UML 2.0: CLASS DIAGRAM + OCL

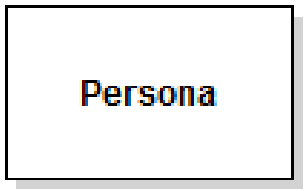
Ingegneria del Software a.a. 2023-24

Agenda

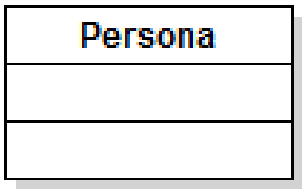
- Una precisazione su UML
- Interfacce (in UML)
- Come aggiungere dei **vincoli** a un diagramma delle classi UML
  - Object Constraint Language (OCL)
- Demo Visual Paradigm
- Esempio OCL
- Esercizi sul Class Diagram UML
  - Simili ad esercizio di esame
  - Registrazione disponibile su AW

PRECISAZIONE SUL SIMBOLO RETTANGOLO

Sono equivalenti? Rappresentano la stessa informazione? **No!**



Esprime il fatto che **omettiamo di rappresentare attributi e operazioni**, ma non è detto che la classe ne sia sprovvista

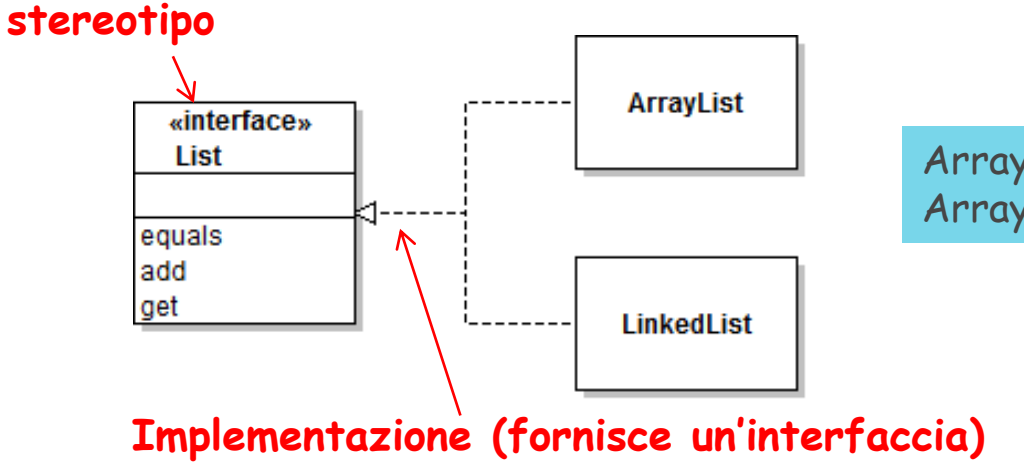


Esprime il fatto che la classe Persona **non ha ne attributi ne operazioni**

L'omissione di un compartimento non permette di effettuare alcuna deduzione. In altre parole se la sezione degli attributi non viene visualizzata, ciò non significa assolutamente che la classe ne sia sprovvista. Tale condizione è rappresentata mostrando il compartimento degli attributi vuoto

INTERFACCIA

- Il termine ***interfaccia*** è usato in due modi:
  - è **l'insieme delle operazioni visibili all'esterno** degli oggetti che sono istanze di quella classe
  - è **un entità "simile" ad una classe**, ma è priva di implementazione (ha solo operazioni pubbliche)
- Una o più classi possono **fornire** l'implementazione dell'interfaccia



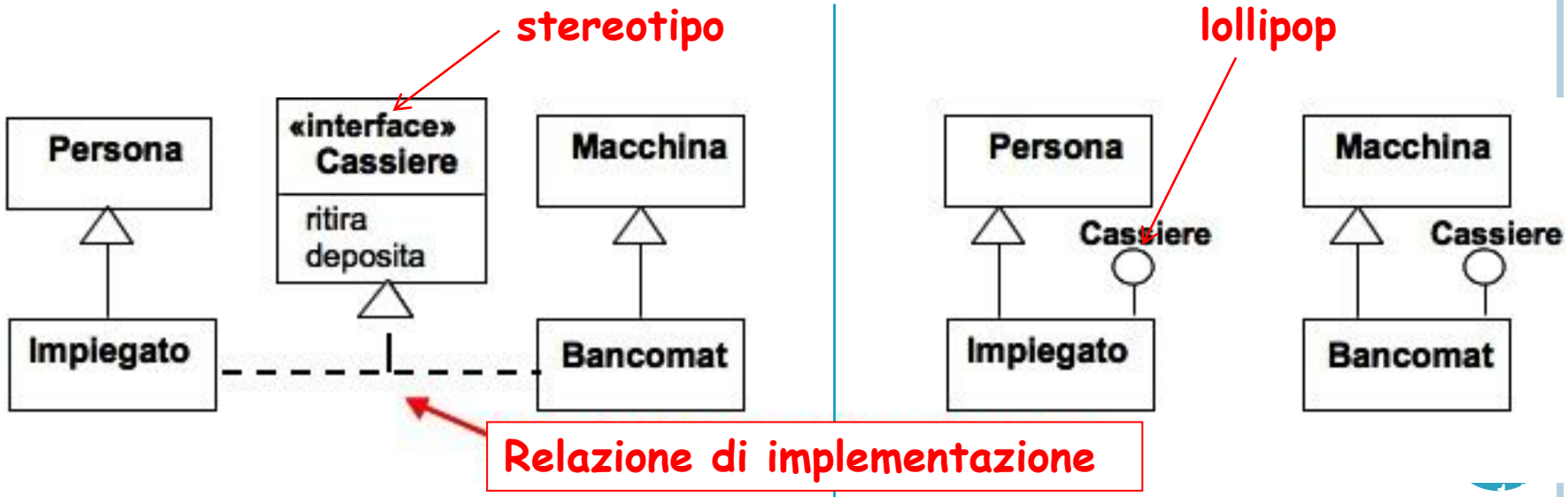
ArrayList implementa List
ArrayList fornisce un interfaccia di tipo List

implements in Java

- Una classe **richiede** un'interfaccia se necessita di una classe che l'implementa per funzionare

INTERFACCIA: ESEMPIO

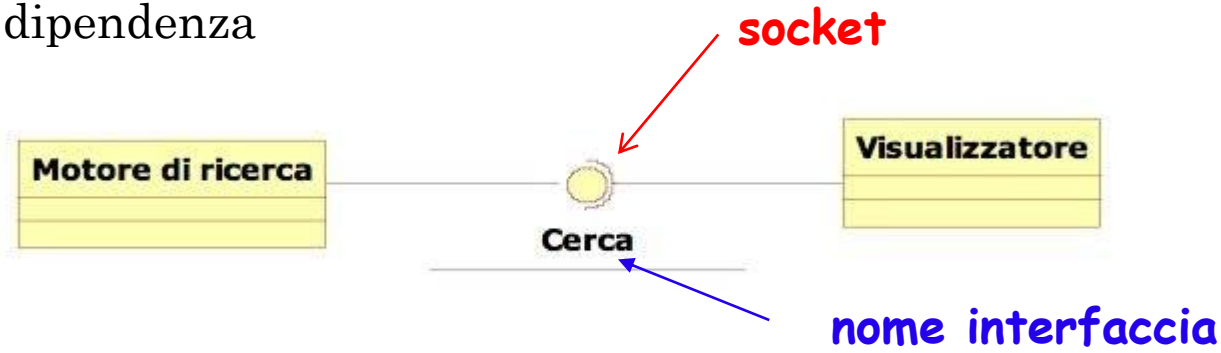
- Impiegato e Bancomat **forniscono** l'interfaccia Cassiere
  - Operazioni **ritira** e **deposita** contante
- Due modi **possibili** di rappresentarlo in UML
  - Relazione di implementazione (realizzazione) o lollipop



Sono equivalenti le notazioni?

"LOLLIPOP"

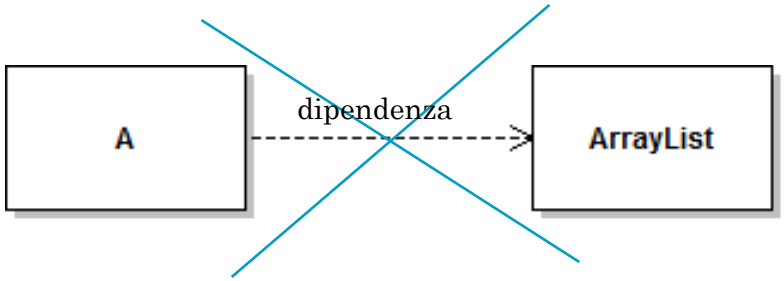
- Una notazione molto utilizzata per le interfacce è quella a "Lollipop":
  - La 'pallina' (**lollipop**) rappresenta l'interfaccia **esposta/fornita** da una classe
  - Il 'semicerchio' (**socket**) rappresenta l'interfaccia **richiesta**
    - quindi è una dipendenza



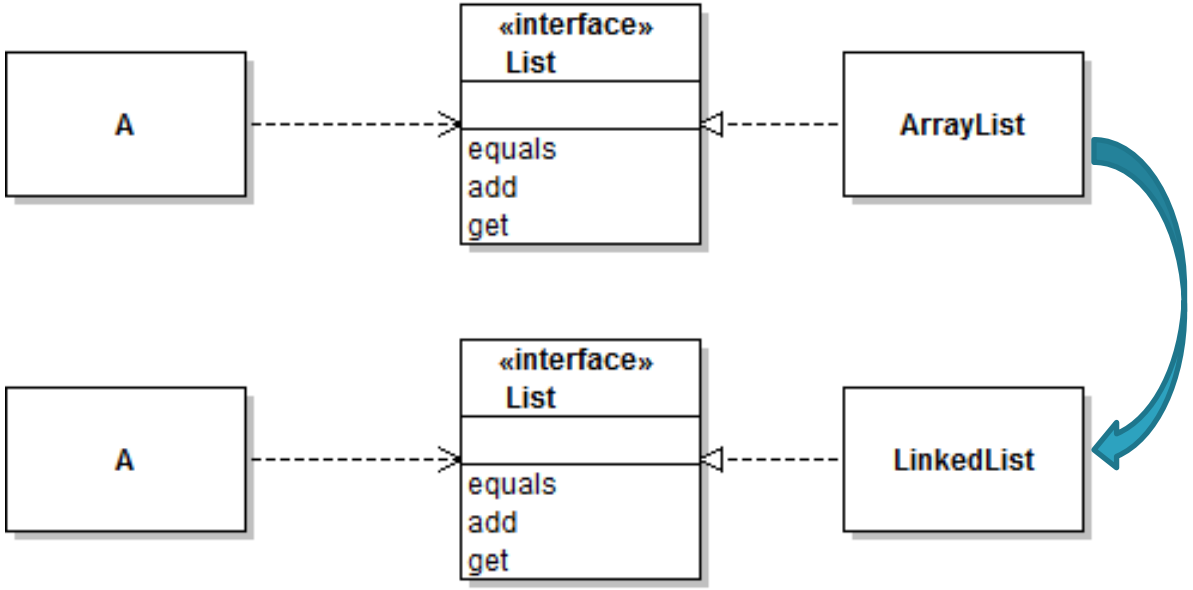
- **Esempio:**
  - un motore di ricerca fornisce la possibilità di accedere al proprio servizio "Cerca" tramite un'interfaccia
  - La classe visualizzatore utilizza il servizio "Cerca"

Perché usare le interfacce?

- Per separare l'implementazione di una classe da quella che è l'interfaccia vera e propria
  - Così diminuisce l'accoppiamento tra classi ...



Anche per avere ereditarietà multipla in linguaggi che non la supportano



Si cambia l'implementazione di List senza modificare il codice di A

https://www.gsd.inesc-id.pt/~romanop/files/TdP/Interfacce.pdf

ESEMPIO

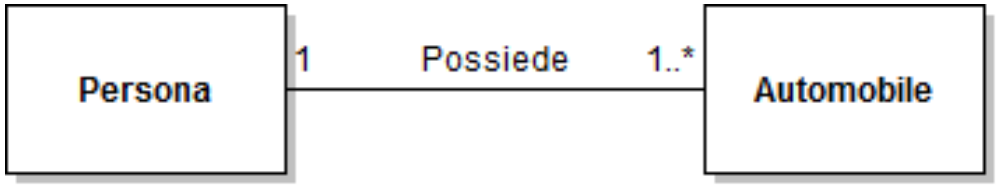
```
import java.util.*;
public class ListExample1{
public static void main(String args[]){
 //Creating a List
 List<String> list=new ArrayList<String>();
 //Adding elements in the List
 list.add("Mango");
 list.add("Apple");
 list.add("Banana");
 list.add("Grapes");
 //Iterating the List element using for-each loop
 for(String fruit:list)
  System.out.println(fruit);

}
}
```

Linked List

Regole di vincolo

- Un diagramma delle classi definisce dei vincoli

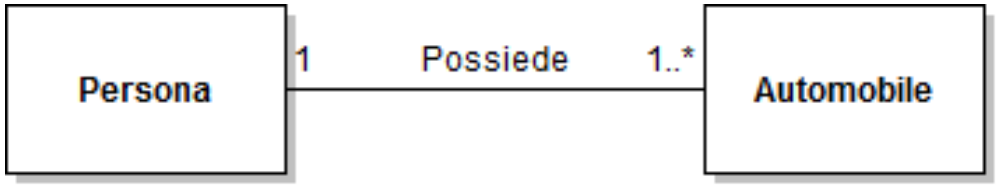


Un automobile è posseduta solo da una persona!

- Associazioni, attributi, generalizzazione sono **costrutti base per esprimere vincoli** ma non possono bastare a rappresentarli tutti!
  - In particolare i + complessi (vedi dopo)
- UML permette di specificare **ulteriori vincoli** ad un class diagram usando:
  - Linguaggio naturale (o linguaggio di programmazione)
  - **OCL (Object Constraint Language)**
    - Fa parte di UML
    - Linguaggio di specifica formale
    - Basato su **logica del primo ordine**

REGOLE DI VINCOLO

- Un diagramma delle classi definisce dei vincoli



Un automobile è posseduta solo da una persona!

