



УНИВЕРЗИТЕТ У НОВОМ САДУ  
ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА У  
НОВОМ САДУ

---



Наташа Шћекић

# **ВИЗУАЛИЗАЦИЈА БЕЗНАПОНСКИХ ДЕЛОВА НА 2Д ШЕМИ ЕЛЕКТРОДИСТРИБУТИВНЕ МРЕЖЕ**

ДИПЛОМСКИ РАД  
- Основне академске студије -


Нови Сад, 2022.

## КЉУЧНА ДОКУМЕНТАЦИЈСКА ИНФОРМАЦИЈА

|   |  |                               |                |
|---|--|-------------------------------|----------------|
| Редни број, <b>РБР:</b>   |  |                               |                |
| Идентификациони број, <b>ИБР:</b>   |  |                               |                |
| Тип документације, <b>ТД:</b>   | Монографска публикација  |                               |                |
| Тип записа, <b>ТЗ:</b>  | Текстуални штампани документ/ЦД  |                               |                |
| Врста рада, <b>ВР:</b>  | Завршни-bachelor рад   |                               |                |
| Аутор, <b>АУ:</b>   | Наташа Шћекић  |                               |                |
| Ментор, <b>МН:</b>  | Проф. др Драган Иветић   |                               |                |
| Наслов рада, <b>НР:</b>   | ВИЗУАЛИЗАЦИЈА БЕЗНАПОНСКИХ ДЕЛОВА НА 2Д ШЕМИ ЕЛЕКТРОДИСТРИБУТИВНЕ МРЕЖЕ  |                               |                |
| Језик публикације, <b>ЈП:</b>   | Српски(ћирилица)/Српски (латиница)   |                               |                |
| Језик извода, <b>ЈИ:</b>  | Српски/Енглески  |                               |                |
| Земља публикавања, <b>ЗП:</b>   | Србија   |                               |                |
| Уже географско подручје, <b>УГП:</b>  | Војводина  |                               |                |
| Година, <b>ГО:</b>  | 2022   |                               |                |
| Издавач, <b>ИЗ:</b>   | Ауторски репринт   |                               |                |
| Место и адреса, <b>МА:</b>  | Факултет Техничких Наука (ФТН), Д. Обрадовића 6, 21000 Нови Сад  |                               |                |
| Физички опис рада, <b>ФО:</b><br>(поглавља/страна/ цитата/табела/слика/графика/прилога) | 9/41/0/0/27/0/0  |                               |                |
| Научна област, <b>НО:</b>   | Електротехничко и рачунарско инжењерство   |                               |                |
| Научна дисциплина, <b>НД:</b>   | Примењено софтверско инжењерство   |                               |                |
| Предметна одредница/Кључне речи, <b>ПО:</b>   | Електроенергетика, визуализација   |                               |                |
| <b>УДК</b>  |  |                               |                |
| Чува се, <b>ЧУ:</b>   | Библиотека ФТН, Д. Обрадовића 6, 21000 Нови Сад  |                               |                |
| Важна напомена, <b>ВН:</b>  |  |                               |                |
| Извод, <b>ИЗ:</b>   | <p>Рад обрађује приказ електродистрибутивне мреже града Новог Сада и приказ безнапонских делова мреже. Решење је имплементирано у C# програмском језику уз помоћ WPF софтвера компаније Microsoft. Као резултат, добијена је графичка корисничка апликација која приказује шему електродистрибутивне мреже, дозвољава манипулацију ентитетима на мрежи и даје приказ симулације губитка напона у одређеним деловима града.</p> |                               |                |
| Датум прихватања теме, <b>ДП:</b>   |  |                               |                |
| Датум одбране, <b>ДО:</b>   |  |                               |                |
| Чланови комисије, <b>КО:</b>  | Председник:  | Др Александар Селаков, доцент |                |
|   | Члан:  | Др Дуња Врбашки, доцент       | Потпис ментора |
|   | Члан, ментор:  | Др Драган Иветић, ред. проф.  |                |

## KEY WORDS DOCUMENTATION

|  |   |                                    |                |
|--|---|------------------------------------|----------------|
| Accession number, <b>ANO</b> :   |   |                                    |                |
| Identification number, <b>INO</b> :  |   |                                    |                |
| Document type, <b>DT</b> :   | Monographic publication   |                                    |                |
| Type of record, <b>TR</b> :  | Textual material, printed/CD  |                                    |                |
| Contents code, <b>CC</b> :   | Bachelor thesis   |                                    |                |
| Author, <b>AU</b> :  | Nataša Šćekić   |                                    |                |
| Mentor, <b>MN</b> :  | Dragan Ivetić, PhD, full professor  |                                    |                |
| Title, <b>TI</b> :   | VISUALIZATION OF VOLTAGE-FREE PARTS ON THE 2D SCHEME OF THE ELECTRICAL DISTRIBUTION NETWORK   |                                    |                |
| Language of text, <b>LT</b> :  | Serbian (cyrillic script)/Serbian (latin script)  |                                    |                |
| Language of abstract, <b>LA</b> :  | Serbian/English   |                                    |                |
| Country of publication, <b>CP</b> :  | Serbia  |                                    |                |
| Locality of publication, <b>LP</b> :   | Vojvodina   |                                    |                |
| Publication year, <b>PY</b> :  | 2022  |                                    |                |
| Publisher, <b>PB</b> :   | Author reprint  |                                    |                |
| Publication place, <b>PP</b> :   | Faculty of Technical Sciences, D. Obradovića 6, 21000 Novi Sad  |                                    |                |
| Physical description, <b>PD</b> :<br>(chapters/pages/ref./tables/pictures/graphs/appendixes) | 9/41/0/0/27/0/0   |                                    |                |
| Scientific field, <b>SF</b> :  | Electrical and computer engineering   |                                    |                |
| Scientific discipline, <b>SD</b> :   | Power Software Engineering  |                                    |                |
| Subject/Key words, <b>S/KW</b> :   | Electrical energy, visualization  |                                    |                |
| <b>UC</b>  |   |                                    |                |
| Holding data, <b>HD</b> :  | Library of the Faculty of Technical Sciences, D. Obradovića 6, 21000 Novi Sad   |                                    |                |
| Note, <b>N</b> :   |   |                                    |                |
| Abstract, <b>AB</b> :  | <p>The paper deals with the presentation of the electrical distribution network of the city Novi Sad and the presentation of voltage-free parts of the network. The solution is implemented in C# programming language with the help of WPF software from Microsoft company. As a result, a graphical user application was obtained that shows the scheme of the electrical distribution network, allows the manipulation of entities on the network, and provides a representation of the simulation of voltage loss in certain parts of the city.</p> |                                    |                |
| Accepted by the Scientific Board on, <b>ASB</b> :  |   |                                    |                |
| Defended on, <b>DE</b> :   |   |                                    |                |
| Defended Board, <b>DB</b> :  | President:  | Aleksandar Selakov, PhD, docent    | Menthor's sign |
|  | Member:   | Dunja Vrbaški, PhD, docent         |                |
|  | Member, Mentor:   | Dragan Ivetić, PhD, full professor |                |

|   |  |        |
|---|--|--------|
|  | УНИВЕРЗИТЕТ У НОВОМ САДУ • ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА<br>21000 НОВИ САД, Трг Доситеја Обрадовића 6 | Број:  |
|   | <b>ЗАДАТАК ЗА ИЗРАДУ ДИПЛОМСКОГ<br/>(BACHELOR) РАДА</b>  | Датум: |
|   |  |        |

(Податке уноси предметни наставник - ментор)

|                                  |  |
|----------------------------------|--|
| Врста студија:                   | <input type="checkbox"/> Основне академске студије |
| Студијски програм:               | Примењено софтверско инжењерство                   |
| Руководилац студијског програма: | Доц. др Александар Селаков                         |

|          |  |               |           |
|----------|--|---------------|-----------|
| Студент: | Наташа Шћекић                            | Број индекса: | PR10/2018 |
| Област:  | Електротехничко и рачунарско инжењерство |               |           |
| Ментор:  | Проф. др Драган Иветић                   |               |           |

НА ОСНОВУ ПОДНЕТЕ ПРИЈАВЕ, ПРИЛОЖЕНЕ ДОКУМЕНТАЦИЈЕ И ОДРЕДБИ СТАТУТА ФАКУЛТЕТА ИЗДАЈЕ СЕ ЗАДАТАК ЗА ЗАВРШНИ (Bachelor) РАД, СА СЛЕДЕЋИМ ЕЛЕМЕНТИМА:

- проблем – тема рада;
- начин решавања проблема и начин практичне провере резултата рада, ако је таква провера неопходна;
- литература

### НАСЛОВ ДИПЛОМСКОГ (BACHELOR) РАДА:

|   |
|---|
| Визуализација безнапонских делова на 2Д шеми електродистрибутивне мреже |
|---|

### ТЕКСТ ЗАДАТКА:

|   |
|---|
| Проучити алгоритме за визуализацију чворова електроенергетске шеме с могућношћу приказа напонског стања |
|---|

|                                  |              |
|----------------------------------|--------------|
| Руководилац студијског програма: | Ментор рада: |
|                                  |              |

|  |
|--|
| Примерак за: <input type="radio"/> - Студента; <input type="radio"/> - Ментора |
|--|

Образац Q2.НА.15-04 - Издање 2

## Списак коришћених скраћеница

| <i>Скраћеница</i> | <i>Значење скраћенице</i>                      |
|-------------------|--|
| WPF               | Windows Presentation Foundation                |
| API               | Application Programming Interface              |
| CLI               | Common Language Infrastructure                 |
| XML               | Extensible Markup Language                     |
| XSLT              | Extensible Stylesheet Language Transformations |
| HTML              | Hypertext Markup Language                      |
| CSS               | Cascading Style Sheets                         |
| XAML              | Extensible Application Markup Language         |
| UI                | User Interface                                 |
| UTM               | Universal Transverse Mercator                  |
| UML               | Unified Modeling Language                      |
| BFS               | Breadth First Search                           |

## Садржај

|   |    |
|---|----|
| 1. Увод.....                              | 1  |
| 1.1. Визуализација .....                  | 2  |
| 1.2. Електродистрибутивне мреже.....      | 3  |
| 2. Коришћене софтверске технологије ..... | 4  |
| 2.1. Microsoft Visual Studio.....         | 4  |
| 2.2. Windows Presentation Foundation..... | 4  |
| 3. Модел података .....                   | 5  |
| 4. Имплементација .....                   | 7  |
| Исцртавање ентитета .....                 | 7  |
| Исцртавање водова .....                   | 9  |
| Исцртавање додатних облика .....          | 11 |
| Undo, Redo и Clear функционалности .....  | 12 |
| Безнапонски делови мреже .....            | 13 |
| Додатне функционалности .....             | 14 |
| 5. Упутство за употребу апликације .....  | 15 |
| 6. Закључак .....                         | 29 |
| Литература.....                           | 30 |
| Додатак А.....                            | 31 |
| Подаци о кандидату .....                  | 34 |

## 1. Увод

Проблем који решава рад представља софтвер за израду апликације која омогућава дводимензионални приказ мреже електродистрибуције града Новог Сада, са акцентом на визуални део корисничког интерфејса. На исцртаном моделу мреже, кориснику је омогућен низ функционалности – цртање облика и текста по моделу и њихова измена, операције уклањања и враћања исцртаних елемената, уклањање и враћање одређеног дела мреже, промена боја ентитета модела и чување тренутног приказа као слике. Елементи за које корисник може да мења приказ су различити типови постројења и водови.

Посебна функционалност апликације јесте приказ безнапонских делова мреже на основу различитих фактора – у њих спадају штедња, одржавање, промена структуре, преоптерећење мреже и природне непогоде. Услед недовољних ресурса, држава може увести рестрикције електричне енергије у различитим деловима мреже како би се уштедели ресурси. У случају кvara неког елемента у систему, доћи ће до губитка напона. Из тог разлога, требало би поварени елемент заменити новим у што краћем року, односно требало би одржавати мрежу функционалном. Оптимизација система је некад неопходна и углавном се морају вршити промене архитектуре услед чега долази до губитка напона док се не успостави нова структура. У неким деловима мреже постоји пуно елемената на мањој површини и може доћи до преоптерећења мреже. Услед природних непогода, може доћи до оштећења елемената и самим тим и до привременог губитка напона.

У наредним поглављима описан је графички приказ електродистрибутивног система и потребан модел података са описом сваког ентитета. Обрађен је начин исцртавања сваког ентитета система без преклапања истих, као и алгоритам претраге најкраћег пута за исцртавање водова. Објашњене су и додатне функционалности система и начин визуализације безнапонских делова мреже.

## **1.1. Визуализација**

У овом контексту, визуализација је процес графичког представљања података у виду слика, дијаграма или анимација и интеракције над њима, како би се стекао бољи увид у податке. Њена примена омогућава олакшано разумевање представљених апстрактних података.

