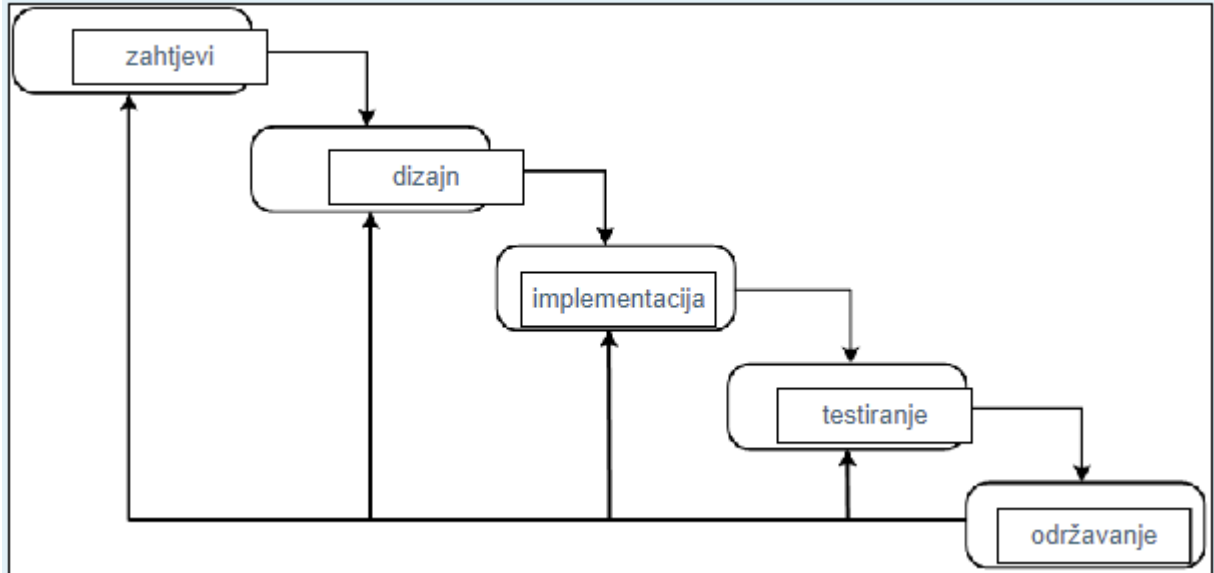


1. DOMACI

1. Prevlačenjem odgovarajućih tekstualnih blokova, popunite model vodopada prikazan na slici:



2. Domen softvera je dok je realni svijet .
3. U ranim fazama razvoja softvera preporučljivo je da se koriste dijagrami a u kasnijim fazama razvoja se koriste dijagrami.
4. Ispod je dat pojmovnik alata koji olakšavaju komunikaciju između developera i ostalih aktera u razvoju softvera. Povežite pojmove.

Razbijanje sistema na djelove

Odgovor 1

Opisuje šablone i najbolje prakse u razvoju softvera

Odgovor 2

Simbolički jezik

Odgovor 3

Omogućava planiranje, praćenje i mjerenje progressa i kvaliteta razvijenog softvera

Odgovor 4

5. Povežite faze razvoja softvera i njihova odgovarajuća objašnjenja:

Provjera funkcionalnosti pojedinačnih klasa i cijelog sistema

Odgovor 1

testiranje

Razumijevanje scenarija u kojima se koristi softver. Isporuka statičkog modela

Odgovor 2

Specifikacija zahtjeva

Dodjela odgovornosti (uloga) objektima i specifikiranje detaljne dinamike njihove interakcije u različitim scenarijima

Odgovor 3

Dizajn

Funkcionisanje sistema, ispravka grešaka i dodavanje novih sadržaja

Odgovor 4

Isporuka, održavanje i evolucija

Kodiranje dizajniranog softvera u programskom jeziku

Odgovor 5

Implementacija

6. Povežite metode razvoja softvera i njihova objašnjenja.

Kontinuirane povratne informacije od korisniku ključne, petlje sa povratnim informacijama su višestruke i na više nivoa

Odgovor 1

Agilni pristup

Jednosmjernan, na sljedeći korak se prelazi tek kad se postojeći završi

Odgovor 2

Metod vodopada

Razvoj dijela funkcionalnosti, ponavljanje razvoja u petlji sa povratnim informacijama

Odgovor 3

Iterativni+ Inkrementalni pristup

7. Povežite termine koji se tiču razumijevanja modela problema

Scenariji korišćenja sistema (ne razmatra se šta se dešava u sistemu)

Odgovor 1

Use Cases - slučajevi korišćenja

Agenti mimo sistema, koji interaguju sa sistemom

Odgovor 2

Akteri

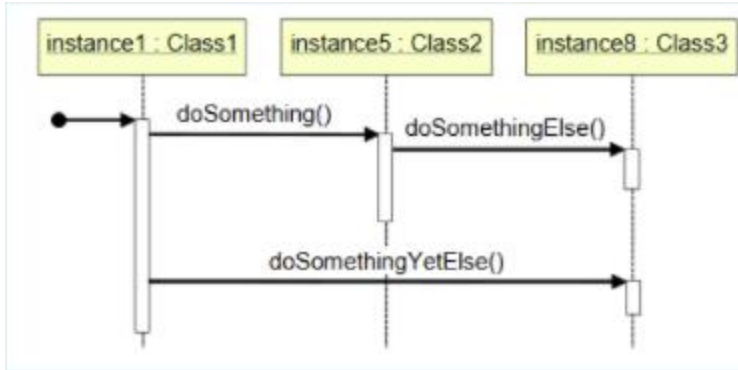
Agenti koji rade unutar sistema i omogućavaju njegovo funkcionisanje

Odgovor 3

Koncepti/ objekti

8. Označite prihvatljive definicije softverskog inženjerstva (SI, SE). Izaberite jedan ili više odgovora:
- **Softversko inženjerstvo se, između ostalog, bavi konceptima dizajna, testiranja i održavanja softvera**
 - **"SI je primjena sistematskih, disciplinovanih, mjerljivih pristupa razvoju, rukovanju i održavanju softvera; drugim riječima, primjeni inženjerstva na softver"**

9. Šta je prikazano na slici?



