Sistemski dizajn

Kontrola autobuskih stanica

verzija 1.0

Elektrotehnički fakultet

Banja Luka

Sadržaj

1. Uvod

Prvo poglavlje ovog dokumenta daje kratak pogled na namjenu, ciljeve sistema, kao i definicije, skraćenice i referentne dokumente koji su korišteni prilikom specifikacije sistemskog dizajna. Na početku je navedena namjena sistama i njegove komponente, sa posebnim naglaskom na ono šta rezultantni softver treba da radi, kao i projektni ciljevi koji bi se razvojem sistema trebali zadovoljiti. Nakon toga su navedene skraćenice i definicije pojmova od interesa koji se pojavljuju u specifikaciji, te su navedeni referentni dokumenti. Na kraju su opisane predložene arhitekture, te je dat pregled strukture kompletnog dokumenta radi lakše navigacije tokom čitanja.

1.1. Namjena sistema

Sistem "BUST" je namjenjen autobuskim prevoznicima, autobuskim stanicama i turističkim agencijama i omogućava potpunu automatizaciju rada autobuske stanice kroz:

- Administracija sistema (upravljanje parametrima sistema, korisničkim nalozima, pravima pristupa, zaposlenima, finansijama i sl.)
- Podsistem za prodaju i rezervaciju karata (klijent-server aplikativno rješenje za pretragu polazaka, komunikaciju sa serverom kako bi svim učesnicima u sistemu pružilo informaciju o zauzetosti sjedišta u realnom vremenu, rezervisanje i prodaju karata, vođenje dnevnika saobraćaja i sl.)
- Podsistem za upravljanje parkingom.

1.2. Projektni ciljevi

Projektni ciljevi:

- Pouzdanost, konkurentnost i integritet sistema se oslanja na rad DBMS-a.
- Sigurnost je obezbjeđena putem forme za prijavu. Svaki korisnik se na sistem prijavljuje putem korisničkog imena i šifre, a dodatna sigurnost se ostvaruje time što su korisnici podjeljeni u različite grupe, od kojih svaka grupa pristup samo onima funkcionalnostima i podacima za koje ima privilegije.

Fizička sigurnost sistema podrazumjeva da će se centralni serverski računar nalaziti u zaštićenoj sobi sa odgovarajućim mjerama sigurnosne zaštite. Opremu je potrebno fizički zaštititi od neovlaštenog pristupa, oštećenja i ometanja.

- Visoke performanse se ostvaruju ograničavanjem broja korisničkih zahtjeva koji se mogu u jednom trenutnku uputiti serveru. Ne očekuje se da će taj broj prelaziti 200 istovremenih korisničkih zahtjeva. Vrijeme odziva sistema prilikom unosa, čitanja, brisanja, ažuriranja i prikaza podataka, u opštem slučaju ne smije biti veće od 5s.
- Efikasnost se ogleda u nastojanju da se softver primjeni u poslovanju autobuske stanice tako da on bude djelotvoran u smislu onoga šta putnici traže.
- Prenosivost - sistem će biti zasnovan na Java platformi, te će ga biti moguće koristiti na svakom operativnom sistemu uz pretpostavku da je instaliran Java Runtime Enviroment.
- Isplativnost se ogleda u ubrzanom procesu prodaje karata, većom sigurnosti, usklađenosti između autobuskih stanica, smanjenoj vjerovatnoći greški itd.
- Robusnost se realizuje backup-om podataka na drugu lokaciju, koja će biti zaštićena i neće biti pogođena štetom kao osnovna lokacija, pri čemu će se backup vršiti jednom sedmično. Sistem će signalizirati ukoliko se backup izvrši neuspješno. U slučaju nestanka struje ili kvara na HW-u, sistem će se automatski vratiti na zadnje sačuvane podatke.
- Skalabilnost – dobar dizajn sistema će omogućiti dodavanje novih funkcionalnosti u skladu sa potrebama klijenata.
- Lako održavanje se odnosi na mogućnosti zamjene i nadogranje HW i SW. Zamjena i nadogradnja HW će biti omogućena bez prekida rada sistema pri zamjeni redudantnih komponenti, a pri zamjeni i nadogradnji ostalih komponenti, nadogradnja i zamjena će biti omogućena izvan radnog vremena. Nadogradnja SW će biti omogućena izvan radnog vremana.
- Upotrebljivost (laka upotreba) podrazumjeva da je dizajn grafičkog interfejsa, kao i samog sistema, prilagođen korisnicima kako bi se mogli što jednostavnije i intuitivnije obavljati potrebni zadaci.
    - Korisnički grafički interfejs će biti bez suvišnih detalja i nedvosmislen, te će se sastojati od adekvatnih grafičkih kontrola sa čitkim fontovima.
    - Ukoliko korisnik pogriješi prikazaće se odgovarajuće poruke upozorenja.

