Softversko inženjerstvo

Specifikacija i inženjering zahteva

dr Miloš Stojanović*

Visoka tehnička škola strukovnih studija Niš

2017.

*milos.stojanovic@vtsnis.edu.rs

Upravljanje zahtevima

- Upravljanje zahtevima znači prevođenje zahteva korisnika u skup njihovih potreba i funkcija sistema.
- Ovaj skup se kasnije pretvara u detaljnu specifikaciju funkcionalnih i nefunkcionalnih zahteva.
- Detaljna specifikacija se prevodi u test procedure, projekat i korisničku dokumentaciju.
- Potrebno je definisati proceduru u slučaju promene zahteva korisnika.

Zahtevi

- Funkcionalni
- Nefunkcionalni

Funkcionalni zahtevi

- **KO (Interfejs) koji ljudi organizacije ili sistemi smeju (ne ljudi, organizacije, ili ne smeju) da pristupe sistemu.**
- **ŠTA (Podaci) Informacije potrebne sistemu u smislu zadovoljenja funkcija.**
- **GDE (Mreža) Fizička ili Internet lokacija koja mora imati pristup rešenju.**
- **KADA (Događaji) Vremensko usaglašavanje poslovnih događaja.**
- **ZAŠTO (Businnes rules) Poslovna pravila kojima mora biti prilagođen sistem.**
- **KAKO (Proces) Koje zadatke ili procesiranje i funkcije sistem mora da izvrši.**

Nefunkcionalni zahtevi

- Ograničenja u projektu se moraju takođe definisati.
- Kategorije ograničenja:

- Nivo kvaliteta
- Nivo cene
- Funkcionalnost
- Dizajn
- Politička
- Kulturološka
- Pravna

Upravljanje zahtevima

- Svaki od funkcionalnih i nefunkcionalnih zahteva evoluira kroz različite fokuse i perspektive, definišući različite nivoe detalja u zahtevima.
- Svaka nenamerna praznina u zahtevima povećava RIZIK kraha projekta ili dodavanja drugog defekta.

Problemi u specifikaciji zahteva

- Korisnici u početku:
  - ne znaju tačno šta hoće,
  - menjaju svoja očekivanja u toku projekta,
  - uvek misle da se može više uraditi, i
  - nikada nisu zadovoljni.

Problemi u specifikaciji zahteva

- Važno je za uspeh projekta da KUPAC uoči razliku između toga ŠTA ŽELI i ŠTA MU TREBA.

"Mi imamo više nego 100%
onoga što Vama treba.
Mi nemamo 100 %
onoga što Vi želite."

Korišćenje standarda u specifikaciji zahteva

- Omogućava KUPCU da postavi zahteve na način da ga isporučioc razume.
- Omogućava ISPORUČIOCU da razume zahteve kupca i analizom zahteva bude siguran da je u mogućnosti da proizvede softverski proizvod odgovarajućeg kvaliteta.

Specifikacija zahteva

- Organizacija mora da definiše zahteve kupca koji obuhvataju:

➢ zahteve za proizvod koje je kupac definisao, uključujući i zahteve u pogledu dostupnosti, isporuke i podrške proizvodu;

➢ zahteve koje kupac nije definisao, ali koji su neophodni za planiranu ili željenu upotrebu;

➢ obaveze koje se odnose na proizvod, uključujući propise i zakonske zahteve.

Upravljanje zahtevima

- Upravljanje zahtevima je više od dokumentovanja istih.
- Šest preporuka “dobre prakse” za upravljanje zahtevima:

1. Odvojite dovoljno vremena za proces definisanja zahteva.
2. Izaberite pravi pristup za iznošenje zahteva.
3. Prenesite – saopštite zahteve svim interesnim stranama.
4. Razvrstajte zahteve po prioritetu.
5. Zahtevi za ponovno korišćenje – reuse.
6. Pratite zahteve kroz životne cikluse softvera.

Upravljanje zahtevima

- Proces upravljanja zahtevima sastoji se od međusobno nerazdvojivih podprocesa, koji se ne mogu zasebno i sekvencijalno odvijati:

➢ Prikupljanje zahteva

➢ Analiza zahteva

➢ Specifikacija zahteva

➢ Validacija zahteva

Značaj specifikacije zahteva

- Kvalitetan softver je proizvod vrlo dobro realizovanog dizajna zasnovanog na tačnim zahtevima, koji su rezultat efektivne komunikacije i saradnje – **partnerstva** između projektanata i kupaca.

Prikupljanje zahteva

- Tri glavne kategorije učesnika su:

➢ **Kupac**

➢ **Korisnik**

➢ **Developer**

i NIKO od učesnika u ovoj fazi nema kompletnu sliku o softverskom proizvodu.

