Dijagrami klasa: napredni elementi

Stereotip

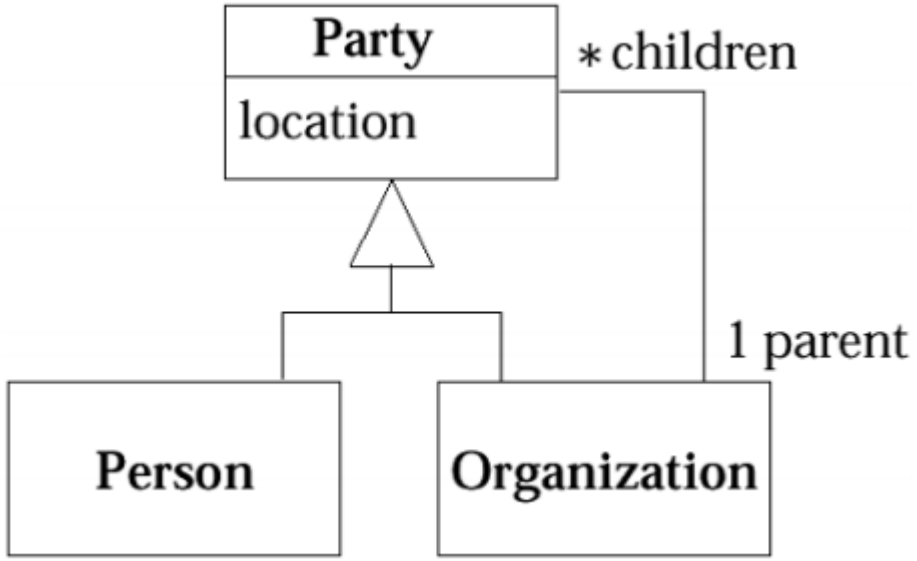
- Primjer stereotipa je interfejs
  - Interfejs je klasa sa deklarisanim javnim metodama
  - Specijalan oblik klase
  - Označava se se *<<interface>>*

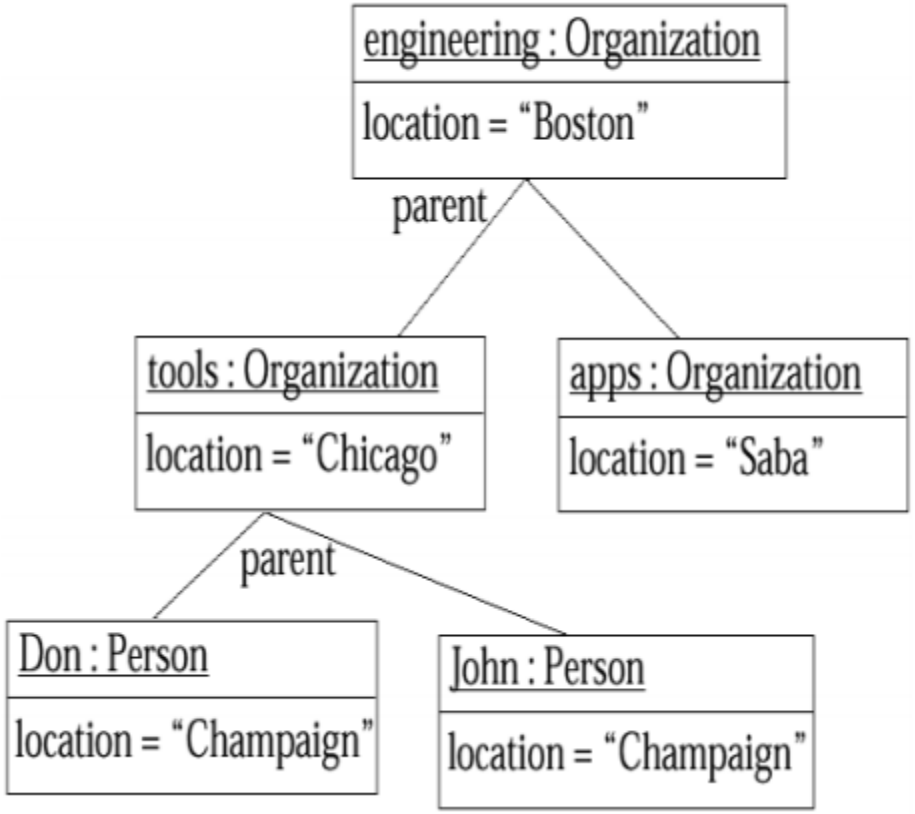
- Da li je *<<abstract>>* stereotip?

Dijagrami objekata

- Snapshot sistema u određenom trenutku
  - Naziva se i dijagram instanci
  - Koristi se u slučajevima komplikovanih relacija između objekata

