Tema 10: **UML specifikacija, tehnike i dijagrami**

PROJEKTOVANJE INFORMACIONIH SISTEMA

dr Vladislav Miškovic

FAKULTET ZA POSLOVNU INFORMATIKU

2011/2012

Sadržaj predavanja

1. Uvod
2. Osnovni UML dijagrami
3. Upotreba UML dijagrama u projektu softvera
4. Primer objektno-orijentisanog razvoja

1. Uvod

- Nastanak standarda UML
- Jezik za modeliranje UML
- UML 2.0 dijagrami

Nastanak standarda

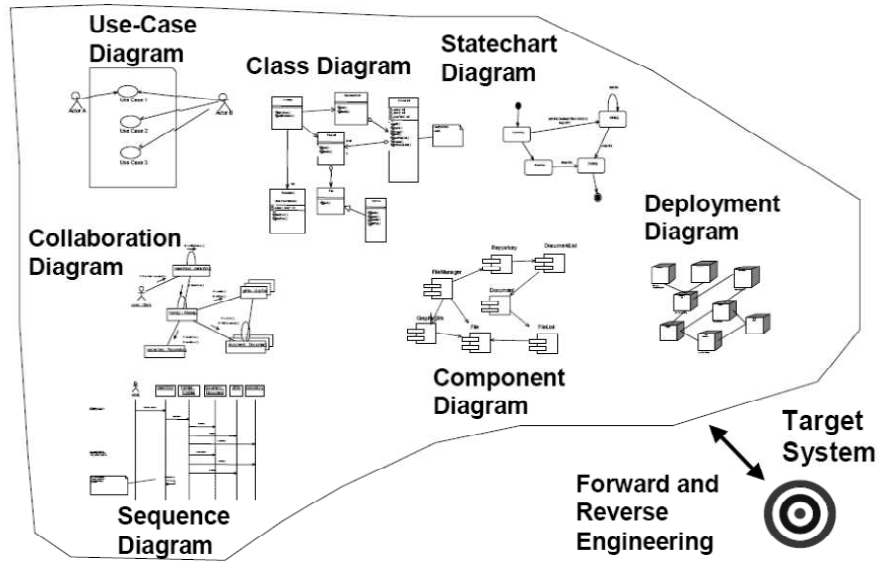
- Objektno orijentisani programski jezici (1960-te,1970-te): *Simula* (Kristen Nygaard), *Smalltalk* (Alan Kay)
- Objektno orijentisana analiza i projektovanje: Grady Booch, *Object-Oriented Design* (1982)
- Autori Booch, Rumbaugh i Jacobson osnovali Rational Software Inc 1994/1995 www.rational.com
- UML verzija 0.8 u oktobru 1995
- UML definisalo udruženje *Object Management Group* (www.omg.org), svetski konzorcijum prodavaca objektno orijentisanih proizvoda
- UML standardizovan 1997, sadašnja UML verzija 2.x

Jezik za modeliranje UML (*Unified Modeling Language*)

- UML je jezik za specifikaciju, vizualizaciju, konstruisanje i dokumentovanje razvoja softvera
- Jezik za modeliranje čine:
  - elementi modela: koncepti i semantika
  - notacija: vizuelni izgled elemenata modela
  - preporuke: za upotrebu elemenata i notacije

UML 2.0 dijagrami

- Definisano 13 dijagrama u tri kategorije, koji opisuju različite aspekte sistema:
  - statička struktura (6)
  - opšte ponašanje objekata (3)
  - interakcije objekata (4)



Osnovni UML dijagrami prema redosledu izrade

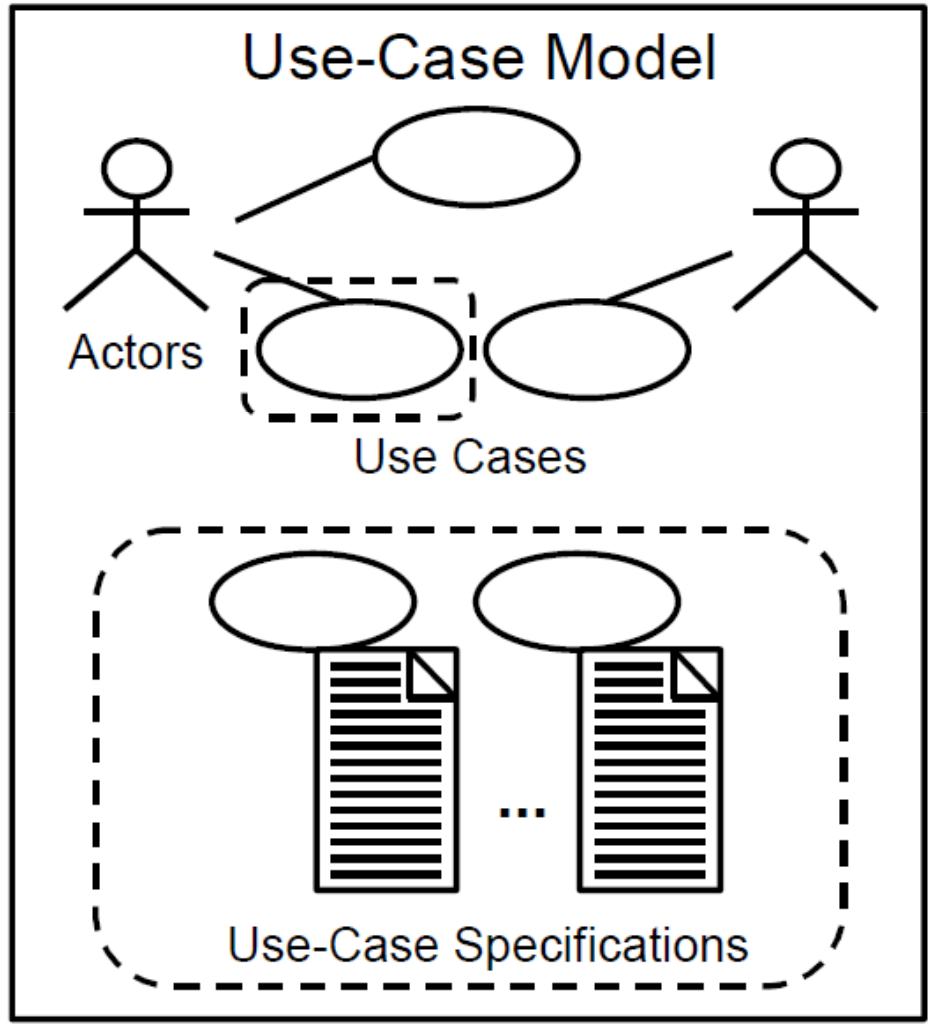
1. Dijagrami slučajeva korišćenja (*Use-Case Diagrams*)
2. Dijagrami klasa i objekata (*Class & Object Diagrams*)
3. Dijagrami ponašanja (*Behavior*)
   - dijagrami stanja (*Statechart Diagrams)*
   - dijagrami aktivnosti (*Activity Diagrams*)
4. Dijagrami interakcije (*Interaction*)
   - dijagram sekvenci (*Sequence Diagram*)
   - dijagram saradnje (*Collaboration Diagram*)
5. Dijagrami implementacije (*Implementation*)
   - dijagram komponenti (*Component Diagram*)
   - dijagram realizacije (*Deployment Diagram*)

2. Osnovni UML dijagrami

1. Dijagrami slučajeva korišćenja
2. Dijagram klasa
3. Dijagrami ponašanja
4. Dijagrami interakcije
5. Dijagrami implementacije

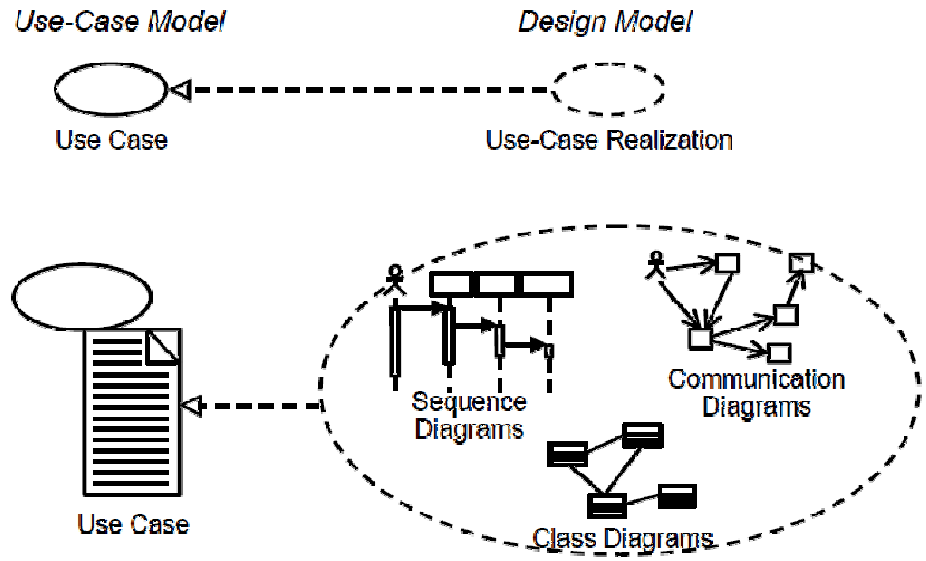
2.1 Dijagrami slučajeva korišćenja

- Dijagram slučajeva korišćenja (*Use Case Model*)
  - opis interakcija između sistema i korisnika / eksternih sistema
- Specifikacija korisničkih zahteva obuhvata još
  - Specifikacije (*Use-Case narrative/Specifications*)
    - *tekstualni* opis zadataka i načina na koji će korisnik saobraćati sa sistemom radi izvršenja zadataka
  - Rečnik (*Glossary*)
  - Dodatne specifikacije



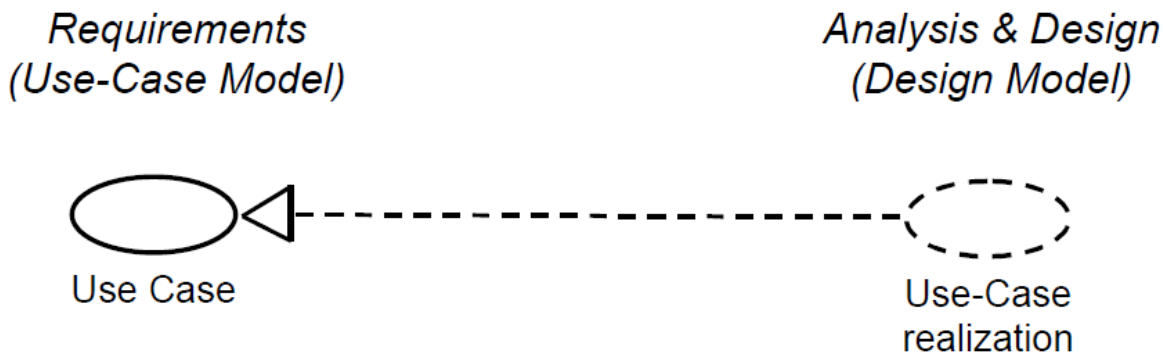
Realizacija slučaja korišćenja

- Predstavlja opis slučaja korišćenja i projektnom modelu (*design model*)
  - koje *klase* se moraju izgraditi za svaki slučaj korišćenja
  - skup UML dijagrama koji modeluju
    - klase/objekte koji implementiraju slučajeve korišćenja i njihove relacije (*dijagram klasa*)
    - interakcije saradnje (*dijagrami saradnje i sekvenci*), koji pokazuju u kakvoj su interakciji klase/objekti kada izvršavaju slučaj korišćenja



