# Analiza i modeliranje sustava upravljanja zahtjevima korisnika za servisom u servisnom centru

**Belužić, Lidija**

**SVEUČILIŠTE U ZAGREBU**

**FAKULTET PROMETNIH ZNANOSTI**

**Lidija Belužić**

# ANALIZA I MODELIRANJE APLIKACIJE ZA NADZOR USLUGA SERVISERA PREMA KRAJNJIM KORISNICIMA

**DIPLOMSKI RAD**

**ZAGREB, 2019**

**SVEUČILIŠTE U ZAGREBU**
**FAKULTET PROMETNIH ZNANOSTI**
POVJERENSTVO ZA DIPLOMSKI ISPIT

Zagreb, 1. travnja 2019.

Zavod: **Zavod za informacijsko komunikacijski promet**
Predmet: **Analiza i modeliranje prometnih sustava**

# DIPLOMSKI ZADATAK br. 5161

Pristupnik: **Lidija Belužić (0135234102)**
Studij: Promet
Smjer: Informacijsko-komunikacijski promet

Zadatak: **Analiza i modeliranje sustava upravljanja zahtjevima korisnika za servisom u servisnom centru**

Opis zadatka:

Analizirati i prikazati značajke modeliranja poslovnih procesa primjenom UML-a, s obzirom na to da UML daje skup dobro definiranih grafičkih simbola sa odgovarajućim značenjem.
Definirati zahtjeve na sustav upravljanja zahtjevima za servisom u servisnom centru te primjenom odgovarajućih UML dijagrama (dijagram aktivnosti, dijagram stanja, dijagram rasporeda i komponenti te interakcijski dijagrami) prikazati poslovni proces upravljanja zahtjevima za servisom u servisnom centru.
Razviti Web aplikaciju koja će omogućiti upravljanje zahtjevima za servisom u servisnom centru čija je svrha povećanje kvalitete usluge servisera, skraćenje vremena isporuke usluga te minimiziranje troškove pružanja usluge.

Mentor:

prof. dr. sc. Štefica Mrvelj

Predsjednik povjerenstva za diplomski ispit:

Sveučilište u Zagrebu

Fakultet prometnih znanosti

**DIPLOMSKI RAD**

# ANALIZA I MODELIRANJE APLIKACIJE ZA NADZOR USLUGA SERVISERA PREMA KRAJNJIM KORISNICIMA

# ANALYSIS AND MODELING OF CUSTOMER REQUIREMENTS MANAGEMENT FOR REPAIR SERVIC COMPANIES

**Mentor:** prof. dr. sc. Štefica Mrvelj

**Student:** Lidija Belužić

**JMBAG:** 0135234102

Zagreb, rujan 2019.

ANALIZA I MODELIRANJE APLIKACIJE ZA NADZOR USLUGA SERVISERA PREMA KRAJNJIM KORISNICIMA

# SAŽETAK

Većina današnjih tvrtki radi u natjecateljskom okruženju, što vodi ka potrebi povećanja konkurentne pozicije na tržištu. Da bi tvrtke bile konkurentne na tržištu moraju nužno optimizirati svoje unutarnje procese. Ta optimizacija zahtijeva poslovne modele koji predstavljaju strukturu poslovanja koja omogućava organizacijama analizu i simulaciju promjena prije implementacije. U radu je analizirana problematika upravljanja zahtjevima za servisom u servisnom centru te su primjenom odgovarajućih UML (engl. Unified Modeling Language) dijagrama prikazani poslovni procesi upravljanja tim zahtjevima. Razvijena je web aplikacija koja omogućava cjelokupni proces vođenja termina i evidencija lokacija predviđenih za servisere koji odlaze do krajnjeg korisnika. Predloženi način upravljanja zahtjevima u servisnom centru omogućava povećanje kvalitete usluge servisera i minimiziranje troškova pružanja usluge.

*Ključne riječi: poslovni proces; UML dijagram; web aplikacija*

# SUMMARY

Most of today's businesses operate in a competitive environment, leading to the need to increase their competitive position in the market. In order to be competitive in the market, companies must necessarily optimize their internal processes. This optimization requires business models that represent a business structure that enables organizations to analyze and simulate changes before implementation. The paper analyzes the problems of managing service requests in a service center and presents the business processes of managing these requests using the appropriate Unified Modeling Language (UML) diagrams. A web application has been developed that enables the complete process of keeping dates and records of locations intended for end-user repairers. The proposed method of managing requests in a service center allows to increase the quality of service of the servicer and to minimize the cost of providing the service.

*Keywords: business process; UML diagram; Web application*

# Sadržaj

# 1. Uvod

U današnje vrijeme na tržištu postoji veliki broj tvrtki koje funkcioniraju u natjecateljskom okruženju, te samim time to vodi prema potrebi za povećanjem konkurentne pozicije na tržištu. Da bi postajala konkurentnost na tržištu tvrtke moraju optimizirati svoje unutarnje poslovne procese iz kojih proizlazi potreba za povećanjem kvalitete ponuđenih proizvoda i usluga, minimiziranjem troškova proizvodnje i povećanjem dobiti, te skraćenjem vremena isporuke proizvoda i usluga koje tvrtka ima u ponudi. Poslovni modeli su ključni faktori koji su potrebni kod optimizacije i predstavljaju detaljno opisanu strukturu poslovanja, simulaciju promjena prije implementacije, te omogućavaju prikaz pojedinačne analize organizacije.

Organizacijska struktura i poslovni procesi prate ovisnosti koje zahtijevaju transformacije poslovnih modela u izvršne procese. Inicijativna je zamisao nadopunjavanje tehnologije uz pomoć UML-a (engl. *The Unified Modeling Language*). Na navedeni način se osigurava pojednostavljenje poslovnih zahtjeva, razvoj kompleksnih poslovnih logika i pretvaranje rješenja u aplikativno podržan oblik. UML pruža mogućnost za dobro definiranje dizajna aplikacije, implementaciju, testiranje i razvoj iste, te elimininiranje eventualnih pogrešaka za kvalitetnije i korektnije vođenje poslovnog modela.

U ovom diplomskom radu će biti analizirana problematika upravljanja zahtjevima za servisom u servisnom centru pod naslovom *Analiza i modeliranje aplikacije za nadzor usluga servisera prema krajnjim korisnicima.*

Svrha rada je analiza i prikaz različitih poslovnih procesa primjenom UML-a, ključnih za razvitak kvalitetne aplikacije koja bi zadovoljila krajnje korisnike i tvrtku. Stoga su u ovom radu istražene značajke modeliranja poslovnih procesa primjenom UML-a, s obzirom na to da UML daje skup dobro definiranih grafičkih simbola sa odgovarajućim značenjem.

Cilj rada je uz pomoć UML dijagrama prikazati poslovni proces upravljanja zahtjevima i razviti aplikaciju koja će omogućiti skraćivanje vrijeme isporuke usluga i povećati kvalitetu usluga servisera, te minimizirati troškove pružanja usluge.

Diplomski rad sastoji se od devet funkcionalno povezanih teza:

1. Uvod
2. Modeliranje poslovnih procesa i značajke modeliranja poslovnih procesa primjenom UML-a
3. Definiranje zahtjeva na sustav upravljanja zahtjevima za servisom u servisnom centru
4. Modeliranje kontrole toka i toka podataka primjenom dijagrama aktivnosti
5. Modeliranje vrsta, svojstava i stanja objekata

6. Modeliranje organizacijske strukture sustava
7. Modeliranje interakcije između sudionika u procesu upravljanja zahtjevima za servisom
8. Web aplikacija za upravljanje zahtjevima za servisom u servisnom centru
9. Zaključak

U drugom poglavlju pod nazivom *Modeliranje poslovnih procesa i značajke modeliranja poslovnih procesa primjenom UML-a* navedene su i opisane vrste poslovnih procesa.

Treće poglavlje *Definiranje zahtjeva na sustav upravljanja zahtjevima za servisom u servisnom centru* objašnjena je potreba za navedeni sustav, te je prikazana primjena dijagrama slučaja uporabe.

U četvrtom poglavlju *Modeliranje kontrole toka i toka podataka primjenom dijagrama aktivnosti* prikazane su aktivnosti koje je potrebno izvršiti za uspješno proveden poslovni proces.

*Modeliranje vrsta, svojstava i stanja objekata* peto je poglavlje rada gdje su navedene i prikazane vrste, svojstva i stanja objekata strukturirane dijagramom klasa ili objekata.

U šestom poglavlju *Modeliranje organizacijske strukture sustava* objašnjeni su dijagrami komponenata i rasporeda, te je dijagramom komponenata opisana struktura organizacije.

*Modeliranje interakcije između sudionika u procesu upravljanja zahtjevima za servisom* pomoću dijagrama međudjelovanja i suradnje prikazuje se komunikacija između subjekata uključenih u poslovni proces.

U osmom poglavlju pod nazivom *Web aplikacija za upravljanje zahtjevima za servisom u servisnom centru* prikazana je kreirana web aplikacija na temelju problematike rada.

# 2. Modeliranje poslovnih procesa i značajke modeliranja poslovnih procesa primjenom UML-a

Poslovni procesi koriste se u svim smjerovima industrijskog razvitka. Procesi se sastoje od niza koraka koje određena skupina sudionika uključena u sam proces provodi kako bi se ostvario krajnji cilj. Također se može definirati kao skup aktivnosti i odluka koje se izvode na vanjski poticaj kako bi se ostvario vanjski cilj koji je moguće mjeriti, traje određeno vrijeme i troši ulazne resurse pretvarajući ih u specifične produkte od značaja za krajnje kupce.

Svaki korak koji sadrži zadatak namijenjen je specifičnoj poslovnoj funkciji odnosno sudioniku procesa, te organizacijskoj jedinici poslovanja sa vrlo preciznim i detaljnim skupom odgovornosti i aktivnosti koje je potrebno izvršiti. Ujedinjeni zadaci svakog pojedinačnog sudionika i poslovnih funkcija procesa nakon pozitivnog izvršenja vrše kvalitetno odrađen poslovni proces.

Upravljanje poslovnim procesima je sustavni pristup koji vodi ka poboljšanju razdijeljenih manjih segmenata procesa i pomaže u postizanju funkcionalne organizacijske strukture poslovnog sustava. Segmentiranje poslovnih procesa pruža tvrtki vizualni prikaz funkcionalnih cjelina različitih procesa i bolju vidljivost koja omogućuje pomoć pri jačanju operativne učinkovitosti poslovanja [1].

Poslovni procesi su prema [1] ovisno o industriji, prirodi posla kojom se tvrtka bavi i organizaciji podijeljeni u različite kategorije koje uključuju:

1) Operativni procesi – predstavljaju poslovne aktivnosti koje direktno utječu na ostvarenja poslovnih ciljeva organizacije. Bave se osnovnim poslovima te su povezane sa vrijednosnim lancem[1]. Operativni procesi uključuju isporuku vrijednosti produkata[2] kupcu.

2) Podupirući procesi ili sekundarni procesi – povezuju osnovne procese i funkcije unutar organizacije. Pod funkcije organizacije smatraju se pojedini sektori koji su potrebni da bi tvrtka funkcionirala kvalitetno i pravilno npr. računovodstvo, prodaja i slično.

3) Procesi upravljanja – kontroliraju, mjere i prate aktivnosti vezane za poslovne sustave i procedure. Proces upravljanja usmjeren je na internu komunikaciju, strateško upravljanje, određivanje financijskog ulaganja u određene sektore i određivanje koliko su kompetentni i potrebni, te upravljanje infrastrukturnom mrežom odnosno kapacitetom. Svaka poslovna

[1] Vrijednosni lanac - prikazuje ukupnu vrijednost koju stvara poduzeće, a zasniva se na promatranju poduzeća kao skupa odvojenih ali i povezanih aktivnosti kojima se stvara vrijednost za kupce [2]

[2] Produkt – ili proizvod u širem smislu, označava svaku ponudu (uslugu, proizvod, ideju) koja je ponuđena na tržištu i može zadovoljiti potrebe i želje korisnika [3]

organizacija koristi navedenu strategiju kako bi nadgledale sve poslovne procese uključene u radu i osigurale kvalitetan i nesmetan rad. Pomaže pri poboljšanju procesa eliminirajući viškove i greške ukoliko ih ima, radi djelotvornog i učinkovitog djelovanja. Vrlo bitne aktivnosti upravljanja poslovnih procesa sadrže korake poput modeliranja poslovnih procesa, izvršavanje, praćenje i optimizaciju.

Modeliranje poslovnih procesa je prikaz skupa različitih komponenti procesa, sustava ili određenog područja predmeta međusobno povezanih u jednu upotrebljivu cjelinu. Postupak modeliranja obuhvaća korištenje objekata od strane softverskih inženjera. Sam objekt se definira kao dio softvera koji uključuje pasivne podatke, akcije i reagira na unutarnje ili vanjske događaje koji su potrebni da bi se kroz izvršavanje akcije realizirala radnja. Međusobno djelovanje objekata vrši karakterističnu promjenu stanja koje su vremenski utemeljene, nisu determinističke, te se koriste kao pomoć za vizualizaciju ljudima i bolje razumijevanje sustava.

Organizacijski inženjering pokriva stvaranje obrazaca poslovnih procesa radi lakšeg razumijevanja cjelokupnog poslovnog procesa, iz razloga što su sve organizacije različite, ali izvršavaju slične procese u provođenju svojih tvrtki. Obrasci služe za stvaranje novih poslovnih procesa (ponajviše onih koji se koriste u informacijsko-telekomunikacijskim tehnologijama), redizajniranju postojećih poslovnih procesa i podjeli različitih ideja o praktičnim organizacijama. Metodologija koja se koristi za kreiranje poslovnih procesa vrlo je slična modeliranju softvera.

Općenito modeliranje poslovnih procesa objektno modeliranog softvera sadrži hijerarhiju specijaliziranih objekata koji najčešće imaju pridružene akcije, odnosno metode. Također navedene metode moraju imati svoj redoslijed događanja i izvršenja. Sam cilj i svrha poslovnog modeliranja su izrada i apstrakcija složenih stvari koje uključuju osnovne poslovne funkcije, poboljšanje poslovanja, djelovanja kao baza podrške informacijskim sustavima, jasnije razumijevanje ključnih mehanizama točno određenog poslovnog procesa, identificiranje novih poslovnih mogućnosti i korekcije postojećeg sustava, radikalne promjene u poslovanju ili identificiranje mogućeg outsorcinga[3]. Poslovni modeli se najčešće prezentiraju kroz jedan ili više dijagrama kojih može biti nekoliko vrsta, što ovisi o situaciji i specifičnim strukturama poslovanja koje je potrebno precizirati i prikazati. UML dijagramima osim što je primarni cilj prikazati poslovni proces, također prikazuju pravila, svrhu, ciljeve, objekte poslovanja i odnose između objekata kao i same interakcije. Skup komponenata procesa, sustava i područje predmeta koji prikazuju model poslovnog procesa sadrže također određen broj aktivnosti koje objekt preuzima kao ulaz i dodaje mu dodanu vrijednost kako bi uspio zadovoljiti pridružene zahtjeve [1].

[3] Outsorcing – davanje određenog posla vanjskim dobavljačima, odnosno uzimanje vanjskih dobavljača za određeni posao koji primjenjuju tvrtke radi smanjenja troškova u djelatnostima koje im nisu temeljne [4].

U procesu modeliranja prema [5] postoje vrlo važni pojmovi koji se koriste u toku procesa, a to su:

1) Proces – označava apstrakciju koja pokazuje broj aktivnosti koji je određen i čini skup ulaznih objekata koje procesuiraju izlazni skup objekata, vrijednost za kupca (unutarnju ili vanjsku organizaciju tvrtke). Ulazni, izlazni i objekti koji su potrebni tijekom procesa smatraju se poslovnim resursima.

2) Događaj – je stanje promjene koje najavljuje da se dogodila nova promjena u procesu koja ga generira i prima jedan ili više drugih različit procesa.

3) Resursi – su sve supstance, faktori ili objekti koji su potrebni za razvitak, rast i reprodukciju poslovnih procesa.

4) Ciljevi – su složeno željeno stanje jednog ili više resursa. Također su povezani tijekom cijelog procesa i obrađeni u svim segmentima.

5) Poslovna pravila – uređuju sustav, pokazuju kako postići ravnotežu između strukture i kreativnosti, vodeći k uspjehu koji se može usmjeriti prema boljoj kvaliteti poslovnog procesa općenito, a ponajviše u tržišnom udjelu.

6) Opći mehanizmi – koriste se u svim dijagramima, jer jedna referentna bilješka može sadržavati naziv vanjskih dokumenata ili drugi dijagram koji pokazuje gdje se mogu pronaći detaljnije informacije o modelu.

Svrha izrade modela poslovnog procesa je vrsta grafičkog prikaza koja omogućuje prikaz za bolje razumijevanje, analiziranje, poboljšanje ili zamjenu poslovnog procesa. Također prikaz vizualnih prezentacija modela može se formulirati u različitim formatima. Vrlo bitna stavka modeliranja je odlučivanje koliko je potrebnih detaljnih informacija potrebno uključiti u model kako bi se steklo razumijevanje teme. Bitno je da se relevantne informacije ne zanemaruju slučajnim odabirom činjenica vezanih uz poslovni proces kako bi se steklo razumijevanje i pravilno primijenila tehnika odabira jer posljedica navedenog neće točno predstaviti modelirano. Iz tog razloga potreban je logičan način prikazivanja podskupa informacija kako bi se osiguralo da je model točan. U svrhu modeliranja koristi se apstrakcija mehanizama koji omogućava filtriranje bitnih informacija, jer kvalitetna apstrakcija naglašava detalje koji su značajni za čitatelje, korisnike ili programere procesa sustava. Stvaranje efektivne apstrakcije nije lako ostvariti jer je odluka vrlo subjektivna pri odabiru tehnike koja će podržavati modeliranje poslovnog procesa. Potrebno je da odabrana tehnika omogućava simulaciju, procesno mapiranje, dijagrame aktivnosti i sheme, definiciju integracije za modeliranje funkcija kao i mnoge druge. Bez obzira na tehniku, uvijek mora postojati jedinstveni cilj koji je zajednički i povezan sa svima za vizualni prikaz objekata koji izvršavaju svoje funkcije, aktivnosti i daju konkretan rezultat. Vizualna slika olakšava, odnosno smanjuje otpor i služi za reinženjering poslovnog procesa ili softverskog

inženjerstva kako bi se jednostavno koristilo objašnjenje navedenih tema. Time se zatim eliminira nepotrebno i vrši provjera razvija li se poslovni proces uz dobro dokumentiranu tehniku i pomoću naprednih alata putem softverskog inženjerstva [6].

UML jezik općenito je primjenjiv za svaku osobu koja je uključena u izradu, implementaciju, testiranje i održavanje softvera. U taj proces mogu biti uključeni arhitekti i tvorci koji dizajniraju sustav te im je zadatak zadovoljavanje korisničkih potreba, odnosno zahtjeva krajnjih korisnika, zatim programeri koji provode izrađenu arhitekturu u izvršni kod i osposobljen sustav. Također u kreiranju poslovnog procesa javlja se osoblje zaduženo za testiranje, održavanje, osiguranje kvalitete i provođenje verifikacije i validacije same strukture sustava kao i osoblje za kreiranje, dokumentaciju, katalogizaciju i provjeru da li su sva postavljena pravila poštovana. Da bi poslovanje funkcioniralo nužno je angažirati kvalitetne i sposobne voditelje projekta koji će osigurati smjer upravljanja resursima potrebnim za isporuku uspješnog sustava.

Za dobro shvaćenu arhitekturu objektno orijentiranog sustava potrebno je nekoliko međusobno povezanih i komplementarnih gledišta ili pogleda kao što su:

1. pogled sa gledišta slučaja uporabe kao predstavljeni zahtjevi sustava

2. pogled sa gledišta dizajnera gdje su obuhvaćeni rječnik problemskog područja i rješenja

3. pogled sa gledišta procesa u modeliranju distribucije spojnica između sustavskih procesa

4. pogled sa gledišta implementacije kada je fizička realizacija izvršena, te

5. pogled osobe koja razvija sustav poslovnog procesa koja je fokusirana na probleme sustavskog inženjerstva.

Da bi se uspješno upravljalo organizacijom, potrebno je kvalitetno kreirati poglede modeliranja poslovnog procesa u svrhu povećanja učinkovitosti postizanja postavljenih ciljeva, djelovanje organizacije, te povezivanja skupa radnih koraka za koje je moguće odrediti trajanje i izvođenje uz potrebne resurse [7]. Pogledi poslovnog modeliranja dijele se na 4 različita prikaza:

1. Poslovni pogled (engl. *Business View*) - To je opći poslovni pogled. Ovaj prikaz opisuje strukturu ciljeva tvrtke i ilustrira probleme koji se moraju riješiti kako bi se postigli ti ciljevi.

2. Poslovni procesi - ovaj prikaz predstavlja aktivnosti i vrijednost koju stvara poslovanje. Također ilustrira interakciju između procesa i resursa kako bi se postigao cilj svakog procesa. Ovo gledište još uvijek pokazuje postojeću interakciju između različitih procesa.

3. Poslovna struktura - ovaj pogled predstavlja strukturu poslovnih resursa, kao što su poslovna organizacija ili struktura proizvedenih proizvoda.

4. Poslovno ponašanje - predstavlja pojedinačno ponašanje svakog resursa i procesa poslovnog modela.

Ti pogledi nisu zasebni modeli, već različiti pogledi na jedan ili više poslovnih aspekata. Kada su zajedno, pogledi stvaraju cjelovit poslovni model [5].

U sljedećim primjerima biti će navedeni, objašnjeni i vizualno prikazani primjeri UML dijagrama na temelju problema sa ciljem i svrhom rješavanja.

# 3. Definiranje zahtjeva na sustav upravljanja zahtjevima za servisom u servisnom centru

Veliki broj tvrtki ili privatnih djelatnosti koje se bave pružanjem popravaka kvarova danas su vrlo nezadovoljne postojećim odnosom i komunikacijom svojih djelatnika servisera i korisnika. U današnje vrijeme veliki je problem dogovor servisera i korisnika za određeno vrijeme u kojem bi se oboje uskladili. Korisnici u većini slučajeva rade preko dana, te nisu u mogućnosti biti kod kuće i dočekati servisera u vrijeme kada bi njemu to odgovaralo. Također kasnije kada su korisnici slobodni, serviseri imaju vrlo puno korisnika te nikako ne stižu odraditi sve dobivene zahtjeve ili vrlo loše procjene svoj dolazak kod korisnika. Tako se često dogodi da se krajnji korisnik i serviser dogovore za određeno vrijeme u koje serviser ne stiže doći, već kasni po nekoliko sati. U tom slučaju korisnik je „izgubio“ mnogo vremena čekajući na dolazak servisera.

Zadovoljstvo korisnika je najbitnija stavka svake tvrtke. Ukoliko korisnik nije zadovoljan u većini slučajeva neće se više vratiti u istu ili koristiti njene usluge. Iz tog razloga uvelike od koristi mogu biti različiti programi ili aplikacije koje će korisnik znati koristiti, shvatiti ih, te na kraju i biti potpuno zadovoljan cjelokupnom uslugom.

Ukoliko se dogodi kvar određenog kućanskog aparata kod krajnjeg korisnika kod kuće, on uglavnom koristi pomoć servisera radi popravka. Najčešće se radi o tehnici koju nije moguće prenositi radi velike težine ili glomaznosti. U tom slučaju korisnik poziva servisera za otklanjanje kvara na njegovoj kućnoj adresi.

Veliki problem servisera je što imaju vrlo malo vremena u kojemu mogu poslužiti sve svoje korisnike u vremenu koje odgovora korisniku. Većina korisnika je raspoloživa dočekati servisera u isto doba dana, odnosno nakon završetka svog radnog dana. Također serviser najčešće ne može znati o kakvom se kvaru radi kod korisnika dok ne dođe kod njega te ne vidi problem i odredi očekivani vremenski period koji mu je potreban za izvršenje zadatka (otklanjanje kvara).

Za pomoć pri otklanjanju navedenih problema i ispunjenje zahtjeva korisnika i servisera u ovom radu predložena je web aplikacija koja je zamišljena na način da dispečer, odnosno osoba koja ima admin prava, komunicira sa serviserom te prima od njega određene podatke. Kada serviser dođe kod prvog korisnika u danu u dogovoreno vrijeme i vidi određeni kvar na kućnom pomagalu, te procjeni određeno vrijeme koje mu je potrebno za popravak, putem web aplikacije unosi vrijeme dolaska, vrijeme potrebno za popravak i po završetku potvrđuje vrijeme kada odlazi od korisnika. Serviser će putem web aplikacije primiti informaciju o svom sljedećem odredištu koje je definirao dispečer. Dispečer u centru temeljem prikupljenih informacija koje je unio serviser i informacija koje je unio korisnik planira vremena dolaska servisera kod sljedećeg korisnika. Proces se ponavlja nakon svakog dolaska kod korisnika od strane

servisera prema dispečeru, koji ga dalje upućuje na određene lokacije. Ukoliko se prolongira dolazak servisera korisniku, korisnik dobiva obavijest putem mail-a. Na taj način je korisnik u svakom trenutku upućen u koje vrijeme će serviser stići kod njega, te može planirati kako će iskoristiti svoje vrijeme.

## 3.1. Dijagram slučaja uporabe

Dijagrami slučaja uporabe (engl. *Use Case Diagram*) su jedni od najučinkovitijih dijagrama kad se kombiniraju sa slučajevima uporabe koji koriste tekst i koji definiraju navedene slučajeve u dijagramu. Vrlo su elegantno rješenje za vizualni prikaz sustava i općenito poslovnih procesa, te ih je lako shvatiti. Također nemaju puno nepotrebnog zapisa, što dijagram čini lako čitljivim. Višenamjenski su u upotrebi softverskog inženjerstva jer sa na temelju njih mogu dokumentirati zahtjevi koji mogu poslužiti kao test mogućih slučajeva i podršku marketinškim zadacima navodeći najistaknutije značajke potrebne sustavu. Dijagramom slučaja uporabe moguće je prenijeti zbirku slučajeva koji definiraju ili demonstriraju proces i kao prikaz važne apstrakcije daju jednostavnu notaciju za laku izgradnju i lako razumljiv poslovni proces kojeg je potrebno realizirati [6].

Za početak da bi se model problemskog područja dobro definirao potreban je opis skup klasa u početku samog razvoja. Prikupljanje zahtjeva korisnika i potreba u odnosu na sustav koji se želi modelirati ili analizirati, koriste se za opis odnosa sustava i korisnika što vrlo često bude dugotrajan i složen postupak, ali iz tog razloga UML dijagram slučaja uporabe takvu problematiku pojasni i vizualizira. Model slučaja uporabe mogao bi se definirati kao sredstvo za određenje funkcionalnosti koje se očekuju od rada u sustavu.

### 3.1.1. Sudionici i funkcionalnosti slučaja uporabe

Akteri slučaja uporabe predstavljaju uloge koje su u odnosu na modelirani sustav i gledaju se kao zasebne jedinke ili entiteti izvan modeliranog sustava kao što su baze podataka ili dugi sustavi. Prikazuju se simbolom osobe i trebaju sadržavati kratak naziv koji ne određuje tu određenu osobu već označava ili predstavlja ulogu u poslovnom procesu kao korisnik, vanjski sustav ili određeni dio sustava koji je potrebno modelirati i ima opise uporabe. Također sudionik može imati i više uloga u istom modeliranom sustavu kojima inicijalizira i prima određenu vrijednost iz slučaja uporabe.

Skup scenarija koji opisuju željeno ponašanje sustava i slijed aktivnosti sustava koje ispunjavaju ciljeve korisnika opisuje se od strane sudionika. Također moguće je prikazati mogući način uporabe sustava, gdje se izražava u naredbodavnom obliku i sadašnjem vremenu i prikaz slučaja uporabe ne predlaže način modeliranja ili

ostvarenja sustava. Funkcionalnosti sustava slučaja uporabe opisuje ponašanje sustava, odnosno što sustav treba raditi, ali ne prikazuje način na koji postići to ponašanje [8]. Skup sudionika unutar modela mora uključivati sve sudionike koji sudjeluju i potrebni su u procesu, te razmjenjuju informacije sa sustavom i sve funkcionalne zahtjeve postavljenje prema sustavu. Putem dijagrama slučaja uporabe prikazuju se odnosi sudionika i slučaja uporabe. Sudionik ili akter koji direktno sudjeluje i izvršava aktivnost određenog slučaja uporabe smješta se sa lijeve strane dijagrama. Ostali sudionici kojima su namijenjeni rezultati određenog slučaja uporabe ili istovremeno iniciraju određeni slučaj uporabe i namijenjeni su mu rezultati tog slučaja uporabe smještaju se s desne strane dijagrama. Uz pomoć strelica prikazuje se koji sudionici su uključeni u koje slučaje [9].

### 3.1.2. Opis ponašanja sustava i podjela tijeka događaja

Ponašanje sustava se može opisati na više načina uz pomoć korištenja različito strukturiranog teksta. Tekstovi koji se koriste potrebni su za opis akcija koje sudionik izvodi (pozivi koje upućuje sustav), odazive sustava na aktivnosti sudionika i pozive i odazive koji obuhvaćaju tijek nužnih događaja za ostvarenje ponašanja sustava. Opis se dijeli na:

1. neformalno strukturiran tekst
2. formalno strukturiranim tekstom s uvjetima usmjeravanja
3. pseudokodom[4].

Tijek događaja u slučaju uporabe može se podijeliti na različite segmente koje ovise u kakvim okolnostima sudionik i sustav prolaze, način na koji se izvodi, događaju li se pogreške u sustavu ili ne te kakav tijek događaja predstavlja. Podjela događaja se izvodi na:

1. Osnovni tijek događaja - predstavlja rad sustava u normalnim okolnostima, odnosno pretpostavlja se da ključni akter i sustav ne prave nikakve pogreške tj. od početka do kraja rad sustava se izvodi točno kako je napisano. Opis slučaja uporabe uvijek sadrži osnovni tijek događaja i predstavlja poželjan tijek.
2. Zamjenski ili alternativni tijek događaja – događa se u manjem broju slučajeva i opisuje dodatna ponašanja sustava. Ishod zamjenskog tijeka jednak je ishodu osnovnog tijeka događanja samo se do njega dođe drugim putem.

[4] Pseudokod – ( grč. *Pseudeos* – laž ) sastoji se od izraza koji su kratki u govornom jeziku, te ukratko opisuju i objašnjavaju pojedine zadatke [10].

3. Iznimni tijek događaja – pojavljuje se i izvršava u slučaju pojave pogreške ili neispunjavanja svih zadanih uvjeta, predstavlja poželjan tijek događaja i slučaj uporabe općenito uvijek sadrži barem jedan iznimni tijek događaja.

Tekstualni oblik uvelike pomaže u opisivanju poslovnog procesa i rada sustava. Osnovni tijek koji se opisuje sastoji se od jednog do tri odjeljka teksta, zamjenski tijek se opisuje sa jednom do dvije rečenice, a zahtjevi se očitavaju iz navedenog teksta kojim se obavlja rad sustava. Pojedini slučajevi uporabe na sebe vežu nekoliko zahtjeva, ali moguće je da se više slučajeva uporabe vezuje uz jedan zahtjev. Bilježenjem ponašanja sustava pruža se put programerima da se dođe do zajedničkog razumijevanja s krajnjim korisnicima sustava i stručnjacima za navedene domene. Također se koristi za provjeru valjanosti arhitekture sustava i kako se razvija tijekom kreiranja i ostvaruju se suradnje čiji elementi djeluju zajedno u svakom slučaju uporabe.

### 3.1.3. Modeli i relacije slučaja uporabe

Odabir modela slučaja uporabe utječe na sve osnovne modele razvoja programske podrške s povezanim procesom. Model također omogućuje brzu izradu prototipa u kojem ga zajedno izrađuju korisnik i programer čime se dokazuje izvedivost sustava, jednostavna izrada osnovnog tijeka događaja, a unaprijediti sustav na temelju postojećeg pruža mogućnost integracije na temelju raspoložive dokumentacije. Postoje tri osnovna modela koja se dijele na analizu, plan izvedbe i model ostvarenja i rasporeda. Model analize koristi se u slučaju prikazivanja moguće realizacije slučaja uporabe, detaljnijeg izražavanja jezikom programera i u slučaju prikaza realizacije slučaja u kojem se pridjeljuje ponašanje opisano konkretnim objektima.