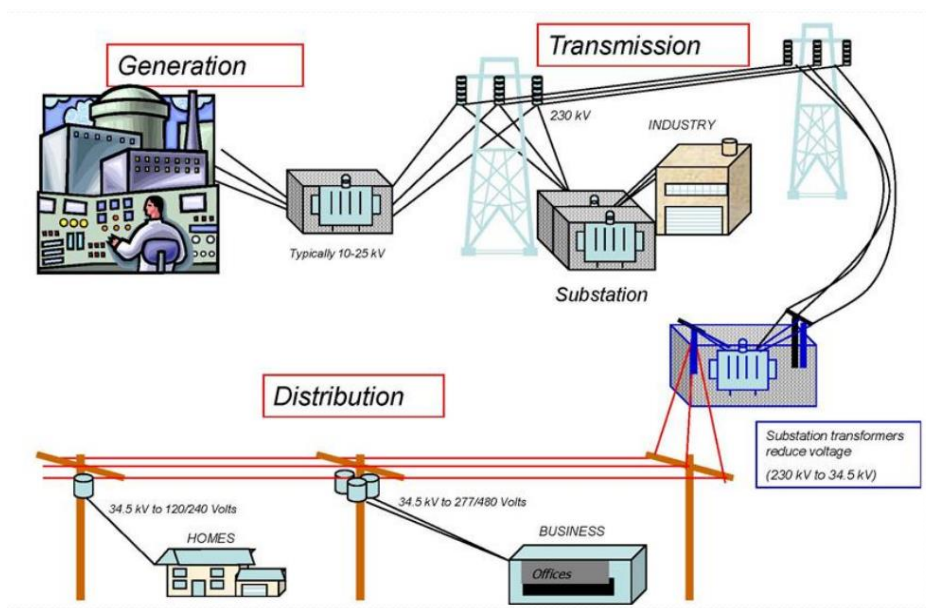
Визуализација информација се базира на коришћењу компјутерски подржаних алата за истраживање велике количине апстрактних података. Њена практична примена у компјутерским програмима укључује селекцију, трансформацију и представљање апстрактних података у облику који људима олакшава разумевање. Важни аспекти визуализације информација су динамика визуалне репрезентације и интерактивност. Кориснику се путем овог начина визуализације омогућава модификација визуализације у реалном времену.

Визуализација података се бави статистичком графиком и геопросторним подацима који се апстрахују у шематски облик. Овај вид визуализације је посебно ефикасан када постоји велики број података. Проучавају се визуалне репрезентације апстрактних података ради побољшања људске спознаје. Апстрактни подаци обухватају и нумеричке и ненумеричке податке. Из академске перспективе, визуализација података се може посматрати као мапирање између оригиналних података и графичких елемената.



## 1.2. Електродистрибутивне мреже

Електроенергетика представља дисциплину у оквиру науке о електрицитету (електрике), у којој се изучавају трансформације различитих облика енергије у електричну енергију, њене унутрашње трансформације, пренос и дистрибуција, као и њене трансформације у употребне облике, односно њене трансформације о које се човек окоришћује. На Слици 1 приказана је структура електроенергетског система са обухваћеним свим подсистемима – производња, пренос, дистрибуција и потрошња.



Слика 1 – Структура електроенергетског система

Различити напони у електроенергетском систему обезбеђују најефикаснији и најекономичнији начин достављања електричне енергије потрошачима. У преносном подсистему, напони су високи како би се избегли већи губици, док су у подсистему дистрибуције напони ниски првенствено ради безбедности људског елемента, али и других фактора.

Електродистрибутивна мрежа представља скуп међусобно спојених постројења и водова који се користе за дистрибуцију електричне енергије. Када се апстрахују елементи, електродистрибутивна мрежа је одређена чворовима и гранама мреже. Дакле, електродистрибутивну мрежу је могуће представити њеним графом. Најважније елементе електродистрибутивне мреже чине трафостанице, чворови, прекидачи и водови. Трафостаница односно разводно постројење је део електроенергетског система који трансформише вредности напона. Чвор представља тачку у којој се спајају водови у систему. Прекидач је једноставни механички уређај који затвара или отвара струјно коло у неком делу електроенергетског система.

## 2. Коришћене софтверске технологије

У пројекту су, као технологије и алати, коришћени *Microsoft Visual Studio* окружење, *Windows Presentation Foundation* и програмски језик *C#* уз *.NET Framework*.

### 2.1. Microsoft Visual Studio

*Microsoft Visual Studio* представља интегрисано развојно окружење компаније *Microsoft*. Користи се за развој рачунарских програма, као и веб-сајтова, веб-апликација и мобилних апликација. За потребе рада, подржава рад са *WPF* [2, 3] апликацијама (за приказ графичких елемената) и рад у програмском језику *C#*. Поред тога, *Visual Studio* користи друге *Microsoft* платформе за развој софтвера, као што су *Windows API*, *Windows Forms*, *Windows Store* и *Microsoft Silverlight*.

*Visual Studio* подржава тридесет шест различитих програмских језика и омогућава уређивачу кода и програму за отклањање грешака да подрже (на различитим нивоима) скоро сваки програмски језик. Уграђени језици укључују *C*, *C++*, *C++/CLI*, *Visual Basic .NET*, *C#*, *F#*, *JavaScript*, *TypeScript*, *XML*, *XSLT*, *HTML* и *CSS*. Подршка за друге језике доступна је преко додатака.

*Microsoft .NET Framework* је софтверска платформа подржана у развојном окружењу *Visual Studio* и укључује велики број готових библиотека кодова за уобичајене проблеме у програмирању и виртуалну машину која управља извршавањем програма. Једна од специфичних функционалности наведене платформе јесте писање програма за дистрибуирана окружења.

### 2.2. Windows Presentation Foundation

*Windows Presentation Foundation* је графички подсистем отвореног кода који је првобитно развила компанија *Microsoft* за приказивање корисничких интерфејса у апликацијама заснованим на *Windows* платформи. *WPF* [2, 3] омогућава олакшано креирање десктоп апликација и обезбеђује конзистентан модел програмирања за израду апликација. Као једна од предности овог графичког подсистема може се издвојити могућност одвајања корисничког интерфејса од пословне логике. *WPF* [2, 3] користи *XAML* [2], изведен из *XML*-а, како би обезбедио декларативни модел за програмирање графичког дела апликације тј. како би дефинисао и повезао различите *UI* елементе. *WPF* [2, 3] апликације могу бити развијене као самостални десктоп програми или као уграђени објекти у веб-страницама. Циљ му је обједињавање низа заједничких елемената интерфејса.

