Projektovanje softvera

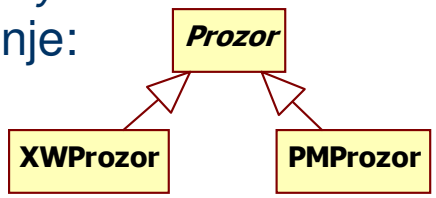
Most

Most (1)

- Ime i klasifikacija:
  - *Most* (eng. *Bridge*) – objektni uzorak strukture
- Namena:
  - razdvaja apstrakciju od njene implementacije da bi se mogle nezavisno menjati
- Drugo ime:
  - Ručka/Telo (Handle/Body)

Most (2)

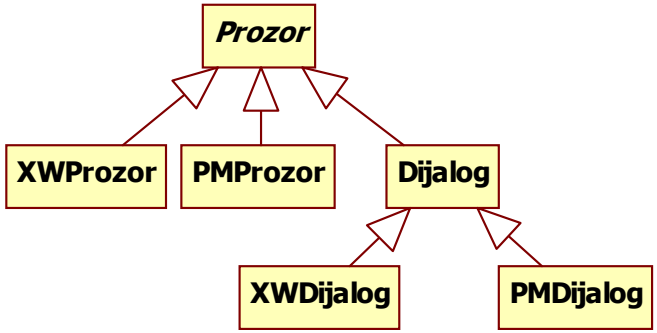
- Motivacija:
  - problem: postoji više implementacija za jednu apstrakciju
  - tradicionalno OO rešenje: apstraktna klasa i izvedene klase
  - primer: apstrakcija `Prozor` u alatima za GUI
    - potrebno je razvijati aplikacije za razne prozorske sisteme
      - npr. *X Window System* i *IBM Presentation Manager*
    - tradicionalno rešenje:



  - rešenje nije dovoljno fleksibilno
    - nasleđivanje čvrsto vezuje implementaciju za apstrakciju
    - teško se nezavisno menjaju, proširuju i ponovo koriste

Most (3)

- Motivacija (nastavak): dva problema
  - (1) uvodi se nova apstrakcija - prozor za dijalog `Dijalog`
    - da bi se apstrakcija implementirala na obe platforme dobija se:



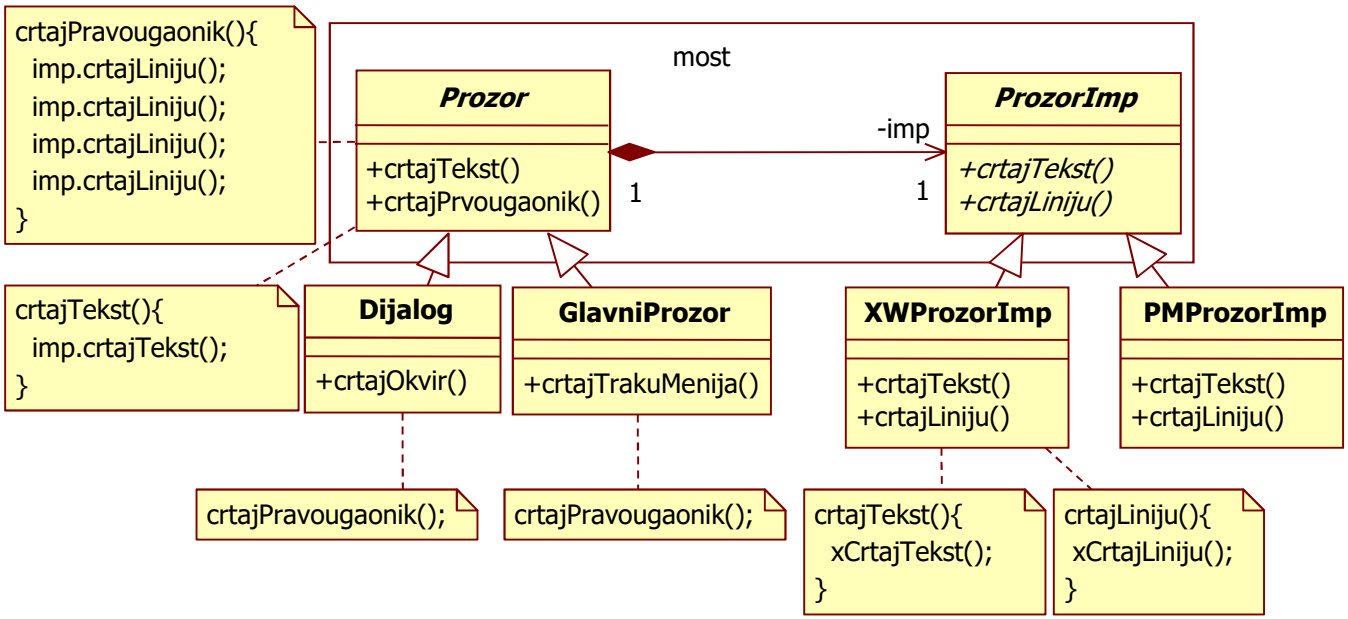
    - za svaku novu vrstu prozora – moraju se definisati po dve klase
    - podrška za novu platformu – po jedna klasa za svaku vrstu prozora

Most (4)

