Introduzione a UML

a cura di **Angelo Furfaro**
da "UML Distilled"
Martin Fowler

Dipartimento di
Ingegneria Informatica, Elettronica, Modellistica e Sistemistica
Università della Calabria, 87036 Rende(CS) - Italy
Email: a.furfaro@unical.it
Web: http://angelo.furfaro.dimes.unical.it

Che cosa è lo Unified Modeling Language (UML)

- UML è una famiglia di notazioni grafiche
- Servono a supportare la descrizione ed il progetto di sistemi software
- Sono particolarmente adatte a sistemi realizzati con il paradigma Orientato agli Oggetti (OO)
- UML è uno standard relativamente aperto controllato dall'Object Management Group (OMG)
- UML nasce dalla fusione di alcuni linguaggi grafici di modellazione OO sviluppati tra gli anni '80 e '90.
- UML è stato riconosciuto dall'OMG nel 1997 nella versione 1.0
- Versioni successive:
  - 1.2 del 1998, con una serie di aggiustamenti rispetto alla versione precedente
  - 1.3 del 1999, contenente una revisione più profonda del linguaggio
  - 1.4 del 2000, con varie modifiche estetiche e funzionali minori
  - 2.0 del 2003, che comprende numerose estensioni per settori applicativi specifici (es. per applicazioni in tempo reale)

Usi di UML

UML come abbozzo

- Gli sviluppatori usano UML per documentare alcuni aspetti del sistema
- Gli abbozzi UML possono essere utilizzati sia in fase di costruzione di un sistema (forward engineering) che nel reverse engineering di un sistema esistente
- Lo scopo di questi diagrammi informali è supportare la comunicazione, la discussione delle idee e la valutazione delle alternative all'interno del team di sviluppo
- Non servono a descrivere dettagliatamente il codice da sviluppare, ma solo gli aspetti più importanti di cui si vuole discutere
- Nel reverse engineering, diagrammi abbozzati servono a comprendere il funzionamento di parti del sistema

Usi di UML

UML come progetto

- È l'uso che di UML si fa nella fase di progettazione
- Lo scopo è quello di fornire ai programmatori un modello dettagliato da implementare
- I diagrammi devono essere sufficientemente completi in modo da esprimere tutte le decisioni progettuali
- Nella creazione di progetti dettagliati si possono descrivere tutti gli aspetti di un sistema o ci si può concentrare su una determinata area
- Spesso si definiscono in dettaglio i modelli delle interfacce tra i vari sottosistemi e si lasciano gli aspetti prettamente implementativi a cura del programmatore
- Nel reverse engineering i progetti dettagliati servono a documentare il codice esistente in modo dettagliato
- I diagrammi sono prodotti per mezzo di strumenti di CASE (Computer Aided Software Engineering)

Usi di UML

UML come linguaggio di programmazione

- Quanti più dettagli vengono specificati in fase di progettazione tanto più meccanica diviene la fase di programmazione
- La programmazione può essere automatizzata fino ad un certo livello
- Alcuni strumenti CASE includono forme di generazione automatica del codice
- Nel caso estremo, quando tutti gli aspetti sono dettagliati con precisione è possibile derivare direttamente il codice eseguibile a partire dal progetto

Prospettive software e concettuale

- **Prospettiva software**: si usa UML per modellare sistemi, gli elementi del modello corrispondono ad elementi del sistema.
- **Prospettiva concettuale**: UML rappresenta la descrizione dei concetti del dominio di interesse.

Diagrammi di UML 2

- UML2 definisce 13 diagrammi ufficiali (vedi tabella nella slide successiva)
- I diagrammi costituiscono il punto di partenza di chi si avvicina allo studio di UML
- Gli autori di UML non considerano i diagrammi l'aspetto centrale del linguaggio
- La sintassi dei diagrammi non è molto rigida: si possono usare elementi di un tipo di diagramma all'interno di un altro
- Lo standard UML indica quali elementi sono tipicamente inclusi in determinati diagrammi

Diagrammi UML2

Diagramma	Scopo	Origine
Casi d'uso	interazioni tra utenti e sistema	UML 1
Classi	classi, loro caratteristiche e relazioni	UML 1
Sequenza	interazioni tra oggetti, con enfasi sulla sequenza	UML 1
Oggetti	configurazione esemplificativa di istanze	UML 1*
Package	struttura gerarchica a tempo di compilazione	UML 1*
Deployment	distribuzione artefatti in diversi nodi	UML 1
Macchine a stati	reazioni di un oggetto agli eventi	UML 1
Attività	comportamento procedurale e parallelo	UML 1
Comunicazione	interazione tra oggetti con enfasi sui collegamenti	UML 1†
Struttura composita	scomposizione di una classe a runtime	UML 2
Componenti	struttura dei componenti e loro connessioni	UML 1
Interazione generale	fusione di un diagramma di sequenza ed uno di attività	UML 2
Temporizzazione	interazione tra oggetti con enfasi sul tempo	UML 2

* presente in modo non ufficiale, † era chiamato "diagramma di collaborazione"

UML nel processo di sviluppo

Analisi dei requisiti

- I diagrammi di casi d'uso descrivono le interazioni tra le entità esterne ed il sistema
- Un diagramma delle classi di livello concettuale può essere utilizzato per definire un *vocabolario* per descrivere il dominio di interesse ed esprimere le relazioni tra i concetti di tale dominio
- Diagrammi di attività possono essere impiegati per:
  - illustrare il flusso di lavoro che interessa il sistema
  - chiarire il funzionamento dei casi d'uso più complessi
- Un diagramma di stato è utile per descrivere entità che hanno comportamenti che possono cambiare in funzione del verificarsi di determinati eventi
- È importante che in questa fase i diagrammi siano semplici e comprensibili dagli utenti

UML nel processo di sviluppo

Progettazione

- I diagrammi delle classi sono disegnati usando la prospettiva software ed illustrano le classi e le loro relazioni (statiche)
- I diagrammi di sequenza sono utilizzati per documentare gli scenari più comuni
- I diagrammi di package mostrano l'organizzazione del software su larga scala
- I diagrammi di stato sono utilizzati per descrivere le classi le cui istanze hanno un ciclo di vita complesso
- I diagrammi di deployment illustrano la disposizione fisica del sistema

UML nel processo di sviluppo

Documentazione

- Dopo aver implementato il sistema, UML può essere utilizzato per produrre la sua documentazione
- I diagrammi sono utili per fornire una rappresentazione generale del sistema
- Un diagramma dei package è un'ottima mappa logica del sistema. Illustra come è suddiviso e le dipendenze tra le sue parti
- Un diagramma di deployment mostra l'aspetto fisico del sistema ad un alto livello di astrazione
- All'interno di ciascun package si può includere un class diagram che evidenzi solo le caratteristiche più importanti
- Un certo numero di diagrammi di interazione può accompagnare un class diagram per documentare le interazioni più importanti
- Ove utile, si possono includere diagrammi di stato per descrivere comportamenti complessi, frammenti di codice o diagrammi di attività per illustrare algoritmi complicati