1.3. Definicije, skraćenice

Pojam	Opis
BUST	Sistem za kontrolu autobuskih stanica.
Klijent-server aplikacije	Svaka aplikacija se sastoji od dijela koji se izvodi na poslužitelju (serveru) i dijela koji treba instalirati na korisnički računar (klijent). Klijent preko svog interfejsa daje zahtjeve serveru koji ih izvodi, klijent dobija izvještaje.
Putni nalog	Akt na osnovu kojeg vozilo može da obavlja prevoz u drumskom saobraćaju.
Dnevnik saobracaja	Dokument o prometu autobusa kroz autobusku stanicu ili stajalište.
Izvještaj	Finansijski - formalni pregled finansijskih aktivnosti nekog pravnog ili fizičkog lica i drugog obveznika finansijskog izvještavanja na kraju poslovnog perioda. Izvještaj o prodanim kartama - dokument o prodanim kartama za neki polazak koji se predaje vozaču najkasnije 5min prije polaska autobusa.
DAO	Data access object
DTO	Data transfer object
MVC	Softverski patern Model-view-controller koji odvaja prikaz informacija od interakcije korisnika sa tim informacijama.
Use case	Prikazuju skup slučajeva korištenja i aktera.
Specifikacija	Dokument u kome se utvrđuju tehnički zahtjevi za proizvode i postupci ocjenjivanja usaglašenosti.
DBMS	Database management system

1.4. Referentni dokumenti

[1] Kontrola autobuskih stanica, SRS

[2] Bernd Bruegge and Allen H. Dutoit, "Object-Oriented Software Engineering", Third Edition, Chapter 7

[3] System Design Document, TRAMP Project, February 07, 2002

[4] B. Bruegge, A.H. Dutoit, Object-Oriented Software Engineering Using UML, Patterns, and Java

1.5. Kratak pregled dokumenta i predložene arhitekture

Ostatak dokumenta sadrži dva poglavlja. Drugo poglavlje daje pregled nekoliko posmatranih arhitektura, te izbor jedne.

Treće poglavlje sadrži kratak pregled izabrane arhitekture i funkcionalnosti sistema, te opis osam bitnih aktivnosti u projektovanju.

2. Arhitektura postojećeg sistema

Arhitekturalni stilovi koje smo uzeli u razmatranje su:

- Troslojni arhitekturalni stil
- Četvoroslojni arhitekturalni stil
- Peer-to-peer arhitekturalni stil
- Klijent-server arhitekturalni stil
- MVC arhitekturalni stil.

Softver “Kontrola autobuskih stanica” je zamišljen kao samostalna desktop aplikacija sa bazom podataka smještenom na serverskom računaru. Zbog toga smo izabrali MVC koji će više biti opisan u sledećem poglavlju.

3. Predložena arhitektura

Model-view-controller je softverska arhitektura, tj. arhitektualni šablon koji se koristi u softverskom inžinjeringu. Šablon razdvaja aplikativnu logiku za korisnika od korisničkog interfejsa, i time omogućava nezavisan razvoj, testiranje i održavanje. MVC šablon kreira aplikacije koje razdvajaju različite aspekte aplikacije, a pritom obezbjeđuje labavu vezu između ovih elemenata.