- Associazioni, attributi, generalizzazione sono **costrutti base per esprimere vincoli** ma non possono bastare a rappresentarli tutti!
  - In particolare i + complessi (vedi dopo)
- UML per
  - Ling
  - **OCL (Obj**
    - Fa parte
    - Ling
    - Basato su **logica del primo ordine**

'Tutti gli uomini sono mortali'

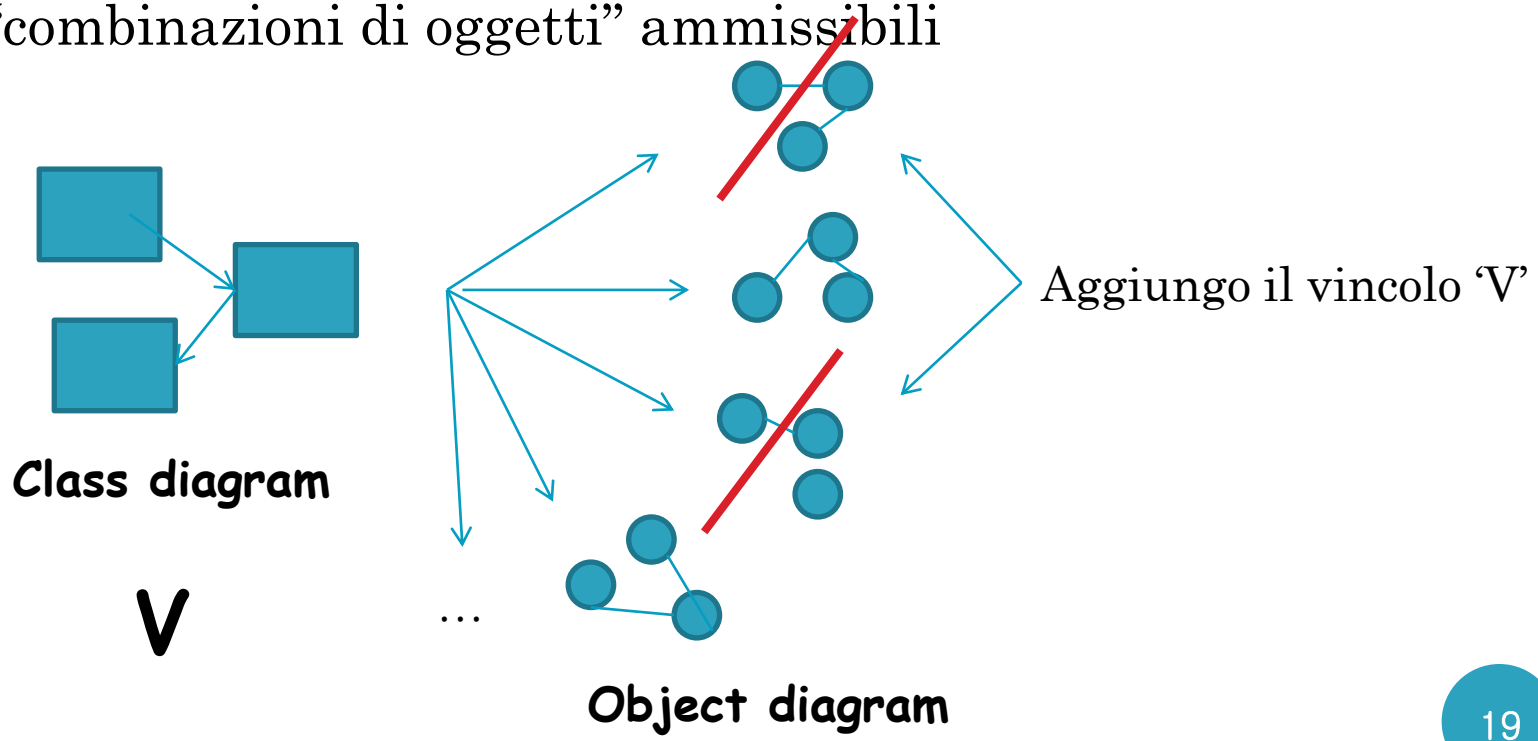
$\forall x\ (UOMO(x) \Rightarrow MORTALE(x))$

'Lassù qualcuno mi ama'

$\exists x\ (lassu'(x) \wedge ama(x, Io))$

REGOLE DI VINCOLO: SIGNIFICATO

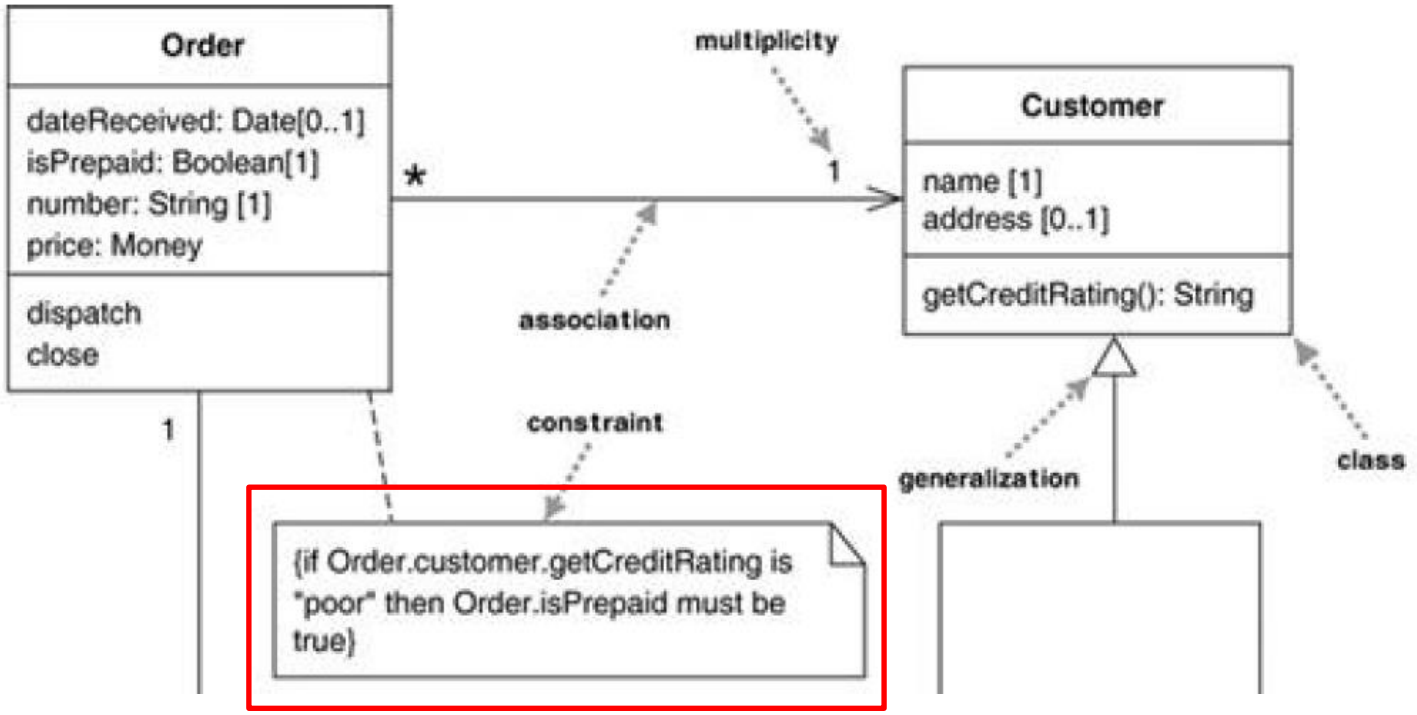
- Cosa vuole dire aggiungere **un vincolo** ad un class diagram?
  - Sostanzialmente **ridurre il numero** di "combinazioni di oggetti" ammissibili



Il credit scoring è un metodo statistico che consente di valutare l'affidabilità creditizia e la solvibilità di una determinata persona

COME SI RAPPRESENTANO I VINCOLI?

- **Sono indicati tra parentesi graffe**
  - Es. {moglie e marito non devono essere parenti}
- **Sono associati ad una classe tramite una nota**



Object Constraint Language (OCL)

- Basato su: **logica del primo ordine**

'Tutti gli uomini sono mortali'

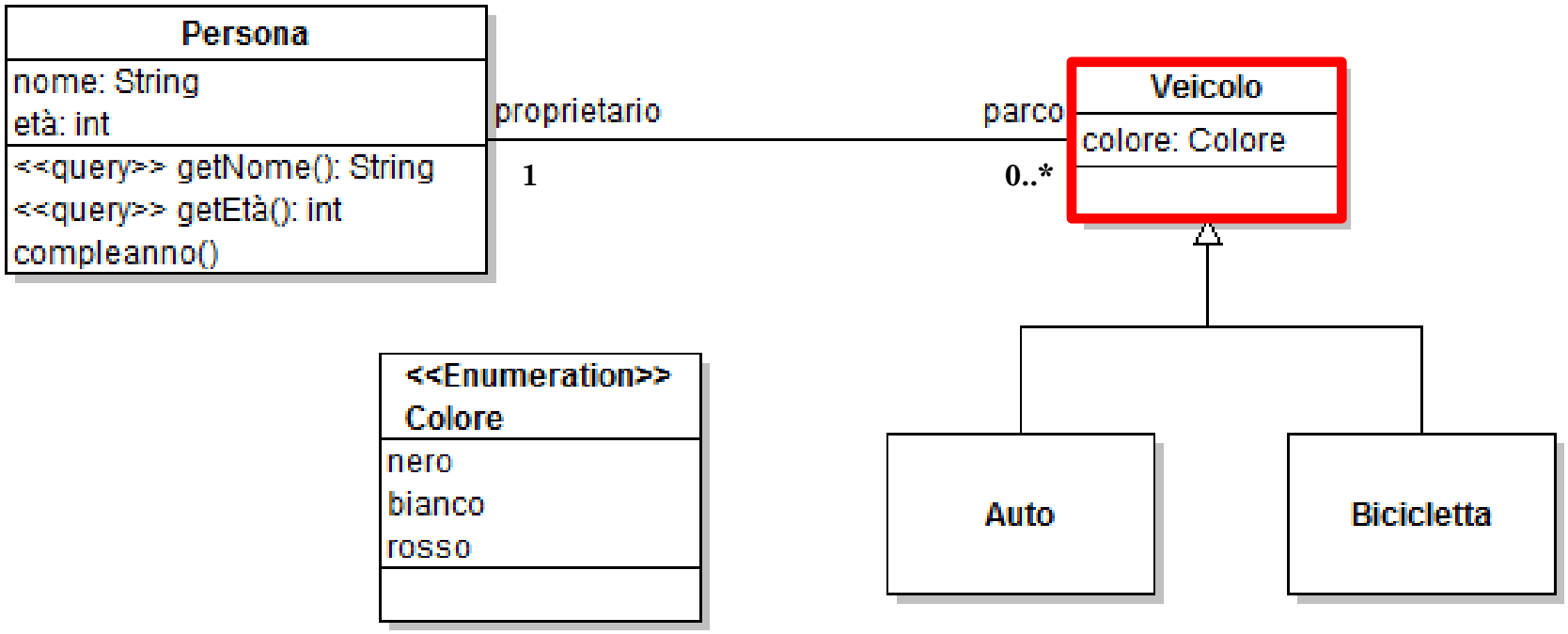
$\forall x\ (UOMO(x) \Rightarrow MORTALE(x))$

- Utilizza i "concetti" di
  - **Invariante**
  - **Pre-post condizioni**

Che voi conoscete già …
(design by contract)

ESEMPIO OCL (1)

Context specifies which elements we are talking
"." is used for navigation
Inv = Invariant
Self = this (java)



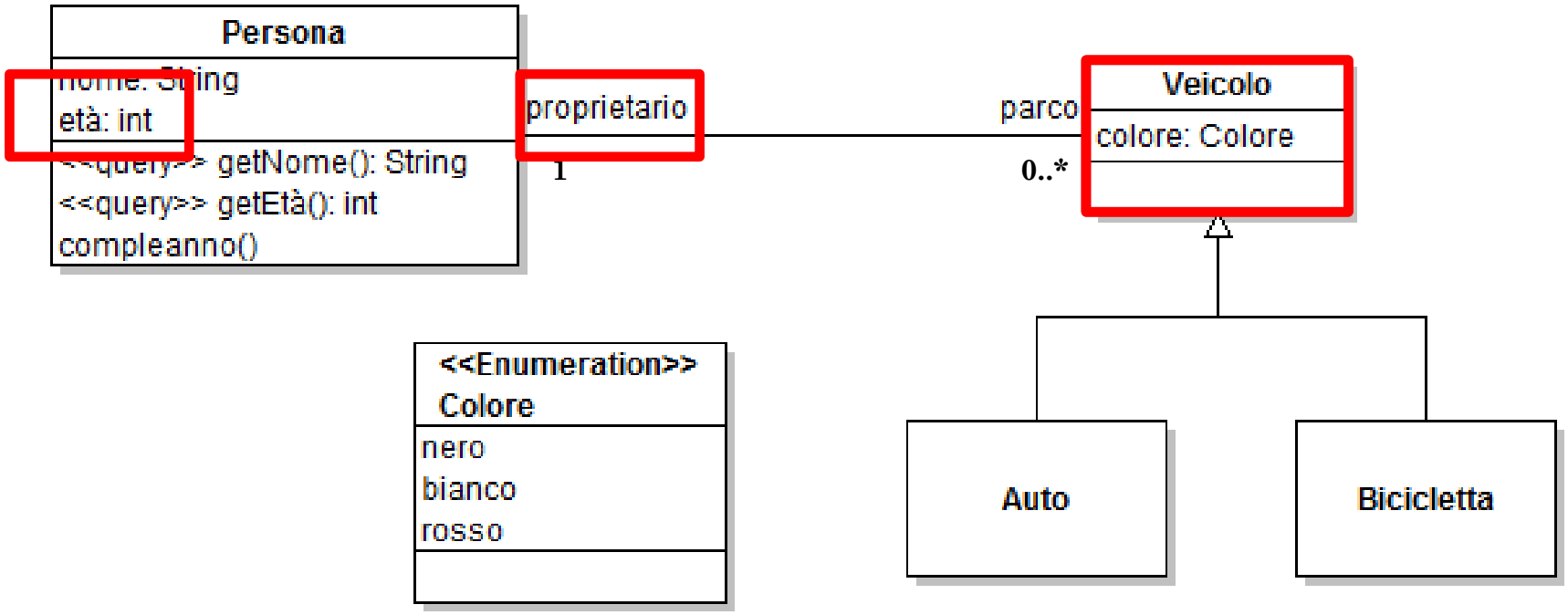
"Il proprietario di un veicolo deve avere almeno 18 anni"

