UML i tronivojska arhitektura

Bojan Tomić

Dokumentovanje - vrste

- Dokumentovanje projekta (prve tri faze procesa razvoja)
  - Korisnički zahtev
  - UML dijagrami
  - Šema baze podataka
- Dokumentovanje koda (implementacija i testiranje)
  - Komentari
  - javaDoc

Dokumentovanje - principi

- “Manje vredi više”
- Ne mora se dokumentovati sve
- Obavezno dokumentovati ključne ili složene detalje
- “Živa” dokumentacija
  - Elektronska (lakše se čita, distribuira, pretražuje menja i održava od papirne)
  - Poželjno automatizovano ažuriranje (direktna vezanost za sam program)

UML

- Unified Modeling Language (jedinstveni jezik za modelovanje)
- Skup grafičkih notacija za opisivanje i projektovanje softverskih sistema
- Tvorci - OMG (Object Management Group)
- Literatura
  - Martin Fowler, “UML Distilled”, Addison-Wesley (“UML ukratko” - prevod Mikro knjiga)

UML - motiv

- Jedinstveni jezik za sporazumevanje između različitih učesnika u razvoju softverskog sistema
- Jedinstveni jezik za specifikaciju softverskog sistema
- Automatsko generisanje koda na osnovu dijagrama
- Poluautomatsko generisanje dijagrama na osnovu koda
- Generisanje projektne dokumentacije na osnovu dijagrama

Neki popularni UML alati

- ArgoUML
- NetBeans 6.7.1 UML plugin
- MagicDraw (demo)
- Umbrello (Linux)
- Papyrus Eclipse plugin
- JDeveloper UML plugin
- ObjectAid UML Explorer
- UMLet
- yWorks UML Doclet (yDoc)

UML

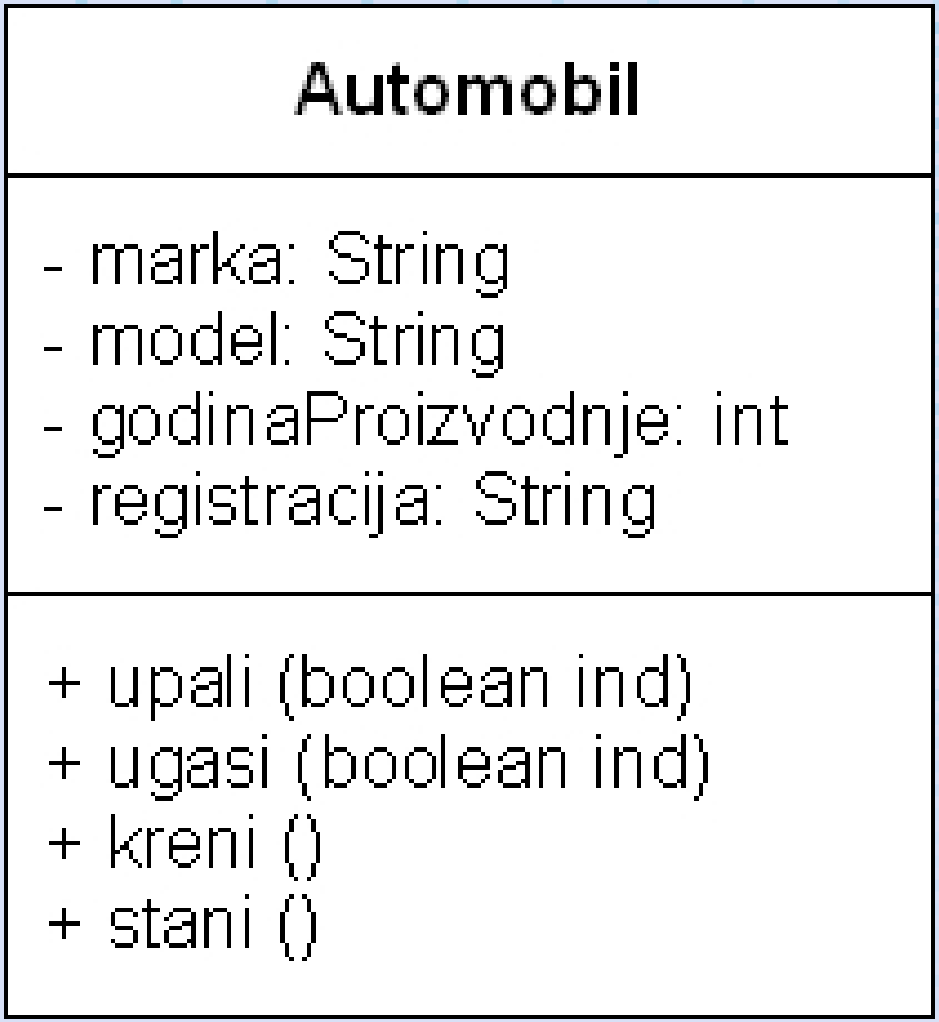
- Dve osnovne grupe dijagrama
- Dijagrami strukture
  - Služe za predstavljanje statičkih karakteristika softverskog sistema - njegove strukture
  - Arhitektura, klase, paketi i njigohi elementi...
- Dijagrami ponašanja
  - Služe za predstavljanje dinamičkih karakteristika softverskog sistema
  - Ponašanje metoda, pozivi od jedne klase ka drugoj, stanja u toku izvršavanja