- Glavni problem: neadekvatna komunikacija

Prikupljanje zahteva

- Tokom procesa prikupljanja zahteva mora se uspostaviti partnerski odnos među interesnim stranama. **Učešće kupca je najkritičniji faktor** od kojeg zavisi kvalitet softvera, i najteži problem je deljenje vizije o proizvodu sa kupcem.
- Ako interesne grupe ne komuniciraju efektivno, svaka od njih tražiće način da iskaže nadmoć i uticaj nad ostalima, što jeste cilj u politici, ali ne i u upravljanju zahtevima za proizvod.

Lista prava kupca SW-a

1. Očekuje da analitičar govori njegovim jezikom,
2. Očekuje da analitičar razume njegovo poslovanje i ciljeve, i u tom svetlu mesto i ulogu novog softverskog proizvoda,
3. Očekuje da analitičar informacije koje mu prezentira evidentira, razradi i izradi specifikaciju zahteva za sistem,
4. Da projektanti objasne zahteve za softver na nivou podsistema i nižim nivoima,

Lista prava kupca SW-a

5. Očekuje da ga projektanti tretiraju sa respektom i da održavaju klimu saradnje i profesionalnosti,

6. Analitičar prezentira ideje i alternative i za njegove zahteve i za implementaciju,

7. Da se opišu karakteristike koje će obezbediti lako korišćenje proizvoda i da se opišu scenariji upotrebe proizvoda,

Lista prava kupca SW-a

8. Budu prikazane mogućnosti da se već postojeće softverske komponente uključe ili prilagode njegovim zahtevima,

9. Mu se ukaže poverenje na njegovu procenu cene i uticaja na softver kada zahteva promenu zahteva za softver,

10. Primi sistem koji zadovoljava zahteve za funkcionalnost i kvalitet, do nivoa koji je utvrđen u komunikaciji sa projektantima, usaglašen i dokumentovan u specifikaciji i dopunama specifikacije.

Lista odgovornosti kupca SW-a

1. Upozna analitičara sa svojim poslovanjem i definiše žargon.
2. Potroši vreme da obezbedi zahteve i pojasni ih u svetlu svojih poslovnih potreba.
3. Bude precizan u vezi sa zahtevima za sistem.
4. Blagovremeno donese odluke u vezi zahteva, kada se to zahteva od njega.
5. Poštuje projektantske ocene u vezi cene i izvodljivosti zahteva.

Lista odgovornosti kupca SW-a

6. Definiše prioritete za svaki pojedinačni zahtev, osobinu sistema i scenario primene.
7. Izvrši preispitivanje dokumenata (zahteva) i prototipova.
8. Odmah dostavi promene u zahtevima za proizvod.
9. Postupa saglasno dokumentovanom procesu promena zahteva razvojne organizacije.
10. Poštuje proces upravljanja zahtevima koji koristi razvojna organizacija.

Kategorije zahteva

- Funkcije: “šta” sistem mora da bude sposoban da uradi.
- Osobine: “koliko dobro” će funkcije biti izvršavane.
- Cena: koliko će koštati (bilo koji ulazni resurs: novac, ljudi, ili vreme) kreiranje i održavanje funkcija i njihovih osobina.
- Ograničenja: bilo koja restrikcija u slobodi definisanja zahteva ili dizajnu.

Karakteristike zahteva

1. Svi zahtevi su testabilni na prisustvo u realnom svetu njihove implementacije.

2. Svi zahtevi reflektuju nečije subjektivne vrednosti i prioritete.

3. Za bilo koji set zahteva, postoji neki drugi set zahteva koji verovatno isto tako dobro ili bolje zadovoljava realne potrebe, vrednosti i prioritete kupca.

Karakteristike zahteva

- 4. Svi zahtevi su u prirodnom konfliktu sa svim ostalim u odnosu na resurse.
- 5. Zahtevi nisu statični, nego se stalno menjaju kako se menja svet menjajući naše potrebe, vrednosti i prioritete.
- 6. Upravljanje zahtevima je sistematičan proces utvrđivanja kompletnog seta relevantnih vrednosti kojima raspolažu interesne strane, i procesiranja istih sve dok se od njih ne dobije zadovoljavajući nivo “isporuke zahtevanih krajnjih stanja”. To implicira da se mora uključiti dizajn, testiranje, kontrola kvaliteta, upravljanje projektom, specifikacija jezika i sve ostale relevantne discipline da se omogući da projekat uspe.

Šta treba ubaciti na kraju specifikacije...

SPECIFIKACIJA (tekst ispred potpisa)

“Slažem se da ovaj dokument reprezentuje naše najbolje shvatanje zahteva za softver za ovaj projekat. Naredne izmene osnovne konfiguracije mogu se izvršiti kroz definisan proces upravljanja promenama. Prihvatam da usaglašene izmene mogu zahtevati da pregovaramo o ceni, resursima i rokovima realizacije projekta.”

