Corso di

Basi di Dati e Sistemi Informativi

Modellazione dei dati in UML

Angelo Montanari
Dipartimento di Matematica e Informatica
Università degli Studi di Udine

Introduzione

- **UML** (Unified Modeling Language): linguaggio grafico per la modellazione di applicazioni software basate sul paradigma orientato agli oggetti.
- Ambito: **Ingegneria del Software.**
- Viene utilizzato per modellare, per mezzo di opportuni **diagrammi**, i dati, le operazioni, i processi e le architetture di un'applicazione software.

UML e basi di dati - 1

- UML quale **alternativa** al modello ER per la rappresentazione concettuale dei dati.

- I **diagrammi delle classi**: descrivono le classi di oggetti di interesse per l'applicazione e le loro relazioni.

- In base al principio dell'incapsulamento, proprio del paradigma o.o., possono essere rappresentati anche gli **aspetti procedurali** di un'applicazione.

UML e basi di dati - 2

- **Alcuni** dei **costrutti** del modello ER (ad esempio, gli identificatori interni ed esterni) **non** sono **previsti** in UML. In questo caso si ricorre a notazioni non standard.

- **Nota bene**: il diagramma delle classi di un'applicazione e lo schema concettuale della sua componente base di dati non vanno confusi.

UML: le origini

- Proposto a metà degli anni ’90 quale **formalismo unificante** per la modellazione o.o. di applicazioni software, è stato standardizzato sotto l’egida dell’Object Management Group (OGM).

- UML offre una molteplicità di diagrammi, corredati da una descrizione testuale della loro semantica: **molteplicità di viste** della medesima applicazione.

UML: il modello dell'applicazione

- L'insieme dei diagrammi definisce il **modello dell'applicazione** (controparte dello schema ER): UML è un metamodello per la descrizione di modelli di applicazioni software.

- Nella sua versione corrente UML prevede un certo numero di **diagrammi fondamentali**: diagramma delle classi, degli oggetti, dei casi d'uso, di sequenza, di comunicazione, delle attività, degli stati, dei componenti e di distribuzione dei componenti.

UML: I diagrammi principali - 1

- Diagramma delle **classi**: descrive le caratteristiche statiche e dinamiche delle componenti (**classi**) e delle loro relazioni (**associazioni**).

- Diagramma degli **oggetti**: rappresentazione delle possibili istanze delle classi (**oggetti**) e dei loro collegamenti.

- Diagramma dei **casi d'uso**: modalità di utilizzo del sistema da parte di persone/altri sistemi (**attori**) e interazioni tra sistema e attori.

UML: I diagrammi principali - 2

- Diagramma di **sequenza**: ordinamento temporale di messaggi (invocazione di **metodi**) scambiati tra i diversi oggetti dell'applicazione (funzione analoga ha il diagramma di **comunicazione**).

- Diagramma delle **attività**: comportamento dinamico di un processo dell'applicazione in termini dei flussi di attività da svolgere.

- Diagramma degli **stati**: descrive il ciclo di vita di un oggetto dell'applicazione attraverso gli stati che può assumere.

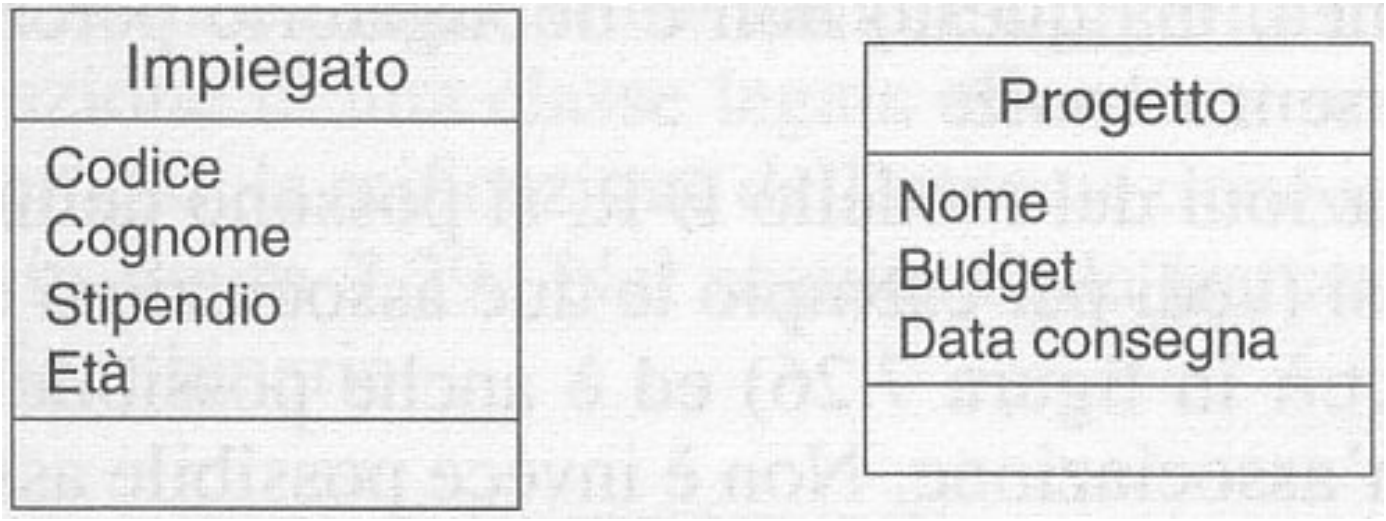
UML: I diagrammi principali - 3

- Diagramma dei **componenti**: descrive l'organizzazione delle componenti fisiche del sistema (file, moduli, ..) e le loro dipendenze.

- Diagramma di **distribuzione dei componenti**: descrivere la dislocazione dei nodi hardware del sistema e le loro associazioni.

