OO design pattern

Design pattern: motivazioni

- La progettazione OO è complessa
- Progettare sw OO riusabile ed evitare (o, almeno, limitare) la riprogettazione è ancor più complesso
- I progettisti esperti non risolvono ogni problema partendo da zero ma riusano soluzioni che hanno già funzionato
- L'uso dei pattern è teso a realizzare architetture più piccole, semplici e comprensibili
- È difficile trovare un sistema OO che non usi almeno due dei pattern più comuni e sistemi di dimensioni più elevate li usano quasi tutti

Design pattern: definizioni

- "Un pattern descrive un problema che ricorre nell'ambiente e l'essenza della soluzione del problema, in modo tale che si possa riusare questa soluzione milioni di volte senza mai ripeterla in maniera identica due volte" (Christopher Alexander, 1977, riferendosi a edifici e città)
- Progetto importante e ricorrente nei sistemi OO, la cui realizzazione è orientata al riuso, inteso ad aiutare il progettista a progettare bene più in fretta
- È come un template che può essere applicato in molte situazioni diverse
- È una collaborazione comune fra oggetti di un sistema

N.B. Nei pattern si assumono disponibili le caratteristiche dei due linguaggi di programmazione prescelti (C++ e Smalltalk)

Design pattern: elementi essenziali

Elemento	Significato
Nome	È la descrizione di livello di astrazione più elevato del pattern (la sua maniglia), utile per discutere del pattern e pensare allo stesso
Problema	Descrizione astratta del contesto, delle precondizioni di applicazione del pattern e del problema da esso risolto (che può essere rappresentato a diversi livelli)
Soluzione	Descrizione a) degli elementi del progetto (classi e oggetti), delle loro relazioni, responsabilità e collaborazioni b) delle decisioni, delle alternative considerate e dei trade-off che hanno condotto al progetto (queste informazioni sono necessarie per il riuso)
Conseguenze	• Spesso sono relative al trade-off spazio-tempo • Possono toccare questioni relative al linguaggio (C++ e Smalltalk) e all'implementazione, fornendo suggerimenti ed esempi • Comprendono l'impatto sulla flessibilità, estensibilità e portabilità del sistema • Sono cruciali nella valutazione delle alternative di progettazione

Design pattern: elementi descrittivi

Elemento	Significato
Nome e classificazione	Nome (vedi sopra) + duplice classificazione (vedi oltre)
Intento (Scopo)	Problema considerato
Noto anche come (Detto anche …)	Sinonimi
Motivazione	Scenario che esemplifica l'uso del pattern
Applicabilità	• Situazioni in cui il pattern può essere applicato • Esempi di progetti scadenti in cui il pattern può essere utile • Come riconoscere situazioni e progetti di cui sopra
Struttura	Rappresentazione grafica basata su OMT (Object Modeling Technique) e diagrammi di interazione
Partecipanti	Classi e oggetti e loro responsabilità
Collaborazioni	Come i partecipanti collaborano per affrontare le loro responsabilità

Design pattern: elementi descrittivi (cont.)

Elemento	Significato
Conseguenze	• In che misura il pattern soddisfa i suoi obiettivi • Trade-off e risultati nell'uso del pattern • Quali aspetti della struttura del sistema possono essere indipendentemente modificati
Implementazione	• Tranelli, suggerimenti e tecniche di cui tenere conto nella programmazione • Questioni specifiche al linguaggio
Codice d'esempio	Frammenti di codice C++ o Smalltalk
Utilizzi noti	Almeno due esempi di applicazione del pattern in sistemi reali che afferiscono a domini diversi
Pattern correlati (altri due meccanismi di classificazione dei pattern)	• Elenco dei pattern correlati al corrente e delle sostanziali differenze reciproche • Elenco dei pattern che dovrebbero essere usati insieme al corrente

Classificazione dei pattern

- In base al proposito (purpose)
  - ✓Teso alla creazione di oggetti (creational)
  - ✓Teso alla composizione di classi/oggetti (structural)
  - ✓Teso a caratterizzare l'interazione di classi/oggetti e la distribuzione di responsabilità (behavioral)

- In base alla applicabilità (scope)
  - ✓Focalizzato principalmente sulle classi e sulle loro sottoclassi, cioè sulle loro relazioni statiche (ovvero note al momento della compilazione) di ereditarietà (class)
  - ✓Focalizzato principalmente sugli oggetti e sulle loro relazioni dinamiche (mutevoli durante l'esecuzione) (object)

Classificazione dei pattern (cont.)

