Projektovanje softvera

Komanda

Komanda (1)

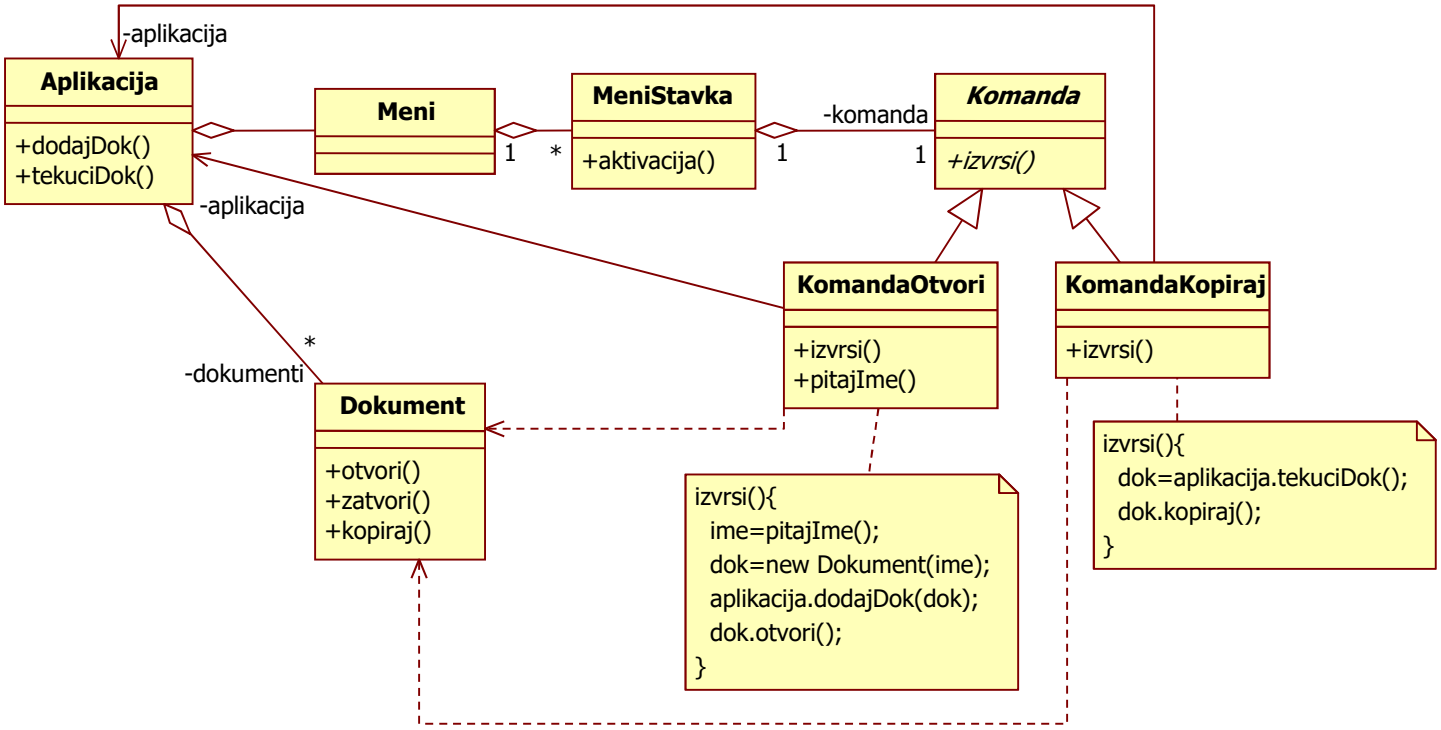
- Ime i klasifikacija:
  - Komanda (engl. *Command*) – objektni uzorak ponašanja
- Namena:
  - kapsulira zahtev u jedan objekat, omogućavajući:
    - da se pokretači izvršenja parametrizuju različitim zahtevima,
    - da se zahtevi isporučuju pokretačima kroz red čekanja,
    - da se pravi dnevnik *(log)* zahteva i
    - da se efekti izvršenog zahteva ponište (*undo*)
- Drugo ime:
  - Akcija, Transakcija (engl. *Action*, *Transaction*)

Komanda (2)