- **UML dijagram interakcije**

10. Odaberite tačne konstatacije.

- **Softverska aplikacija se u OO dizajnu posmatra kao skup objekata koji međusobno interaguju**
- **Ključna aktivnost dizajna u razvoju softvera je dodjela odgovornosti softverskim objektima**

11. Objektno-orijentisano programiranje podrazumijeva da se programi grupišu oko:

- **Dobro definisanih načina pristupa podacima**
- **Podataka**

12. Kada kažemo da je softver kompleksan, tada mislimo:

- **Softver je sastavljen od velikog broja međusobno povezanih dijelova**

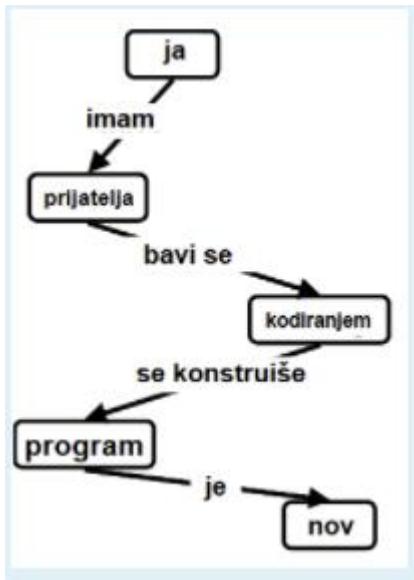
13. Zaokružite tačne konstatacije vezane za Use Cases - slučajeve korišćenja.

- **Definišu koje informacije moraju preći granice Sistema**
- **Ne razmatraju šta se dešava u sistemu**
- **Predstavljaju scenarije korišćenja Sistema**

14. Metod razvoja softvera kod kojeg su ključne povratne informacije od korisnika, i kod kojeg se pojavljuju višestruke petlje na više nivoa se naziva:

- **Odgovor: Agilni pristup**

15. Kako se naziva alat (metod za opisivanje domena problema) sa donje slike?



- Odgovor: **Mapa koncepata**

16. Kako se naziva metod softverskog inženjerstva kod kojeg se proces razvoja vrši jednosmjerno, odnosno, na sljedeći korak se prelazi tek kada se prethodni korak završi?

- Odgovor: **Vodopad**

17. Od softverskog inženjera se ne očekuje da nauči domen problema.

- **Netačno.**

18. Programer i softverski inženjer može biti ista osoba.

- **Tačno.**

19. Softversko inženjerstvo se bavi isključivo implementacijom softvera (programiranjem).

- **Netačno.**

20. Dopunite kod ispod, tako da Python program štampa 2.3 puta veći broj od onog broja koji je unio korisnik.

```
f = eval(Odgovor) ('Unesite broj: '))
print('Novi broj je:', Odgovor)
```

2. DOMACI

1. Kada u OO modelu specificiramo interfejs, mi smo samo zainteresovani oko toga ali ne i .
2. U OO modelu su dio interfejsa.
3. U OO pristupu, su labavija grupisanja potprograma i podataka, dok enkapsuliraju podatke, podaci su sada , individualne karakteristike, osobine.
4. U OO terminologiji, objekti komuniciraju jedni s drugima .
5. Povezati objašnjenja sa odgovarajućim modifikatorom pristupa atributa i/ili metoda objekta.
Ako je metod ili atribut definisan ovim modifikatorom pristupa, samo taj specifičan objekat može pristupiti tom metodu ili atributu.
Ako je metod ili atribut definisan ovim modifikatorom pristupa, samo vezani, izvedeni ili objekti djeca mogu pristupiti tom metodu ili atributu
Ako je metod ili atribut definisan ovim modifikatorom pristupa, drugi objekti mu mogu direktno pristupiti.

Odgovor 1

Odgovor 2

Odgovor 3
6. Povežite termine i odgovarajuća značenja
Proces kojim se klase dijele na javni intefejs i privatnu implementaciju
Korišćenje varijabli koje su reference na druge objekte
OO koncept koji podrazumijeva da se isti metod različito ponaša za različite potklase date klase
Skup metoda zajedno sa egzaktnim formatom za poziv svakog metoda
Metod koji se poziva pri kreiranju objekata neke klase

Odgovor 1

Odgovor 2

Odgovor 3

Odgovor 4

Odgovor 5

Prenošenje zajedničkih osobina kroz proširenja klasa od superklasa ka izvedenim klasama

Odgovor 6

nasljeđivanje

7. Povezati odgovarajuće termine i vezane konstatacije vezane za OO metod u softverskom inženjerstvu.

Fokusirano na organizaciju

objektno-orijentisano

Baziran na rezultatima, iterativno se vrše izmjene, kupci se aktivno angažuju u cilju feedback-a, zasnovan na rezultatu a ne na procesima.

Aqilni razvoj

Bilo koji međuproizvod u razvoju softvera, dakle, bilo šta što je napravljeno kako bi softver mogao biti razvijen (UML dijagram, sistemski zahtjev, softverski kod)

artifact

Testiranjem vođen razvoj

TDD

Inicijalna verzija softvera sa osnovnim funkcionalnostima

prototip

Rješenje reprezentovano sekvencom koraka

procesno orijentisano

8. Kako se nazivaju stavke podataka objekata sa odgovarajućim identifikatorima (varijablama), koje sadrže informacije specifične za posmatrani objekat?

- **Atributi**

9. Kako se naziva jedinica računarskog programa (softvera) sastavljena od enkapsuliranih podataka i programskog koda (metoda), koji mogu međusobno interagovati u cilju pružanja ili dobijanja servisa?

- **Objekat**

10. Kod OO modela, podaci (vrijednosti atributa) unutar objekata treba da budu dostupni isključivo uz korišćenje:

- **akcesora**
- **setera**
- **getera**
- **mutatora**

11. Koja je relacija između klasa sa donjeg dijagrama?

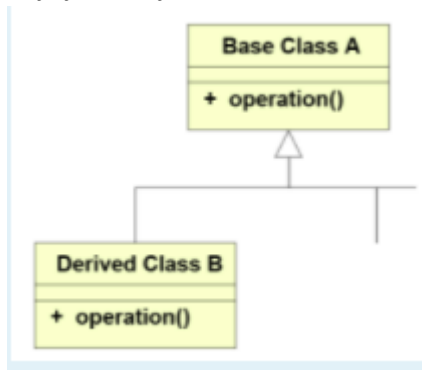


- **Kompozicija**

12. Čime su definisani prihvatljivi pozivi u slučaju objekata, odnosno, pomoću čega je moguće pristupiti servisima koje nude određeni objekti u objektnom modelu?

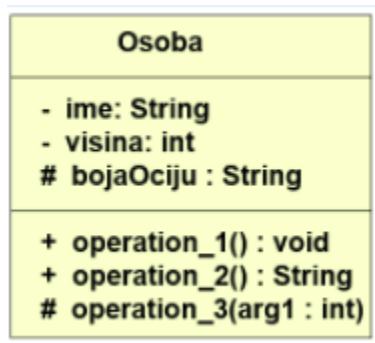
- **Metodama**

13. Koja je relacija između klasa sa donjeg dijagrama?



- **Nasljeđivanje**

14. Šta je predstavljeno na donjoj slici?



- **UML klasa**

15. OO koncept koji podrazumijeva da se krije stanje objekata, odnosno, koncept koji podrazumijeva da se stanje objekata može mijenjati jedino preko metoda objekata naziva se

- **Odgovor: enkapsulacija**

16. Skup metoda sa egzaktnim formatom za poziv svakog objekta, ili, drugim riječima, potpis objekata kod objektnog modela je poznat pod nazivom:

- **Odgovor: interfejs**

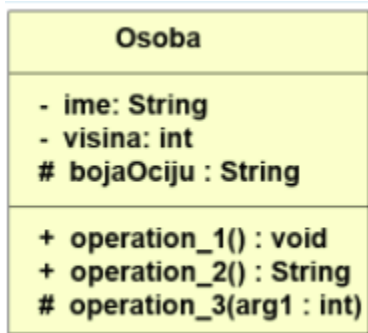
17. Kako se (u objektnom modelu) naziva kolekcija objekata koji dijele isti skup atributa i metoda (odnosno intefrejs), odnosno, drugim riječima, kako se naziva matrica odnosno šablon za kreiranje objekata?

- **Odgovor: klasa**

18. Kako se naziva posebni metod objekta koji se poziva pri kreiranju objekta, a čija je uloga da se objekat dovede u validno stanje sa atributima koji su adekvatno inicijalizovani?

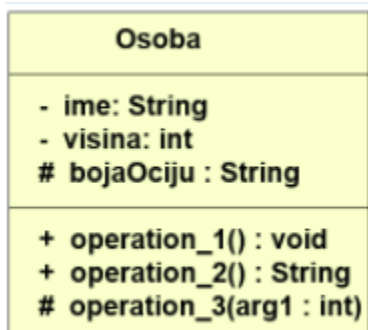
- **Odgovor: konstruktori**

19. Šta na donjoj slici ima modifikator pristupa private?



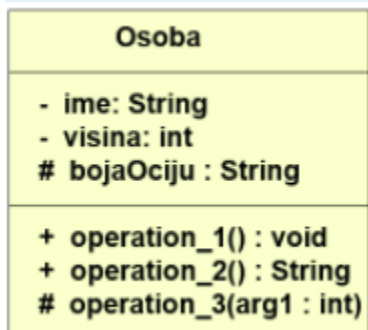
- **Odgovor:** ime: String, visina: int

20. Šta na donjoj slici ima modifikator pristupa protected?



- **Odgovor:** bojaOciju: String, operation_3(arg1: int)

21. Šta na donjoj slici ima modifikator pristupa public?



- **Odgovor:** operation_1(),operation_2()

22. Konstruktor nema povratnu vrijednost.

- **Tačno.**

3. DOMACI

1. Povežite odgovarajuće definicije i termine vezane za SCM

kolekcija proizvoda rada ("artifacts" -
artifakti) kreirana tokom životnog ciklusa
projekta

Odgovor 1

Konfiguracija softvera

submit-ovanje, odnosno, podnošenje
softverske konfiguracije projektnoj bazi –
repozitorijumu

Odgovor 2

commit

verzija programskog koda (koji je dio
softverske konfiguracije), kao i proces kroz
koji se programski kod konvertuje u stand-
alone (samostalnu) formu koja se može
pokrenuti na računaru ("executable code" –
koji se može izvršiti)

Odgovor 3

build

snapshot softverske konfiguracije ili
konfiguracionog item-a u specifičnom
trenutku

Odgovor 4

Verzija

bilo koje parče znanja (a.k.a. "radni proizvod"
odnosno "artifact") kreiran od strane osobe
(developer-a ili customer-a, tj.
naručioca/kupca/korisnika)

Odgovor 5

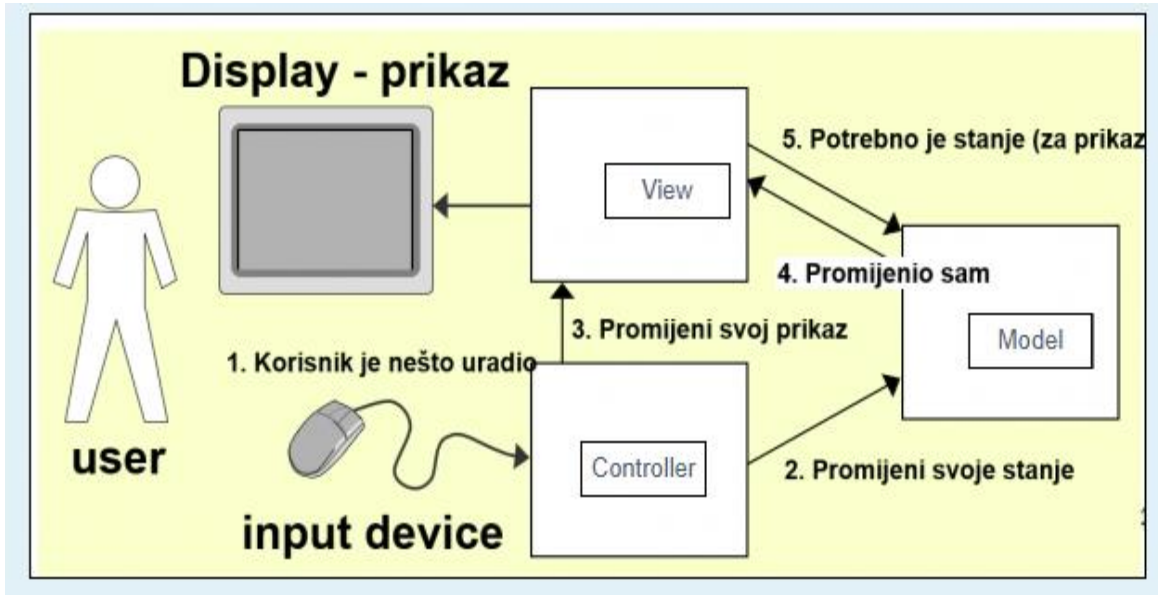
Software configuration item (SCI)