Uso del diagramma delle classi per la modellazione dei dati

1. Le **classi**:controparte delle entità del modello ER.

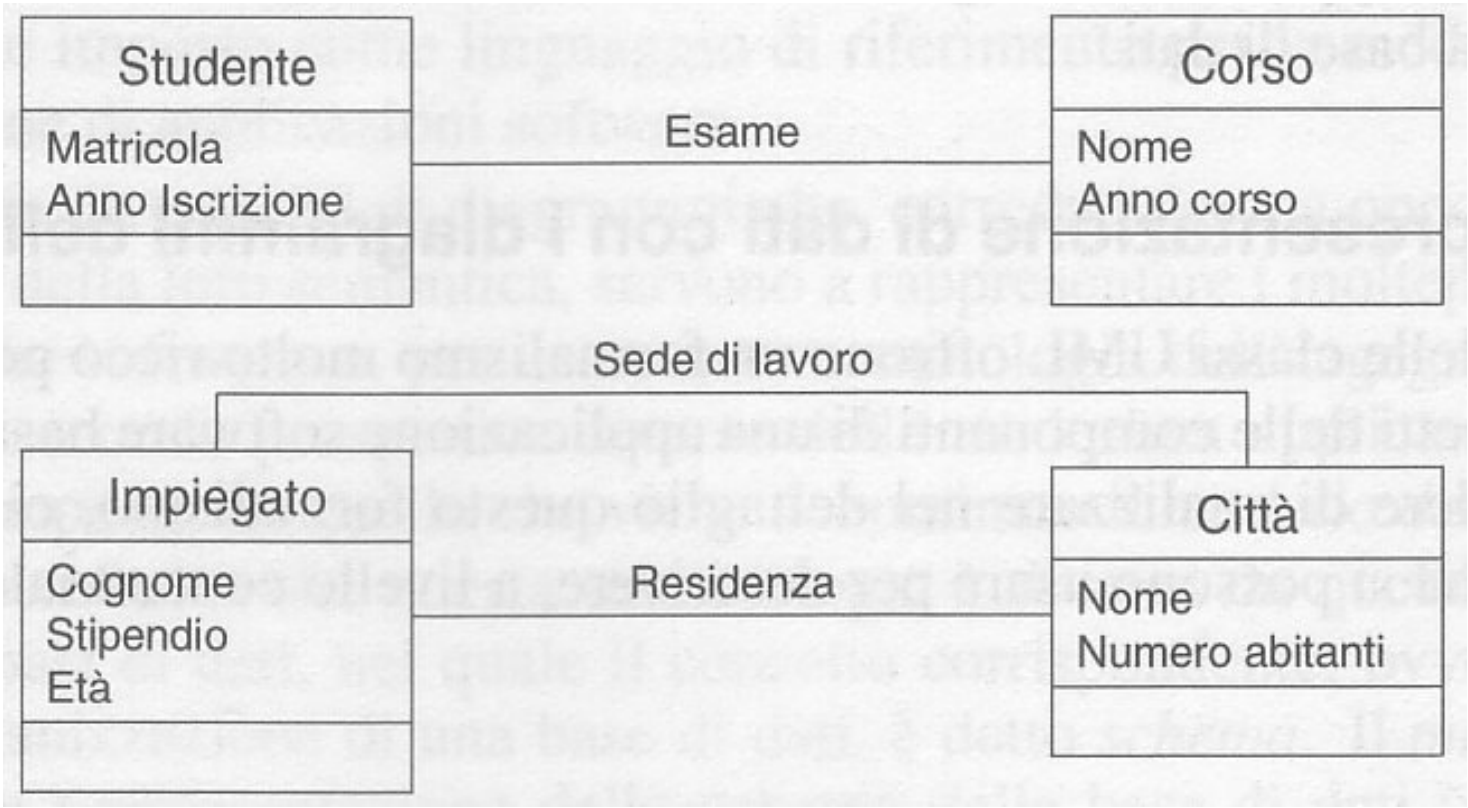


Le classi

- Ogni classe viene rappresentata con un rettangolo contenente il **nome** della classe (in alto), gli **attributi** e, a differenza del modello ER, i **metodi** (esempio, il metodo AggiornaStipendio, che consente di incrementare lo stipendio di una certa percentuale).
- Agli attributi vengono associati i **domini** e **proprietà** di interesse, quali molteplicità, vincoli e visibilità (non rilevante).
- **Non** sono ammessi **attributi compositi** (si possono modellare come domini strutturati).

Le associazioni

- Le **associazioni**: controparte delle relazioni del modello ER.