Context Veicolo

Inv: **self.** proprietario.età >= 18

ESEMPIO OCL (1)

Context specifies which elements we are talking
"." is used for navigation
Inv = Invariant
Self = this (java)



"Il proprietario di un veicolo deve avere almeno 18 anni"

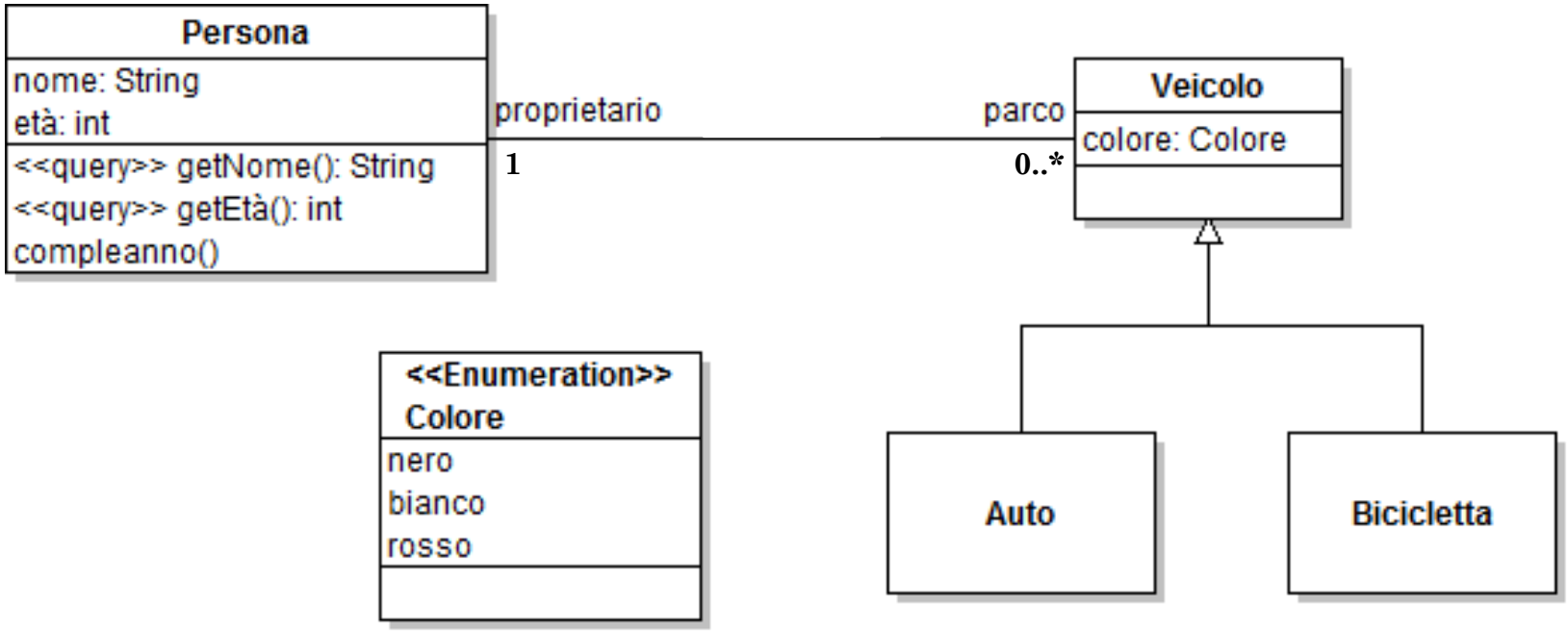
Context Veicolo

Inv: **self.** proprietario.età >= 18

ESEMPIO OCL (2)

Operations on collections: →
Alcune op: size, forAll, exists, iterate, …

"Collection supertipo di Insiemi e Liste"



"nessuna Persona ha più di tre veicoli" → *"tutte le Persone hanno meno di o esattamente tre veicoli"*

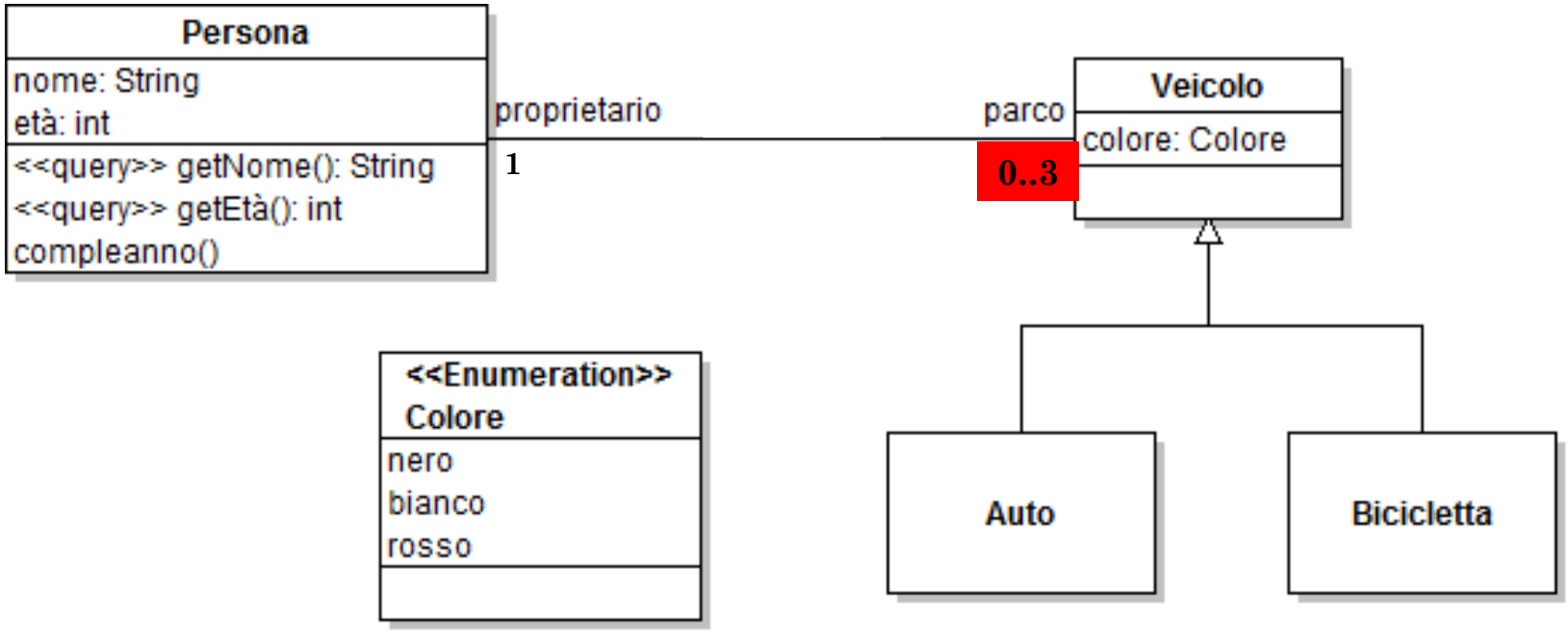
Context Persona

Inv: **self.**parco → **size** <= 3

Numero di elementi

Collezione di tutti i veicoli

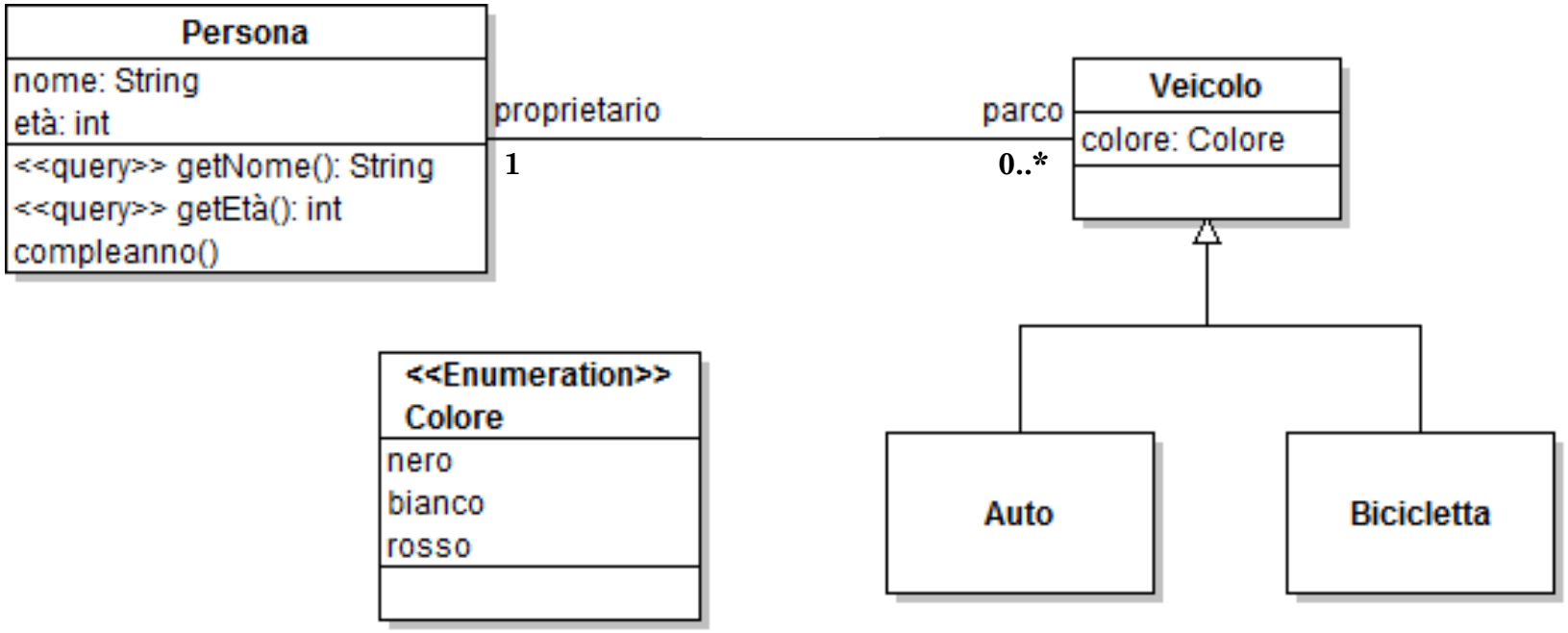
ESEMPIO OCL (2BIS)



"nessuna Persona ha più di tre veicoli" ⇒ *"tutte le Persone hanno meno di o esattamente tre veicoli"*

Soluzione più semplice ...

ESEMPIO OCL (3)

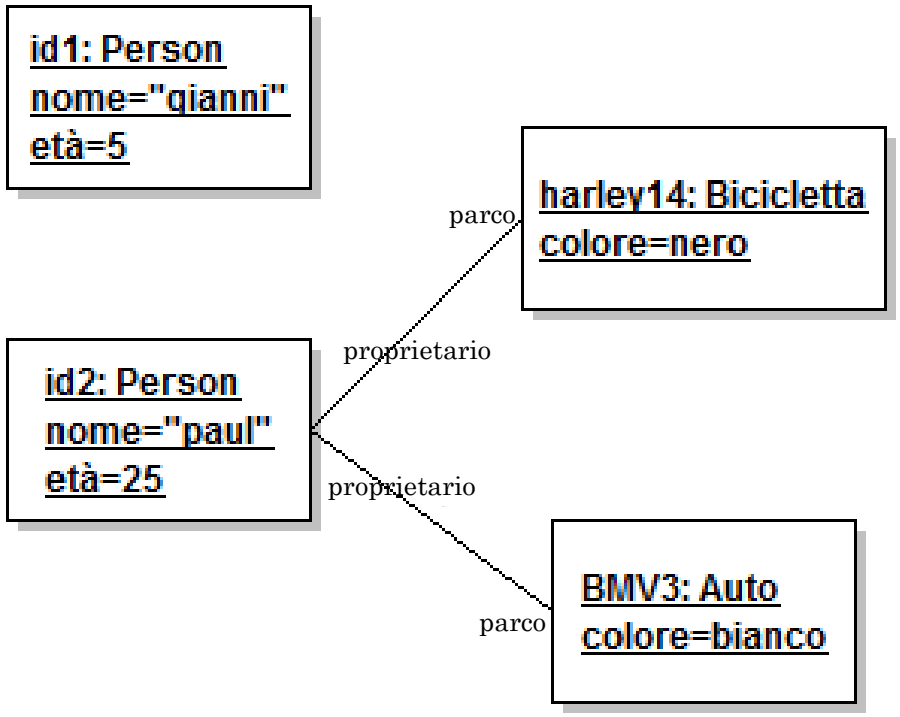


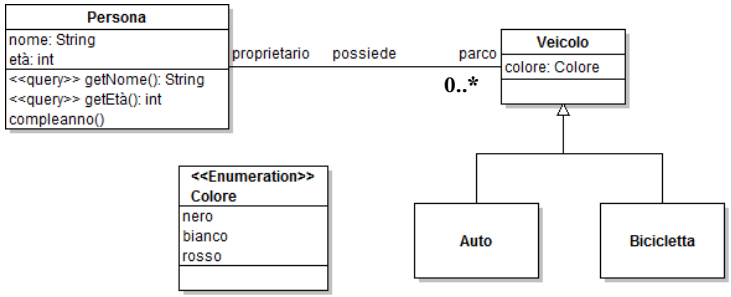
"Tutti i veicoli di una persona sono neri"