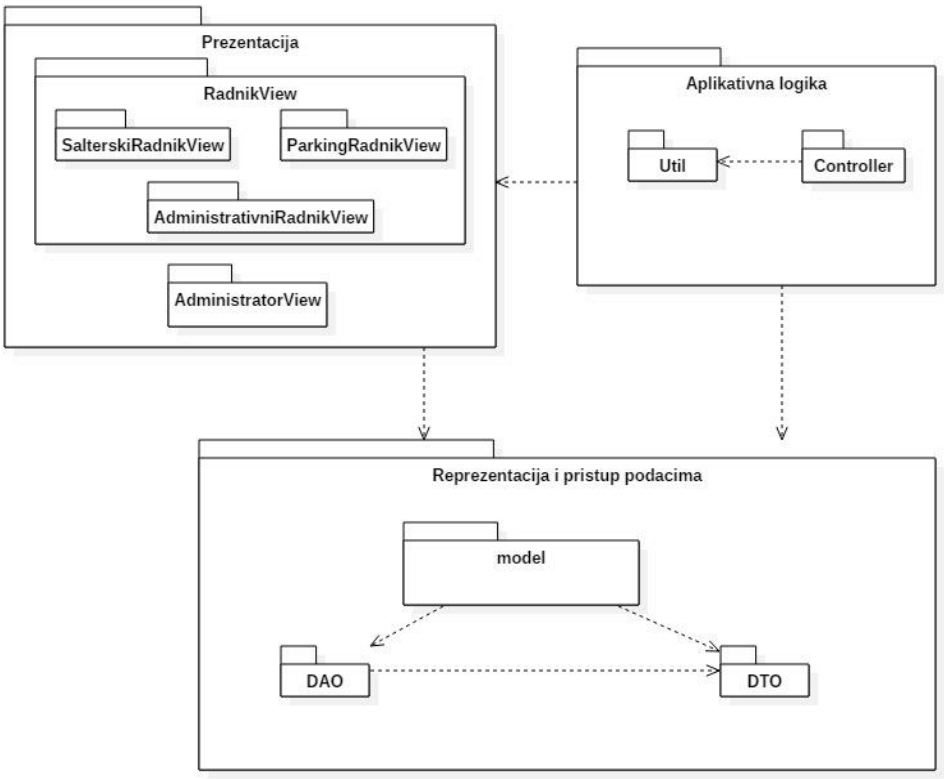
3.1. Kratak pregled arhitekture i funkcionalnosti podsistema

Softver “Kontrola autobuskih stanica” je zamišljen kao klijent-server aplikacija koja se realizuje pomoću MVC arhitekture. Sistem se sastoji iz baza podataka koje se nalaze na odgovarajućim serverima i korisničke aplikacije koja se nalazi na personalnim računarima radnika.

3.2. Dekompozicija sistema

Aplikacija se sastoji od tri funkcionalna podsistema: prezentacija, aplikativna logika, reprezentacija i pristup podacima. Podsistem prezentacije je odgovoran za prikaz podataka krajnjem korisniku i sastoji se od sledećih pogleda, RadnikView i AdministratorView. Aplikativna logika se sastoji od paketa Controller i Util. Controller je odgovoran za sekvencu interakcija sistema i korisnika i prosljeđivanje pomjena modela prema pogledu. Util paket sadrži pomoćne klase čije usluge koriste klase Controllera.

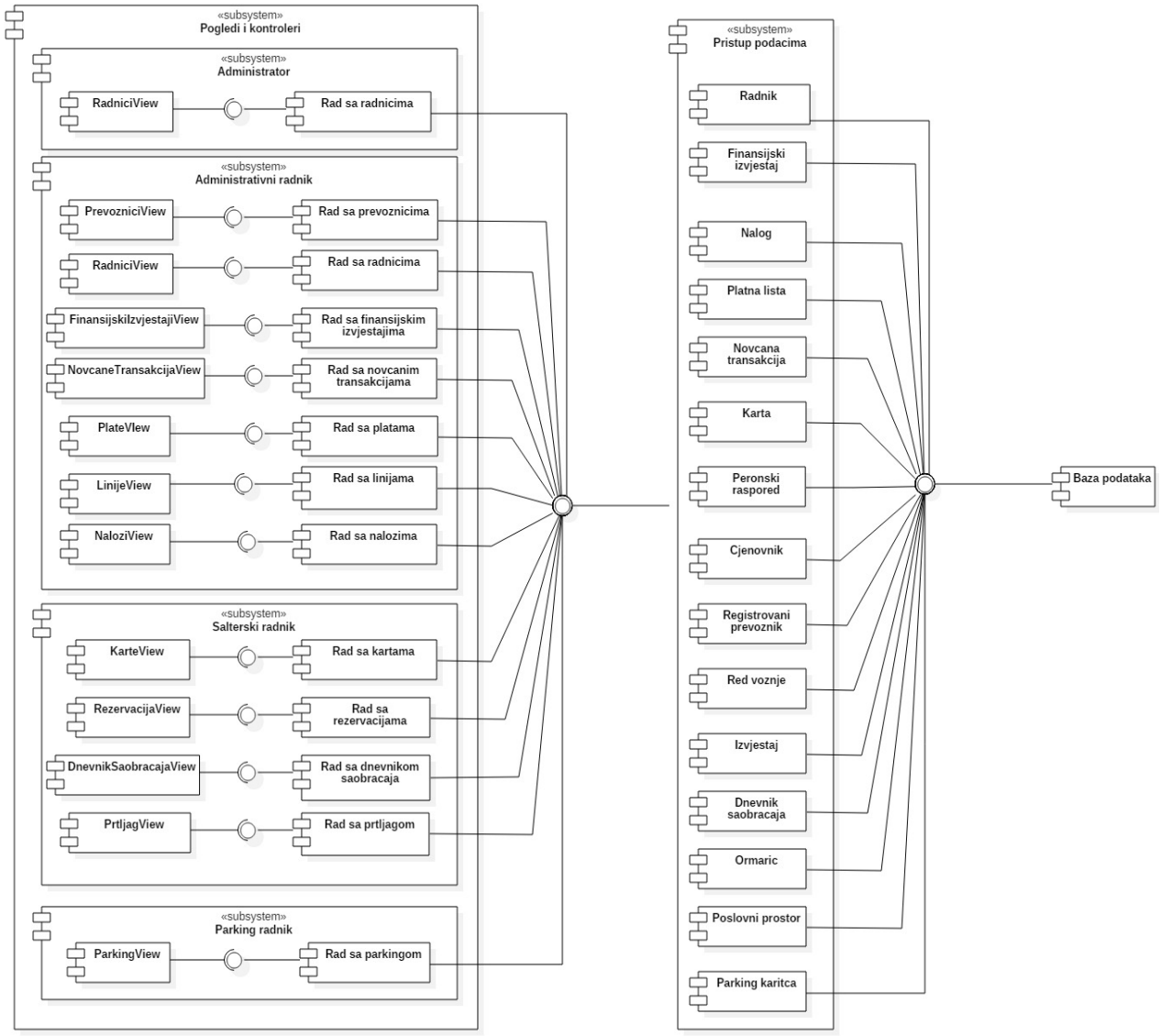
Paket model sadrži modele podataka. Paket DTO ili objekat za prijenos podataka je objekt koji enkapsulira podatke te ih šalje od jednog podsistema ili sloja aplikacije do drugog. DAO predstavlja komponentu koja obezbjeđuje interfejs između aplikacija i ciljnih DBMS-ova.