UML

- Vrste dijagrama
  - **Dijagrami klasa**
  - **Dijagrami slučajeva korišćenja**
  - **Dijagrami sekvenci**
  - **Dijagrami komunikacije (kolaboracioni)**
  - **Dijagrami paketa**
  - **Dijagrami aktivnosti**
  - **Dijagrami komponenata**
  - **Dijagrami raspoređivanja**
  - **Dijagrami mašine stanja**
  - **Vremenski dijagrami**
  - **Dijagrami složene strukture**
  - **Dijagrami objekata**

UML dijagram klasa

- Koristi se za predstavljanje klasa i njihovih atributa, metoda i relacija
- Klasa je predstavljena pravougaonikom koji se sastoji iz tri dela
  - Naziv klase
  - Lista atributa
  - Lista metoda



UML dijagram klasa

- Znak ispred nekog elementa označava nivo pristupa

  + public **- private**

  # protected **~ default (paketski)**

- Atributi se pišu u formatu

  pristup nazivAtributa : tip

- Metode se pišu u formatu

  pristup nazivMetode (ulazni_parametri)

- Na dijagramu klasa se **<u>obično ne navode get i set metode</u>**

UML dijagram klasa

- Relacije - odnosi između klasa
- Vrste relacija
  - Asocijacija
    - Obična asocijacija
    - HAS-A asocijacija (kompozicija, agregacija)
    - IS-A asocijacija (nasleđivanje, implementacija)
  - Relacija korišćenja (USING)

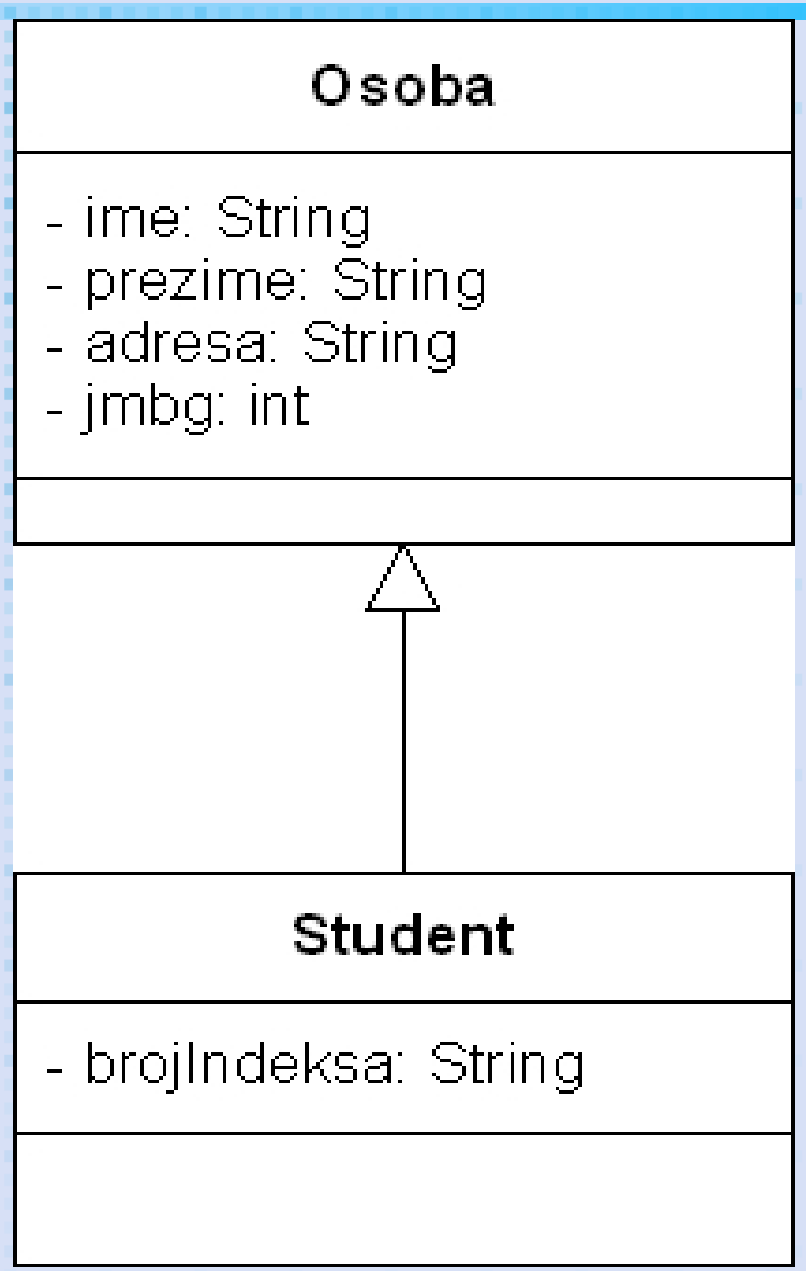
UML dijagram klasa

- Svaka relacija može imati **naziv**
- Svaka relacija se može posmatrati u dva **smera** a svaki smer ima svoju kardinalnost
- **Kardinalnost** relacije pokazuje broj objekata jedne klase koji umogu biti u relaciji sa jednim objektom druge klase