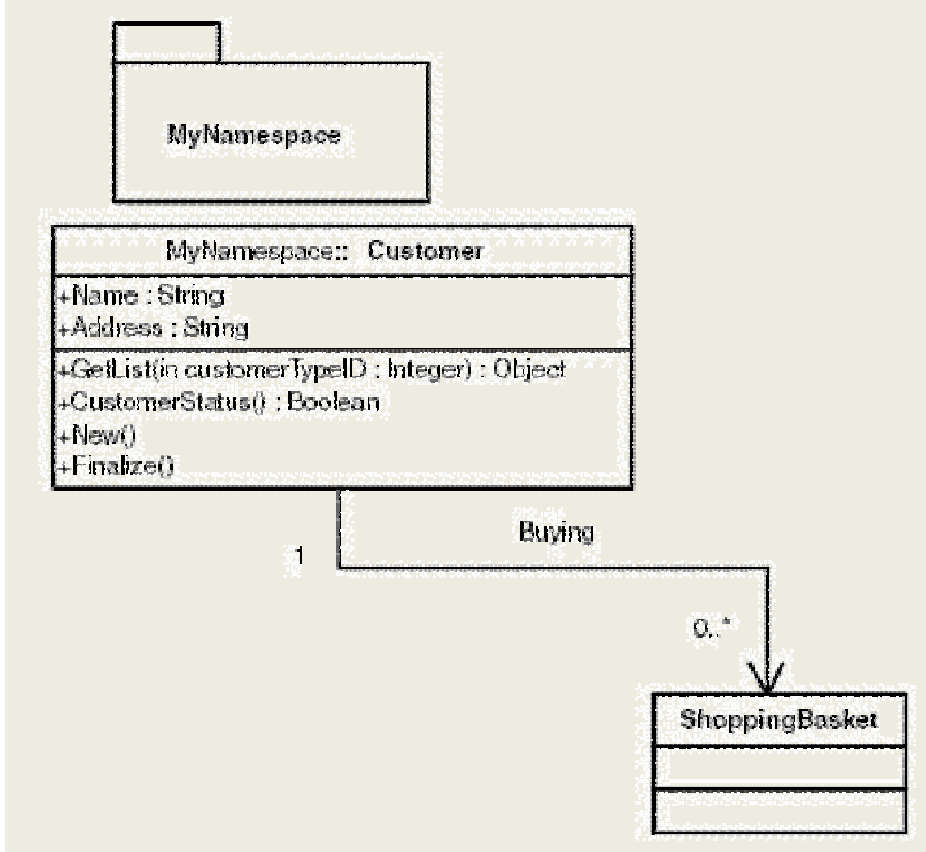
Značaj realizacije slučaja korišćenja

- Omogućava praćenje realizacije korisničkih zahteva kroz ceo proces analize i projektovanja
- Odgovornost arhitekte softvera



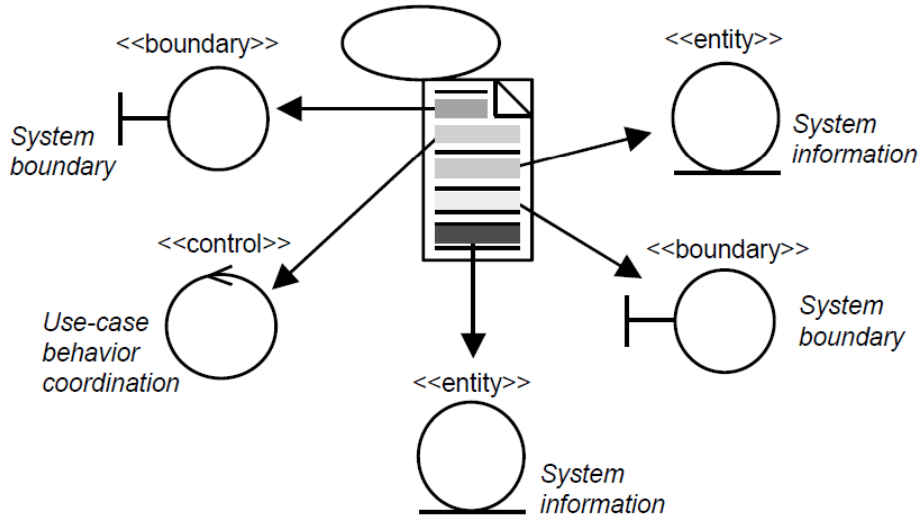
2.2 Dijagram klasa *(Class Diagram)*

- Model objekata i relacija između objekata
  - klase
    (iz *Use Case* i dr.)
    - atributi
    - metodi
  - relacije



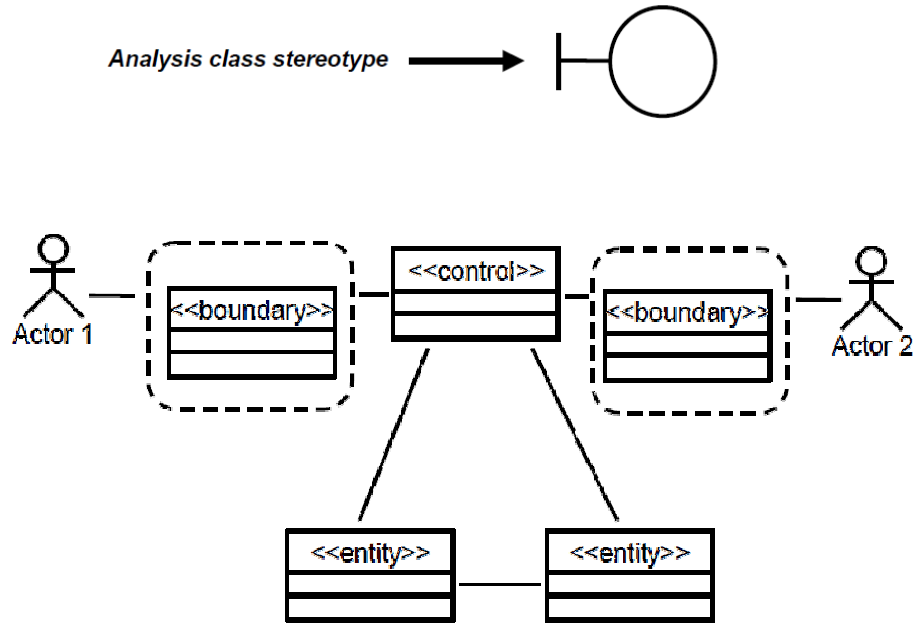
Analitičke klase

- Ponašanje slučaja korišćenja se raspoređuje na klase
- Vrste
  - granična klasa
  - klasa entiteta
  - klasa upravljanja



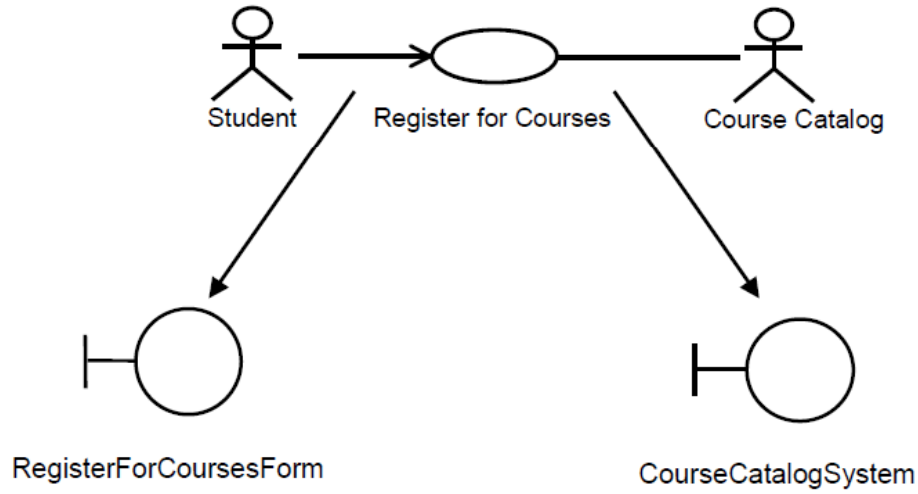
Granične klase (*boundary*)

- Posreduju između interfejsa i spoljašnjih entiteta
  - Po jedna granična klasa za svaki par akter-slučaj korišćenja
    - klasa korisničkog interfejsa
    - klasa sistemskog interfejsa
    - klasa interfejsa uređaja
  - Modeluju interakciju između sistema i okruženja



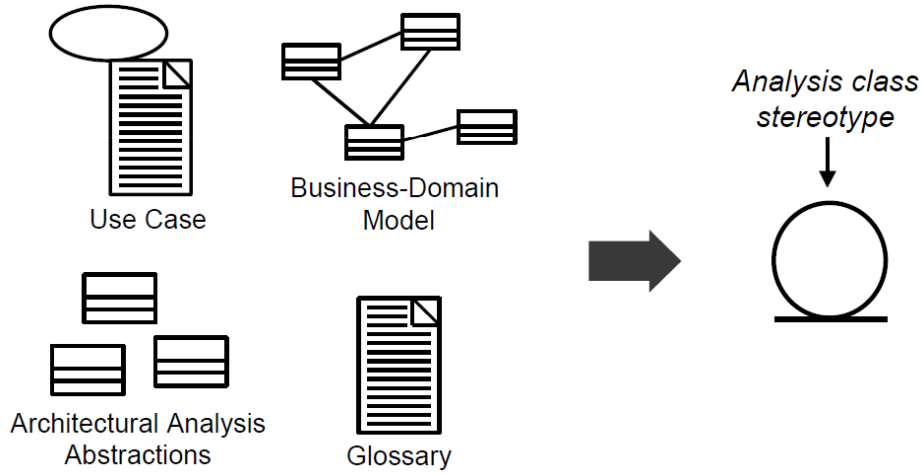
Primer: Identifikacija graničnih klasa

- Po jedna granična klasa za svakog aktera



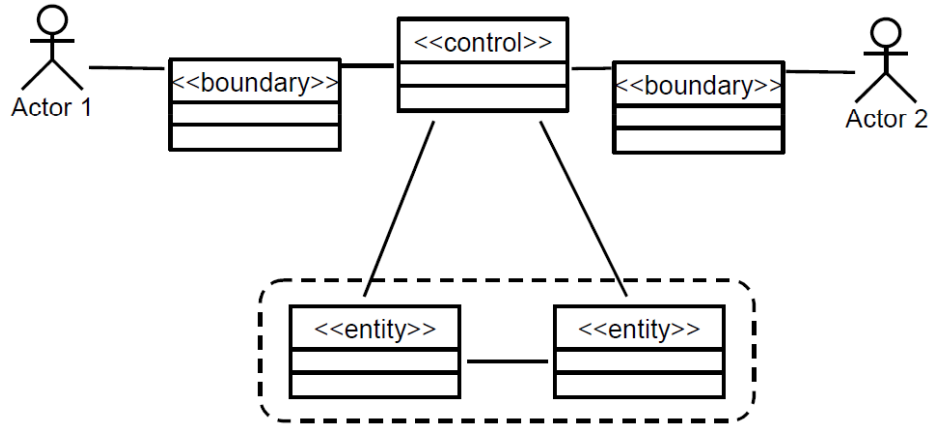
Klase entiteta (*entity*)

- Ključne *apstrakcije* u sistemu, nezavisne od okruženja
- Pojmovi iz
  - rečnika
  - poslovnog modela
  - tokova događaja slučajeva korišćenja
  - ključnih apstrakcija analize arhitekture
- Klase surogati
  - klase entiteta koje modeluju informacije o akterima *unutar* sistema