### 3. Модел података

Основу целог пројекта представља датотека *Geographic.xml* у којој су садржани сви подаци који описују електродистрибутивну мрежу Новог Сада и који су потребни за приказ и додатне функционалности над моделом поменуте мреже. Подаци су моделовани путем тагова приказаних у Листингу 1.

```
<? version="1.0" encoding="utf-8"?>
<NetworkModel>
  <Substations>
  <Nodes>
  <Switches>
  <Lines>
</NetworkModel>
```

Листинг 1 - *Geographic.xml* модел података

У датотеци су садржани подаци о свим ентитетима мреже – трафостаницама (*Substations*), чворовима (*Nodes*), прекидачима (*Switches*) и водовима (*Lines*). Сваки ентитет типа трафостаница, чвор или прекидач садржи податке о свом идентификатору (*Id*), називу (*Name*) и географским координатама (*X*, *Y*). Поред наведених, ентитет типа прекидач садржи и податак о тренутном статусу (*Status*). Ентитети типа вод садрже више података – идентификатор (*Id*), назив (*Name*), да ли је подземни или надземни (*IsUnderground*), отпорност (*R*), материјал проводника (*ConductorMaterial*), тип вода (*LineType*), топлотна константа (*ThermalConstantHeat*), идентификатор ентитета из ког вод излази (*FirstEnd*), идентификатор ентитета у који вод улази (*SecondEnd*) и темена (*Vertices*) која садрже координате (*X*, *Y*) тачке (*Point*) где вод мења правац путање. У Листингу 2 приказан је део наведене *xml* датотеке у коме су дефинисане вредности једног прекидача.

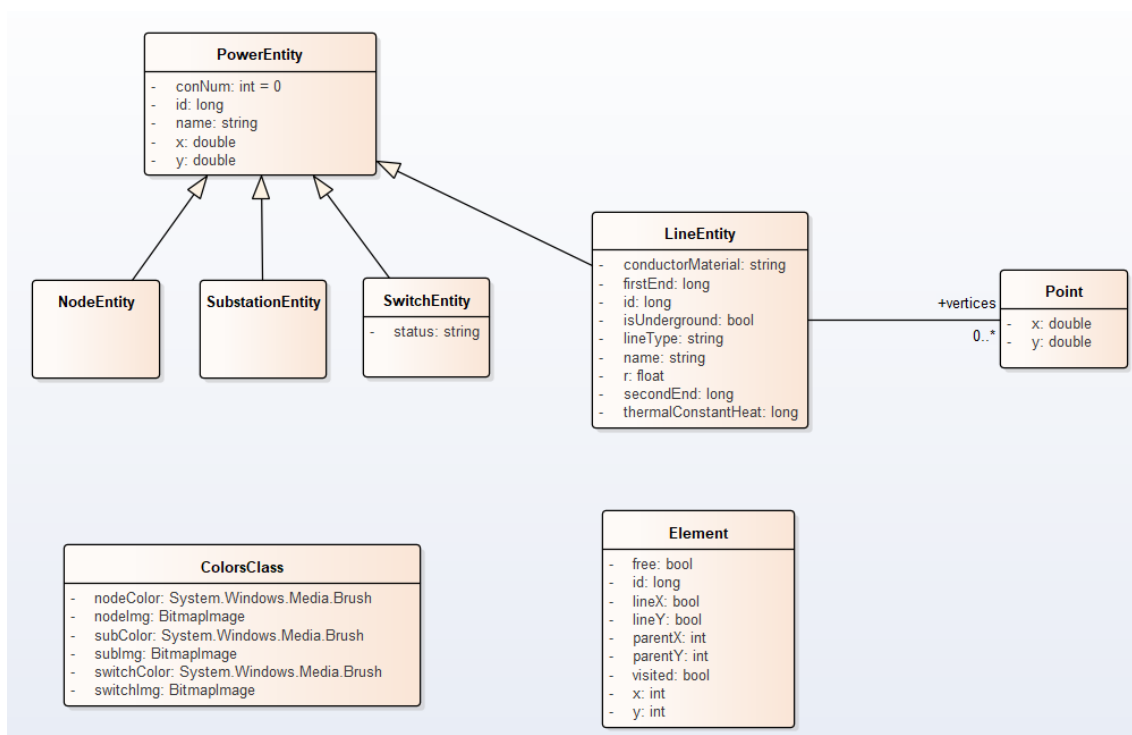
```
<SwitchEntity>
  <Id>40330</Id>
  <Name>disconn_399431983407</Name>
  <Status>Closed</Status>
  <X>417284.31131909514</X>
  <Y>5017643.7788739977</Y>
</SwitchEntity>
```

Листинг 2 – Део *Geographic.xml* датотеке

Како би се подаци о ентитетима могли искористити у имплементационом делу апликације, неопходно је сврстати их у објекте класа. Класе ентитета се могу пронаћи у директоријуму *Models* унутар пројекта у *Visual Studio* развојном окружењу под називима *SubstationEntity*, *NodeEntity*, *SwitchEntity* и *LineEntity*. Ентитети су заједнички описани преко апстрактне класе *PowerEntity* и њу наслеђују све претходно наведене класе. Класа *Point* моделује теме које садржи координате тачке где вод мења правац путање. Све координате унутар *Geographic.xml* датотеке представљене су кроз универзални трансверзални Меркаторов координатни систем (*UTM*). Поред наведених, постоје и две помоћне класе за имплементацију – *Element* и *ColorsClass*. Класа *Element* садржи неопходне податке који ће се користити приликом исцртавања графичких елемената и представља један подеок мреже путем које се мапирају

позиције ентитета. У атрибуте наведене класе спадају позиција подеока на мрежи ( $x$ ,  $y$ ), постојање ентитета у подеоку ( $free$ ), идентификатор постојећег ентитета у подеоку ( $id$ ), постојање хоризонтално исцртаног вода преко подеока ( $lineX$ ), постојање вертикално исцртаног вода преко подеока ( $lineY$ ), посећеност подеока приликом претраге најкраће путање за исцртавање вода ( $visited$ ) и позиција претходно посећеног подеока приликом претраге најкраће путање за исцртавање вода ( $parentX$ ,  $parentY$ ). Класа *ColorsClass* описује боје и слике путем којих ће се приказивати ентитети типа трафостаница, чвор и прекидач. Њени атрибути су боја трафостанице ( $subColor$ ), боја чвора ( $nodeColor$ ), боја прекидача ( $switchColor$ ), слика трафостанице ( $subImg$ ), слика чвора ( $nodeImg$ ) и слика прекидача ( $switchImg$ ).

Све наведене класе, релације између њих и припадајући атрибути, приказани су на Слици 2, преко *UML* класног дијаграма.



Слика 2 – *UML* класни дијаграм модела података

## 4. Имплементација

У овом поглављу је у целости описана имплементација решења приказа електродистрибутивне мреже и свих додатних функционалности апликације.