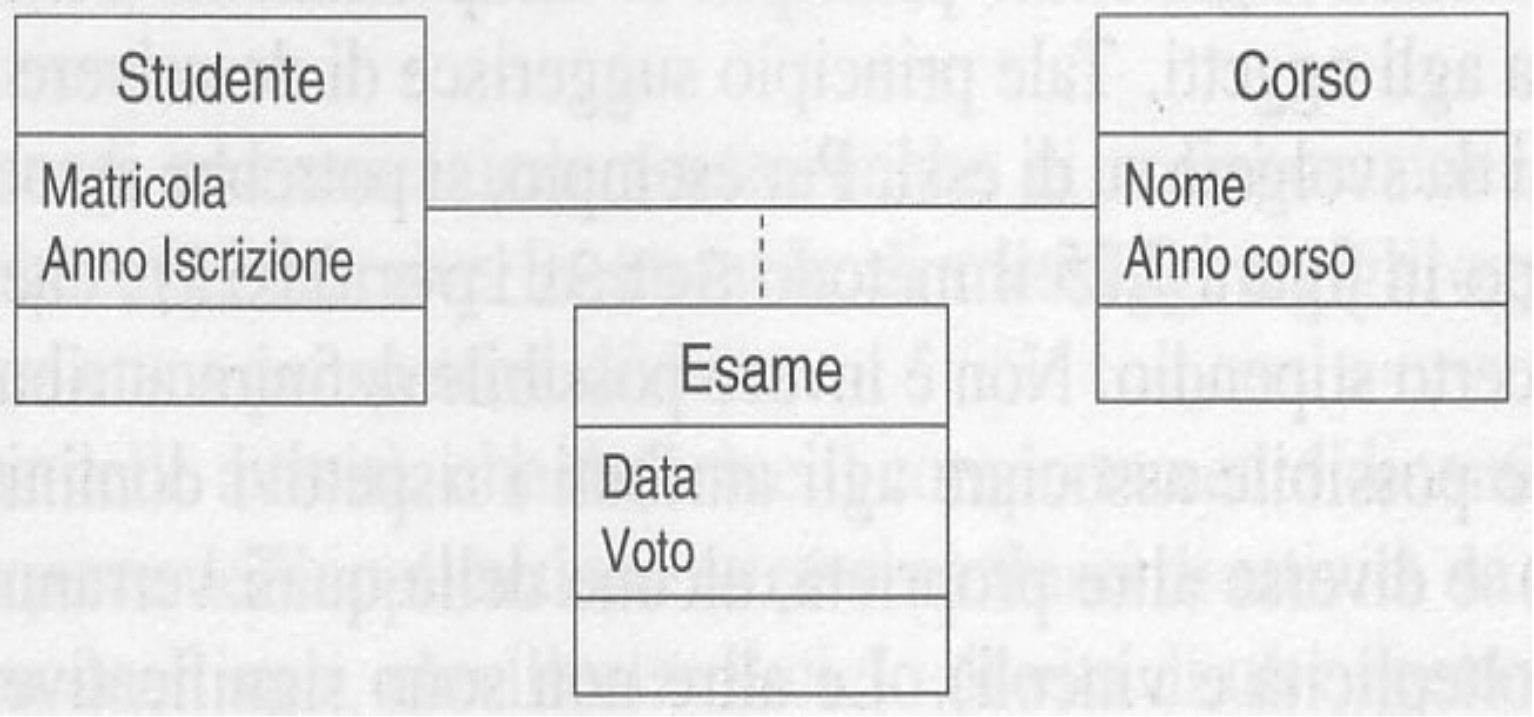


Caratteristiche delle associazioni

- Le associazioni binarie si rappresentano con **linee** che collegano due classi (eventuale nome associato alla linea).
- Vi possono essere **più associazioni** tra le medesime classi e si possono assegnare dei **ruoli** alle classi coinvolte.
- **Non** si possono assegnare **attributi** alle associazioni. Devono essere usate **classi di associazione** (descrivono proprietà di un'associazione e sono legate ad essa per mezzo di una linea tratteggiata).

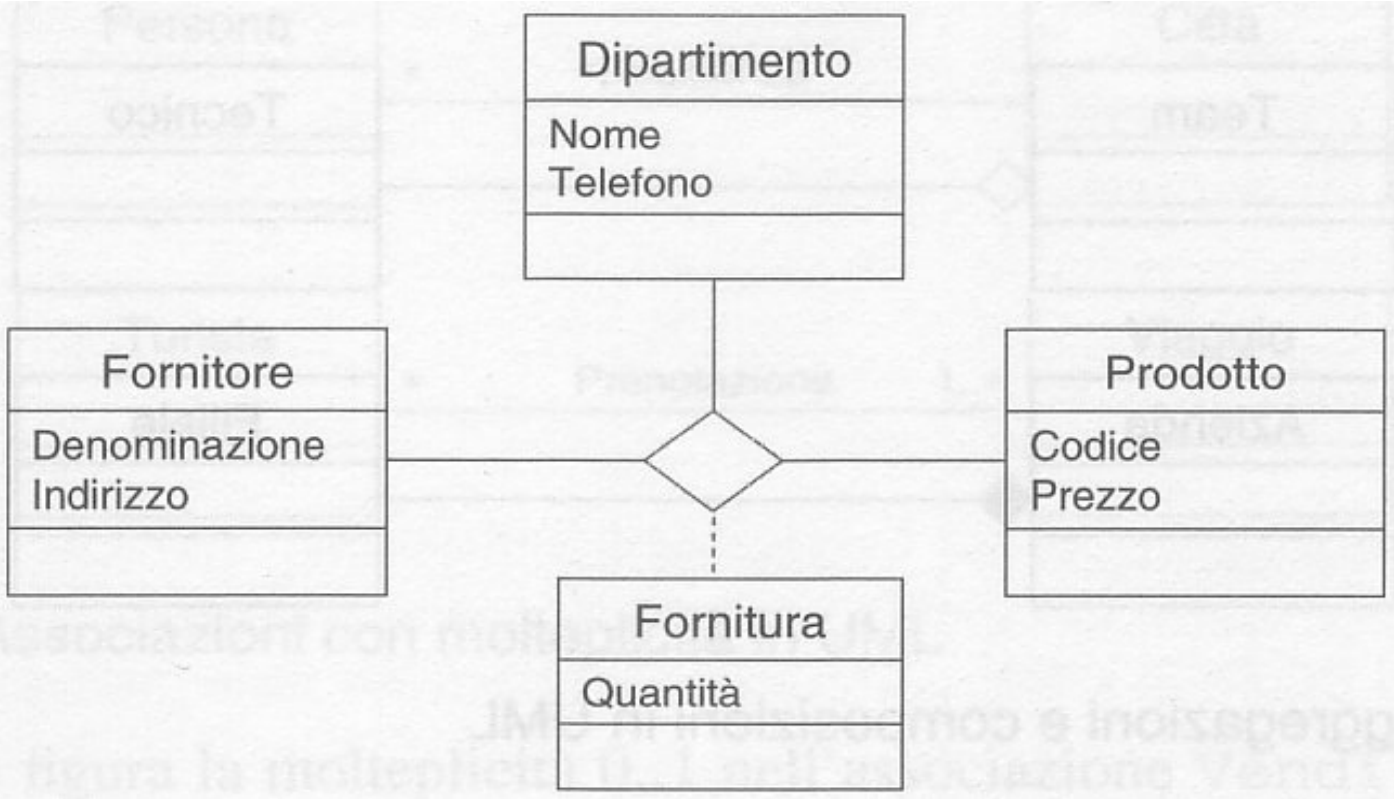
Esempio di classe di associazione

- *Osservazione*: non occorre associare un nome all'associazione.



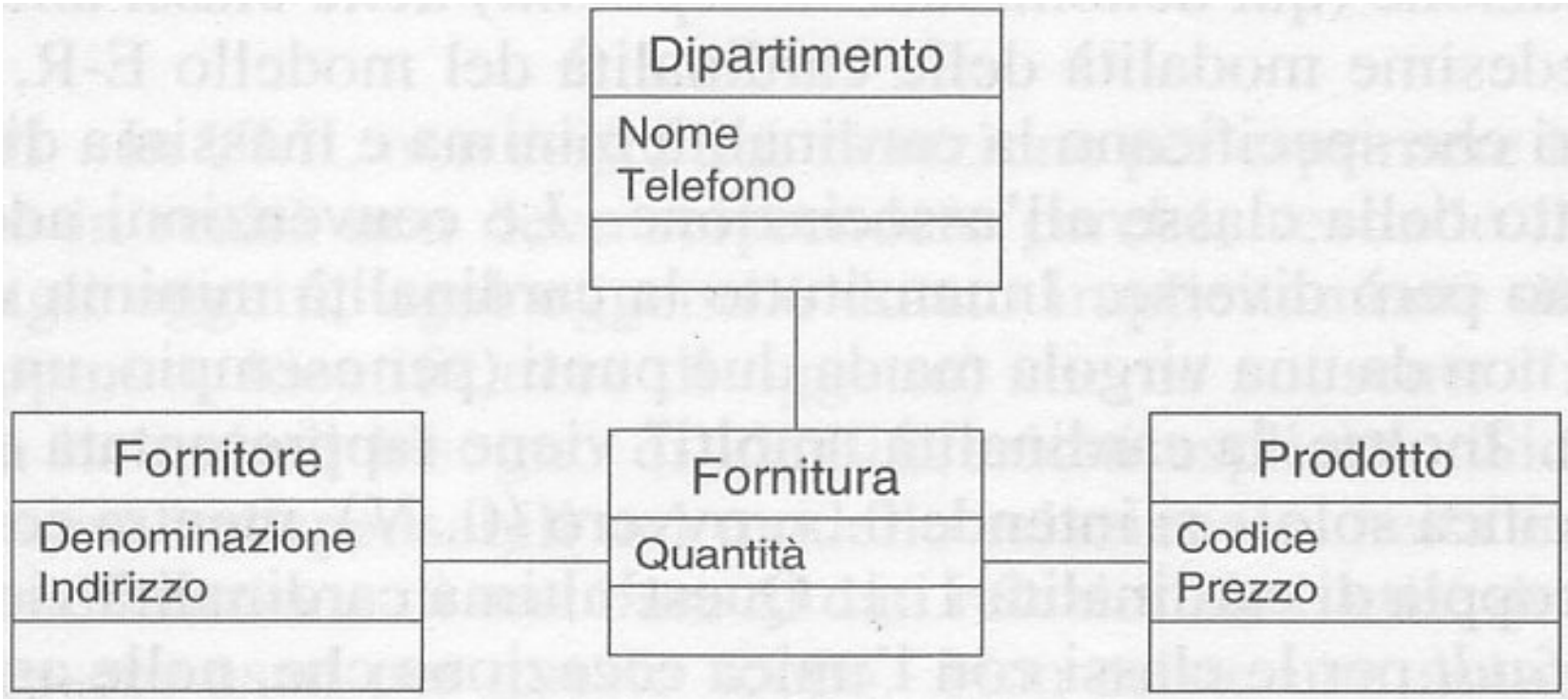
Le associazioni n-arie

- Per le **associazioni n-arie** si usa la notazione del modello ER (fornitura è una classe di associazione).



