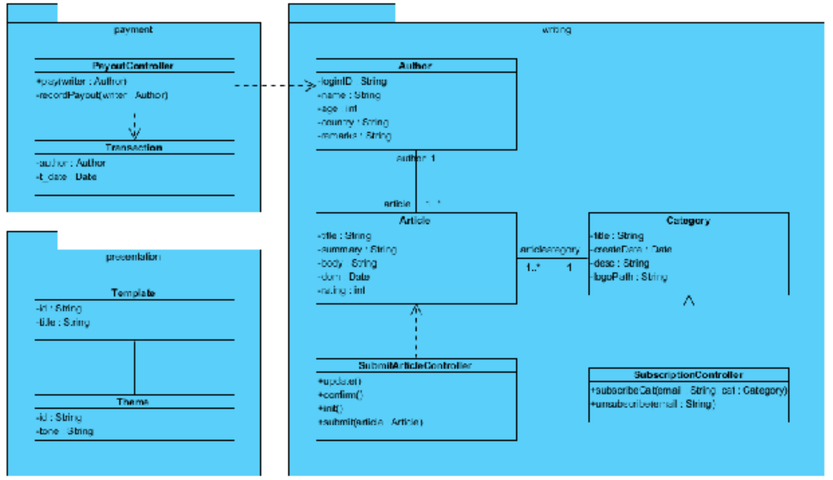
- Quando si realizza una bozza, lo scopo è aiutare la comunicazione e la discussione delle idee, esplorando le soluzioni alternative
- I diagrammi non devono essere esaustivi e definire tutti gli aspetti del codice
- Le bozze sono disegnate spesso in poco tempo e in modo collaborativo, non rispettando tutte le regole formali dello standard



PROGETTO DETTAGLIATO: ESPRESSIVITÀ + COMPLETEZZA

Vincenzo Riccio
Ingegneria del Software 2025/2026
Università degli Studi di Udine

- Quando si realizza o si deve capire un progetto, lo scopo è aiutare la comprensione e la completezza
- Il progettista sviluppa un modello di progetto che lo sviluppatore dovrà realizzare, salvandolo in file condivisi
- La completezza e non ambiguità del modello aiutano il programmatore, che sarà guidato dal modello e non dovrà avere aspetti ambigui da interpretare



UML COME LINGUAGGIO DI PROGRAMMAZIONE

Vincenzo Riccio
Ingegneria del Software 2025/2026
Università degli Studi di Udine

- L'approccio **MDA (Model Driven Architecture)** [Kleppe et al.] esplora la possibilità di usare UML come linguaggio di programmazione*
- Si vorrebbe stabilire una sintassi e una semantica precisi per UML, che portino alla generazione automatica di codice eseguibile rappresentativo del modello
- La sfida risiede nel modellare precisamente anche la logica del progetto
- Il vantaggio consiste nel generare codice per diverse piattaforme target partendo da un modello indipendente dalla piattaforma:
- **Platform Independent Model > Platform Specific Model > Codice**

** UML non è propriamente un linguaggio di programmazione ma di modellazione*

INFORMAZIONI SOPPRESSE

Vincenzo Riccio
Ingegneria del Software 2025/2026
Università degli Studi di Udine

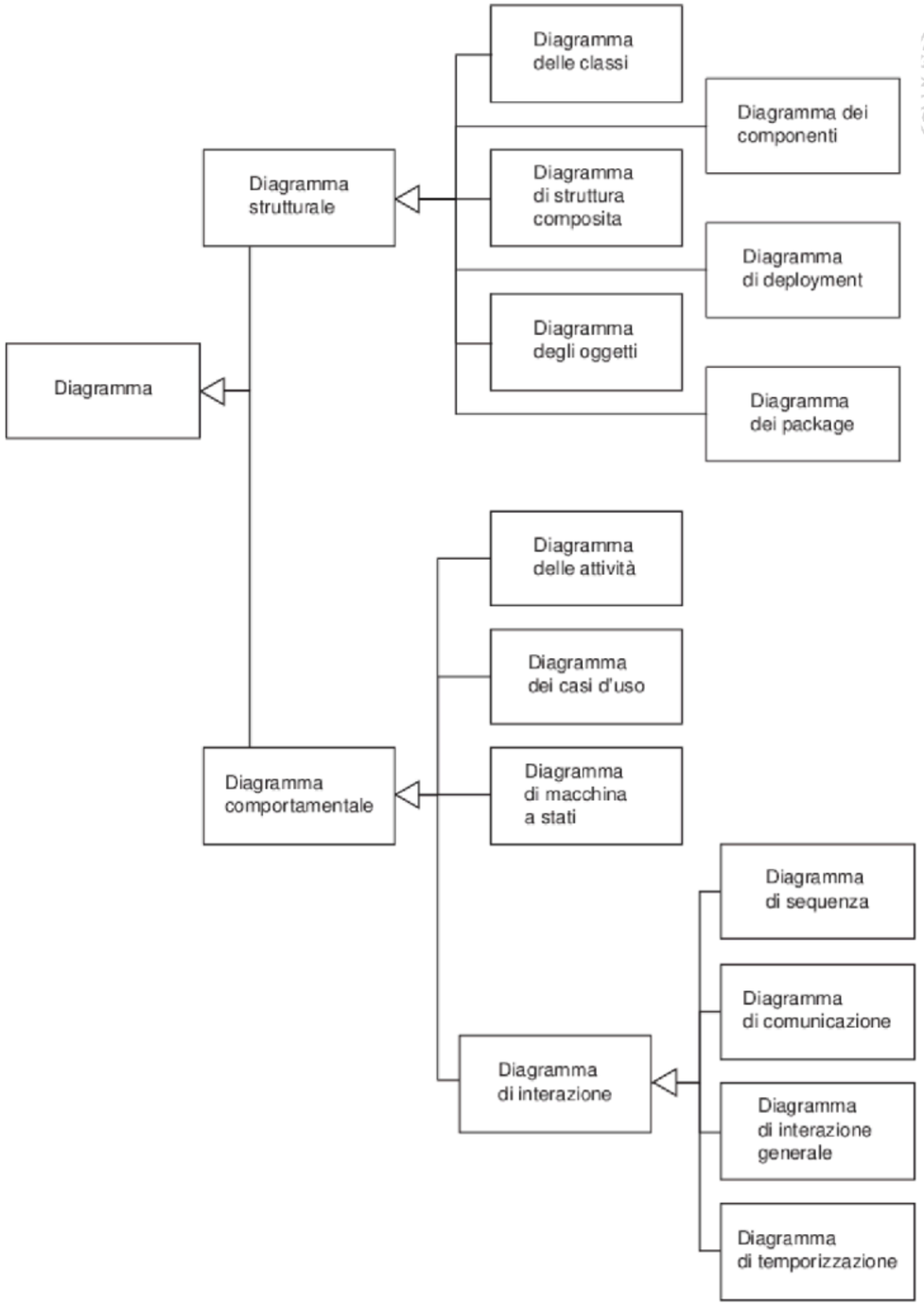
- L’assenza di qualche informazione in un diagramma UML non significa, in generale, che tale informazione non esista
- Alcuni aspetti del problema potrebbero essere assenti da un diagramma perché non ancora trattati nella fase in cui è stato tracciato il diagramma (o non si volevano palesare in tale diagramma per poi inserirli in diagrammi differenti)
- Ad esempio la precisa sequenza di eventi per realizzare una funzionalità può non essere pronta durante l’analisi dei requisiti in cui sono abbozzati i casi d’uso