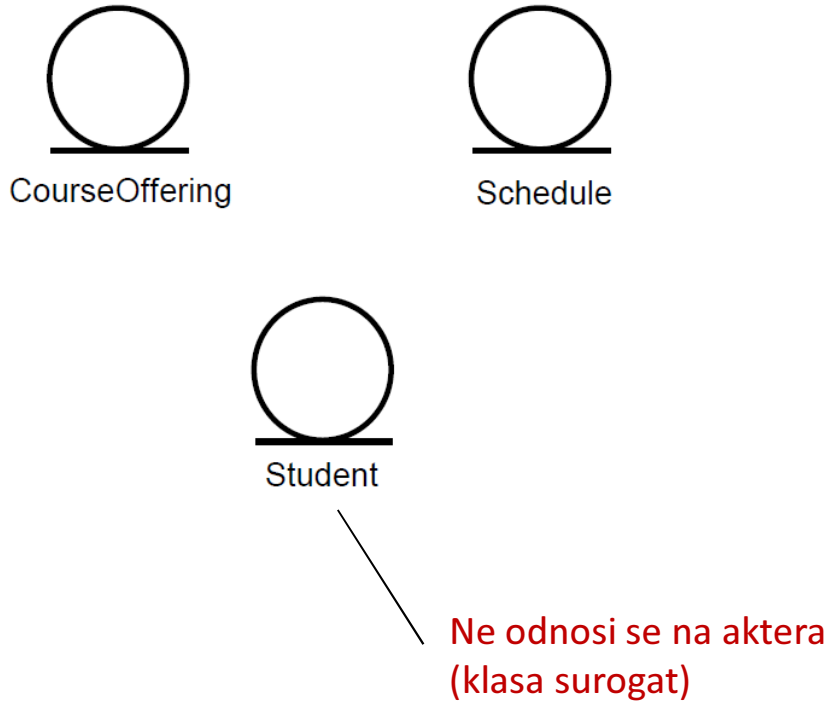
Uloga klasa entiteta

- Pamćenje informacija u sistemu
  - događaji, lica i drugi stvarni entiteti
- Upravljenje informacijama
  - perzistentni, atributi i relacije su dugoročni ili permanentni



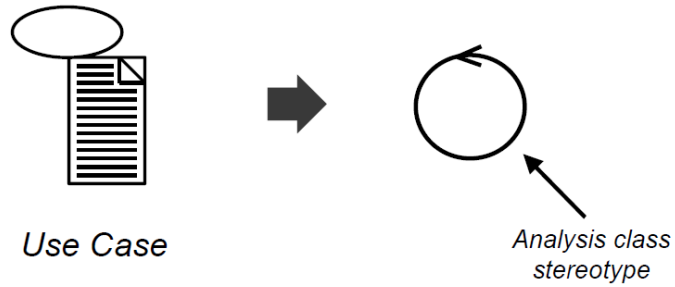
Primer: Identifikacija klasa entiteta

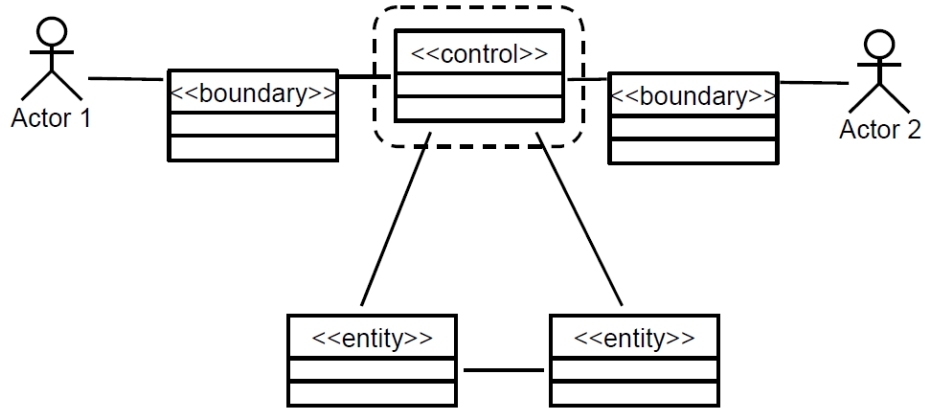
- Na osnovu toka događaja slučaja korišćenja
  - Ključne apstrakcije slučaja korišćenja
- Tradicionalna tehnika: selekcija imenica
  - označiti *imenice* u tekstu toka događaja slučaja korišćenja
  - ukloniti redundantne i dvosmislene
  - ukloniti *aktere*
  - ukloniti opis implementacije
  - ukloniti nazive svojstava (*atributi*, za kasnije) i *operacije*



Klase upravljanja (*control*)

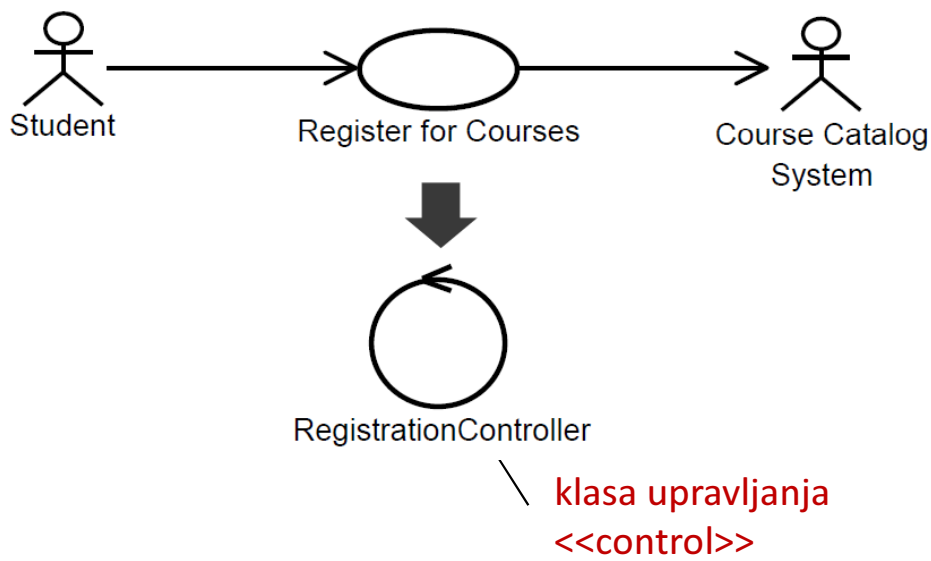
- Klasa upravljanja je koordinator ponašanja slučaja korišćenja
  - složeniji slučajevi korišćenja imaju jednu ili više klasa upravljanja
  - razdvajaju granične klase i klase entiteta
- Zavisi od slučaja korišćenja i okruženja



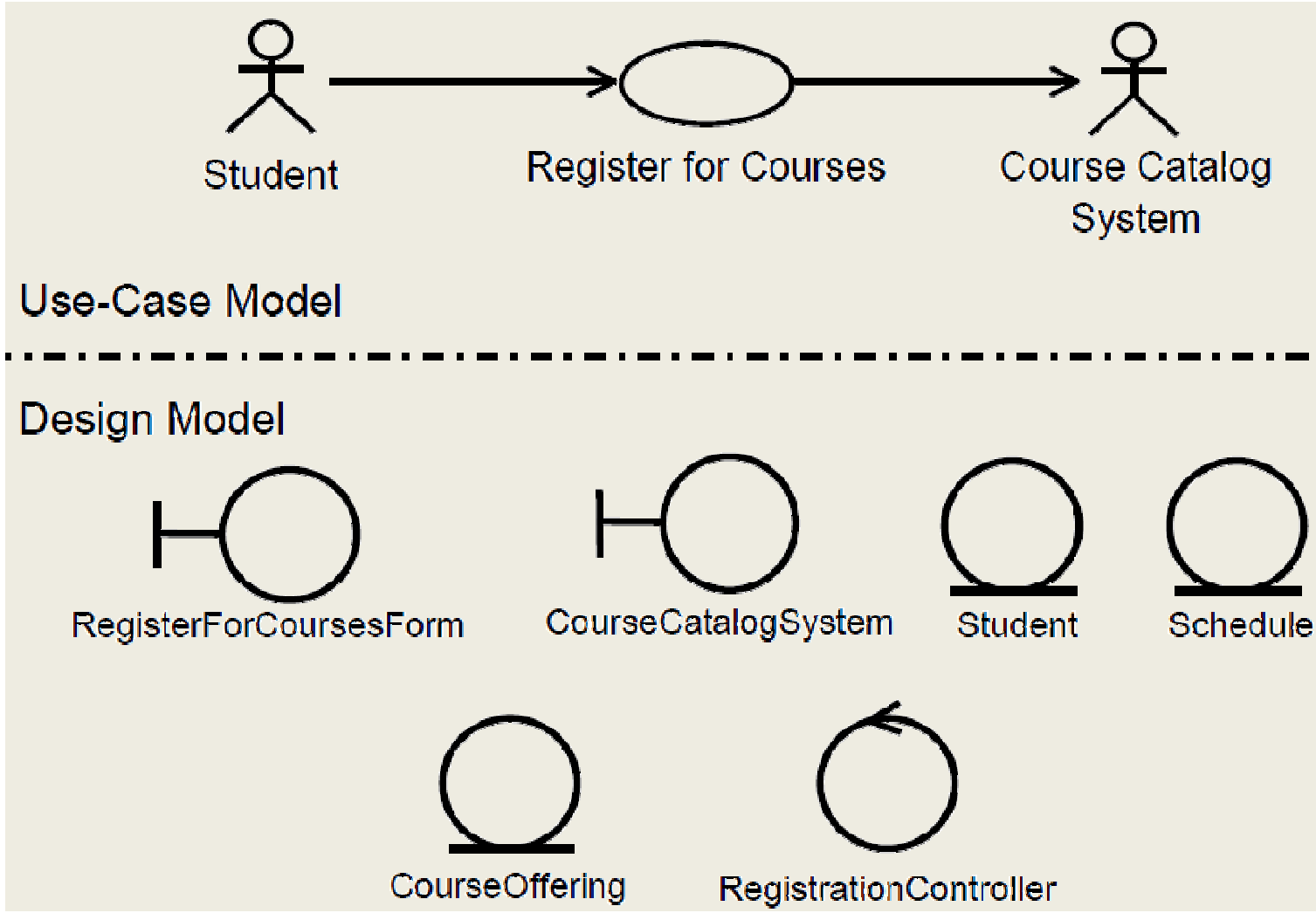


Primer: Identifikacija klasa upravljanja

- Načelno, jedna klasa upravljanja po slučaju korišćenja
  - u toku analize se broj klasa upravljanja može povećavati