1	**Tačno jedan**
0..1	**Nula ili jedan**
0..* (*)	**Nula, jedan ili više**

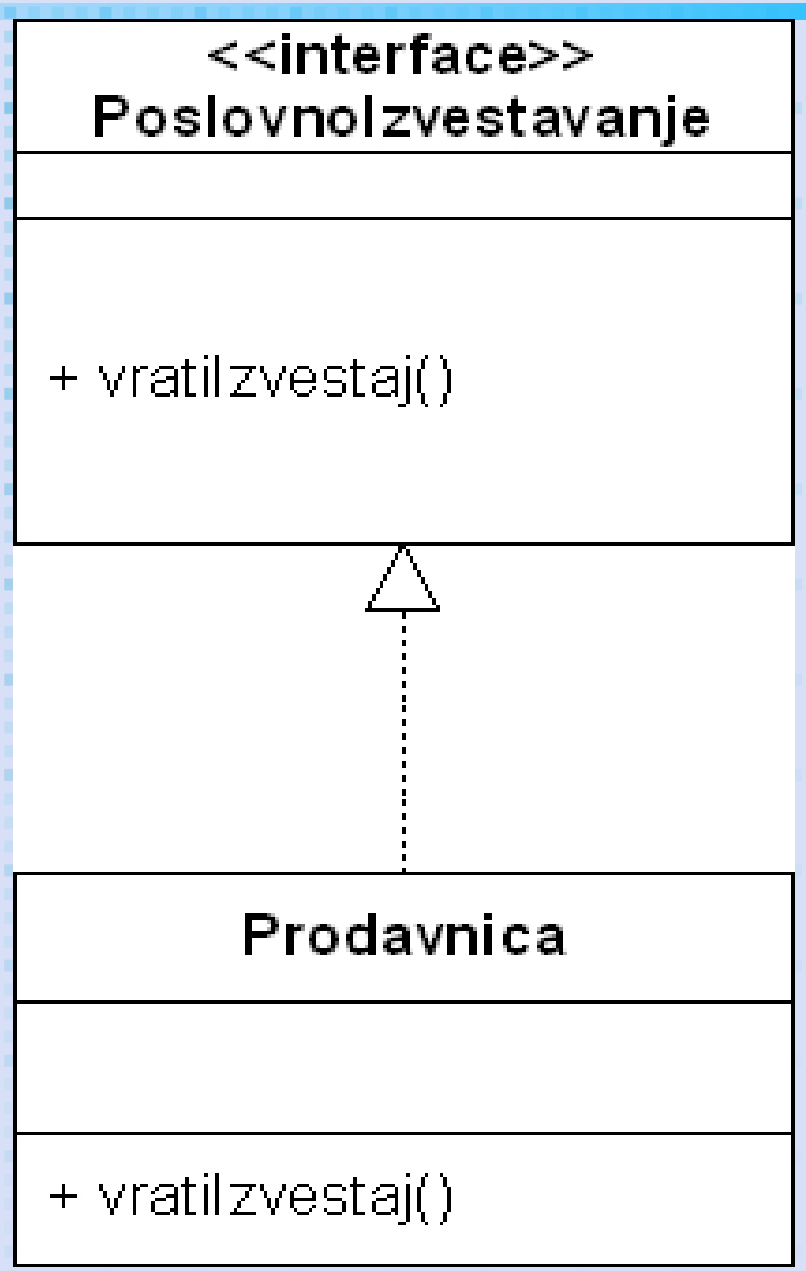
UML dijagram klasa

- Relacija “IS-A” - nasleđivanje
- Predstavlja se strelicom koja ima prazan trougao na vrhu
- Vrh strelice je uperen ka nadklasi