Značaj specifikacije zahteva...

- NIKO ne želida dobije proizvod koji ne može da koristi.
- Vaš KUPAC nije izuzetak.
- Čak i mala greška u zahtevima može dovesti do velikog problema….

Razlozi neuspeha softverskih projekata

1.	Nekompletni zahtevi	13,1%
2.	Nedostatak učešća korisnika	12,4%
3.	Nedostatak resursa	10,6%
4.	Nerealna očekvanja	9,9%
5.	Nedostatak podrške rukovodstva	9,3%
6.	Promenjivi zahtevi	8,7%
7.	Nedostatak planiranja	8,1%
8.	Nije više potreban	7,5%
9.	Nedostatak IT menadžmenta	6,2%
10.	Tehnološka nepismenost	4,3%
	Ostali	9,9%

Metode inženjeringa zahteva

- Procesi inženjeringa zahteva su obično “vođeni” odgovarajućom metodom.
- Metode inženjeringa zahteva definišu sistematske načine za dobijanje modela sistema.

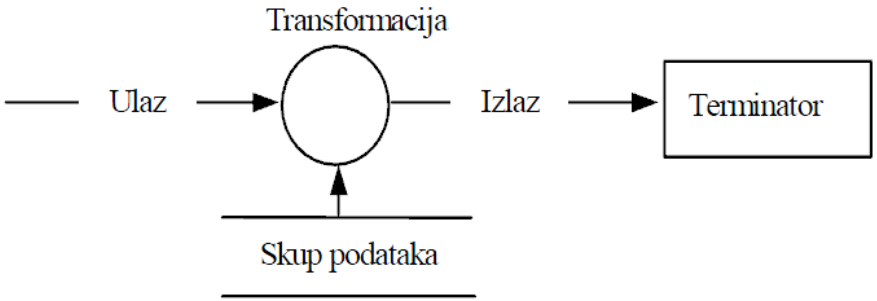
Osnovne karakteristike metoda inženjeringa zahteva

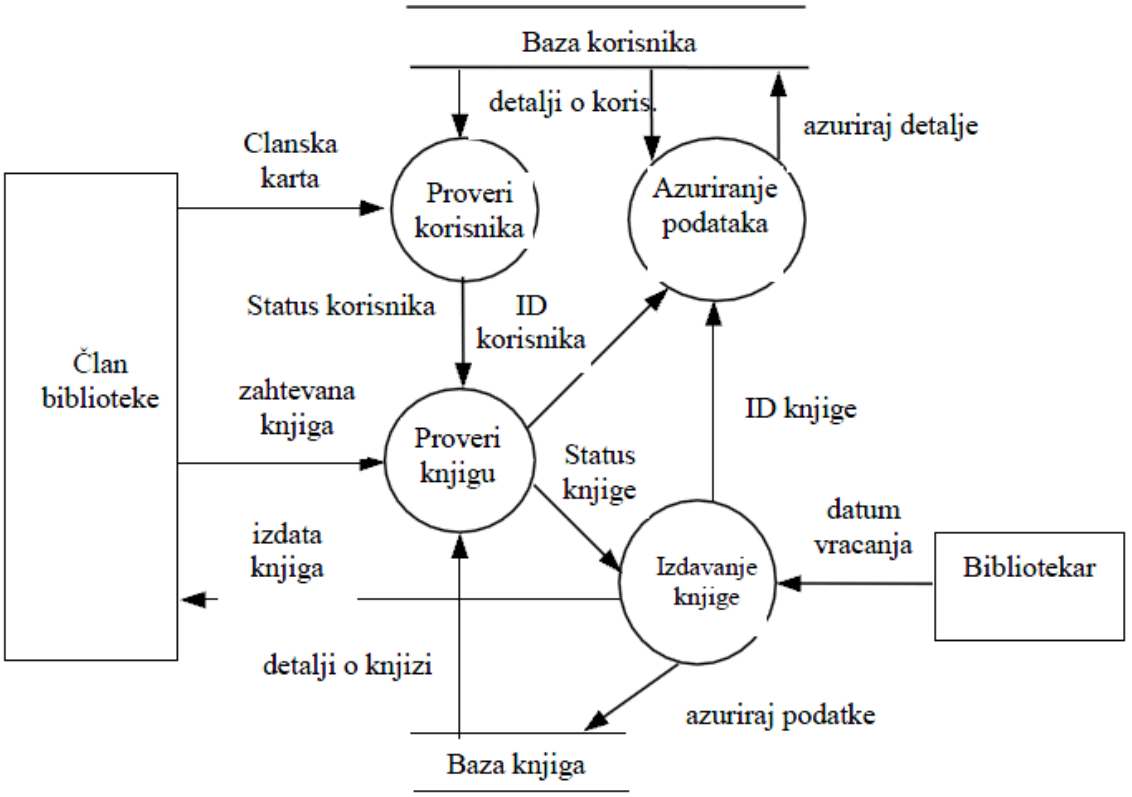
- Pogodnost za usaglašavanje sa krajnjim korisnikom.
- Preciznost definicije i odgovarajuće notacije.
- Pomoć kod formulisanja zahteva.
- Fleksibilnost.
- Podržanost odgovarajućim SW alatima.