3.3. HW/SW mapiranje

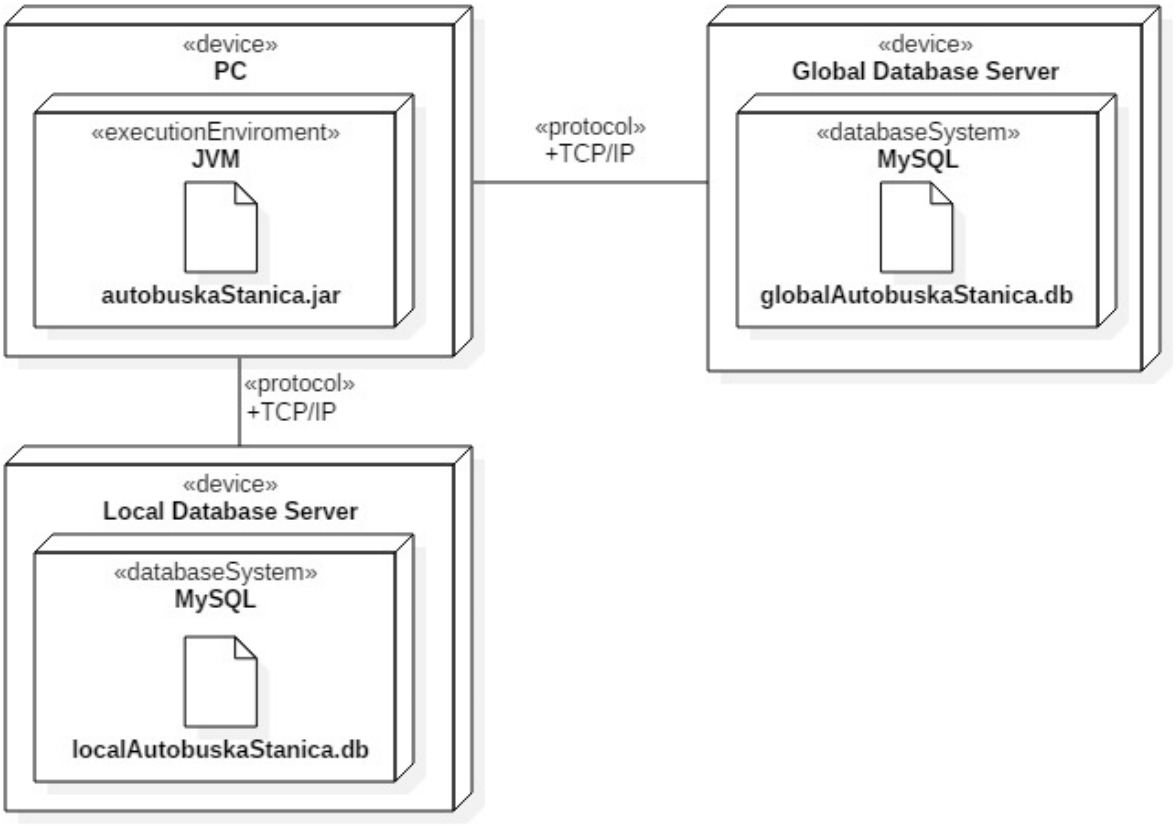
UML dijagram komponenti prikazuje strukturne relacije između softverskih komponenti sistema. On ilustruje djelove softvera, ugrađene kontrolere i slično, definiše relacije i zavisnosti između objekata.