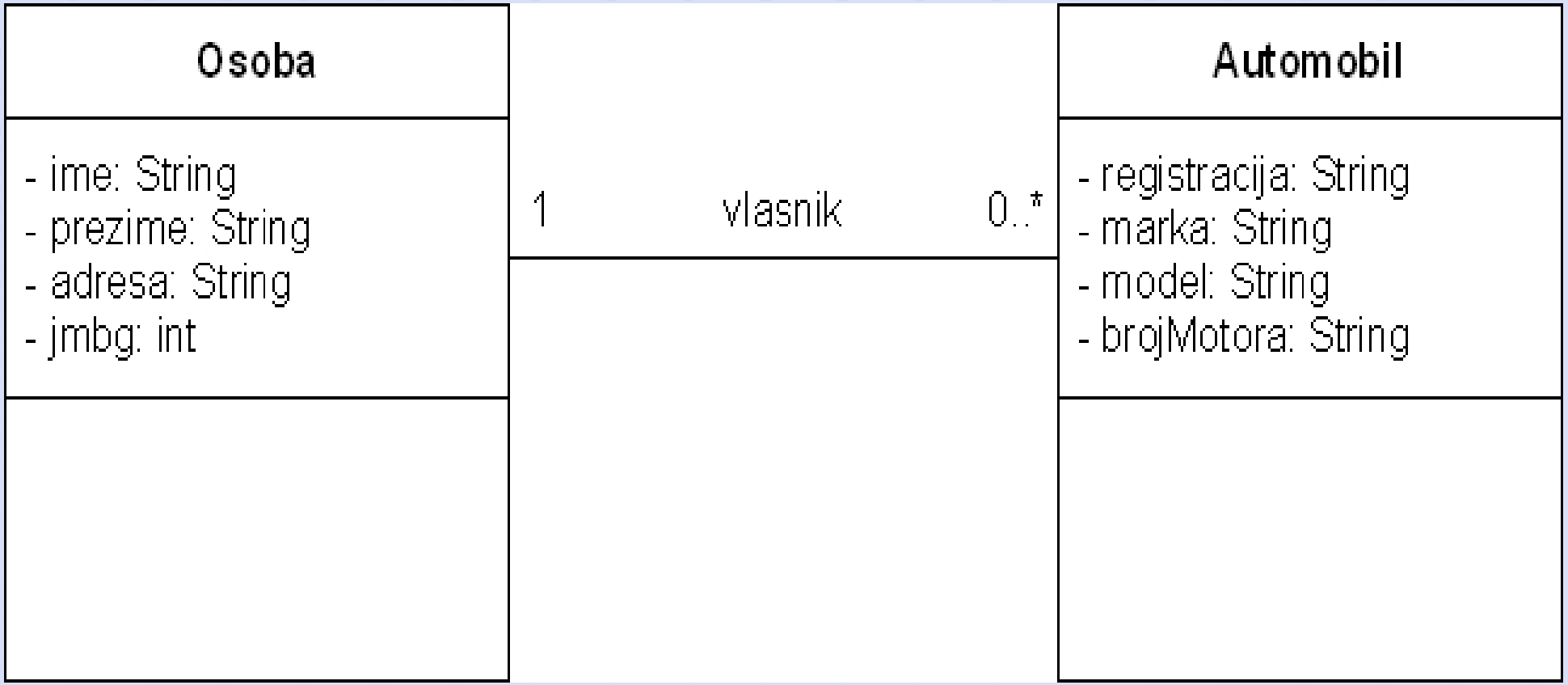
UML dijagram klasa

- Relacija “IS-A” - implementacija
- Implementacija (realizacija) interfejsa se prikazuje na sličan način kao nasleđivanje samo što je linija strelice isprekidana



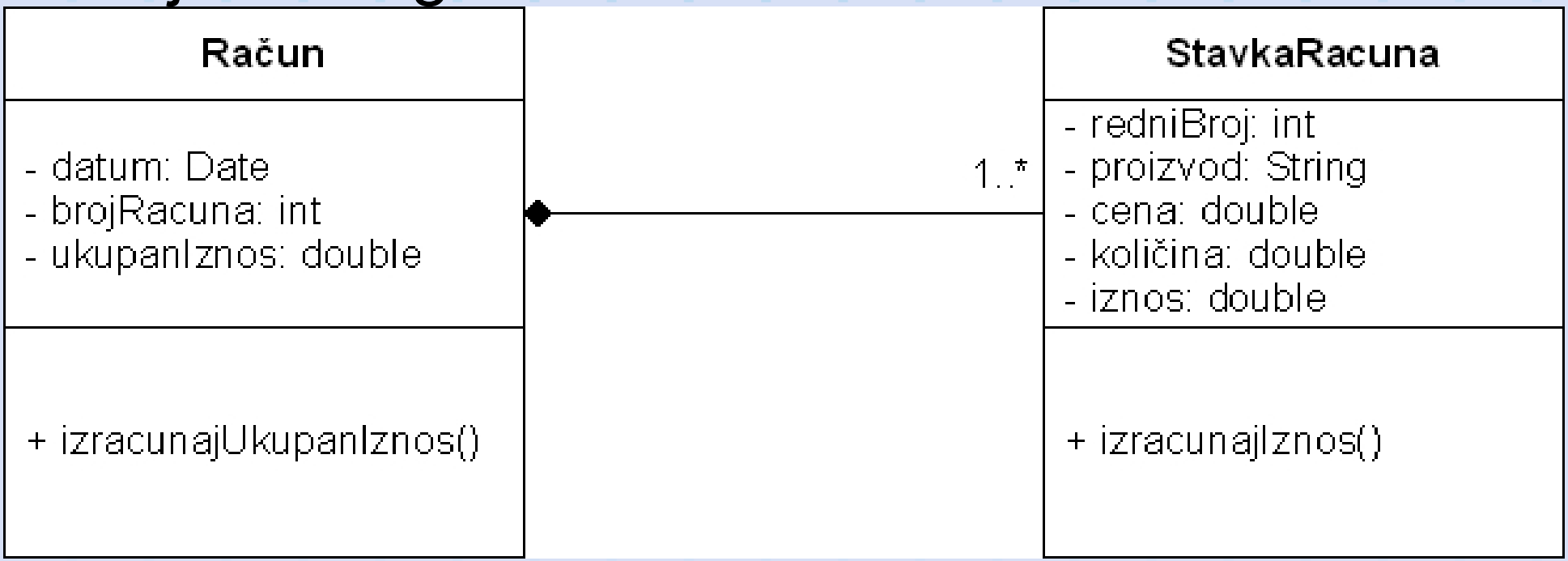
UML dijagram klasa

- Relacija asocijacije (opšta)
- Predstavlja se punom linijom između klasa koje su u relaciji



