2024/2025

Računalništvo04
IS, UML, PB, SQL, ER

predavatelj: Aleksandar Lazarević

Agenda

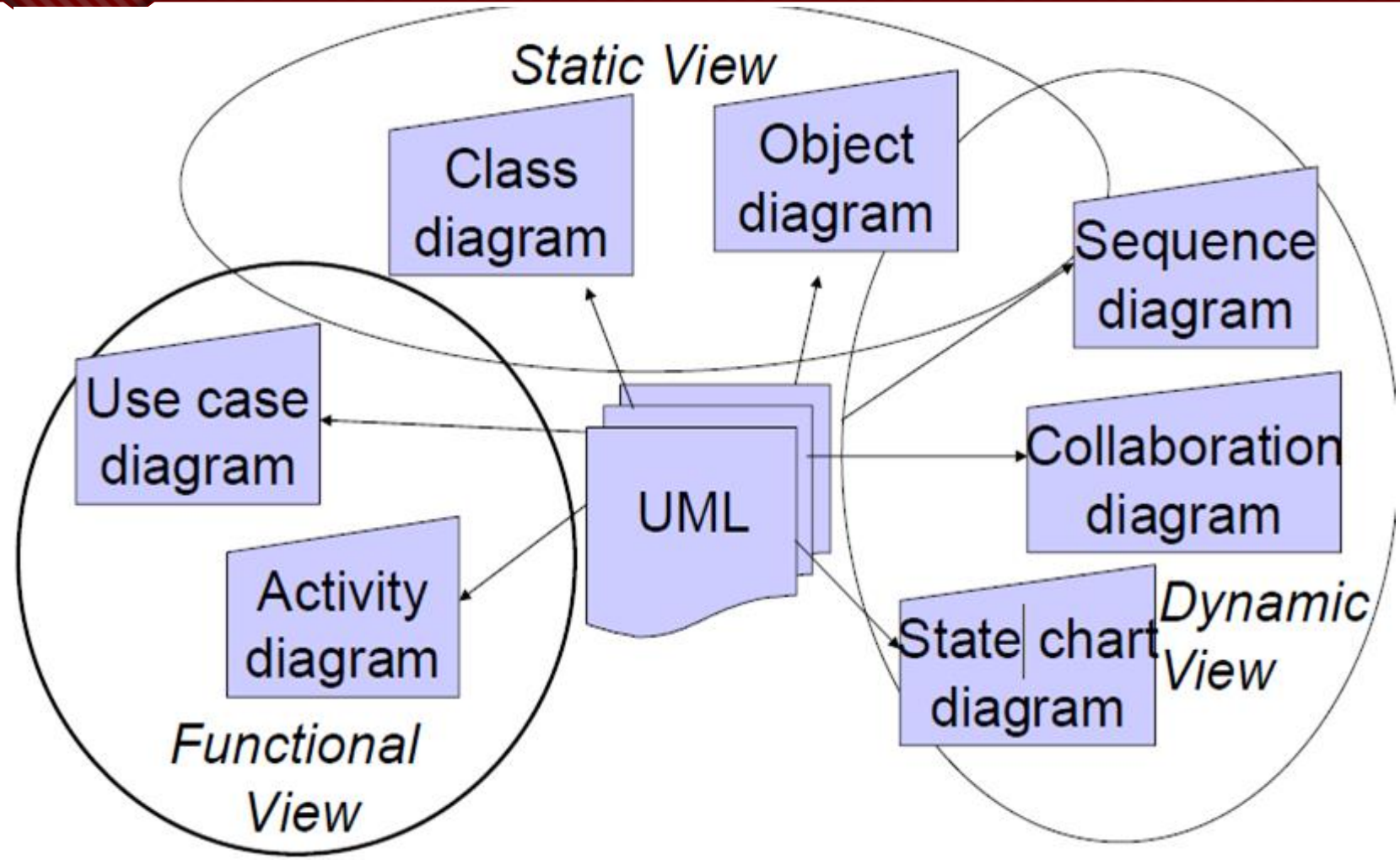
- UML
- Diagrami primerov uporabe
- Razredni diagrami
- Diagrami aktivnosti

2024/2025

UML

UML: diagrami primerov uporabe, diagrami aktivnosti, razredni diagrami

UML vidiki


Static View
Class diagram
Object diagram
Sequence diagram
Use case diagram
UML
Collaboration diagram
Activity diagram
State chart diagram
Dynamic View
Functional View

UML 2.5.1

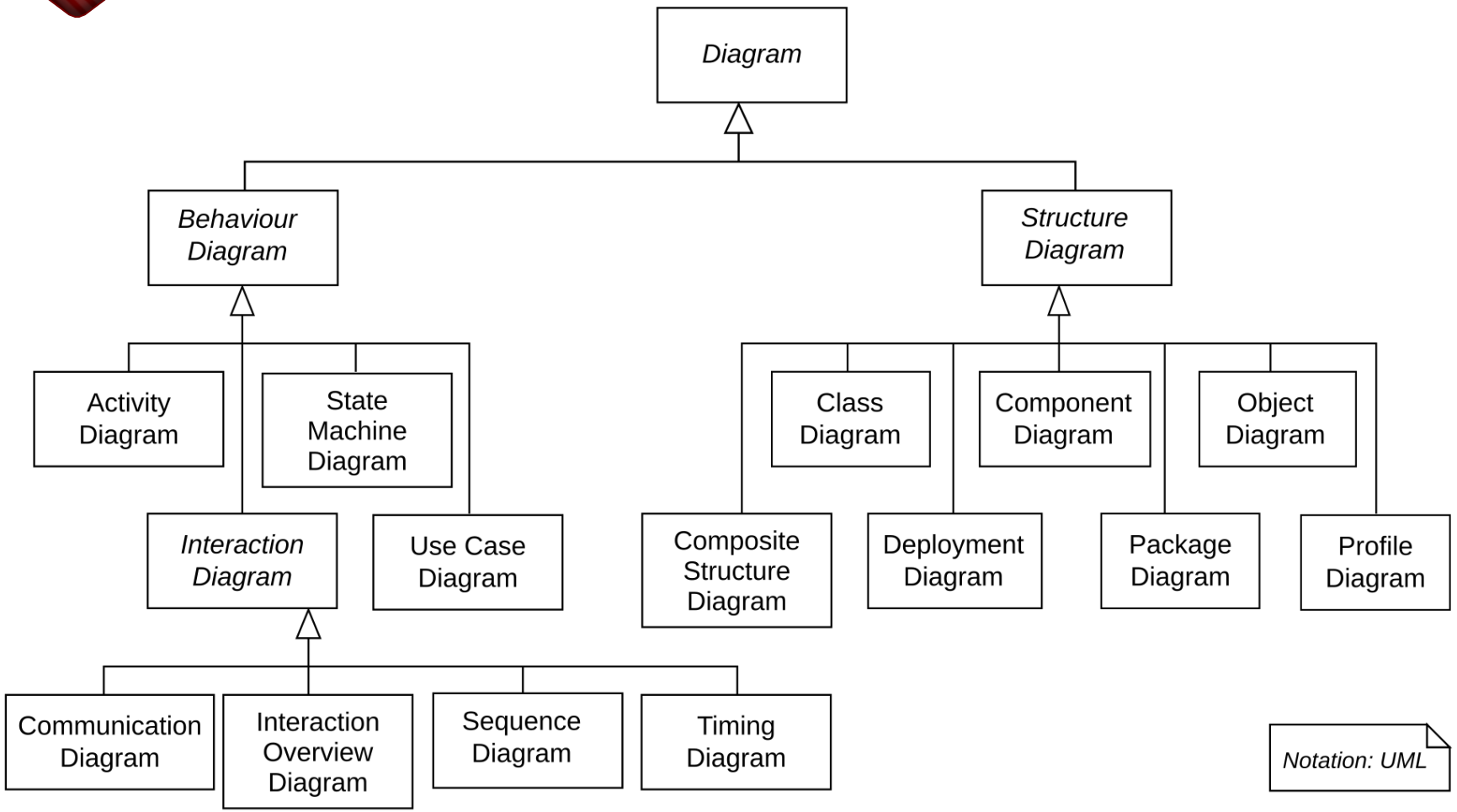
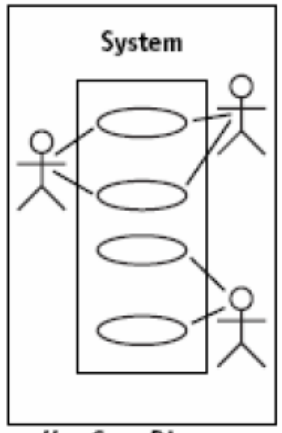


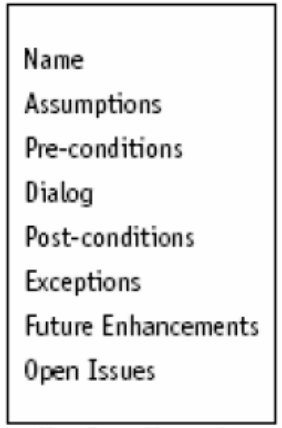
Diagram
Behaviour Diagram
Structure Diagram
Activity Diagram
State Machine Diagram
Interaction Diagram
Use Case Diagram
Class Diagram
Component Diagram
Object Diagram
Composite Structure Diagram
Deployment Diagram
Package Diagram
Profile Diagram
Communication Diagram
Interaction Overview Diagram
Sequence Diagram
Timing Diagram
Notation: UML

Funkcionalni vidik

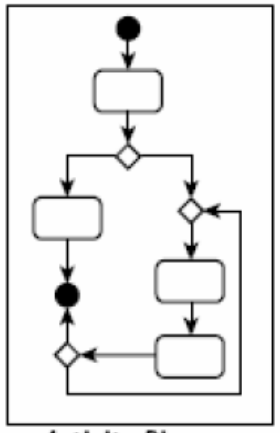
- Opisuje kako naj sistem dela.
- *Use Case diagram*
  - opisuje lastnosti sistema, ki jih pričakuje uporabnik.
- *Activity diagram*
  - opisuje procese sistema kot niz nalog, ki morajo biti opravljene, pogojev in vzporednih obdelav (podobno kot flowchart).