2. Koja vrsta dekompozicije problema u softverskom inženjerstvu omogućava razumijevanje komponenti u kontekstu njihove upotrebe?
- **Projekcija**
3. Kod koje vrste dekompozicije problema u Softverskom inženjerstvu je karakteristično da se definišu dijeljeni interfejsi (API) kojom se podgrupe timova "priključuju" na dijeljenu infrastrukturu?
- **Projekcija**
4. Kojom naredbom signaliziramo Git-u da želimo da "dodamo sadržaj sljedećem commit-u"?
- **git add**
5. Nakon koje naredbe će Git dodati označeni fajl u projekat?
- **git commit**
6. Kojom naredbom Git briše sve commit-e nakon ciljnog commit-a, tako da ne ostanu zapisi o napuštenim verzijama?
- **git reset**
7. Kojom Git naredbom se radi undo commit-a poništavanjem promjena ali tako da ostanu zapisi o napuštenim verzijama?
- **git revert**

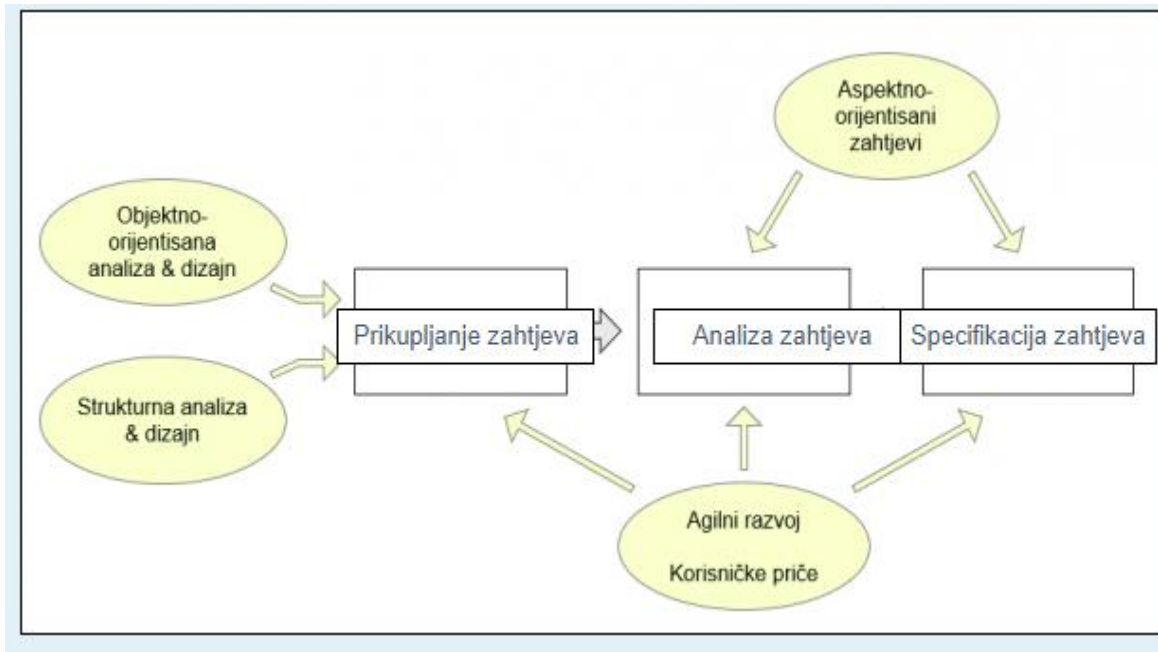
8. U Software Configuration Management terminologiji, koherentno integrisanje doprinosa različitih članova tima je poznato kao:
 - **merge**
9. U Software Configuration Management terminologiji, vraćanje na stariju verziju koda (projekta), odnosno, "undo" je poznat pod nazivom:
 - **revert**

4. DOMACI

1. Prevlačenjem blokova dopuniti prikazanu softversku arhitekturu.



2. Prevlačenjem odgovarajućeg teksta dovršiti donju sliku



3. U Softverskom inženjerstvu, se fokusira na nefunkcionalne zahtjeve ("međusektorske brige") i na dekompoziciju funkcionalnih zahtjeva dok se fokusira na implementaciju funkcionalnih zahtjeva.
4. Povezati sastavne djelove koji se tipično pojavljuju u definisanju različitih stilova softverske arhitekture i odgovarajuće definicije.

Tačke povezivanja između komponenti i konektora

Rasporedi komponenti i konektora koji čine arhitekturu

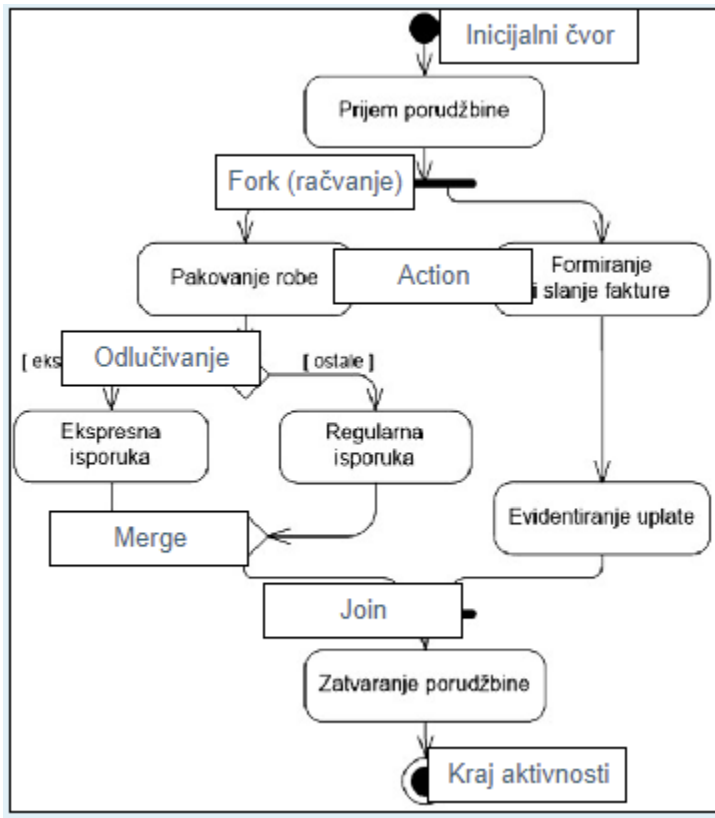
Elementi koji procesuiraju, tj. završavaju posao