### Исцртавање ентитета

Након учитавања свих података из *Geographic.xml* датотеке, потребно је превести вредности координата ентитета у одговарајући формат и апроксимирати позиције ентитета на приказу модела. Модел се приказује на платну (*canvas*) одређене ширине и висине. Проналажење разлике између минимума и максимума координата ентитета (по обе осе – *X* и *Y*) даје релативан однос ширине и висине који помаже при апроксимацији координата на платно. Да би се вредности координата правилно поставиле на платно, преведене су у формат географске ширине и висине, а затим је пронађена пропорционална вредност која ће представљати позицију ентитета на платну. У Листингу 3 је приказан начин на који је добијена наведена пропорција.

```
offsetX : (X - minX) = canvas.Width - width
offsetY : (Y - minY) = canvas.Height - height
```

Листинг 3 – Пропорција за проналажење позиције ентитета

Стварна ширина и висина платна су дефинисане преко *canvas.Width* и *canvas.Height*, *width* и *height* су разлике између минималних и максималних координата ентитета након превођења у формат географске ширине и географске висине, (*X - minX*) и (*Y - minY*) су скалиране координате ентитета (такође преведене у формат географске ширине и географске висине) у односу на координатни почетак платна, а *offsetX* и *offsetY* су коначне координате платна где ће се учитани ентитет исцртати.

Међутим, због постојања могућности преклапања ентитета на одређеној позицији, додата је функционалност која проналази најближу слободну позицију за одређени ентитет, како би се преклапање избегло. У Листингу 4 приказана је имплементациона логика коришћена приликом креирања објекта који представља подеок мреже на којем ће се ентитет исцртати.

```
Element el;
if (IsFree(offsetX, offsetY) == false)
{
    int offset = 1;
    while (true)
    {
        el = CheckAround(offsetX, offsetY, offset);
        if (el != null)
        {
            offsetX = el.X;
            offsetY = el.Y;
            break;
        }
    }
}
```

```

        offset++;
    }
    el.Id = se.Id;
    elements.Add(el);
}
else
{
    elements.Add(new Element(offsetX, offsetY, false, se.Id));
}

```

Листинг 4 – Креирање објекта који представља подеок мреже

Логика провере заузетости одређеног подеока и промена њене вредности у случају проналаска слободног подеока приказана је у Листингу 5.

```

public bool IsFree(int offsetX, int offsetY)
{
    if (divs[offsetX, offsetY].Free)
    {
        divs[offsetX, offsetY].Free = false;
        return true;
    }

    return false;
}

```

Листинг 5 – Логика провере заузетости подеока

На Листингу 6 приказана је функционалност проналажења околних подеока у односу на тренутни.

```

public Element CheckAround(int x, int y, int offset)
{
    for (int i = x - offset; i <= x + offset; i++)
    {
        if (i < 0)
            continue;
        else if (i >= divCountX)
            continue;

        for (int j = y - offset; j <= y + offset; j++)
        {
            if (j < 0)
                continue;
            else if (j >= divCountY)
                continue;

            if (IsFree(i, j))
                return divs[i, j];
        }
    }

    return null;
}

```

Листинг 6 – Функционалност проналажења околних подеока

За сваки ентитет типа трафостаница, чвор и прекидач, врше се функције из Листинга 4, 5 и 6. Прво се проверава да ли је одређена позиција на платну слободна помоћу функције *IsFree* која проверава вредност атрибута *free*. Уколико је позиција слободна, она се заузима и нови објект класе *Element* се додаје у листу свих постојећих елемената *elements*. У супротном, позива се функција *CheckAround* итеративно. Функцији *CheckAround* прослеђује се позиција ентитета уз померај који означава удаљеност од првобитне позиције. У принципу, радијално се проверава заузетост сваког подеока све док се не наиђе на први слободан подеок. Након тога, уписују се

одговарајуће вредности атрибута и нови објект класе *Element* се додаје у листу свих постојећих елемената *elements*.

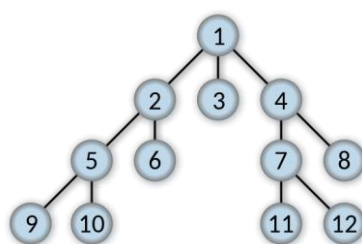
Ентитети се исцртавају на платно помоћу објекта класе *Ellipse* која припада уграђеној библиотеци окружења *Visual Studio*, при чему су одређени атрибути постављени на одређене вредности. Висина и ширина елипсе постављене су на вредност 1. Боја елипси је иницијално постављена у зависности од броја конекција ентитета (наранџаста за мање од три конекције, црвена за број конекција између три и пет, бордо за више од пет конекција). Атрибут *ToolTip* садржи информације о ентитету које ће се приказивати приликом преласка курсором преко њега.

## Исцртавање водова

Следећи корак је учитавање ентитета типа вод из *Geographic.xml* датотеке. Пролази се кроз све учитане водове и посматрају се идентификатори из атрибута *FirstEnd* и *SecondEnd*. Уколико се наиђе на дупликат, вод се прескаче. Такође, прескачу се водови који у атрибуту *FirstEnd* или *SecondEnd* садрже непостојећи идентификатор ентитета.

Исцртавање водова на платно је имплементирано кроз алгоритам претраге у ширину (*Breadth First Search, BFS*). Зарад лакшег разумевања, алгоритам је приказан у Додатку А. *BFS* представља алгоритам за претрагу у структурама података у облику стабла. Почиње од одређеног чвора стабла који се назива изворни чвор и истражује све суседне чворове пре него што пређе на чворове на следећем нивоу. Процес се обавља итеративно све док се не добије жељени резултат.

На Слици 3 је приказан пример графа у облику стабла с редоследом посећености чворова који је добијен путем *BFS* алгоритма.



Слика 3 – Редослед посећености чворова у графу

За сваки вод се позива функција *BFS* која враћа ред елемената који чине путању вода, уколико је пронађена. Првобитно се атрибут *visited* почетног елемента мења, тако да означава да је подеок посећен, а затим се елемент поставља у два реда – један за обраду и један у којем ће се чувати сви посећени елементи из тренутног позива функције (копирани ред). За све суседе тренутно обрађиваног елемента се проверава да ли су посећени и, уколико нису, промениће се вредност атрибута *visited* и поставиће се у редове. Уз то, атрибути *parentX* и *parentY* се постављају на координате родитељског елемента тј. елемента чији је сусед био тренутни елемент. Поступак се врши итеративно за сваки претходно посећени елемент тако што се скине први елемент из реда за

обраду и проверавају се његови суседи. У свакој итерацији се проверава да ли је нађен крајњи елемент или су обрађени сви елементи из реда за обраду. Уколико су обрађени сви елементи, значи да путања није пронађена и, у том случају, свим претходно посећеним елементима из реда морају се вратити вредности атрибута *visited*, *parentX* и *parentY*, како би постали слободни за тражење путање у наредним позивима функције. У случају да је путања пронађена, обрада реда престаје. Редослед елемената из копираног реда се обрће, а затим се скидају сви елементи са почетка док се не дође до крајњег елемента вода. То су елементи који су унети у ред након уношења крајњег елемента, а пре његове обраде. Након тога, итеративно се скидају елементи са копираног реда. Сваки елемент чије су координате једнаке са атрибутима *parentX* и *parentY* из претходне итерације поставља се у ред који представља повратну вредност функције, а осталим елементима се враћају вредности атрибута *visited*, *parentX* и *parentY*. Након обраде свих водова, исцртавају се путање на моделу где се линије крећу само под правим углом и не постоје пресеци. Водови исцртани преко *BFS* алгоритма су приказани на Сlici 4.