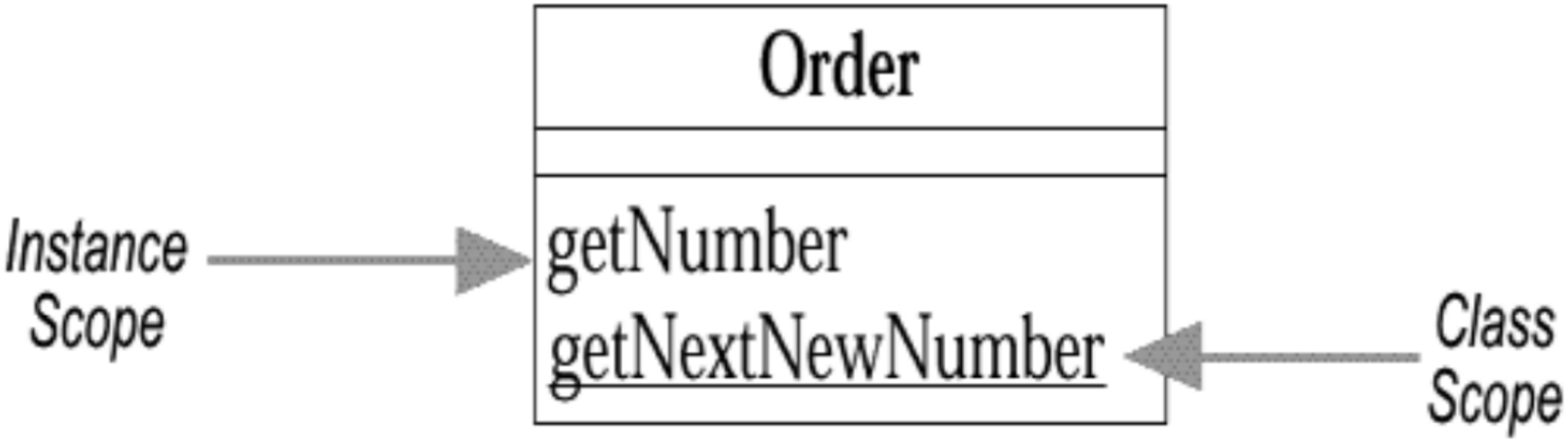
Dijagram objekata, primjer





Statičke operacije i atributi

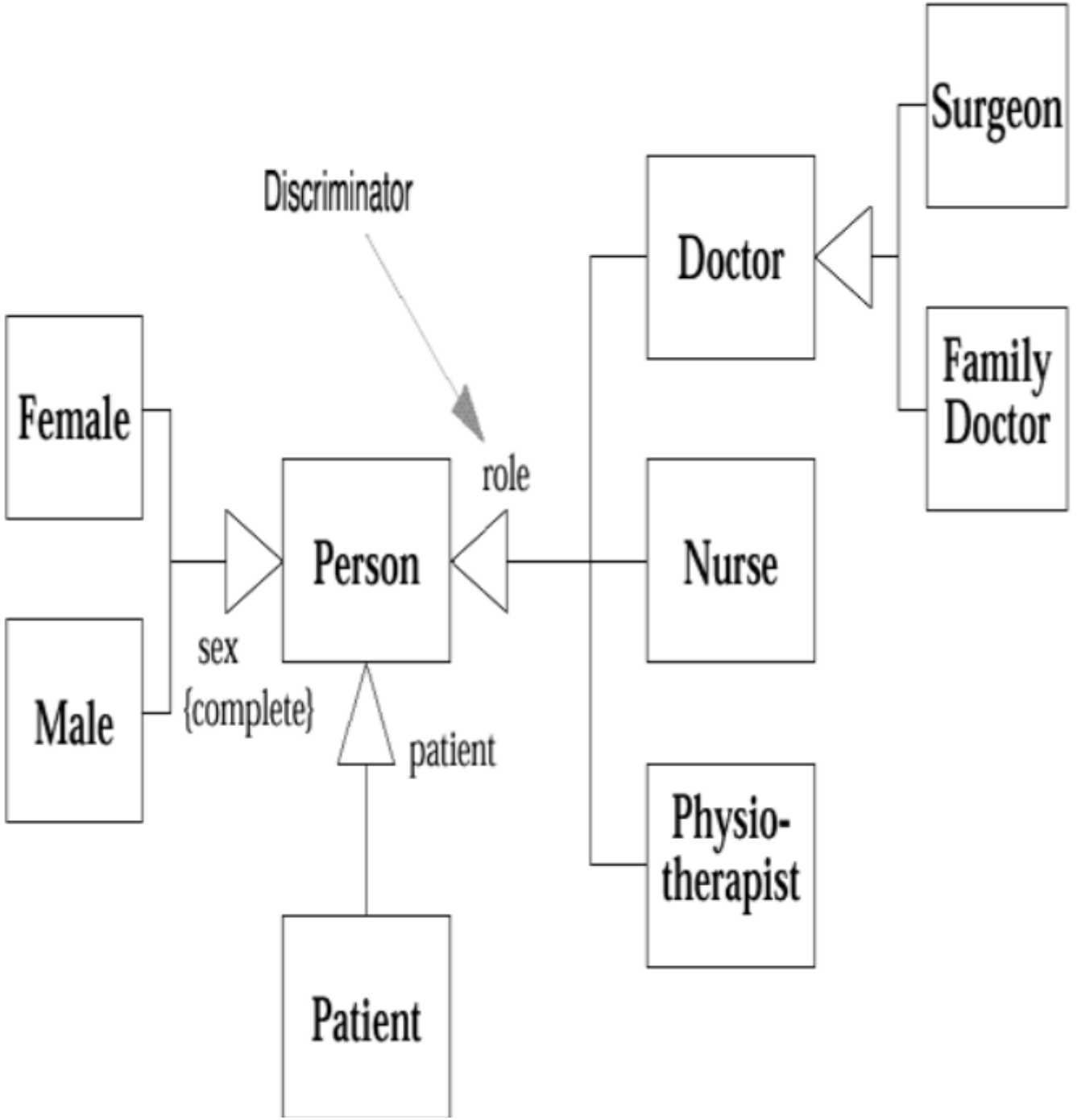
- Operacije i atributi koji se odnose na cijelu klasu a ne na jednu njenu instancu zovu se statički
  - Brojač kreiranih objekata klase



Višestruka i dinamička klasifikacija

- Klasifikacija je veza između objekta i njegovog tipa
  - Jednostruka klasifikacija - jedan objekat pripada jednoj klasi koja može biti izvedena iz drugih klasa
  - Višestruka klasifikacija – jedan objekat se opisuje pomoću nekoliko tipova, koji ne moraju biti povezani nasljeđivanjem
- Višestruka klasifikacija vs. višestruko nasljeđivanje