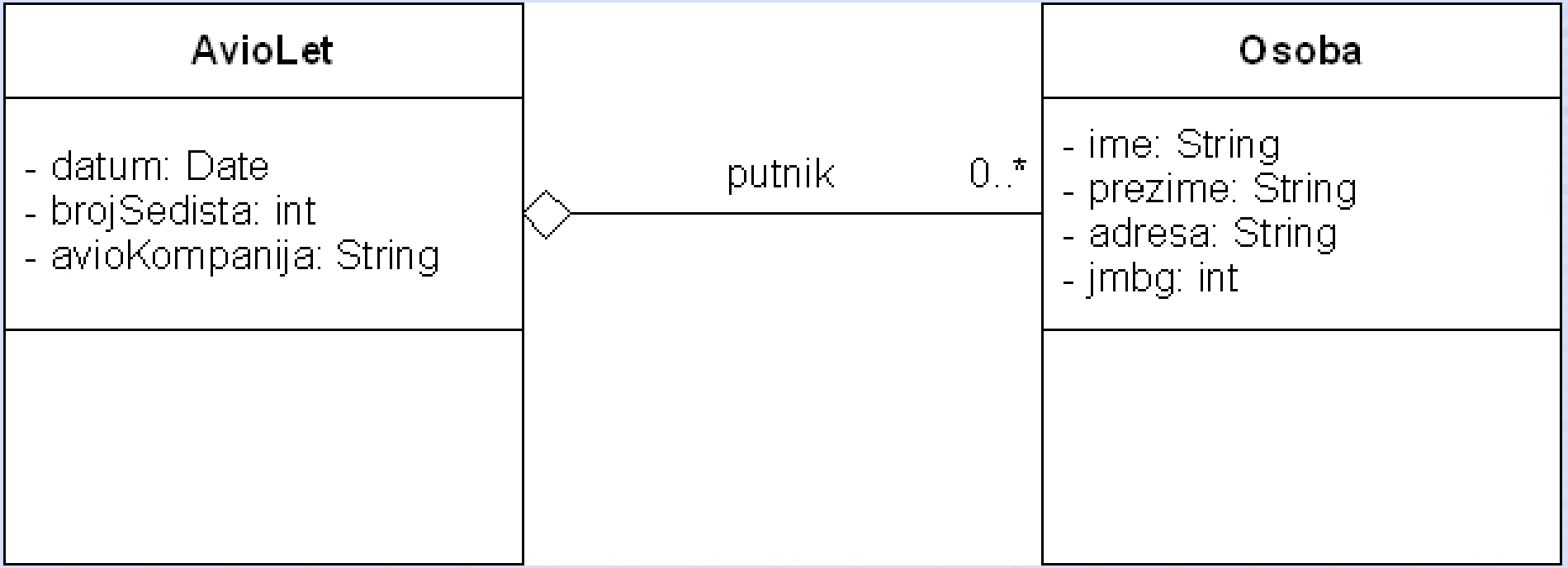
UML dijagram klasa

- Relacija “HAS-A” - kompozicija (posebna vrsta asocijacije)
- Objekat jedne klase sadrži, kao sastavni deo, objekte druge klase
- Objekti druge klase nisu samostalni



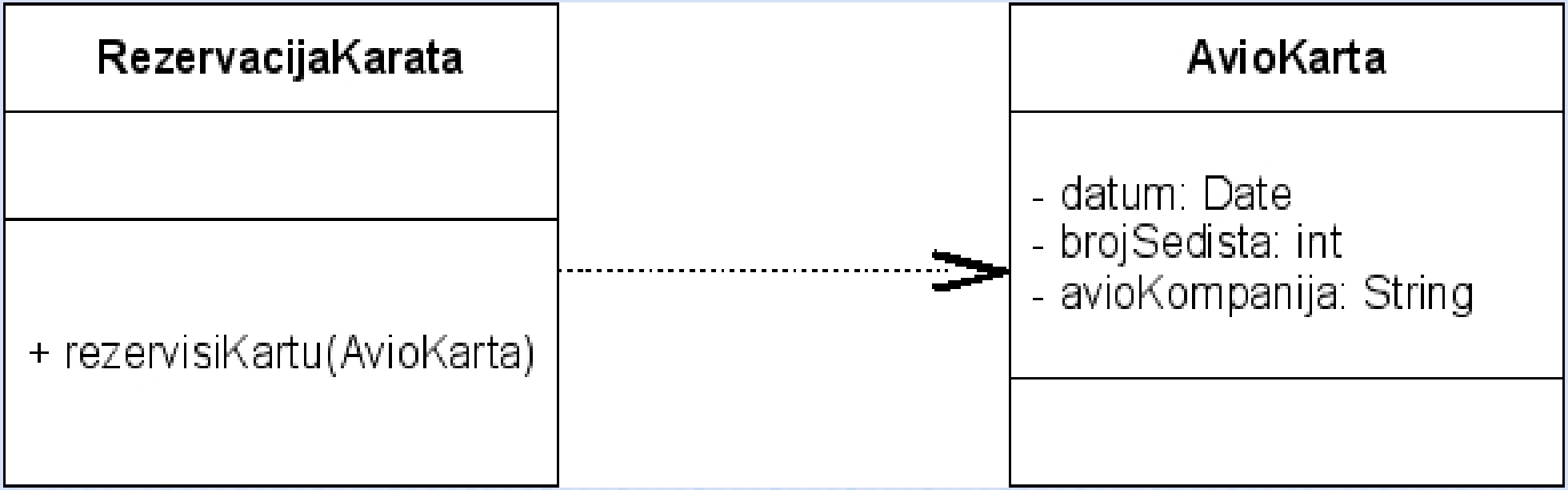
UML dijagram klasa

- Relacija “HAS-A” - agregacija (posebna vrsta asocijacije)
- Objekat jedne klase sadrži, kao sastavni deo, objekte druge klase
- Objekti druge klase mogu da budu samostalni



UML dijagram klasa

- Relacija “USING” - korišćenje
- Posebna vrsta relacije u kojoj jedna klasa koristi objekte druge klase ali kao lokalne promenljive (ne atribute), povratne vrednosti metoda ili parametre metoda

