voltriX

Prosti simulator i editor električnih kola

Arhitekturni projekat

Verzija 1.0

Pregled izmena

|  |  |  |  |
| --- | --- | --- | --- |
| **Datum** | **Verzija** | **Opis** | **Autor** |
| 10.04.2019. | 1.0 | Inicijalna verzija | Marko, Vladeta |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Sadržaj

1. Cilj dokumenta 5

2. Opseg dokumenta 5

3. Reference 5

4. Predstavljanje arhitekture 5

5. Ciljevi i ograničenja arhitekture 5

6. Pogled na slučajeve korišćenja 5

6.1 Dijagrami slučajeva korišćenja 6

6.2 Kratak opis slučajeva korišćenja 10

6.2.1 Rešavanje kola 10

6.2.1.1 Rešavanje snaga na otpornicima 11

6.2.1.2 Rešavanje struja u granama 11

6.2.1.3 Rešavanje napona na čvorovima 11

6.2.2 Pretraživanje baze 11

6.2.2.1 Pretraživanje po godini 11

6.2.2.2 Pretraživanje po težini 11

6.2.3 Crtanje kola 11

6.2.3.1 Dodavanje čvora 11

6.2.3.2 Dodavanje grane 11

6.2.3.3 Dodavanje komponente 11

6.2.4 Izmena kola 11

6.2.4.1 Brisanje elemenata kola 11

6.2.4.2 Pomeranje elemenata kola 12

6.2.5 Učitavanje kola iz bazu 12

6.2.6 Dodavanje kola u bazu 12

6.2.7 Pamćenje šeme u jpg formatu 12

6.2.8 Prijavljivanje na sistem 12

6.2.9 Menjanje profila 12

6.2.9.1 Menjanje ličnih informacija 12

6.2.9.2 Menjanje poslovnih informacija 12

6.2.10 Korišćenje alata za označavanje 12

6.2.10.1 Dodavanje komentara 12

6.2.10.2 Slobodno crtanje 12

6.2.11 Brisanje profesorskog naloga 12

6.2.12 Dodavanje profesorskog naloga 13

7. Pogled na logičku arhitekturu sistema 13

7.1 Pregled arhitekture – organizacija paketa i podsistema u slojeve 13

7.1.1 Aplikaciona logika 13

7.1.2 Pristup podacima 13

7.1.3 .NET 13

7.1.6 MySQL 13

8. Pogled na procese 14

8.1 Procesi 14

8.1.1 .NET desktop aplikacija 14

8.1.2 OS 14

8.1.3 MySQL Server 14

9. Pogled na raspoređivanje sistema 14

9.1 Aplikacija 15

9.3 DBMS server 15

10. Pogled na implementaciju sistema 15

10.1 Model domena 15

10.2 Šema baze podataka 15

10.3 Komponente sistema 16

10.3.1 Komponente aplikacije 16

10.3.2 Komponente aplikacione logike 16

11. Performanse 19

12. Kvalitet 19

Arhitekturni projekat

# Cilj dokumenta

Cilj ovog dokumenta je detaljni opis arhitekture voltriX simulatora i editora električnih kola.

# Opseg dokumenta

Dokument se odnosi na voltriXsimulator koji će biti razvijen od strane kratkiSpoj-a. Namena sistema je efikasno resavanje i editovanje električnih kola.

# Reference

Spisak korišćene literature:

1. VoltriX – Predlog projekta, kratkiSpoj-VoltriX-01, V1.0, 2019, kratkiSpoj.
2. VoltriX – Plan realizacije projekta, V1.0, 2019, kratkiSpoj.
3. VoltriX– Vizija sistema, V1.0, 2019, kratkiSpoj.
4. VoltriX – Specifikacija zahteva, V1.0, 2019, kratkiSpoj.

# Predstavljanje arhitekture

Arhitektura sistema u dokumentu je prikazana kao serija pogleda na sistem: pogled na slučajeve korišćenja, pogled na logičku arhitekturu sistema, pogled na procese, pogled na razmeštaj komponenti sistema i pogled na implementaciju. Ovi pogledi su predstavljeni odgovarajućim UML dijagramima.