Omogućavaju komunikaciju između komponenti

5. Odabrati alate koji se koriste u inženjerstvu zahtjeva
- **Korisničke priče**
 - **Slučajevi korišćenja**
6. Kako se naziva softverska arhitektura koja podrazumijeva da postoji "triggering process" (onaj koji započinje komunikaciju); a na koji odgovara reaktivni dio, koji ili izvršava ili odbija zahtjev i šalje odgovor nazad procesu koji je poslao zahtjev?
- **Client/Server**
7. Kako se naziva trosegmentni stil softverske arhitekture koji podrazumijeva da postoji segment koji čuva podatke i koji posjeduje API-e za preuzimanje stanja i slanje notifikacija o izmjenama stanja jednom drugom segmentu, zatim segment koji pruža korisniku prezentaciju stanja prvog segmenta i konačno, treći segment koji tumači korisnički input?
- **Model-View-Controller**
8. Kako se naziva softverska arhitektura koja podrazumijeva skup komponenti koje transformišu ulaze u izlaze a koje su međusobno vezane tokovima podataka, gdje jedna komponenta proslijeđuje svoj izlaz drugoj komponenti? Tipičan primjer je UNIX shell skripta.
- **Pipe-and-Filter**
9. Kojim terminom je obuhvaćen skup odluka visokog nivoa koje određuju strukturu rješenja (djelove budućeg softverskog sistema i odnose među njima)
- **Softverska arhitektura**
10. Odluke u vezi arhitekture softvera su glavne odluke donijete tokom razvoja i evolucije softverskog sistema. Ove odluke se rano prave i utiču na velike djelove sistema. One se teško kasnije mijenjaju.
- **Tačno.**

5. DOMACI

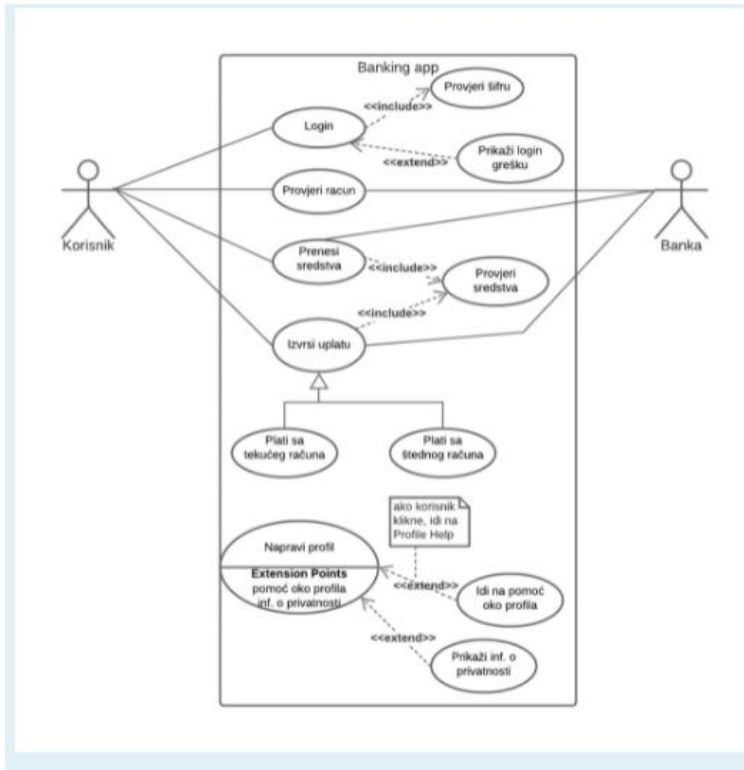
1. Prevlačenjem označite djelove dijagrama aktivnosti. Uputstvo: odaberete blok ispod slike, i prevućete isti na odgovarajuću poziciju na slici



2. Povežite elemente UML use case dijagrama sa odgovarajućim objašnjenjem.

	Relacija proširivanja ✓
	Relacija asocijacije ✓
	Relacija uključivanja ✓
	Relacija generalizacije ✓
	Akter ✓
	Use case ✓

3. Odaberite stavke koje se odnose na aktera u Use case modelu (odnosno, odaberite šta od navedenog može biti akter).
- **Interni korisnik koji ima određenu ulogu u sistemu1**
 - **Hardverski uređaj**
 - **Drugi sistemi koji su u interakciji sa posmatranim sistemom**
 - **Informacioni sistem, aplikacija**
 - **Spoljni korisnik**
4. Za Use Case dijagram na slici, odaberite tačne konstatacije:



- Use Case *Provjeri šifru* se **uvijek dešava** (kada se dešava Use Case *Login*)
- Use case *Prikaži login grešku* se **ne dešava uvijek**

5. Za Use Case dijagram na slici, odaberite tačne konstatacije:

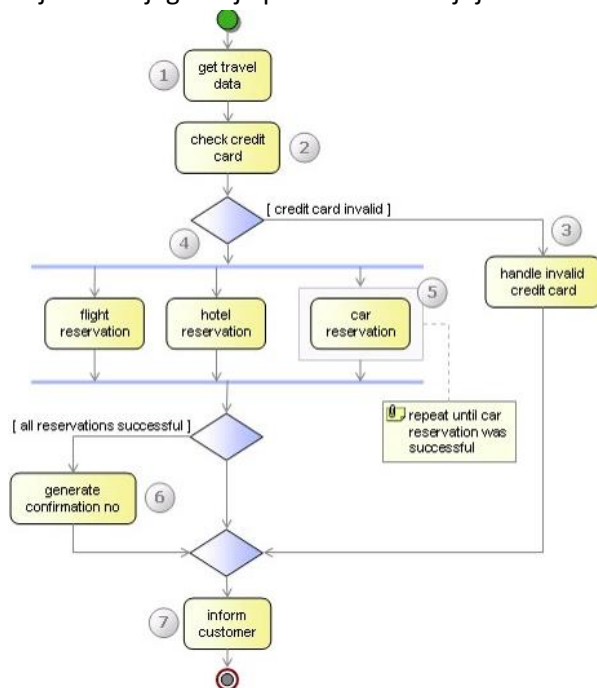


- Use Case *Provjeri sredstva* se **uvijek dešava** (kada se dešava Use Case *Prenesi sredstva*)
- Use case *Prikaži inf. o privatnosti* se **ne dešava uvijek** (kada se dešava Use Case *Napravi profil*)