- Motivacija:
  - nekad je potrebno izdati zahtev da se izvrši neka operacija bez znanja o ciljnoj akciji i njenom izvršiocu (primaocu zahteva)
  - npr. GUI biblioteke sadrže objekte kao što su dugmad ili meniji
    - ovi objekti treba da izvrše zahteve koji su posledica interakcije sa korisnikom
    - biblioteka klasa ne može da pretpostavi adekvatne akcije koje se zahtevaju
    - samo ciljna aplikacija zna šta je specifična akcija za neko dugme
    - projektant biblioteke ne zna ništa o konkretnoj akciji ni o primaocu zahteva
    - on može samo da pretpostavi da je potrebno izvršiti neku operaciju kada korisnik pritisne dugme, odnosno odabere stavku menija
  - uzorak *Komanda*:
    - dopušta objektima biblioteke da zahtevaju izvršenje operacije sa posledicom da nepoznata akcija bude izvršenea od nepoznatog objekta aplikacije
    - to postiže smeštajući zahteve za operacijama u posebne objekte
    - pokretač izvršenja se parametrizuje takvim objektom zahteva
  - objekat sa zahtevom može da se zapamti ili prosledi drugom objektu

Komanda (3)

- Motivacija (nastavak):



Komanda (4)

- Motivacija (nastavak):
  - ključna apstrakcija uzorka je klasa *Komanda*
    - deklariše ugovor za izvršenje operacija
    - u najjednostavnijoj formi, ugovor se sastoji od apstraktne operacije *izvrsi()*
  - potklase klase *Komanda* specificiraju par (primalac komande, akcija)
    - primalac zna kako da izvrši akcije koje su potrebne za ispunjenje zahteva
  - meniji lako mogu da se implementiraju koristeći objekte potklasa klase *Komanda*
  - uzorak raspreže objekat pozivaoca operacije od onog ko zna kako da je izvrši
    - objekat koji izdaje komandu treba da zna samo kako se ona izdaje
    - on ne treba da zna ništa o tome kako se izvršava i ko je izvršava
  - komande mogu da se menjaju dinamički
    - u konkretnom primeru menija, ovo omogućava kontekstno zavisne menije
  - jednostavna je implementacija skriptova (složenih komandi - sekvenci operacija)

Komanda (5)

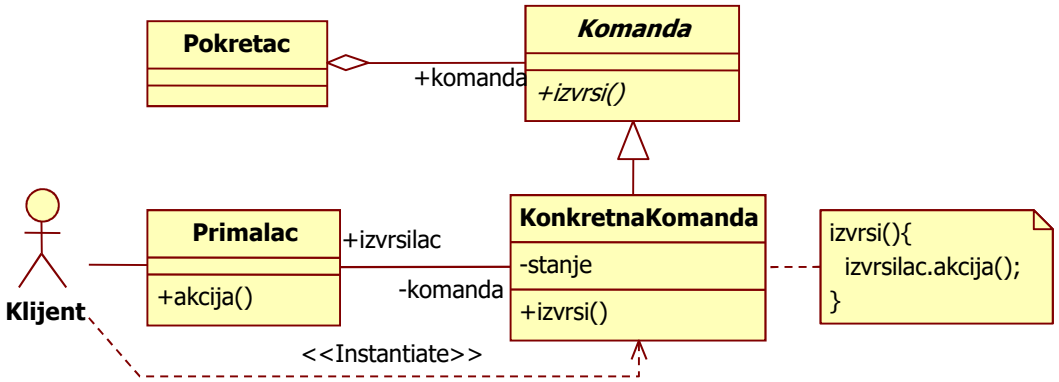
- Primenljivost: uzorak treba koristiti kada je potrebno
  - da se objekti parametrizuju akcijom koju treba da obave
    - zamena za funkciju povratnog poziva (*callback*) u tradicionalnim jezicima
  - specificirati i stavljati u red čekanja zahteve, a kasnije ih izvršavati
    - objekat komande može da ima različit životni vek od onog ko izdaje zahtev
    - objekat komande može da se prepusti drugom procesu (promena adresnog prostora), ako primalac može da se adresira univerzalno
  - podržati mehanizam *undo/redo*
    - `izvrsi()` operacija može da sačuva prethodno stanje
    - ugovor treba da sadrži i operaciju `ponisti()` koja restaurira stanje (*undo*)
    - može da sadrži i operaciju `vrati()` koja ponavlja akciju (*redo*)
    - neograničen nivo *undo* i *redo* se postiže smeštanjem objekata izvršenih komandi u listu, odnosno prolaskom kroz listu unazad (*undo*) i unapred (*redo*)

Komanda (6)

- Primenljivost: uzorak treba koristiti kada je potrebno
  - podržati oporavak u slučaju kraha (*recovery*)
    - sa određenom periodikom se snima stanje objekata u trajnoj memoriji
    - snima se dnevnik izvršenih komandi posle snimanja stanja, da bi te komande mogle ponovo da se izvrše
    - kada dođe do kraha, oporavak se postiže
      - učitavanjem i restauracijom simljenog stanja iz trajne memorije
      - ponovnim izvršavanjem komandi iz dnevnika
    - u ugovor klase `Komanda` se dodaju operacije za perzistenciju dnevnika promena
  - podržati transakcije
    - transakcije su složene operacije sastavljene od primitivnih
    - transakcije kapsuliraju skup promena podataka