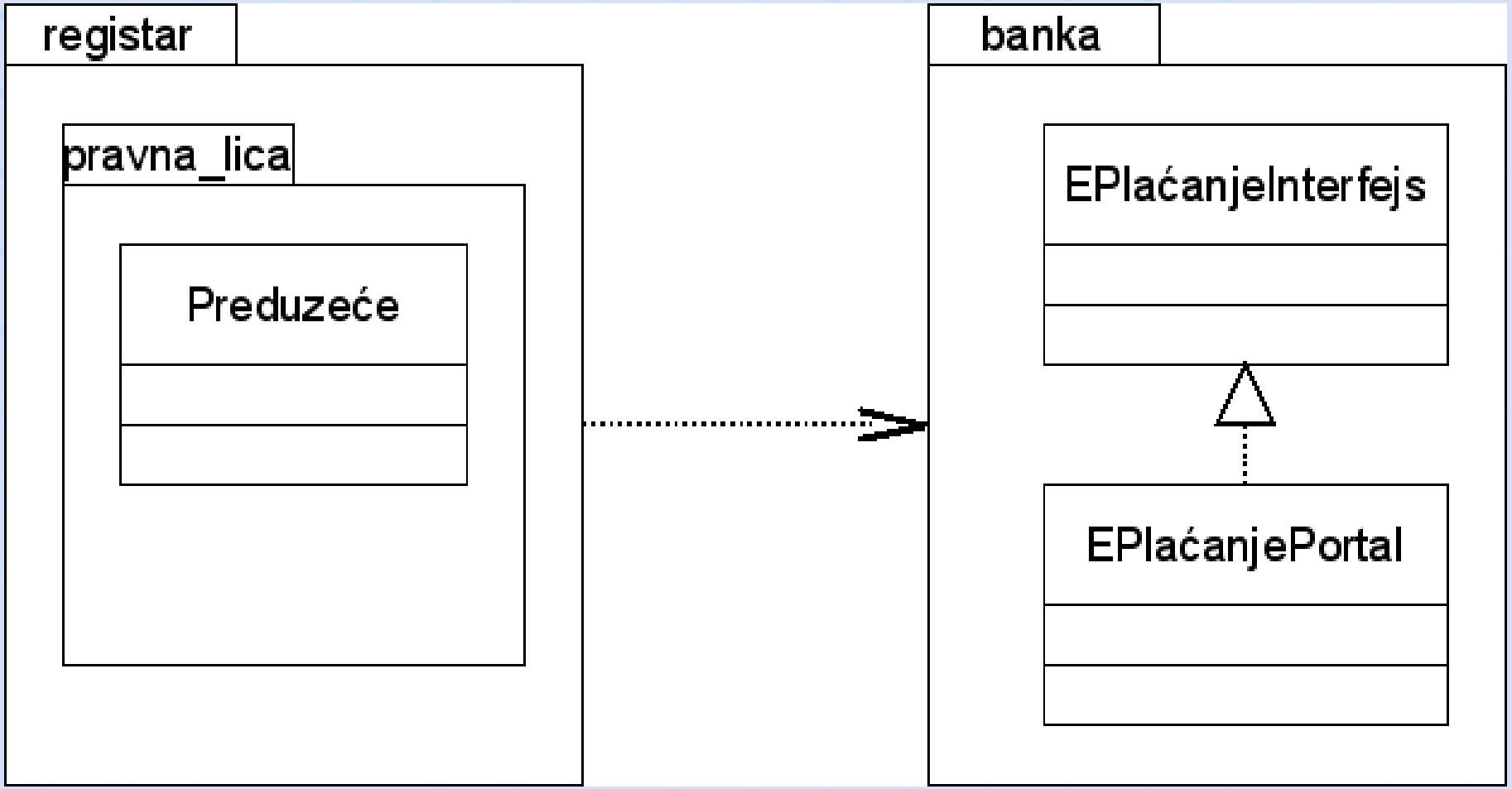


UML dijagram paketa

- Svrha: pregled zavisnosti između elemenata velikog sistema
- Svaki paket je predstavljen posebnim elementom
- Svaki paket sadrži druge UML elemente
- Paketi mogu biti ugnježdeni tj. formirati hijerarhiju
- Moguće je prikazati veze između elemenata u paketu kao veze između paketa