Komponente iz kojih se sistem sastoji su prikazane na sledećem dijagramu:



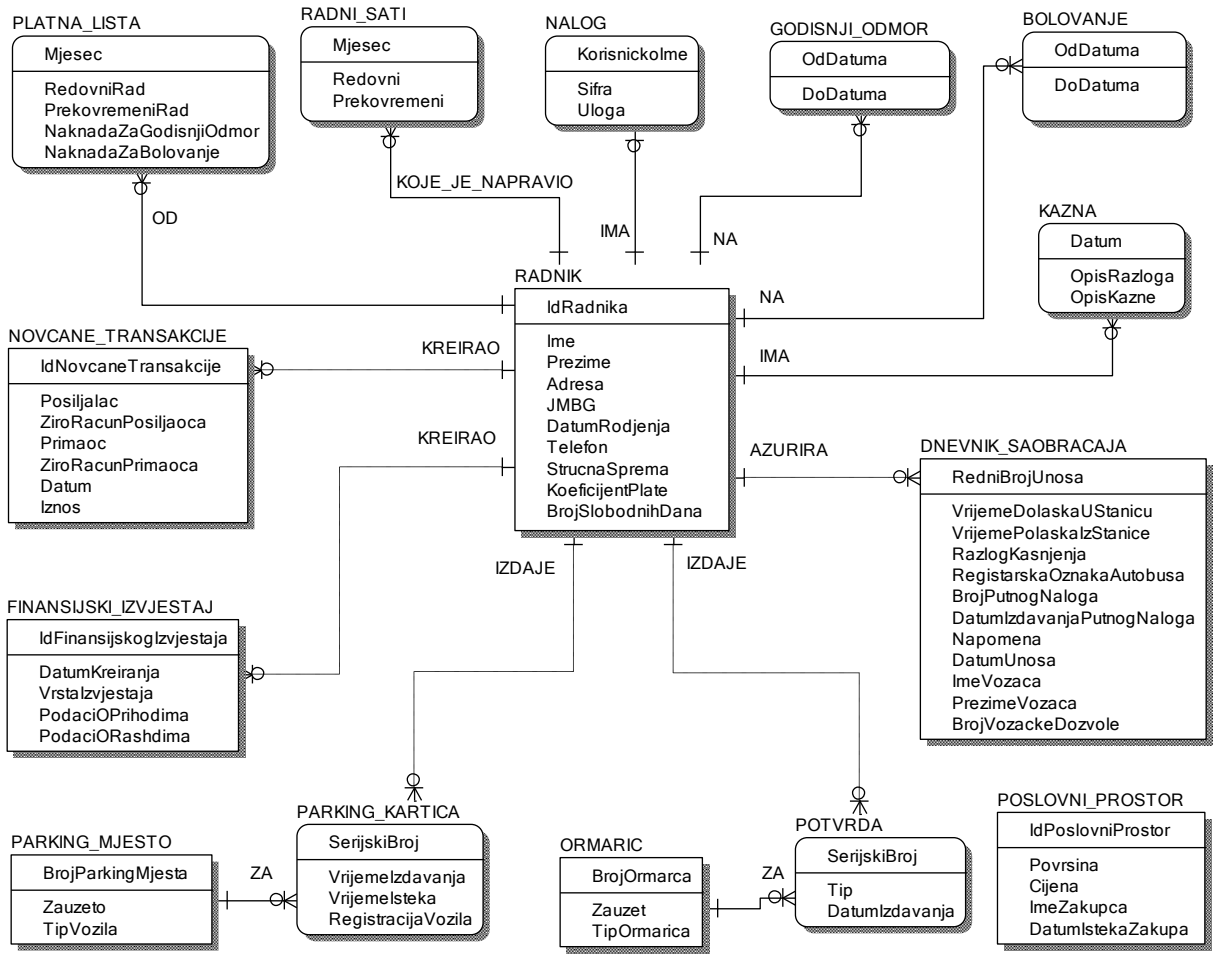
Komponente su podjeljenje u podsisteme, pogledi i konroleri i prikaz podataka. Svaki podsistem sadrži komponente koje se odnose na određenje funkcionalnosti.