6. Šta se od navedenog odnosi na dijagram aktivnosti? Izaberite jedan ili više odgovora:

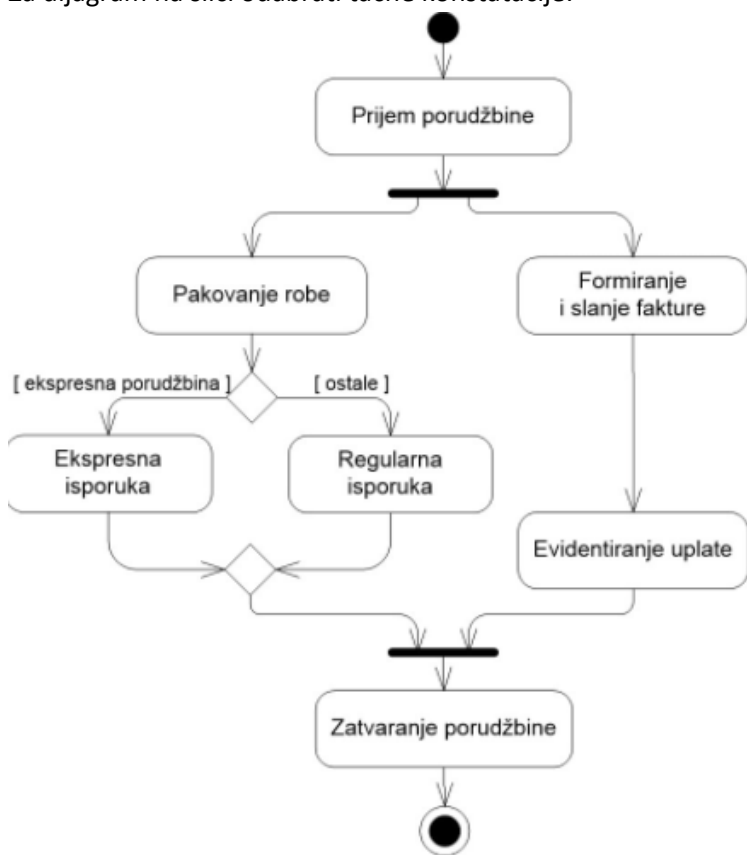
- **Detaljno opisuje slučajeve korišćenja, aktivnosti, scenarija, ponašanja**
- **Opisuje šta sistem radi, ali ne i kako sistem radi**
- **Daje detaljnu specifikaciju use case modela**


7. Koji UML dijagram je prikazan na donjoj slici?



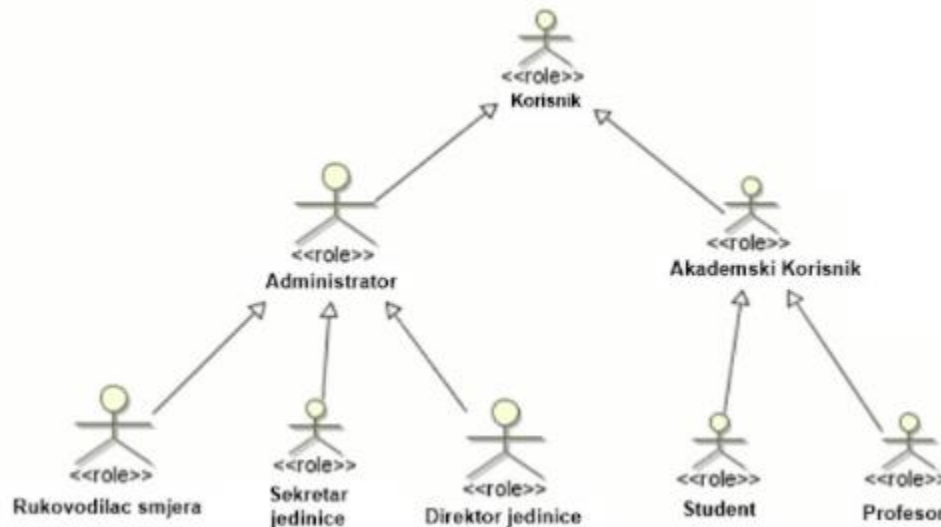
- **Dijagram aktivnosti**

8. Za dijagram na slici odabrati tačne konstatacije:



- Dio u obliku  predstavlja kontrolni čvor
- **Pakovanje robe** i **Formiranje i slanje fakture** se uvijek izvršavaju
- **Pakovanje robe** i **Formiranje i slanje fakture** su aktivnosti ili akcije

9. Šta je predstavljeno na donjoj slici?



- **Primjer generalizacije**

10. Veza između aktera i Use case-a u UML Use Case modelu prikazuje se relacijom:

- **Asocijacije**

11. Na relaciju generalizacije kod UseCase UML dijagrama se odnose sljedeće konstatacije:

- **Veza opšteg i specifičnog use case-a koji nasljeđuje opis opšteg use case-a**
- **Ukazuje na opšti use case i ili akter, i njemu određenije (konkretnije) use case-ove (ili aktere)**

12. Na relaciju proširivanja kod UseCase UML dijagrama se odnose sljedeće konstatacije:

- **Ukazuje se na alternativni, opcioni ili onaj use case koji se događa pod određenim uslovima**

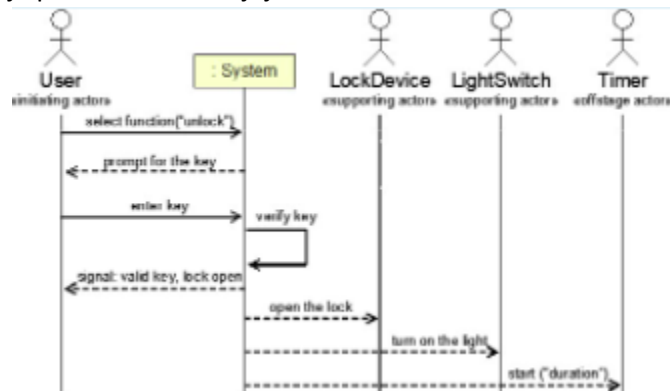
13. Na relaciju uključivanja kod UseCase UML dijagrama se odnose sljedeće konstatacije:

- **Ukoliko više use-case-ova koristi istu obaveznu funkcionalnost (npr. login), tada se ona može izdvojiti u poseban use case i *include* relacija je usmjerena od osnovnog do izdvojenog use case-a**
- **Ukazuje koja funkcionalnost obavezno prethodi**

14. Na relaciju zavisnosti kod UseCase UML dijagrama se odnose sljedeće konstatacije:

- **Ukazuje na to da use case zahtijeva druge use case-ove za svoje određeniju specifikaciju ili implementaciju**

15. Šta je prikazano na donjoj slici?