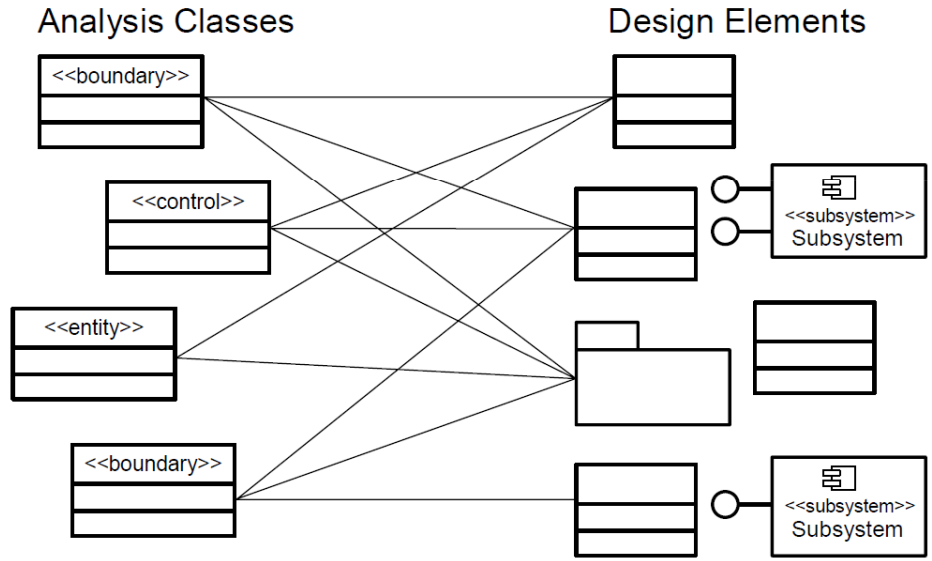


Ilustracija: Analitičke klase slučaja korišćenja



Prelazak sa analitičkog na projektni model

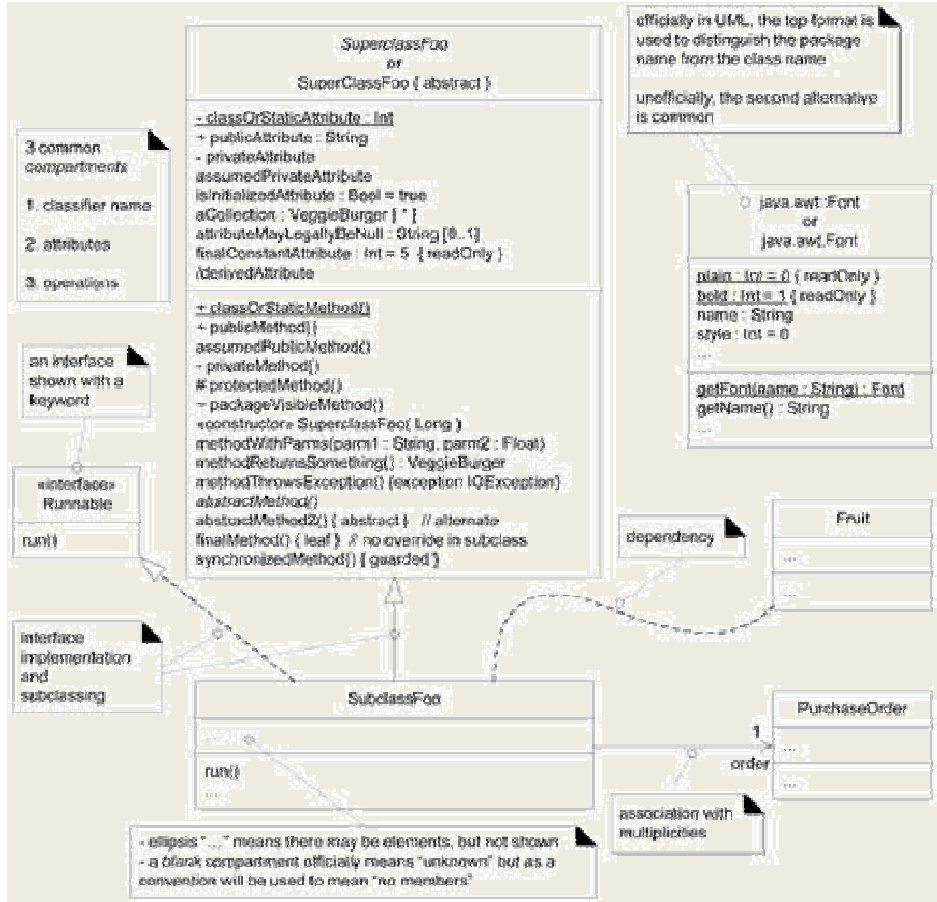
- Analitičke klase se kreiraju u toku analize slučajeva korišćenja
- U analizi klasa se vrši preciziranje analitičkih klasa
  - analitičke klase prikazuju funkcionalne *zahteve* u terminima problema
  - projektni model predstavlja nefunkcionalne zahteve, odnosno *rešenja*
- U procesu identifikacije projektnih klasa se određuje koje analitičke klase su
  - *projektne klase* (treba ih projektovati)
  - *podsistemi* (tek će se dekomponovati)
  - *postojeći elementi* (ne treba ih projektovati)



preslikavanje "više-prema-više" analitičkih klasa u projektne elemente

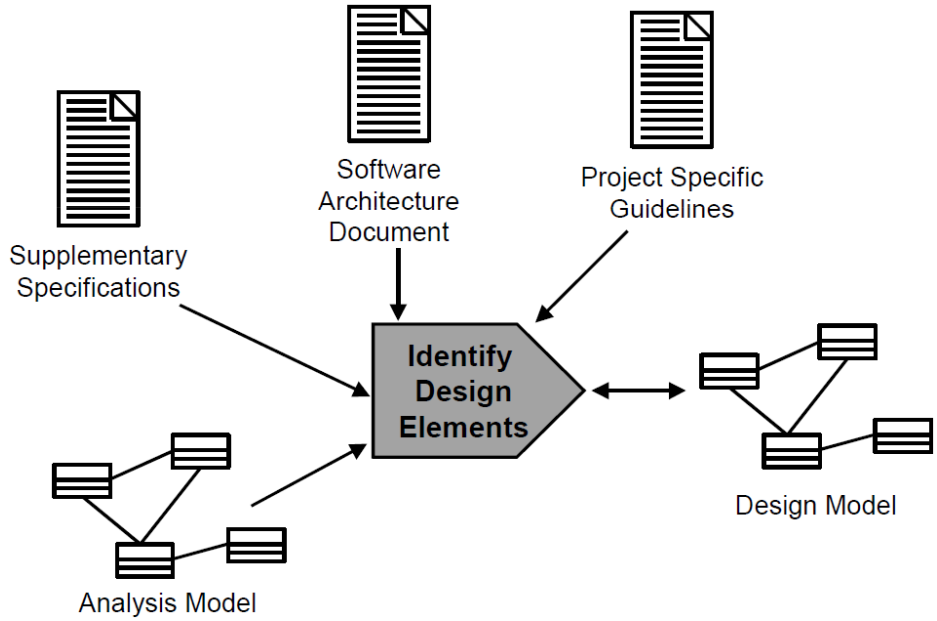
Elementi dijagrama klasa

- Dijagram klasa



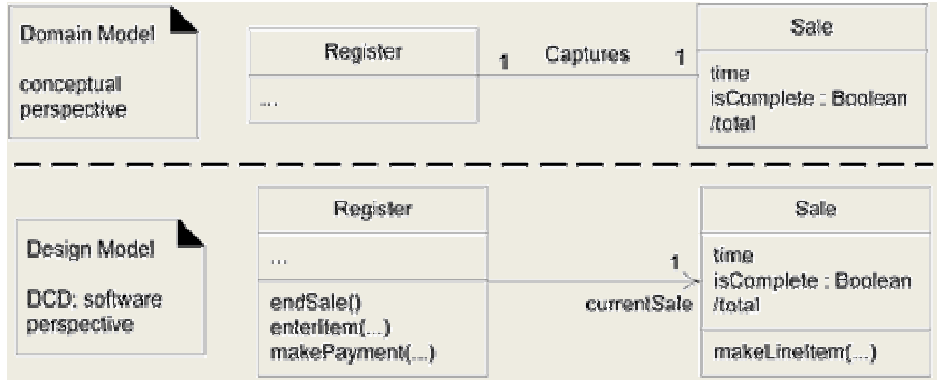
Proces identifikacije projektnih elemenata

- Identifikacija projektnih elemenata se vrši u svakoj iteraciji
- Analiziraju se interakcije *analitičkih* klasa radi pronalaženja elemenata projektnog modela
  - analitičke klase se menjaju u projektnom modelu (proširuju, sažimaju, kombinuju ili brišu)
- Rezultat: *projektni model*
  - klase
  - paketi
  - podsistemi



Različiti nivoi detaljnosti dijagrama klasa

- Konceptualni model
- Projektni model

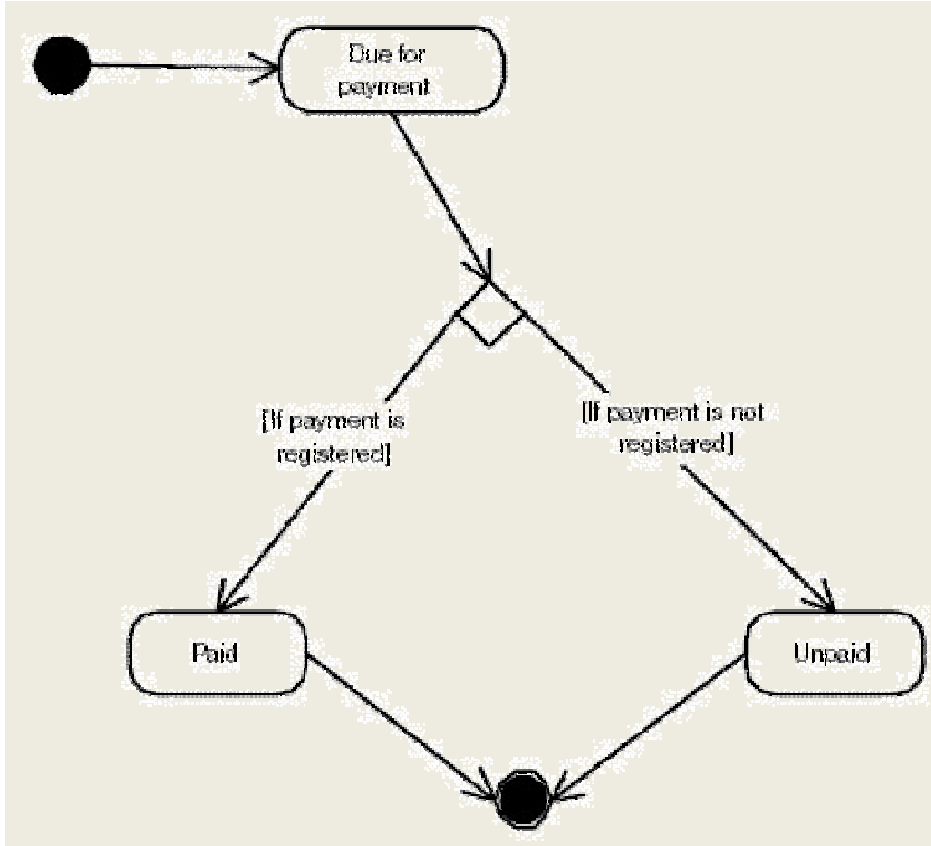


2.3 Dijagrami ponašanja

- Dijagrami stanja (*Statechart Diagrams)*
- Dijagrami aktivnosti (*Activity Diagrams*)

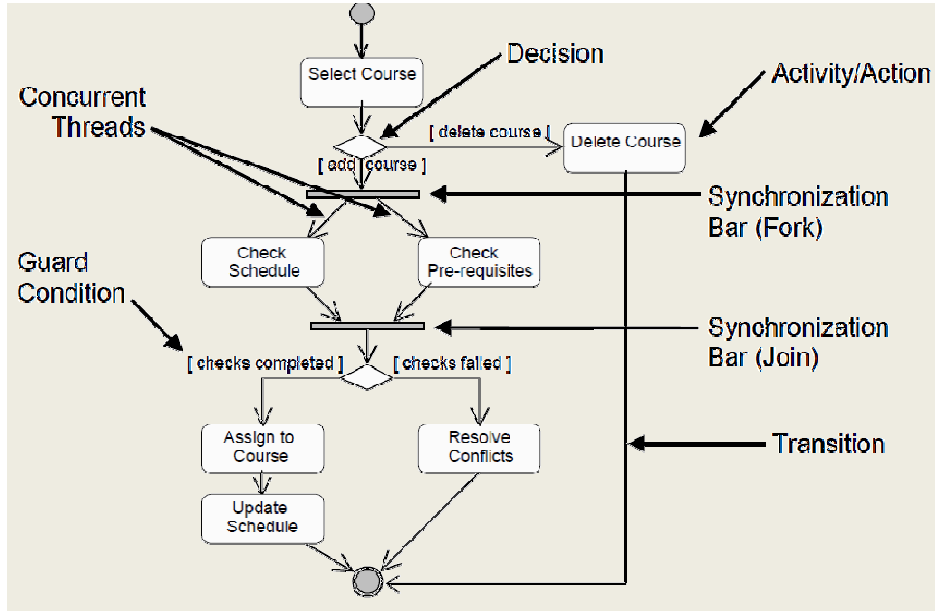
Dijagrami stanja (*Statechart Diagrams)*

- Opis internog ponašanja objekata
  - stanja (*states*)
  - prelazi (*transitions*)
  - grananja (*decisions*)



Dijagrami aktivnosti (*Activity Diagrams*)