Use Case Diagram



Use Case Narrative

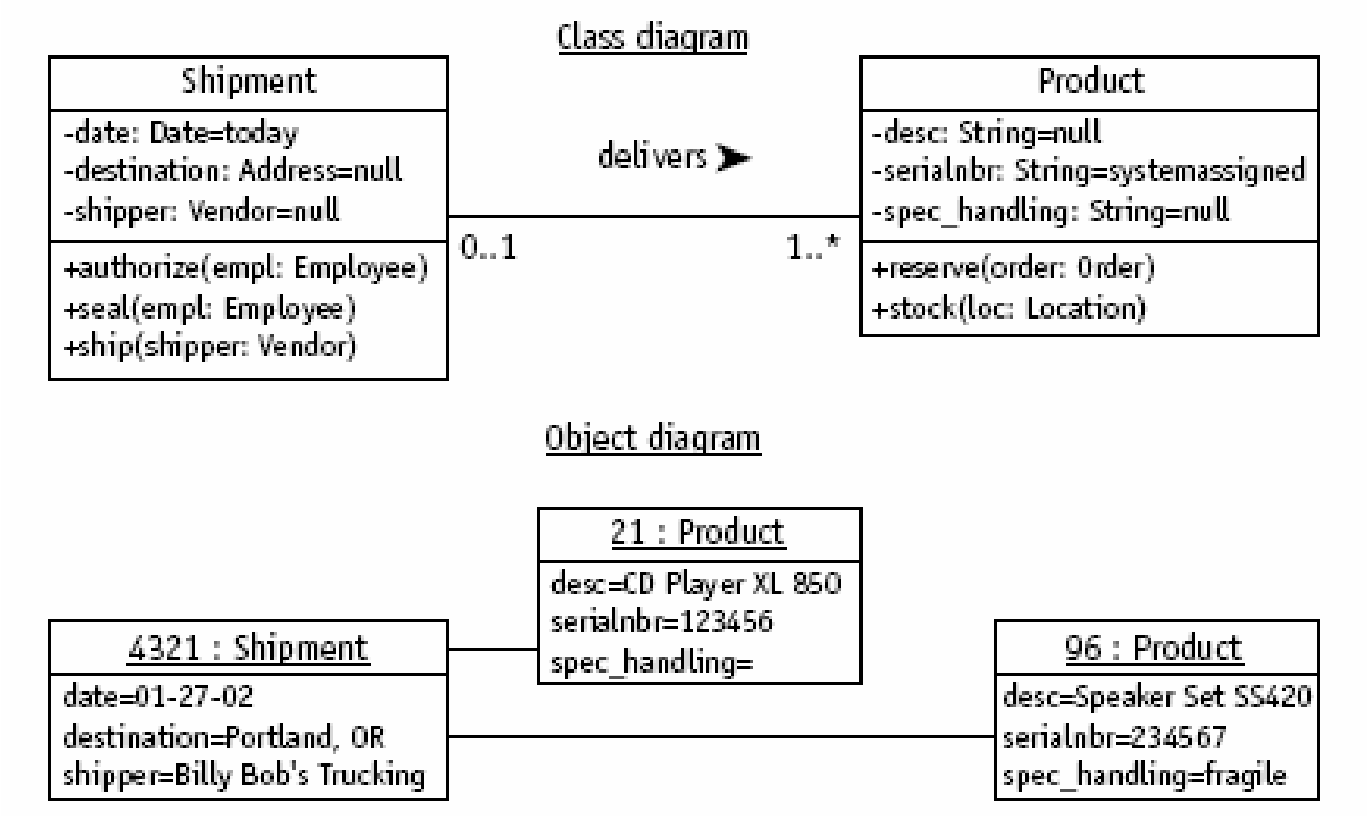

Activity Diagram

Statični vidik

- Opisuje gradbene elemente sistema ampak ne način kako med seboj sodelujejo.

- Tipični predstavniki so:
  - *Class diagram*
    - pogled na vse vire (razrede) in njihove lastnosti. Skoraj vedno se uporablja za generiranje kode in reverzni inženiring.
  - *Object diagram*
    - konkretizira splošno sliko sistema prikazano z razrednim diagramom.

Statični vidik

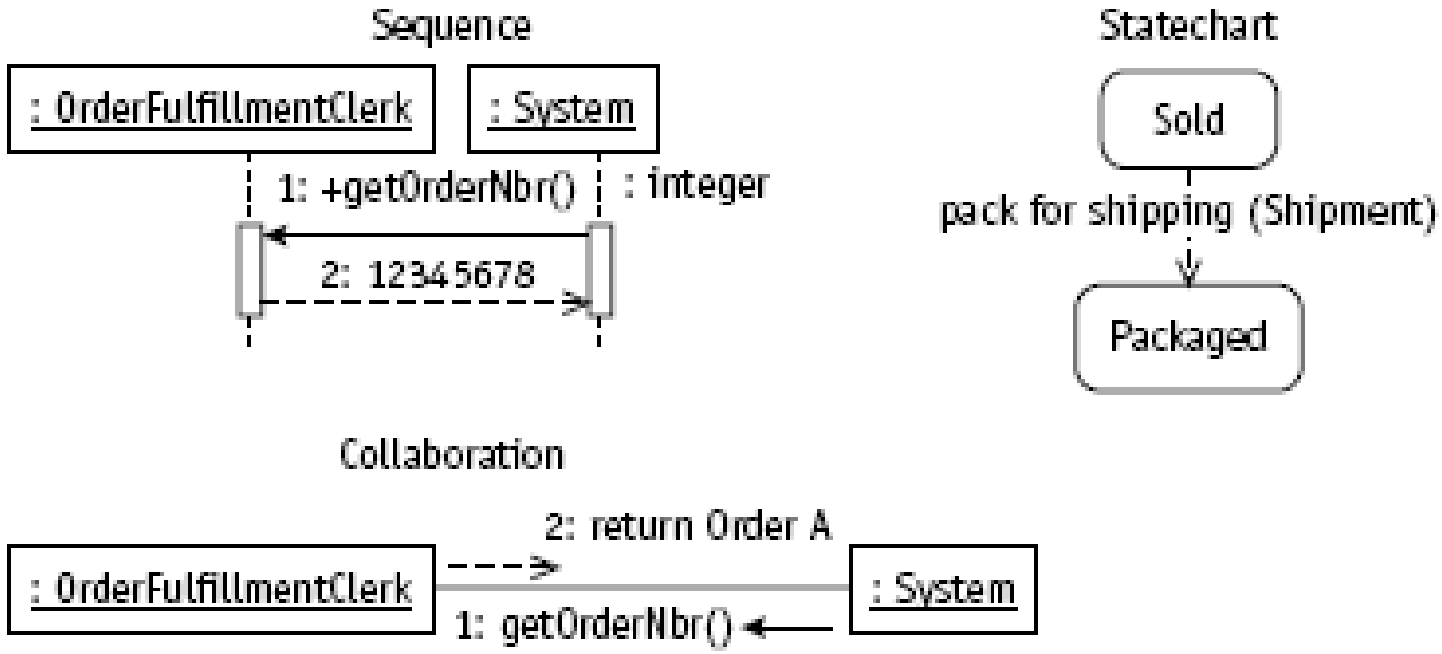


Dinamični vidik

- Določa obnašanje sistema – medsebojno delovanje elementov sistema, kot odgovor na spodbude iz okolja.

- *Sequence and collaboration* diagrami
  - opisujejo interakcije.
- *State chart* diagram
  - kako i zakaj se objekti spreminjajo, kot odgovor na spodbude iz okolja.

Dinamični vidik



2024/2025

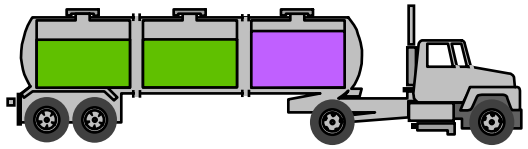
Razredni diagrami
(ang. *class diagrams*)

Osnovni gradniki razrednih diagramov: razred, atributi, metode, dostopnost, povezave; abstrakcija, generalizacija

...

UML - objekt

- Neformalno, objekt predstavlja fizičen, konceptualni ali programski pojem (entiteto).
  - fizični pojem



Tovornjak

  - konceptualni pojem

Kemijski proces

  - programski pojem

Povezan seznam

UML - objekt

: Professor

Samo ime razreda

ProfessorClark

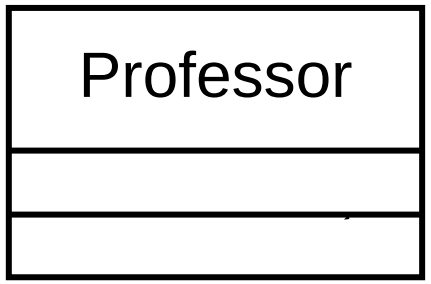
Samo ime objekta

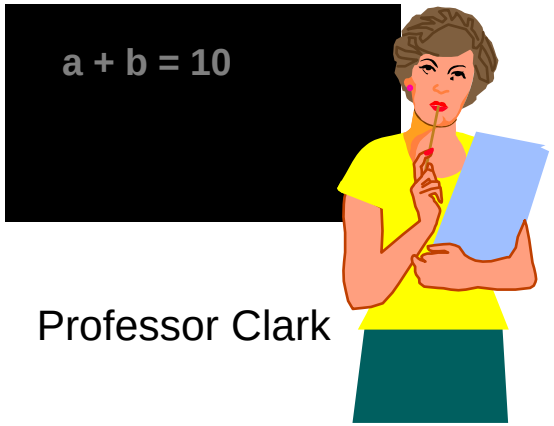
- Formalna definicija: objekt je koncept, abstrakcija, ali stvar z natančno določenimi mejami in pomenom za aplikacijo.
- Objekt je nekaj, kar ima: stanje, obnašanje, identiteto.
- Objekt je predstavljen s pravokotnikom in podčrtanim imenom.

(definicija razreda sledi)

UML - razred

- Razred je opis skupine objektov z enakimi lastnostmi (atributi), enakim obnašanjem (operacije), povezavami, in semantiko (pomenom).
  - objekt je primerek razreda.
- Razred je abstrakcija, ki:
  - poudarja pomembne karakteristike in izpusti ostale (nepomembne) karakteristike.
- Razred je predstavljen s pravokotnikom.

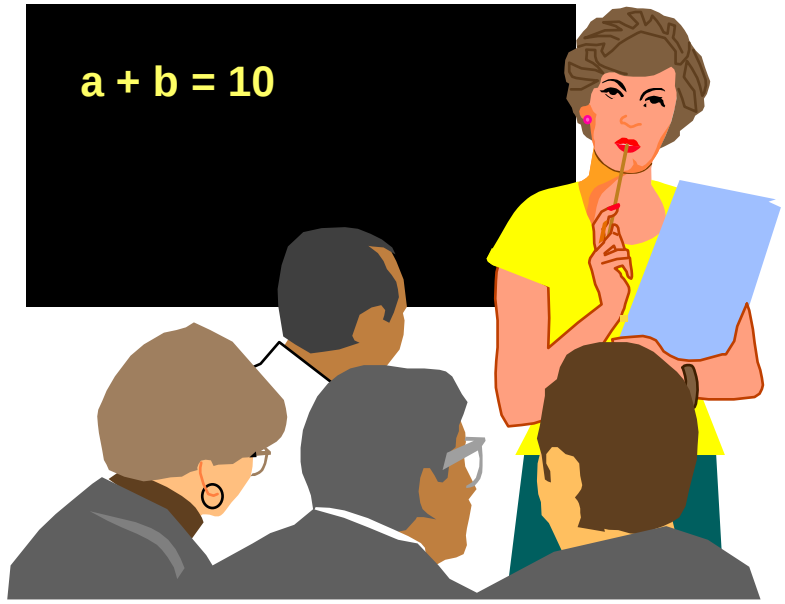




UML - razred

Razred
študij

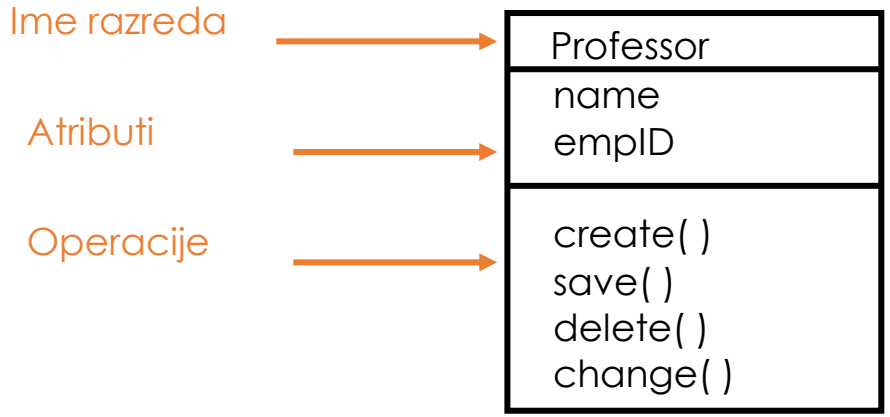
Lastnosti
ime
lokacija
št. dni
št. ur
začetek
konec



Obnašanje
Dodaj študenta
Zbriši študenta
Preglej seznam študentov
Ugotovi ali je seznam poln

UML - razred

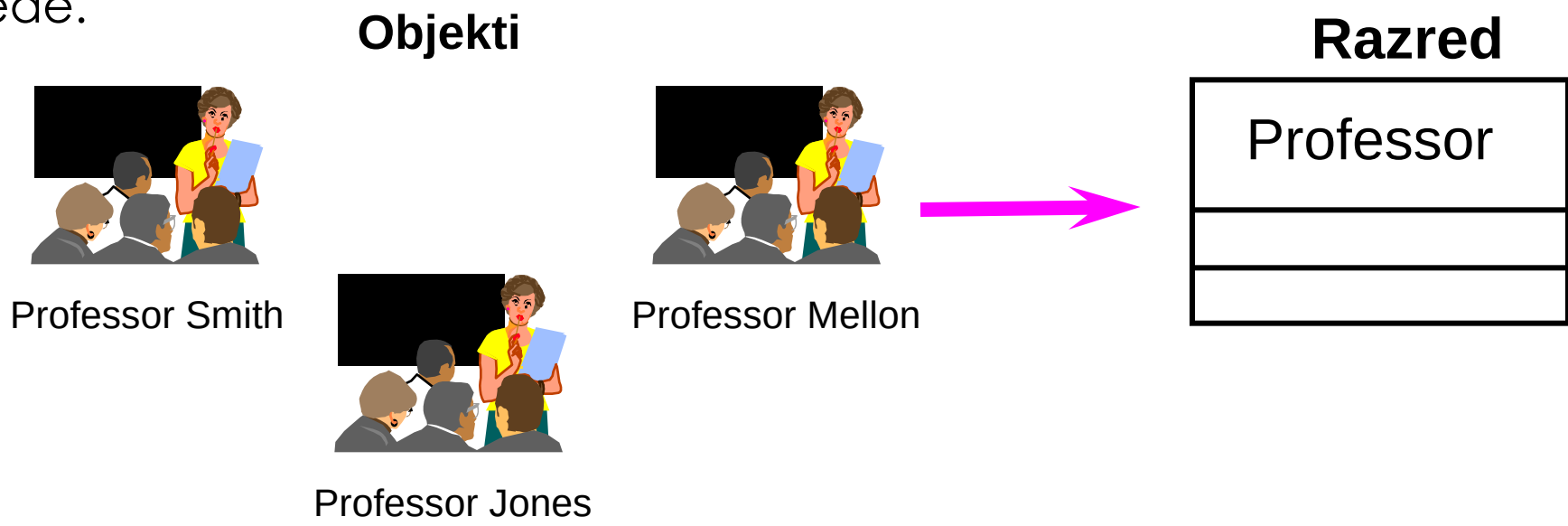
- Razred vsebuje tri razdelke:
  - prvi razdelek vsebuje ime razreda.
  - drugi razdelek prikazuje strukturo (atribute).
  - tretji razdelek prikazuje obnašanje (operacije).