# Ciljevi i ograničenja arhitekture

Ključni zahtevi i sistemska ograničenja koja imaju značajan uticaj na izbor arhitekture i projektovanje sistema su:

1. VoltriXsistem će biti implementiran kao desktop aplikacija zasnovana na .NETtehnologiji i Oracle bazi podataka [4].
2. VoltriX sistem se moze izvršavati na Windows sistemima.
3. Svi zahtevi u pogledu performansi dati u [5] morajubitiuzetiu obzir pri izboru arhitekture i razvoju sistema.
4. Klasifikaciju publikacija treba standardizovati sa postojećom tipologijom propisanom od strane Ministarstva za nauku.

# Pogled na slučajeve korišćenja

U ovom odeljku je dat pogled na slučajeve korišćenja definisane u specifikaciji zahteva [5].

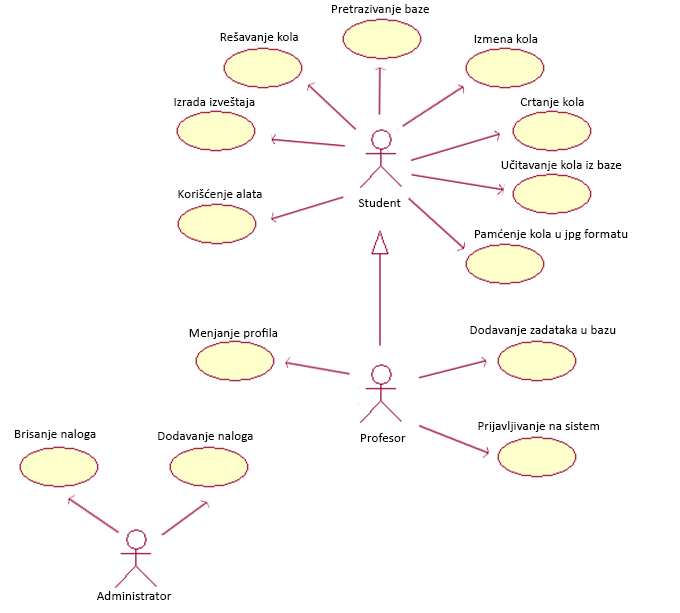
Slučajevi korišćenja VoltriXsistema su:

* Menjanje profila
  + Menjanje ličnih podataka
    - Menjanje imena
    - Menjanje prezimena
    - Menjanje šifre
  + Menjanje poslovnih informacija
    - Menjanje katedre
    - Menjanje fakulteta
* Rešavanje kola
  + Rešavanje snage na otpornicima
  + Rešavanje struje u granama
  + Rešsavanje napona u čvorovima
* Crtanje kola
  + Dodavanje elementa
    - Dodavanje čvora
    - Dodavanje grane
      * Dodavanje komponenata
* Izmena kola
  + Pomeranje
    - Pomeranje cele šeme
    - Pomeranje čvora
  + Brisanje
    - Brisanje grane
    - Brisanje čvora
    - Brisanje komponente
* Pretraživanje baze
  + Pretraživanje po godini zadatka
  + Pretraživanje po težini zadatka
* Korišćenje alata za označavanje
  + Dodavanje komponenata
  + Crtanje po šeme
* Prijavljivanje na sistem
  + Unos username/password
* Brisanje naloga
* Dodavanje naloga

Ove slučajevi korišćenja mogu da iniciraju studenta, profesor ili administrator.

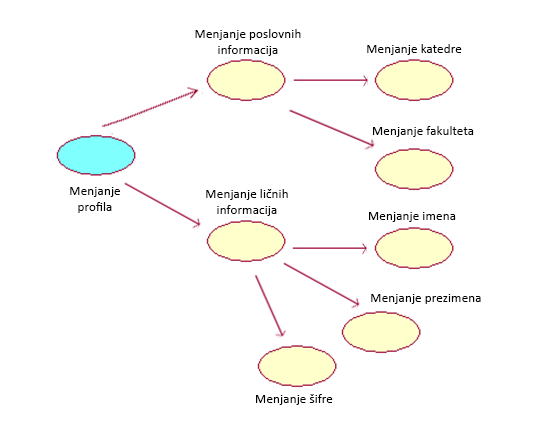
## Dijagrami slučajeva korišćenja

Osnovni UML dijagram koji prikazuje korisnike i slučajeve korišćenja VoltriX sistema prikazan je na sledećoj slici:

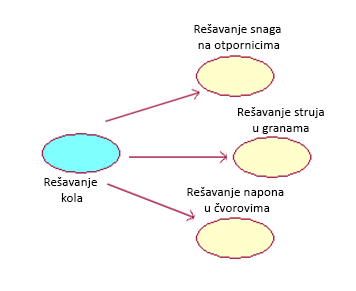


Slučajevi korišćenja *menjanje profila, prijava na sistem, rešavanje kola, pretraživanje baze, izmena kola, crtanje kola, korišćenje alata za označavanje* obuhvataju složenije radnje koje se mogu razložiti dalje razložiti na pojedinačne slučajeve korišćenja.

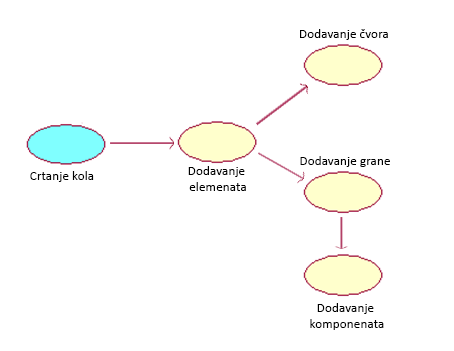
Detaljni UML dijagram za slučaj menjanjanja profila je prikazan na sledećoj slici:



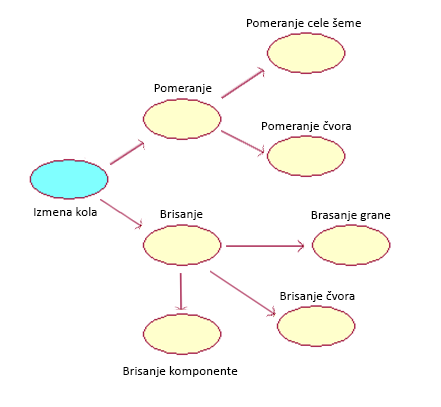
Detaljni UML dijagram za rešavanja kola je prikazan na sledećoj slici:



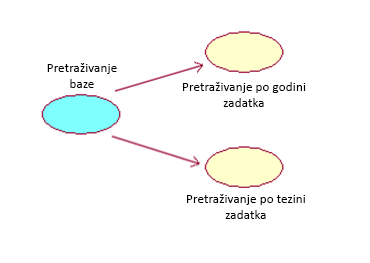
Detaljni UML dijagram za slučaj crtanja kola je prikazan na sledećoj slici:



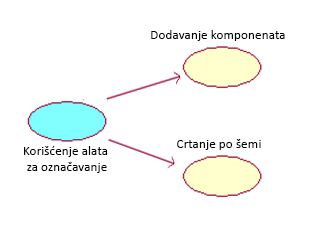
Detaljni UML dijagram za slučaj izmene kola je prikazan na sledećoj slici:



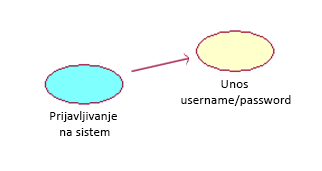
Detaljni UML dijagram za slučaj prikazivanja baze podataka je prikazan na sledećoj slici:



Detaljni UML dijagram za slučaj korišćenja alata za označavanje je prikazan na sledećoj slici:



Detaljni UML dijagram za slučaj prijavljivanja na sistem je prikazan na sledećoj slici:



## Kratak opis slučajeva korišćenja

### Rešavanje kola

Kratak opis: Rešavanje parametara električnog kola (struje u granama, napona na čvorovima, snage na otpornicima).

Akteri koji iniciraju slučaj korišćenja: Student, Profesor.

#### *Rešavanje snaga na otpornicima*

Kratak opis: Korisnik dobija statistički prikaz u željenom formatu (pie, graph...), u vidu nove forme.

Akteri koji iniciraju slučaj korišćenja: Student, Profesor.

#### *Rešavanje struja u granama*

Kratak opis:Korisnik dobija vizuelni prikaz o smeru i intenzitetu struja kroz željenu granu.

Akteri koji iniciraju slučaj korišćenja: Student, Profesor.

#### *Rešavanje napona u čvorovima*

Kratak opis:Korisnik dobija vizuelni prikaz o intenzitetu napona u tom čvoru u odnosu na referentni čvor koji je jasno označen.

Akteri koji iniciraju slučaj korišćenja: Student, Profesor.

### Pretraživanje baze

Kratak opis: Pretrazivanje baze električnih šema po godini zadatka i/ili oceni za koju je predvidjen zadatak.

Akteri koji iniciraju slučaj korišćenja Student ,Profesor .

#### *Pretraživanje baze po godini*

Kratak opis:Pretrazivanje baze električnih šema po godini zadatka.

Akteri koji iniciraju slučaj korišćenja Student ,Profesor .