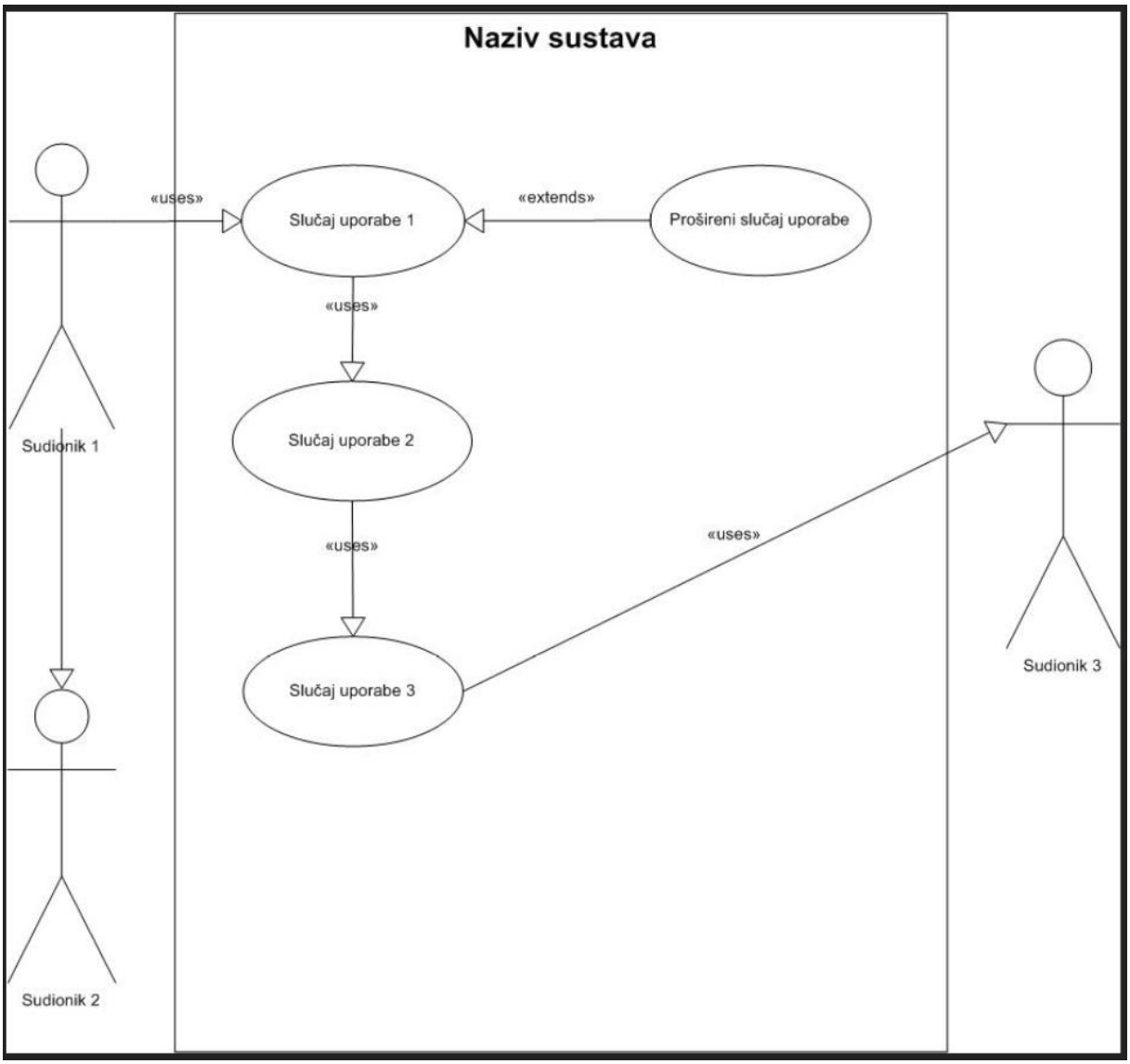
Relacije koje se koriste u slučajevima uporabe olakšavaju razumijevanje ponašanja sustava, izdvajanjem i povezivanjem zajedničkog ponašanja, te akcija između aktera i slučaja. Funkcije koje su zajedničke u više slučaja uporabe izdvajaju se relacijom obuhvaćanja. U toku tvorbe slučaja uporabe vrlo je bitno da se osnovni slučaj definira neovisno i mora biti jasan. Uključene su tri moguće vrste relacija:

1. Obuhvaćanja (engl. *Include*) – relacija obuhvaćanja koristi se kada slučaj uporabe izričito obuhvaća funkcionalnosti drugog slučaja u određenim trenucima tijeka događaja. Slučaj obuhvaćanja ne može postojati samostalno već mora biti povezan sa minimalno jednim ili više osnovnih slučajeva. Relacija obuhvaćanja omogućava umetanje jednog slučaja uporabe u drugi slučaj. Moguća je realizacija u osnovnom slučaju uporabe uključiti slučaj obuhvaćanja koji je potrebno izvršiti „preko reda“, kako bi osnovni slučaj mogao nastaviti s radom. Zajedničke korake se uzima u jedan scenariji i stavljaju se u poseban slučaj, zatim se isti povezuje relacijom

obuhvaćanja sa slučajevima uporabe gdje se isti koraci pojavljuju. Svi koraci obuhvaćanja se moraju izvršiti.

2. Proširenja (engl. *Extend*) – ukoliko nije nužno da osnovni slučaj uporabe obuhvaća funkcionalnosti drugog slučaja na određenom tijeku događaja, koriste se mjesta proširenja, odnosno relacija proširenja koja se implicitno uključuje u ponašanje drugog slučaja na mjesto koje je indirektno određeno. Navedenom relacijom pokušava se izdvojiti funkcionalnosti za koje postoji mogućnost da se slobodno odaberu ili se pojavljuju u iznimnim i određenim uvjetima. Moglo bi se reći da kreiranjem potpuno novog slučaja uporabe u svrhu zamjenskog tijeka događaja koji uz sebe povezuje nekoliko koraka koristi proširenje, odnosno povezivanje slučajeva uporabe. Sastoji se od jednog ili više segmenata ponašanja koja opisuju dodatno prošireno ponašanje osnovnog slučaja. Segmenti mogu naknadno biti umetnuti u osnovni slučaj na različitim mjestima tijeka događaja koje se nazivaju točke proširenja. Bitno je naglasiti da prošireni slučaj uporabe ima pristup svim atributima osnovnog slučaja, kao i izmjenjivati ih, dok osnovni slučaj uporabe ne može izvršiti kontra efekt prema proširenom slučaju. U model se mogu dodavati relacije proširenja da bi se pokazale situacije poput opcionalnog ponašanja sustava, tijek koji se izvršava samo pod određenim uvjetima i ukoliko je potrebno umetnuti skup segmenata ponašanja u osnovni slučaj uporabe.

3. Poopćavanja (engl. *Generalization*) - relacija poopćavanja se koristi kada podslučaj uporabe nasljeđuje ponašanje i karakteristike višeg slučaja. Podslučaj može dodati novo ponašanje ili preko ponašanja kojim je do tog trenutka funkcioniralo, prepisati novo [9].



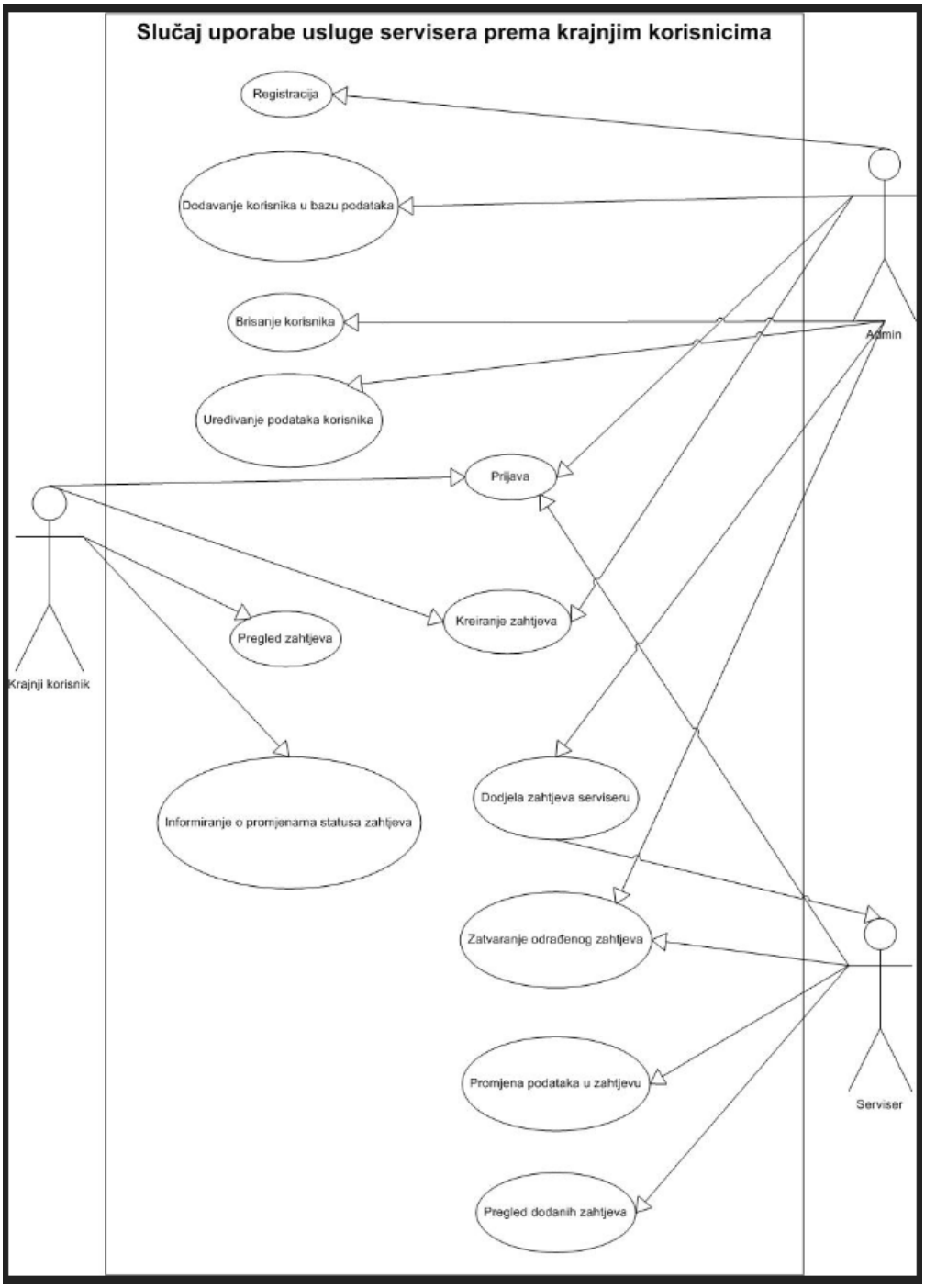

*Slika 1. Simboli i veze kod dijagrama slučaja uporabe*

Slika 1 prikazuje simbole sudionika, opisa slučaja te moguće veze između sudionika i opisa slučaja uporabe. Također između Sudionika 1 i Sudionika 2 upotrjebljena je veza poopćavanja ili nasljeđivanja.

## 3.2. Dijagram slučaja uporabe web aplikacije servisera prema krajnjim korisnicima

Dijagrami slučaja uporabe omogućavaju da se dugotrajni procesi prikupljanja informacija i opisi sustava svedu na minimalni postupak konkretno razvijenog poslovnog procesa. Problemsko područje potrebno je vrlo dobro definirati, da bi se drugi modeli dijagrama, a i sama realizacija procesa mogli pravilno i dobro kreirati, te

imaju svoju svrhu za rješavanje problema. Upravo se od dijagrama slučaja uporabe prvo polazi iz razloga što najbliže opisuje problematiku sustava. Na slici 2 prikazan je slučaj uporabe koji opisuje što sustav radi, odnosno funkcionalnosti web aplikacije servisera prema krajnjim korisnicima.



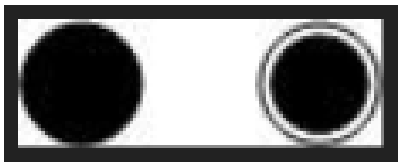

*Slika 2. Dijagram slučaja uporabe aplikacije usluge servisera prema krajnjim korisnicima*

Sudionici u navedenom slučaju su krajnji korisnici, admini i serviseri. Da bi korisnici mogli pristupiti aplikaciji potrebno ih je prethodno registrirati od strane admina, a zatim se trebaju prijaviti u sustav. Admin ima ulogu registracije korisnika u sustav, koji mogu biti krajnji korisnici, serviseri ili novi admini, zatim uređivati i brisati korisnike sustava i kreirati zahtjev u ime krajnjeg korisnika ukoliko sam nije u mogućnosti to učiniti. Također kreirane zahtjeve od strane krajnjih korisnika ili njih samih ima mogućnost dodijeliti zahtjev određenom serviseru i zatvoriti zahtjev po završetku. Krajnji korisnik posjeduje ulogu kreiranja zahtjeva, pregleda istog i informiranje o promjenama statusa zahtjeva u kojem se trenutno nalazi. Nakon što admin dodjeli zahtjev određenom serviseru, serviser odlazi na lokaciju problema i pokušava ga ukloniti. Ukoliko zahtjev bude uspješno riješen serviser zatvara zahtjev, a ukoliko nije u mogućnosti riješiti problem u tom trenutku npr. iz razloga što mu je potreban određen dio kućanskog aparata za popravak, ali ga nema kod sebe, on se dogovara s krajnjim korisnikom za sljedeći termin u kojem su oboje raspoloživi. Serviser ima informaciju u koje vrijeme je slobodan jer ima mogućnost pregleda svih zahtjeva koji su mu dodijeljeni.

# 4. Modeliranje kontrole toka i toka podataka primjenom dijagrama aktivnosti

Dijagram aktivnosti opisuje na koji način su aktivnosti koordinirane kako bi se pružila zamišljena usluga prema krajnjim korisnicima na različitim razinama apstrakcije. Događaj se izvršava u diskretnom vremenskom trenutku prema određenoj operaciji, naročito kada je operacija namijenjena za postizanje različitih koraka koje zahtijevaju upravljanje i koordinaciju zadataka. U slučaju jednostruke uporabe događaji se odnose jedni na druge, osobito kada se koriste slučajevi u kojima se aktivnosti preklapaju. Dijagram aktivnosti predstavlja tijek poslovnog procesa u kojem je potrebno identificirati slučajeve korištenja kroz ispitivanje poslovnih tijeka prije i poslije ispunjavanja uvjeta. U postupku kreiranja dijagrama aktivnosti potrebno je izvršiti modeliranje do detalja na koje se odnosi unutarnji rad objekta, rad objekata prema drugima i računski procesi [11].

Uz pomoć dijagrama aktivnosti se želi prikazati redoslijed izvođenja aktivnosti koje su združene promatranom objektu, te obuhvaća prijelaze, račvanja i spajanja, te grananja i stapanja. Označavanje početnog i završnog stanja prikazuje se krugovima prikazanim na slici 3. Također navedeni dijagram koristi se za modeliranje radnog tijeka poslovnog procesa i opisa rada algoritma ili detalja računanja.



*Slika 3. Početak i kraj dijagrama aktivnosti*

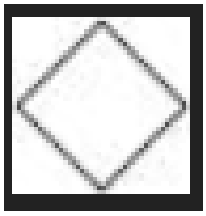
## 4.1. Aktivnosti, akcije i simboli dijagrama aktivnosti

Aktivnosti se sastoje od svog vremena trajanja, dok je akcijska radnja kraćeg trajanja. Aktivnosti su radnje koje potrebno da objekt izvodi neprekidno, također se izvođenje aktivnosti može prekinuti ili rastaviti na više različitih aktivnosti. Što se tiče akcija one su skupina izvršnih računanja koje rezultira promjeni vrijednosti jednog ili više atributa u poslovnom procesu, povratku izračunate vrijednosti ili oba navedena scenarija. Izvođenje akcije nije moguće prekinuti kao u slučaju aktivnosti jer je akcija zapravo neprekinuta aktivnost.

Aktivnosti i akcije označavaju se s istim simbolom elipse, prikazano na slici 4.


*Slika 4. Simboli akcije i aktivnosti*

Slijedeća podjela odnosi se na stapanje ili grananje aktivnosti i akcija koje se općenito izvode jedna iza druge slijedno. Potreba za tim postoji jer se razdvajanje slijeda može dijeliti u dva ili više pravaca pa se stapa i nastavlja izvršavati u jednoj grani što se naziva mjesto grananja (engl. *Branch*) ili postoji mogućnost povezivanja dva ili više pravaca koji se u daljnjem izvođenju stapaju u jednu granu, odnosno mjesto stapanja ( engl. *Merge* ). Na slici 5 je prikazan simbol romba kojim se označava stapanje akcija i aktivnosti [12].



*Slika 5. Simboli račvanja i stapanja*

Da bi se grananje uspjelo izvršiti, po najprije je potrebno ispuniti određene uvjete (engl. *Guard*). Uvjeti su Boole-ovi[5] izrazi s ishodom istina (engl. *True*) ili laž (engl. *False*). Grana će se slijediti samo ako je uvjet ispunjen, a skup svih grana iz mjesta grananja mora uzeti u obzir sve ishode i mogućnosti slijeda da se pritom uvjeti međusobno ne preklapaju. Uvjet se upisuje u uglate zagrade uz granu na koju se uvjet odnosi te strelice koje ulaze i izlaze prikazuju aktivnosti prijelaza ( engl. *Transition* ) i tijek kontrole.

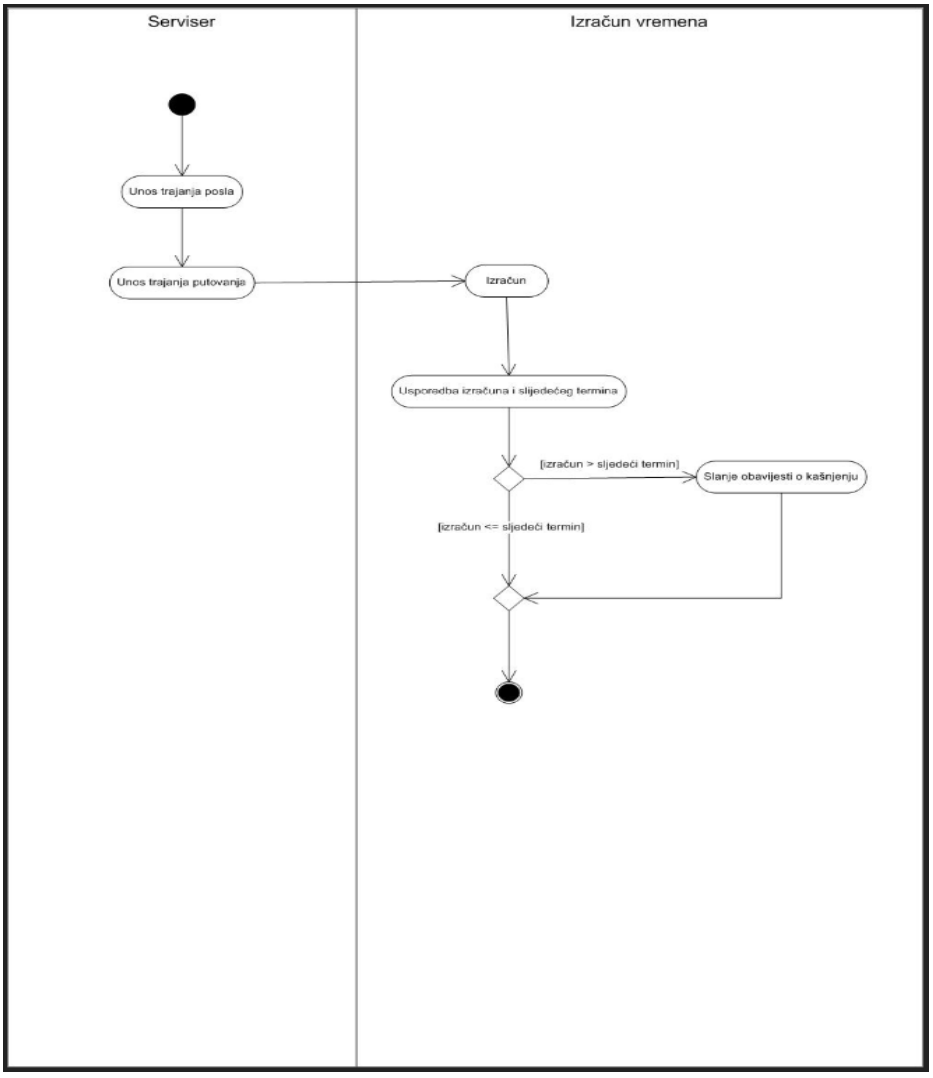
Mjesta spajanja i račvanja označavaju se dugim uskim crnim pravokutnikom koji se naziva sinkronizacijska crta (engl. *Sychonization bar*). Račvanje (engl. *Fork*) je mjesto u kojem se tijek događaja izvođenja razdvaja na dvije ili više nezavisne radnje, odnosno nezavisna i istodobna tijeka. Mjesto spajanja (engl. *Join*) je usklađivanje dva ili više tijeka događaja u jedan. U slučaju da se pojavljuje mjesto račvanja, ono dolazi nakon izvođenja prve aktivnosti ili akcije koje se izvode na paralelnim putanjama s različitim metodama ili ista metoda ako je operacija u navedenoj klasi objekta označena svojstvom istodobnosti. Završetkom svih aktivnosti ili akcija u paraleli pravci se spajaju i objekt nastavlja sa slijednom obradom procesa.

[5] Boole - ova algebra je dio matematičke logike s osnovnim operacijama koje uključuju I, ILI, NE [13].

Ukoliko u poslovnom procesu postoji veći broj različitih aktivnosti i akcija, može se upotrijebiti grupiranje istih na dijagramu radi lakšeg praćenja procesa uz pomoć plivačkih staza (engl. *Swimline*) koje se označavaju crnim okomitim crtama unutar kojih se nalaze dijelovi procesa. U poslovnom procesu objekti utječu jedni na druge. Tijek objekata (engl. *Object Flow*) predstavlja zavisnosti u dijagramu, te detaljno prikazuje utjecaj aktivnosti na njih same. Prikazi vrijednosti stanja se prikazuju u uglatim zagradama koje prikazuje skup vrijednosti atributa u određenom trenutku [14].

## 4.2. Dijagram aktivnosti aplikacije za servisere prema krajnjim korisnicima

Na slici 6 prikazan je dijagram aktivnosti koji se odnosi na računanje vremena nakon unosa parametara od strane servisera kada stigne na lokaciju korisnika koji potražuje rješavanje problema putem zahtjeva koji je kreirao.




*Slika 6. Dijagram aktivnosti slanja obavijesti krajnjem korisniku*

Dijagramom na slici 6 prikazan je tijek aktivnost nakon što je serviser došao na lokaciju te unosi u zahtjev parametre. Dijagram započinje početnim stanjem. Zatim slijedi unos podataka o trajanju posla na lokaciji na kojoj se nalazi i unos trajanja putovanja prema sljedećoj lokaciji od strane objekta „Serviser“. Na temelju parametara, zbraja se vrijeme termina navedenog zahtjeva, trajanja posla i putovanja. Zatim se uspoređuju dobiveni izračun i slijedeći termin. Ukoliko je dobiveni zbroj veći i prelazi termin slijedećeg korisnika što je prikazano u uglatim zagradama na dijagramu, odradit će se akcija generiranja maila, te će biti poslan krajnjem korisniku koje je potencijalno vrijeme dolaska servisera s obzirom da neće stići na dogovoreno vrijeme. Ukoliko je izračun zbroja manji ili jednak sa sljedećim terminom, obavijest se neće slati sljedećem krajnjem korisniku iz razloga što će serviser kod njega stići u terminu koji je korisnik potraživao.

# 5. Modeliranje vrsta, svojstava i stanja objekata

Prije početka kreiranja samog sustava potrebno je definirati objekte, njihova stanja, vrstu i svojstva. S obzirom da većina dijagrama uzima u obzir vremenski pomak, aktivnosti i akcije, potrebno je putem određenog dijagrama proanalizirati dio sustava na način da opisuje strukturu sustava u smislu da je statičan i ne obazire se na vremensku komponentu. Modeliranjem objekata vizualiziraju se entiteti koji u stvarnosti predstavljaju komponentu koja je dobro definirana i dio je sustava. Grupe objekata koje posjeduju slična svojstva, predstavljaju klase u kojima je svaki objekt instanca određene klase. Potaknuto vanjskom aktivnosti klasa odgovara određenim ponašanjem u kojem se trenutno stanju objekt nalazi. Stanje u kojem se objekt nalazi je životni vijek u kojem se nalazi određeno vrijeme, te u tim stanjima može mijenjati ili ispunjavati jedan ili više uvjeta.

## 5.1. Dijagram klasa

Dijagram klasa služi za opisivanje klase, statičnih odnosa međusobnih veza i vrste njihovih objekata unutar određenog sustava. S dijagramom klasa želi se postići opisivanje strukture sustava, ali bez opisivanja stanja, aktivnosti, događaja ili promjenjive karakteristike sustava za koje se odnosi vremenski pomak iz razloga što je navedeni dijagram statičan s obzirom na vremensku komponentu.

Objekti služe za predstavljanje entiteta iz realnog svijeta ili općenito neki koncept nečega što se nalazi u dobro definiranim granicama i ima smisao u sustavu, dok su klase opis grupe objekata sa sličnim svojstvima, u kojima su objekti obavezno pojedinci (instance) pojedine klase. Klase se koriste za opisivanje objekata jer su objekti vrlo usko povezani s klasama, a klase su nestatički dijelovi koji opisuju objekte. Klasama je bitno odrediti:

1. Naziv klase – pojavljuje se u prvoj particiji i generalno označava pojam koji klasa označava.

2. Atribut klase – predstavljaju stanje objekta u klasama, strukturne ili statičke značajke, označavaju definiranje koji objekti klase imaju mogućnost što učiniti i odrediti interakciju između objekata. Vrste atributa prikazuju se iza znaka dvotočke, te se pojavljuju u drugoj particiji. Potrebno im je odrediti naziv i tip podataka.

3. Operacije klasa – su usluge koje klase pružaju. Nalaze se u trećoj particiji. Iza dvotočke se nalazi tip metode koja je vraćena, a također su i vrste parametara prikazane iza istog. Operacijama je potrebno definirati definiciju koja uključuje naziv operacije i sve ulazne i izlazne parametre [11].

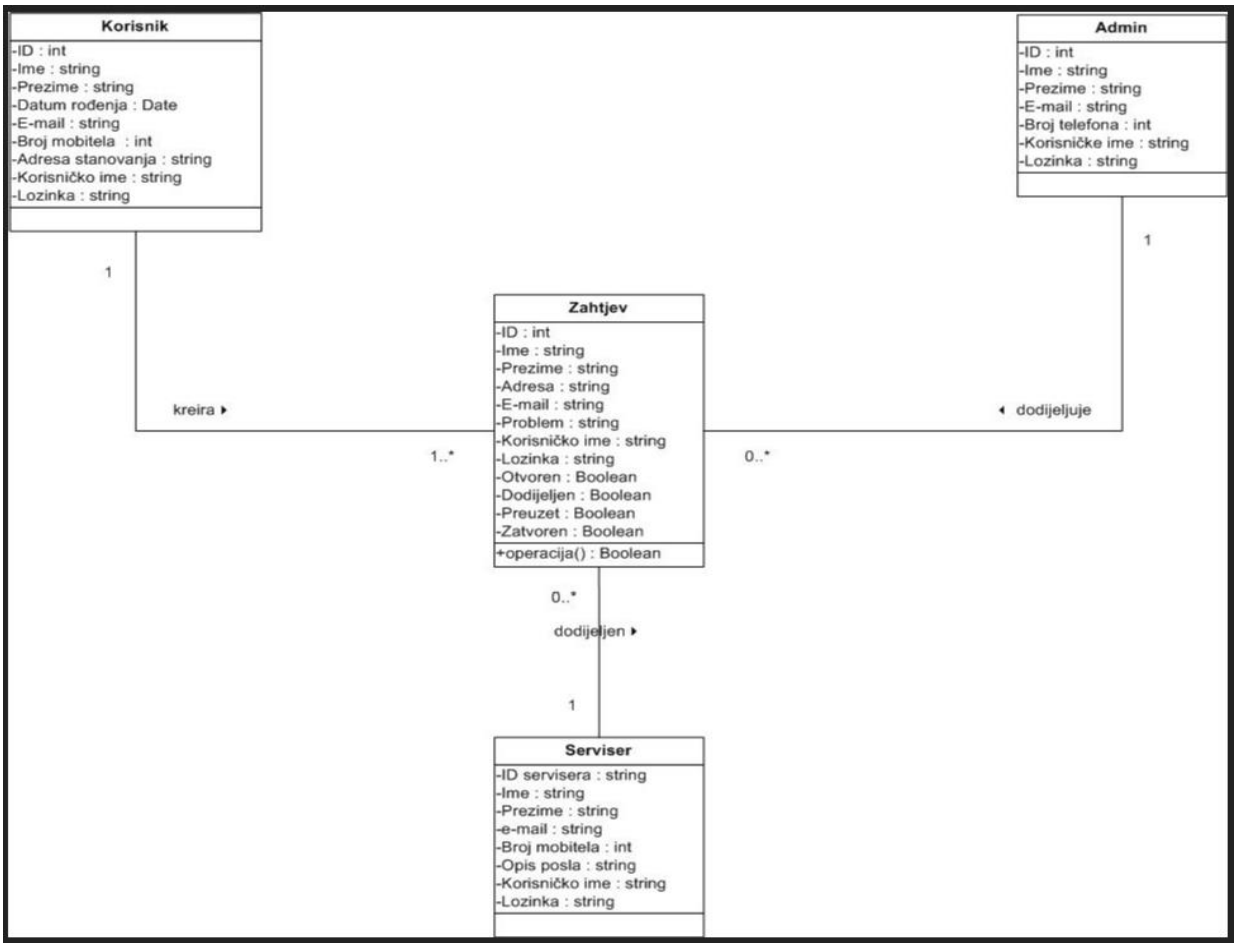
Odnosi između klasa predstavljaju objekte s njihovim javnim svojstvima, zaštićenim privatnim strukturama podataka, hijerarhijama nasljeđivanja i asocijacija, te metodama. Odnosi između klasa su vrlo bitna stavka u kreiranju i čine osnovnu strukturu modela, a postoje tri vrste veza:

1. Relacija udruživanja (engl. *Association*) – opisuje statične odnose između pojedinaca, odnosno instance klasa, a mogu biti jednosmjerne, dvosmjerne ili refleksivne.

2. Relacija sastavljanja (engl. *Agregation*) – postoje dvije vrste sastavljanja, a to su agregacija i kompozicija.

3. Relacija poopćavanja (engl. *Generalization*) – odnosi se na mehanizam nasljeđivanja u međusobno suprotnim smjerovima specijalizacije i generalizacije

Ovisno o udruživanju smjer veza dijeli se na jednosmjerne i dvosmjerne, od kojih jednosmjerne imaju definiran smjer prema samo jednome vrhu, dok se kod dvosmjernih vrh pojavljuje u oba smjera. Ukoliko putanja nije definirana, smatra se nepoznatom vezom ili dvosmjernom, a jednosmjernim vezama vrhovi moraju biti međusobno inverzni. Iako su relacije udruživanja i sastavljanja bidirekcionalne, potrebno je ograničiti putanju u samo jednome smjeru što se označava strelicom u smjeru putanje [15]. Slikom 7 prikazan je dijagram klasa u koje je uključeno kreiranje zahtjeva korisnika od strane krajnjeg korisnika i dodjeljivanje istog serviseru.




*Slika 7. Dijagram klase kreiranja zahtjeva*

Slika 7 prikazuje primjer upotrebe dijagrama klase u kojem je glavni objekt „Zahtjev“. Sporedni objekti u primjeru su „Korisnik“, „Admin“ i „Serviser“. Krajnji korisnik je sudionik u sustavu koji kreira zahtjeve. U dijagramu su također postavljene oznake na krajevima relacija udruživanja koje prikazuju da jedan korisnik može kreirati jedan ili više zahtjeva, dok pojedinačan određen zahtjev može biti kreiran od strane samo jednog korisnika. Admin dodjeljuje zahtjev serviseru. Jednom adminu može biti slučaj da ne dodijeli ni jedan zahtjev serviseru, a može ih dodijeliti neograničen broj koji je otvoren. Jedan specifičan zahtjev može biti dodijeljen od strane samo jednoga admina. Kada se zahtjev nalazi u stanju otvoren admin ga dodjeljuje slobodnom serviseru koji je predviđen za traženi posao. U tom slučaju zahtjev se nalazi u stanju dodijeljen. Jedan serviser može imati nula i više dodijeljenih zahtjeva, dok jedan zahtjev može pripadati samo jednom serviseru.

## 5.2. Dijagram stanja objekata

Dijagram stanja prikazuje ponašanje klasa kao odgovor na vanjske aktivnosti. Opisuje ponašanje jednog objekta kao odgovor na niz događaja koji se izvršavaju u sustavu. Ovaj UML dijagram modelira dinamički tijek kontrole od stanja do stanja određenog objekta unutar poslovnog procesa. Dijagram stanja također prikazuje stvarne promjene u sustavu, ali ne prikazuje procese ili aktivnosti koje su stvorile te promjene. U suštini promatra se unutarnji rad objekata i kroz što prolazi za vrijeme aktivnog stanja unutar sustava, predstavlja pojavu od značaja.

### 5.2.1. Događaji stanja

Događaji imaju svoju podjelu koja se odnosi na aktivnosti koje se izvršavaju, odaziv na događaje ovisi o stanju u kojem se objekt u tom trenu nalazi, a to su:

1. Događaj poziva (engl. *Event Call*)– sinkroni način komunikacije se vrši između objekata, u smislu objekt koji poziva metodu drugog objekata, poziva vlastitu metodu i učinka kao poziva akcije.

2. Vremenski događaj (engl. *Time Event*) – nakon proteka određenog predviđenog vremena izvršava se pomoću riječi „poslije“ navedenog vremenskog pomaka..

3. Događaj promjene ( engl. *Change Event*) – ukoliko se uvjet ispuni, nastupa događaj promjene, koji se izvršava uz pomoć riječi „kada“ nakon koje slijedi Boole-ov izraz.

4. Signal – komuniciranje na asinkroni način između objekata.

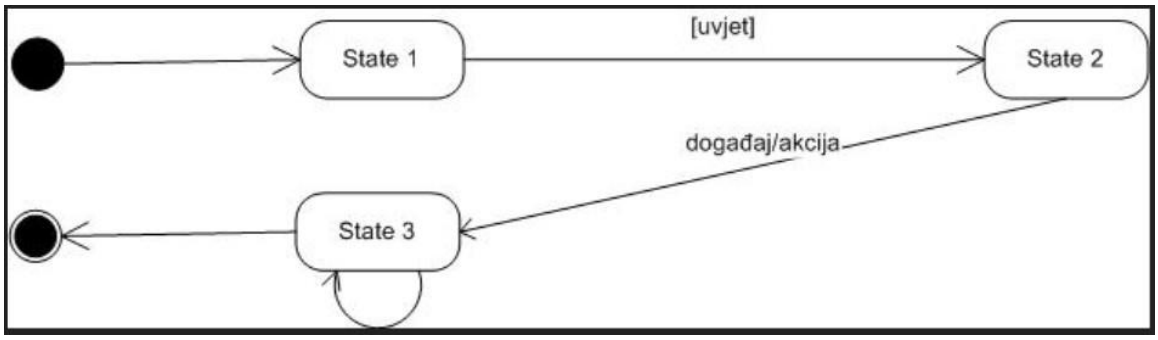
### 5.2.2. Uvjeti čuvanja, stanja i prijelazi

Kod kreiranja UML dijagrama stanja moguća je situacija da se objekt može nalaziti u raznim uvjetima i prelaziti iz jednog u drugi. Stanje u kojem se objekt nalazi je zapravo uvjet životnog vijeka u kojem se objekt nalazi određeno konačno vrijeme, te objekt u tim stanjima može biti aktivan, u stanju čekanja na pojavu konkretnog događaja i ispunjavati jedan ili više uvjeta. Simbol kojim se prikazuju stanja objekata je pravokutnik zaobljenih vrhova [16].

Boole-ovi izrazi označavaju prije prijelaza koji uvjet mora biti ispunjen. Navedeni uvjet se ispisuje u uglate zagrade s navedenim događajem napisanim neposredno ispred samih zagrada. Taj način pisanja odnosi se na događaje uvjeta koji su pridruženi, ali isti oblik pisanja se koristi i za uvjete događaja pridruživanja uz akciju u

kojem se slučaju akcija piše iza naziva događaja, uglatih zagrada i crte ukoso. Ukoliko je ishod uvjeta Booleov izraz laž, tada za oba navedena uvjeta objekt zanemaruje događaj i ne mijenja svoje stanje. U vlastitom prijelazu koriste se uglate zagrade u koje se upisuje čuveni uvjet [17].

Prelaženje objekata iz jednog u drugo, odnosno iz početnog stanja u sljedeće stanje započinje ispunjavanjem uvjeta kada se pojavi određeni događaj zbog kojeg se pokreće prijelaz. Prijelaz može biti i bezuvjetni prelazak u situaciji kada je događaj završio u trenutnom stanju te je spreman prijeći u sljedeće bez određenog „pokretača“ iz razloga što je završila aktivnost početnog stanja. Primjeri simbola koji označavaju početak i završetak, stanje i uvjete prelaska objekata nalaze se na slici 8.

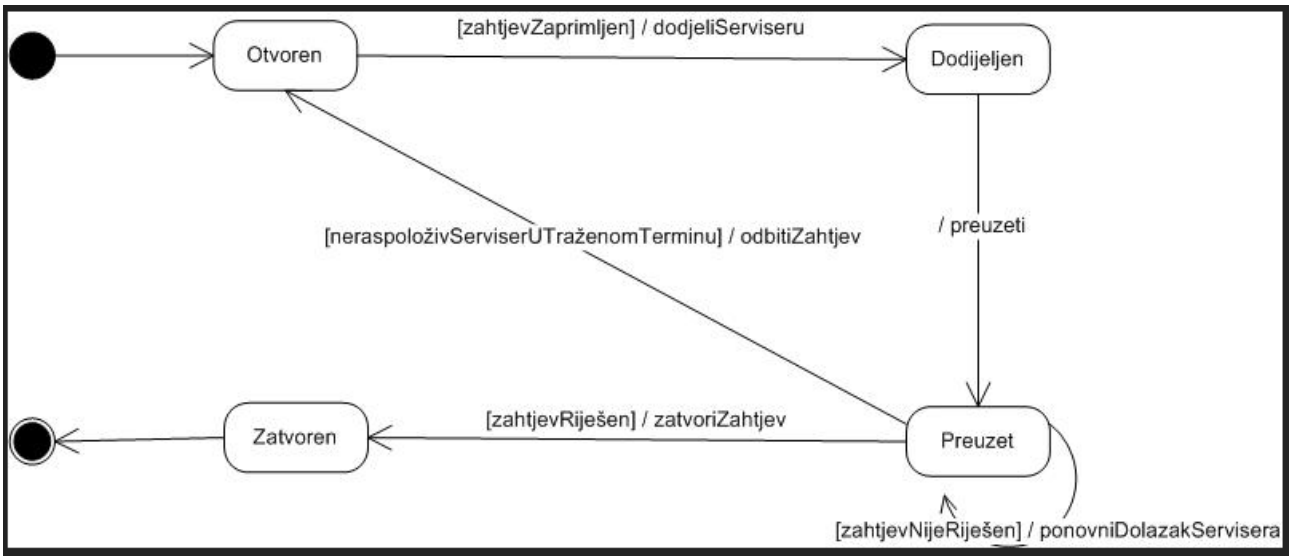



*Slika 8. Simboli, stanja i uvjeti dijagrama stanja*

Ukoliko kod prijelaza objekata iz jednog stanja u drugi postoji uvjet, njemu može biti pridružena i akcija, odnosno ukoliko nema zaštitnih uvjeta, akcija će se provesti bezuvjetno prije nego što objekt prijeđe u drugo odredišno stanje. Ukoliko se izvršava vlastiti prijelaz, kod njega su izvorno i odredišno stanje potpuno jednaki.

## 5.3. Dijagram stanja usluge servisera prema krajnjim korisnicima

Dijagramom stanja prikazuje se ponašanje jednog objekta u odnosu na razne događaje koji se događaju u sustavu. Na slici 9 prikazana su stanja objekta zahtjev u koja, ovisno o raznim događajima i aktivnostima ostalih korisnika sustava, može prijeći.




*Slika 9. Dijagram stanja objekta zahtjeva*

Dijagram započinje i završava točkama početnog i završnog stanja. Nakon početnog stanja nalazi se veza bezuvjetnog prijelaza u kojemu zahtjev koji se promatra kao objekt se nalazi u stanju „Otvoren“. Iz navedenog stanja u stanje „Dodijeljen“ zahtjev prelazi akcijom poziva metode „dodjeliServiseru“ i pojavom događaja da je zahtjev zaprimljen. Sljedeće stanje u koje objekt zahtjev prelazi je „Dodijeljen“. Akcijom poziva metode „preuzeti“ zahtjev prelazi u stanje „Preuzeto“. Ukoliko serviser na lokaciji na koju je stigao ne uspije riješiti problem u jednom dolasku te mijenja parametre ponovnog dolaska u kreiranom zahtjevu, isti ostaje u stanju „Preuzeto“ do ponovnog dolaska servisera, rješavanja problema i zatvaranja zahtjeva. Ako se dogodi slučaj da je admin dodijelio zahtjev serviseru koji nije slobodan u terminu koji je potreban, serviser će zahtjev odbiti, te će zahtjev ponovno prijeći u stanje „Otvoren. Događajem „zahtjevRiješen“ i akcijom „zatvoriZahtjev“, zahtjev prelazi u stanje „Zatvoren“.

# 6. Modeliranje organizacijske strukture sustava

Kada je rad sa dokumentacijom koja je vezana uz željeni sustav za projektiranje preopširan i kompleksan UML dijagrami komponenata i rasporeda dijele sustav na manje komponente. Vrlo često se događa situacija da je arhitektura sustava previše kompliciranа, te iz tog razloga navedeni dijagrami obuhvaćaju više različitih područja ili im pridružuju drugačije tehnologije. Dijagrami komponenata i rasporeda prikazuju strukturu programskog koda i implementaciju u realnom vremenu.

## 6.1. Dijagram komponenata i rasporeda

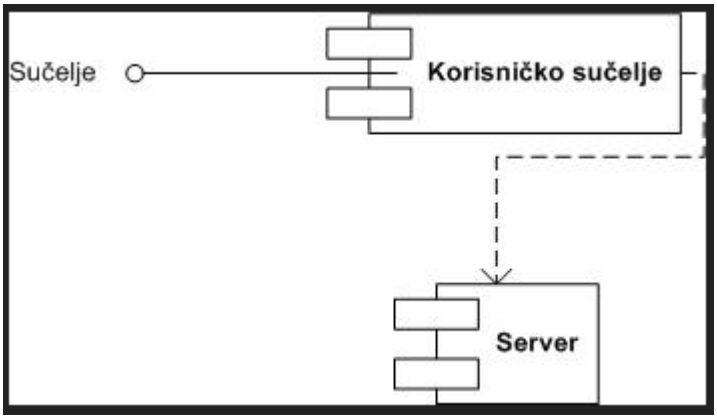
Komponente predstavljaju pakete logičkih apstrakcija, odnosno klasa u implementaciji koje mogu imati atribute i akcije, a komponente samo akcije do kojih je moguće pristupiti samo uz pomoć sučelja. Klase mogu izložiti svoje atribute nižim klasama i svim vanjskim objektima što se definira kao javna vidljivost, a nižim klasama kao zaštićena vidljivost. Fizički su zamjenjivi dijelovi sustava i prilagođavaju se skupu sučelja. Mehanizam grupiranja informacija osim komponenata je paket koji predstavlja grupiranje elemenata modela i fizičko grupiranje komponente sučelja. Također razvoj temeljen na komponentama temelji se na pretpostavkama da se ranije konstruirane komponente mogu ponovno upotrijebiti ili zamijeniti sa drugim ekvivalentima u slučaju potrebe.

Cilj dijagrama komponenata je prikaz ovisnosti programskih komponenata i fizičke strukture koda uključenog u kodnim komponentama koje se nalaze u vremenu provođenja, povezivanja i izvršavanja. Dijagram komponenata želi omogućiti prikaz povezanih komponenata u zavisnim vezama radi omogućavanja lakše analize reakcije svih na promjenu u jednoj komponenti. Također se koristi za vizualizaciju organizacije poslovnog procesa i povezanosti komponenata u sustavu, kao i izradu izvršnih krajnjih sustava. Komponente mogu predstavljati logičku i fizičku strukturu u UML dijagramu. Logičke komponente bi bile poslovne jedinice i komponente procesa. Čvorovi i rubovi najčešće su prikazani u dijagramu komponenata kao sučelje koje može biti predviđeno, potrebno, zatim se prikazuju klase, portovi, konektori, ovisnosti i upotreba navedenih segmenata poslovnog procesa.

Dijagram rasporeda prikazuje vizualnu topologiju fizičkih komponenti, na koji način su softverske komponente raspoređene u sustavu za kvalitetno funkcioniranje poslovnog procesa. Koristi se za statički raspored komponenti. U suštini prikazuje čvorove, njihove ovisnosti i međusobna združivanja pod kojim se smatra fizička povezanost. Projektant je osoba koja ih označuje pomoću stereotipa. Fizički dijagrami su ti koji povezuju i uključuju dijagrame komponenata i rasporeda. Nakon navedenih ujedinjenja u cjelinu, modeli koje predstavljaju dijagram komponenata i rasporeda su modeli ostvarenja [11].

## 6.2. Dijagram komponenata web aplikacije servisera prema krajnjim korisnicima

Dijagramom komponenata prikazuju se programske komponente koje se nalaze u pozadini sustava i putem kojih web aplikacija usluge servisera prema krajnjim korisnicima funkcionira. Na slici 10 prikazan je sustav web aplikacije.




*Slika 10. Dijagram komponenta web aplikacije*

U primjeru na slici 10 sučelje predstavlja terminalni uređaj dostupan korisniku putem kojeg pristupa korisničkom sučelju na web aplikaciji, a web aplikacija sadrži bazu podataka i sve svoje funkcionalnosti putem kojih izvršava svoj zadatak pristupom na server.

# 7. Modeliranje interakcije između sudionika u procesu upravljanja zahtjevima za servisom

Za izradu početnog modela zajedničkog rada objekata i preciznijeg modeliranog poslovnog procesa koriste se statičke i dinamičke značajke i analiza sustava. Analiza sustava se postiže pregledom teksta koji opisuje slučajeve uporabe, određivanja objekata i podjelom objekata u klase na temelju zajedničkih svojstva. Da bi se aplikacija razvila, potrebno je pratiti ujedinjeni proces razvoja koji se sastoji od 4 faze koje prolaze kroz 5 koraka radnih tijekova koje je potrebno poštovati za dobro organiziranu strukturu poslovnog procesa. Spomenute 4 faze od kojih se razvoj aplikacije sastoji su početna faza, faza razrade i izrade, te faza prijelaza. Radni tijek analize sustava poslovnog procesa kreće početnom fazom u kojoj se sustav počinje graditi da se prikažu zahtjevi visoke razine. Zatim slijedi faza razrade i izrade u kojima se konkretni zahtjevi raščlanjuju, usavršavaju i strukturiraju, te na temelju navedenog izgrađuje se model analize u dijelovima koji nisu striktno navedeni u fazi razrade. U fazi prijelaza izvršavaju se zadnje korekcije za finalnu verziju sustava.

Na početku kreiranja dijagrama međudjelovanja potrebno je napraviti analizu klasa koje postavljaju početni model sustava i poslovnog procesa, na način da se dodjeljuju postupci klasama koje mogu biti rubne klase, klase entiteta i upravljačke klase. Objekti su ti koji opisuju klase, a predstavljaju instance navedenih klasa.

Prvi simbol koji se koristi je objekt pomoću kojeg se pristupa u slučaju uporabe, odnosno sudionik pristupa sustavu preko njega. Rubni objekt odgovara imenicama u tekstu. Sljedeći objekt je onaj koji sadrži informacije koje mogu biti promjenjive ili trajne. Kada je riječ o povezivanju s bazom podataka, tada se radi o trajnijim informacijama, dok se primjerice rezultati pretrage promatraju kao privremene informacije. Također, navedeni objekt odgovara imenicama u tekstu opisa. Postoji još jedna vrsta koja se odnosi na objekte u dijagramima međudjelovanja, a to su upravljački objekti. Oni se odnose na glagole u tekstu opisa slučaja uporabe i obuhvaća cjelokupnu logiku, aktivnosti sustava. Upravljački objekt koordinira i upravlja redoslijedom kojim će se što izvoditi, povezuje rubne objekte i objekte jedinke.

U dijagramima međudjelovanja mogu se pojaviti osnovni, zamjenski, ali i iznimni tijek događaja. Izrada za osnovni tijek sastoji se od opisivanja ponašanja objekata u opisu teksta slučaja uporabe gdje je potrebno analizirati i dodijeliti određenim imenicama i glagolima njihove pripadajuće objekte, poput rubnih, zasebnih i upravljačkih. Ukoliko postoje drugi tijekovi događaja osim osnovnog, potrebno ih je posebno naznačiti. Također je cilj vizualizirati objekte i odnose radi lakšeg shvaćanja funkcionalnosti sustava. Ovaj dijagram je u jednu ruku poseban jer postoje pravila dozvoljene komunikacije između objekata koji se nalaze u njemu. Rubni objekti ne smiju međusobno komunicirati iz kojeg je razloga osmišljen upravljački objekt i kroz sljedeće faze koje se odvijaju u modeliranju upravljački objekti se prevode u metode drugih objekata. Objekti se povezuju strelicama koje mogu biti jednosmjerne i

dvosmjerne. Jednosmjerna strelica označava da se sa navedenog objekta aktivnost prenosi na sljedeći, dok dvosmjerna strelica označava da prethodni objekt očekuje povratnu informaciju na poslanu aktivnost [11].

Između objekata treba naznačiti poruke i akcije koje je potrebno izvršiti. Poruke služe za međusobnu razmjenu informacija i unutar samog objekta, koje najčešće rezultira određenom akcijom ili više njih. UML dijagrami podržavaju 5 vrsta akcija. Akcija se ne izvršava bez naredbe i tada stvara promjene vrijednosti atributa objekata, vraća povratne vrijednosti prethodnom objektu koji je uputio poruku i rezultira općenito promjenama i povratom vrijednosti. Prethodno navedena akcija poziva (engl. *Call Action*) je prizivanje na izvršavanja akcije slijedećeg ili odredišnog objekta i može biti sinkrona akcija u kojem slučaju može čekati povratnu informaciju primatelja za kojeg očekuje da je spreman na informaciju koju je uputio i objekt može pozivati sam sebe.

U slučaju akcije povrata (engl. *Return Action*) objekt očekuje povratnu vrijednost odredišnog objekta na poslanu akciju poziva.

Sljedeće dvije akcije koje se koriste u dijagramima međudjelovanja su tvorbe ( engl. *Create Action*) i dokidanja (engl. *Destroy Action*). Akcija tvorbe ima svoju specifičnost što može kreirati novi objekt, odnosno klasa svoju instancu. Dok akcija dokidanja ima mogućnost uništavanja nepotrebnih objekata u sustavu nastavkom izvršavanja akcija. Akciju dokidanja objekt može izvesti i sam na sebi.

Zadnja akcija od 5 je akcija odašiljanja (engl. *Send Action*). Sastoji se od signala koji pruža jednosmjerne komunikacije između objekata, odnosno u ovom slučaju kada pošiljatelj pošalje poruku ili signal odredištu, ne očekuje povratnu informaciju. Signali se označavaju pomoću atributa signala koji služi kao parametar, te označava smjer odašiljanja signala između objekata. Dodatni odjeljak klase sadrži nazive signala na koje objekt može odgovoriti.

## 7.1. Dijagram međudjelovanja

Dijagram međudjelovanja (engl. *Interaction Sequence Diagram*) koncentrira se na vremenski slijed razmjene poruka između objekata i povezan je za prethodno navedenim planom izvedbe u skupnom procesu razvoja programskog sustava. Služi kao pomoć projektnom timu za dodjelu operacija nad klasama na temelju metoda koje su dodijeljene objektima.