UML – povezava med razredi in objekti

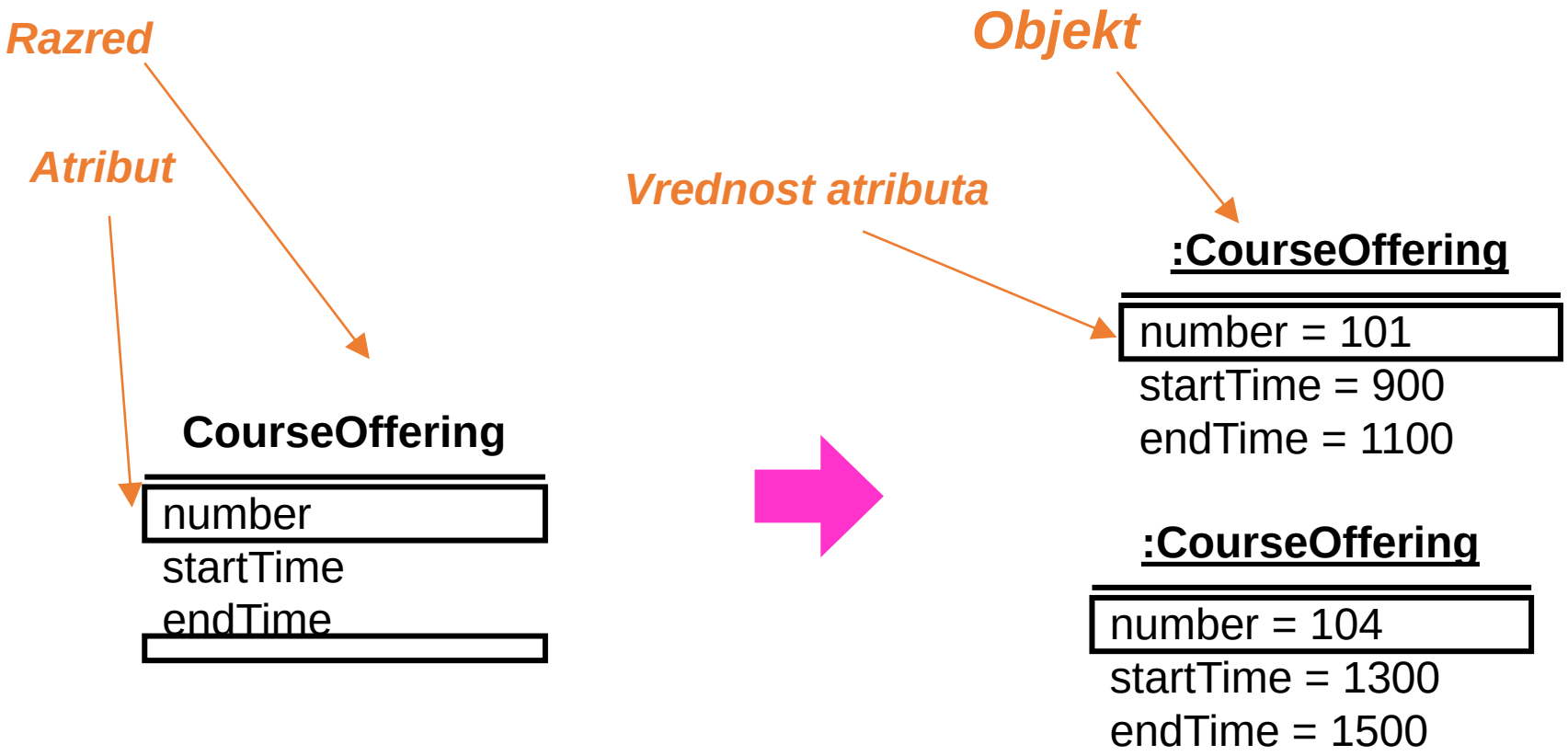
- Razred je abstraktna definicija objekta:
  - definira strukturo in obnašanje vsakega objekta v razredu,
  - služi kot predloga za kreiranje objektov .

- Objekti so grupirani v razrede.



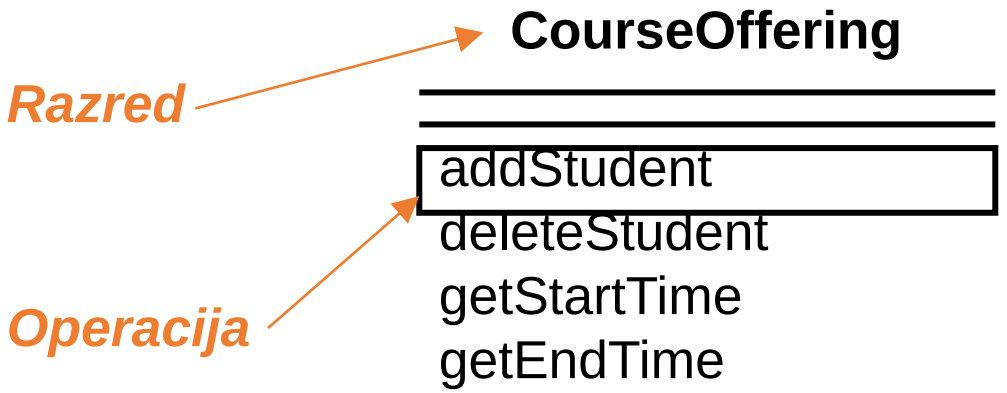
UML - atribut

- Atribut je poimenovana lastnost razreda, ki opisuje obseg vrednosti, ki jih primerki lastnosti lahko zavzamejo.
- Atribut je določenega tipa, ki definira tip njegovih primerkov.
- Stanje objekta je definirano z vrednostmi njegovih atributov.



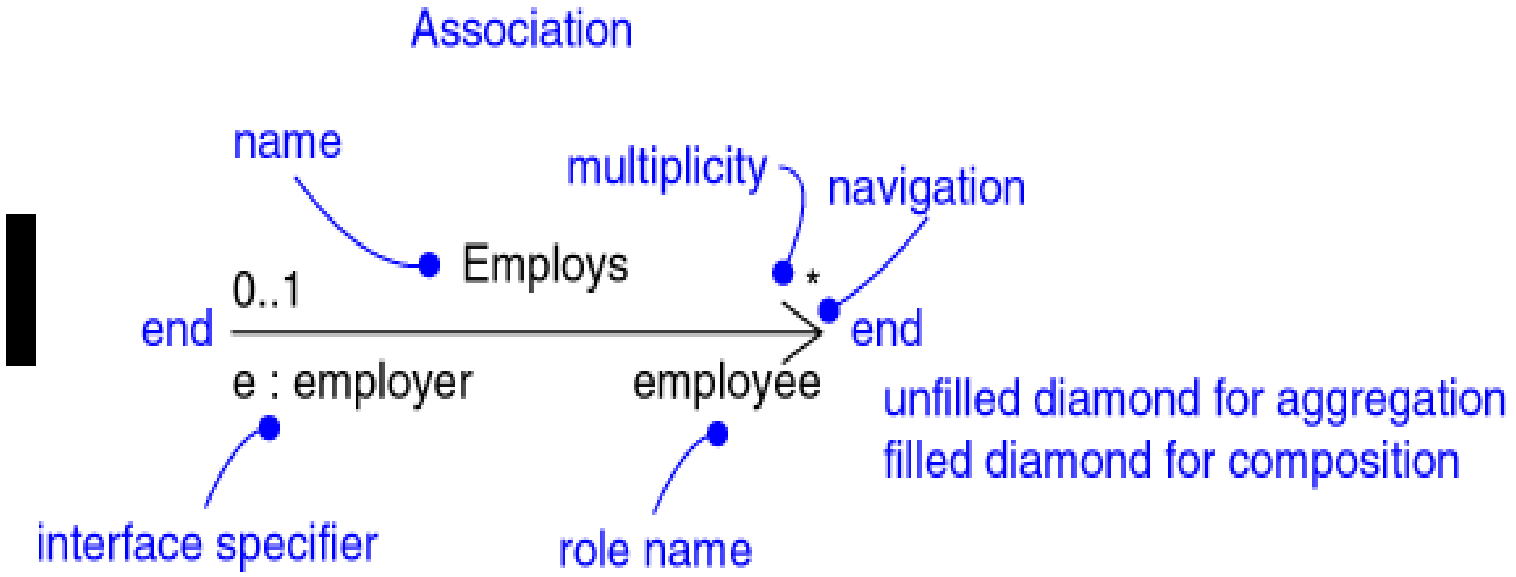
UML - operacija

- Operacija je implementacija storitve, ki jo lahko zahteva katerikoli objekt razreda, da vpliva na obnašanje.
- Operacija je lahko ukaz ali vprašanje. Vprašanje ne sme nikoli spremeniti stanja objekta. Samo ukaz lahko spremeni stanje objekta.
- Rezultat operacije je odvisen od trenutnega stanja objekta.