### 6.2.2.2 Pretraživanje po težini

Kratak opis: Pretrazivanje baze električnih šema po težini zadatka.

Akteri koji iniciraju slučaj korišćenja Student ,Profesor .

### Crtanje kola

Kratak opis:Iscrtavanje elemenata šeme pomoću grafičkog editora.

Akteri koji iniciraju slučaj korišćenja Student ,Profesor .

#### *Dodavanje čvora*

Kratak opis:Klikom na površinu grafičkog editora dodaje se novi čvor na lokaciji kursora miša.

Akteri koji iniciraju slučaj korišćenja Student ,Profesor .

#### *Dodavanje grane*

Kratak opis:Selektovanjem dva čvora (izvornog i odredišnog) otvara se forma za dodavanje i editovanje grane izmedju selektovanih čvorova.

Akteri koji iniciraju slučaj korišćenja Student ,Profesor .

#### *Dodavanje komponenata*

Kratak opis:U formi za dodavanje grane/izmenu grane korisniku se daje mogučost za dodavanje komponente na toj grani.

Akteri koji iniciraju slučaj korišćenja Student ,Profesor .

### Izmena kola

Kratak opis:U formi za dodavanje grane/izmenu grane korisniku se daje mogučost za dodavanje komponente na toj grani.

Akteri koji iniciraju slučaj korišćenja: korišćenja Student ,Profesor.

#### *Brisanje elemenata kola*

Kratak opis:Pomoću grafičkog editora korisnik moze selektovati elemente koje zeli da obriše (komponentu, čvor ili granu ).

Akteri koji iniciraju slučaj korišćenja: korišćenja Student ,Profesor.

#### *Pomeranje elemenata kola*

Kratak opis:Pomoću grafičkog editora korisnik moze selektovati elemente koje zeli da pomerie (čvor ili celo kolo ).

Akteri koji iniciraju slučaj korišćenja: korišćenja Student ,Profesor.

### Učitavanje kola iz baze

Kratak opis: Korisnik moze korišćenjem poseben forme učitati šemu po gore definisanom kriterijumu pretrage..

Akteri koji iniciraju slučaj korišćenja: Student, Profesor.

### Dodavanje kola u bazu

Kratak opis:Prikaz Dodavanje šeme u bazu.

Akteri koji iniciraju slučaj korišćenja: Profesor.

### Pamćenje šeme u jpg formatu

Kratak opis: Selektovanjem opcije u editoru kolo se pamti na lokaciji koju korisnik unese u file explorer-u u vidu jpg slike.

Akteri koji iniciraju slučaj korišćenja: Profesor, Student

### Prijavljivanje na sistem

Kratak opis: U formi za prijavljivanje na sistem korisnik se može prijaviti kao profesor ili kao student

Akteri koji iniciraju slučaj korišćenja: Student, Profesor.

### Menjanje profila

Kratak opis: Profesor moze, po želji, menjati podatke o svom profilu.

Akteri koji iniciraju slučaj korišćenja: Profesor.

#### *Menjanje ličnih podataka*

Kratak opis:Profesor moze, po želji, menjati lične podatke (ime, prezime, šifra).

Akteri koji iniciraju slučaj korišćenja: Profesor.

#### Menjanje poslovnih informacija

Kratak opis:Profesor moze, po želji, menjati poslovni podatke (fakultet, katedra).

Akteri koji iniciraju slučaj korišćenja: Profesor.

### Korišćenje alata za označavanje

Kratak opis: Korisnik može da na posebnoj formi koja sadrži kontrole za unos oznaka na šemi dodaje komentare ili slobodno crta po šemi.

Akteri koji iniciraju slučaj korišćenja Student, Profesor.

#### *Dodavanje komentara*

Kratak opis:Korisnik pomoću kontrole na grafičkom editoru može birati boju, font, providnost i stil za dodati komentar.

Akteri koji iniciraju slučaj korišćenja Student, Profesor.

#### *Slobodno crtanje po šemi*

Kratak opis:Korisnik pomoću kontrole na grafičkom editoru može birati boju, providnost i veličinu linije za crtanje.

Akteri koji iniciraju slučaj korišćenja Student, Profesor.

### Brisanje profesorskog naloga

Kratak opis:Dodavanje U zasebnoj administratorskoj aplikaciji administrator može iz prikazanog niza profesorskih naloga izbrisati željeni.

Akteri koji iniciraju slučaj korišćenja: Administrator.