- Opis aktivnosti u sistemu - model poslovnih procesa u analizi sistema, kao i logika slučajeva korišćenja (*Use Case*)
- Elementi
  - aktivnost
  - grananje
  - spajanje
  - početak
  - kraj
  - prekid (abort)

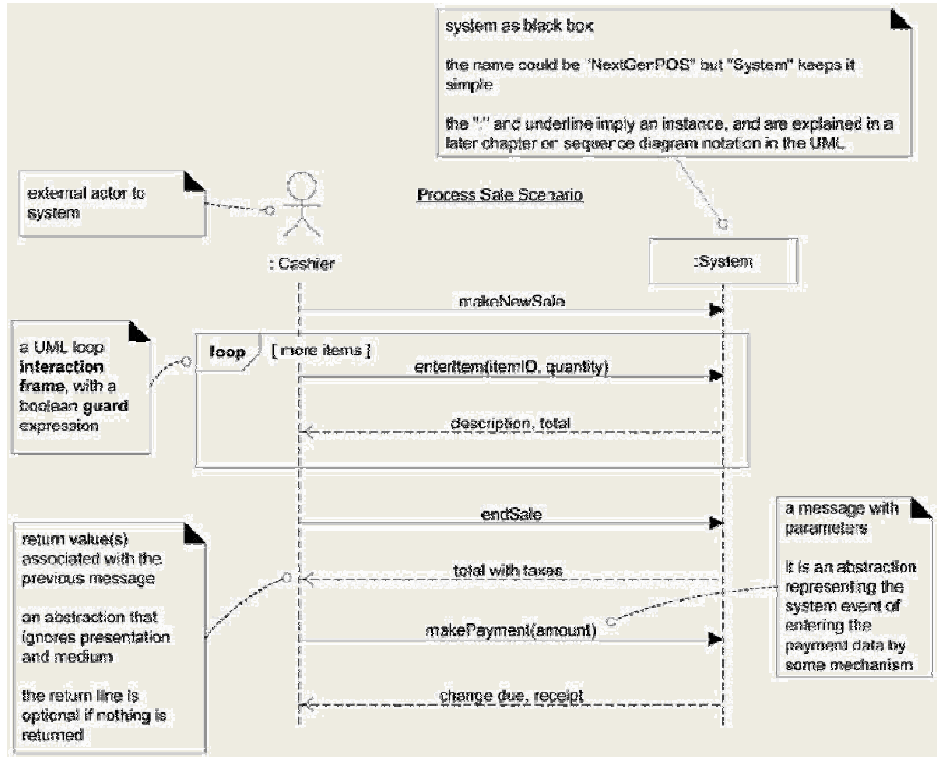


2.4 Dijagrami interakcije

- Dijagram sekvenci (*Sequence Diagram*)
- Dijagram saradnje (*Collaboration Diagram*)

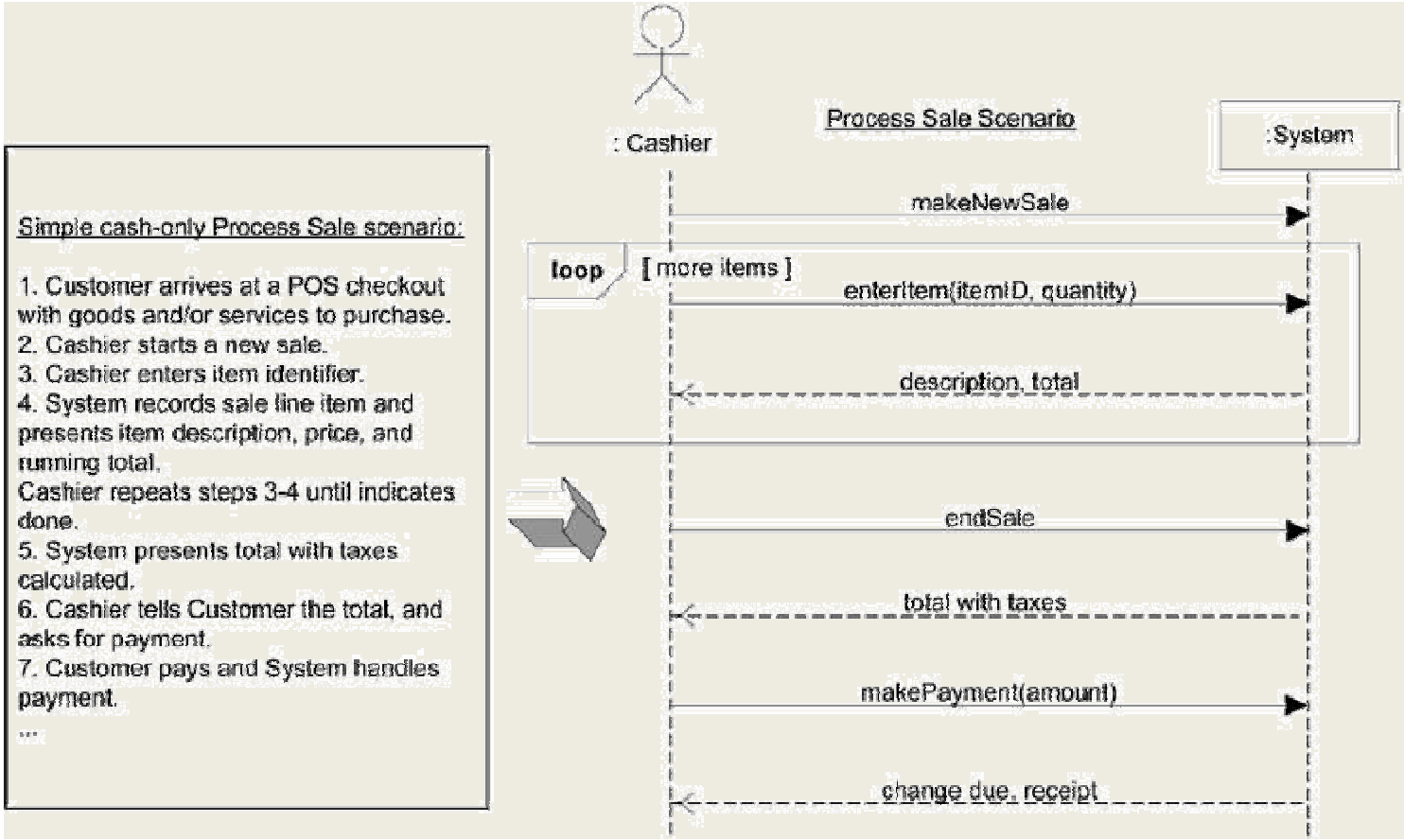
Dijagram sekvenci (*Sequence Diagram*)

- Elementi
  - vertikalna vremenska skala
  - objekti
  - poruke



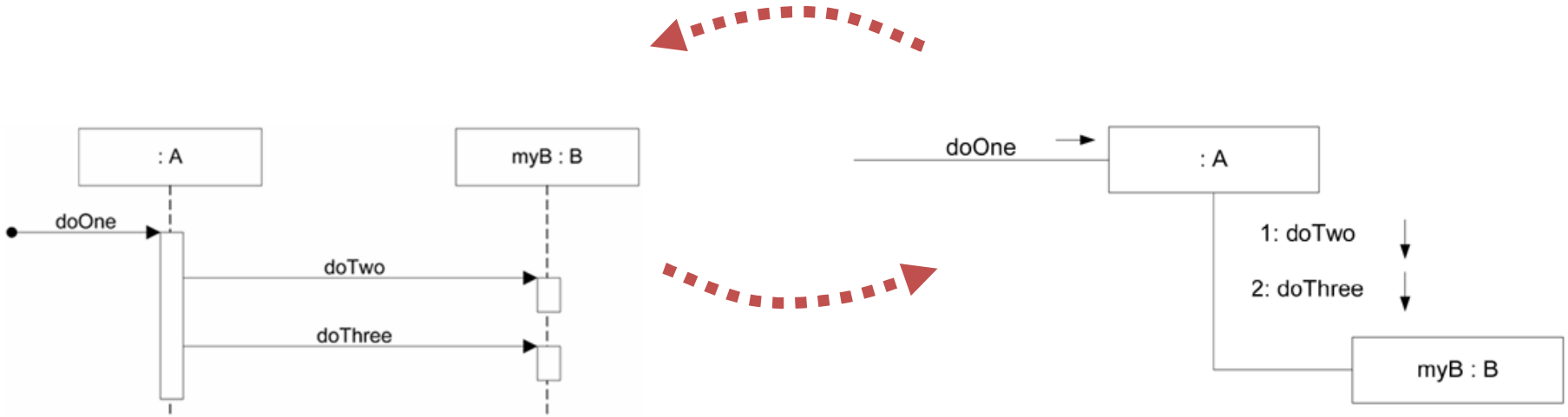
Veza dijagrama sekvenci i slučajeva korišćenja

- Dijagrami sekvenci se mogu koristiti u razradi slučajeva korišćenja



Dijagram saradnje (*Collaboration Diagram*)

- Dobija se iz dijagrama sekvenci (izomorfnost)
  - CASE alati ih konvertuju automatski

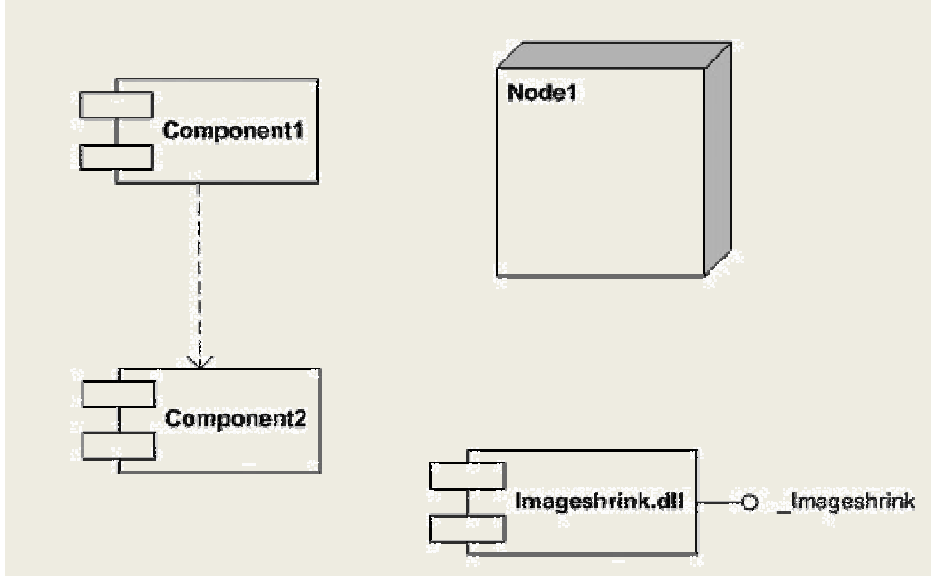


2.5 Dijagrami implementacije

- Dijagram komponenti (*Component Diagram*)
- Dijagram realizacije (*Deployment Diagram*)

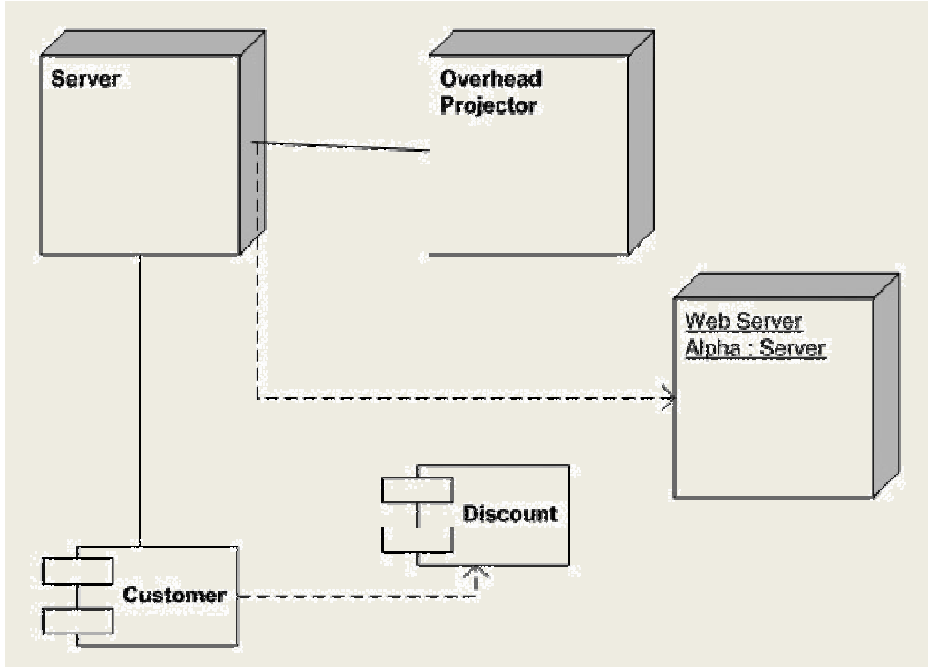
Dijagram komponenti (*Component Diagram*)