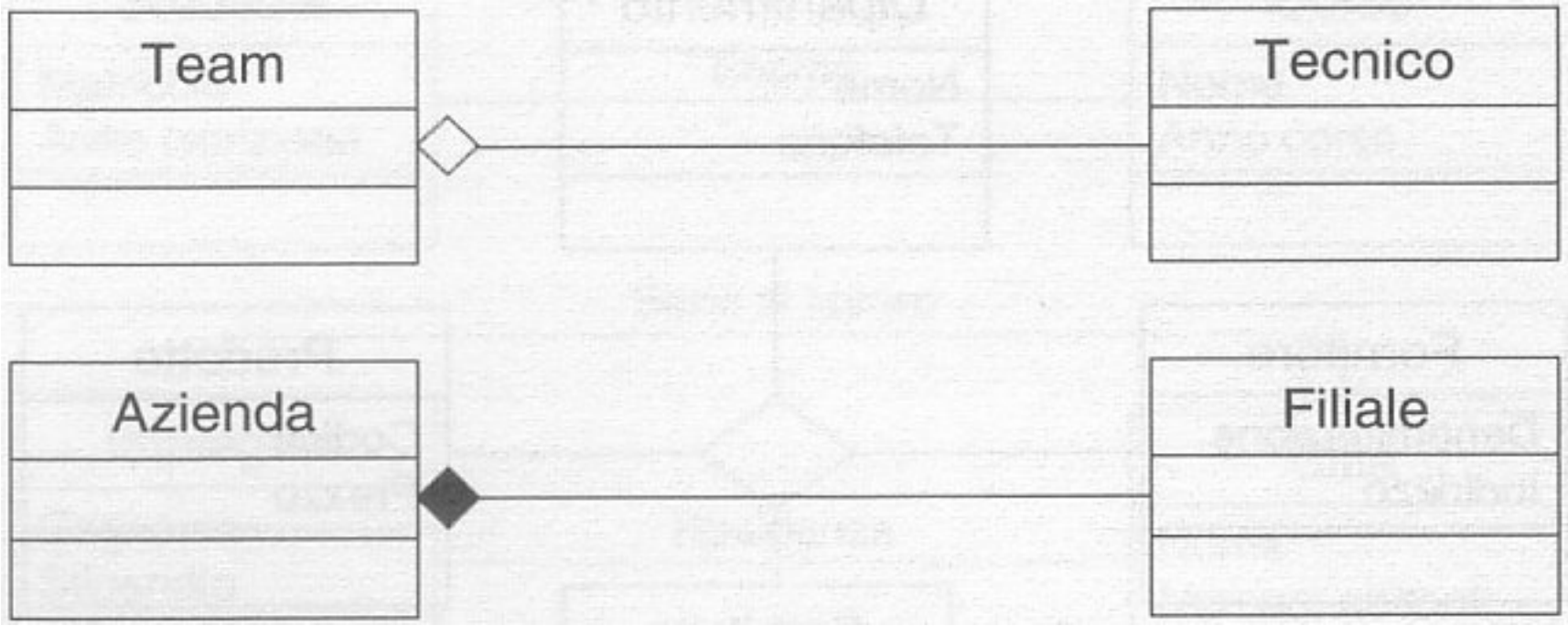
Associazioni come classi

- Come nel caso del modello ER, le relazioni di grado superiore al secondo si possono eliminare tramite **reificazione**.



Proprietà delle associazioni - 1

- Con una freccia si può indicare un verso privilegiato di **navigabilità** di un'associazione.
- Si possono specificare associazioni che sono **aggregazioni** di concetti (composto / componenti).



Proprietà delle associazioni - 2

- Rombo **bianco** = un oggetto della classe parte può esistere senza dover appartenere ad un oggetto della classe composta.
- Rombo **nero** = non vi può essere componente senza composto (**composizione**).

Vincoli di integrità: le molteplicità - 1

- E' possibile indicare la cardinalità (detta **molteplicità**) di partecipazione delle classi alle associazioni attraverso una coppia (min,max) come nel modello ER. Tale molteplicità si può associare anche agli attributi.

  Le **convenzioni** sono diverse:

- (x,y) diventa x..y e N diventa * (esempio: (1,N) diventa 1..*)
- * sta per 0..N
- 1 sta per 1..1 (default, a parte il caso delle aggregazione ove il default per la classe aggregante è *; può essere omesso).

Vincoli di integrità: le molteplicità - 2

- **Differenza principale** a livello di notazione: le cardinalità di partecipazione minima e massima di una classe ad un'associazione non vengono riportate accanto alla classe stessa, ma accanto all'altra classe che partecipa alla relazione.

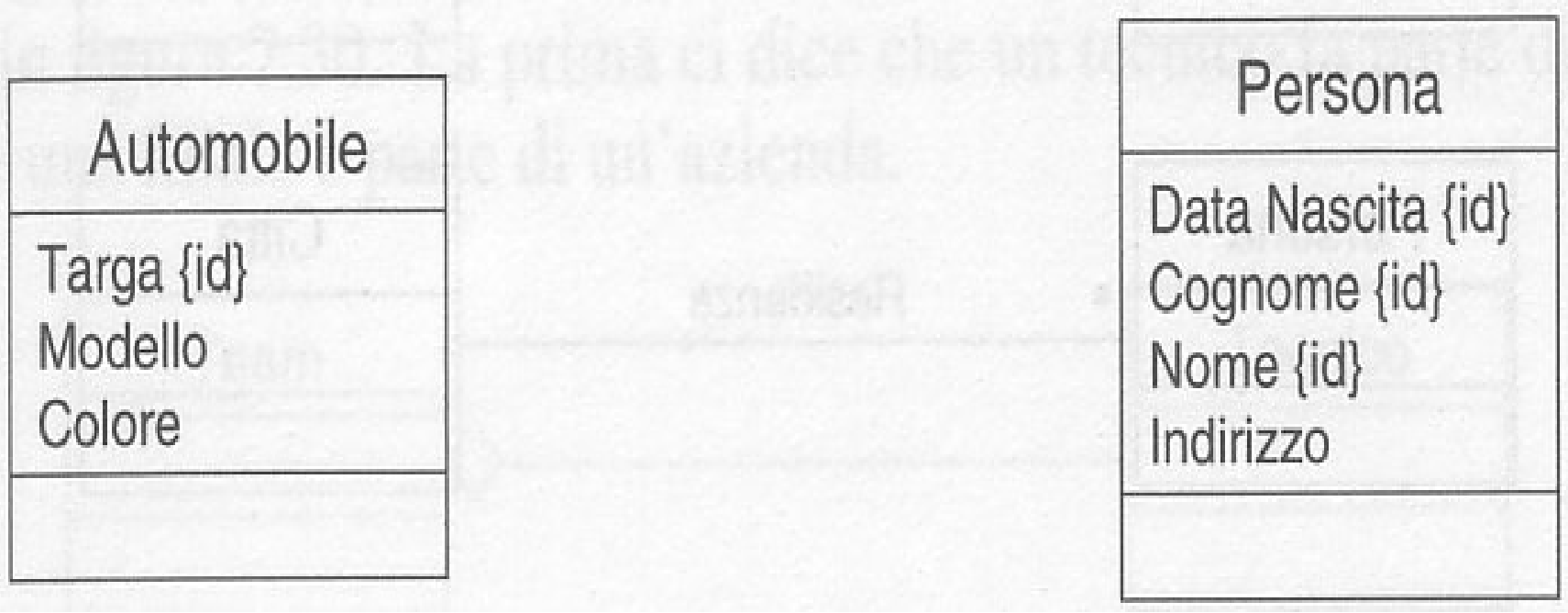
Identificatori interni - 1

- **Non** vi è una **notazione standard** per gli identificatori (interni) delle classi (UML è un modello basato sull'identità degli oggetti).
- Una possibilità: utilizzare un **vincolo utente.**