Creazione di oggetti

- nei creational class pattern è in parte demandata a sottoclassi
- nei creational object pattern è in parte demandata a un altro oggetto

Composizione

- di classi mediante l'ereditarietà negli structural class pattern
- di oggetti negli structural object pattern

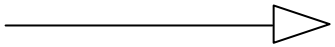
Comportamento

- che si esplica mediante l'ereditarietà (tipicamente algoritmi e flusso del controllo) nei behavioral class pattern
- che si esplica mediante la cooperazione fra oggetti (tipicamente per realizzare un compito che nessun oggetto può portare a termine da solo) nei behavioral object pattern

		Purpose		
		Creational	**Structural**	**Behavioral**
Scope	**Class**	Factory Method (107)	Adapater (class) (139)	Interpreter (243) Template Method (325)
	Object	Abstract Factory (87) Builder (97) Prototype (117) Singleton (127)	Adapater (object) (139) Bridge (151) Composite (163) Decorator (175) Facade (185) Flyweight (195) Proxy (207)	Chain of Responsibility (223) Command (233) Iterator (257) Mediator (273) Memento (283) Observer (293) State (305) Strategy (315) Visitor (331)

Notazione OMT (Object Modeling Technique)

È molto simile al diagramma delle classi UML. Ad es., sono identici nei seguenti elementi:

- relazione di generalizzazione 
- nomi di classi astratte e metodi astratti (in italico)

Alcune piccole differenze:

- Nella scatola che rappresenta la classe, prima sono elencate le operazioni, poi i dati (in UML l'ordine è inverso)
- La relazione di dipendenza ha linea continua e punta triangolare piena 
  (in UML la linea è tratteggiata e la punta biforcuta)