### Dodavanje profesorskog naloga

Kratak opis: U zasebnoj administratorskoj aplikaciji administrator može unosmo novog profesorskog user name-a stvoriti novi nalog koji kasnije profesor sam menja.

Akteri koji iniciraju slučaj korišćenja Administrator.

# Pogled na logičku arhitekturu sistema

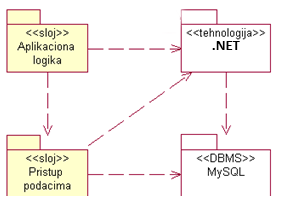
U ovom odeljku je dat pregled logičke arhitekture sistema. Ovaj pogled sadrži opis najznačajnijih klasa, njihove organizacije u pakete i podsisteme, i organizacija podsistema u slojeve. U cilju opisivanja dinamičkih aspekata arhitekture, ovaj odeljak može da uključi opise realizacije najznačajnijih slučajeva korišćenja. Da bi se ilustrovala veza između arhitekturno značajnih klasa, podsistema, paketa ili slojeva moguće je uključiti i odgovarajuće dijagrame klasa.

Logički pogled na VoltriXsistem obuhvata glavna paketa: Aplikaciona logika, Pristup podacima.

Paket *Aplikaciona logika* predstavlja glavni sloj sistema koji sadrži kod aplikacijei editora zadužene za realizaciju funkcionalnosti specifičnih za domen sistema koji se razvija.

Paket *Pristup podacima* sadrži SQL skripte koje predstavljaju interfejs za pristup, dodavanje i ažuriranje podataka koji se čuvaju u bazi podataka.

## Pregled arhitekture – organizacija paketa i podsistema u slojeve



### Aplikaciona logika

sloj

Sloj aplikacione logike je glavni sloj u arhitekturi VoltriXsistema. Sadrži C#source kod koji realizuje funkcionalnost karakterističnu za domen primene sistema i uspostavljaju vezu između editora i sloja za pristup podacima.

### Pristup podacima

sloj

Sloj za pristup podacima se nalazi na dnu arhitekture sistema i sadrži SQL skripte zadužene za pribavljanje, dodavanje i ažuriranje podataka koji se čuvaju u Oracle bazi podataka.

Ovaj sloj zavisi od aplikacionog, tako sto aplkikacioni poziva njega.

### .NET

tehnologija

Tehnologija .NET-a obezbeđuje funkcionalnost editora i svih ostalih funkcionalnosti koje sistem ima(algoritam za izračunavanje parametara kola i editovanje šeme...).

### MySQL

DBMS

MySQL predstavlja sistem za upravljanje bazama podataka koji će se koristiti za realizaciju perzistiranja podataka VoltriXsistema.

# Pogled na procese

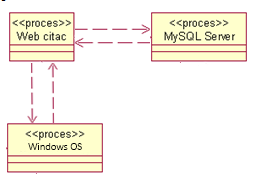
U ovom odeljku je sadržan pogled na procesnu arhitekturu sistema. Ovaj opis treba da sadrži specifikaciju različitih zadataka (procesa i niti) uključenih u rad sistema. Takođe je potrebno dati dijagrame koji pokazuju njihovu interakciju i konfiguraciju. Dodela objekata i klasa na određene zadatke takođe spada u opis procesne arhitekture.

Web aplikacije zasnovane na PHP-u imaju relativno jednostavan procesni model koji je u potpunosti pod kontrolom Web servera. Sa stanovišta projektanta PHP Web aplikacije nije potrebno voditi računa o načinu rada Web servera i načinu izvršavanja skripti.

Ilustracije radi u nastavku je dat opis procesa uključenih u izvršenje PeNcIL portala kao Web aplikacije.

## Procesi

Na sledećem UML dijagramu klasa prikazani su procesi koji učestvuju u izvršenju VoltriX sistema. Dijagram je opšteg tipa i može se primeniti na bilo koju Web aplikaciju zasnovanu na PHP-u i MySQL bazi podataka.



### .NET Desktop Aplikacija

Aplikacija je proces koji se izvršava, i realizuje funkcionalnosti predodredjene njenim kodom na PC-u uz pomoć podrške Windows OS-a koja se realizuje uz pomoć sistemskih poziva.

### OS

Softver koji izvršava i kontroliše rad PC-a, pruža sistemsku podršku procesima koji se na toj mašini izvršavaju pomoću rešavanja sistemskih poziva.