Komanda (7)

- Struktura:

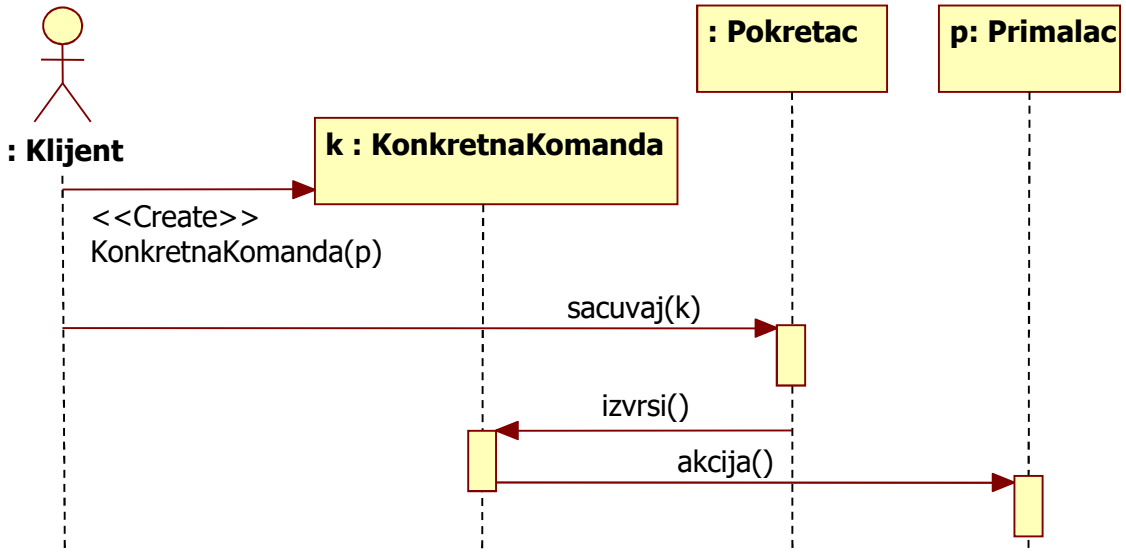


Komanda (8)

- Učesnici:
  - *Komanda* (klasa *Komanda*)
    - deklariše ugovor za izvršenje neke operacije
  - KonkretnaKomanda (klase KomandaOtvori, KomandaKopiraj)
    - definiše vezu između jednog objekta Primalac i konkretne akcije
  - Kijent (klasa Aplikacija)
    - kreira objekat KonkretnaKomanda
    - može da postavi njen objekat Primalac
      ili da omogući objektu KonkretnaKomadna da dohvati objekat Primalac
  - Pokretac (klasa MeniStavka)
    - traži od Komande da izvrši zahtevanu operaciju
  - Primalac (klasa Dokument)
    - izvršava konkretnu akciju kojom se ispunjava zahtev

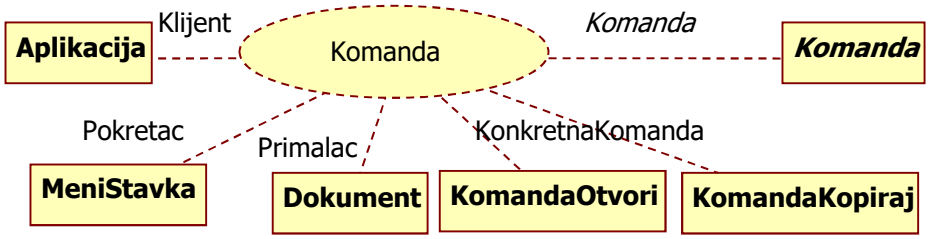
Komanda (9)

- Saradnja:



Komanda (10)

- UML notacija:



- Posledice:
  - raspreže objekat koji pokreće operaciju od onog koji zna kako da je izvrši
  - komande su objekti kao i svi drugi i njima može da se manipuliše
  - komande mogu da se grupišu u složene komande (makrokomande)
  - jednostavno je dodavanje novih komandi, ne treba menjati postojeće klase
- Povezani uzorci:
  - *Sastav* se koristi za stvaranje makrokomandi (skriptova)
  - *Podsetnik* može da čuva stanje pre izvršenja komande, potrebno za *Undo*
  - komanda čija se kopija stavlja u dnevnik se ponaša kao *Prototip*