Context Persona

Inv: **self.**parco → **forAll(**v | v.colore=nero**)**

OBJECT DIAGRAM E VINCOLI: ESEMPIO





Senza vincoli

Ammissibile! ✓

Context Veicolo

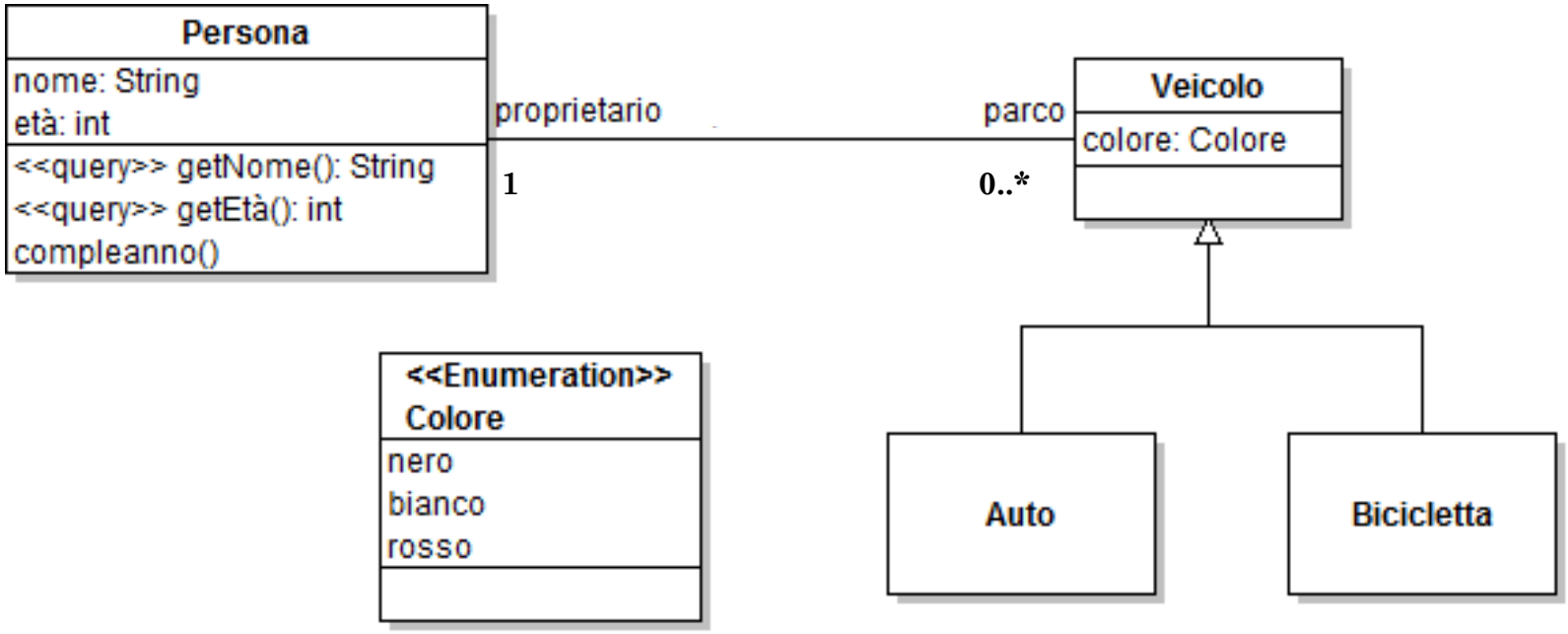
Inv: **self.** proprietario.età > 18 ✓

Context Persona

Inv: **self.**parco → **forAll(**v | v.colore=nero**)** ☒

FINO ADESSO SOLO INVARIANTI DI CLASSE …

OCL può anche essere usato anche per specificare un'operazione mediante pre/post condition



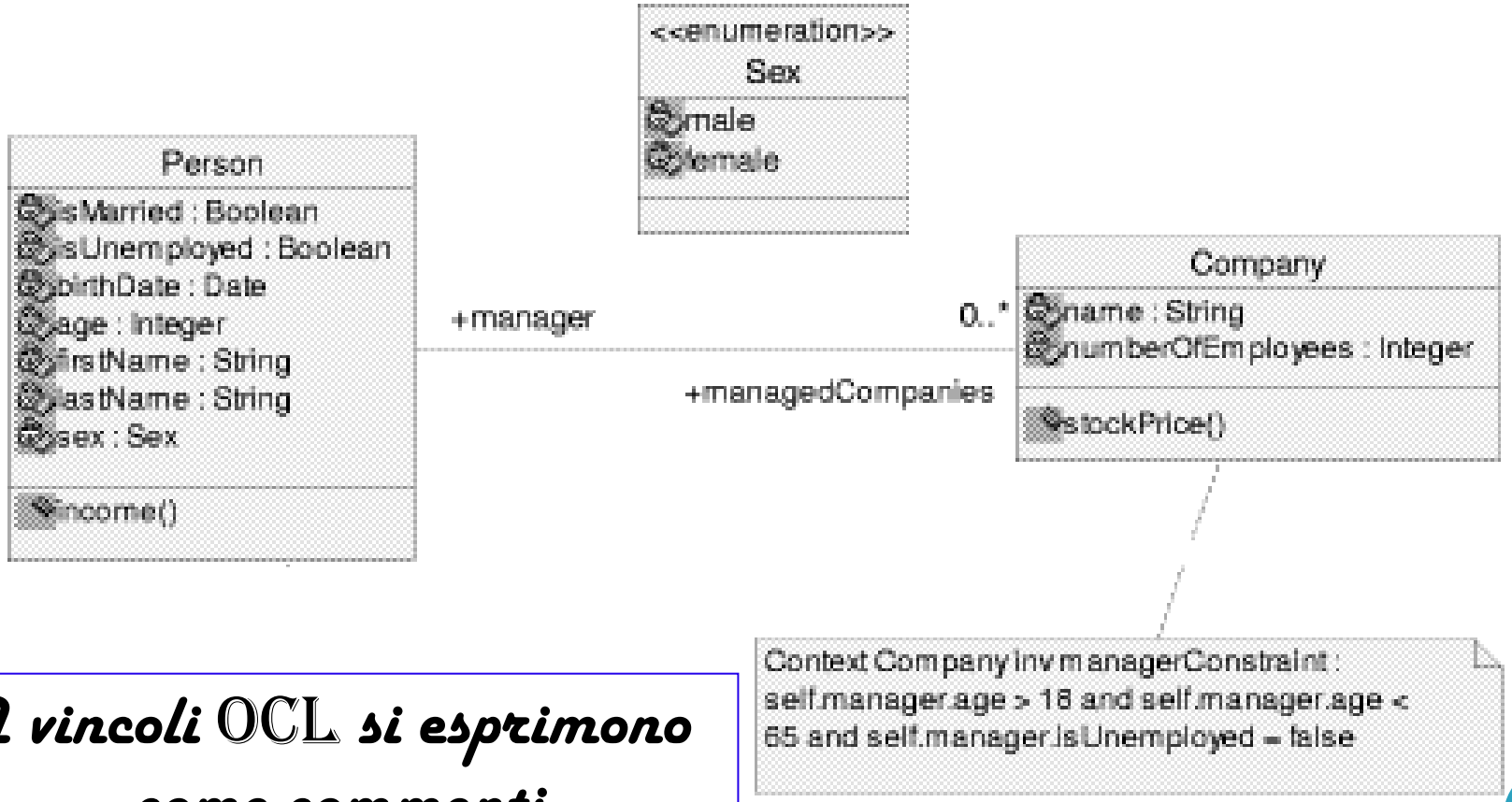
"compleanno() incrementa di 1 l'età di una Persona"

Context Persona::compleanno()

post: **self.**età **= self.**età**@pre** + 1

Valore prima dell'applicazione dell'operazione

ESEMPIO DI VINCOLO OCL

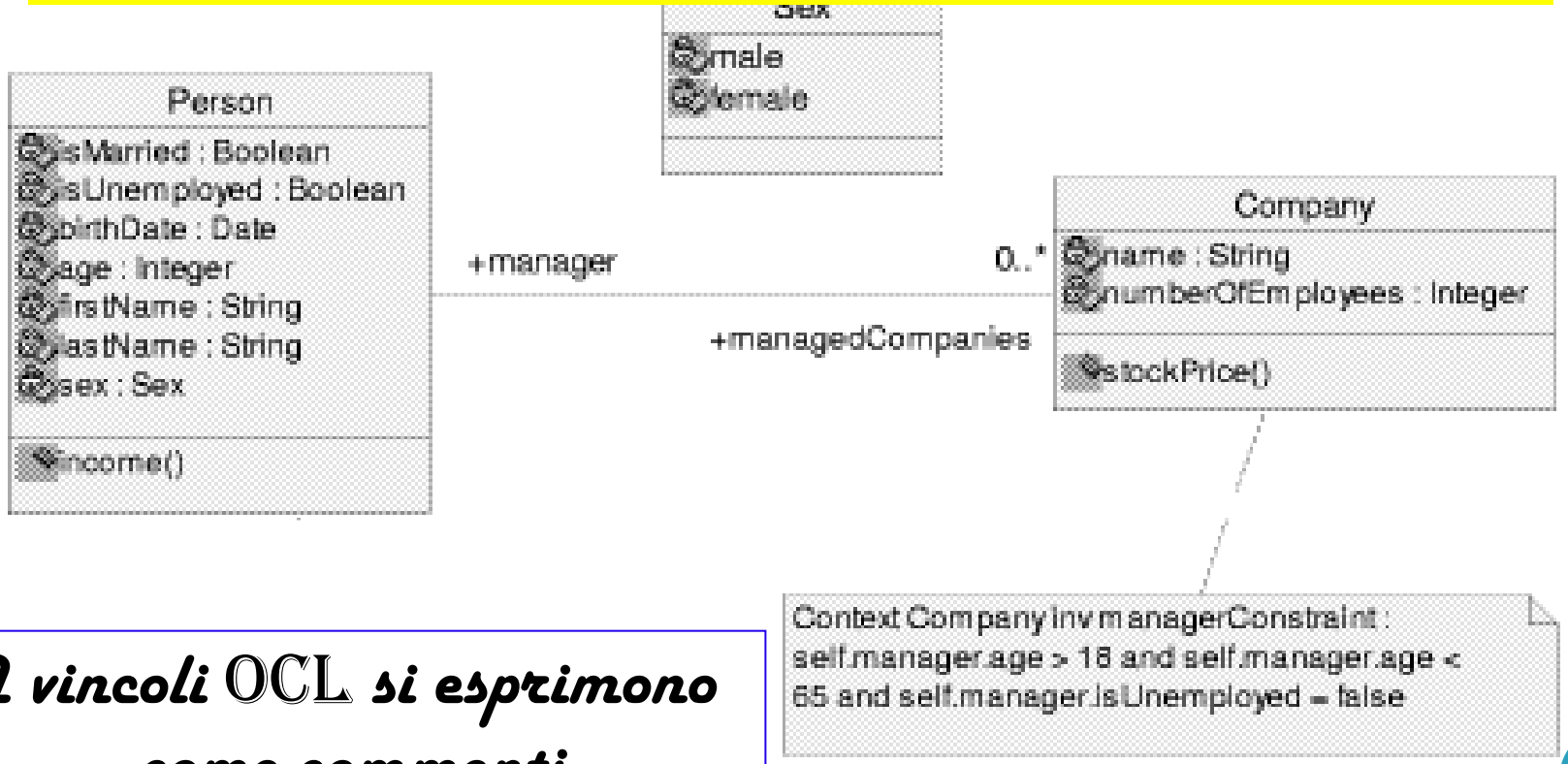


I vincoli OCL si esprimono come commenti

Constraint per essere manager

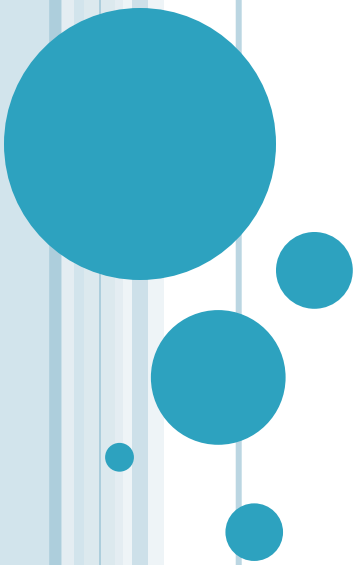
Esempio di vincolo OCL

Warning: Su OCL c'è molto molto di più …



I vincoli OCL si esprimono come commenti

Constraint per essere manager



Esercizi sul class diagram

ESAME 31 AGOSTO 2012

- L'applicazione da progettare dovrà gestire **contravvenzioni stradali** irrogate dalla Polizia Municipale per violazione del Codice della Strada.
- Un **vigile**, che ha un nome, un cognome e una matricola, può effettuare zero o più contravvenzioni.
- Una **contravvenzione** ha un numero (numero_verbale) che è un intero ed una descrizione nella quale viene inserito il luogo dove è avvenuta la contravvenzione.
- Inoltre una contravvenzione è relativa ad un **veicolo** il quale ha una targa, un colore e un tipo_veicolo (es. auto o moto).
- Può anche accadere che allo stesso veicolo vengano irrogate più contravvenzioni.
- Infine ad ogni veicolo corrisponde un solo propietario che può essere una persona fisica o una persona giuridica (ad esempio una società).