Primjer

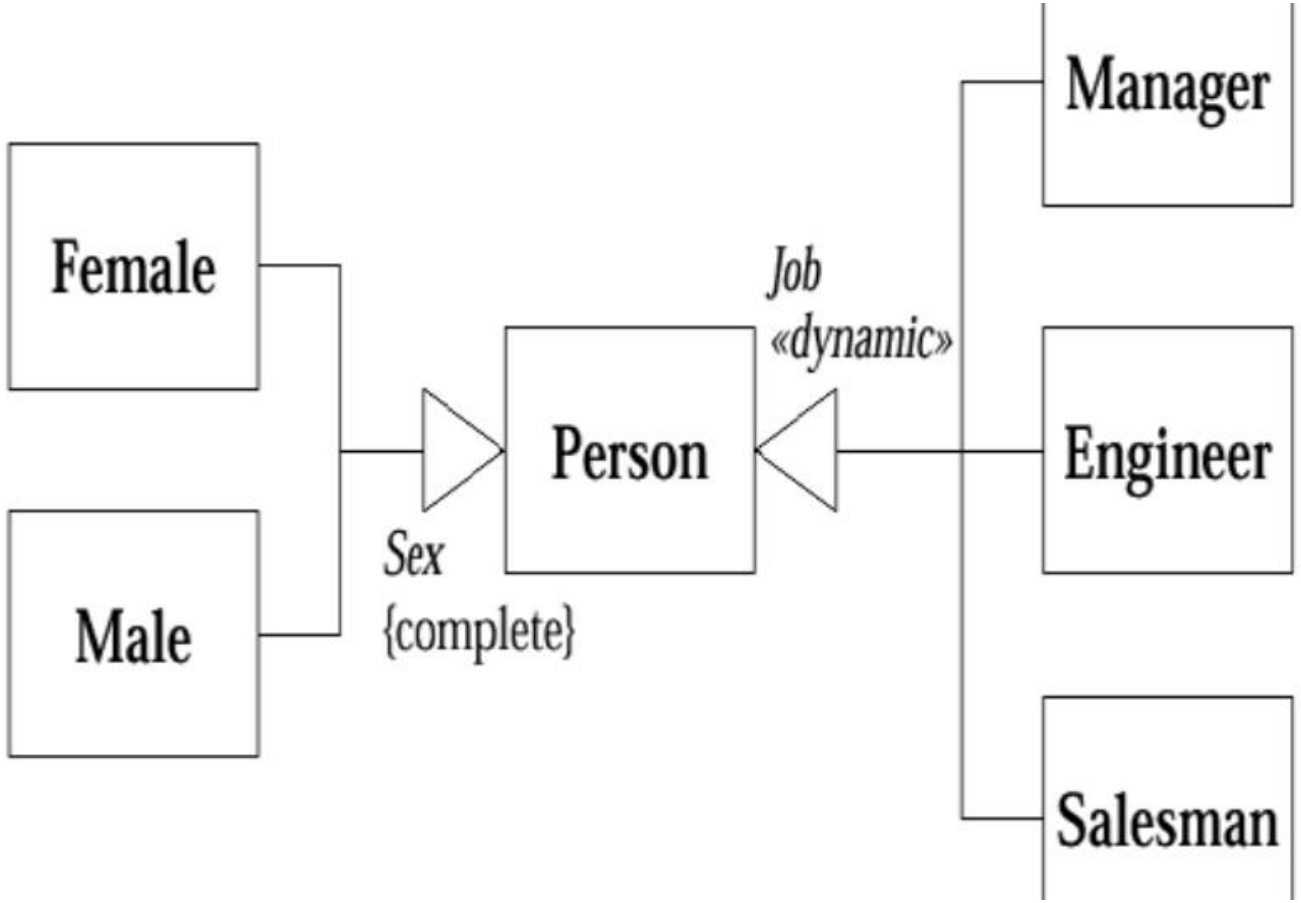

Discriminator
Surgeon
Doctor
Family Doctor
Female
role
Person
Nurse
Male
sex
{complete}
patient
Physio-therapist
Patient

Generalizovani skupovi

- Vrh strelice generalizacije označava se imenom generalizovanog skupa
  - U verziji UML 1 – diskriminator
  - Pretpostavka je da su tipovi u generalizovanom skupu disjunktni
  - Constraint {complete} označava da instanca nad-klase mora biti instanca jedne klase u generalizovanom skupu
- Koje kombinacije su ispravne
  - (Female, Patient, Nurse)
  - (Male,Physiotherapist)
  - (Female, Patient)
  - (Female, Doctor, Surgeon)
  - (Patient,Doctor)
  - (Male, Doctor, Nurse)

Dinamička klasifikacija

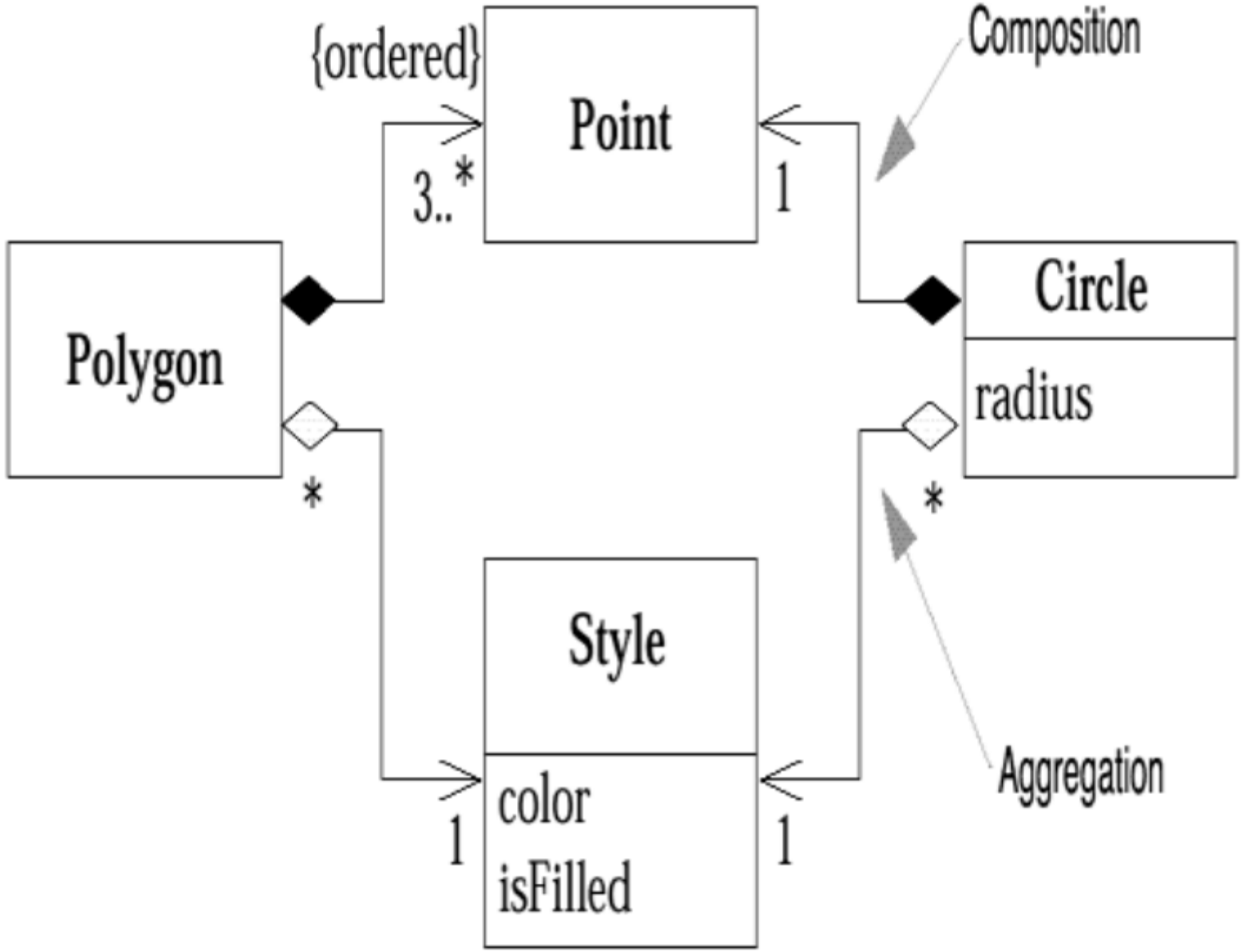
- Dinamička klasifikacija dozvoljava da objekti mijenjaju tip
  - Kombinovanje tipa i stanja
  - Kako na drugi način modelovati dio *Job* *<<dinamic>>*?