- Model fizičke implementacije softvera
  - komponente (*components*)
    - izvorni kôd
    - COM+
      - .exe, .dll
    - .NET
      - klase
      - Web servisi
  - čvorovi (*nodes*)
  - interfejsi (*interfaces*)
  - **zavisnosti** (*dependencies*)



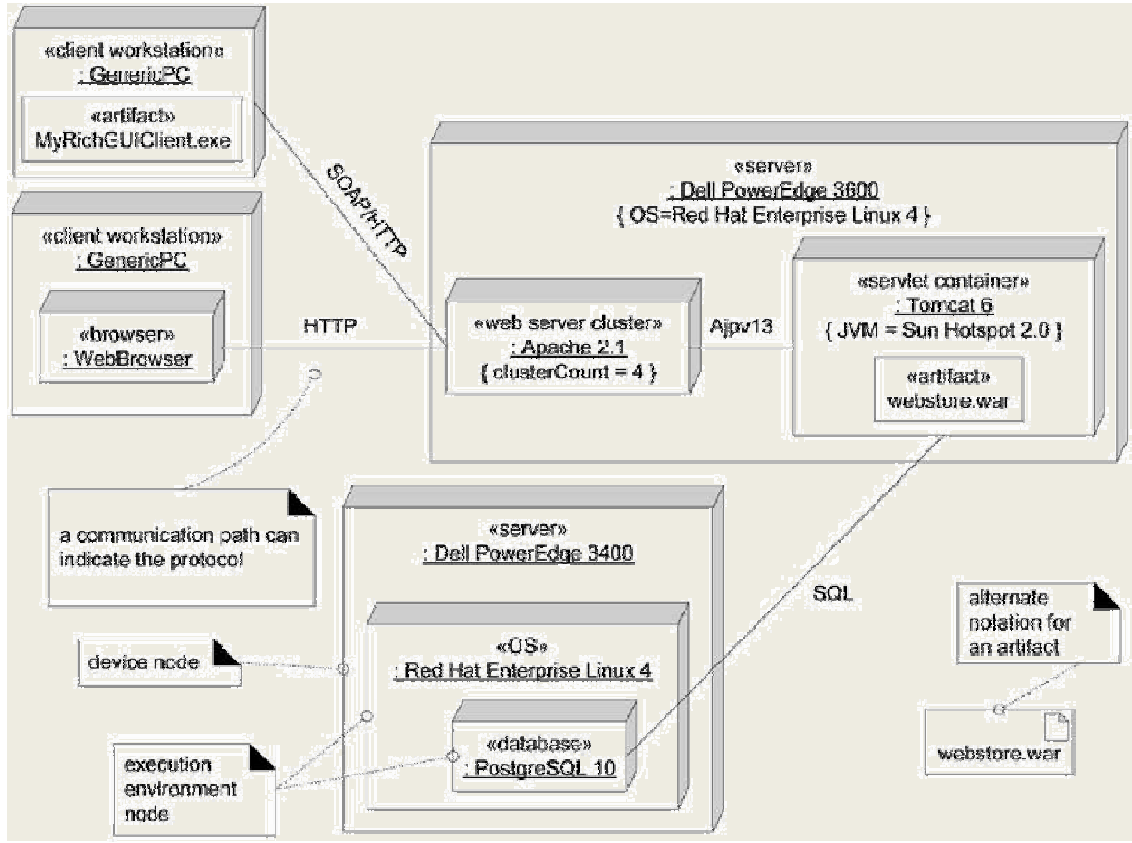
Dijagram realizacije (*Deployment Diagram*)

- Način realizacije rešenja
  - procesori
  - uređaji
  - komunikacije (procesor-uređaj)
- Definicija mrežne konfiguracije
- Alokacija (raspored) procesa na čvorove
- Definicija mehanizama distribucije



Primer: Dijagram realizacije/isporuke

- Konkretni softverski elementi, hardverski uređaji i komunikacije

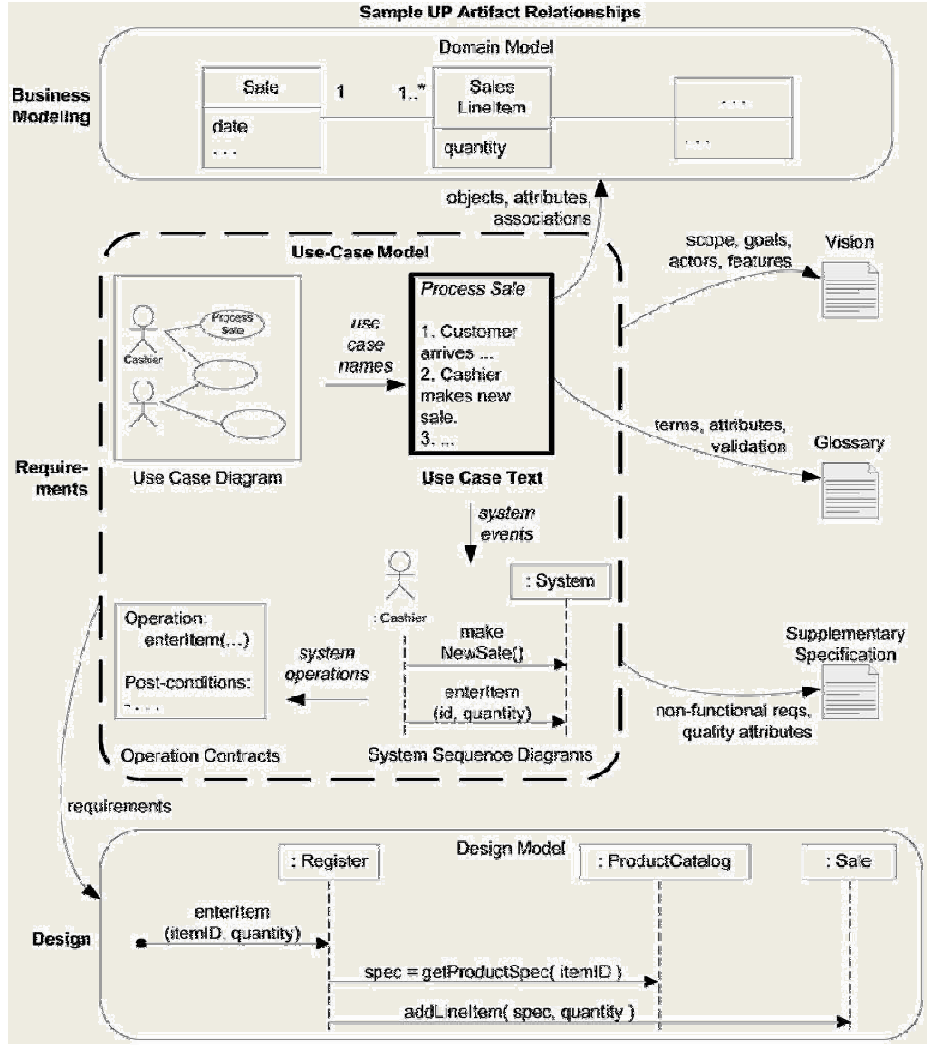


3. Upotreba UML dijagrama u projektu softvera

- Međusobna zavisnost proizvoda procesa OO razvoja
- Raspored

Međusobna zavisnost proizvoda procesa OO razvoja

- Na slici je prikaz zavisnosti nekih od produkata objektno-orijentisanog procesa razvoja softvera (UP)



4. Primer objektno-orijentisanog razvoja

1. Problem i zahtevi korisnika
2. Korisnički zahtevi (*Requirements*)
3. Dijagrami sekvenci (*Sequence*)
4. Dijagrami klasa (*Class Diagram*)

4.1 Problem i zahtevi korisnika

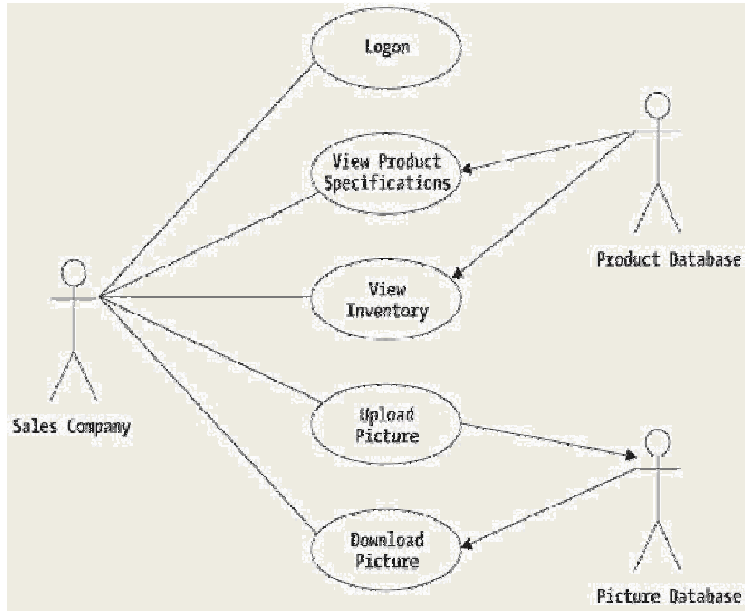
- Holding kompanija proizvodi visokokvalitetni pribor za serviranje kafe - šolje, servise, itd. Menadžment kompanije traži informatičko rešenje koje mogu da koriste njihove prodajne kompanije u različitim zemljama
- Zadatak razvoja je kreiranje rešenja (informacionog sistema/aplikacije) koje zadovoljava sledeće *zahteve*:
  - svakoj prodajnoj kompaniji omogućava brz pristup informacijama o karakteristikama i raspoloživim količinama proizvoda
  - svakoj prodajnoj kompaniji omogućava preuzimanje digitalnih fotografija za potrebe marketinga iz centralne baze podataka
  - omogućava centralizovanu kontrolu pristupa sistemu: sve prodajne kompanije vide sve informacije, ali ažuriraju samo bazu podataka fotografija; ostali korisnici (kupci) vide samo karakteristike proizvoda.