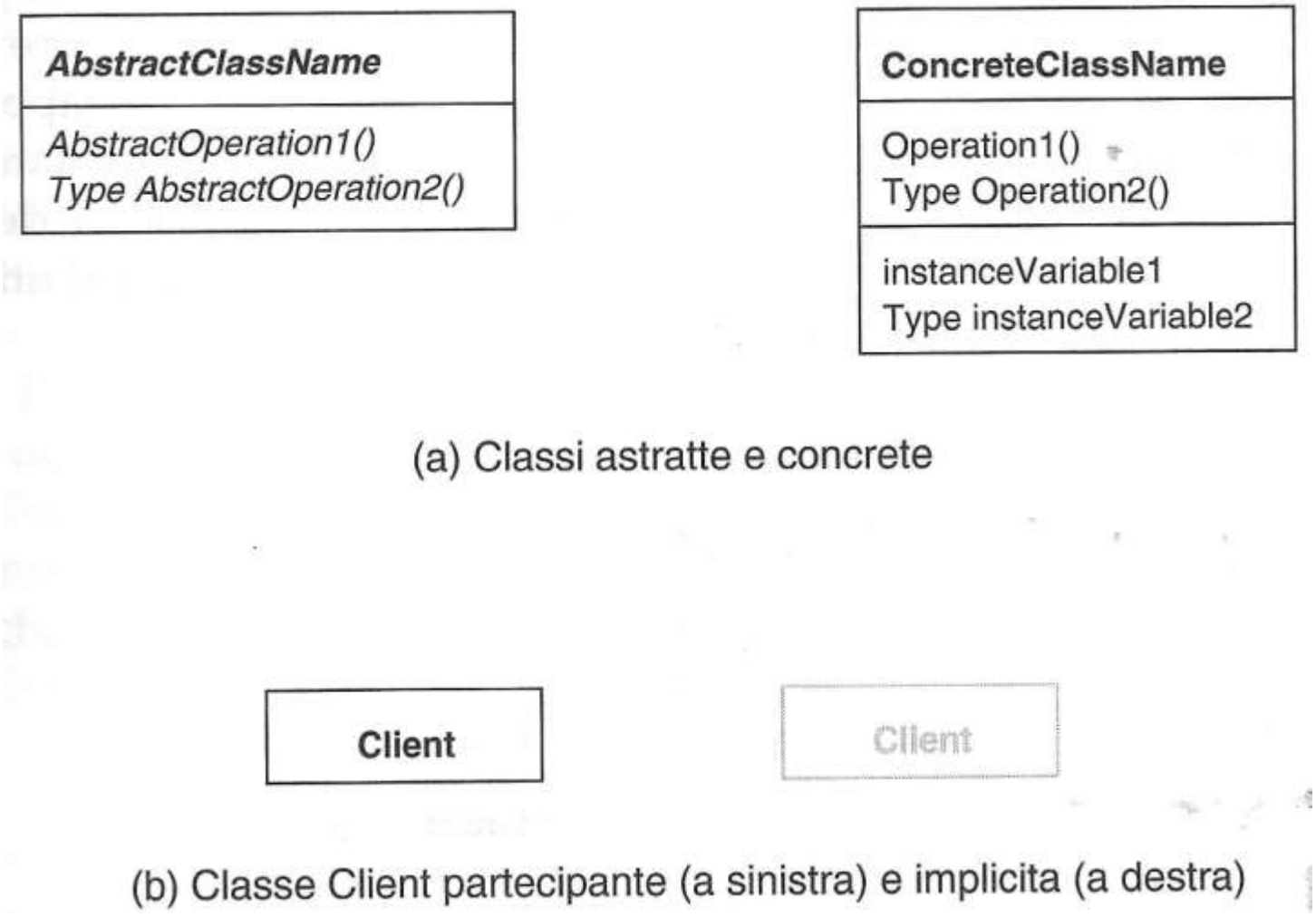
Vocabolario

In OMT i dati di una classe sono chiamati variabili di istanza (instance variables)