- **Sistemska dijagram sekvenci**

16. Specifikacija i modelovanje korisničkih zahtjeva obavlja se pomoću

- **Dijagrama aktivnosti**
- **Use case dijagrama**

17. U softverskom inženjerstvu, korak-po-korak opis kako će korisnik koristiti budući sistem, kako bi ostvario biznis ciljeve, poznat je pod nazivom:

- **Use case**

18. Na koju fazu razvoja softvera se odnose Use Case dijagrami?

- **Specifikaciju softvera**

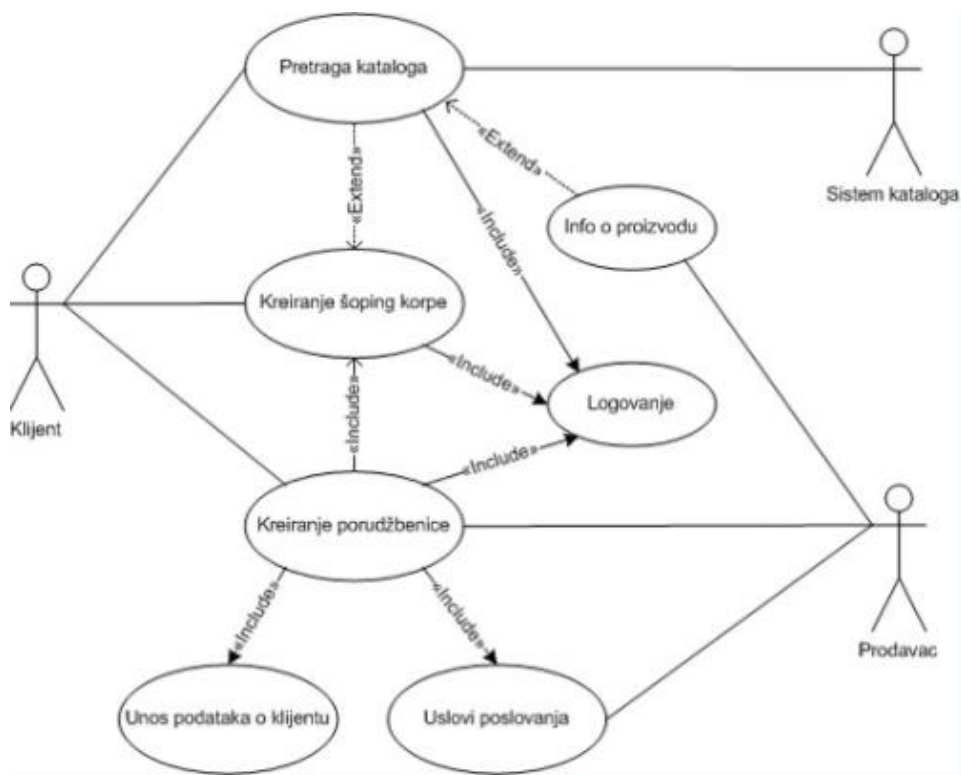
19. Šta predstavlja Use case, odnosno, na šta se on odnosi?

- **Funkcionalnost sistema**
- **Sekvencu aktivnosti koje sistem izvršava a koje su od posebne važnosti za aktera**
- **Šta sistem radi, ali ne i kako sistem radi**
- **Dogovor između naručioca softvera i razvojnog tima oko toga šta sistem treba da radi**

20. Odaberite uloge use-case UML modela (koje se konstatacije odnose na ovaj UML model).

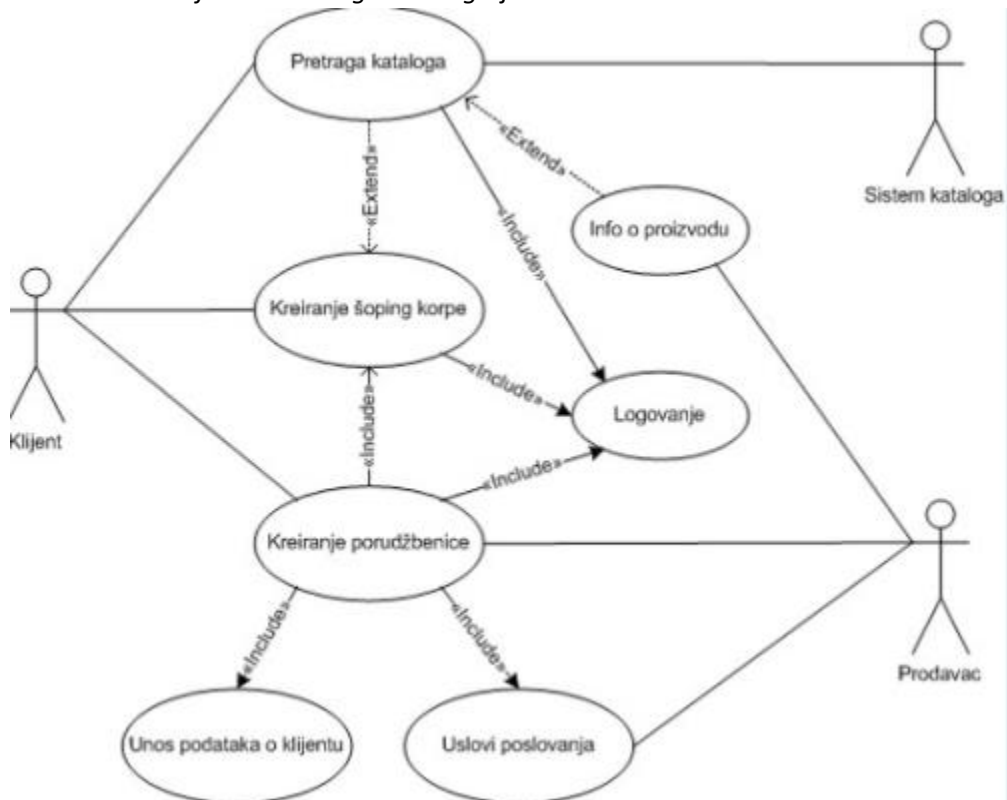
- **Obezbeđuje komunikaciju sa krajnjim korisnicima i ekspertima**
- **Identifikuje korisnike sistema i šta sistem treba da radi**