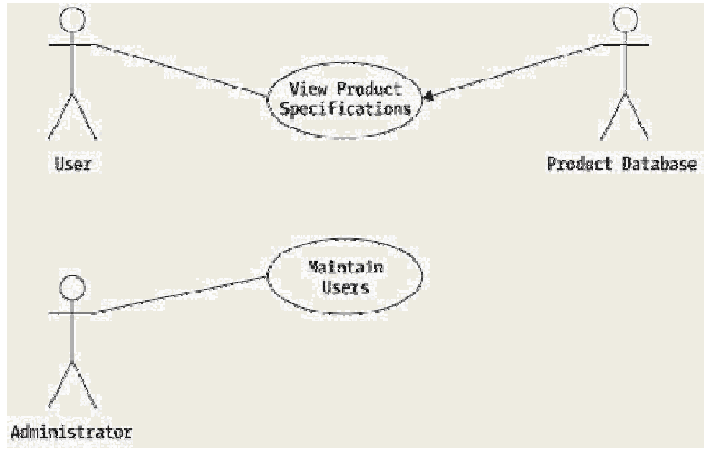
Napomena o UML

- U praksi se za opis sistema koristi samo deo mogućnosti jezika UML (oko 20%)
- ali, nije uvek istih 20%

4.2 Korisnički zahtevi (*requirements*)

- Korisnički zahtevi se analiziraju kroz model slučajeva korišćenja (*Use Case Model*)
  - učesnici
  - slučajevi korišćenja
  - relacije
- Dijagrami i njihove relacije su samo pomoćna sredstva
  - slučajevi korišćenja su tekstualni dokumenti - njihova izrada je pisanje teksta





4.3 Dijagrami sekvenci (*Sequence*)

- Diagram sekvenci slučaja *Upload Picture* (prva verzija)
- Diagram sekvenci slučaja *Upload Picture* (detaljniji)

Diagram sekvenci slučaja *Upload Picture* (prva verzija)

- Sistemski zahtevi se dalje analiziraju
  - slučaj korišćenja *Upload Picture*

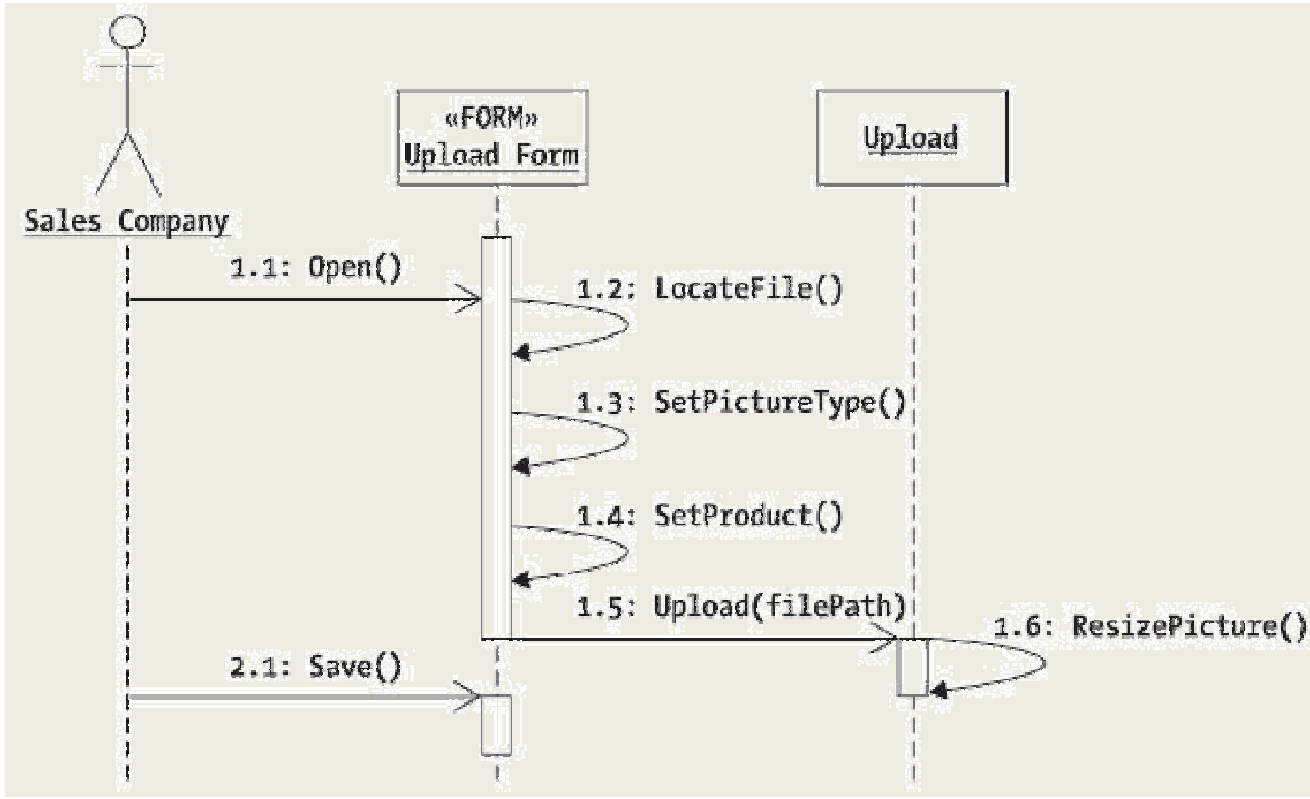
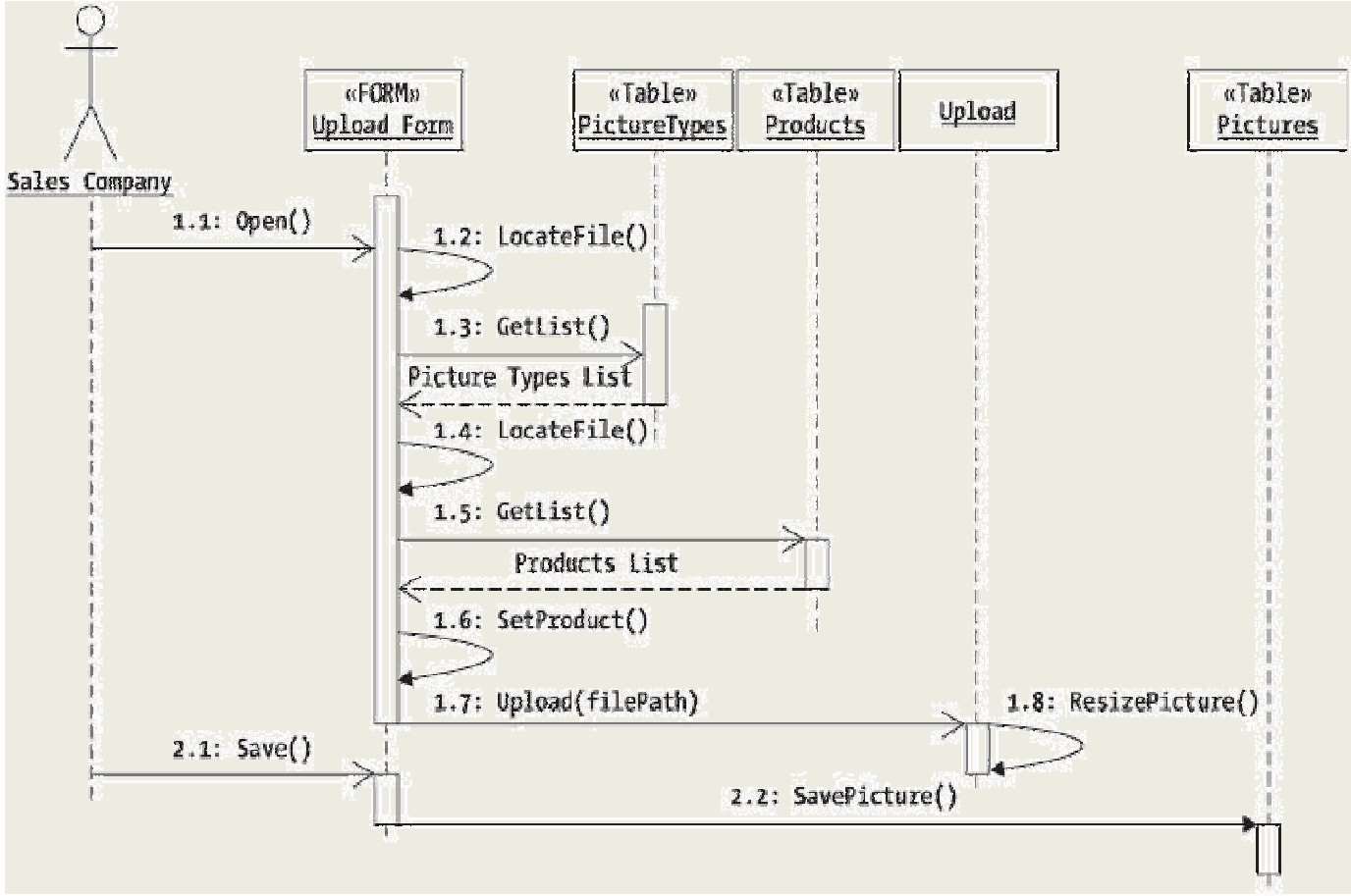


Diagram sekvenci slučaja *Upload Picture* (detaljniji)

- Detaljniji dijagram

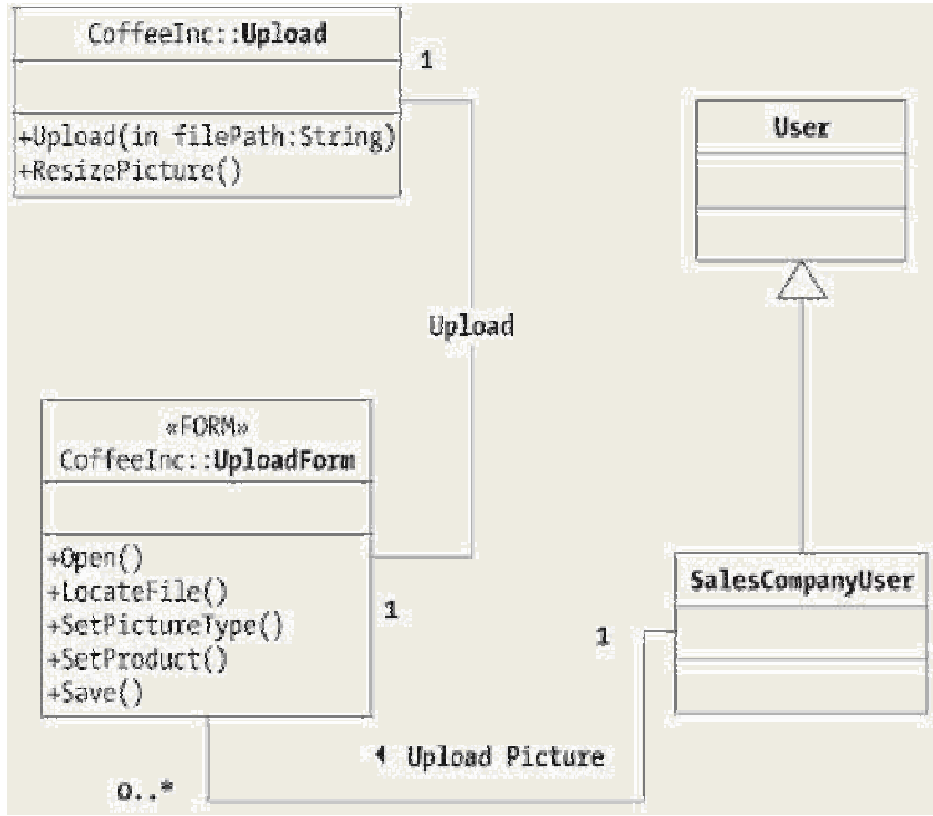


4.4 Dijagrami klasa (*Class Diagram*)

- Dijagram klasa (prva verzija)
- Dijagram klasa (druga verzija)
- Dijagram klasa (treća verzija)

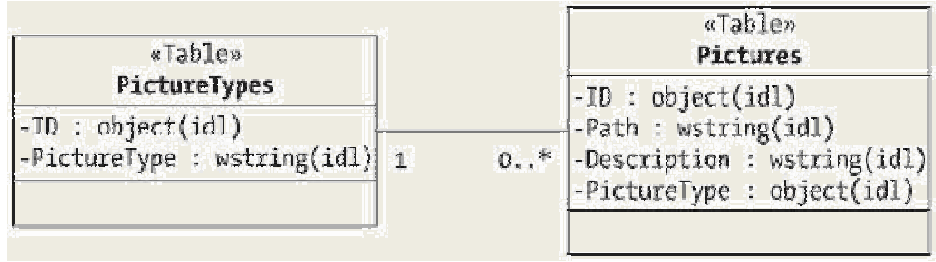
Dijagram klasa (prva verzija)