ESAME 31 AGOSTO 2012

- L'applicazione da progettare dovrà gestire **contravvenzioni stradali** irrogate dalla Polizia

La descrizione non è completa; ad esempio manca la data e l'importo della contravvenzione. Se non è richiesto esplicitamente non dovete aggiungere attributi e associazioni

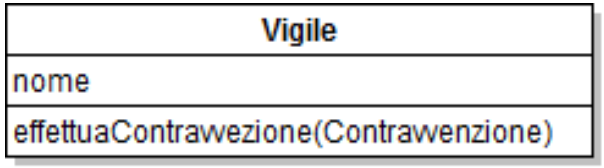
- Una **contravvenzione** ha un numero (numero_verbale)

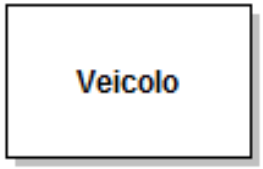
Cercate di seguire la descrizione il più possibile!!

- Inoltre una contravvenzione è relativa ad un **veicolo** il quale ha una targa, un colore e un tipo_veicolo (es. auto o moto).
- Può anche accadere che allo stesso veicolo vengano irrogate più contravvenzioni.
- Infine ad ogni veicolo corrisponde un solo propietario che può essere una persona fisica o una persona giuridica (ad esempio una società).

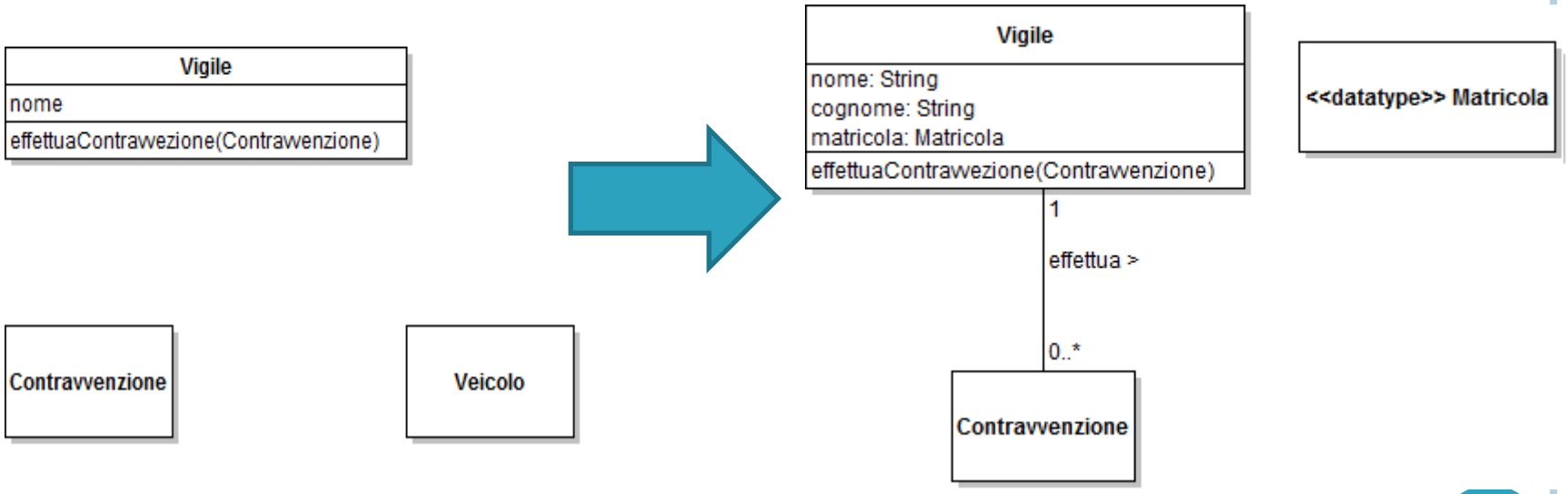
Domanda a)

- Completare il seguente diagramma delle classi UML (**prospettiva concettuale**) seguendo la descrizione data sopra. Aggiungere i tipi agli attributi, i nomi o ruoli alle relazioni e la molteplicità

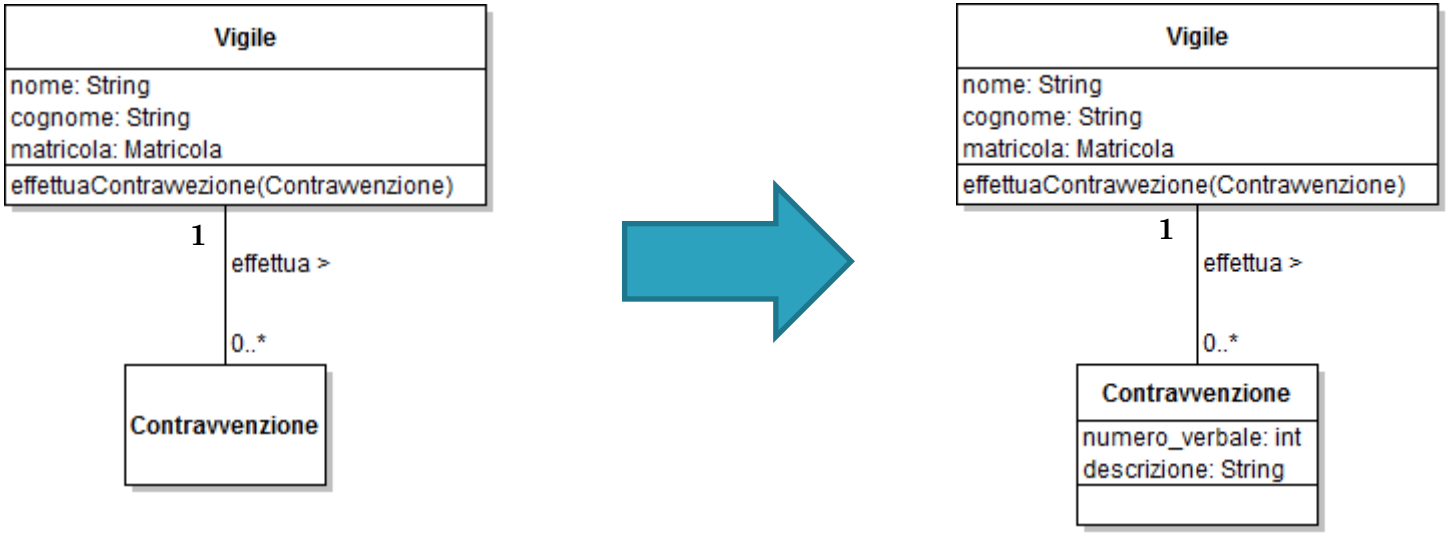




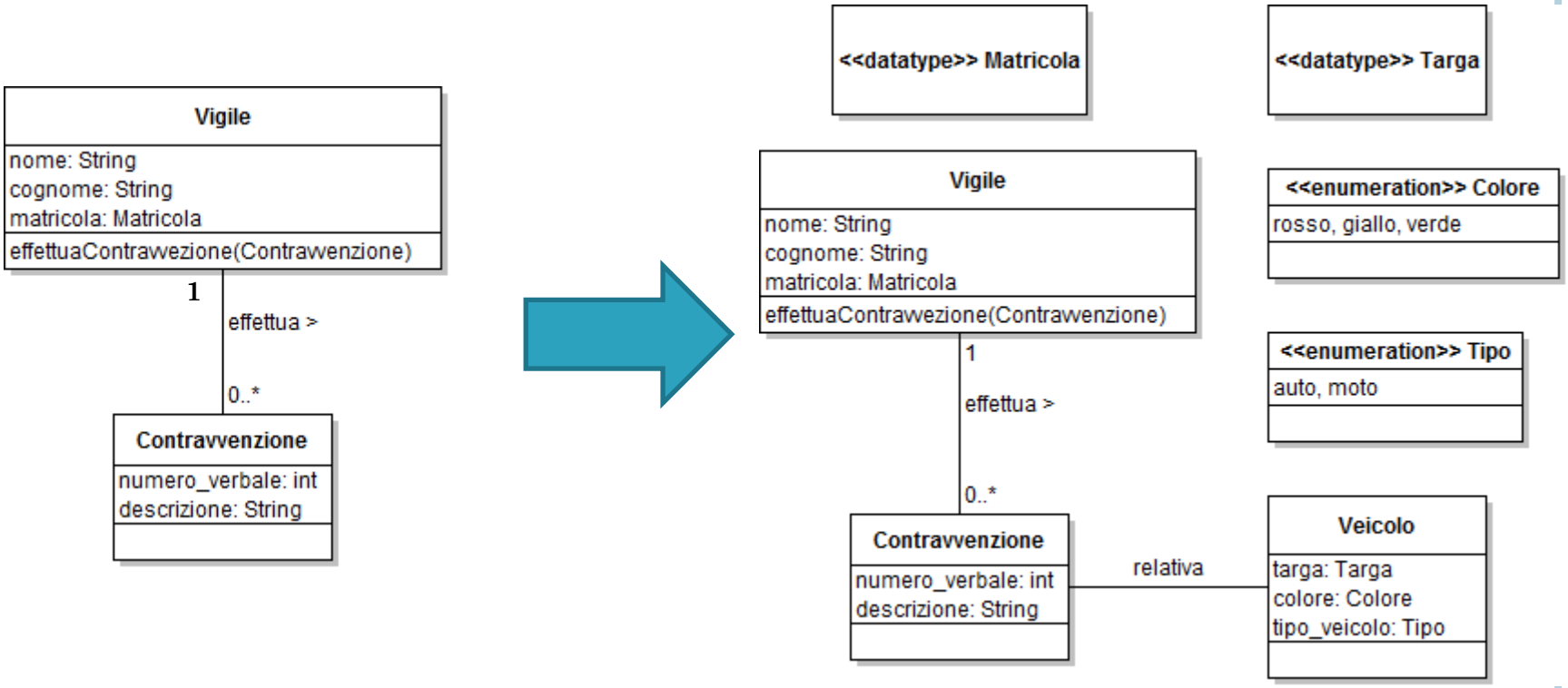
- Un **vigile**, che ha un nome, un cognome e una matricola, può effettuare zero o più contravvenzioni



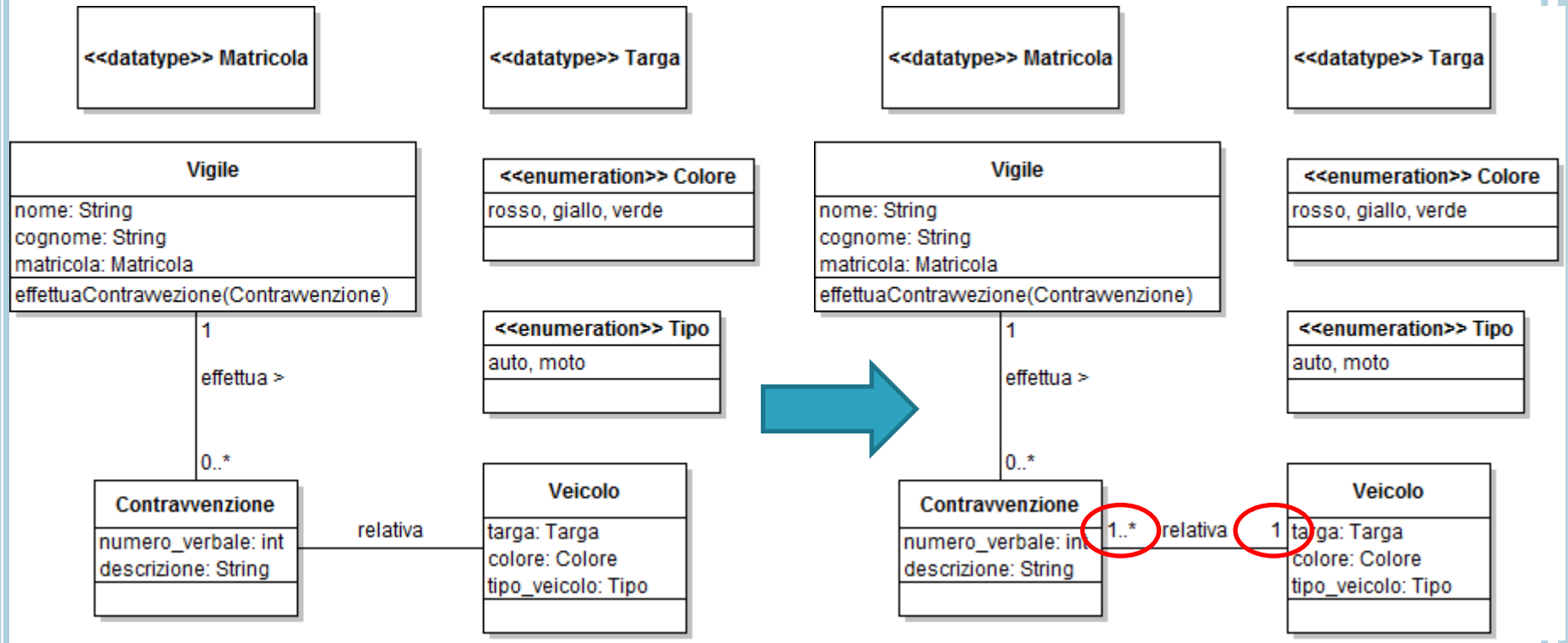
- Una **contravvenzione** ha un numero (numero_verbale) che è un intero ed una descrizione nella quale viene inserito il luogo dove è avvenuta la contravvenzione



