Unified Modeling Language (UML)

- È una famiglia di notazioni grafiche che si basano su un singolo meta-modello
- Serve per definire, progettare, realizzare e documentare sistemi sw (in particolare quelli OO)
- Copre l'intero ciclo di vita del sw senza imporre alcun processo di sviluppo predefinito
- È indipendente da qualsiasi linguaggio di programmazione
- È utilizzabile in domini applicativi diversi e per progetti di diverse dimensioni
- È basato sui modelli, che sono uno strumento per gestire la complessità
- È estensibile (per modellare meglio le diverse realtà)
- È sponsorizzato dalle maggiori case produttrici di sw

UML: cronologia

Nato dalla fusione di molti linguaggi grafici di modellazione OO sviluppati tra la fine degli anni ‘80 e l’inizio dei ‘90

Novembre 1997: UML 1.1 (risultato della fusione di UML 1.0, versione rilasciata da Rational Software, con altre proposte) diventa uno standard OMG

Inizio 1999: UML 1.3

Giugno 2003: UML 2.0

Rational Software ora fa parte di IBM

Object Management Group (OMG)

http://www.omg.org

- È un'organizzazione (consorzio) a cui aderiscono circa 800 aziende (leader in campo internazionale)
- Slogan: "Setting vendor-neutral software standards, and enabling distributed, enterprise-wide interoperability"
- Obiettivo: produrre e manutenere un corpo di specifiche che supportino tutte le fasi del ciclo di vita di sw distribuito ed eterogeneo
- Le specifiche sono scritte, influenzate e adottate dalle aziende aderenti
- Chiunque può scaricare gratuitamente le specifiche dal sito web del gruppo
- Ogni azienda, istituzione, organizzazione pubblica può divenire membro del gruppo
- Specifiche OMG:
  - ✓UML (standardizza le rappresentazioni di analisi e progettazione)
  - ✓CORBA (Common Object Request Broker Architecture, fissa gli standard per l'interoperabilità delle applicazioni)

Diagrammi UML

Diagrammi strutturali

- delle classi
- dei componenti
- di struttura composita
- di deployment
- degli oggetti
- dei package

Diagrammi comportamentali

- di attività
- dei casi d'uso
- di stato (o di macchina a stati)
- di sequenza
- di comunicazione (ex collaborazione)
- di interazione generale
- di temporizzazione

} Diagrammi di interazione

Spesso si possono usare elementi di un tipo di diagramma all'interno di un altro

Nessuno comprende o utilizza UML nella sua interezza; la maggior parte delle persone ne usa un piccolo sottoinsieme

Modalità di utilizzo di UML

- Abbozzo (sketch) – su cui si concentra il testo di Fowler
- Progetto dettagliato (blueprint)
- Linguaggio di programmazione

UML come abbozzo

Diagrammi informali (schizzi, bozzetti) incompleti (cioè concentrati solo sulle informazioni più importanti) di parti selezionate del sistema, realizzati, tipicamente su lavagna, in poco tempo e in modo collaborativo dai programmatori in fase di

- Forward engineering (cioè prima di scrivere il codice), al fine di
  - Aiutare la comunicazione e discussione delle idee
  - Considerare alternative
  - Pianificare giorni/settimane di programmazione
- Reverse engineering (cioè a sistema esistente, quando si intende costruire il diagramma UML dello stesso per comprenderlo meglio), al fine di
  - Spiegare come funziona una parte del sistema

Per la loro espressività possono essere inclusi anche nella documentazione del sw

Sono abbozzi (per la loro incompletezza) la maggior parte dei diagrammi UML inclusi nei libri

UML come progetto

Modelli completi dettagliati di un sistema/sottosistema, realizzati, tipicamente usando strumenti CASE, in fase di

- Forward engineering dal progettista e forniti ai programmatori per documentare senza ambiguità tutte le decisioni di progetto da implementare (spesso con lo scopo di ridurre la programmazione a un'attività semplice e relativamente meccanica); Fowler crede che siano difficili da realizzare e rallentino il processo di sviluppo
- Reverse engineering al fine di documentare il codice esistente

UML come linguaggio di programmazione

Diagrammi (di interazione, di stato e di attività) che vengono compilati direttamente in codice eseguibile → codice sorgente e rappresentazione UML coincidono e la nozione di forward e reverse engineering non ha più senso

Significato dei diagrammi

Due prospettive

- Software
  - Di specifica (interfaccia)
  - Di implementazione (es. conversione di codice sorgente in diagrammi UML)
- Concettuale (cioè relativa al dominio applicativo), che ha lo scopo di costruire un vocabolario comune (es. uso di diagrammi UML per comprendere i molti significati dell'espressione *beni patrimoniali* con un gruppo di contabili)

Notazione e meta-modello

Definizione di UML = notazione + meta-modello

Notazione = sintassi grafica del linguaggio di modellazione = insieme degli elementi grafici di ciascun diagramma, dove ogni elemento grafico rappresenta un concetto

↓

quesito: qual è il significato di ciascun concetto?
risposta: manca una definizione formale

↓

Meta-modello = diagramma (solitamente diagramma delle classi) che definisce i concetti del linguaggio (è importante se si vuole usare UML come linguaggio di programmazione)

∄ una definizione formale di corrispondenza fra UML e un qualsiasi linguaggio di programmazione

Quale UML è legale?

UML nella pratica odierna è definito

- sia da regole prescrittive, dettate dallo standard OMG → uso standard (o legale o normativo o ben formato), l'unico consentito come linguaggio di programmazione
- sia da regole descrittive, il cui significato è stabilito per convenzione comune → uso convenzionale

Ulteriore complessità deriva

- da UML 2, che stabilisce notazioni contrastanti con quelle date in UML 1 o con l'uso comune
- dal principio UML di "soppressione", secondo cui qualsiasi informazione può essere soppressa → quando si interpreta un diagramma, di fronte a un'info mancante non si sa se l'autore intendesse semplicemente ometterla o invece usare il valore di default (incluso nel meta-modello)

UML: Motivazioni

Si usa la notazione semiformale UML perché:

- favorisce la comunicazione fra i membri del gruppo di sviluppo
- favorisce la comunicazione con i clienti (in particolare, grazie ai casi d'uso)
- è di grande aiuto nella fase di elicitazione e analisi dei requisiti (collocata entro la fase di Elaborazione del RUP)
- aiuta a sfruttare i vantaggi dell'OO (i linguaggi OO permettono tali vantaggi ma non li forniscono)
- esistono strumenti CASE basati su UML