UML dijagram paketa

- Zavisnost između paketa se prikazuje kao relacija korišćenja

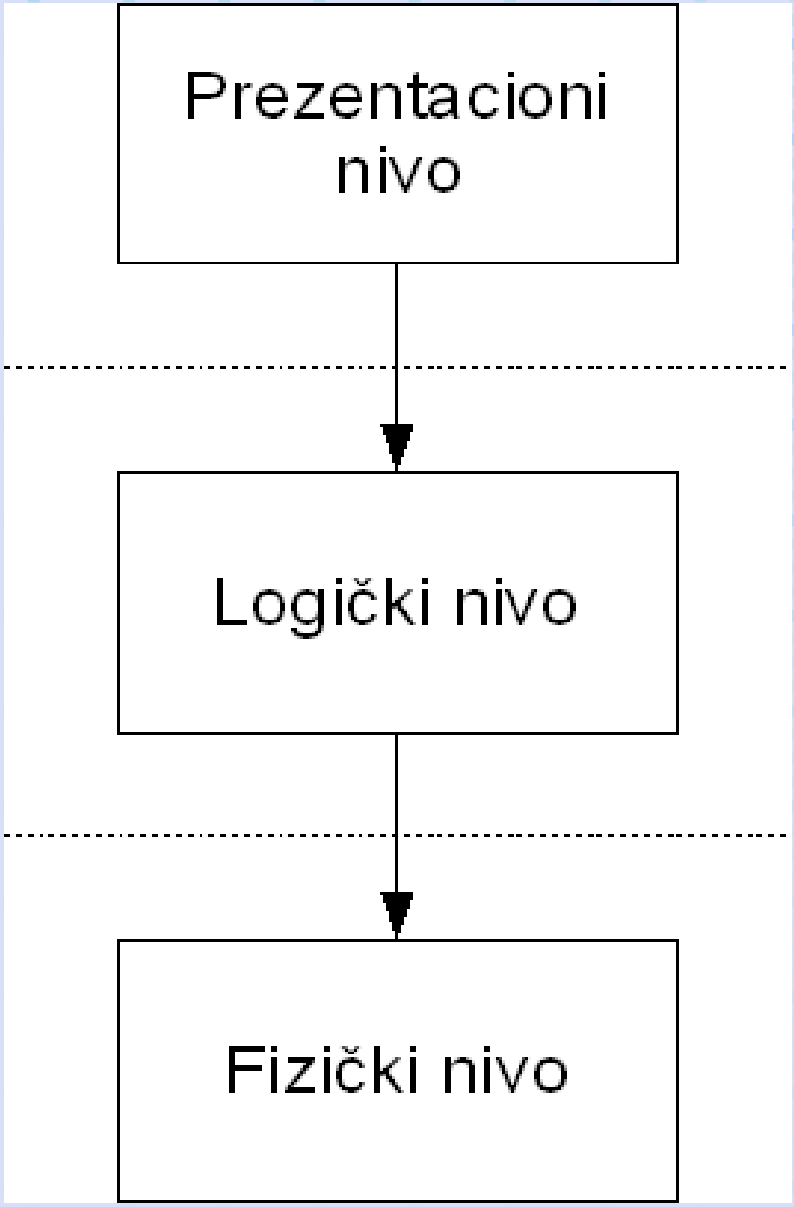


Tronivojska (troslojna) arhitektura

- Three-tier architecture
- Često se razvija i u “višenivojsku” arhitekturu (Multi-tier architecture) daljim raslojavanjem
- Ideja: razdvojiti softverski sistem u različite nivoe (slojeve) radi lakše nadogradnje i održavanja
- Svaki nivo bi trebalo da se bavi samo nekom vrstom aktivnosti
- Nivoi bi trebalo da međusobno budu što manje zavisni

Tronivojska (troslojna) arhitektura

- Tri osnovne vrste aktivnosti
  - Aktivnosti prezentacije (grafičkog prikaza)
  - Aktivnosti poslovne logike
  - Aktivnosti fizičkog skladištenje podataka
- Prema tome su nastala tri nivoa



Tronivojska (troslojna) arhitektura

- Prezentacioni (korisnički) nivo
  - Isključivo vrši funkciju prikaza podataka krajnjim korisnicima i unosa podataka
  - Svaki korisnik ima svoj “pogled” na sistem
  - Prezentacioni nivo **ne sme da sadrži poslovnu logiku**
  - Ovaj nivo **vrši pozive isključivo ka logičkom nivou** i **ne može direktno da pristupa fizičkom nivou**
  - Logički nivo i fizički nivo “ne vide” korisnički nivo i ne mogu da vrše pozive ka korisničkom nivou (pozivi su usmereni **“odozgo nadole”**)

Tronivojska (troslojna) arhitektura

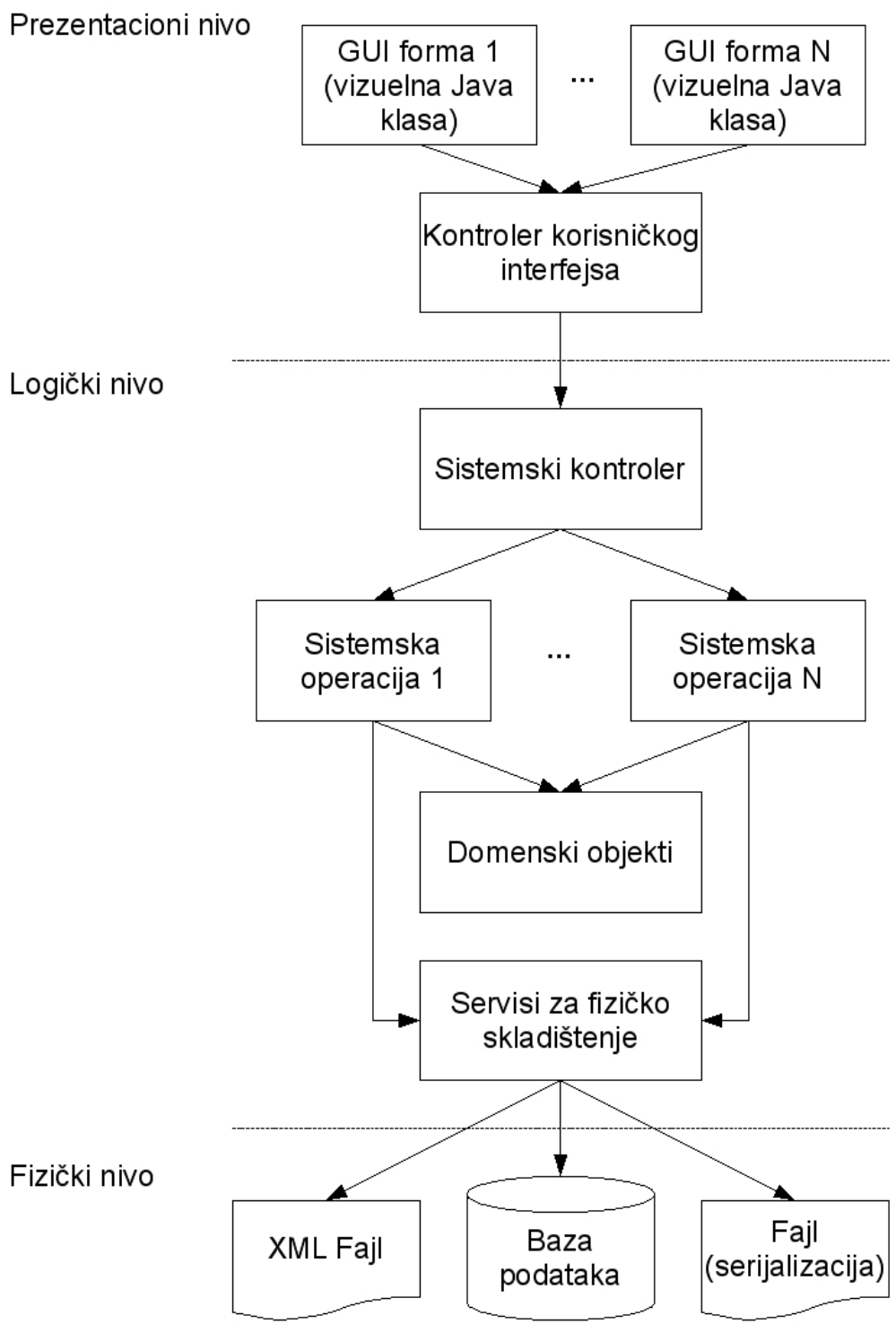
- Logički nivo
  - Isključivo sadrži poslovnu logiku (logička kontrola, domenski objekti, sistemske operacije)
  - **Ne sme da sadrži kod za prikaz podataka** niti za unos podataka
  - Može da **poziva isključivo fizički nivo** (pozivi sa fizičko skladištenje)
  - **Ne može da poziva korisnički nivo**