Najpoznatije metode

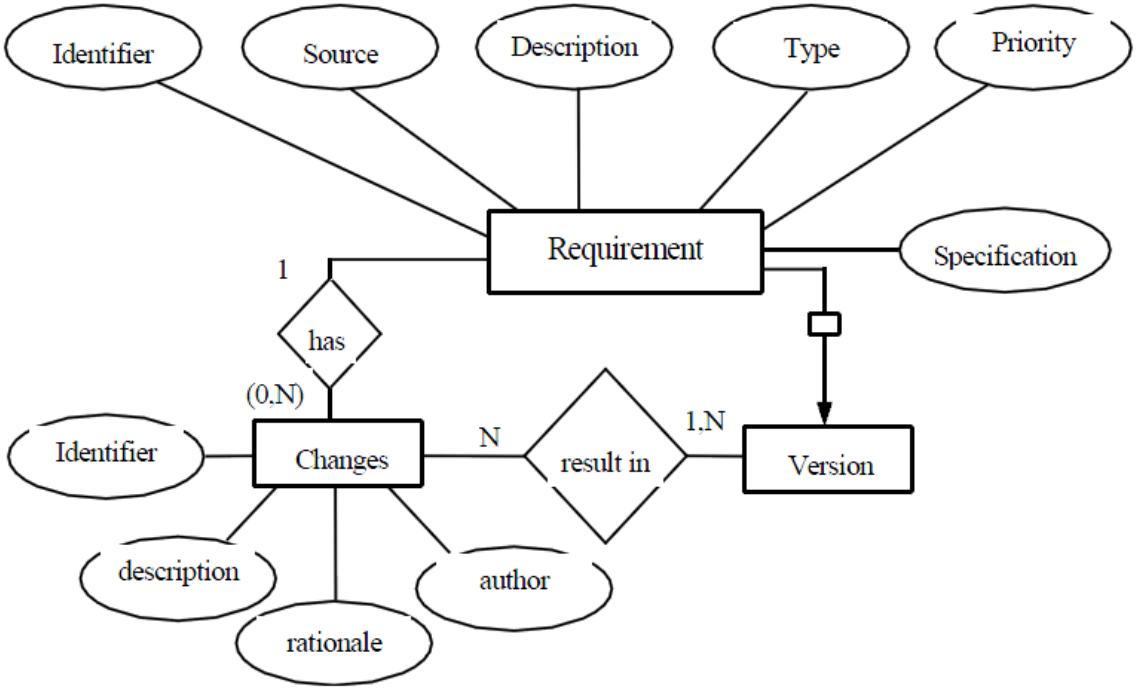
- DFD (Data-Flow-Diagram) dijagrami
- Metode koja koriste relacioni model podataka
- Objektno-orijentisane metode
- Formalne metode
- Metode zasnovane na ponašanju sistema:
  - **Use-case** specifikacija i scenarija ponašanja
  - Viewpoint orijentisane metode

DFD dijagram



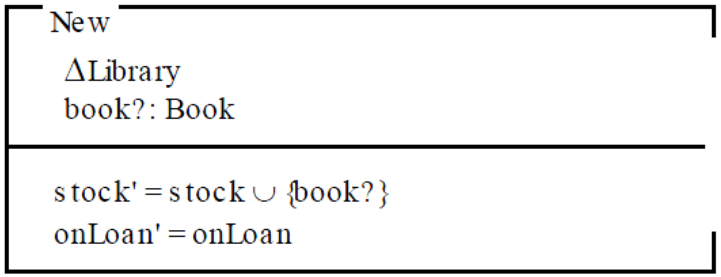


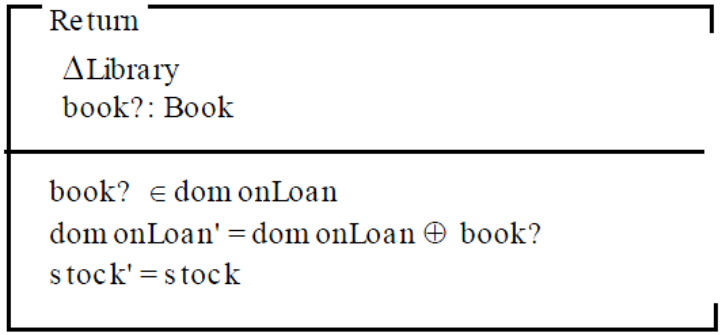
Metode koje koriste relacioni model



Formalne metode

- Bazirane su na matematički formalnoj sintaksi i semantici.
- Primer Z metode:





Objektno-orijentisane metode

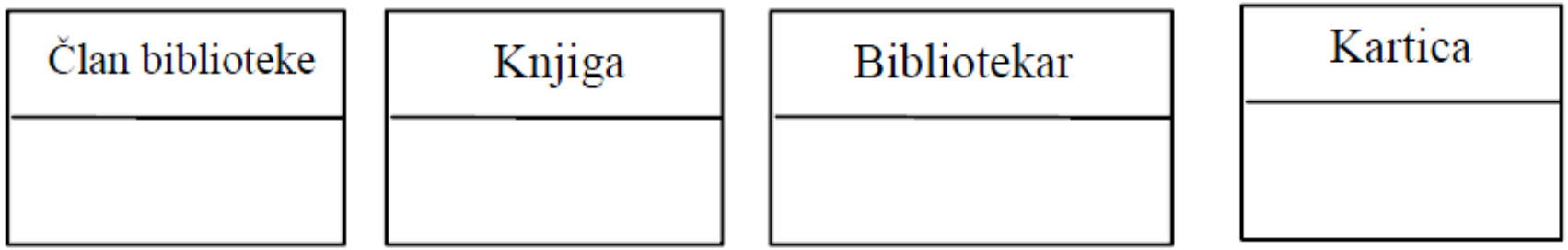
- U procesu prikupljanja zahteva, osnovni elementi koji se posmatraju su objekti.
- Objekti se dobijaju analizom domena problema.
- Objekti uključuju:

➢ Udeđaje sa kojima sistem interreaguje
➢ Sisteme koji sarađuju sa sistemom koji se razvija
➢ Organizacione jedinice
➢ Stvari o kojima se moraju čuvati podaci
➢ Fizičke lokacije
➢ Specifične uloge ljudi

Koraci kod OO metoda

- Identifikacija osnovnih objekata (klasa)
- Definisanje objektne strukture i veza između klasa
- Definisanje atributa i metoda objekata
- Definisanje poruka koje objekti razmenjuju

Korak 1 – Identifikacija osnovnih klasa



Korak 2 – Veze između klasa

- Mogu se identifikovati veze između sledećih zahteva:

➢ Član biblioteke **iznajmljuje** knjigu

➢ Knjiga **je** publikovana od strane izdavača

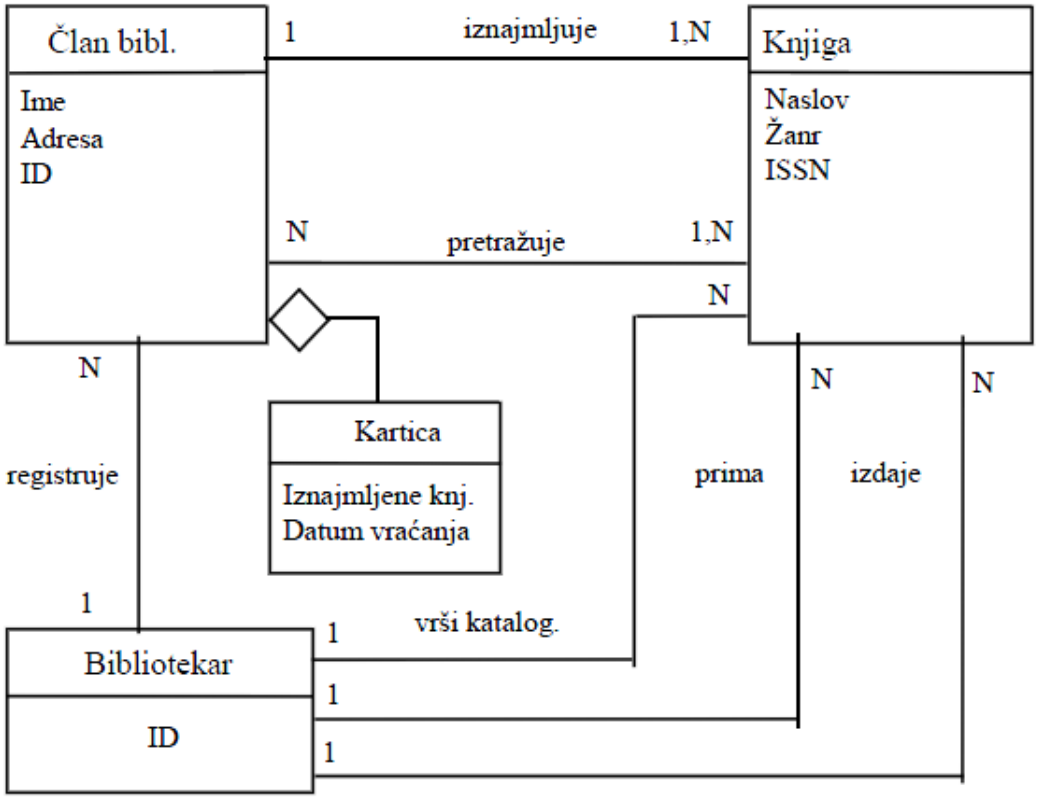
➢ Administrator sistema **registruje** člana