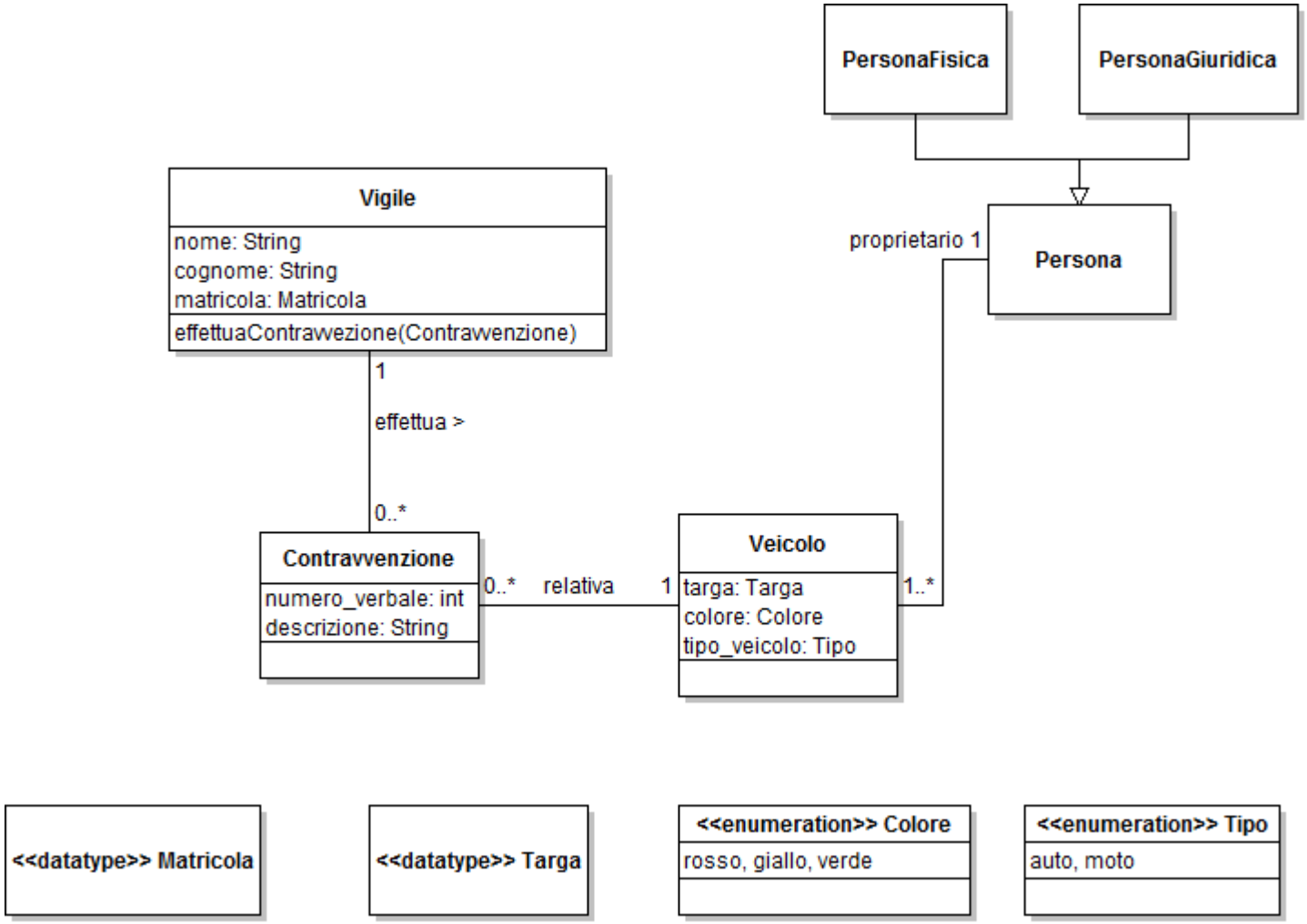
- Inoltre una contravvenzione è relativa ad un veicolo il quale ha una targa, un colore e un tipo del veicolo (es. auto o moto)



- Può anche accadere che allo stesso veicolo vengano irrogate più contravvenzioni



- Infine ad ogni veicolo corrisponde un solo propietario che può essere una persona fisica o una persona giuridica (ad esempio una società)

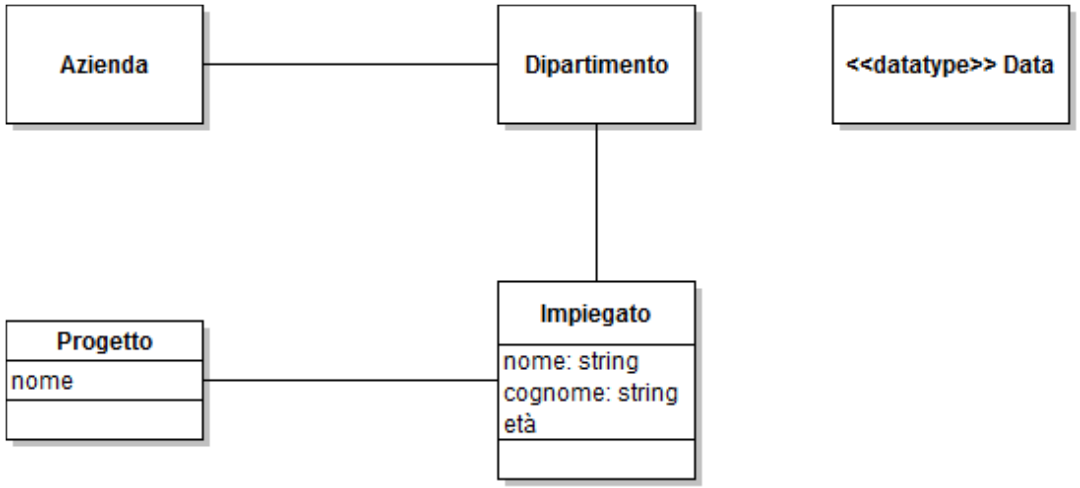


ESAME 6-2-2011

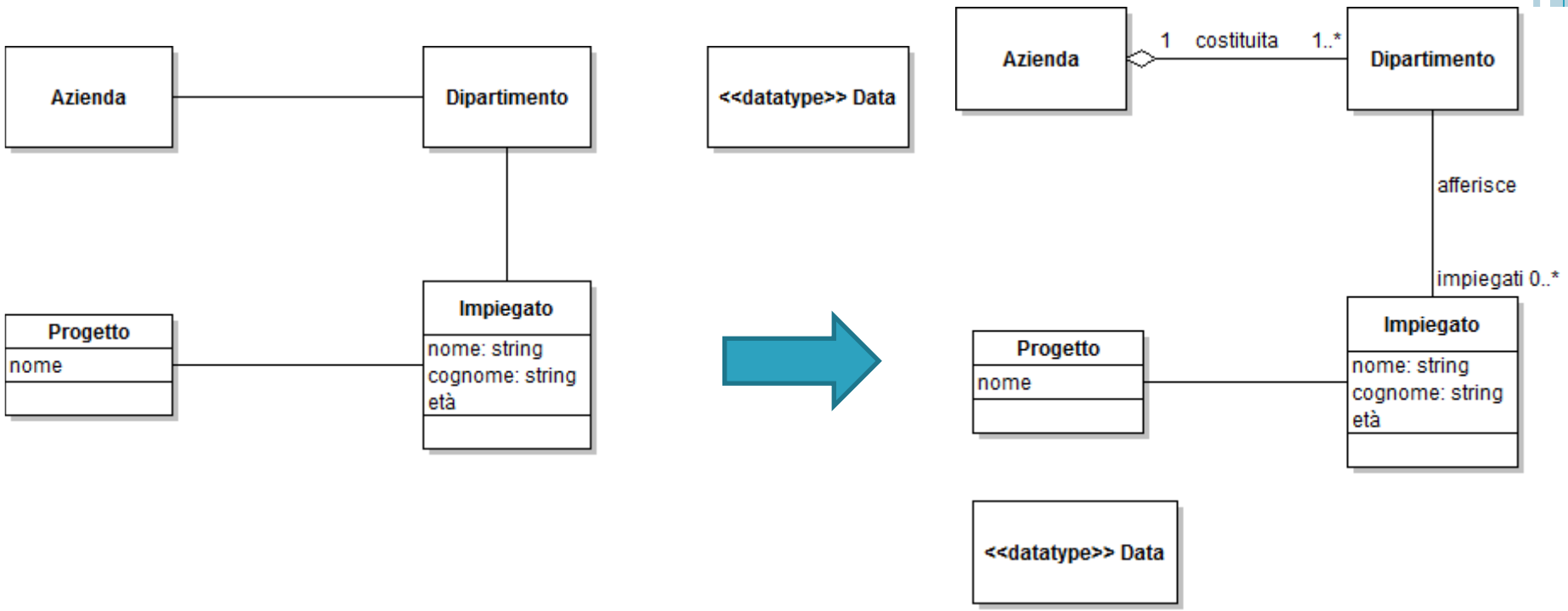
- Un'**azienda** è costituita da uno o più dipartimenti, ad ognuno dei quali afferisce un certo numero di impiegati.
- Ogni **impiegato** (del quale interessa nome, cognome, età e stipendio) afferisce esattamente ad un **dipartimento**.
- Dei dipartimenti interessa il nome, il numero di telefono, la data di afferenza di ognuno degli impiegati che vi lavorano, ed il direttore.
- Il direttore è un impiegato e può dirigere solo un dipartimento.
- Gli impiegati partecipano a vari **progetti** aziendali, dei quali interessa il nome ed il budget.

Domanda a)

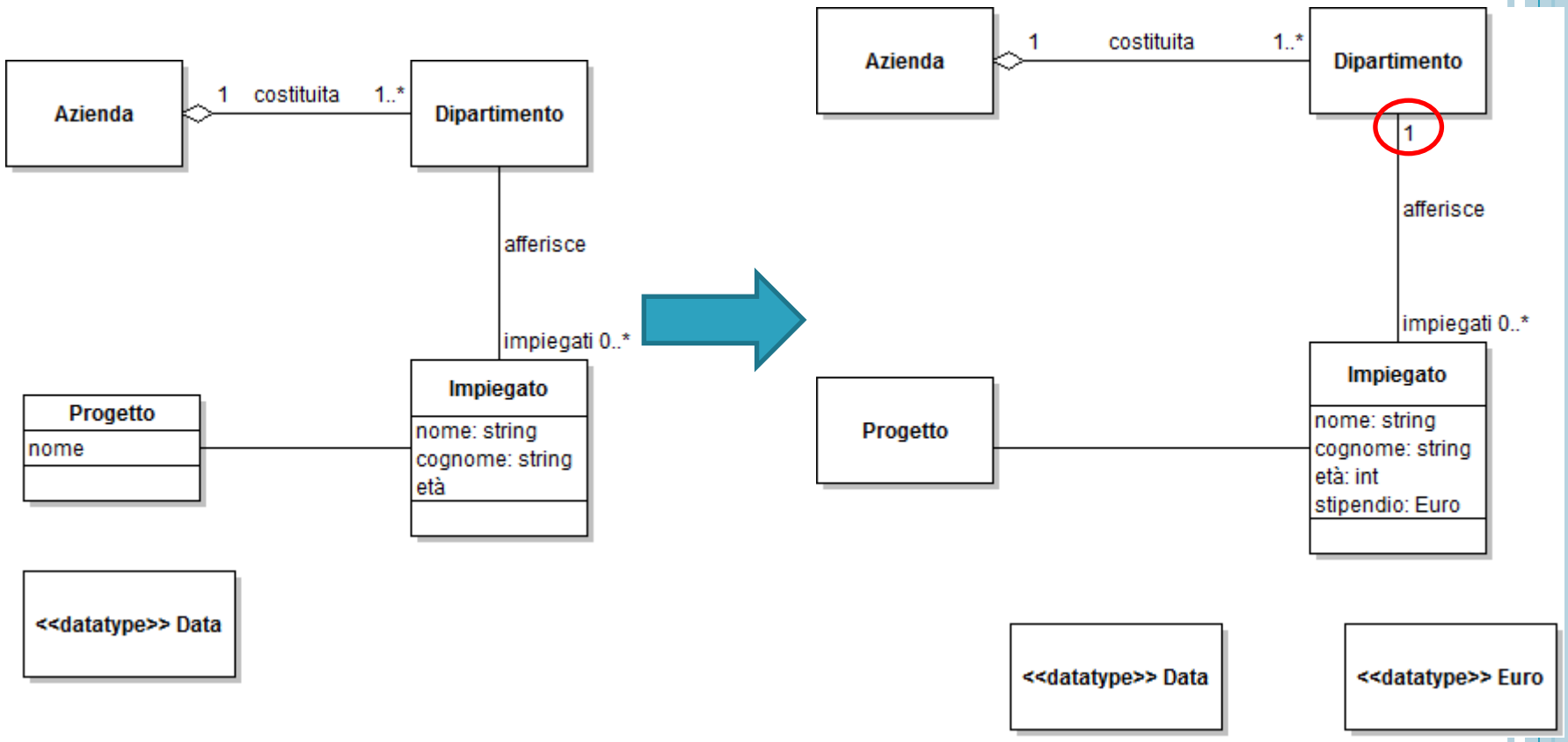
- Completare il seguente class diagram in prospettiva concettuale (aggiungere classi, attributi con i rispettivi tipi, operazioni, associazioni e molteplicità) considerando la descrizione data