- Motivacija (nastavak):
  - (2) klijentski kod postaje zavisan od platforme
    - kad god klijent pravi prozor – pravi primerak konkretne klase sa specifičnom implementacijom za datu platformu
    - otežano prenošenje klijentskog koda na druge platforme
    - klijenti ne bi trebalo da se vezuju za konkretne implementacije
      - oni treba da vode računa samo o različitim apstrakcijama prozora
      - samo bi implementacija prozora smela da zavisi od platforme
  - rešenje problema: uzorak *Most*
    - apstrakcija prozora i njegova implementacija su dva odvojena korena odgovarajućih hijerarhija klasa
    - uspostavlja se most između apstrakcije i implementacije
    - apstrakcije i implementacije se mogu nezavisno menjati

Most (5)

- Motivacija (nastavak):



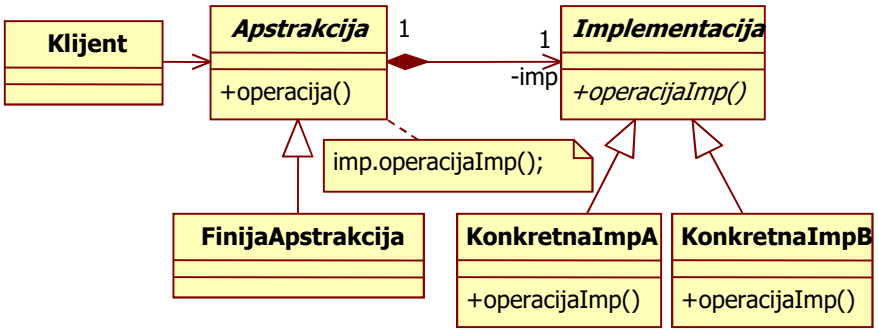
  - sve operacije klase `Prozor` se implementiraju primenom apstraktnih operacija klase `ProzorImp`

Most (6)

- Primenljivost: uzorak treba koristiti kada
  - treba izbeći trajno vezivanje apstrakcije i njene implementacije
    - npr., ako je potrebno implementaciju menjati u vreme izvršenja
  - i apstrakciji i implementaciji je potrebno proširivanje kroz potklase
    - most omogućava kombinovanje različitih apstrakcija i implementacija
  - promena u implementaciji apstrakcije ne sme da utiče na klijente
  - u jeziku C++: potpuno sakrivanje implementacije klase od klijenta
    - definicije klasa su u h fajlovima – delimično se otkriva implementacija
  - postoji opasnost od prevelikog broja klasa
    - npr. u primeru se vidi kvadratni rast (broj sistema x broj tipova prozora)
  - kada se želi da istu implementaciju deli više objekata
    a da to bude sakriveno od klijenta (eventualno uz brojanje referenci)

Most (7)

- Struktura:



- Saradnja:
  - `Apstrakcija` prosleđuje klijentske zahteve objektu `Implementacija`

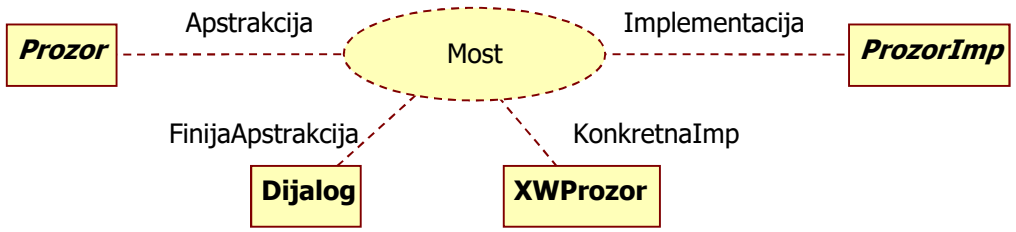
- Učesnici:
  - `Apstrakcija` (klasa `Prozor`)
    - definiše interfejs apstrakcije
    - održava referencu na objekat implementacije
  - `FinijaApstr.` (klasa `Dijalog`)
    - proširuje interfejs apstrakcije
  - `Implementacija` (klasa `ProzorImp`)
    - definiše interfejs klasa implement.
      - interfejs ne mora da liči na interfejs apstrakcije
  - `KonkretnaImpX` (klase `XWProzorImp`, `PMProzorImp`)
    - implementira interfejs implement.
    - definiše konkretnu implementaciju

Most (8)

- Posledice:
  - razdvajanje ugovora od implementacije
    - implementacija nije trajno vezana za apstrakciju, može da se konfiguriše dinamički
    - eliminisane su zavisnosti implementacije i apstrakcije u vreme prevođenja
      - izmena klasa apstrakcije ne zahteva prevođenje klasa implementacije
      - izmena konkretnih implementacija ne izaziva prevođenje klasa apstrakcije
  - bolje mogućnosti proširivanja
    - hijerarhije apstrakcija i implementacija se mogu nezavisno proširivati
  - skrivanje detalja implementacije od klijenta
    - klijent ne vidi ništa od implementacije, vidi samo apstrakciju
    - bitno kada se mora obezbediti kompatibilnost verzija biblioteke klasa
      - može se promeniti kompletna implementacija
      - samo ako se menja njen interfejs, menja se apstrakcija
      - klijent se ne menja do god se ne promeni interfejs apstrakcije

Most (9)

- UML notacija:



- Povezani uzorci:
  - *Apstraktna fabrika* može da kreira i konfiguriše *Most*
  - *Adapter* i *Most* prilagođavaju klijentu interfejs neke implementacije
    - *Adapter* se obično projektuje retroaktivno
    - *Most* se obično projektuje unapred