  UML consente di definire vincoli di integrità su associazioni e su attributi specificandoli tra parentesi graffe vicino all'elemento oggetto del vincolo. Vi sono alcuni vincoli predefiniti, nessuno riconducibile alla nozione di identificatore. Si possono definire vincoli propri, detti vincoli utente.

Identificatori interni - 2

- Un identificatore viene specificato col vincolo utente {id}. Se l'identificatore è costituito da più attributo, il vincolo viene associato a ciascun attributo (solo un identificatore per classe).



Identificatori esterni - 1

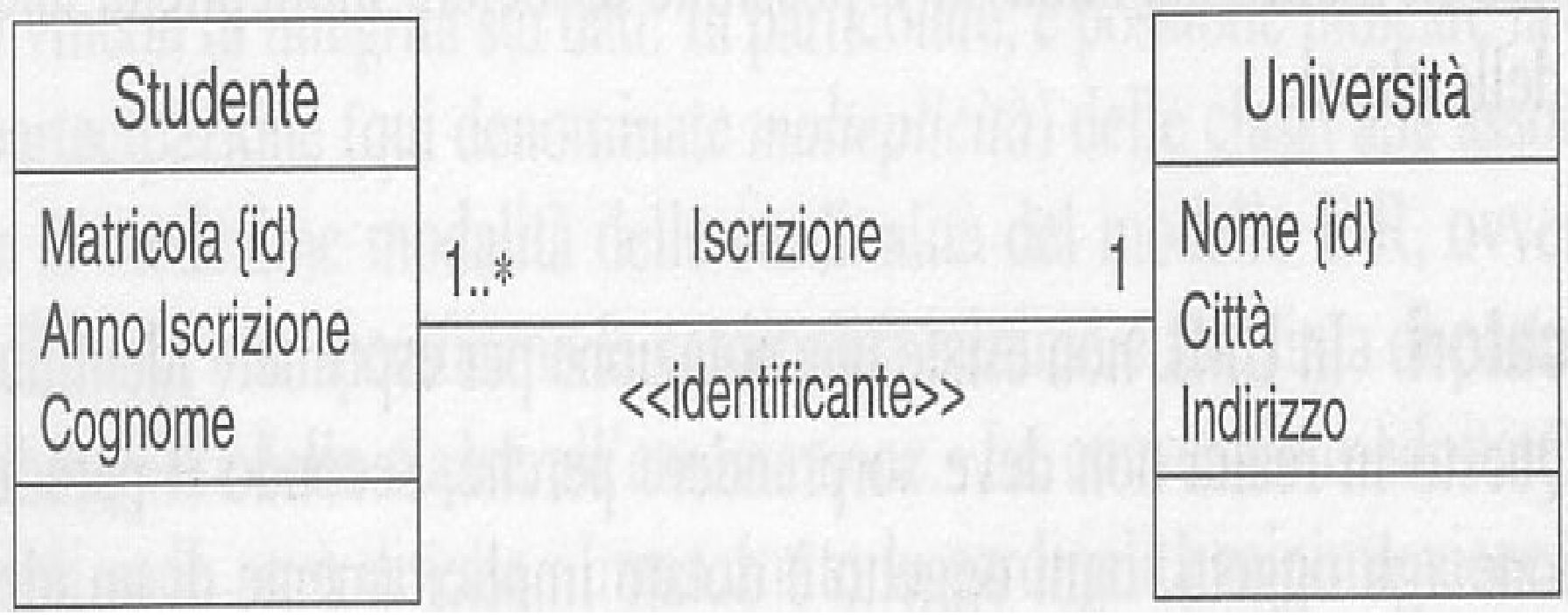
- Per gli identificatori esterni si ricorre agli stereotipi.

  Uno **stereotipo** viene utilizzato da UML per estendere i costrutti base quando si vuole modellare un concetto che non può essere modellato con i costrutti di base. Di norma, gli stereotipi si ottengono per estensione da qualche costrutto di base. Si possono definire anche **stereotipi personalizzati**.

  Gli stereotipi sono rappresentati da un nome racchiuso tra i simboli << e >>.

Identificatori esterni - 2

- *Esempio*: uso di uno stereotipo per modellare una relazione identificante (qualora ci fossero ambiguità sulla classe da identificare, si potrebbe sfruttare il nome dell'identificatore).