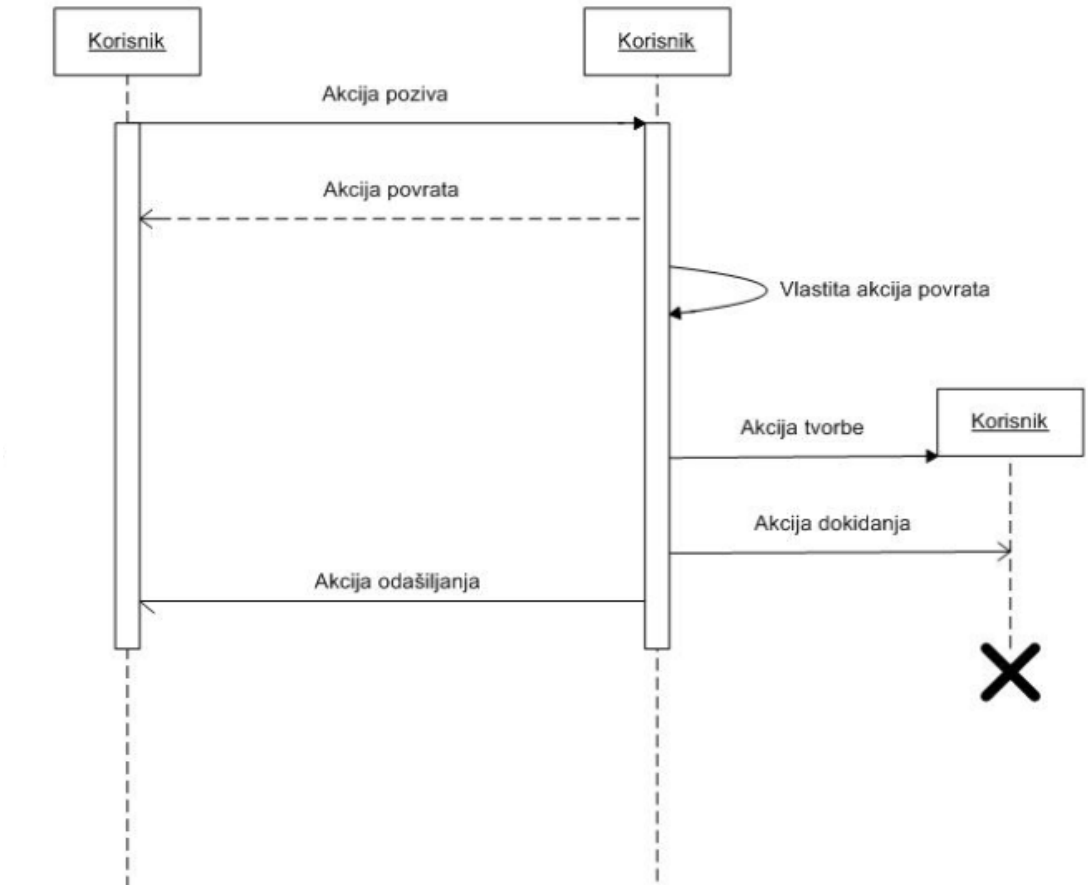
Označavanje elemenata u dijagramu sastoji se od 4 dijela:

1. Oznake objekta – smješta se na vrhu dijagrama.

2. Crte života – prikazuju trajanje života objekata i dokinuće ili smrt, ako se na crti života nalazi oznaka X koja označava trajnost objekata.

3. Težišta nadzora – označava pravokutnik koji se smješta na crtu života i prikazuje vremenski tijek za koji objekt vrši nadzor.

4. Poruke – prethodno objašnjeno, prikazuju koje se akcije između objekata ili objekta samim sa sobom izvršavaju [18].

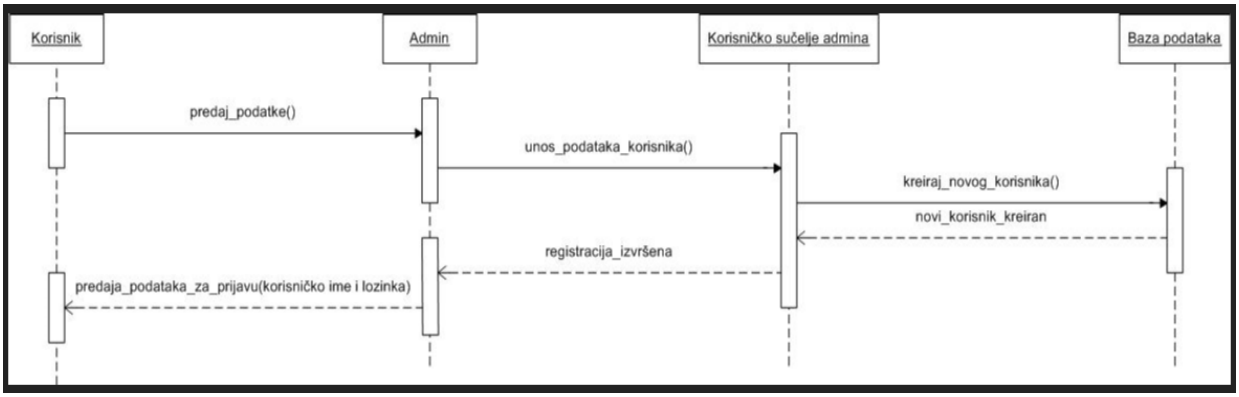



*Slika 11. Simboli, oznake i akcije koje se koriste u dijagramu međudjelovanja*

Na slici 11 prikazani su svi prethodno navedeni i objašnjeni elementi dijagrama međudjelovanja. Simboli koji se koriste su: pravokutnici koji označavaju objekte u dijagramu, crta života i životni vijek koji se nalazi na slici, a zatim slijede svih 5 akcija koje se koriste.

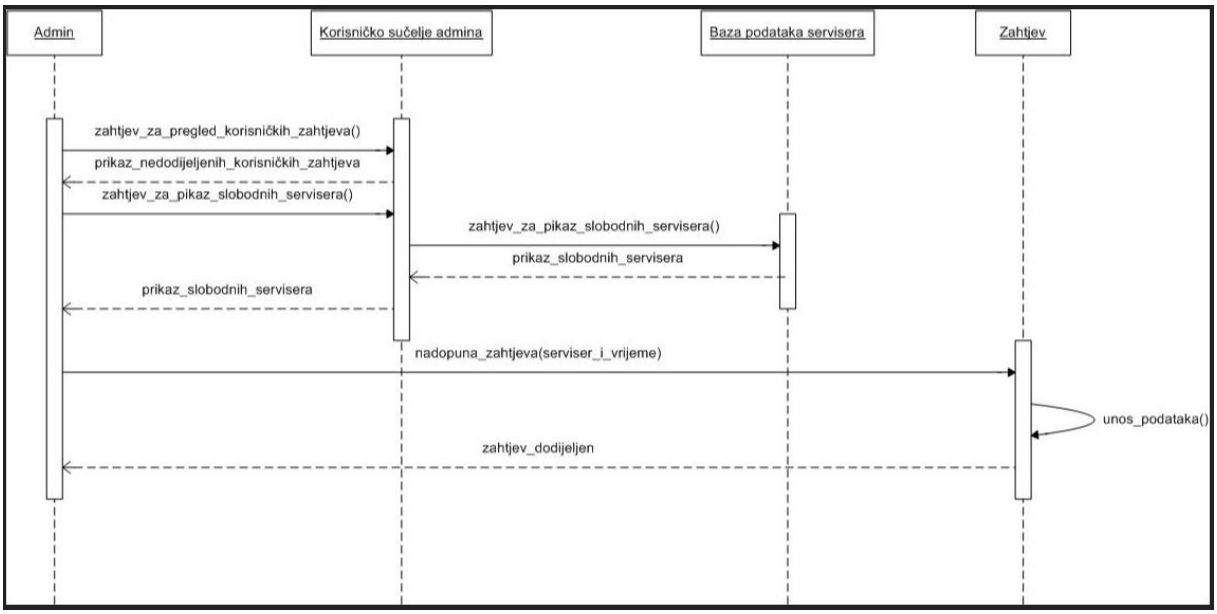
## 7.2. Primjeri dijagrama međudjelovanja

Na slici 12 prikazan je dijagram međudjelovanja u kojem je opisan vremenski slijed razmjene poruka između objekata potrebnih da bi se korisnik registrirao.




*Slika 12. Dijagram međudjelovanja registracije korisnika u sustav*

Kako bi se korisnik mogao prijaviti u sustav, prethodno je potrebno izvršiti registraciju korisnika i dodijeliti mu ulogu putem koje može izvršavati svoje zadatke. Uloge koje je moguće dodijeliti su: admin, korisnik (što se odnosi na krajnjeg korisnika izrađene aplikacije) i serviser, koji izvršava rješavanje problematike kod krajnjih korisnika. Krajnji korisnik započinje razmjenom poruka na način da adminu predaje svoje podatke koje on neposredno nakon primanja unosi u korisničko sučelje za dodavanje novog korisnika. Korisničko sučelje šalje sve potrebne podatke prema bazi podataka koja kreira novog korisnika, te mu kreira korisničko ime i lozinku. Korisničko sučelje admina, podatke koje je kreirala baza podataka, vizualno prikazuje adminu, a admin ih prosljeđuje krajnjem korisniku. S tim podacima krajnjem korisniku je omogućena prijava u sustav.

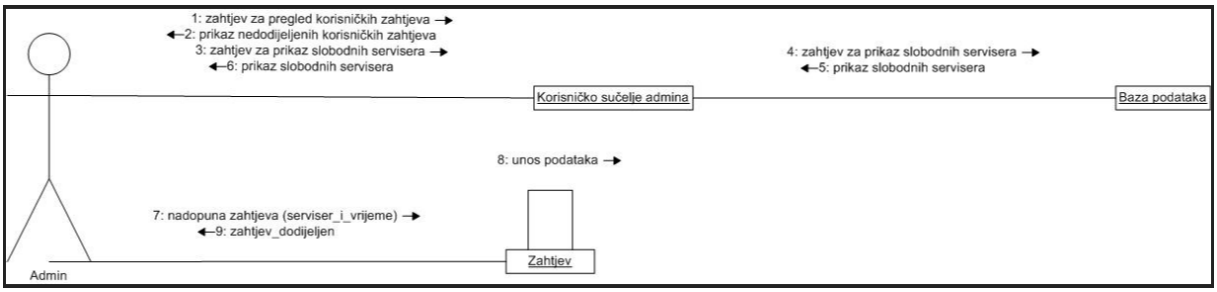



*Slika 13. Dijagram međudjelovanja dodijele zahtjeva*

Slikom 13 prikazan je dijagram međudjelovanja dodjeljivanje zahtjeva serviseru. Sudionici, odnosno objekti su „Admin“, „Korisničko sučelje admina“, „Baza podataka servisera“ i „Zahtjev“. Nakon što je krajnji korisnik kreirao zahtjev, admin ga pregleda putem korisničkog sučelja za admine. Korisničko sučelje mu prikaže sve zahtjeve koji nisu dodijeljeni ni jednom serviseru. Ponovno putem korisničkog sučelja admin potražuje slobodne servisere kako bi mogao dodijeliti prikladnog za traženi problem iz zahtjeva. Od baze podataka, korisničko sučelje potražuje sve slobodne servisere, a nakon dobivenog odgovora prikazuje ih adminu. Pregledom slobodnih servisera, admin izabire određenog i zatim popunjava zahtjev s vremenom i serviserom. Zahtjev unosi podatke koje je admin popunio i time dodjeljuje zahtjev odabranom serviseru, a adminu šalje povratnu poruku da je zahtjev dodijeljen.

## 7.3. Dijagram suradnje

Dijagram suradnje je usmjeren na strukturalnu povezanost objekata. Pokazuje interakcije između objekata i slijed razmijenjenih poruka. Ukoliko postoji manji broj objekata, idealan je za prikaz njihovih aktivnosti. Za razliku od dijagrama međudjelovanja, dijagram suradnje ne prikazuje težište nadzora i crtu života, ali zajedničko im je što prikazuju objekte i poruke koje isti razmjenjuju. Poruke se označavaju brojčano, kako bi bilo lakše pratiti redoslijed komunikacije. Radom objekata zajedno omogućuje se funkcionalnost više razine koju ljudima kojima je potrebno mogu iskoristiti u svrhu programiranja sustava [19]. Na slici 14 prikazan je dijagram suradnje dodijele zahtjeva serviseru od strane admina.




*Slika 14. Dijagram suradnje dodijele zahtjeva serviseru*

Admin putem korisničkog sučelja admina zahtjeva pregled otvorenih i ne dodijeljenih zahtjeva krajnjih korisnika. Nakon prikaza ne dodijeljenih zahtjeva ponovno od korisničkog sučelja potražuje prikaz svih slobodnih servisera. Korisničko sučelje upit prosljeđuje prema bazi podataka koja odgovora popisom i potom bude proslijeđeno prema adminu. Kada je admin dobio uvid u slobodne servisere i izabrao podobnog za rješavanje problema navedenog u zahtjevu, popunjava zahtjev dodatnim parametrima. Nakon unosa parametara servisera i vremena, zahtjev šalje povratnu informaciju da je zahtjev popunjen, odnosno dodijeljen serviseru.

# 8. Web aplikacija za upravljanje zahtjevima za servisom u servisnom centru

Temeljem modela procesa, objašnjenog pomoću UML dijagrama, kreirana je aplikacija koja prati UML-om opisan proces. Za izradu iste, odabran je PHP programski jezik i Microsoft SQL server baza podataka.

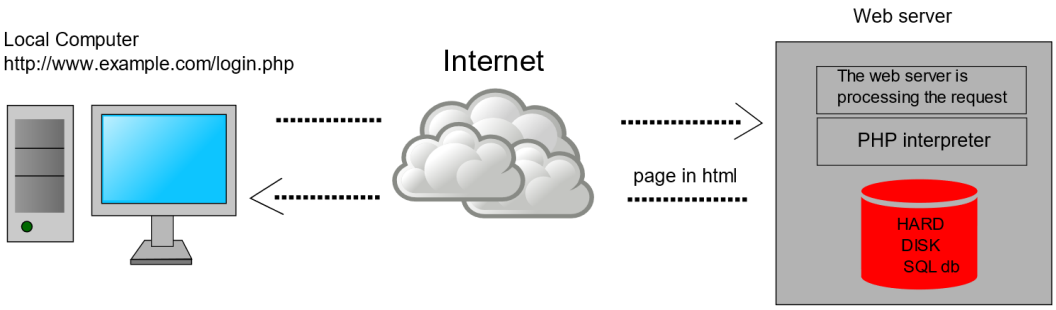
## 8.1. Programski jezik i pohrana podataka unutar aplikacije

Za izradu navedene web aplikacije korišten je programski jezik PHP (engl. *Hypertext preprocessor*). PHP je korišten u kombinaciji s HTML-om (engl. *HyperText Markup Language*), JavaScript-om te jQuery-em[6]. Sve navedeno su web bazirane tehnologije koje se koriste za prikaz i izvršenje web aplikacije. HTML služi za prikaz web aplikacije dok JavaScript i jQuery u kombinaciji s PHP-om služi za izvršavanje određenih radnji. Podatci koji se koriste u sustavu potrebno je i pohraniti, a za pohranu će se koristiti baza podataka Microsoft SQL Server. Koristiti će se besplatna Express verzija baze.

### 8.1.1. Programski jezik PHP

PHP jezik je poslužiteljski orijentiran web baziran jezik koji se koristi za razvoj web aplikacija. Sintaksa je slična C programskom jeziku. Riječ je o besplatnom softwareu otvorenog koda. Najčešće se koristi za izradu dinamičkih web aplikacija, ali se može koristiti i za izradu klasičnih web stranica. Za izradu se koristi u kombinaciji s nekim drugim skriptnim jezicima, a u ovom slučaju bit će korišten s JavaScript ekstenzijom jQuery. Aplikacija koja je napisana u PHP izvršava se pozivanjem linka. Po pozivanju linka server odradi upit te krajnjem korisniku vrati prikaz u obliku HTML-a [20].Na slici 15 prikazana je komunikacija između krajnjeg korisnika i servera na kojemu se nalazi aplikacija.

[6] jQuery - je biblioteka Javascripta koja je kreirana za lakše upravljanje HTML baziranom web stranicom




*Slika 15. Pokretanje PHP aplikacije [21]*

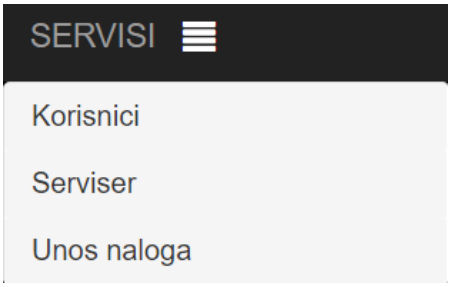
### 8.1.2. Microsoft SQL Server baza podataka

Microsoft SQL Server je relacijska baza podataka razvijena od strane Microsofta i temelji se na SQL programskim upitima prema bazi. SQL (engl. *Structured Query Language*) je programski jezik koji se koristi za dohvat i obradu podataka unutar relacijskih baza podataka, te omogućuje vrlo jednostavno i razvojno okruženje aplikacija koje sadrže bazu podataka. Uz Microsoft koriste ga i druge baze podataka za obradu istih. Neke od njih su Oracle, MySQL, PostgreSQL i mnogi [22] .

Kao i PHP Microsoft SQL Server je cross-platform baza podataka koju je moguće koristiti na više operativnih sustava. Trenutno su aktualne verzije za Windows operativni sustav kao i za Linux.

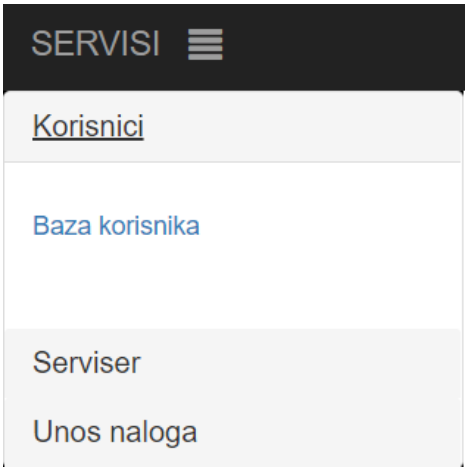
## 8.2. Izvedba aplikacije za nadzor usluga

Web aplikacija Servis sastoji se od tri cjeline, te se svaka cjelina dijeli na podcjelinu. Također na web aplikaciji postoje i tri uloge admin, serviser i korisnik. Slika 16 prikazuje cjeline od kojih se web aplikacija sastoji, a to su korisnici, serviser i stavka unos naloga.