➢ Članovi **su** studenti, osoblje i spoljni korisnici

➢ Administrator sistema **vrši katalogizaciju** knjiga

➢ Bibliotekar **izdaje** knjige

Korak 2 – Veze između klasa



Korak 3 – Atributi objekata

- Atributi se dobijaju analizom zahteva
- Na pimer, ako je zahtev da svi članovi moraju biti registrovani pre iznajmljivanja knjige:

➢ to znači da se mora čuvati datum registracije člana

➢ svaki član mora imati karticu kako bi se beležila sva izdavanja i vraćanja knjige

- Knjiga mora da ima atribute: Naslov, žanr i ISSN
- Član biblioteke mora da ima atribute: Ime i prezime, adresu i ID

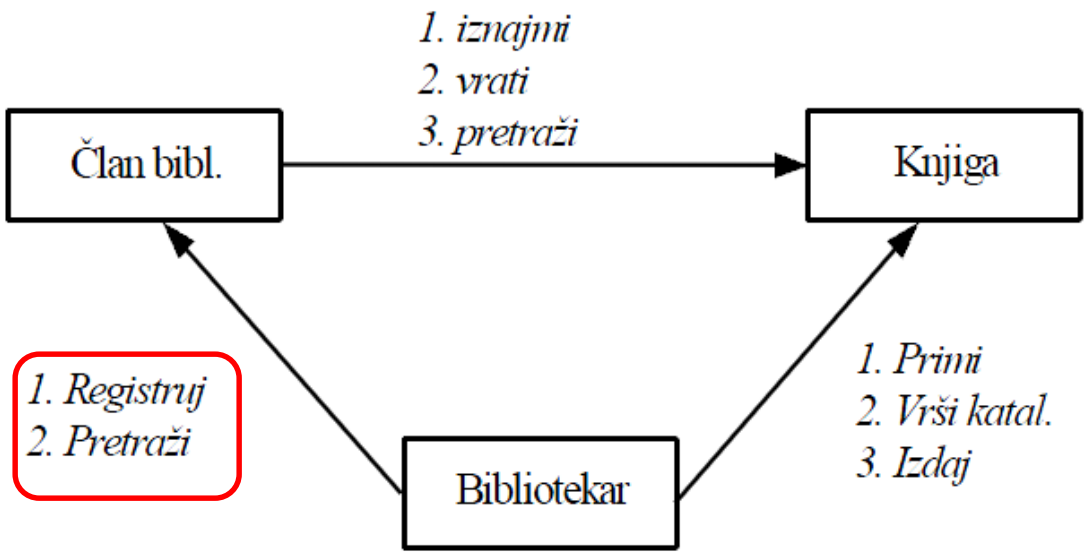
Korak 4 – Metode objekata

- U ovom koraku se vrši identifikacija metoda
- Svakako moraju da postoje metode za pristup i izmenu atributa:

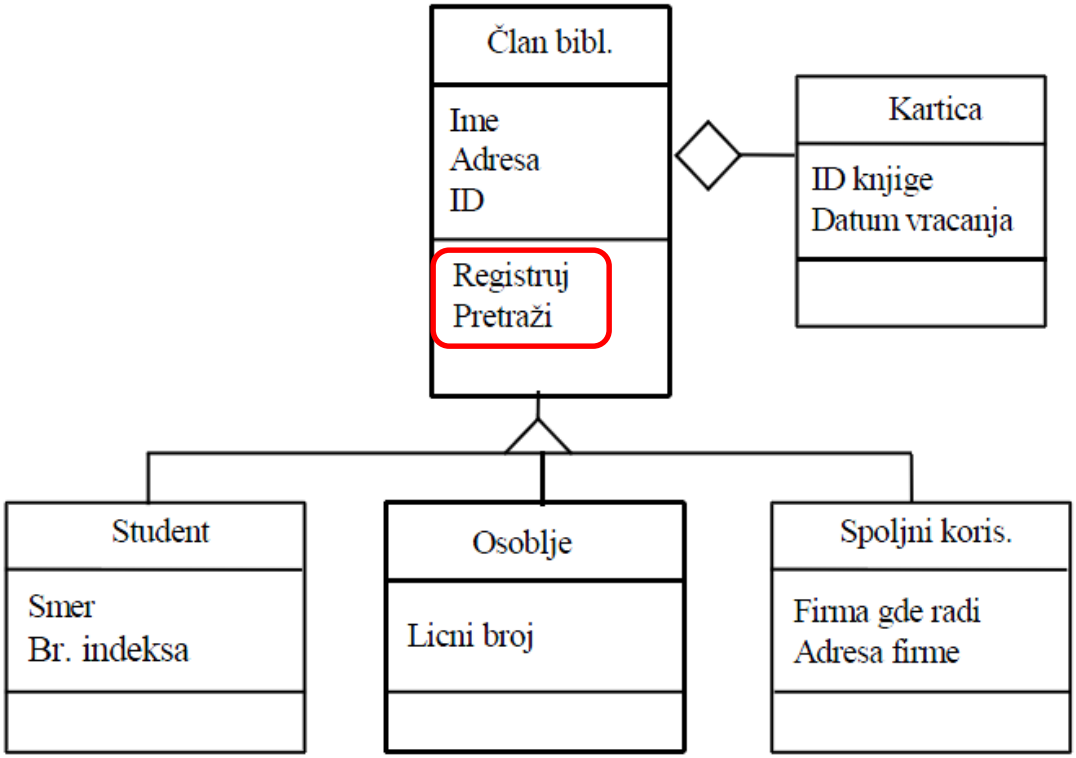
➢ Get-Set metode

- Jedan način identifikovanja metoda je praćenje poruka koje se razmenjuju između objekata.

Korak 4 – Razmena poruka



Korak 4 - Metode



Use-case metoda

- Sistem se posmatra sa stanovišta korisnika sistema.
- Opisuju se slučajevi korišćenja sistema i scenarija ponašanja.
- Koristi se UML notacija.
- Koristi se kod RUP modela razvoja SW-a.

Use-case i scenarija događaja

- Metode objekata mogu biti otkrivene i modeliranjem scenarija događaja za različite funkcije sistema.
- Događaji se prate do objekata koji reaguju na njih.
- Tipičan model scenarija događaja je interakcija između korisnika i sistema.

Scenarija

- Scenarija su primeri kako će sistem biti korišćen u realnom životu.
- Oni bi trebalo da uključe:

➢ Opis početne situacije;

➢ Opis normalnog toka događaja;

➢ Opis izuzetaka (ako se izađe iz normalnog toka);

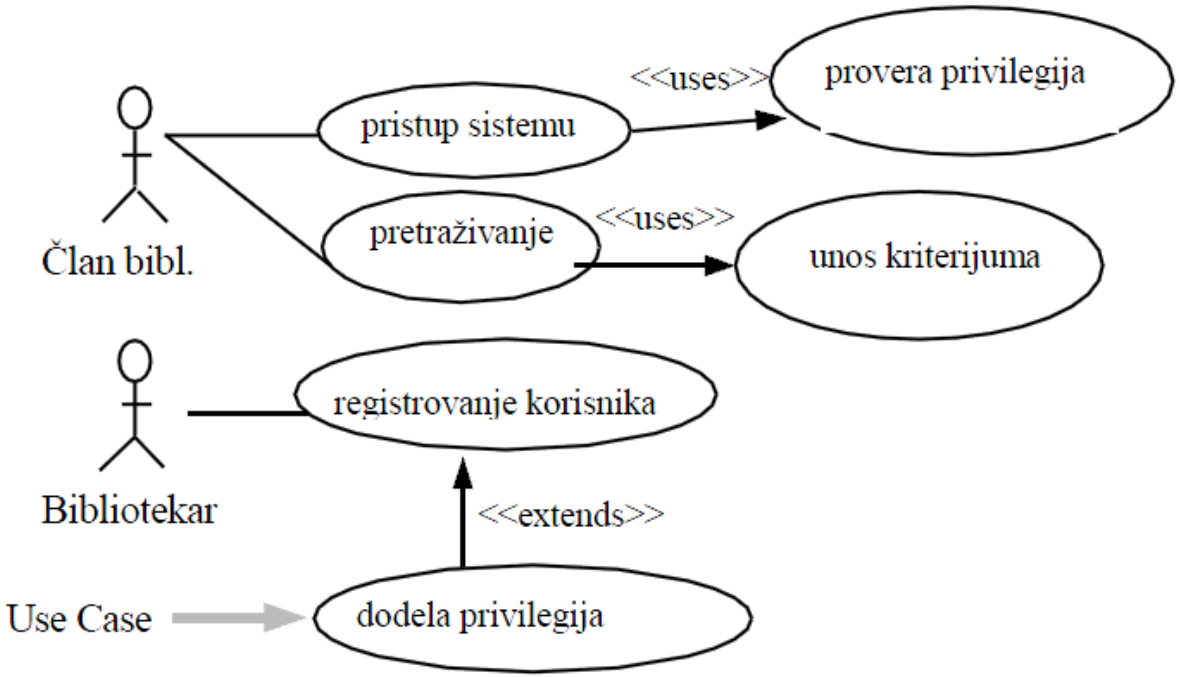
➢ Informacije o konkurentnim aktivnostima;