Le generalizzazioni - 1

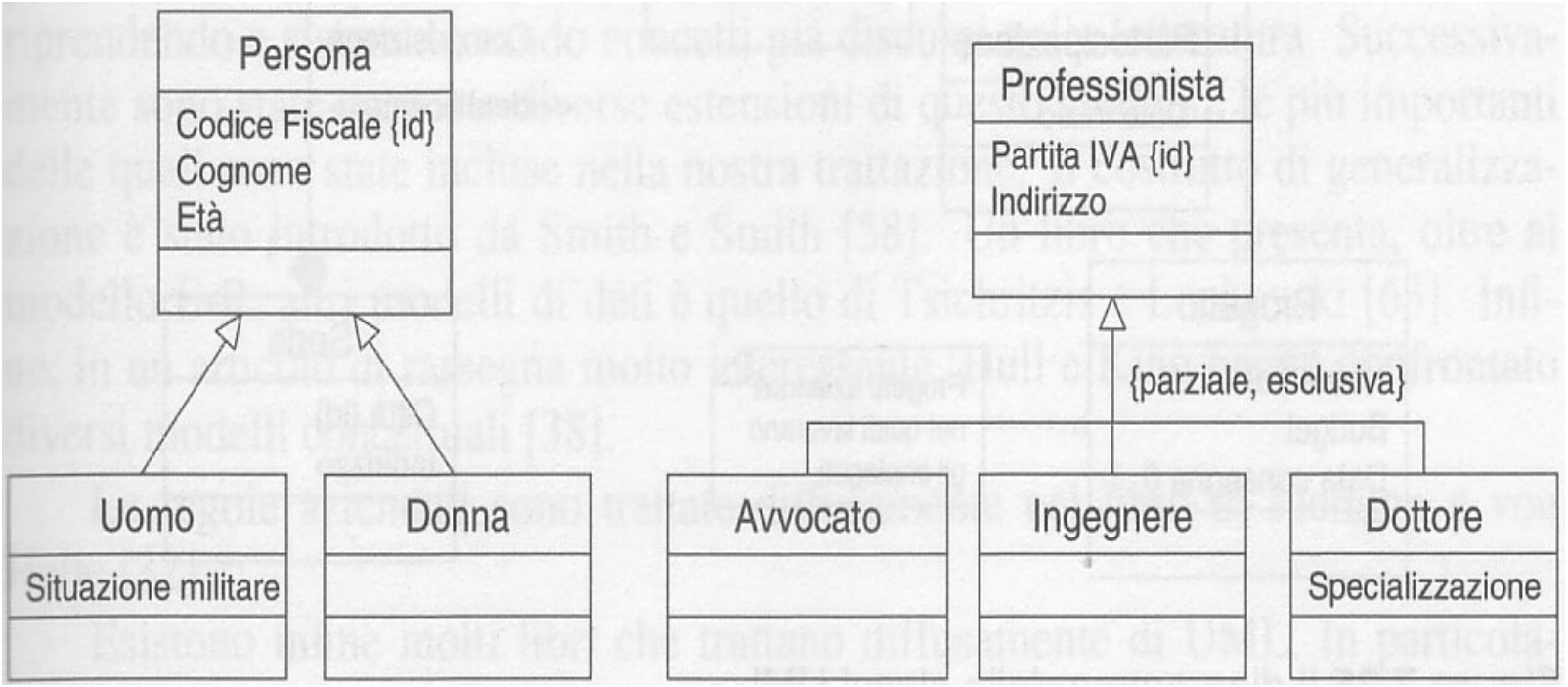
- Definizione simile a quella del modello ER.

- **Differenze**: tipicamente ogni figlio viene collegato al genitore da una freccia distinta, ma tali frecce si possono unire.

- Le **proprietà delle generalizzazioni** possono essere rappresentate con appositi vincoli.

Le generalizzazioni - 2

- Due *esempi* di generalizzazione, con indicazione, nella seconda, delle proprietà della generalizzazione.