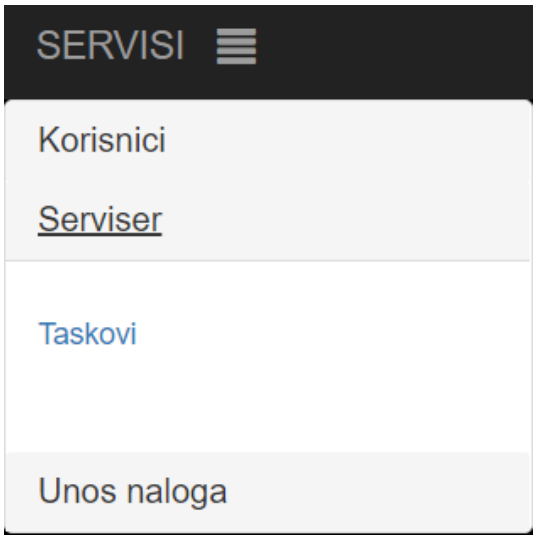

*Slika 16. Glavne cjeline aplikacije*

Slikama 17,18 i 19 prikazane su podcjeline kojima je moguće upravljati uz određenu pridruženu ulogu, ovisnu o kakvom korisniku portala se radi.




*Slika 17. Podcjelina "Korisnici" web aplikacije*

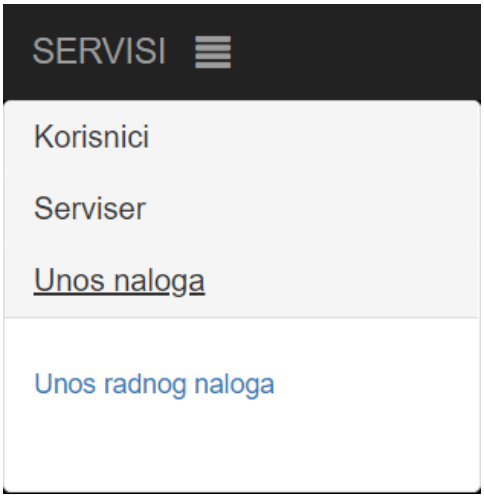
Dodijeljenom admin ulogom moguće je upravljati svim cjelinama web aplikacije. Admin ima pravo dodavati korisnike u bazu podataka, dodavati ili uklanjati operatere koji izvršavaju određeni posao, te mogu unositi nove naloge i brisati obavljene. Baza podataka s korisnicima je pohranjena u pod cjelini „Baza podataka“.




*Slika 18. Podcjelina "Operator" web aplikacije*

Serviseru je dodijeljena uloga s kojom ima mogućnost zatvoriti nalog po završetku i unositi podatke ukoliko postoji izmjena u zahtjevu od strane korisnika, ali

za razliku od admin role nema mogućnost dodavanja novih korisnika u bazu podataka ili brisanje istih koji imaju pristup portalu. Mogućnosti s kojima se serviser služi je u pod cjelini web aplikacije „Taskovi[7]“.



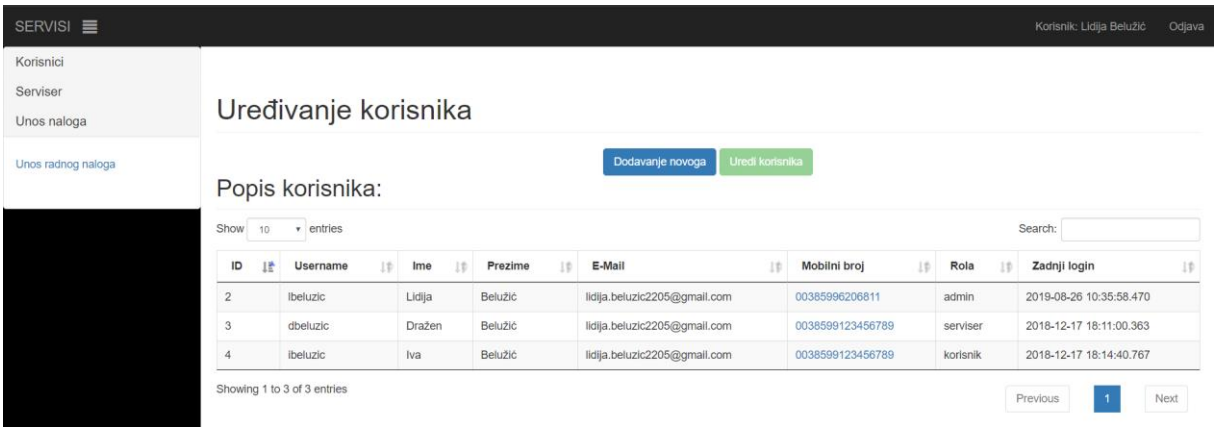

*Slika 19. Podcjelina "Unos naloga" web aplikacije*

Uloga korisnika omogućuje isključivo dodavanje novog naloga kojeg je potrebno izvršiti, te ga u većini slučaja dodaje sam krajnji korisnik kojemu je potrebna određena pomoć u kućanstvu. Korisnik mora odbrati cjelinu „Unos naloga“ u kojoj se nalazi podcjelina „Unos radnog naloga“ u koju unosi određene zahtjeve i odabire opcije između onih koje su mu ponuđene.

### 8.2.1. Upravljanje korisničkim računima

Kao i svaka aplikacija, web aplikacija serviseri ima svoju bazu korisnika. Postoje tri vrste korisnika admin, serviser i korisnik. Tip korisnika koji ima admin ulogu može upravljati korisničkim računima. U navedenoj formi moguće je dodavati korisnika, brisati ili uređivati. Prikaz navedenog segmenta prikazan je na slici 20.

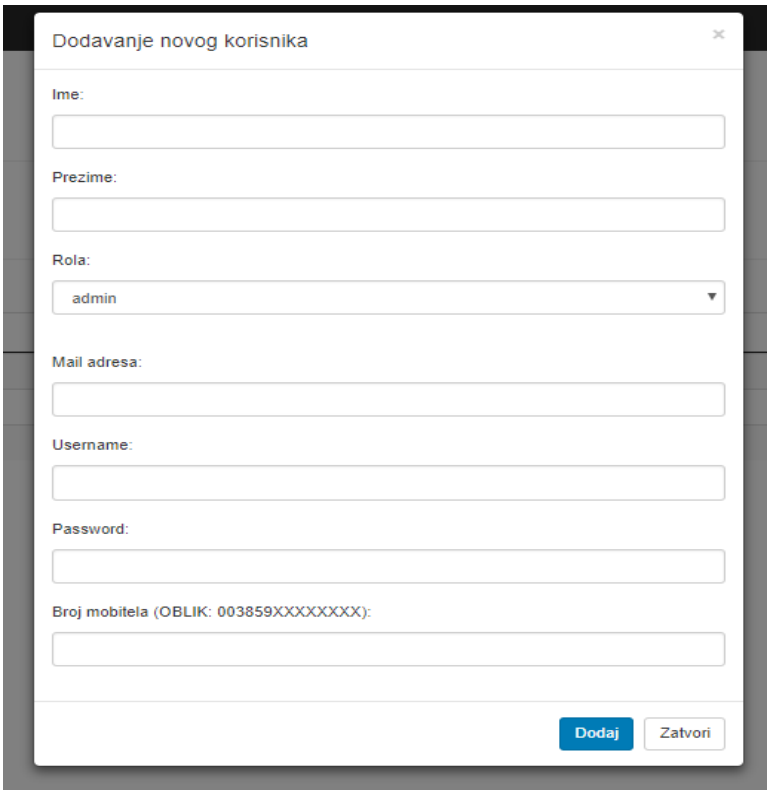
[7] engl. *Taskovi* – zahtjevi koje je potrebno izvršiti od strane servisera




*Slika 20. Sučelje za dodavanje korisnika web aplikacije*

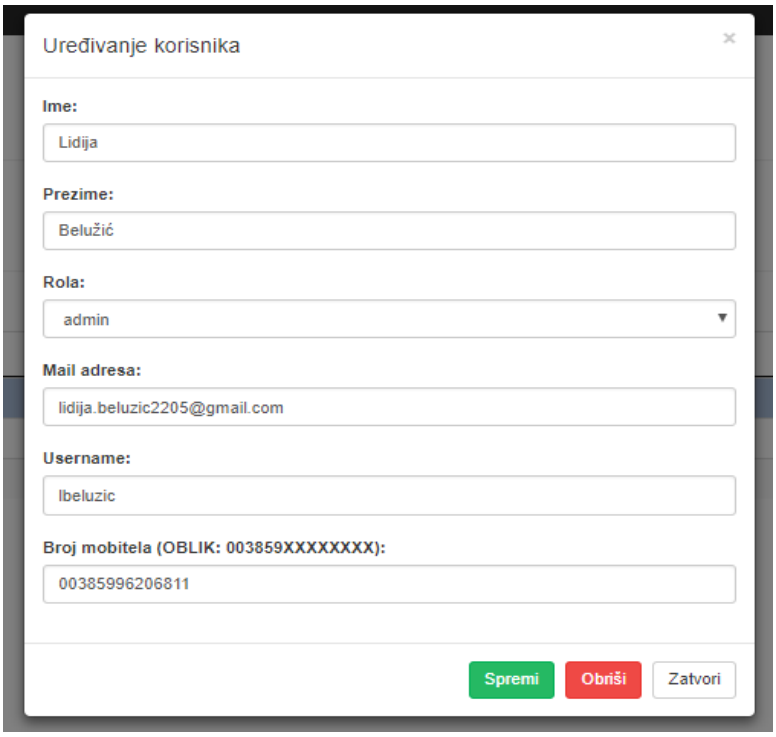
Moguće je dodati neograničen broj korisnika, neovisno radi li se o upravljačkim, serviserskim ili korisničkim ulogama. Dodavanje korisnika u bazu podataka web aplikacije izvršava se klikom na gumb „Dodavanje novog“. U slučaju dodavanje novog korisnika potrebno je unijeti njegovo ime, prezime, dodijeliti ulogu iz padajućeg izbornika uz pomoć koje će imati dodijeljena prava za rad nad web aplikacijom. Nadalje potrebno je navesti mail adresu korisnika kojeg se dodaje, korisničko ime (engl. *User*[8]) i lozinku kojim će se prijaviti u aplikaciju.

[8] engl. *User* – korisničko ime koje se koristi kao oznaka korisnika unutar programa radi autentifikacije određene osobe




*Slika 21. Skočni prozor "Dodavanje novog"*

Slikom 21 prikazan je skočni prozor pomoću kojega osoba koja ima upravljačka prava dodaje novog korisnika te mora unijeti sve navedene elemente koji se traže, kao i po završetku unosa podataka potvrditi sa klikom na gumb „Dodaj“ ili zatvoriti isti.

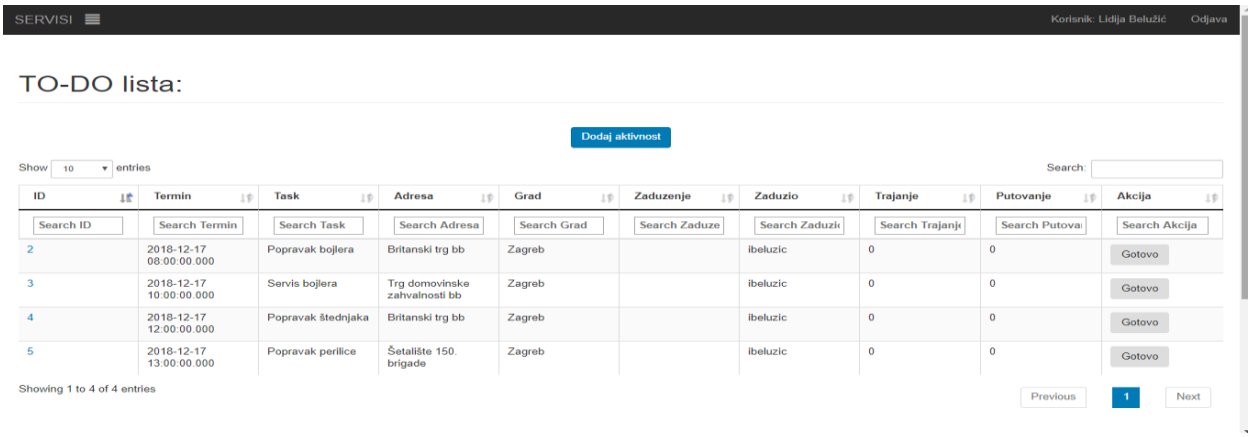



*Slika 22. Skočni prozor za uređivanje korisnika*

Slikom 22 prikazan je skočni prozor uz pomoć kojega upravljačka osoba s admin ulogom ima mogućnost urediti, odnosno promijeniti podatke za pojedinog korisnika. Korisnicima je moguće promijeniti bilo koji od navedenih elemenata koji su ponuđeni, kao i uloga s kojom se trenutno korisnik koristi. Po završetku uređivanja potrebno je spremiti promjene koje su izvršene, moguće je klikom na gumb „Obriši“ potpuno ga izbrisati ili ne izvršiti ni jednu od spomenutih radnji, već samo zatvoriti skočni prozor za uređivanje.

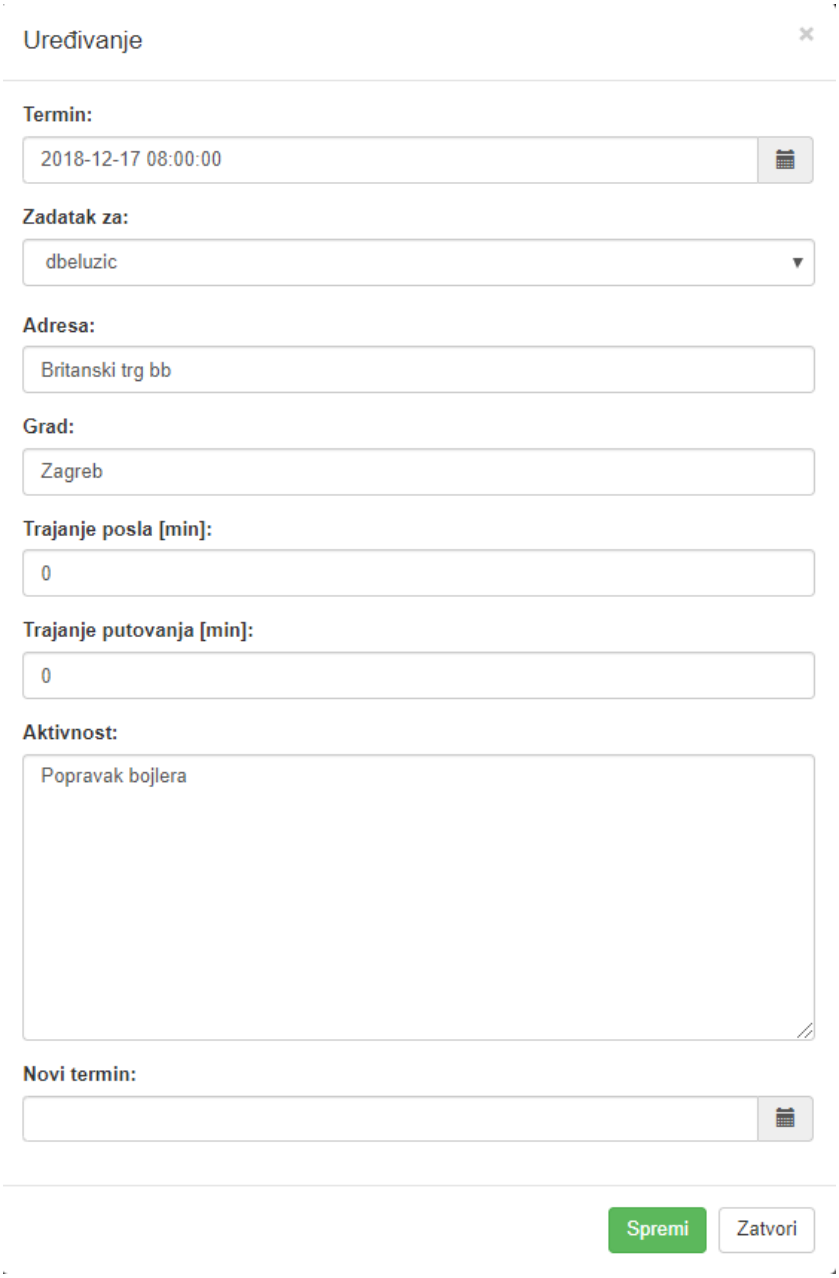
### 8.2.2. Forma za admine i servisere

Opcije kojima se koriste serviseri, a imaju mogućnost i osobe sa admin ulogama, se nalaze u cjelini „Serviseri“ koja još sadrži i pod cjelinu „Taskovi“. Na slici 23 prikazano je početno sučelje kojim se koriste serviseri.




*Slika 23. Sučelje "Taskovi"*

Servisersko sučelje se sastoji od ID-a, odnosno rednog broja zadataka koji je potrebno izvršiti, u kojem vremenu i kojeg datuma je potrebno odraditi, zadataka koji se potražuju, na kojoj adresi i gradu u kojem se nalazi. Kolona „Zaduzenje“ se odnosi na osobu na koju je zadatak opredijeljen, odnosno koji serviser ga treba odraditi, a kolona „Zaduzio“ prikazuje osobu koja je dodijelila zadatak serviseru. Pod „Trajanje“ se podrazumijeva potrebno vrijeme serviseru da odradi potrebnu radnju kod korisnika, koja se nalazi u koloni „Task“, te po završetku unosi potrebno vrijeme za dolazak kod sljedećeg korisnika.