21. Koji UML dijagram je prikazan na donjoj slici?



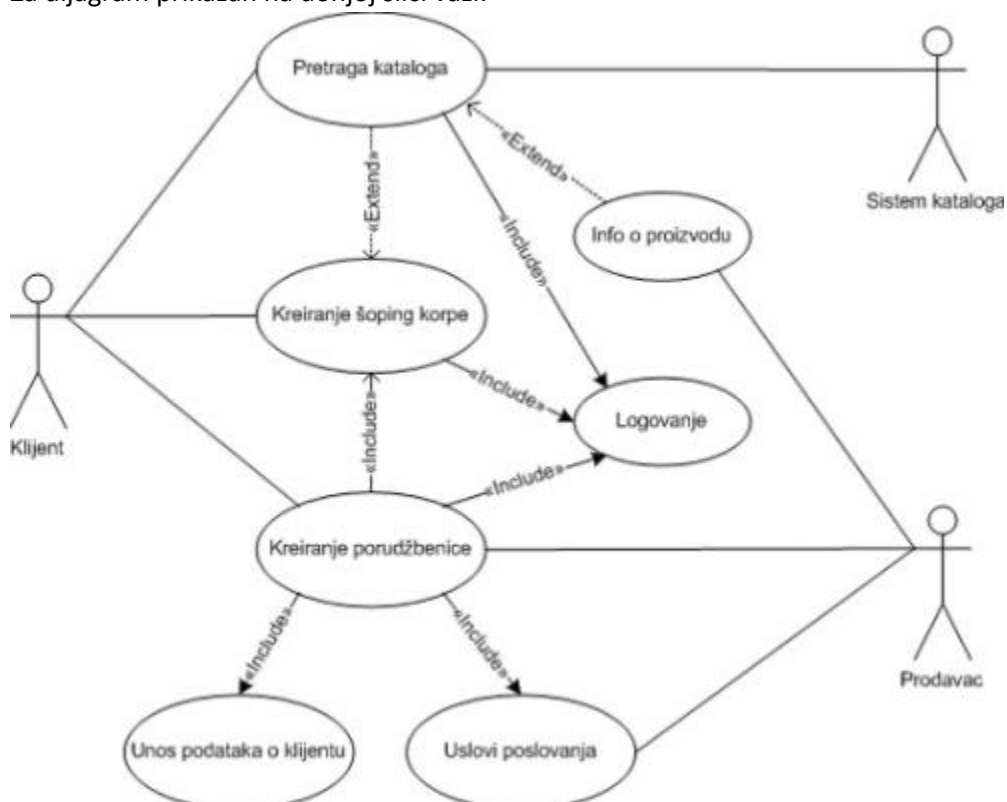
- Use-Case dijagram

22. Veza između *Klijenta* i *Pretrage kataloga* je:



- Relacija asocijacije

23. Za dijagram prikazan na donjoj slici važi:



- Use case *Kreiranje šoping korpe* proširuje use case *Pretraga kataloga*
- Use case *Pretraga kataloga* uključuje use case *Logovanje*

24. Na slici je prikazana matrica koja mapira sistemske zahtjeve u use case-ove. Navedite na osnovu kojih korisničkih zahtjeva je kreiran use case 2? (Navesti u redosljedu od najmanjeg do najvećeg rednog broja)

Req't	PW	UC1	UC2	UC3	UC4	UC5	UC6	UC7	UC8
REQ1	5	X	X						
REQ2	2		X						
REQ3	5	X						X	
REQ4	4	X						X	
REQ5	2	X	X						
REQ6	1			X	X				X
REQ7	2						X		X
REQ8	1					X			X
REQ9	1					X			X
Max PW		5	2	2	2	1	5	2	1
Total PW		15	3	2	2	3	9	2	3

- REQ1, REQ2, REQ5

25. Na slici je prikazana matrica koja mapira sistemske zahtjeve u use case-ove. Navedite na osnovu kojih korisničkih zahtjeva je kreiran use case 7? (Navesti u redosljedu od najmanjeg do najvećeg rednog broja)

Req't	PW	UC1	UC2	UC3	UC4	UC5	UC6	UC7	UC8
REQ1	5	X	X						
REQ2	2		X						
REQ3	5	X						X	
REQ4	4	X						X	
REQ5	2	X	X						
REQ6	1			X	X				X
REQ7	2						X		X
REQ8	1					X			X
REQ9	1					X			X
Max PW		5	2	2	2	1	5	2	1
Total PW		15	3	2	2	3	9	2	3

- **REQ3, REQ4**

26. Na slici je prikazana matrica koja mapira sistemske zahtjeve u use case-ove. Navedite na osnovu kojih korisničkih zahtjeva je kreiran use case 5? (Navesti u redosljedu od najmanjeg do najvećeg rednog broja)

Req't	PW	UC1	UC2	UC3	UC4	UC5	UC6	UC7	UC8
REQ1	5	X	X						
REQ2	2		X						
REQ3	5	X						X	
REQ4	4	X						X	
REQ5	2	X	X						
REQ6	1			X	X				X
REQ7	2						X		X
REQ8	1					X			X
REQ9	1					X			X
Max PW		5	2	2	2	1	5	2	1
Total PW		15	3	2	2	3	9	2	3

- **REQ8, REQ9**

Koji koncept(alat) u inženjerstvu zahtjeva zamjenjuje koncept sistsemskih zahtjeva, izmedju ostalog u cilju rjesenja problema prioritizacije zahtjeva? Korisnicke price

Cime se u inženjerstvu zahtjeva zamjenjuju korisnike prie izmedju ostalog, u cilju rjesenja problema prioritizacije zahtjeva? Sistemske zahtjevi

Komponenta inženjerstva zahtjeva, koja podrazumijeva proces preciscavanja(refining) i modifikacije prikupljenih zahtjeva, nazivaju se:Analiza zahtjeva

Kako se naziva komponenta inženjerstva zahtjeva, koja podrazumijeva dokumentovanje sistemskih zahtjeva na poluformalan ili formalan nacin kako bi se osigurala jasnoca, konciznost i kompletnost?Specifikacija zahtjeva

Koje odluke od ponudjenih, predstavljaju odluke u vezi arhitekture softvera?

Odluka o platformi za razvoj ili o operativnom sistemu

Odluka o dekompoziciji sistema

Odluka o nacinu uklapanja podsistema

Odluka o mapiranju softvera na hardver