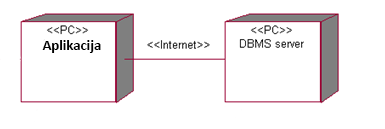
### MySQL Server

MySQL Server je proces koji izvršava funkcionalnost MySQL sistema za upravljanje bazama podataka. Ovaj proces može konkurentno da prihvati određen broj upita, izvrši ih nad bazom podataka i vrati rezultate procesu koji je upite postavio.

# Pogled na raspoređivanje sistema

Pogled na raspoređivanje sistema prikazuje različite fizičke čvorove za najopštiju konfiguraciju sistema. Fizičkim čvorovima koji predstavljaju procesore vrši se dodeljivanje identifikovanih procesa.

Na sledećoj slici dat je UML dijagram raspoređivanja VoltriX sistema.



## Aplikacija

Pristup VoltriX sistemu se obavlja preko PC računara na kojima je isti sistem instaliran. Za povezivanje između aplikacije i DBMS-a se koristi Internet.

## DBMS server

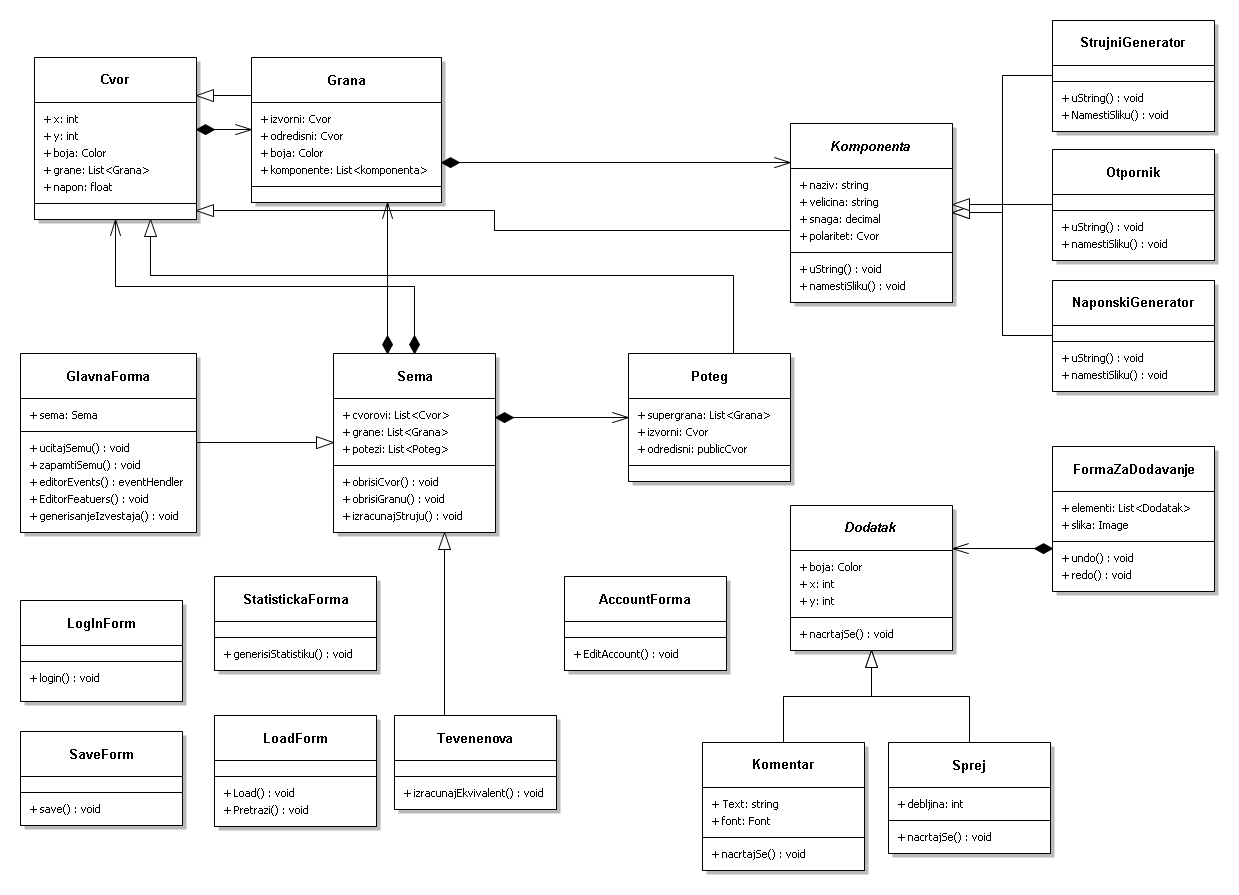
DBMS server je računar na kome se izvršava MySQL Server proces koji realizuje funkcionalnost sistema za upravljanje bazama podataka.