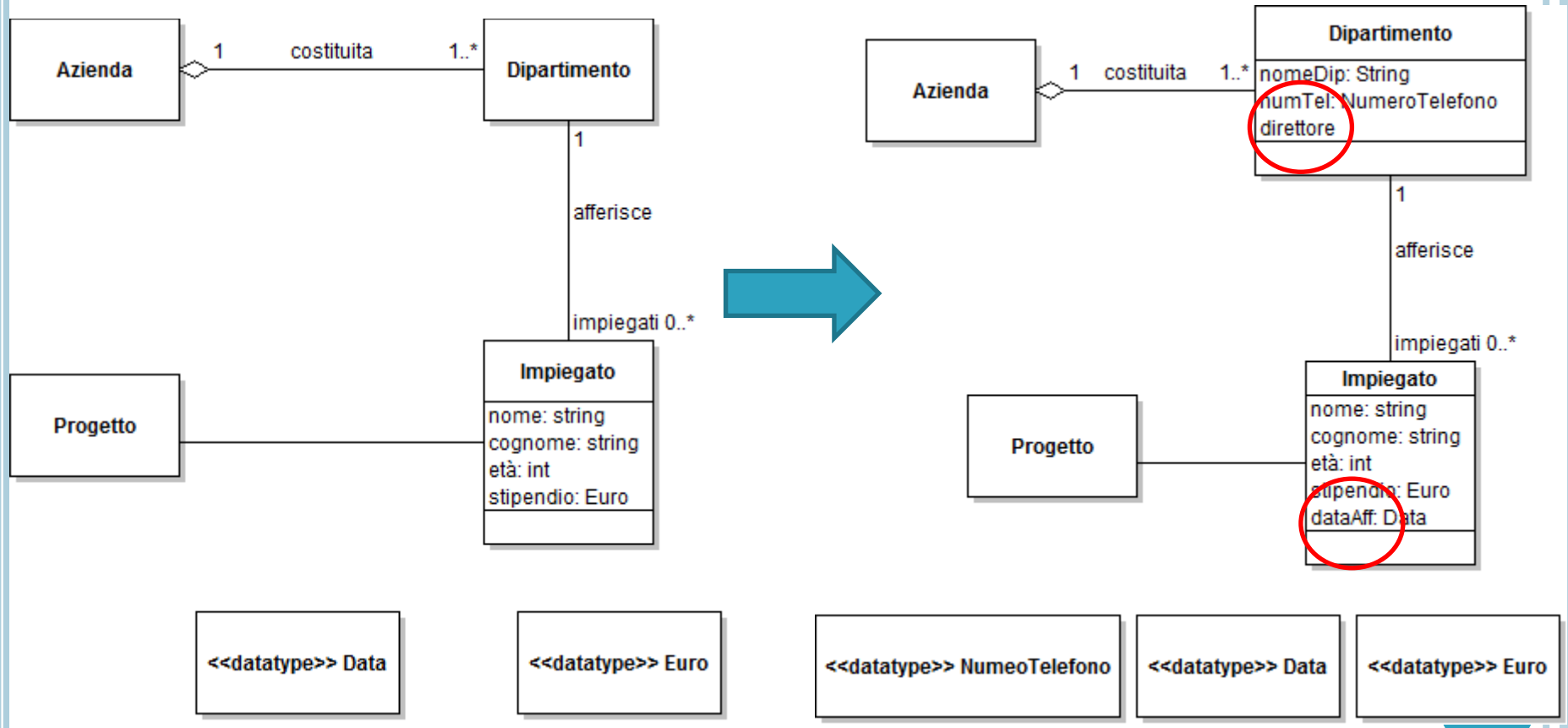
- Un'**azienda** è costituita da uno o più dipartimenti, ad ognuno dei quali afferisce un certo numero di impiegati



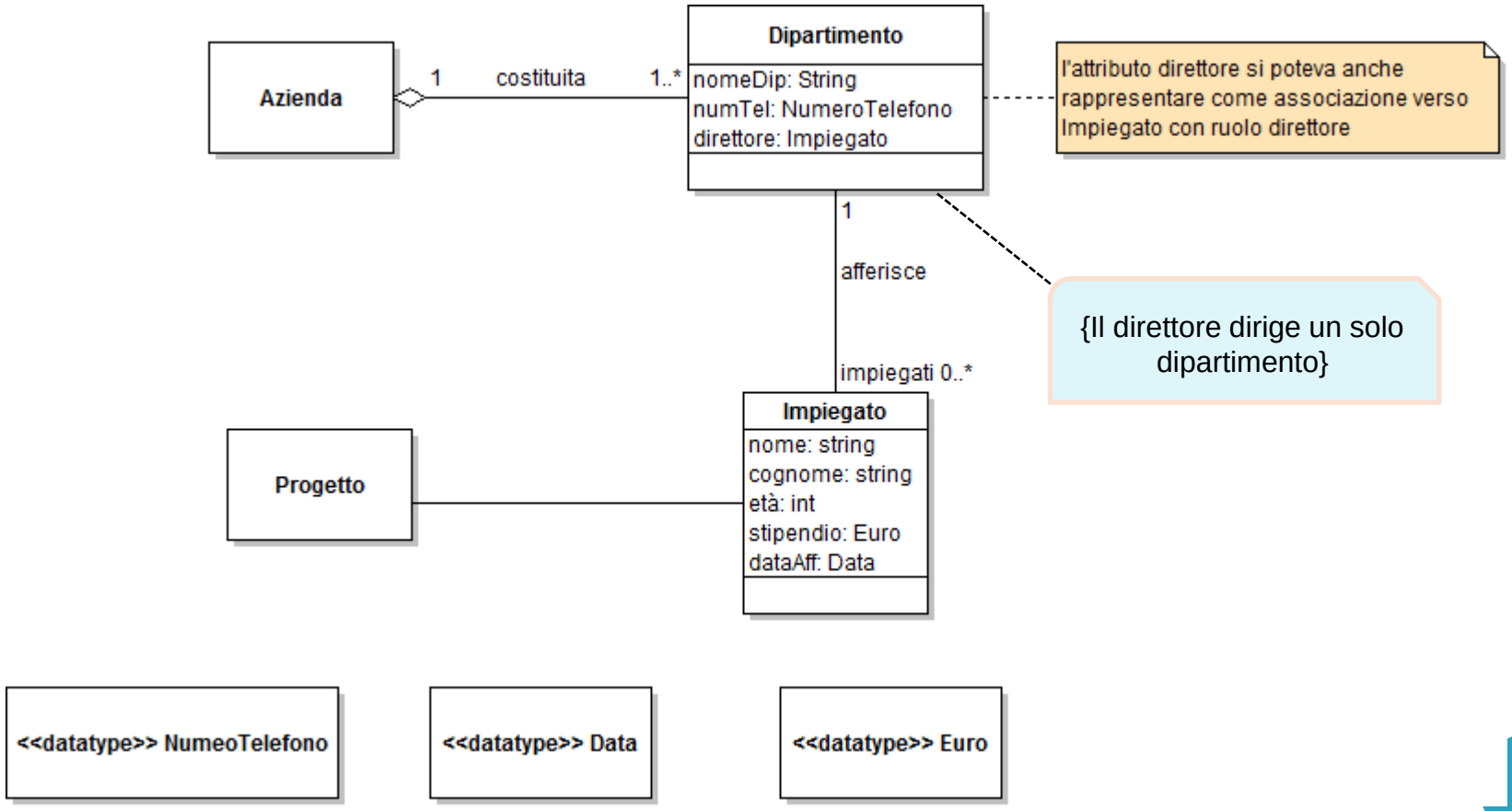
- Ogni **impiegato** (del quale interessa nome, cognome, età e stipendio) afferisce esattamente ad un **dipartimento**



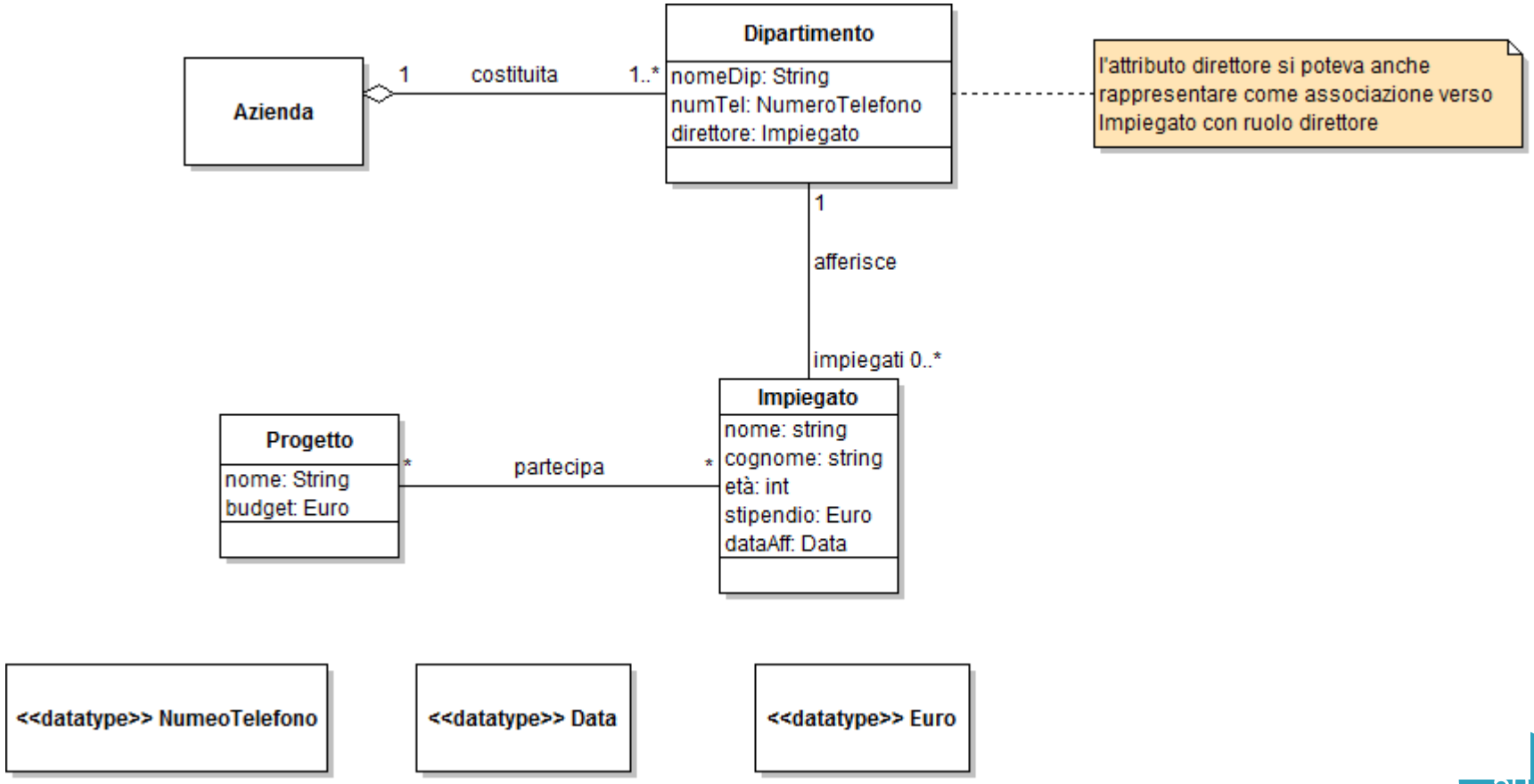
- Dei dipartimenti interessa il nome, il numero di telefono, la data di afferenza di ognuno degli impiegati che vi lavorano, ed il direttore