➢ Opis stanja gde se scenario završava.

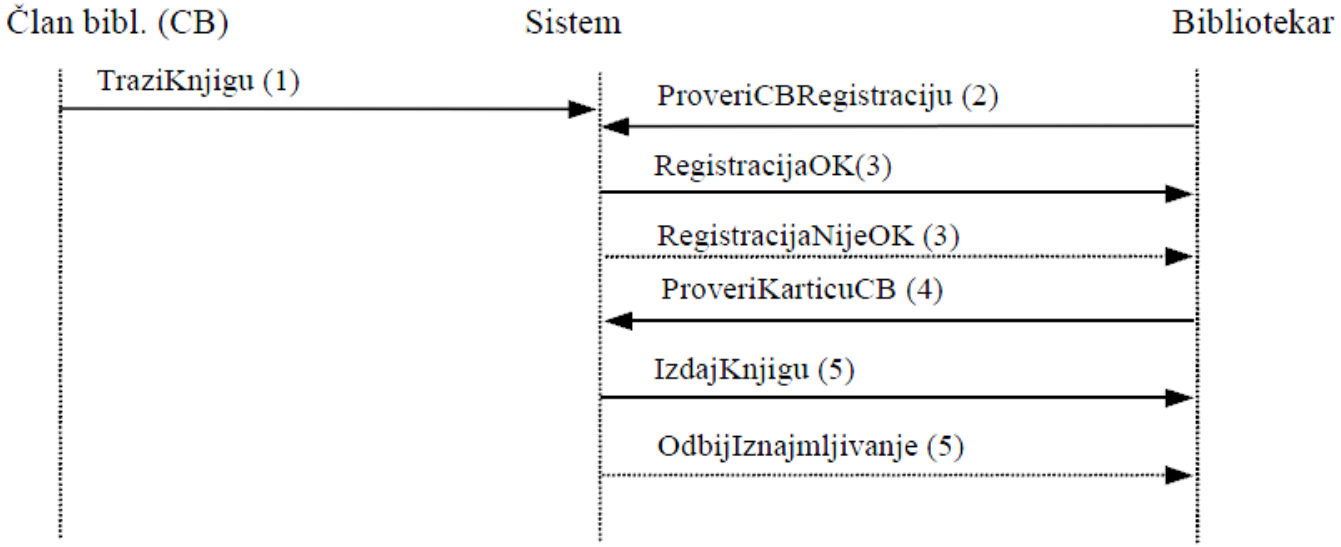
Slučajevi korišćenja (Use cases)

- Slučajevi korišćenja predstavljaju tehniku baziranu na scenarijima i zapisanu pomoću UML dijagrama koja identifikuje aktere u sistemu i slučajeve korišćenja sistema.
- Potpuni skup slučajeva korišćenja opisuje sve moguće interakcije korisnika sa sistemom.
- UML dijagrami sekvenci mogu biti korišćeni za detaljni opis slučajeva korišćenja. Oni daju sliku o obradi događaja u sistemu za izabrani slučaj korišćenja.

Use case


Član bibl.
pristup sistemu
<<uses>>
provera privilegija
pretraživanje
<<uses>>
unos kriterijuma
Bibliotekar
registrovanje korisnika
<<extends>>
Use Case
dodela privilegija

Scenario


Član bibl. (CB)
Sistem
Bibliotekar
TraziKnjigu (1)
ProveriCBRegistraciju (2)
RegistracijaOK(3)
RegistracijaNijeOK (3)
ProveriKarticuCB (4)
IzdajKnjigu (5)
OdbijIznajmljivanje (5)

RUP - Use-case specifikacija

- Use-case specifikacija je izlaz iz faze razrade (elaboracije) i sadrži:

➢ Opis slučajeva korišćenja

➢ Definisanje aktera u sistemu

➢ Određivanje arhitekturno najznačajnijih slučajeva korišćenja

SW alati za Use-case metode

- IBM Rational Rose
- Microsoft Visio
- …