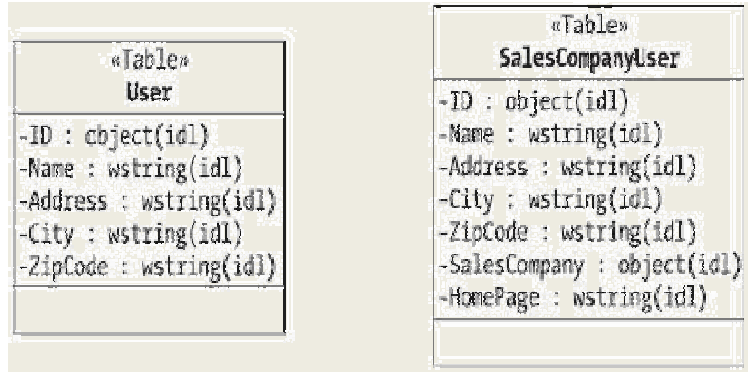
- Klase
  - Upload
  - UploadForm
  - User
  - SalesCompanyUser



Dijagram klasa za Picture Database (User)

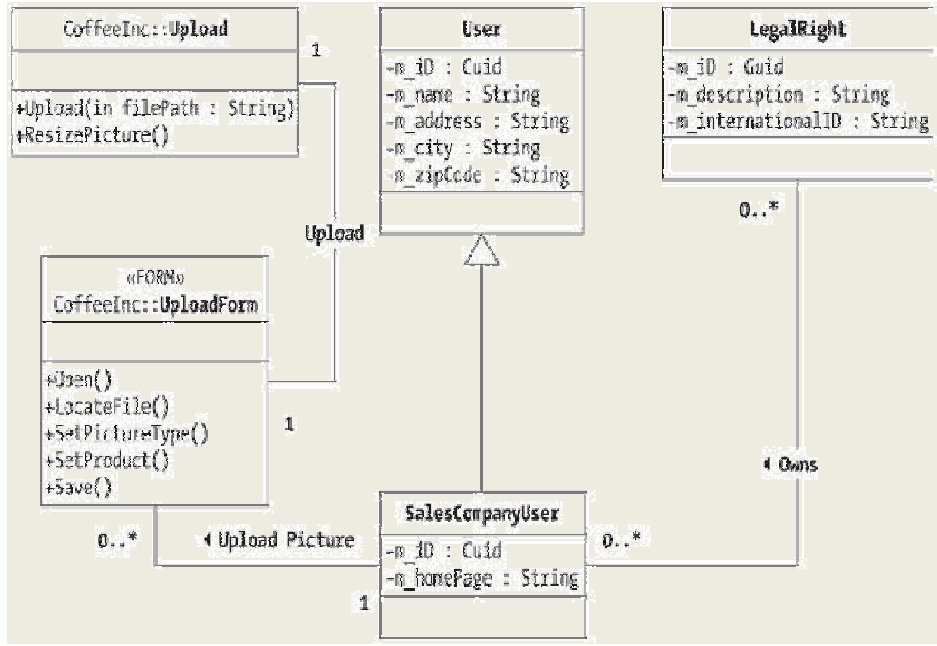
- Klase baze podataka
  - PictureTypes
  - Pictures
- Detalji klasa
  - User
  - SalesCompanyUser





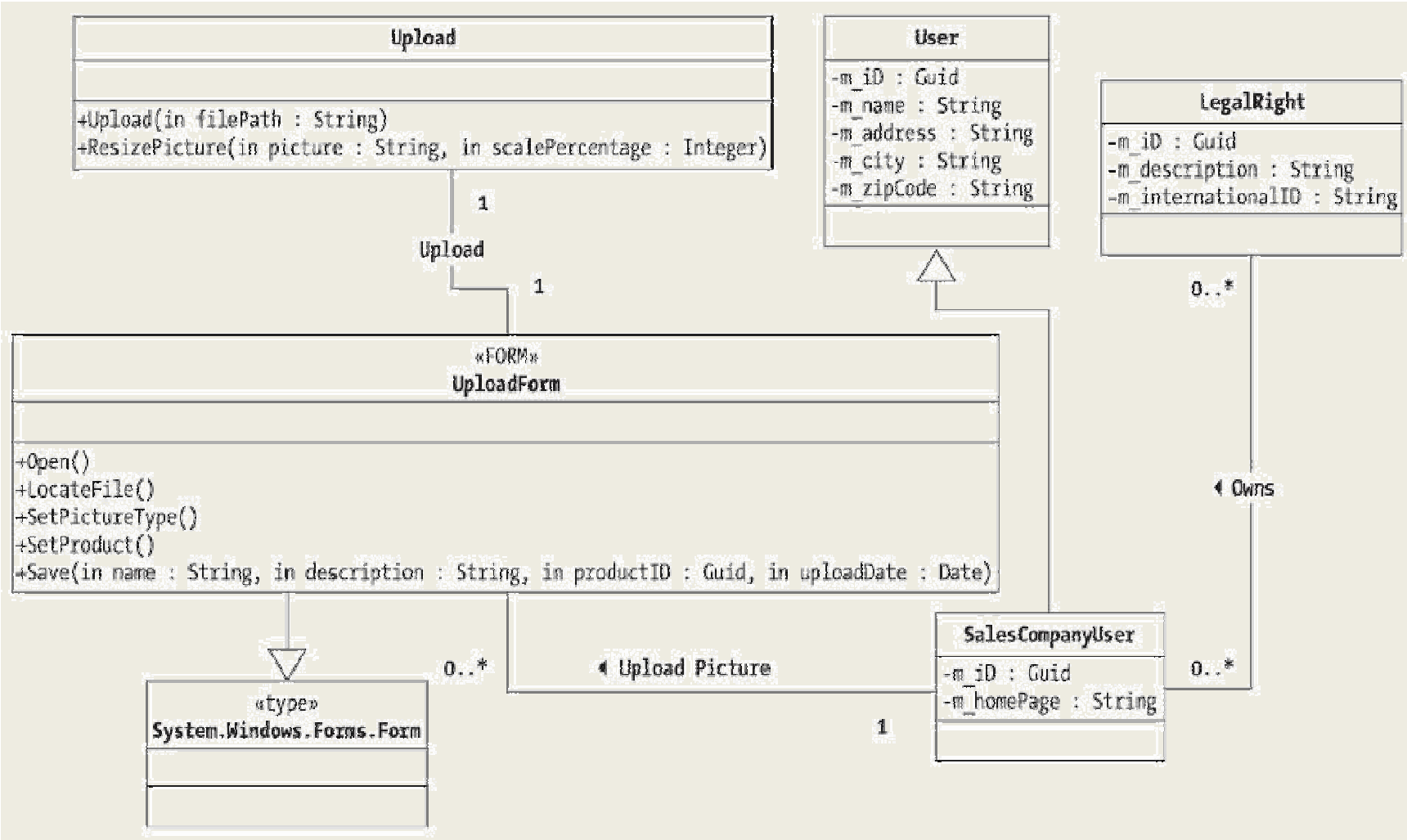
Dijagram klasa (druga verzija)

- Generalizacija klase
  - User
- Nova klasa
  - LegalRight

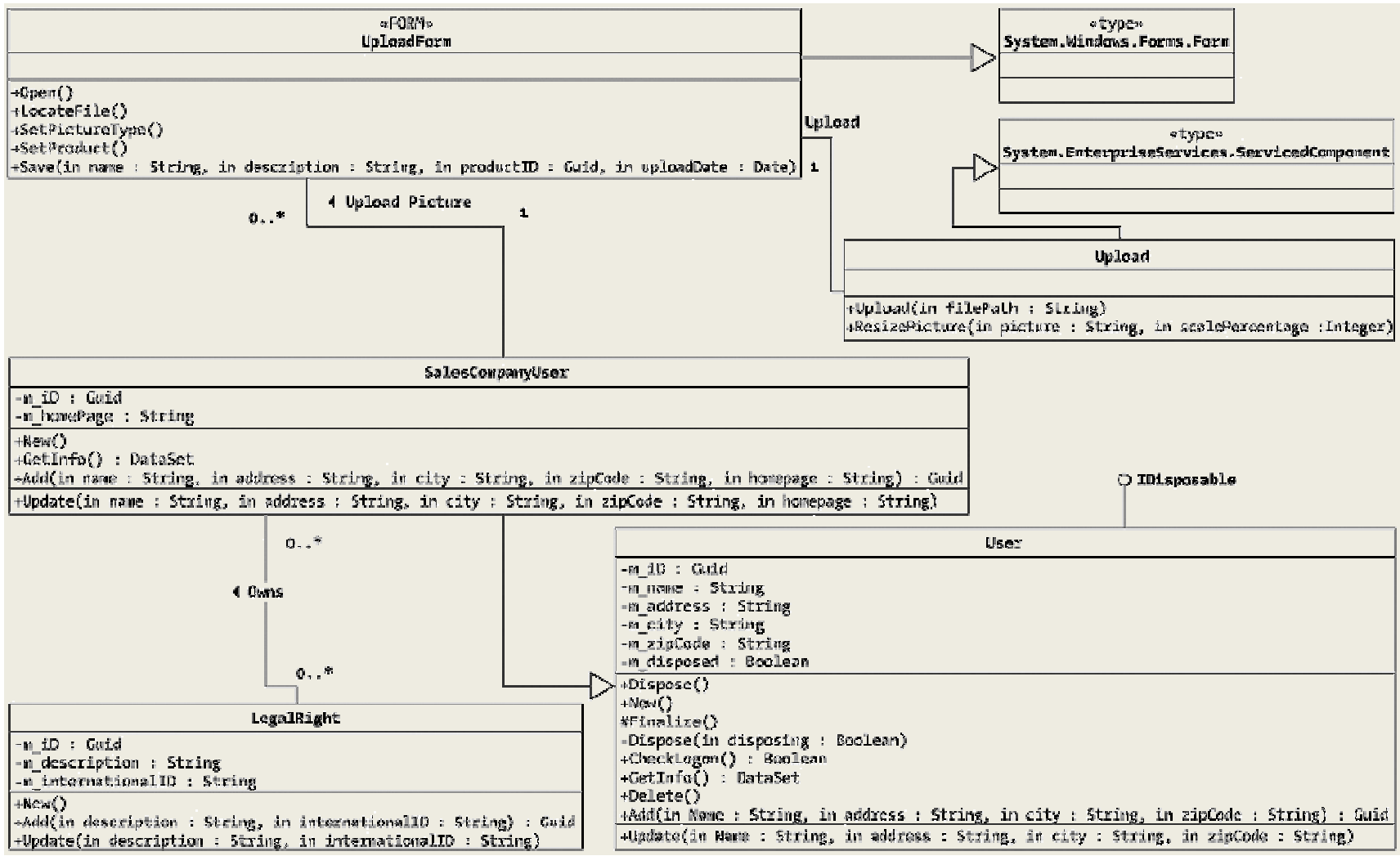


Dijagram klasa (treća verzija)

- Detalji operacija



Dijagram klasa (konačna verzija)



Literatura

1. Miškovic V., *Projektovanje informacionih sistema (predavanja)*, Univerzitet Sinergija, 2012
2. Kurbel K. E., *The Making of Information Systems: Software Engineering and Management in a Globalized World*, Springer, 2008
3. Sommerville I., *Software Engineering*, 9thEd, Addison Wesley, 2011
4. Pressman C., *Software Engineering: A Practitioner's Approach*, 5thEd, McGrow-Hill, 2001.
5. IBM Rational Product Training: *DEV475 Mastering Object-Oriented Analysis and Design with UML*
6. Kroll P., P. Kruchten, *Rational Unified Process Made Easy – A Practitioner's Guide to RUP*, John Waley, 2003
7. Priručnici za programske alate i Web reference

dr Vladislav Miškovic

Tema 10: UML specifikacija, tehnike i dijagrami

Pitanja?