Слика 4 – Водови исцртани путем *BFS* алгоритма

Уколико путања вода није пронађена преко *BFS* алгоритма, вод се исцртава као једна линија (уколико је нека од координата почетног и крајњег елемента једнака) или као две линије под правим углом.

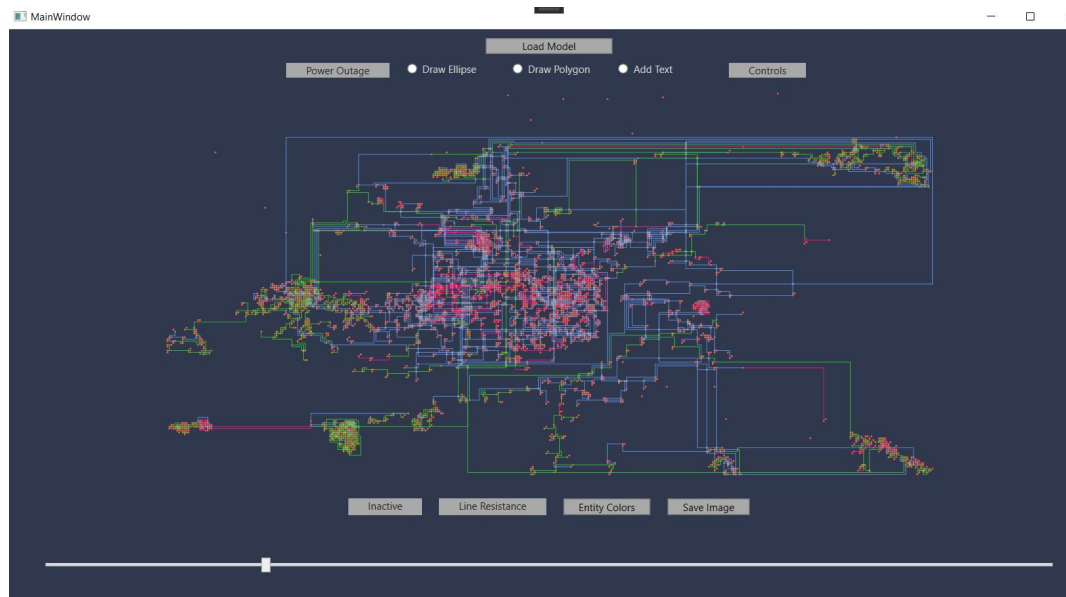
Сви водови се исцртавају преко објекта класе *Polyline* која припада уграђеној библиотеци окружења *Visual Studio*, уз постављање вредности одређених атрибута. Боја ентитета типа вода се бира на основу атрибута *ConductorMaterial*, а атрибут *ToolTip* садржи информације о воду које ће се приказивати приликом преласка курсором преко њега. Такође, имплементирана је функционалност бојења крајњих ентитета вода у црно приликом десног клика на вод, тако што се пролази кроз све елипсе док се не пронађу повезани ентитети и мења се вредност атрибута за боју елипсе.

Након исцртавања свих водова, створиће се пресеци који се морају означити. Пресеци се проналазе преко атрибута *lineX* и *lineY* из објекта класе *Element* који се налази на истој позицији као и сам пресек. Атрибути *lineX* и *lineY* означавају постојање усправног и/или хоризонталног дела



вода на одређеним координатама. Пресеци су означени као знак „X” путем објекта класе *Polygon* која припада уграђеној библиотеци окружења *Visual Studio*.

Коначан изглед исцртаног модела приказан је на Слици 5.



Слика 5 – Коначан изглед модела

## Исцртавање додатних облика

У апликацији је омогућено исцртавање додатних облика – елипси и полигона. Унутар облика, опционо се може поставити и текст. Такође, текст је могуће исписати независно од облика, на било ком делу платна. Корисник левим и десним кликовима на платно одабира позицију и позива исцртавање сваког облика, што ће детаљније бити објашњено у упутству за употребу апликације.

Десним кликом на платно, позива се функција која проверава одређене услове и, у зависности од њихових вредности, реагује складно.

Уколико је изабрано исцртавање елипсе, отвара се нови прозор у којем корисник попуњава вредности атрибута елипсе. Након одабира свих вредности атрибута, кликом на дугме *Submit* се позива функција која новокреирани објект елипсе и евентуални текст унутар ње смешта у ново платно, које се потом убацује у оригинално платно на ком се налази модел електродистрибутивне мреже. Такође, у овој функцији се додаје догађај на елипсу који ће касније омогућити измену атрибута елипсе кликом на њу.

Уколико је изабрано исцртавање полигона, при сваком позиву функције се чува позиција платна на коју је кликнуто у листу која ће заправо садржати сва темена полигона.

Уколико је изабрано исписивање текста, позивом функције се отвара нови прозор у којем корисник попуњава вредности атрибута текста. Након одабира свих вредности атрибута, кликом

на дугме *Submit* се позива функција која новокреирани објект текста смешта у ново платно, које се потом убацује у оригинално платно на ком се налази модел електродистрибутивне мреже. Такође, у овој функцији се додаје догађај на текст који ће касније омогућити измену атрибута текста кликом на њега.

Левим кликом на платно, позива се функција која проверава одређене услове и, у зависности од њихових вредности, реагује складно.

Уколико је изабрано исцртавање полигона, позивом функције се учитавају сва претходно креирана темена полигона, а затим се отвара нови прозор у којем корисник попуњава вредности атрибута полигона. Након одабира свих вредности атрибута, кликом на дугме *Submit* се позива функција која новокреирани објект полигона и евентуални текст унутар њега смешта у ново платно које се потом убацује у оригинално платно на ком се налази модел електродистрибутивне мреже. Такође, у овој функцији се додаје догађај на полигон који ће касније омогућити измену атрибута елипсе кликом на њу.

Свако исцртавање облика или постављање текста на платно је имплементирано додавањем програмских објеката у листу *Children* која припада платну. Кликом на платно, позивају се одређене програмске функције које преузимају координате позиције на коју је кликнуто.