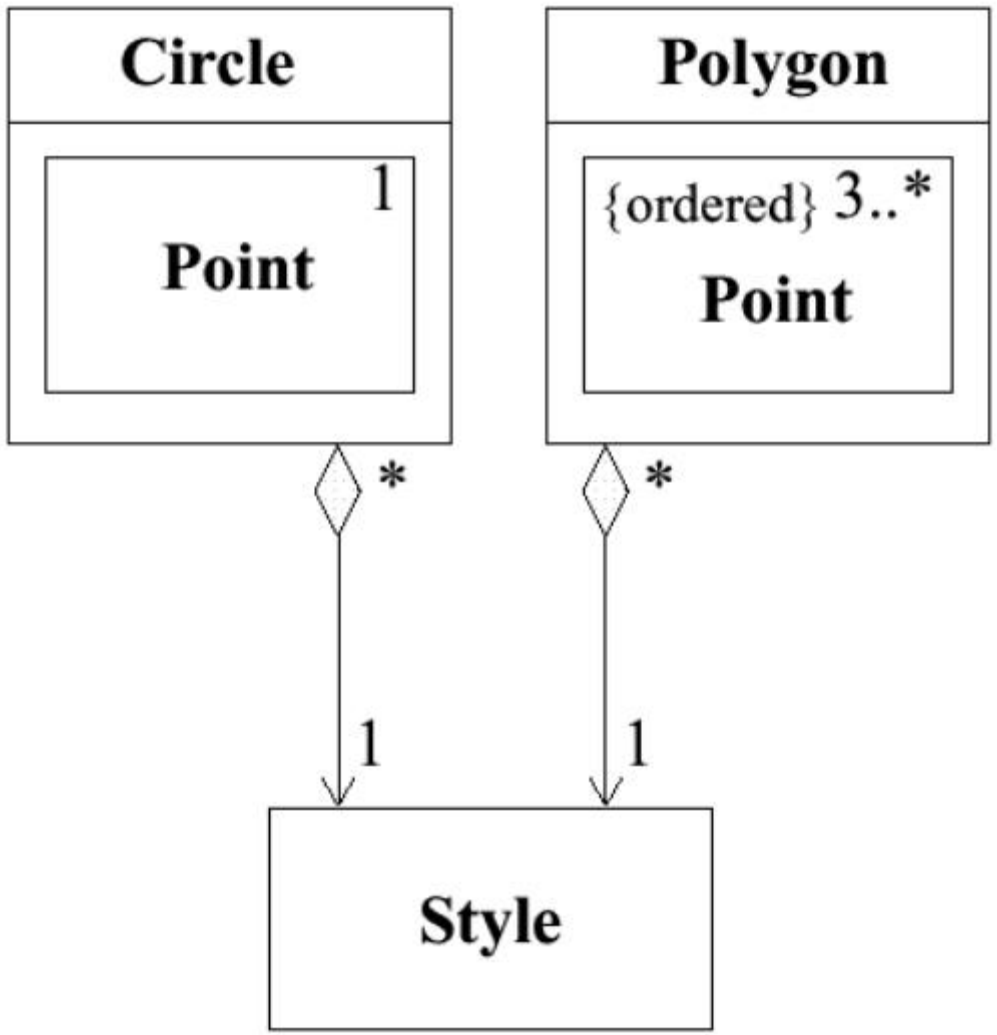
Agregacija i kompozicija

- Agregacija modeluje cjelina-dio (part-of) relaciju
- Kompozicija predstavlja jaču agregaciju
  - Objekat dio pripada tačno jednom objektu cjelina
  - Životni vijek djelova određen je životnim vijekom cjeline
  - Delete cascade

Primjer

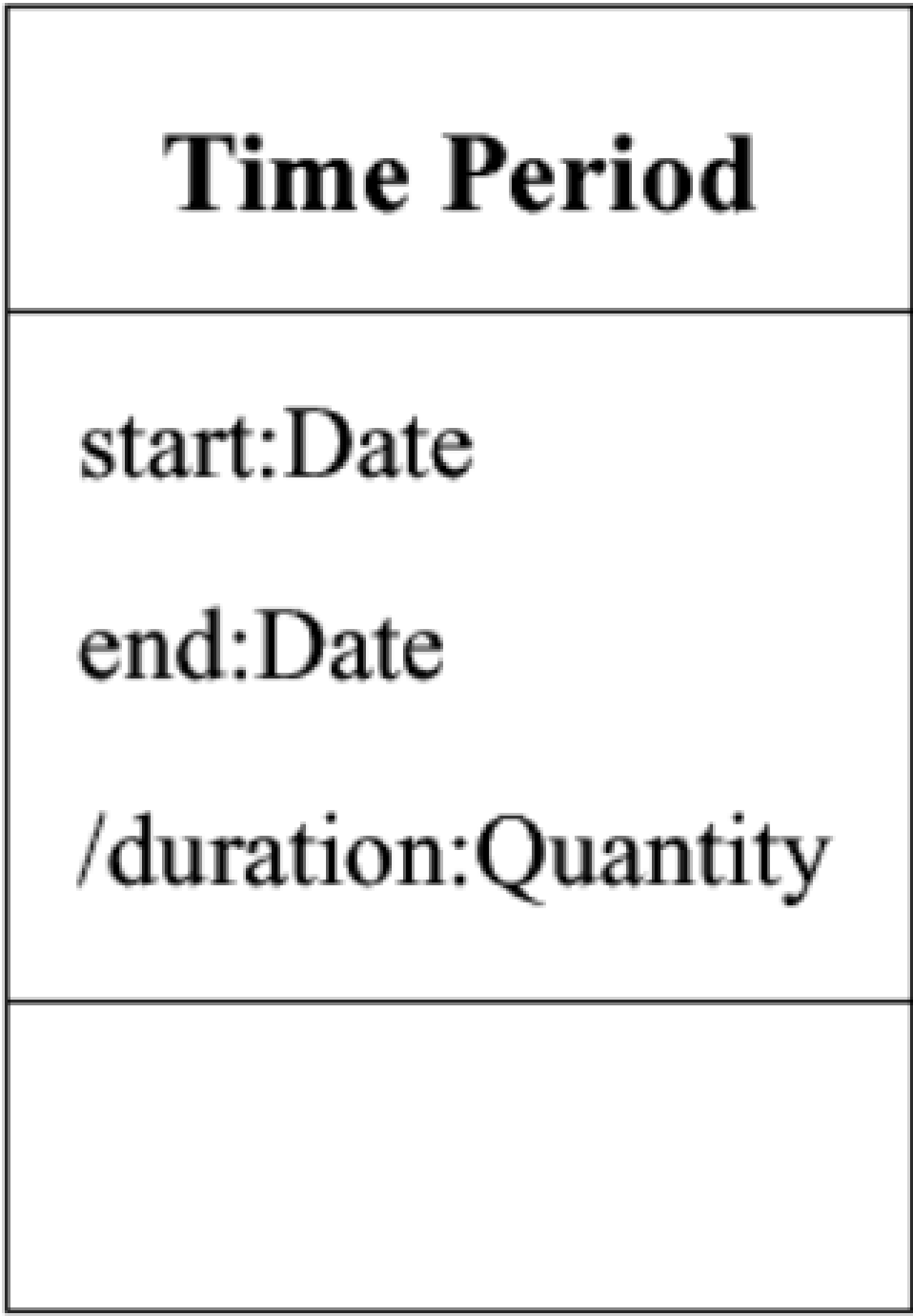

{ordered}
Point
Composition
3..*
1
Polygon
Circle
radius
*
*
Style
color
isFilled
1
1
Aggregation

Primjer 2



Izvedena svojstva

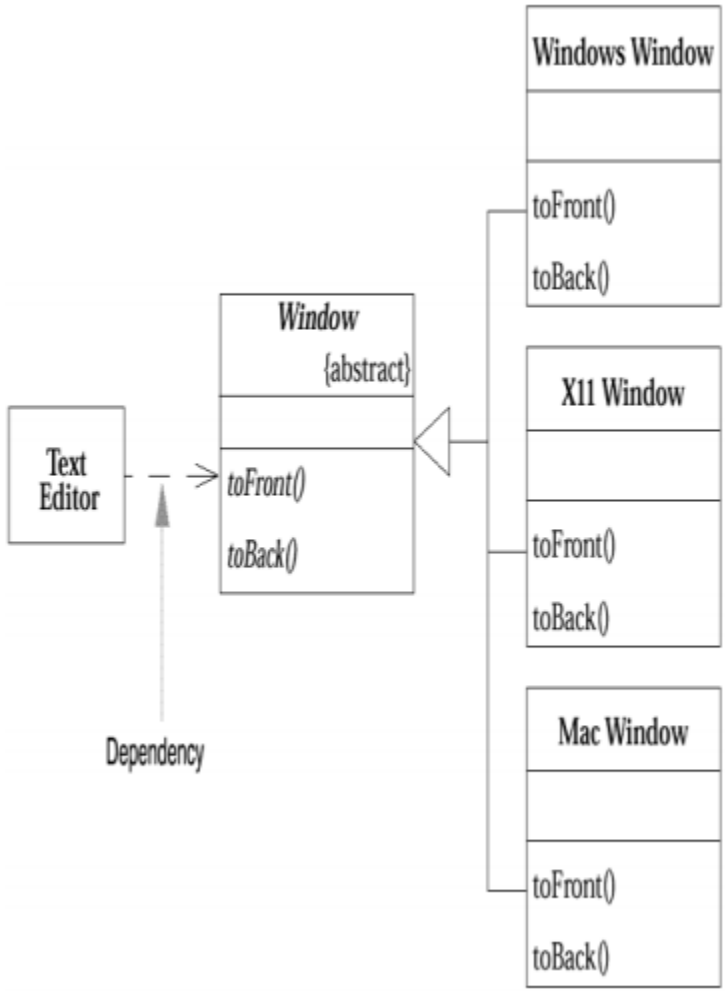
- Izvedena svojstva mogu biti izračunati na osnovu drugih svojstava