- Il direttore è un impiegato e può dirigere solo un dipartimento

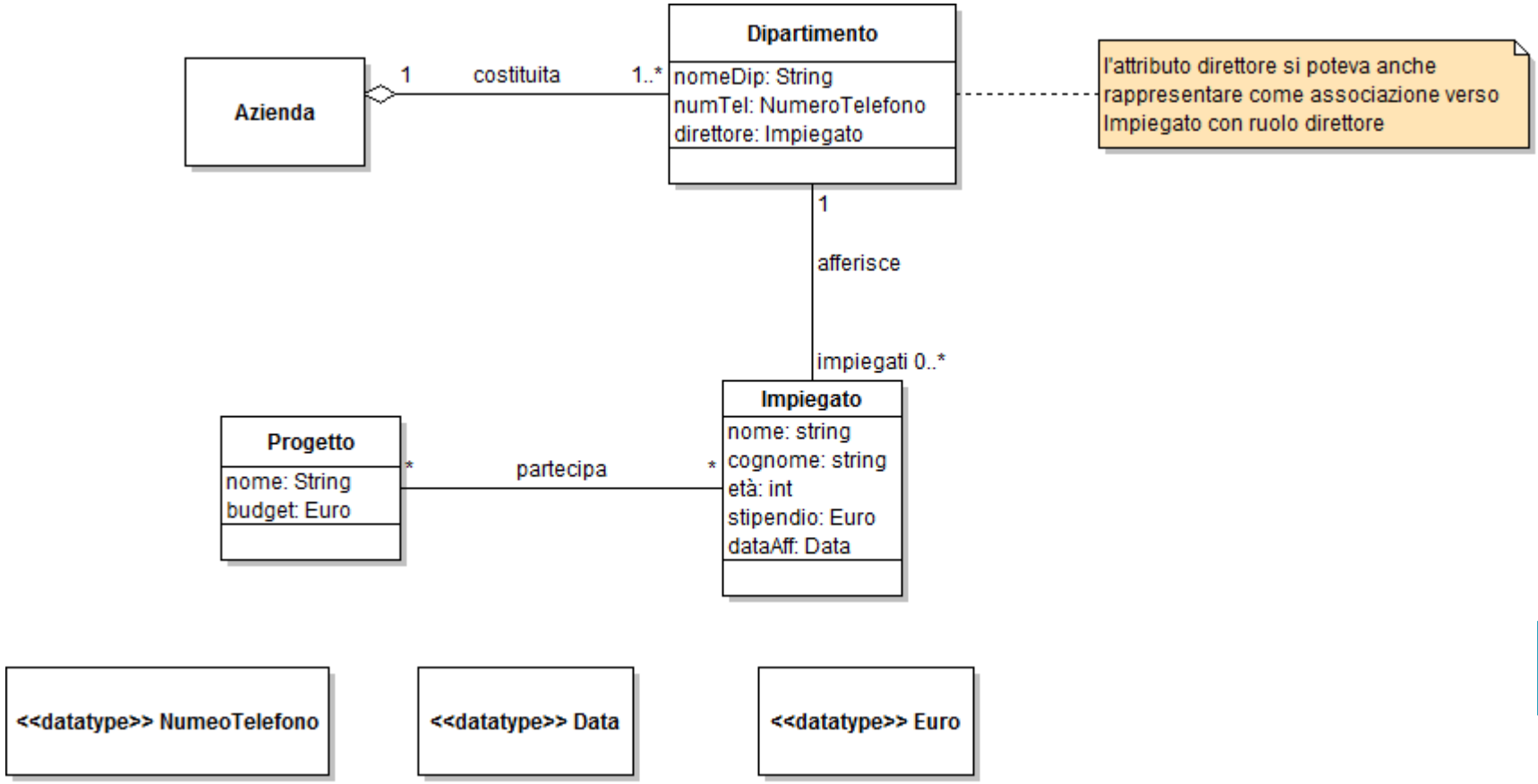


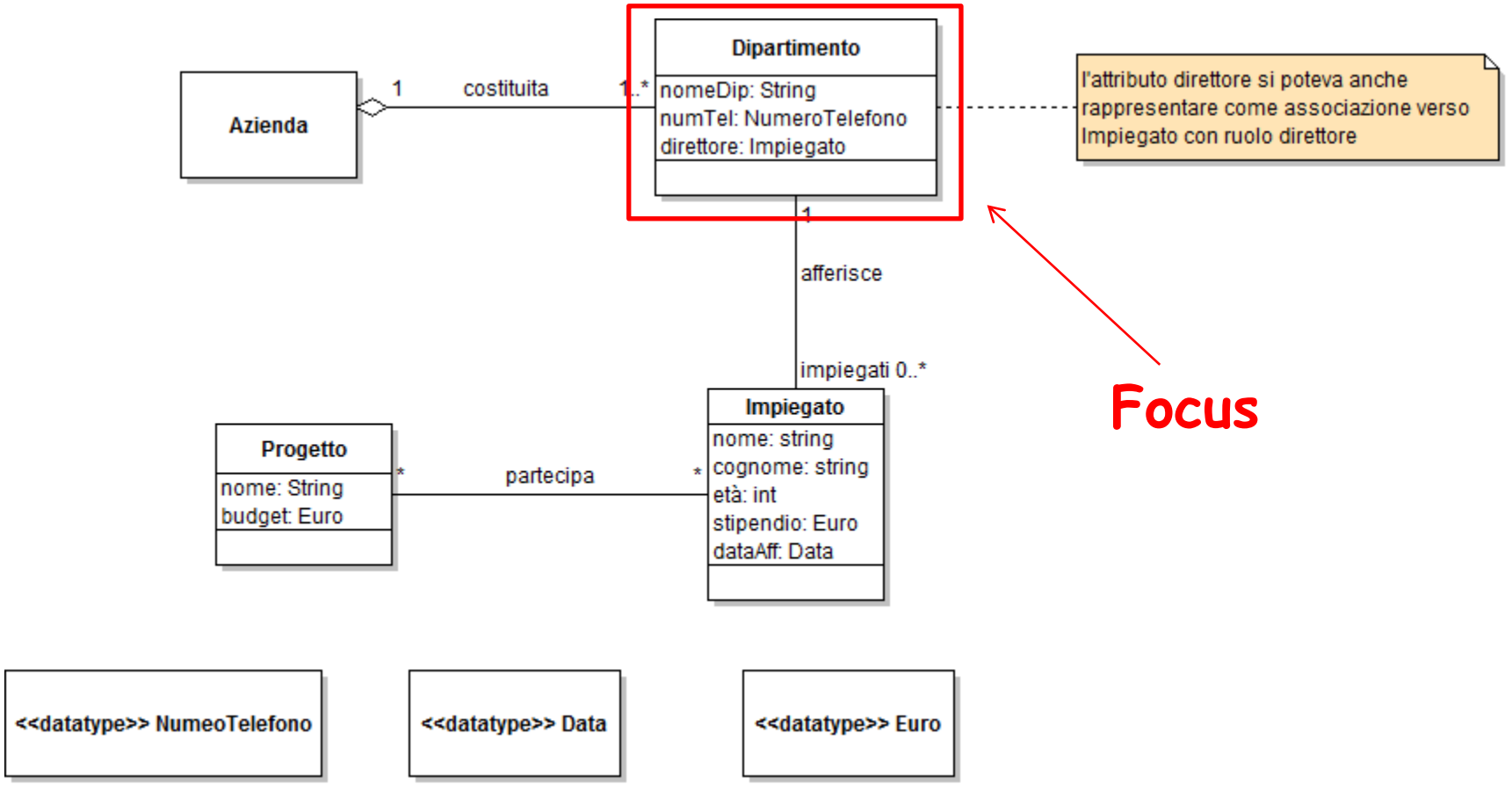
- Gli impiegati partecipano a vari **progetti** aziendali, dei quali interessa il nome ed il budget



Domanda B)

- Passiamo alla prospettiva software. E' richiesto di codificare nel linguaggio Java la classe Dipartimento (aggiungere solo attributi ed associazioni del modello così come modificato per la domanda a)

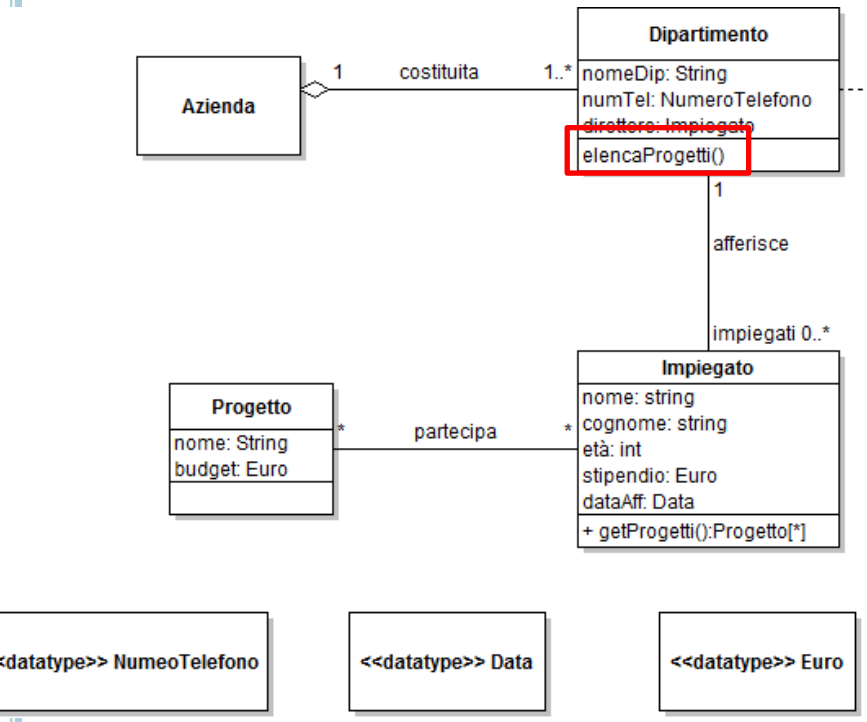





```
public class Dipartimento {
    private String nomeDip;
    private NumeroTelefono numTel;
    private Azienda azienda;
    private Impiegato direttore;
    private HashSet<Impiegato> impiegati;
}
```

Domanda C)

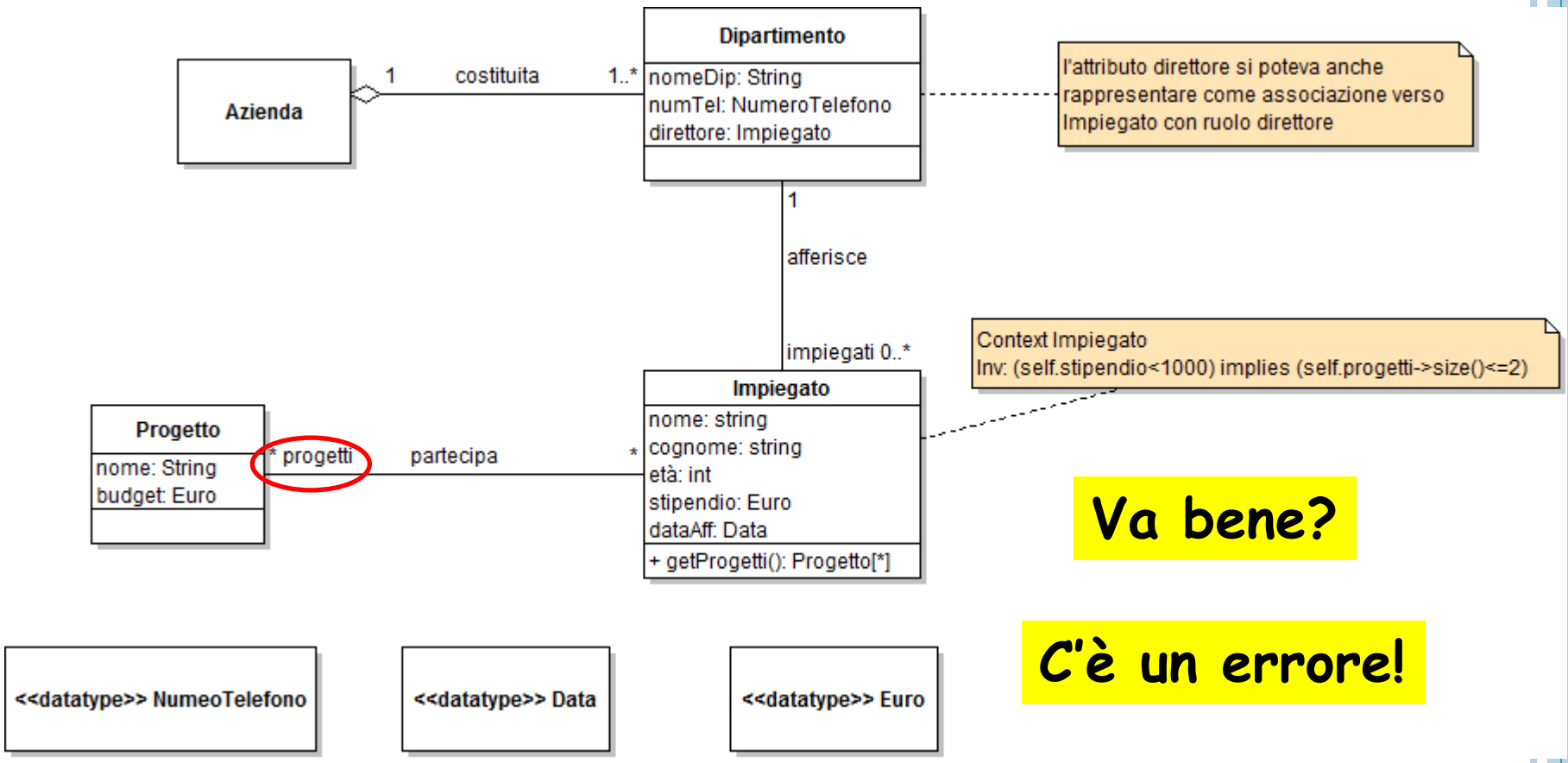
- Scrivere in pseudocodice Java l'operazione **elencaProgetti()** della classe Dipartimento. Tale operazione **stampa** la lista dei nomi dei progetti e **ritorna** la lista di progetti a cui partecipano tutti gli afferenti del dipartimento (attenzione a non stampare più volte lo stesso progetto).



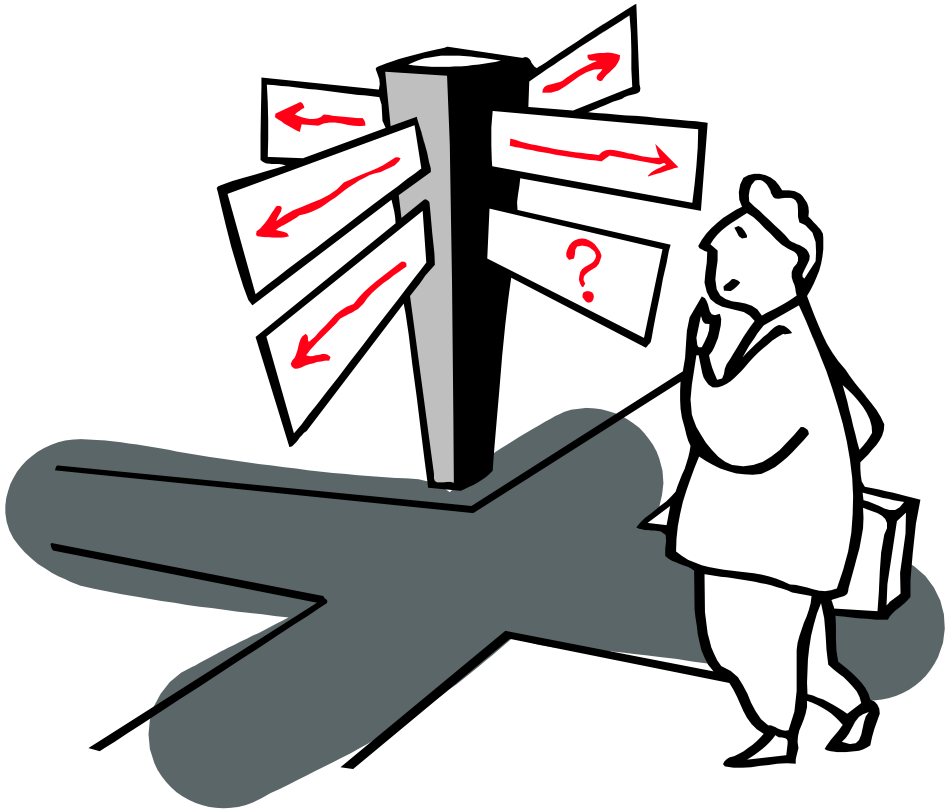

```
Public List<progetto> elencaProgetti() {
      List<progetto> ris = new List<progetto>()
      Foreach (Impiegato i in impiegati)
         Foreach (Progetto p in i.getProgetti())
           If(!ris.contains(p)) {
                 ris.add(p)
                 p.printNome()
           }
            return ris;
}
```

DOMANDA D) (FACOLTATIVA)

- Esprimere in OCL il seguente vincolo sul modello: *gli impiegati che hanno uno stipendio minore di 1000 euro al mese non possono partecipare a più di 2 progetti contemporaneamente*



The end ...



Domande?