## ***Undo, Redo u Clear функционалности***

Одабиром опције *Controls*, кориснику се преко менија нуде опције *Undo, Redo* и *Clear*.

Опција *Undo* омогућава уклањање последње исцртаног облика на платну, док опција *Redo* враћа последње уклоњен облик на платно. Преко одабира опције *Clear* бришу се сви претходно исцртани облици на платну, тако да остаје само модел електродистрибутивне мреже. У том случају, опција *Redo* би вратила све облике назад.

Одабиром опције *Undo*, позива се функција која прво проверава да ли операција може да се изврши, а затим, уколико су сви неопходни услови испуњени, чува последње исцртани облик односно исписани текст преко модела дистрибутивне мреже и након тога га уклања са платна и из листе свих тренутно исцртаних облика и исписаних текстова.

Одабиром опције *Clear*, позива се функција која чува све исцртане облике и исписане текстове преко модела дистрибутивне мреже у једну листу и након тога их уклања са платна и из листе свих тренутно исцртаних облика и исписаних текстова.

Одабиром опције *Redo*, позива се функција која прво проверава да ли је претходно извршена операција *Clear*. У случају да јесте, преузимају се сви објекти из листе која чува објекте након операције *Clear* и враћају се на платно, као и у листу тренутно исцртаних облика тј. исписаних текстова. Такође, сви елементи унутар листе која чува објекте након операције *Clear* се бришу. У супротном случају, ако су задовољени услови, последње уклоњени елемент се враћа на платно и у листу тренутно исцртаних облика и исписаних текстова.

## Безнапонски делови мреже

Апликација имплементира симулацију престанка дистрибуције електричне енергије услед штедни, одржавања мреже, промена структуре мреже, преоптерећења мреже и природних непогода.

Одабиром једне од опција, софтвер насумично одређује део града у којем ће доћи до престанка дистрибуције електричне енергије. Одабир се врши тако што се насумично одреде два броја – први између нула и седам, а други између нула и три. Тиме се добијају тридесет две могуће комбинације та два броја, које ће представљати различите делове мреже. С обзиром да је размера платна 2:1, када се подели број подеока мреже по X-оси са осам, односно број подеока мреже по Y-оси са четири, добија се корак који представља размак између почетака два дела мреже. У Листингу 7 је приказан начин на који долазимо до свих ентитета који се налазе у одређеном делу мреже.

```
for (int i = sirina * x; i < sirina * x + sirina; i++)
{
    for (int j = visina * y; j < visina * y + visina; j++)
    {
        foreach (Element el in elements)
        {
            if (el.X == i && el.Y == j)
            {
                SavingsIds.Add(el.Id);
            }
        }
    }
}

foreach (KeyValuePair<Ellipse, long> pair in ellipses)
{
    if (SavingsIds.Contains(pair.Value))
        pair.Key.Fill = Savings.Foreground;
}
```

Листинг 7 – Промена боје ентитета без напона

Прва петља пролази кроз све подеоке X-осе, а друга кроз све подеоке Y-осе одређеног дела мреже. На тај начин ће се итерирати кроз све подеоке одређеног дела мреже. У трећој петљи се пролази кроз све постојеће елементе на платну и пореде се позиције тренутног елемента и тренутних итератора из прве две петље. Уколико су вредности једнаке, елемент се налази у траженом делу мреже и додаје се у листу која чува све пронађене ентитете. Након проласка кроз све подеоке одређеног дела мреже, у листи ће се налазити сви ентитети до којих електрична енергија не допире и промениће им се боја. У Листингу 7 је приказан део кода који симулира штедњу електричне енергије, али и за све остале факторе нестанка напона имплементирано је решење на сличан начин.



## 5. Упутство за употребу апликације

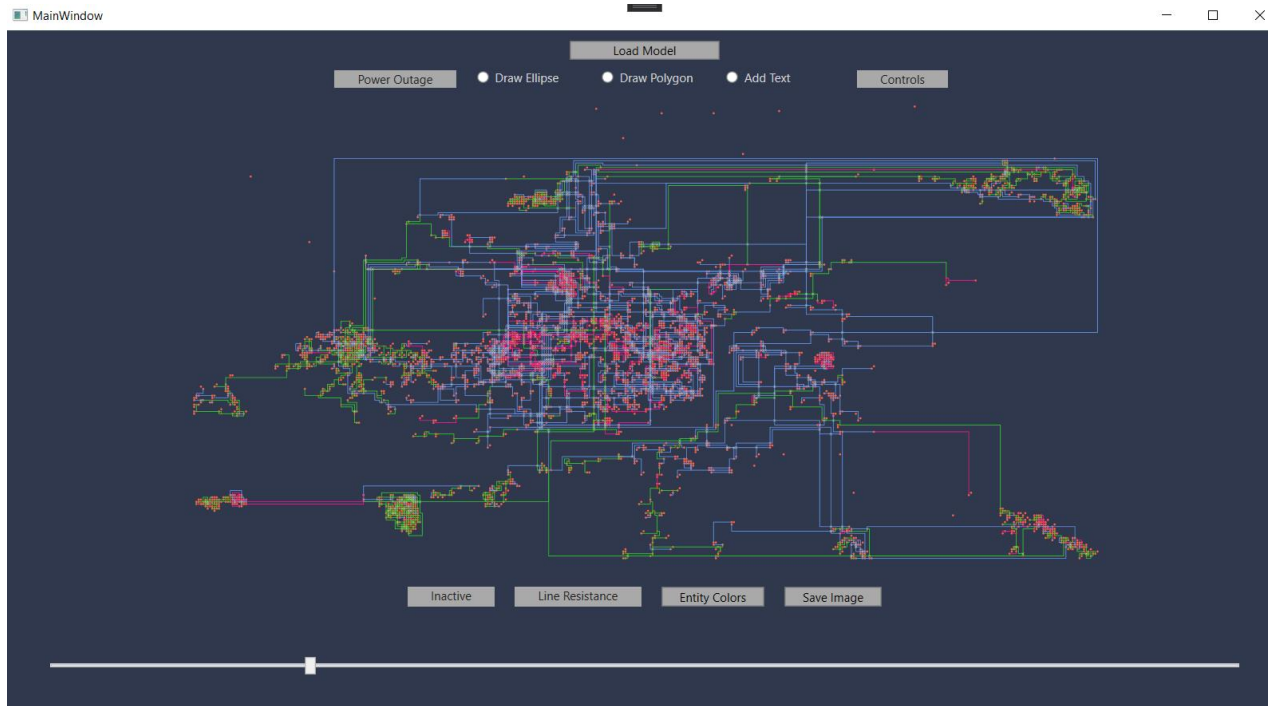
Апликација коју описује овај рад је намењена првенствено за приказ модела електродистрибутивне мреже Новог Сада.