# Pogled na implementaciju sistema

Pogled na implementaciju prikazuje različite aspekte bitne za implementaciju sistema. U slučaju VoltriXsistema ovaj odeljak sadrži model domena, šemu baze podataka i prikaz komponenti sistema razvrstanih u ranije identifikovane pakete.

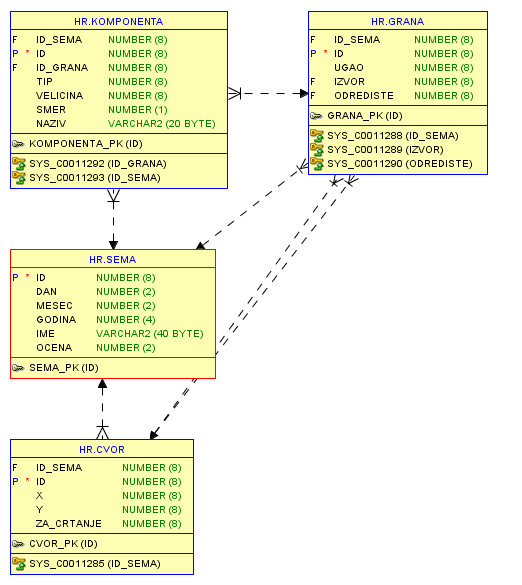
## Model domena

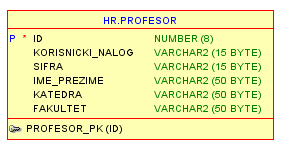
Model domena za koji se VoltriX sistem projektuje je ilustrovan UML dijagramom klasa. U njemu su prikazane glavne klase kao i neki od njihovih atributa i funkcionalnosti.



## Šema baze podataka

Detaljna šema baze podataka je prikazana na sledećem dijagramu. Baza podataka i dijagram su kreirani korišćenjem Oracle SQLDeveloper.





## Komponente sistema

Komponente sistema VoltriX su C# kod i SQL skripte čiji će pregled biti dat po arhitekturnim slojevima. Za ilustraciju će biti korišćeni UML dijagrami komponenti, ali i dijagrami klasa.

### Komponente aplikacije

Arhitektura aplikacije obuhvata dve komponente:



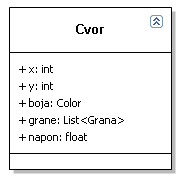
Komponenta **source\_code** implementira aplikacijui komunicira sa SQL bazom.

Komponenta **upiti** predstavlja skup SQL upita koji rešavaju zahteve za podacima i pamćenje podataka koje im **source\_code** šalje.

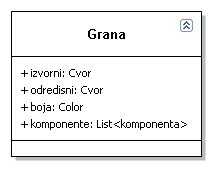
### Delovi aplikacione logike

Glavne klase sistema su čvor, grana, šema, komponenta i poteg. UML dijagrami su dati ispod

1. Čvor je model klasa koja pamti:
   1. položaj čvora na editoru,
   2. njegovu boju i
   3. listu grana kojima je on izvorni čvor
   4. Napon



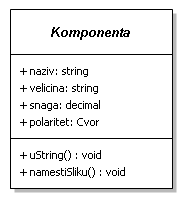
1. Grana je model klasa koja ima
   1. izvorni i odredišni čvor,
   2. listu komponenti koje se nalaze na njoj
   3. Boju



1. Komponenta je roditeljska model klasa koja sadrži:
   1. Vrednost jedinice koju ta komponenta pruža(amperi,volti, omi)
   2. Naziv komponente
   3. Polaritet(usmerena prema kom čvoru)
   4. Snagu

Od bitnijih funkcionalnosti ima:

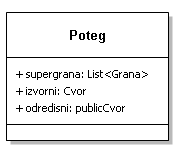
Virtualnu metodu namesti sliku



1. Postoje klase izveden iz komponente (napnski generator, otpornik, strujni generator) od kojih svaka na drugačiji način implemantira metodu namestiSliku(). Svaka od tih klasa ima svoje sitne specifičnosti koje na ovom nivu razmatranja nisu bitne.
2. Klasa poteg je vrsta „kontroler“ klase koja pamti:
   1. svoj izvorni i odredišni čvor i
   2. listu grana (poteg se stvara u procesu izračunavanja struje i ona predstavalja skup, uniju grana povezanih tako da u logički čine granu u elektrotehničkom smislu)
   3. Intenzitet struje kroz tu granu

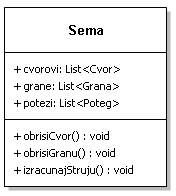
Od bitnijih f-ja ima:

Odredjivanje postojanja strujnog generatora u potegu, izračunavanja ekvivalentne otpornosti, admitanse tog potega.