TIPI DI DIAGRAMMI UML

- UML 2 possiede 13 differenti tipi di diagrammi ufficiali, appartenenti a due categorie:
  - **Diagrammi Strutturali**, modellano l'organizzazione del sistema (ad es. class diagrams, package diagrams, deployment diagrams)
  - **Diagrammi Comportamentali**, modellano il comportamento e le interazioni tra le entità del sistema ad es. use-case diagrams, activity diagrams, state machine diagrams)
- Tali diagrammi rappresentano i deliverables di diverse fasi del ciclo di vita del software, tra cui attività di analisi dei requisiti e attività di progettazione, sia di alto che di basso livello

TIPI DI DIAGRAMMI UML

Diagramma	Capitoli del libro	Scopo	Origine
Attività	11	comportamento procedurale e parallelo	UML 1
Classi	3, 5	classi, loro caratteristiche e relazioni	UML 1
Comunicazione	12	interazione tra oggetti con enfasi sui collegamenti	chiamato "diagramma di collaborazione" in UML 1
Componenti	14	struttura dei componenti e loro connessioni	UML 1
Struttura composita	13	scomposizione di una classe a runtime	introdotto in UML 2
Deployment	8	distribuzione di elaborati in diversi nodi	UML 1
Interazione Generale	16	fusione di un diagramma di sequenza con un diagramma delle attività	introdotto in UML 2
Oggetti	6	configurazione esemplificativa di istanze	presente in UML 1 in modo non ufficiale
Package	7	struttura gerarchica al momento della compilazione	presente in UML 1 in modo non ufficiale
Sequenza	4	interazione tra oggetti, con enfasi sulla sequenza	UML 1
Macchina a stati	10	come gli eventi cambiano un oggetto durante il suo ciclo di vita	UML 1
Temporizzazione	17	interazione tra oggetti, con enfasi sul tempo	introdotto in UML 2
Casi d'uso	9	come gli utenti interagiscono con un sistema	UML 1



PROSPETTIVE UML

Vincenzo Riccio
Ingegneria del Software 2025/2026
Università degli Studi di Udine

- Prospettiva **esterna**: sono modellati il contesto operativo (o ambiente) del sistema
- Prospettiva delle **interazioni**: sono modellate le interazioni tra il contesto e il sistema o tra diverse componenti del sistema
- Prospettiva **strutturale**: sono modellate l'organizzazione del sistema e/o la struttura dei dati
- Prospettiva **comportamentale**: sono modellati il comportamento dinamico del sistema e come esso risponde agli eventi

INTEGRARE UML NEL PROCESSO DI SVILUPPO

Vincenzo Riccio
Ingegneria del Software 2025/2026
Università degli Studi di Udine

- UML può essere utilizzato in diverse fasi del processo di sviluppo
  - **Analisi dei requisiti:** facilita la deduzione dei requisiti, la notazione non deve essere troppo complessa per favorire la comunicazione con il cliente
  - **Progettazione:** modelli più tecnici e dettagliati per descrivere il sistema agli ingegneri che lo devono implementare (nella pratica il codice può distaccarsi dai modelli)
  - **Documentazione (dopo implementazione):** modelli rendono più semplice la descrizione di parti complesse o convogliano messaggi in maniera intuitiva e immediata
  - **Comprensione di software pre-esistente:** evoluzione o reverse-engineering
- Nei processi iterativi ogni iterazione arricchisce i diagrammi delle iterazioni precedenti