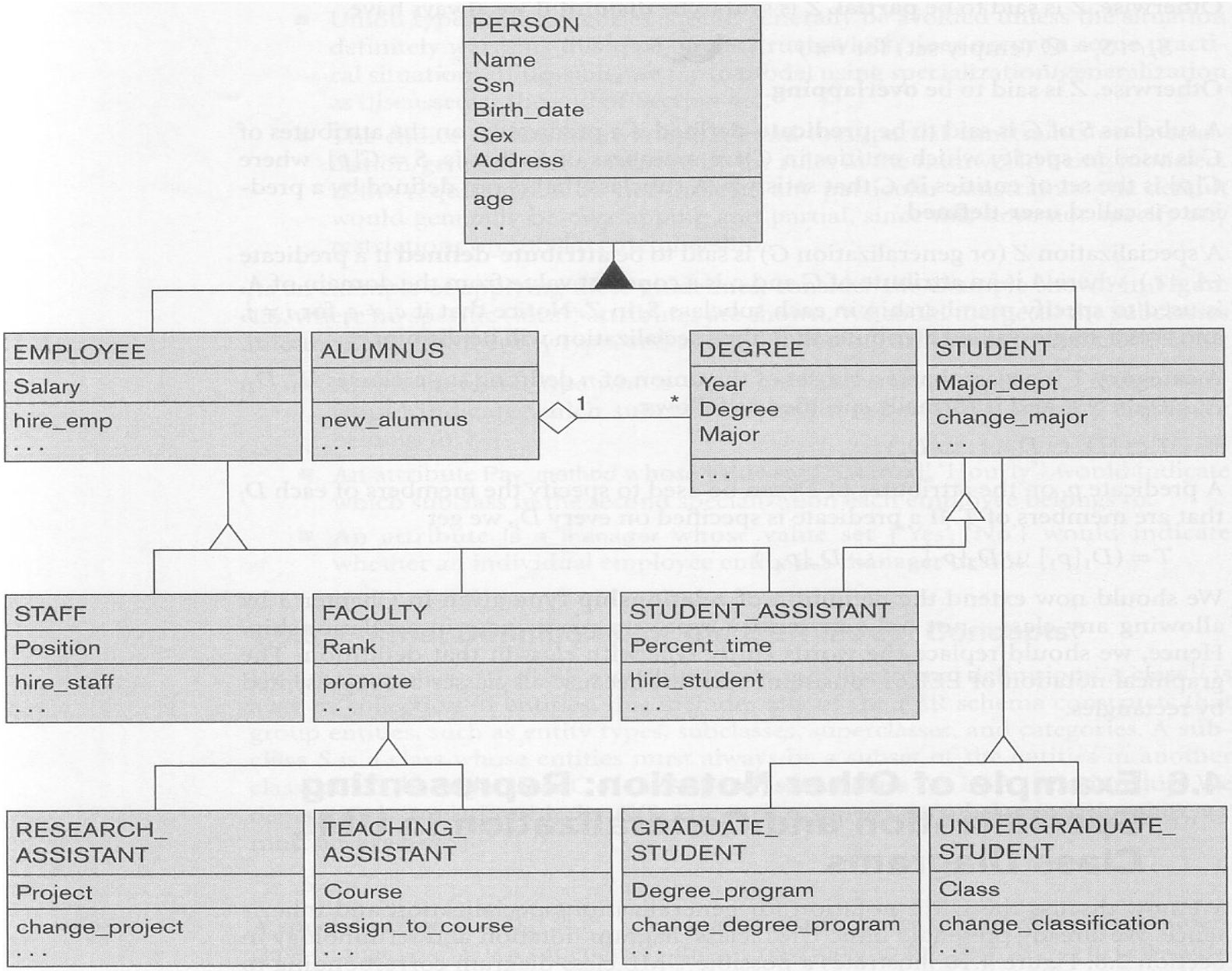
Una notazione alternativa

- Una notazione alternativa per la rappresentazione delle proprietà delle generalizzazioni è riportata nel prossimo lucido.

  Triangolo **bianco** = generalizzazione con figli disgiunti

  Triangolo **nero** = generalizzazioni con figli che possono sovrapporsi

Una gerarchia di generalizzazioni

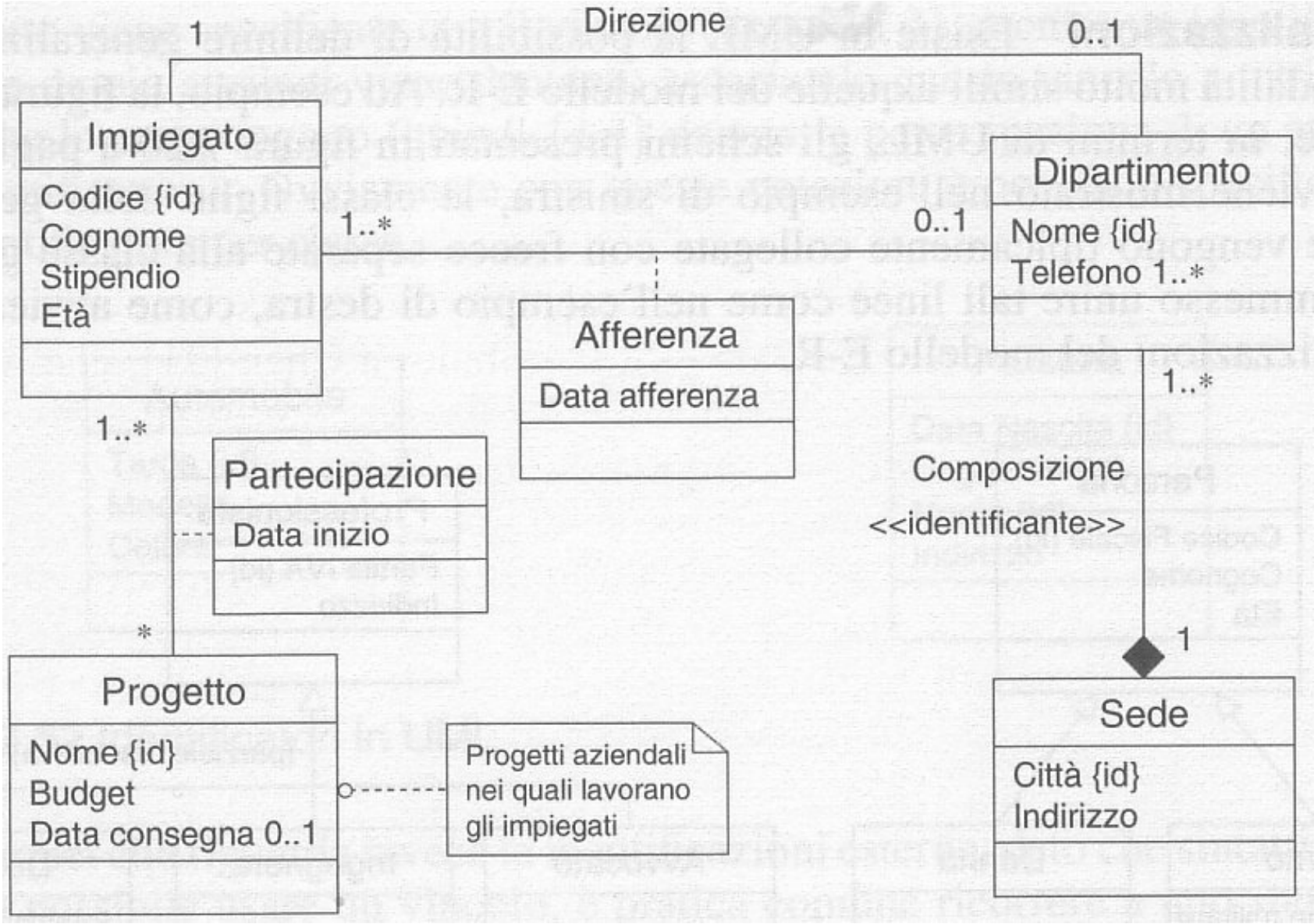


Uso delle note

- E’ possibile documentare un diagramma UML attraverso delle note.

  Una **nota** è un semplice commento testuale riportato nel diagramma all’interno di un rettangolo con l’angolo superiore destro ripiegato, eventualmente collegato all’elemento cui fa riferimento attraverso una linea tratteggiata (tale collegamento non è necessario).

Un esempio completo



Un altro esempio