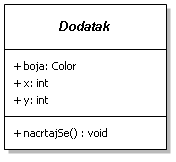
1. Klasa sema predstavlja glavnu klasu VoltriX sistema, koja pamti:
   1. Lista čvorova (one koje je korisnik stvorio u editoru)
   2. Listu grana stvorenih u editoru
   3. Ima referencu na niz potega koji će biti stvoreni u run time-u.

Od funkcionalnosti ima f-ju za izračunavanje struje u granama, brisanje čvora, brisanje grane, pomeranje elemenata

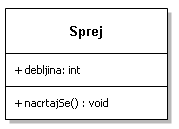
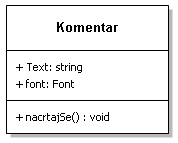


1. Klasa dodatak predstavalja vrstu roditeljske ukrasne klase koja predstavlja model klasu za pamćenje dodataka koje korisnik može unositi po šemi korišćenjem adekvatnog alata, od podataka pamti:
   1. Boju dodatka
   2. Koordinate

Od funkcionalnosti ima virtuelnu metodu nacrtajSe().



1. Iz klase dodatak izvode se klase sprej i komentar od kojih svaka na drugačiji način implemantira metodu nacrtajSe(). Svaka od tih klasa ima svoje sitne specifičnosti koje se mogu videti u gore navedenom UML dijagramu.



Sada ćemo navesi neke od bitnijih formi i njihove funkcionalnosti:

1. LogIn forma ima zadatak da obezbedi pristup sistemu sa adekvatnim privilegijama
2. AccountForma pruža interfejs za predgled profila profesora i editovanje profila u skladu sa zahtevima navedenim u dokumentaciji
3. Save firma i Load forma komuniciraju sa bazom podataka i implementiraju funkcionalnosti odredjene njihovim imenom, Load forma poseduje mogućnost za pretraživanje baze
4. Tevenenova forma, forma koja omogućava računanje ekvivalentne otpornosti izmedju dva čvora u kolu i svodi kola na linearni aktivni dvopol sa ekvivalentnim Tevenenovim naponskim gneneratorom i ekvivalentnim Tevenenovim otpornikom.
5. Forma za ozančavanje od parametara poseduje
   1. Listu dodataka
   2. Sliku šeme po kojoj se crta
   3. Omogućava korisnički interfejs za crtanje dodataka po toj slici i pamcenje te slike u jpg
6. Statisticka forma omogućuje statistički prikaz snaga na otpornicima u nacrtanoj šemi (korisniku se pruža mogućnost biranja više načina predstavljanja)
7. Glavna forma predstavlja osnovnu formu naseg sistema na kojoj se vrši editovanje izračunavanje parametara kola i sa koje se pozivaju Load forma, Statisticka forma, forma za označavanje, Save forma, Account forma u modu profesor i Tevenenova forma. Od podataka forma sadrži instancu šeme. Od funkcionalnosti sadrži:
   1. Editor(i kompletna logika vezana za isti)
   2. Izračunavanje elemenata kola

# Performanse

Izabrana arhitektura softvera podržava brzo i efikasno izračunavanje parametara kola i jednostavno i intuitivno editovanje kola.

1. Izrčunavanje parametara kola za manje od 1sec
2. Intuitivno editovanje šeme.
3. Brz pristup bazi podataka.

Zahtevane performanse su zadovoljene izborom tehnologija na kojima će sistem biti razvijen i definisane hardverske platforme [5].

# Kvalitet

Izabrana arhitektura softvera podržava zahteve u pogledu dostupnosti:

1. Kako je softver zamišljen kao desktop aplikacija, nema posebnih zahteva po pitanju dostupnosti, osim ako korisnik želi pristup arhivi zadataka u tom slučaju potreban je internet konekcija
2. Srednje vreme izmedju dva sukcesivna otkaza ne sme da padne ispod 70 sati.