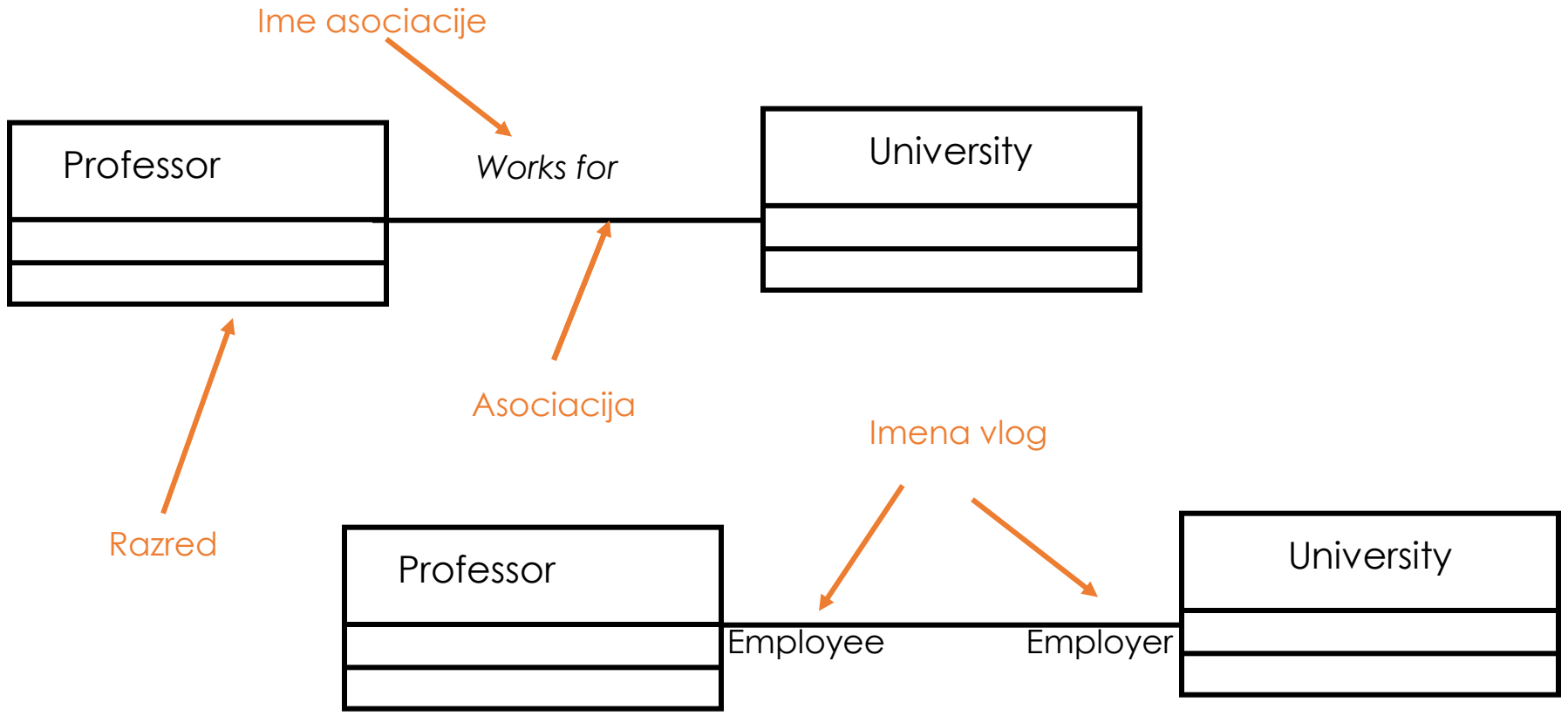
UML – povezave

- Povezave zagotavljajo komunikacijsko pot med različnimi objekti
- Modelirane s črtami, pri čemer različne vrste črt predstavljajo različne vrste povezav:
  - Asociacija
    - Agregacija
    - Kompozicija
  - Odvisnost
  - Generalizacija
  - Realizacija

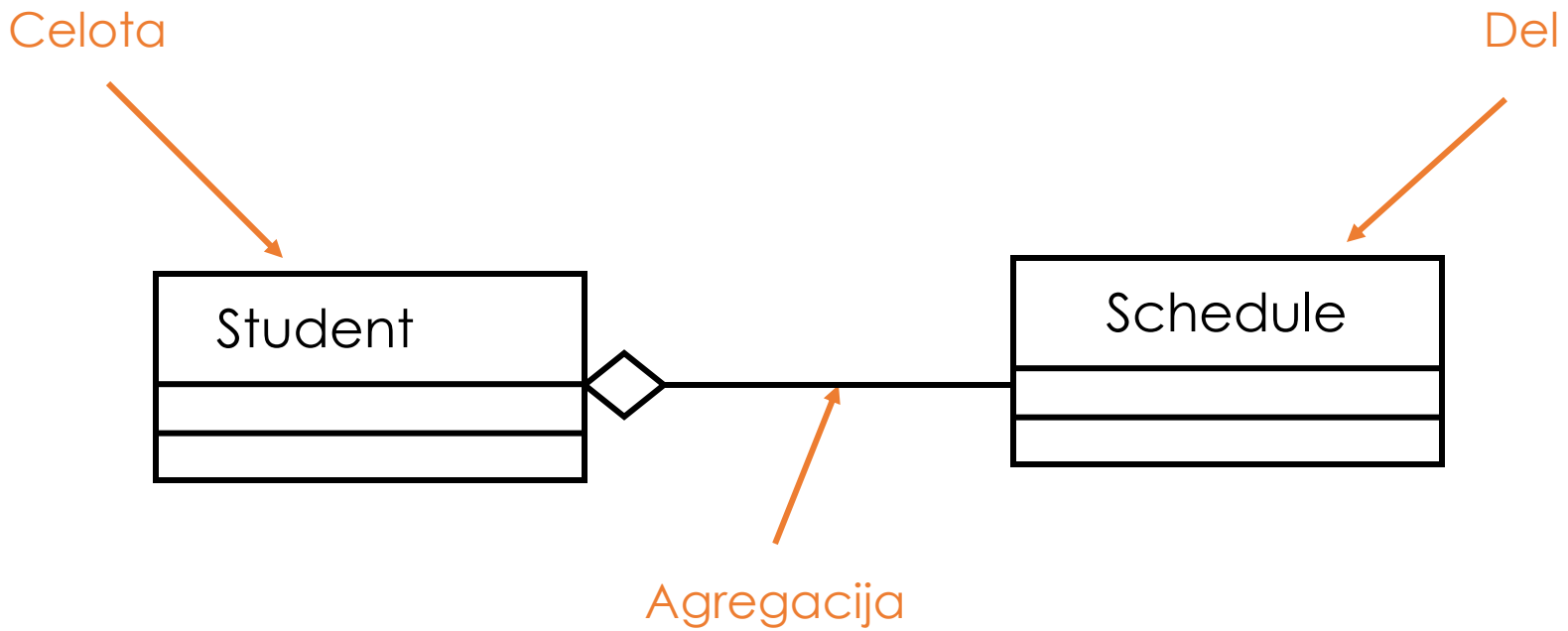


UML – asociacija

- Asociacija je dvosmerna povezava med razredoma – modelira pomensko povezavo.
- Predstavlja strukturni odnos med objektoma različnih razredov.
- Ima lahko ime, vsaka stran asociacije pa vlogo.

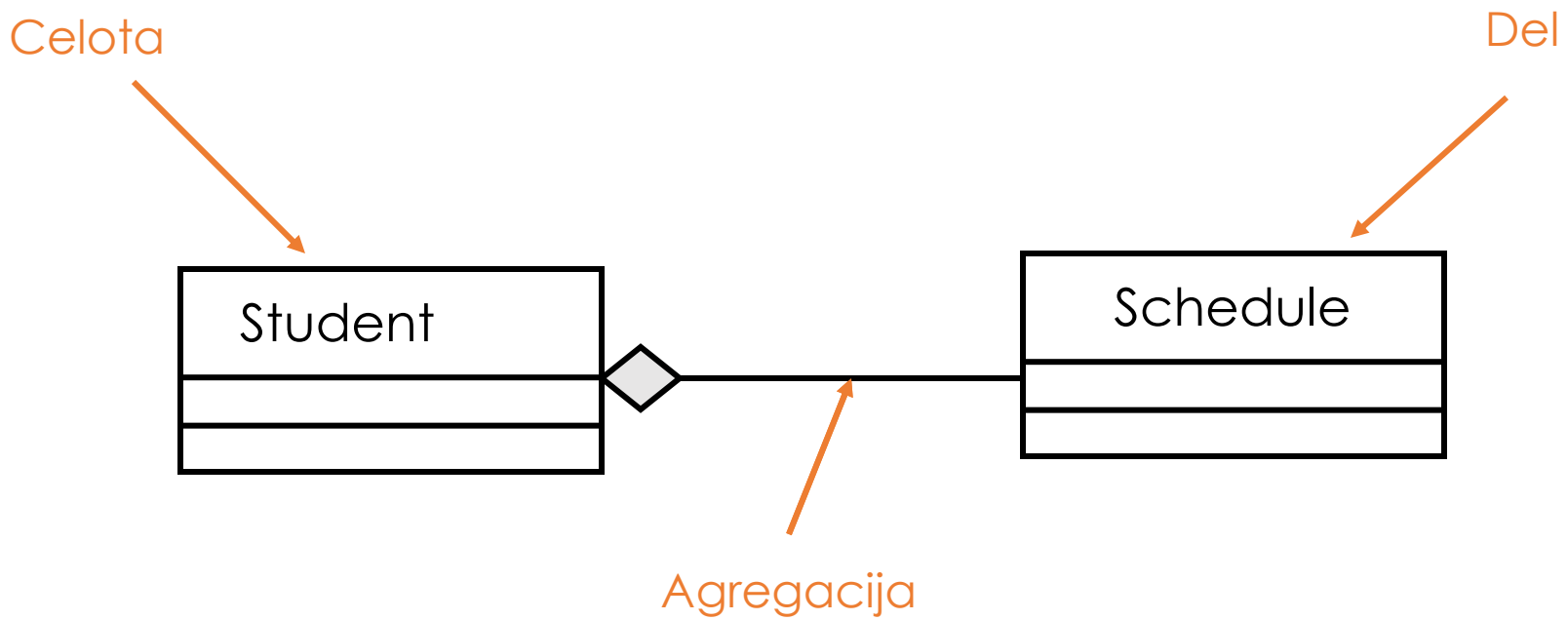


UML – agregacija



- Posebna oblika asociacije, ki modelira povezavo “celota-del” med agregatom (celota) in njenimi deli.
- Strožja oblika povezave.

