SOFTVERSKO INŽENJERSTVO

školska 2024/2025 godina

Vežba 3: UML klasni dijagrami

🌟 **Cilj vežbe**

- Razumevanje osnovnih elemenata UML dijagrama klasa
- Upoznavanje sa konceptima: klasa, atribut, metoda, odnosi između klasa
- Razlikovanje odnosa: nasleđivanje, asocijacija, agregacija i kompozicija
- Vežbanjem kroz praktične primere primeniti teorijsko znanje

Šta je UML?

UML (Unified Modeling Language) je standardizovani jezik za vizuelno modelovanje softverskih sistema. Omogućava prikaz klasa, objekata, njihovih osobina i odnosa između njih. Koristi se u fazi analize i dizajna softverskog sistema.

Najčešće korišćeni dijagram u objektno-orijentisanom programiranju je **dijagram klasa**. On omogućava da se jasno prikažu struktura sistema, odgovornosti pojedinačnih klasa i međusobni odnosi, što znatno olakšava komunikaciju među članovima razvojnog tima i olakšava održavanje koda.

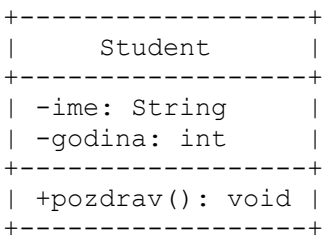
UML podržava više tipova dijagrama (npr. dijagram aktivnosti, sekvenci, slučajeva upotrebe), ali je **dijagram klasa** ključan za strukturisani prikaz statičkih aspekata sistema.

1. Dijagram klasa

Dijagram klasa prikazuje:

- **Klase**: entitete sistema, definisane kroz svoje atribute i metode
- **Atribute**: osobine klase (obeleženi sa - za privatne i + za javne)
- **Metode**: ponašanje klase (funkcije)
- **Veze**: odnose između klasa (nasleđivanje, asocijacija, agregacija, kompozicija)

Primer zapisa:



2. Odnosi između klasa

a) Nasleđivanje (Inheritance)

Predstavlja "jeste" odnos (is-a), gde jedna klasa preuzima osobine i ponašanja druge klase.

- Obeležava se strelicom sa belim trouglom: `--|>`
- Omogućava ponovno korišćenje koda i organizovanje hijerarhije klasa.
- Primer: `Student --|> Osoba`

b) Asocijacija (Association)

Predstavlja vezu između objekata dve klase. Znači da jedna klasa koristi poznaje drugu.

- Može biti jednostrana ili obostrana.
- Obeležava se punom linijom.
- Primer: **`Profesor --------> Fakultet`**
- Može imati oznake za multiplicitet (1, 0..*, 1*..) da pokaže moguće instanci u vezi.

c) Agregacija (Aggregation)

Specijalna forma asocijacije koja predstavlja odnos "deo-celina", gde delovi mogu postojati nezavisno od celine.

- Obeležava se praznim rombom.
- Primer: **`Tim <>--------> Igrac`**
- Koristi se kada postoji slabija veza između celine i delova.

d) Kompozicija (Composition)

Jača forma agregacije, gde deo ne može postojati nezavisno od celine.

- Obeležava se crnim rombom.
- Primer: **`Kuca <#--------> Soba`**
- Ako se objekat celine uništi, automatski se uništavaju i njeni delovi.

📝 **Praktični primer**

Zamislimo sistem za upravljanje školom.

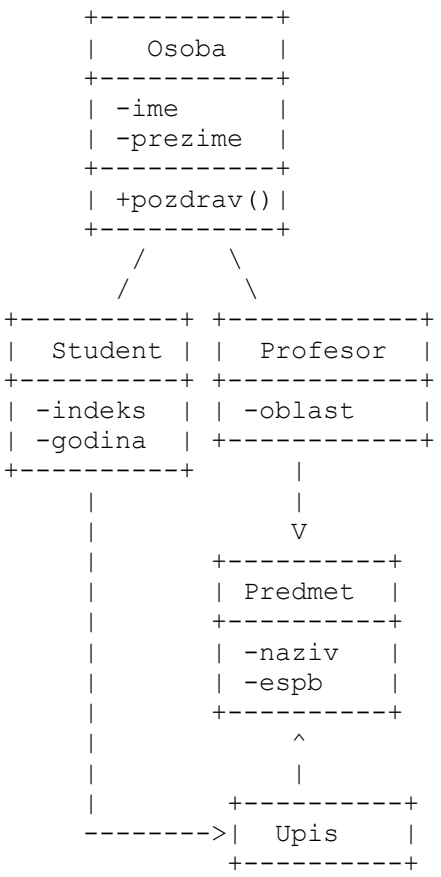
Imamo sledeće klase:

- Osoba (nadklasa): ime, prezime, predstaviSe()
- Student (nasleđuje Osobu): brojIndeksa, godina
- Profesor (nasleđuje Osobu): oblast
- Predmet: naziv, espb
- Upis: veza između Student i Predmet

Odnosi:

- Student --|> Osoba
- Profesor --|> Osoba
- Profesor --------> Predmet (asocijacija: jedan prof. može predavati više predmeta)
- Student <>--------> Upis --------<> Predmet (agregacija: studenti mogu upisivati više predmeta, a upis ne postoji bez njih)

UML dijagram (tekstualni prikaz):



📚 Zadatak za samostalni rad

Nacrtaj UML dijagram za sledeći sistem:

Sistem za biblioteku

- Klasa Knjiga: naslov, autor, isbn
- Klasa Clan: ime, clanskiBroj
- Klasa Zaposleni: ime, pozicija
- Klasa Pozajmica: predstavlja pozajmljivanje knjige

Odnosi:

- Clan pozajmljuje Knjigu preko Pozajmica (kompozicija - pozajmica ne postoji bez knjige i člana)
- Zaposleni izdaje knjigu (asocijacija sa Pozajmica)

Zadatak:

1. Prikazati sve klase sa atributima i metodama po potrebi
2. Označiti odnose: asocijacija, agregacija, kompozicija
3. Nacrtati dijagram ručno na papiru ili preko draw.io

Primeri dijagrama na zadatu temu su navedeni u nastavku: