Projektovanje softvera

Posetilac

Posetilac (1)

- Ime i klasifikacija:
  - Posetilac (engl. *Visitor*) – objektni uzorak ponašanja
- Namena:
  - postoji skup raznorodnih operacija koje treba primenjivati na raznorodne elemente jedne objektne strukture
  - uzorak omogućava definisanje nove operacije bez izmene klasa elemenata nad kojima se izvršava

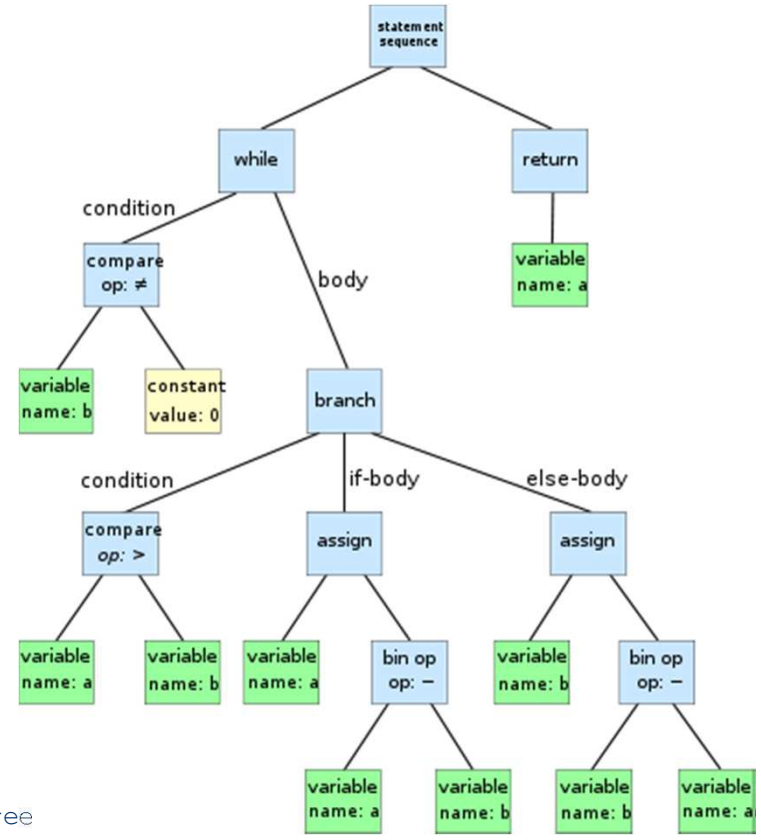
Posetilac (2)

- Motivacija:
  - razmatra se prevodilac koji reprezentuje strukturu programa kao stablo apstraktne sintakse (SAS)
  - potrebno je obezbediti operacije nad SAS za:
    - statičku proveru tipova
    - generisanje koda
    - …
  - većina operacija će na različit način tretirati čvorove SAS za:
    - naredbu dodele vrednosti
    - obraćanje promenljivoj
    - …
  - hijerarhija čvorova SAS zavisi od jezika, ali je za dati jezik stabilna

Posetilac (3)

- Primer SAS za program:

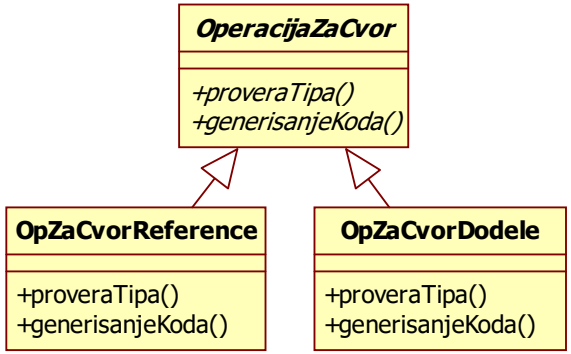
```
while b ≠ 0
  if a > b
    a := a − b
  else
    b := b − a
end_while
return a
```



https://en.wikipedia.org/wiki/Abstract_syntax_tree

Posetilac (4)

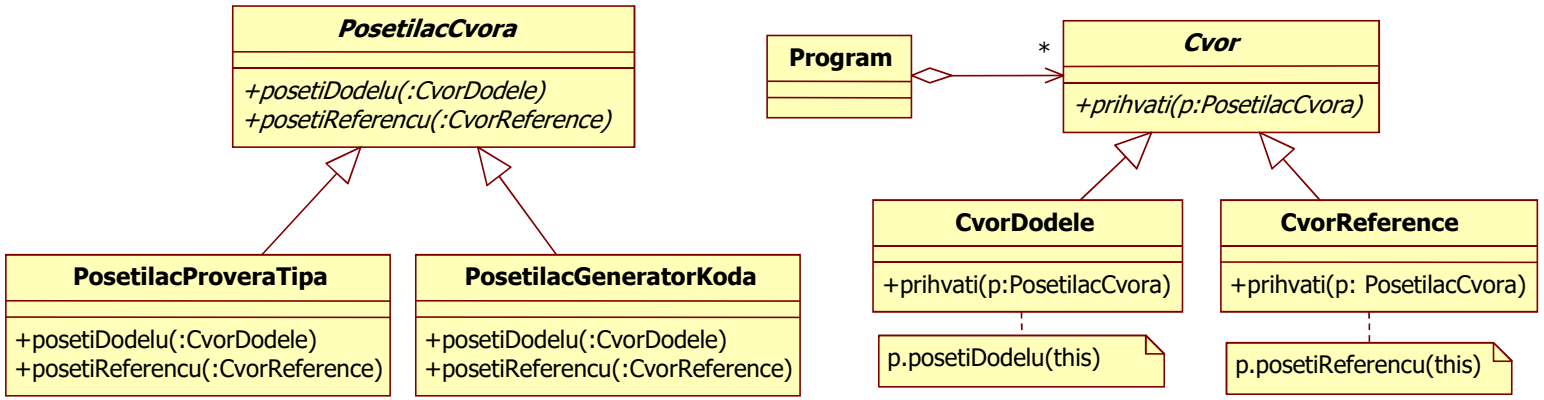
- Motivacija (nastavak):
  - nefleksibilan dizajn:



  - problem: prisustvo svih operacija u različitim klasama čvorova vodi do sistema koji je teško razumeti, održavati i menjati
    - mešanje koda za proveru tipova i koda za generisanje koda kvari razumljivost
    - dodavanje nove operacije zahteva izmenu svih klasa čvorova
  - cilj: razdvajanje hijerarhije čvorova od operacija koje se vrše nad njima
  - rešenje: pakovanje operacija za različite čvorove u objekte posetilaca koji se prosleđuju čvorovima SAS kada se struktura obilazi

Posetilac (5)

- Motivacija (nastavak):
  - posetilac kapsulira jednu operaciju za različite čvorove SAS
  - kada čvor primi posetioca – on poziva operaciju posetioca
    - operacija odgovara klasi čvora
    - čvor dostavlja referencu na sebe kao parametar operacije
  - posetilac tada izvršava operaciju za dotični čvor



Posetilac (6)

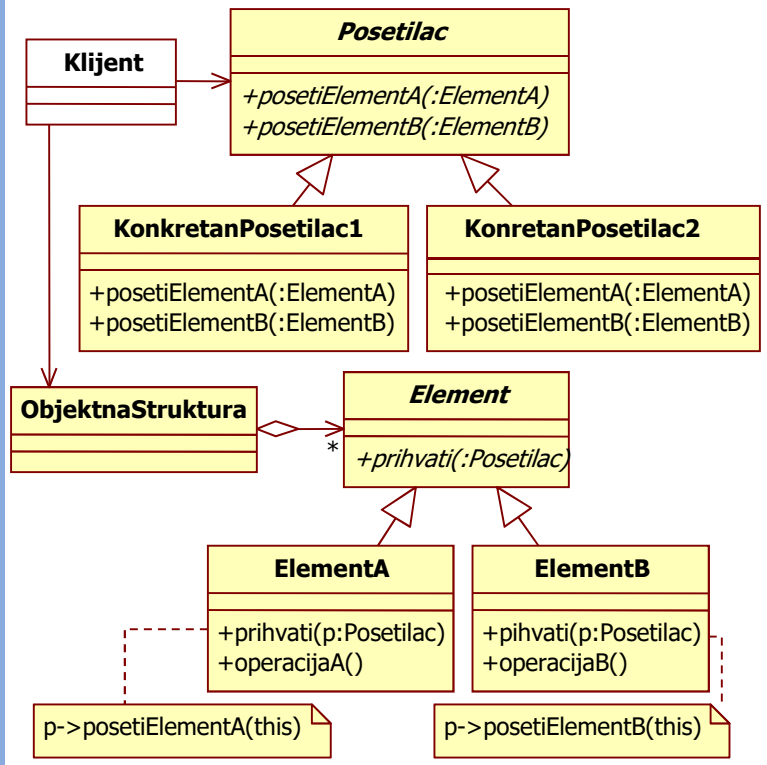
- Motivacija (nastavak):
  - u uzorku *Posetilac* se definišu dve hijerarhije klasa
    - za elemente nad kojima se vrše operacije (h. čvorova)
    - za posetioce koji definišu operacije nad elementima (h. posetilaca)
  - nova operacija se dodaje kao nova potklasa u h. posetilaca
    - ako bi aplikacija trebalo da računa metriku programa
      samo bi se definisala nova potklasa klase `PosetilacCvora`
      a ne bi se dodavao aplikativno zavisan kod u klase čvorova SAS

Posetilac (7)

- Primenljivost: uzorak treba koristiti kada
  - jedna objektna struktura sadrži objekte raznih klasa, a potrebno je nad tim objektima izvršiti raznorodne operacije koje zavise od konkretnih klasa objektne strukture
  - više raznorodnih operacija se izvršava nad objektima strukture, pri čemu se njihove klase ne "zagađuju" tim operacijama
    - posetilac omogućava grupisanje povezanih operacija (koje odgovaraju jednoj primeni) u jednoj klasi
  - klase koje definišu strukturu objekata se retko menjaju, a često se definišu nove operacije nad elementima strukture
    - promena klasa objektne strukture zahteva promenu svih posetilaca
    - može mnogo da košta, pa ako je ovo često, bolje je definisati operacije u tim klasama

Posetilac (8)

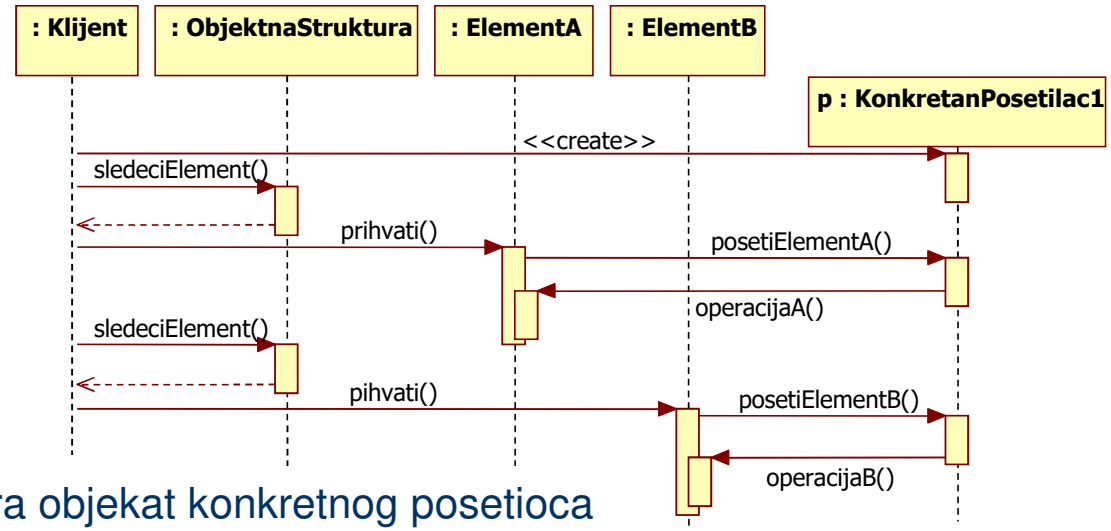
- Struktura:



- Učesnici:
  - `Posetilac` (klasa `PosetilacCvora`)
    - po jedna operacija `posetiX()` za svaku potklasu `Element`
    - potpis operacije identifikuje odgovarajući potklasu `Element`
  - `KonkretanPosetilacX` (klase `PosetilacProveraTipa`,...)
    - implementira sve operacije koje deklariše `Posetilac`
    - predstavlja kontekst algoritma i čuva njegovo stanje koje često akumulira rezultate obilaska strukture objekata
    - svaka operacija implementira deo algoritma za odgovarajuću klasu objekata u strukturi
  - `Element` (klasa `Cvor`)
    - deklariše operaciju `prihvati()` - `Posetilac` je argument
  - `ElementX` (klase `CvorDodele`, `CvorReference`)
    - implementira operaciju `prihvati()` tako što poziva `posetilac.posetiElementX(this)`
  - `ObjektnaStruktura` (klasa `Program`)
    - pruža interfejs visokog nivoa koji omogućava klijentu da obiđe elemente dostavljajući im posetioca
    - može da bude *Sastav* (stablo) ili neka zbirka (npr. lista)

Posetilac (9)

- Saradnja:



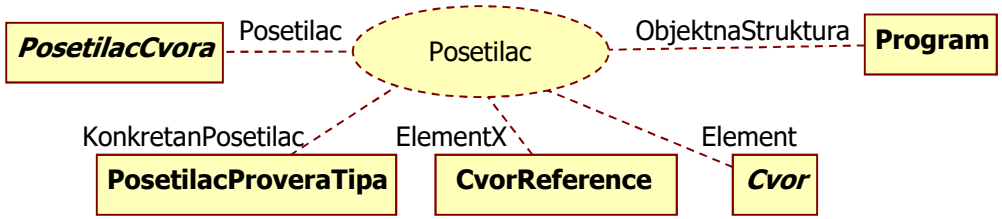
  - Klijent:
    - stvara objekat konkretnog posetioca
    - obilazi objektnu strukturu posećujući svaki njen element
    - svakom elementu prosledi posetioca
  - Element
    - poziva operaciju posetioca kojom mu traži da ga poseti

Posetilac (10)

- Posledice:
  - grupiše srodne i razdvaja različite operacije
  - olakšava dodavanje novih operacija (klasa posetilaca)
  - dodavanje novih klasa konkretnih elemenata nije lako
  - mogućnost akumuliranja stanja za vreme posećivanja elemenata
    - bez posetioca, stanje bi se predavalo operacijama kao dodatni argument
  - moguć nedostatak: probijanje kapsulacije elemenata
    - često su potrebne javne operacije za pristup unutrašnjem stanju elemenata strukture
    - proglašavanje posetioca prijateljem u elementu nije dobro rešenje
      - dodavanje nove vrste operacija (posetioca) bi zahtevalo izmenu svih elemenata

Posetilac (11)

- UML notacija:



- Povezani uzorci:
  - *Sastav* – posetilac može da se koristi da bi se neka operacija primenila na strukturu objekata definisanu kao *Sastav* (*Kompozicija*)
  - *Iterator* se često koristi zajedno sa *Posetiocem*,
    - služi za sistematičan obilazak objekata strukture kojima se šalju posetioci
  - *Interpreter* – posetilac se može koristiti da obavi interpretaciju