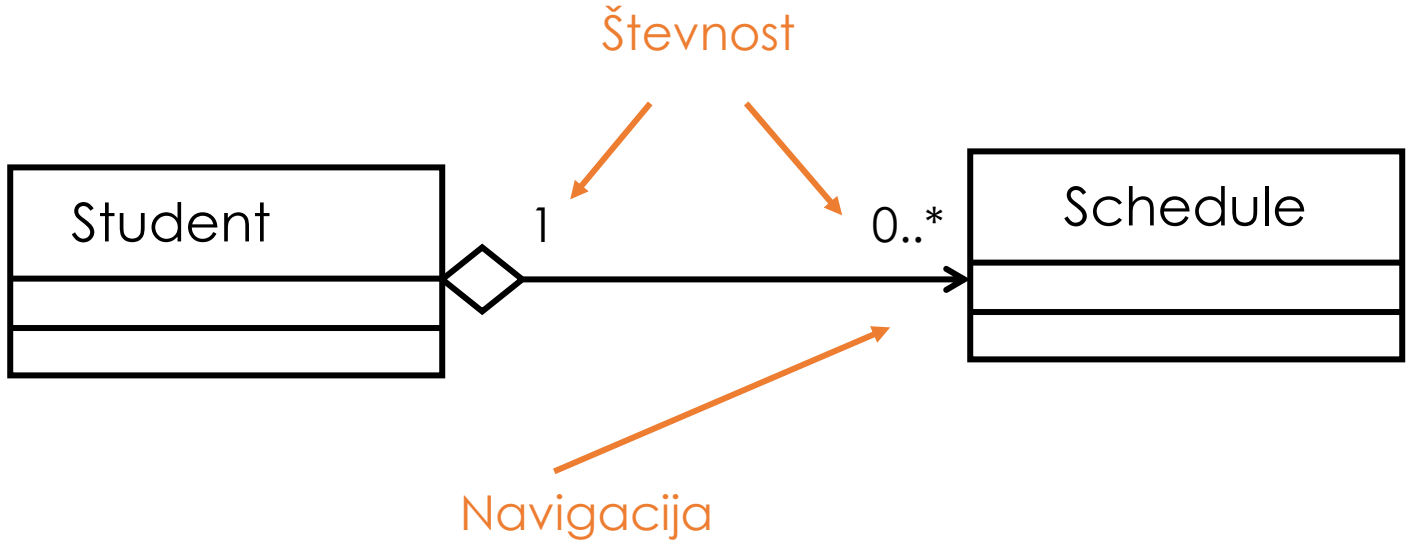
UML – kompozicija



- Oblika agregacije z močnim lastništvom in enako življenjsko dobo kot celota:
  - deli ne morejo *živeti* dlje kot celota/agregat.

UML – asociacija - številnost in navigacija

- Številnost določa koliko objektov je udeleženih v povezavi:
  - število primerkov enega razreda, ki so v povezavi z ENIM primerkom drugega razreda.
  - podana za obe strani (konca) asociacije.
- Asociacija in agregacija sta privzeto dvo-smerni, vendar je zaželeno omejiti navigacijo v eno smer:
  - če je navigacija določena, dodamo puščico, ki kaže smer navigacije.



UML – asociacija - številnost

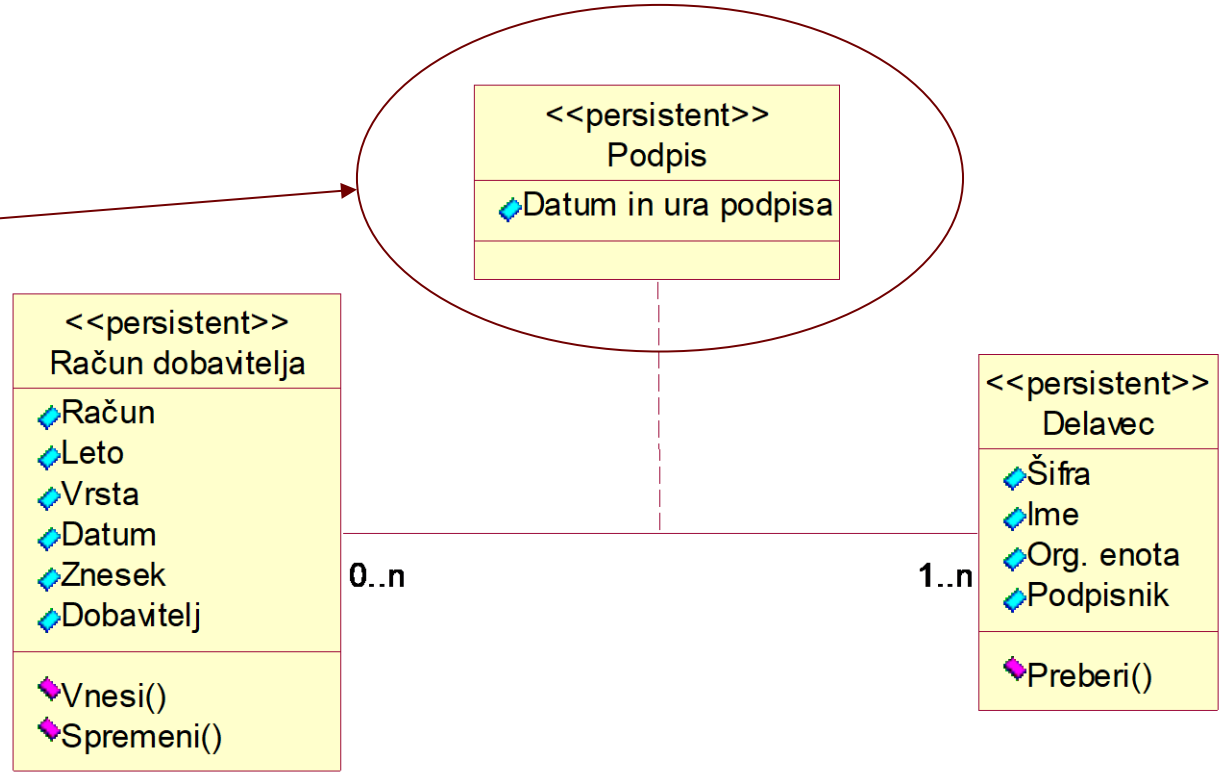
- Nedoločena
- Natanko en
- Nič ali več (več, neskončno)