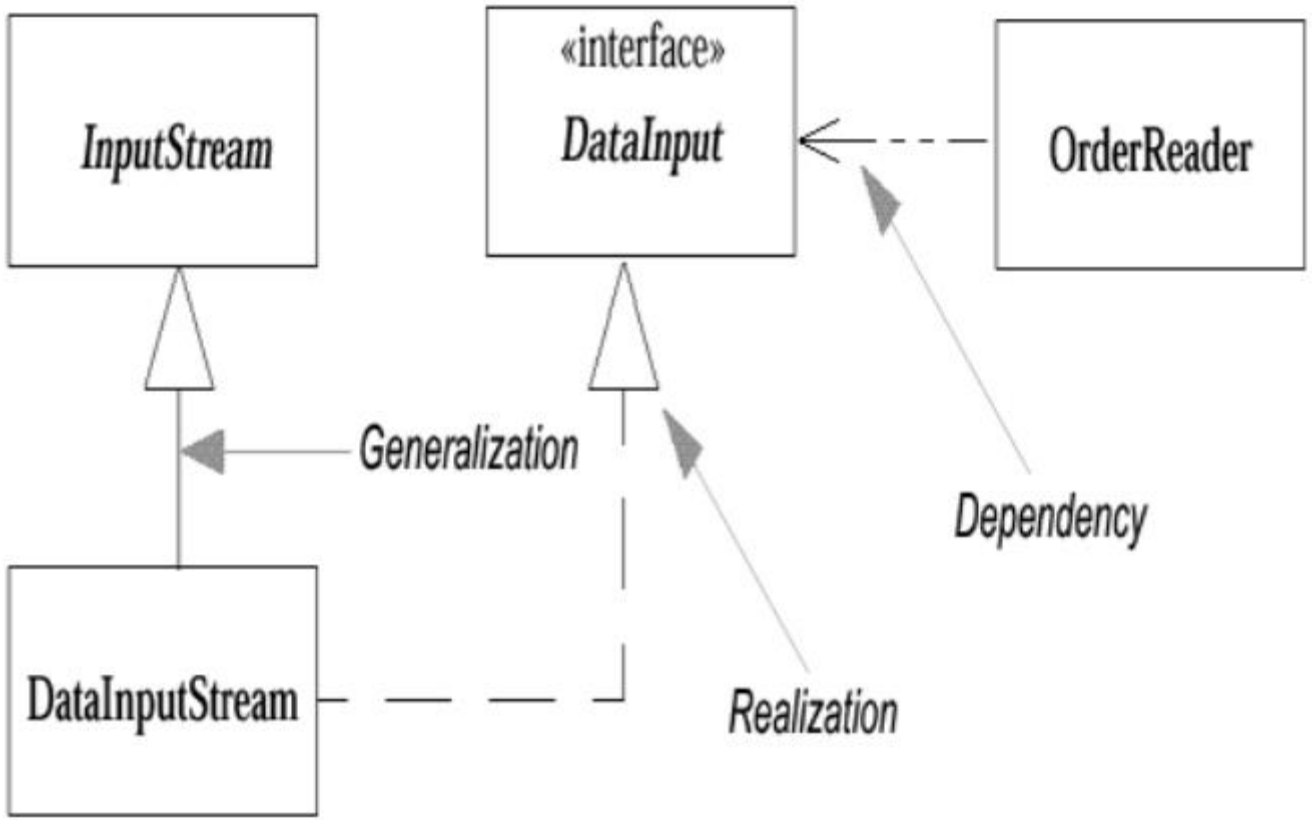
Interfejsi i apstraktne klase

- OO dizajni omogućava da interfejs bude nezavisan od implementacije
- Većina programskih jezika podržava samo klase
  - One sadrže interfejs i implementaciju
  - Apstraktne klase kao interfejsi
    - Naziv klase italic fontom ili {naziv klase}
  - Java podržava interfejs
    - Relacija realizacija označava da jedna klasa realizuje ponašanje definisano u drugoj, slična je konceptualno generalizaciji
    - Kompajler provjerava da li klase koje implementiraju interfejs obezbjeđuju implementacije svih metoda

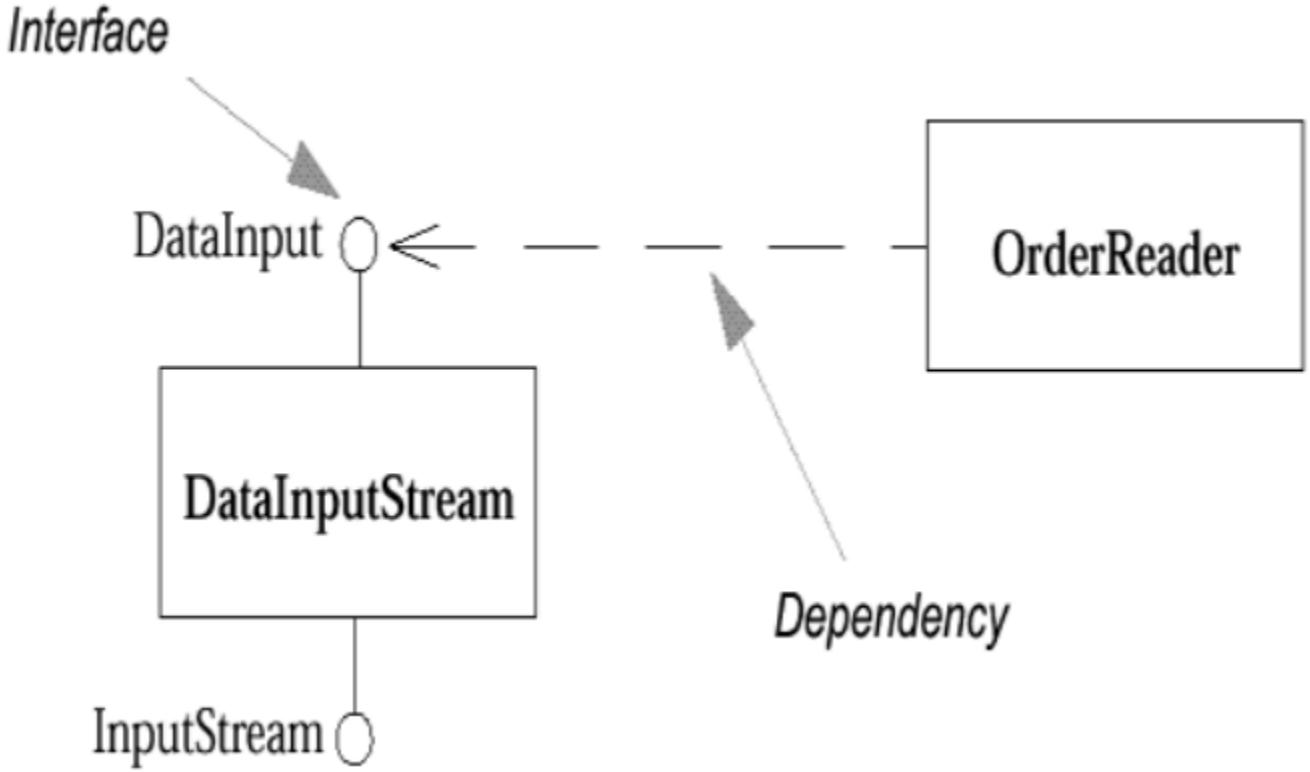
Primjer


Windows Window
toFront()
toBack()
Window
{abstract}
X11 Window
Text Editor
toFront()
toBack()
toFront()
toBack()
Mac Window
toFront()
toBack()
Dependency