Tronivojska (troslojna) arhitektura

- Fizički nivo
  - Uređaji i softver za fizičko skladištenje podataka
  - **Ne sme da sadrži poslovnu logiku niti deo koji se tiče prikaza podataka**
  - **Ne može da poziva nijedan drugi nivo tj. ne “vidi” nijedan drugi nivo**

Tronivojska arhitektura

- Jedan od načina implementacije tronivojske arhitekture u Javi
- Pozivi metoda se vrše isključivo “odozgo-nadole”
- Izuzeci obezbeđuju propagaciju poruka o greškama do gornjeg nivoa



Tronivojska (troslojna) arhitektura prezentacioni nivo

GUI forme (vizuelne Java klase - javax.swing paket)

- Ne sme da sadrži poslovnu logiku niti logičku kontrolu već to preusmerava na niže nivoe
- Jedino sme da sadrži logiku za prikaz podataka
- Poziva isključivo metode kontrolera korisničkog interfejsa i “ne vidi” ništa od klasa koje se nalaze na logičkom nivou
- “Hvata” izuzetke sa nekog nižeg nivoa i prikazuje ih pozivanjem neke pomoćne forme ili već gotovog dijaloga (npr. JOptionPane)

Tronivojska (troslojna) arhitektura prezentacioni nivo

Kontroler korisničkog interfejsa (obična Java klasa)

- Samo **jedna instanca** na nivou programa koju dele sve GUI forme
- Uz pomoć ovog kontrolera se pokreće program
- Pravi i prikazuje GUI forme i sadrži logiku za navigaciju kroz GUI forme
- Prima ulazne podatke sa GUI formi u “sirovom” obliku (onako kako su uneti u GUI - npr. String) , pretvara ih u odgovarajući tip i pravi domenske objekte

Tronivojska (troslojna) arhitektura prezentacioni nivo

Kontroler korisničkog interfejsa (obična Java klasa)

- Poziva niže nivoe isključivo preko Sistemskog kontrolera i “ne vidi” ništa sa logičkog nivoa osim ovog kontrolera

Tronivojska (troslojna) arhitektura logički nivo

Sistemski kontroler (obična Java klasa)

- Poziva sistemske operacije da se izvrše
- Služi kao jedinstvena “fasada” prema prezentacionom nivou
- Samo **jedna instanca** na nivou programa

Tronivojska (troslojna) arhitektura logički nivo

Sistemske operacije (obične Java klase)

- Primer: SOUcitajKupca, SOSacuvajRacun...
- Klase koje implementiraju ponašanje sistema
- Svaka sistemska operacija se implementira sa po jednom klasom i sadrži samo jednu metodu (obično statičku)
- Ove klase i njihove metode barataju sa domenskim objektima i sadrže svu poslovnu logiku, uključujući i onu za fizičko skladištenje (obično pozivi ka nekim Java bibliotekama)

Tronivojska (troslojna) arhitektura logički nivo

- **Domenske klase (obične Java klase)**
- Primer: Račun, Kupac
- Klase koje ne implementiraju ponašanje sistema već stanje i logičku kontrolu ulaznih podataka
- Perzistentne klase su one koje se skladište i koje se mogu učitati i posle gašenja programa
- U njima se NE NALAZI kod za fizičko skladištenje

Tronivojska (troslojna) arhitektura fizički nivo i servisi za skladištenje

Baza podataka

- JDBC (Java DataBase Connectivity)
- Hibernate
- Java Persistance API

XML

- Dom4J
- SAX

Običan fajl (serijalizacija)