- Eden ali več
- Nič ali eden
- Določeno območje
- Več, razčlenjenih območij

1
0..*
*
1..*
0..1
2..4
2, 4..6

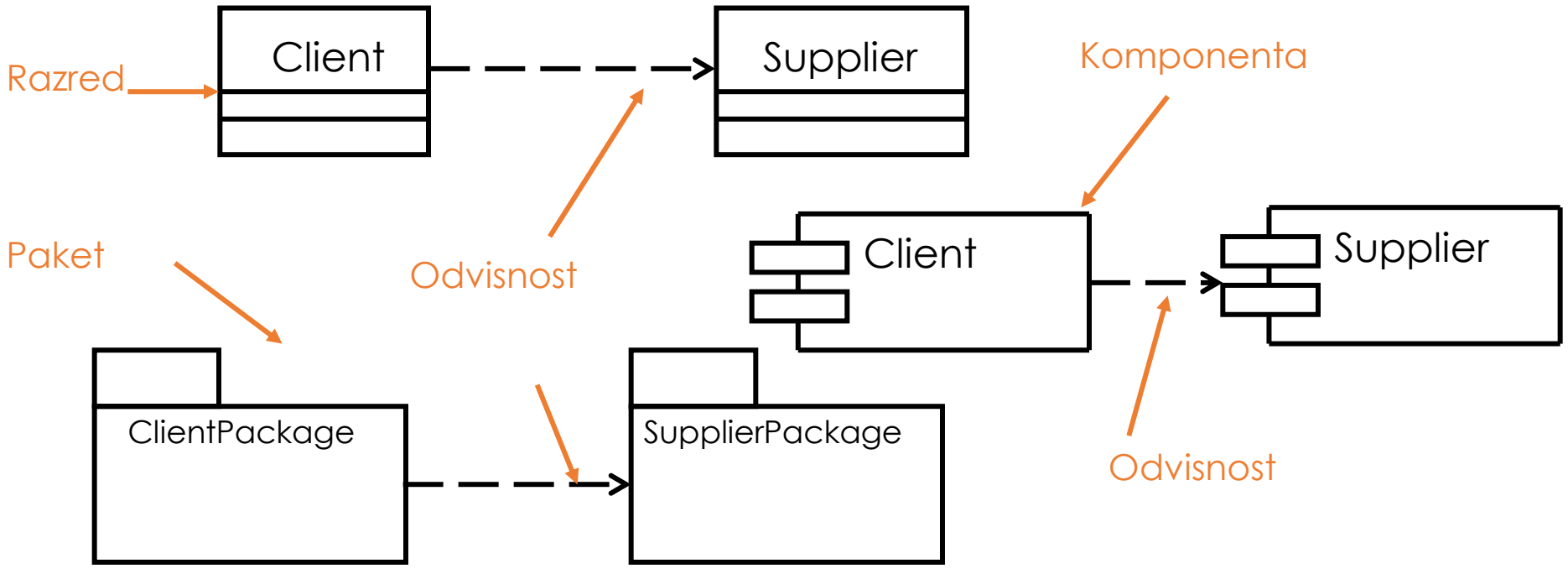
UML – asociacija kot razred

- Asociacije kot razred – dodatne informacije.
  - asociaciji se doda razred.



UML - odvisnost

- Povezava med dvema elementoma modela, kjer lahko sprememba v enem elementu povzroči spremembo v drugem.
- Povezava odvisnosti je šibkejša oblika povezave.
- Povezava tipa “uporablja”.



UML – generalizacija - dedovanje

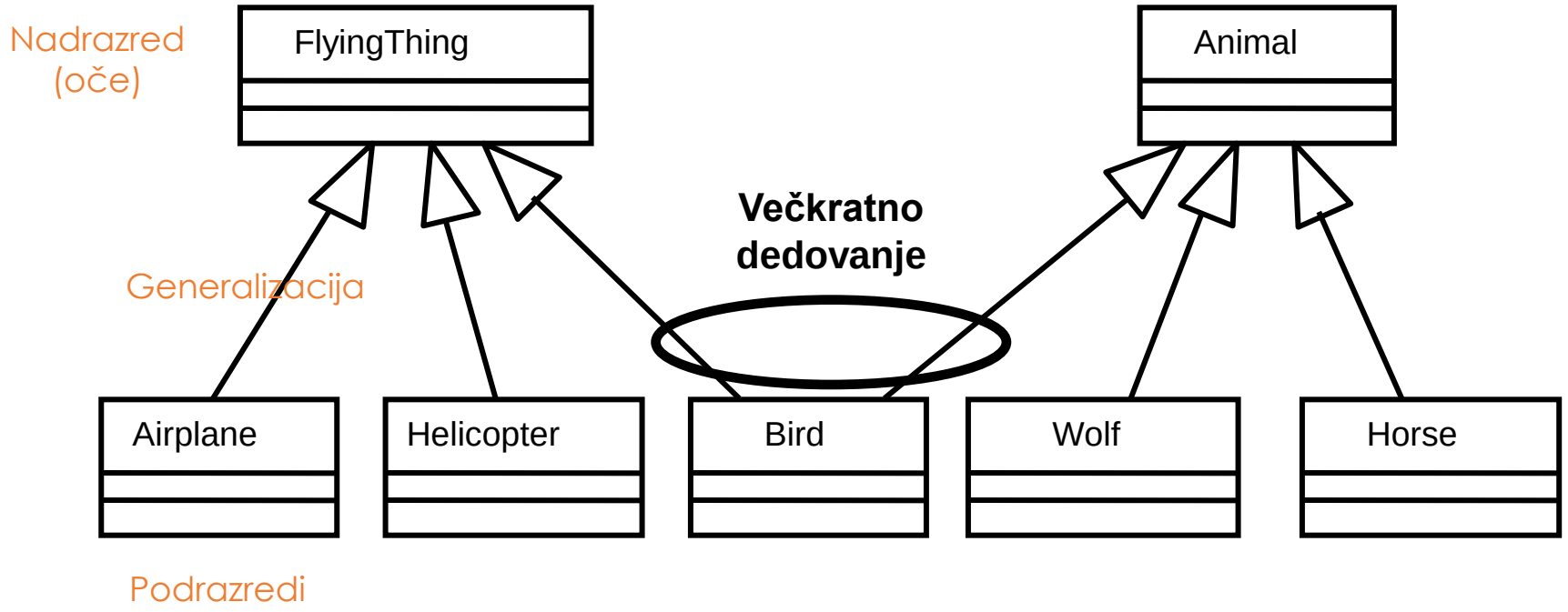
- Povezava med razredi, kjer si en razred deli strukturo in/ali obnašanje enega ali več razredov
- Definira hierarhijo abstrakcije, kjer podrazred deduje od enega ali več nadrazredov:
  - enojno dedovanje,
  - večkratno dedovanje.
- Podrazred podeduje atribute, operacije in povezave nadrazreda

- Podrazred lahko:
  - definira dodatne atribute, operacije in povezave,
  - predefinira podedovane operacije.

- Skupni atributi, operacije in/ali povezave so prikazane na najvišjem primernem nivoju v hierarhiji

UML – generalizacija - dedovanje

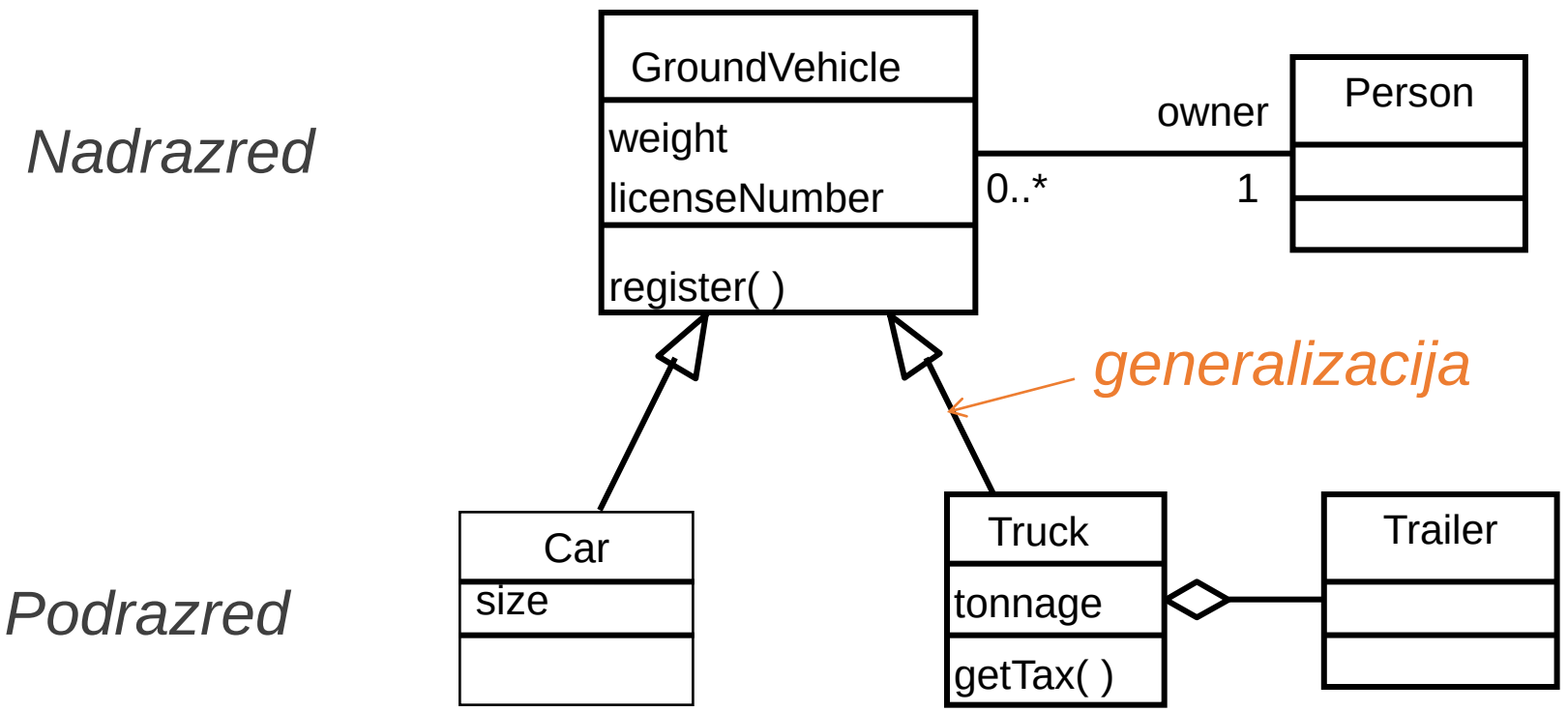
- En razred deduje od drugega.
- Razred lahko deduje iz več razredov.