Java primjer



Alternativna notacija



Referentni i vrijednosni objekti

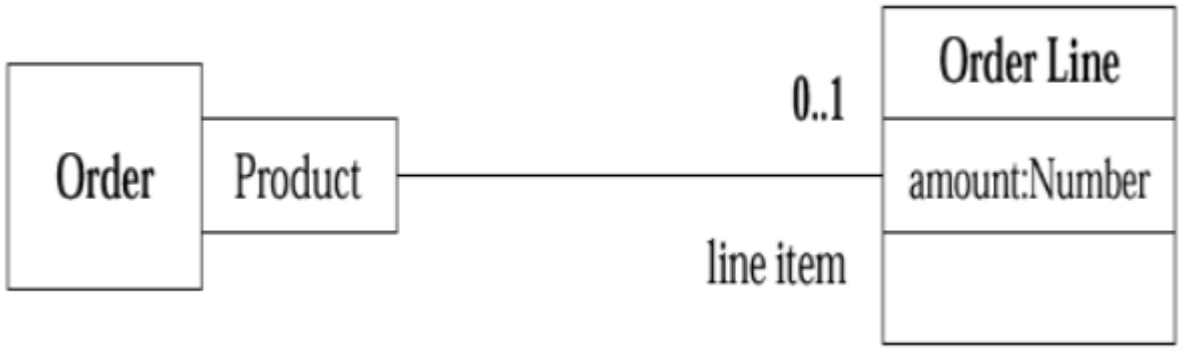
- Referentni objekti imaju identitet i odnose se na jedan objekat iz realnog svijeta
  - Primjer, student
- Vrijednosni objekti predstavljaju više objekata iz realnog svijeta
  - Primjer, datum
  - Upoređuju se po vrijednosti, ne po identitetu
  - Obično su nepromjenljivi (frozen)
    - Frozen vs. read-only

Kolekcije

- Vezene za asocijacije sa multiplikativnošću *više* *
  - Podrazumijevano se predstavlja skupom
  - Constraint
    - {ordered} – implementacija listom
    - {bag} – multiskup, bez uređenja
    - {hierarchy}
    - {dag} – directed acyclic graph

Asocijacije opisane kvalifikatorom

- UML ekvivalent sa asocijativnim nizom ili dictionaty tip podataka u programskim jezicima

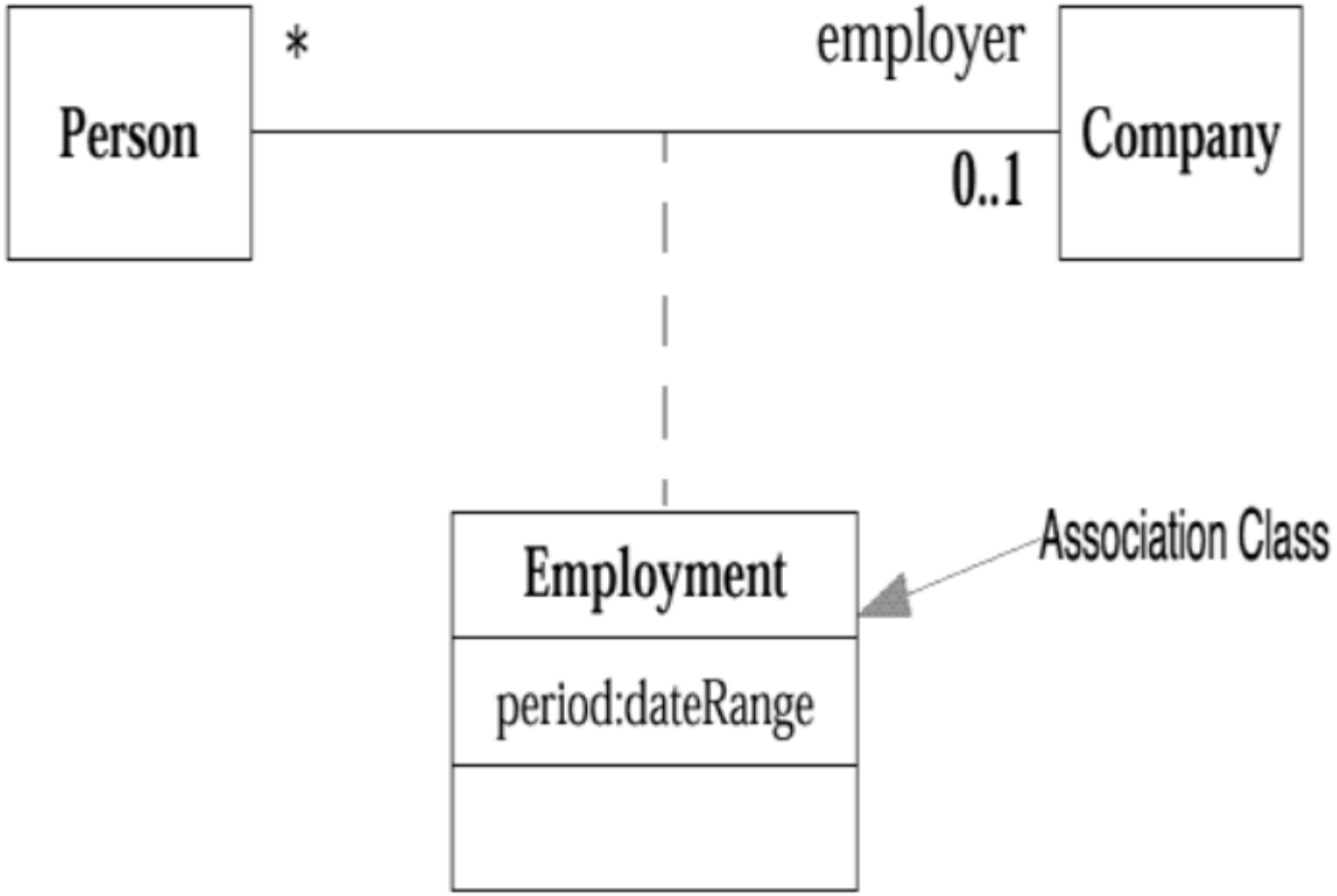



```
class Order {

    public OrderLine getLineItem
        (Product aProduct);
    public void addLineItem
        (Number amount, Product forProduct);
```