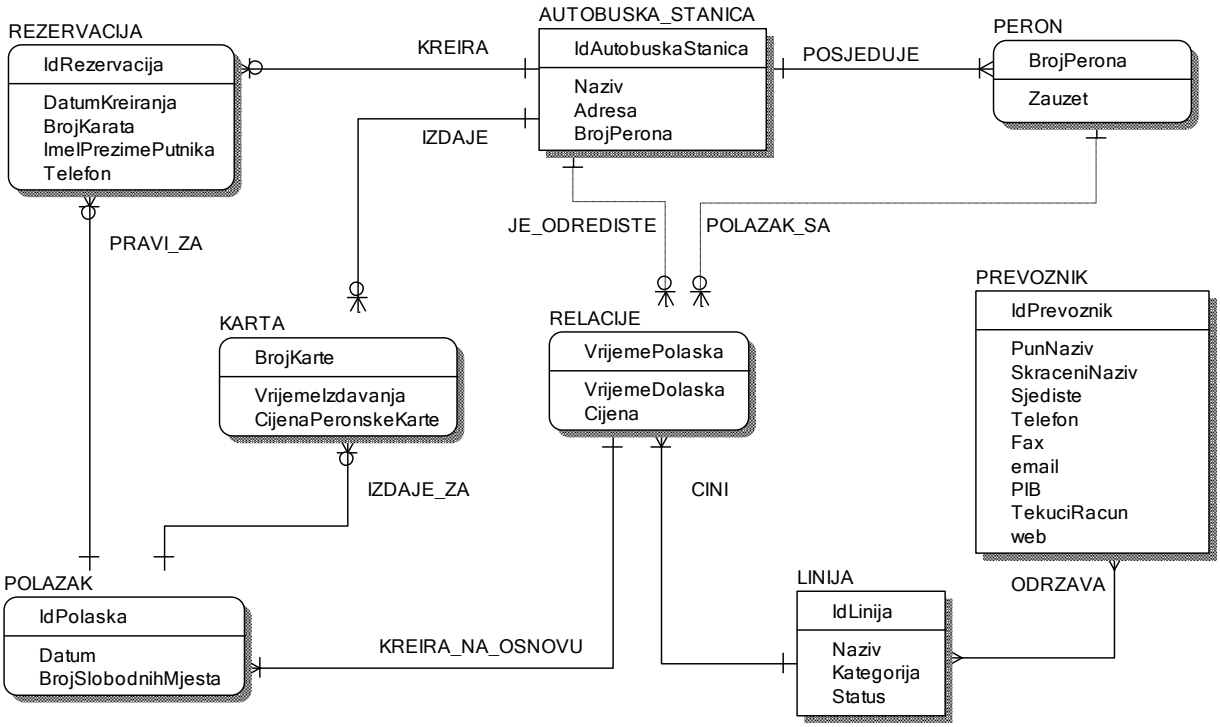
Dijagram razmeštaja prikazuje hardver našeg sistema, softver koji je instaliran na tom sistemu i srednji sloj koji se koristi za povezivanje međusobno razdvojenih mašina. Naš sistem ima lokalnu i globalnu bazu koju koriste aplikacije. Lokalna baza na kojoj se čuvaju podaci za konkretnu autobusku stanicu se nalazi na serveru koji može biti smješten na samoj autobuskoj stanici. U globalnoj bazi se čuvaju podaci koji su dijeljeni od strane svih autobuskih stanica u lancu autobuskih stanica. Ona se nalazi na centralnom serveru. Aplikaicija je smještena na personalnim računarima radnika.



3.4. Perzistentni sloj

Perzistentni sloj smo odlučili da realizujemo preko nekog od popularnih DBMS-ova, zbog relativno velike količine podataka i drugih pogodnosti koje DBMS pruža. Relacione šeme su urađene na osnovu dijagrama klasa. Relacionu šemu smo razdvojili na dve odvojene šeme zbog relavantnosti podataka na lokalnom i globalnom nivou. Na globalnom nivou su na relevantni podaci koji se dijele između autobuskih stanica(druga šema) dok su na lokalnom nivou relevantni podaci isključivo oni koji se tiču jedne konkretne autobuske stanice(prva šema).





3.5. Kontrola prava pristupa i sigurnost

Kontrola pristupa i sigurnost je obezbjeđena podijelom korisnika u različite grupe od kojih svaka grupa ima pristup samo onim funkcionalnostima i podacima za koje ima privilegije. Spisak učesnika i klasa, kao i lista operacija koje svaki učesnik može da izvrši nad objektima date klase su prikazani sledećom matricom pristupa.

Učesnici	**Klase**				
	AutobuskaStanica	Radnik	Nalog	Novčana Transakcija	FinansijskiIzvještaj
Administrator		isBolovanje() isGodišnji() pregledKazni() brojSlobodnihDana() koeficijentPlate()	login() logout()	prikaži()	kreiraj() prikaži() štampaj()
Administrativni radnik		isBolovanje() isGodišnji() pregledKazni() brojSlobodnihDana() koeficijentPlate()	login() logout()		
Računovođa			login() logout()	prikaži()	kreiraj() prikaži() štampaj()

Šalterski radnik			login() logout()		
Parking radnik			login() logout()		
	PlatnaLista	RadniSati	Prevoznik	Linija	RedVožnje
Administrator	kreiraj() prikaži() štampaj()	unesi()		getStatus()	cijenaVožnje() trajanjeVožnje()
Administrativni radnik				getStatus()	cijenaVožnje() trajanjeVožnje()
Računovođa	kreiraj() prikaži() štampaj()	unesi()			
Šalterski radnik				getStatus()	cijenaVožnje() trajanjeVožnje()
Parking radnik					
	Polazak	Karta	DnevnaKarta	Mjesečna Karta	Račun
Administrator		popunjavanjeKarte() štampanjeKarte() prodaja() produženje() vraćanje() otkazivanje()	formiranjeUkupne Cijene()	formiranje Cijene()	štampanjeRačuna()
Administrativni radnik					
Računovođa					
Šalterski radnik		popunjavanjeKarte() štampanjeKarte() prodaja() produženje() vraćanje() otkazivanje()	formiranjeUkupne Cijene()	formiranje Cijene()	štampanjeRačuna()
Parking radnik					
	Rezervacija	Peron	Dnevnik Saobraćaja	Izvještaj	ParkingMjesto
Administrator	popunjavanje () otkazivanje ()	isZauzet()	getDnevniUnos() dodajPodatke()	kreiraj() prikaži()	isZauzeto() getTipVozila()

			štampaj()	štampaj()	
Administrativni radnik		isZauzet()			
Računovođa					
Šalterski radnik	popunjavanje () otkazivanje ()		getDnevniUnos() dodajPodatke() štampaj()	kreiraj() prikaži() štampaj()	
Parking radnik					isZauzeto() getTipVozila()
	ParkingKartica	Ormarić	Potvrda	Ponuda	PoslovniProstor
Administrator	odgovarajućaParkingMjesta() isIstekla() unosPodataka() štampaj()	isZauzet()	štampaj()	prikaži() štampaj()	
Administrativni radnik					
Računovođa					
Šalterski radnik					
Parking radnik	odgovarajućaParkingMjesta() isIstekla() unosPodataka() štampaj()	isZauzet()	štampaj()	prikaži() štampaj()	

3.6. Kontrola toka

Kontrola toka cijelokupnog sistema je distribuirana, dok je kontrola toka same klijentske aplikacije je centralizovana "event-driven" mehanizmom. Zbog distribuiranosti aplikacije na personalnim računarima, cijelokupan sistem je distribuirane kontrole toka. Sami serveri služe samo za čuvanje podataka, pa ne preuzimaju odgovornost za kontrolu toka.