Večkratno dedovanje uporabljamo pazljivo in le ko je to smiselno in potrebno!

UML – generalizacija - dedovanje

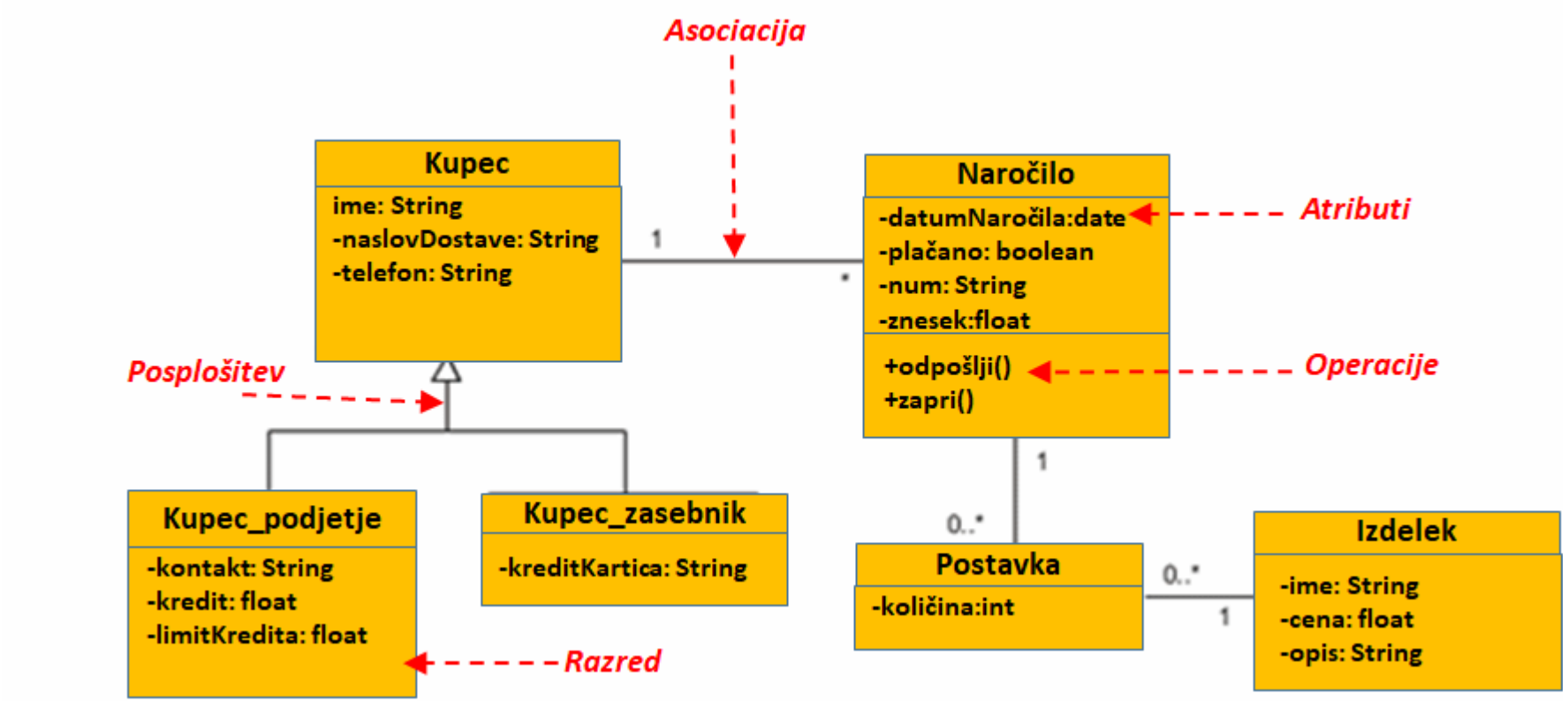
Kaj se podeduje?



UML – generalizacija - dedovanje

- V zgornjem primeru je tovornjak definiran kot vozilo z nosilnostjo in avto kot vozilo z definirano velikostjo.
- Atributi:
  - cestno vozilo ima dva atributa: težo in številko prometnega dovoljenja.
  - avto ima tri atribute: težo, številko prometnega dovoljenja in velikost.
  - tovornjak ima tri atribute: težo, številko prometnega dovoljenja in nosilnost.
- Operacije:
  - cestno vozilo ima eno operacijo : register().
  - avto ima eno operacijo: register().
  - tovornjak ima dve operaciji: register() in getTax().
- Povezave:
  - avto je povezan z osebo (lastnikom).
  - tovornjak je povezan z osebo (lastnikom).
  - tovornjak ima še prikolico.

Razredni diagram - primer



Naloga

- razred **`Oseba`**:
  - zasebni lastnosti **ime in priimek** (niz znakov); zasebna lastnost **starost** (celo število); javna lastnost **elektronski naslov** (niz znakov, lahko 2x); zaščitena lastnost **telefonska številka** (niz znakov)
  - javna metoda **PrikaziPodatke**, ki vrne niz znakov; **get** in **set** metode za zgoraj omenjene lastnosti

- razred **`Vozilo`**:
  - zasebna lasnost **model** (niz znakov); zasebna lastnost **leto izdelave** (celo število); zasebna lasnost **prostornina** (realno število)
  - javna metoda **prikaziPodrobnosti**, ki vrne niz znakov, get in set metode za zgoraj omenjene lastnosti
  - razred ima privzeti **konstruktor**, kjer je vrednost modela = "N/A", leto izdelave 1970, prostornina pa 1,0.

Naloga

- razred **Žival**:
  - zasebna lastnost ime (niz znakov); zasebna lastnost starost (celo število)
  - javni metodi Je in Spi (ne vračata nič)

- razred **Kuža**:
  - je izpeljan iz razreda Žival; ima javno metodo laja (ne vrača nič)

- razred **Muca**:
  - je izpeljan iz razreda Žival; ima javno metodo mijavka (ne vrača nič)

Razredom dodaj še get in set metode za lastnosti ter privzete in parametrične konstruktorje.

Naloga

Razred **Članek** je definiran z *naslovom* in *revijo*. Avtorji knjig so definirani z razredom **Avtor**. Razred **Avtor** ima lastnosti *priimek in ime* ter *datum rojstva* (oboje nizi znakov). Eno knjigo lahko napiše samo en avtor, en avtor lahko napiše več knjig. Razred Članek ima metodo *DodajAvtorja*, razred Avtor pa *VrniPodatkeOAvtorju*. Dodaj še get in set metode za lastnosti ter konstruktorje.

Ustvarite razred **Skodelica**. Razred naj ima dve javni realni lastnosti **premer** in **višina** (v cm), obe morata biti večji od 0. Tretja javna lasnost naj bo **barva**. Napišite konstruktor brez parametrov, ki nastavi premer in višino na 10 in barvo na belo ter še en konstruktor, ki nastavi premer, višino in barvo na vrednosti, podanimi preko parametrov.

Razred naj ima še lastnost **prostornina**, ki naj bo samo za branje. Formula za izračun je: (premer/2)2*π*višina).

Za razred **Skodelica** napišite še metodo, ki izpiše podatke o skodelici v naslednji obliki: "skodelica lahko prejme nnn slastne kave in je bbb barve".

Naloga

Razred **Pravokotnik** je definiran z spodnjim levim vogalom v koordinatnem sistemu ter širino in višino. Razred vsebuje še metodo ***povrsina()***, ki vrne površino pravokotnika. Objekt iz razreda **Pravokotnik** lahko naredimo tako, da mu pri kreiranju pošljemo vse štiri parametre ali s privzetim konstruktorjem. Vsi podatki so zasebni.

Kvadrat je posebne vrste pravokotnik, ki ga lahko ustvarimo samo s podatki o vozlišču ter širini. **PolniPravokotnik** je posebne vrste pravokotnik, ki poleg vseh podatkov vsebuje še barvo. Barva pravokotnika je javni podatek.

Narišite razredni diagram, ki ustreza zgornjemu opisu.

Naloga

Z razredom **`BancniRacun`** želimo predstaviti bančni račun neke osebe. Ta razred vsebuje ime lastnika računa in trenutno stanje (v evrih). Razred vsebuje tudi metodo ***`nakazi(znesek)`***, ki na račun doda podan znesek evrov, in metodo ***`dvigni(znesek)`***, ki z računa odšteje dani znesek v evrih.

Varčevalni račun je posebne vrste bančni račun, ki ima zraven vseh omenjenih lastnosti in metod še podatek o letni obrestni meri (v odstotkih, npr. 3,5 %) ter metodo ***`pristejMesecneObresti()`***, ki na račun na koncu meseca doda ustrezen znesek denarja.

Narišite razredni diagram, ki ustreza zgornjemu opisu.

2024/2025

Dodatne prosojnice