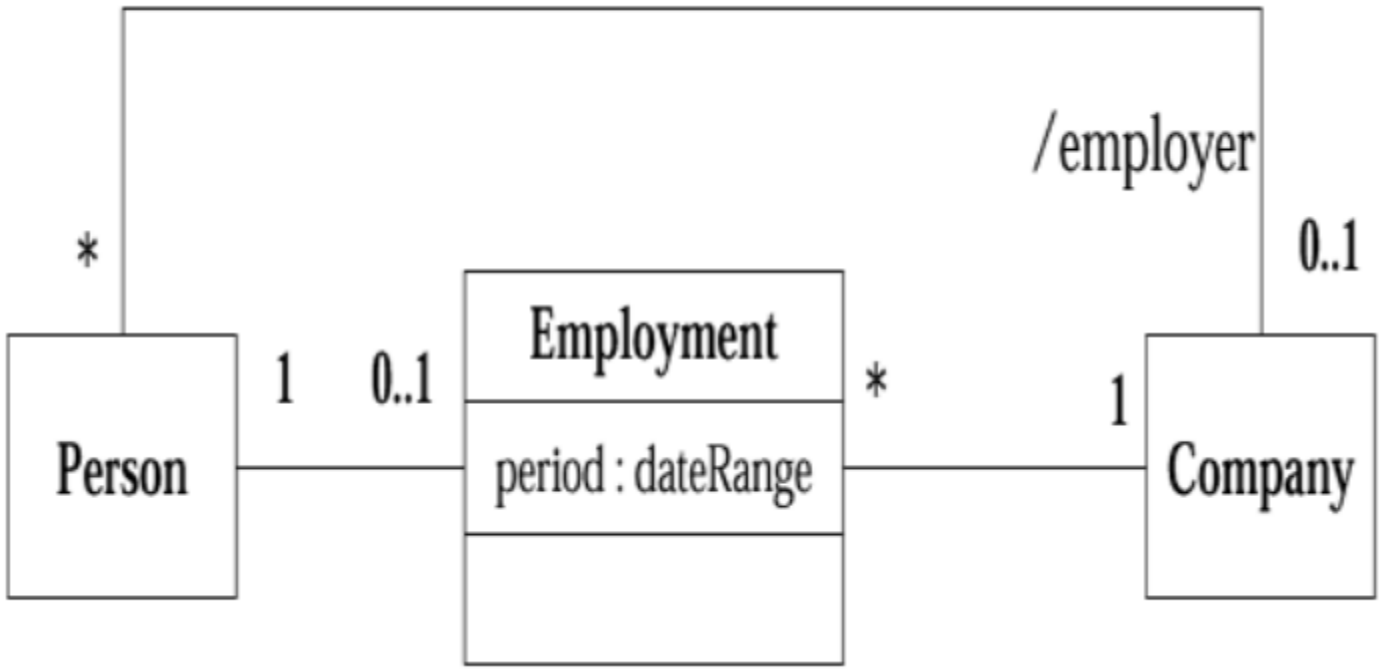
```
Class Order {

    private Map _lineItems;
```

Klasa asocijacija

- Klasa asocijacija omogućava da asocijaciji budu pridruženi atributi i/ili metodi

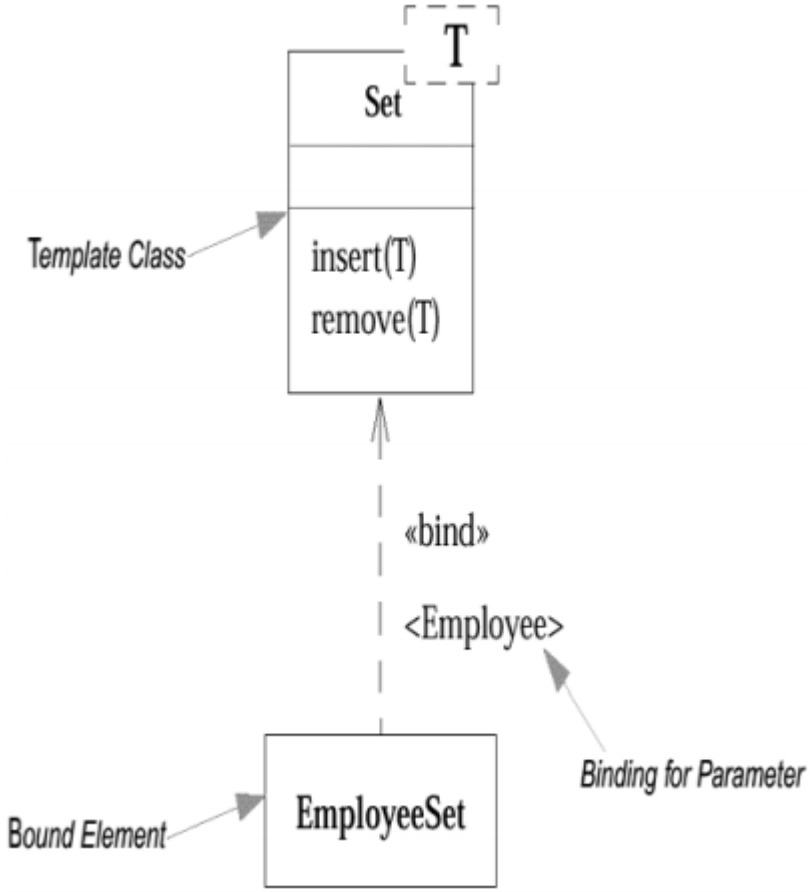


Alternativna reprezentacija



Parametarizovane klase

- U jeziku C++ parametarizovana ili šablonska klasa je template




```
class Set <T> {
    void insert (T newElement);
    void remove (T anElement);

Set <Employee> employeeSet;
```