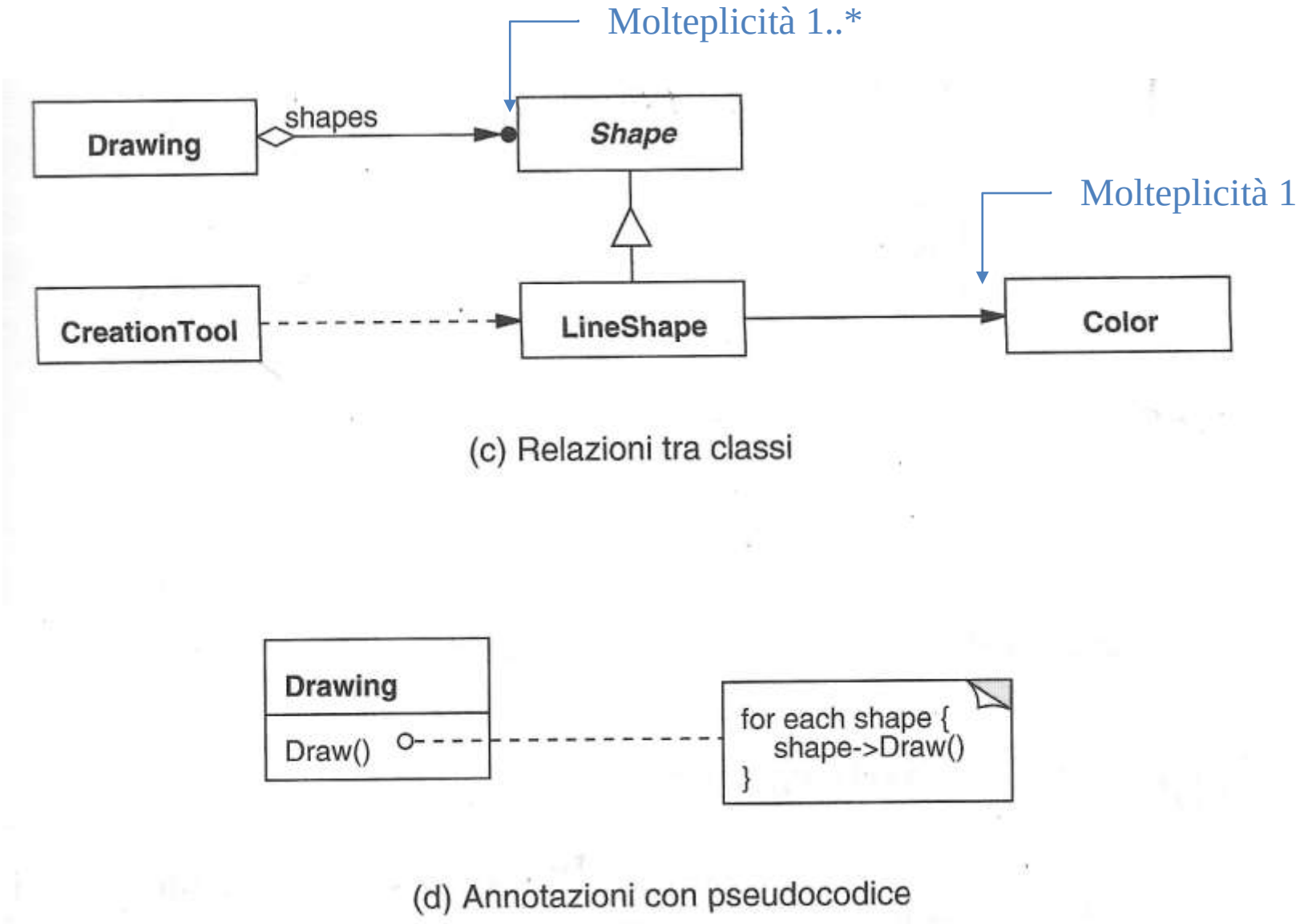
Relazioni fra oggetti

Relazione OMT	Semantica	Implementa-zione	Rappresentazione grafica OMT
Aggregazione	Quella della composizione UML: un oggetto ne possiede o è parte di un altro → i due oggetti hanno la stessa esistenza (lifetime); natura statica	Mediante le variabili membro	Simile all'aggregazione in UML (la freccia ha la punta piena anziché biforcuta)
Conoscenza (acquaintance) o relazione d'uso	Quella della dipendenza UML: un oggetto sa dell'esistenza di un altro → relazione più debole dell'aggregazione, fra i due oggetti l'accoppiamento è minore: essi possono richiedersi l'un l'altro delle operazioni ma non sono responsabili l'uno dell'altro; natura dinamica	Mediante riferimenti e puntatori	Freccia con linea continua e punta piena

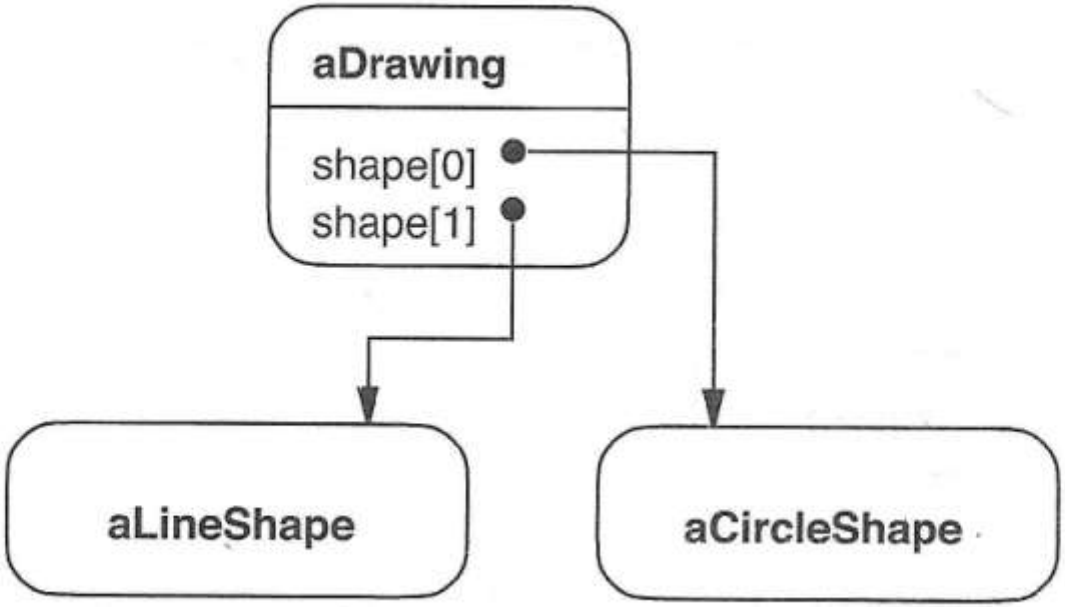
Notazione per diagrammi delle classi



Notazione per diagrammi delle classi



Notazione per diagrammi degli oggetti



Notazione per diagrammi di interazione (sequenza)