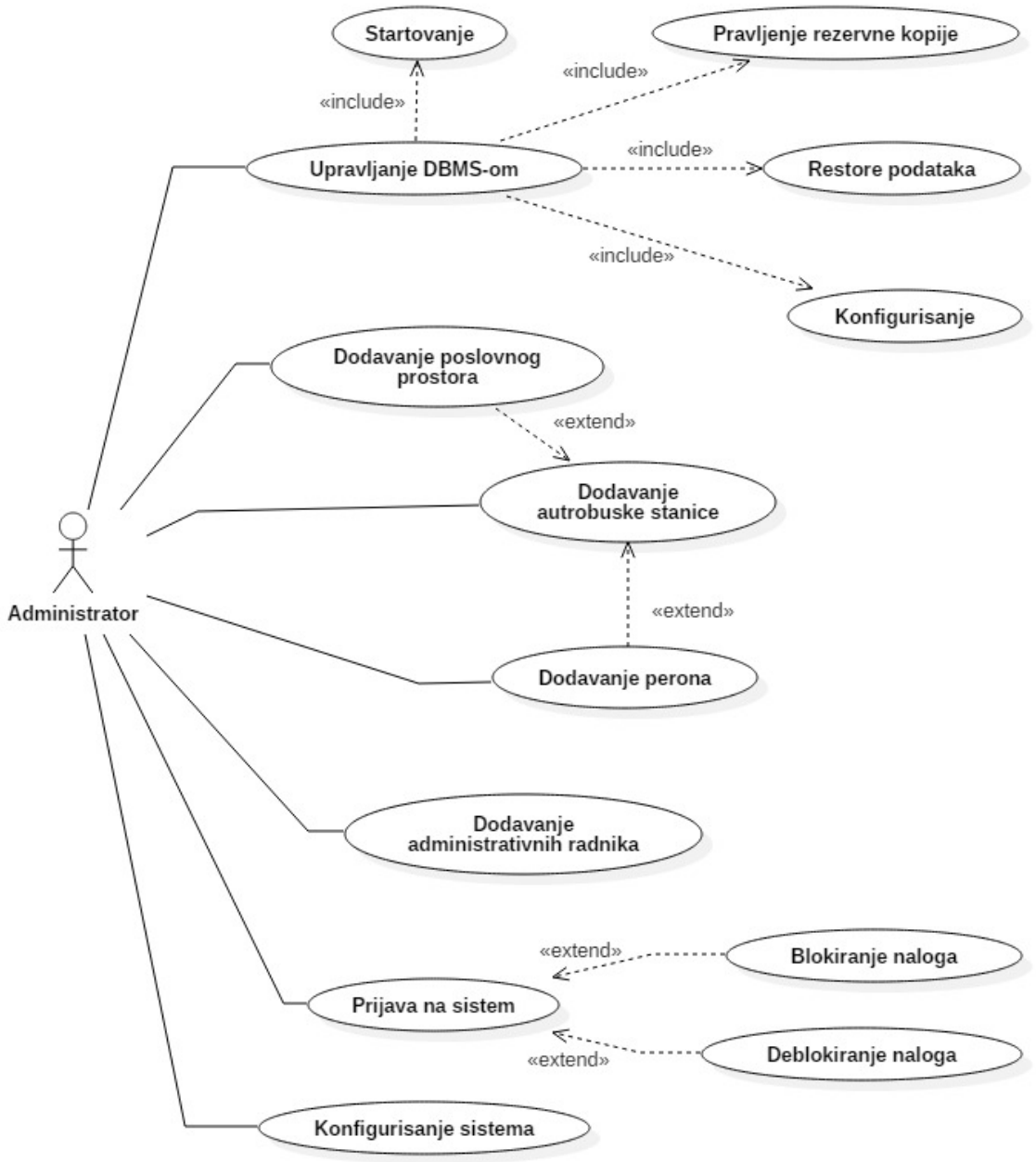
Konkurentnost aplikacije će biti bazirana na konkuretnosti DBMS-a. Fizička konkuretnost se postiže distribuiranjem aplikacije na više računara koji komuniciraju sa centralnizovanim bazama podataka.

3.7. Granična stanja sistema

Granični slučajevi upotrebe tipično se ne identifikuju tokom analize sistema, jer su mnogi granični slučajevi posljedica dizajna.

U slučaju korisničke greške, sistem bi trebao prikazati jasnu poruku o grešci tako da bi korisnik mogao popraviti unos. U slučaju pucanja mrežne konekcije prilikom povezivanja sa bazom podataka, sistem bi trebao sačuvati trenutno stanje tako da je moguće izvršiti oporavak kada se konekcija vrati.

Granična stanja sistema su modelovana sledećim dijagramom graničnih slučajeva upotrebe:


Startovanje
Pravljenje rezervne kopije
«include»
«include»
Upravljanje DBMS-om
«include»
Restore podataka
«include»
Konfigurisanje
Dodavanje poslovnog prostora
«extend»
Dodavanje autrobuske stanice
Administrator
«extend»
Dodavanje perona
Dodavanje administrativnih radnika
«extend»
Blokiranje naloga
Prijava na sistem
«extend»
Deblokiranje naloga
Konfigurisanje sistema