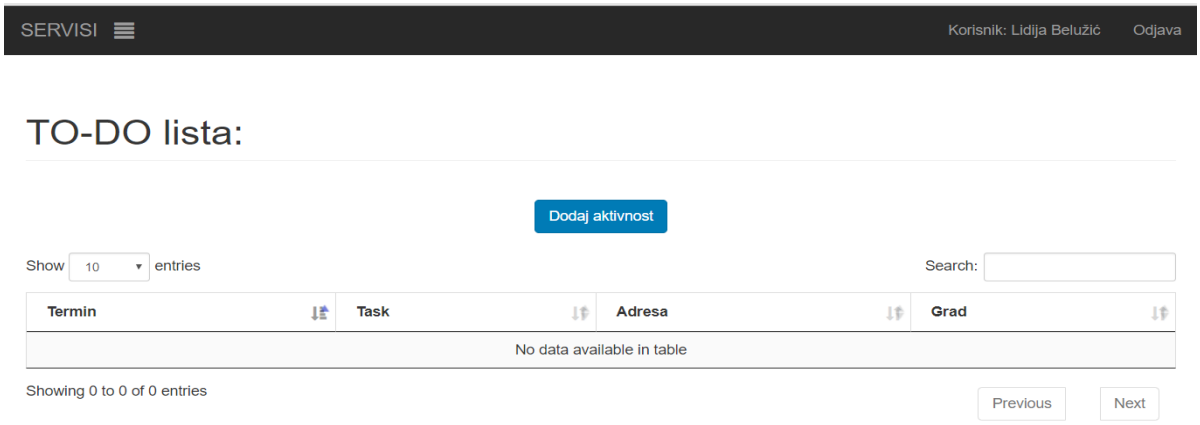



*Slika 24. Skočni prozor za unos naloga*

Slikom 24 prikazan je skočni prozor kojim se koriste serviseri da bi unijeli moguće promjene u nalog pomoću kojeg se usmjerava tijek posla kroz dan. U nalog je potrebno unijeti termin s vremenom i godinom, uz pomoć padajućeg izbornika izabrati određenog servisera koji treba izvršiti zadatak, te adresu i grad. Pod aktivnosti se unosi opis zadatka o kojem se radi, dok trajanje posla i putovanja unosi serviser koji je na mjestu izvršenja posla. Ukoliko serviser ne uspije riješiti problem na trenutnoj lokaciji, dogovara se sa krajnjim korisnikom odmah za sljedeći termin te ga unosi pod rubriku novi termin. Time zahtjev ostaje aktualan i u dogovoreni termin serviser nije slobodan, a admin mu nema mogućnosti dodijeliti drugi zahtjev.

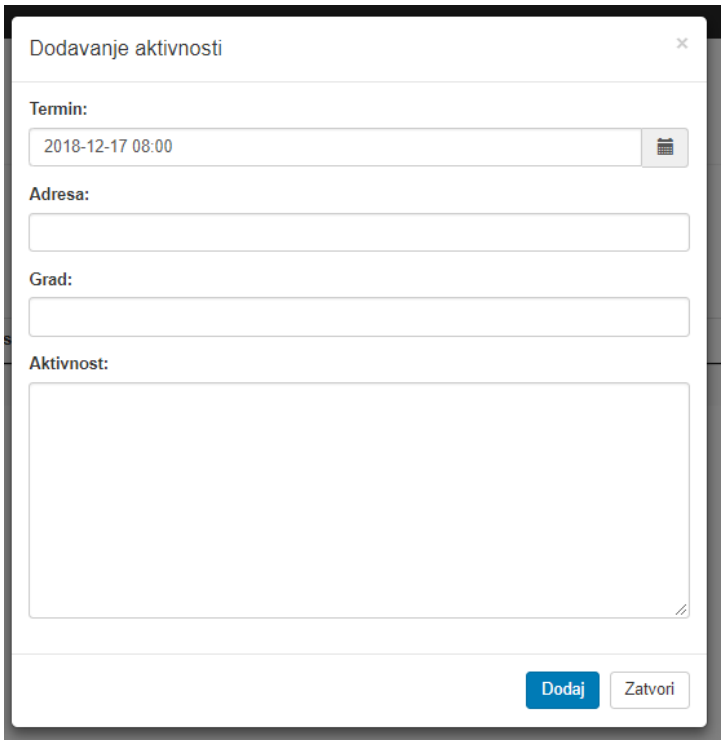
### 8.2.3. Unos naloga od strane krajnjih korisnika

Krajnji korisnik klikom na cjelinu „Unos naloga“, odnosno pod cjelinu „Unos radnog naloga“ ima mogućnost naručivanja servisera u određeno vrijeme i na određenu adresu, radi raznih problema zbog pomoći od strane istog. Krajnji korisnik se također mora nalaziti u bazi podataka da bi mogao imati pristup za otvaranje naloga. Web aplikaciji pristupa pomoću korisničkog imena koje mu je dodijeljeno i lozinke. Na slici 25 prikazano je sučelje kojim se krajnji korisnik služi u svrhu otvaranja zahtjeva radi određenog nedostatka u svom kućanstvu.




*Slika 25. Sučelje za krajnjeg korisnika*

Krajnjem korisniku na ovom sučelju biti će prikazani samo zahtjevi koji mu pripadaju i koji još nisu izvršeni. Ukoliko je nalog zatvoren, korisnik više neće imati mogućnosti vidjet ga. Također krajnjem korisniku je omogućeno ukoliko ima otvoren veći broj naloga, da ga pomoću tražilice (engl. *Search*) može na lakši način pronaći pomoću bilo kojeg podataka koji mu je poznat da je unijet u nalog.




*Slika 26. Skočni prozor krajnjeg korisnika*

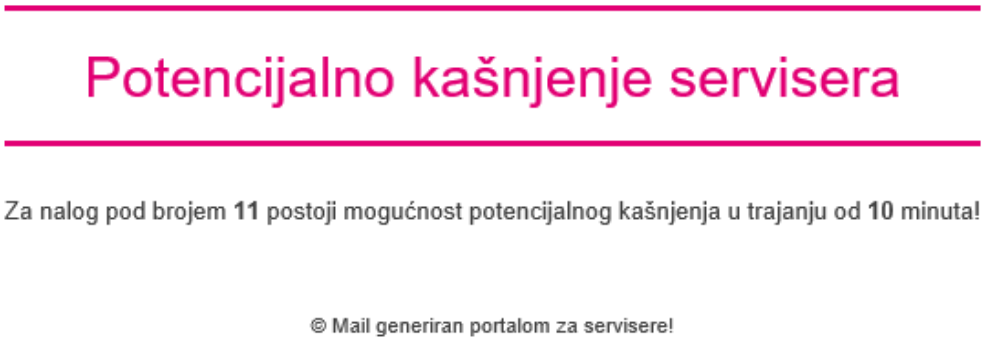
Slika 26 prikazuje skočni prozor u kojem je potrebo popuniti navedena polja od strane krajnjeg korisnika sa odgovarajućim podatcima. Krajnji korisnik treba izabrati određeni datum i vrijeme u koje želi da mu serviser dođe, na koju adresu i u kojem gradu se nalazi, te u polje aktivnost treba navesti kakav problem ima, i na koji kućanski aparat se problem odnosi. Kada krajnji korisnik ispravno popuni sve potrebne podatke koji su se tražili od njega, otvaranje naloga potvrđuje klikom na gumb „Dodaj“ ili ukoliko se predomislio i ipak ne želi otvoriti novi nalog za aktivnosti, može ga zatvoriti klikom na gumb „Zatvori“.

### 8.2.4. Obrada i analiza podataka dobivenih aplikacijom

Sam proces je zamišljen da bi se što preciznije znalo vrijeme dolaska servisera na popravak. Aplikacija ima širinu i univerzalnost da se može primjenjivati u bilo kojem okruženju gdje je potreba za interakcijom između korisnika i poduzeća koje obavlja neku radnju s korisnikom.

Proces ima 4 osnovne faze da bi se nalog ispravno izvršio. Nakon što korisnik otvori nalog on dolazi na grupu koja odrađuje te iste naloge. U tom trenutku nalog nije dodijeljen ni na jednog servisera. Zadaća dispečera, odnosno osobe koja ima admin prava je dodijeliti zadatak serviseru kroz svoj prikaz korisničkog sučelja za servisere. Nakon izvršene radnje, nalog će biti dodijeljen na servisera koji će isti odrađivati. Po

zaprimanju naloga od strane servisera, serviser dobiva zadaću prvenstveno odraditi sami zahtjev no uz to i odabrati određene parametre na samom nalogu. Ti parametri su trajanje posla i vrijeme putovanja koje je izraženo u minutama. Ti podatci su bitni radi procjene slijedećeg naloga. Na temelju zbroja istih sustav određuje postoji li vjerojatnost da će serviser kasniti slijedećem korisniku s kojim ima zakazan termin. U slučaju kašnjenja korisnik dobiva e-mail poruku prikazanu na slici 27.




*Slika 27. Izgled mail poruke za korisnika*

Po odradi samog posla serviser zatvara nalog klikom na gumb gotovo. Time nalog odlazi u arhivu.

# 9. Zaključak

U današnje vrijeme radi vrlo velike konkurencije na tržištu, tvrtke su primorane optimizirati svoje unutarnje poslovne procese. Ključni faktor su poslovni modeli koji se optimiziraju i predstavljaju detaljno opisanu strukturu sustava. Sustav se raščlanjuje na manje segmente radi bolje opisane strukture poslovanja, implementacije i prikaza pojedinačne analize organizacije.

Upravo zbog toga, segmente je moguće bolje analizirati i prikazati ih uz pomoć UML dijagrama. U ovome radu korišteni su odgovarajući UML dijagrami za prikaz svakog pojedinog dijela sustava. Time se osigurao jednostavniji razvoj kompleksnih logika, odnosno pretvaranje navedenog sustava u aplikativno podržan oblik.

Svrha ovog rada je analizirati problematiku upravljanja zahtjevima za servisom u servisnom centru i primjena UML dijagrama za prikaz poslovnih procesa upravljanja istih. Uz pomoć razvijene web aplikacije omogućen je proces vođenja evidencije lokacija predviđenih za servisere i termina u kojem ih je krajnji korisnik potraživao.

Web aplikacija kreirana je uz pomoć PHP programskog jezika uz bazu podataka Microsoft SQL. Primjenom UML dijagrama prikazan je proces i zamisao same aplikacije s kojom se postiže kvaliteta i skraćenje vremena isporuke usluge. Na osnovi prikazanih UML dijagrama, modeliran je cijeli sustav pomoću kojega je moguće izračunati predviđeno vrijeme dolaska servisera kod svakog pojedinog korisnika. Uz pomoć izračuna krajnji korisnik može procijeniti hoće li doći do kašnjenja servisera ili ne.

Dakle, nije više potrebno od strane korisnika provesti nekoliko sati čekajući servisera, već to slobodno vrijeme može iskoristiti na željeni način. Čime je u konačnici krajnji korisnik zadovoljan jer su zadovoljene njegove potrebe.

# LITERATURA


[1] Kissflow. *The Extensive Guide to Business Processes.* Preuzeto sa: https://kissflow.com/bpm/business-process/. [Pristupljeno: srpanj 2019].

[2] Poslovni dnevnik. *Lanac vrijednosti.* Preuzeto sa: http://www.poslovni.hr/leksikon/lanac-vrijednosti-711. [Pristupljeno: srpanj 2019].

[3] Jurčević M. *Tehnologijski marketing i menadžment.* Autorizirana predavanja. Fakultet prometnih znanosti Sveučilišta u Zagrebu; 2019.

[4] Bolje. *Outsorcing.* Preuzeto sa: http://bolje.hr/rijec/outsourcing-gt-izdvajanje-posla/1/. [Pristupljeno: srpanj 2019].

[5] Castela N., Tribolet J., Silva A., Guerra A. *Business Process Modeling With Uml*, Lisabon: 2004. Preuzeto sa: https://www.researchgate.net/publication/2934944_Business_Process_Modeling_With_Uml. [Pristupljeno: srpanj 2019.].

[6] Dewalt C. *BUSINESS PROCESSMODELINGWITH UML*. Johns Hopkins University: 1999. Preuzeto sa: https://www.researchgate.net/publication/220708405_Business_Process_Modeling_with_UML. [Pristupljeno: srpanj 2019].

[7] Brumec J. *Modeliranje poslovnih procesa.* KORIS. Varaždin/Zagreb: 2011. Preuzeto sa: https://koris.hr/preuzmi/koris-uvod-u-modeliranje-poslovnih-procesa.pdf. [Pristupljeno: kolovoz 2019].

[8] The Unified Modeling Language. *UML Use Case Diagrams* Preuzeto sa: https://www.uml-diagrams.org/use-case-diagrams.html. [Pristupljeno: srpanj 2019].

[9] Mrvelj Š. *Prikupljanje zahtjeva na sustav.* Autorizirana predavanja. Fakultet prometnih znanosti Sveučilišta u Zagrebu; 2014./2015.

[10] Šutalo S. *Osnove programiranja u jeziku C++.* Preuzeto sa: https://sites.google.com/site/sandasutalo/oosnove-programiranja/pomocni-postupci-pri-programiranju/pseudo-jezik. [Pristupljeno: srpanj 2019].

[11] Booch G., Rumbaugh J., Jacobson I. *The Unified Modeling Language User Guide.* Addison Wesley, 1998.

[12] The Unified Modeling Language. *Activity Diagrams*. Preuzeto sa: https://www.uml-diagrams.org/activity-diagrams.html. [Pristupljeno: srpanj 2019].

[13] Šutalo S. *Računalstvo*. Preuzeto sa: http://www.sanda-sutalo.from.hr/index.php?option=com_content&view=article&id=39&Itemid=60. [Pristupljeno: srpanj 2019].

[14] Mrvelj Š. *Opis tijekova podataka.* Autorizirana predavanja. Fakultet prometnih znanosti Sveučilišta u Zagrebu; 2018.

[15] Mrvelj Š. *Dijagram klasa.* Autorizirana predavanja.Fakultet prometnih znanosti Sveučilišta u Zagrebu; 2016./2017.

[16] Mrvelj Š. *Praćenje života elemenata.* Autorizirana predavanja. Fakultet prometnih znanosti Sveučilišta u Zagrebu; 2015.

[17] Vrsalović P. *Logička ili Booleova algebra.* Autorizirana predavanja. Prirodoslovna i grafička škola Rijeka. Preuzeto sa: http://vrsa.pgsri.hr/I-razred/I-Predavanja/Logicki-pdf/Booleova.pdf. [Pristupljeno: srpanj 2019].

[18] Mrvelj Š. *Opis zajedničkog rada elemenata.* Autorizirana predavanja. Fakultet prometnih znanosti Sveučilišta u Zagrebu; 2018.

[19] Mrvelj Š. *Dijagram suradnje.* Autorizirana predavanja. Fakultet prometnih znanosti Sveučilišta u Zagrebu; 2015.

[20] PHP tutorial odlican. *PHP.* Peuzeto sa: http://phptutorial.odlican.net/index.php. [Pristupljeno: prosinac 2018].

[21] Preuzeto sa: https://upload.wikimedia.org/wikipedia/commons/4/4f/Scheme_dynamic_page_en.svg. [Pristupljeno: prosniac 2018].

[22] Tutorialspoint. *MS SQL Server Tutorial*. Preuzeto sa: https://www.tutorialspoint.com/ms_sql_server/. [Pristupljeno: prosniac 2018].

# POPIS KRATICA

HTML (engl. *HyperText Markup Language*) prezentacijski jezik za izradu internet stranica

PHP (engl. *Hypertext Preprocessor*) programski jezik namijenjen programiranju dinamičkih Internet stranica

UML (engl. *The Unified Modeling Language*) standardni jezik za modeliranje i vizualni prikaz dizajna sustava

# POPIS SLIKA

Sveučilište u Zagrebu
Fakultet prometnih znanosti
10000 Zagreb
Vukelićeva 4

## IZJAVA O AKADEMSKOJ ČESTITOSTI I SUGLASNOST

Izjavljujem i svojim potpisom potvrđujem kako je ovaj diplomski rad isključivo rezultat mog vlastitog rada koji se temelji na mojim istraživanjima i oslanja se na objavljenu literaturu što pokazuju korištene bilješke i bibliografija.

Izjavljujem kako nijedan dio rada nije napisan na nedozvoljen način, niti je prepisan iz necitiranog rada, te nijedan dio rada ne krši bilo čija autorska prava.

Izjavljujem također, kako nijedan dio rada nije iskorišten za bilo koji drugi rad u bilo kojoj drugoj visokoškolskoj, znanstvenoj ili obrazovnoj ustanovi.

Svojim potpisom potvrđujem i dajem suglasnost za javnu objavu diplomskog rada pod naslovom **ANALIZA I MODELIRANJE APLIKACIJE ZA NADZOR USLUGA SERVISERA PREMA KRAJNJIM KORISNICIMA**

na internetskim stranicama i repozitoriju Fakulteta prometnih znanosti, Digitalnom akademskom repozitoriju (DAR) pri Nacionalnoj i sveučilišnoj knjižnici u Zagrebu.

Student/ica:

U Zagrebu, 17.09.19

Lidija Belužić
*(potpis)*