На Слици 6 приказан је почетни изглед апликације након покретања, где се могу уочити све функционалности које су омогућене кориснику.



Слика 6 – Почетни изглед апликације

Да би се прочитао и исцртао модел електродистрибутивне мреже Новог Сада, неопходно је да корисник кликне на дугме *Load Model* које се налази на врху прозора. Након тога, сви ентитети и водови електродистрибутивне мреже биће визуално приказани, као на Слици 7. Корисник преко траке за померање на дну прозора може да увећава и умањује платно, по жељи. Такође, након увећања, појавиће се траке за померање на дну и са десне стране платна на ком се исцртава модел и оне омогућавају промену увећаног дела који се посматра.



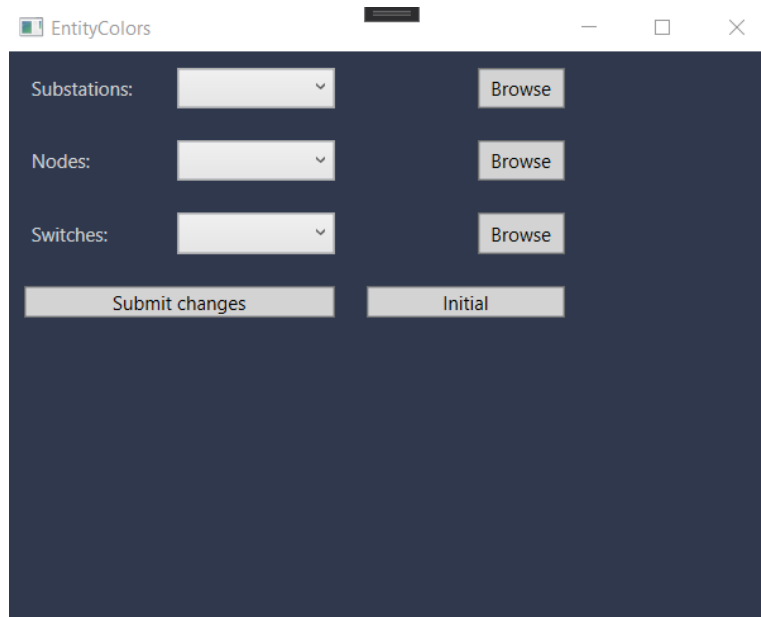
Слика 7 – Изглед апликације након клика на *Load Model* дугме

Такође, на Слици 7 се може приметити да сви ентитети и водови имају предефинисану боју.

Сви ентитети типа трафостаница, чвор и прекидач имају предефинисану боју којом се приказују на основу броја конекција са другим ентитетима које поседују. Тако су ентитети са мање од три конекције приказани наранџастом бојом, ентитети са три до пет конекција приказани су црвеном бојом, а ентитети са више од пет конекција су приказани бордо бојом.

Боје водова се разликују на основу материјала од ког су сачињени. Водови направљени од челика приказани су плавом бојом. Водови направљени од алуминијума приказани су зеленом бојом. Водови направљени од бакра приказани су розе бојом. Сви остали водови који нису направљени од поменутих материјала приказују се сивом бојом.

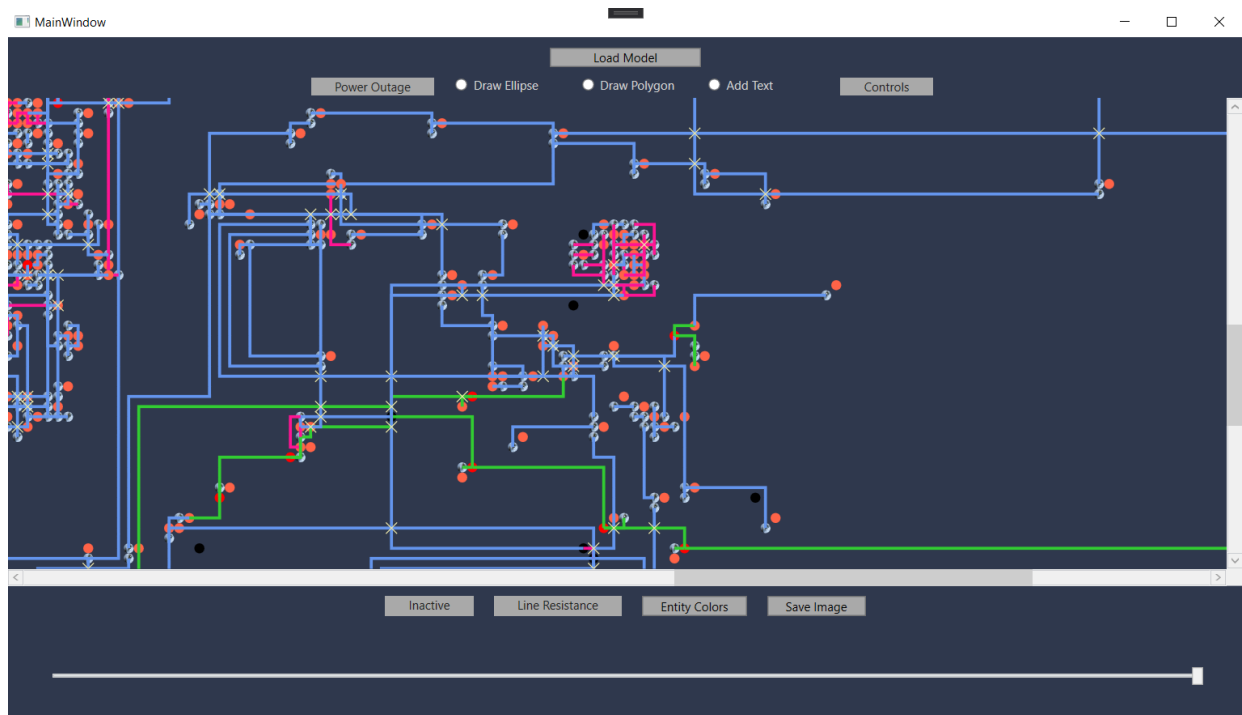
Иако су боје ентитета предефинисане, корисник има могућност да их промени по жељи. Боје ентитета је могуће променити кликом на дугме *Entity Colors*, након чега се отвара нови прозор, приказан на Слици 8.



Слика 8 – Прозор за промене начина приказа ентитета

Корисник преко падајућих листи може променити боју одређеног ентитета или кликом на дугме *Browse* изабрати слику за приказ одређеног типа ентитета. Након клика на дугме *Submit Changes*, све промене које је корисник одабрао ће се аутоматски применити. Уколико се не одабере ни боја, а ни слика за неки ентитет, приказ тог ентитета се неће мењати. Такође, кликом на дугме *Initial* корисник може да врати приказ свих ентитета на иницијални.

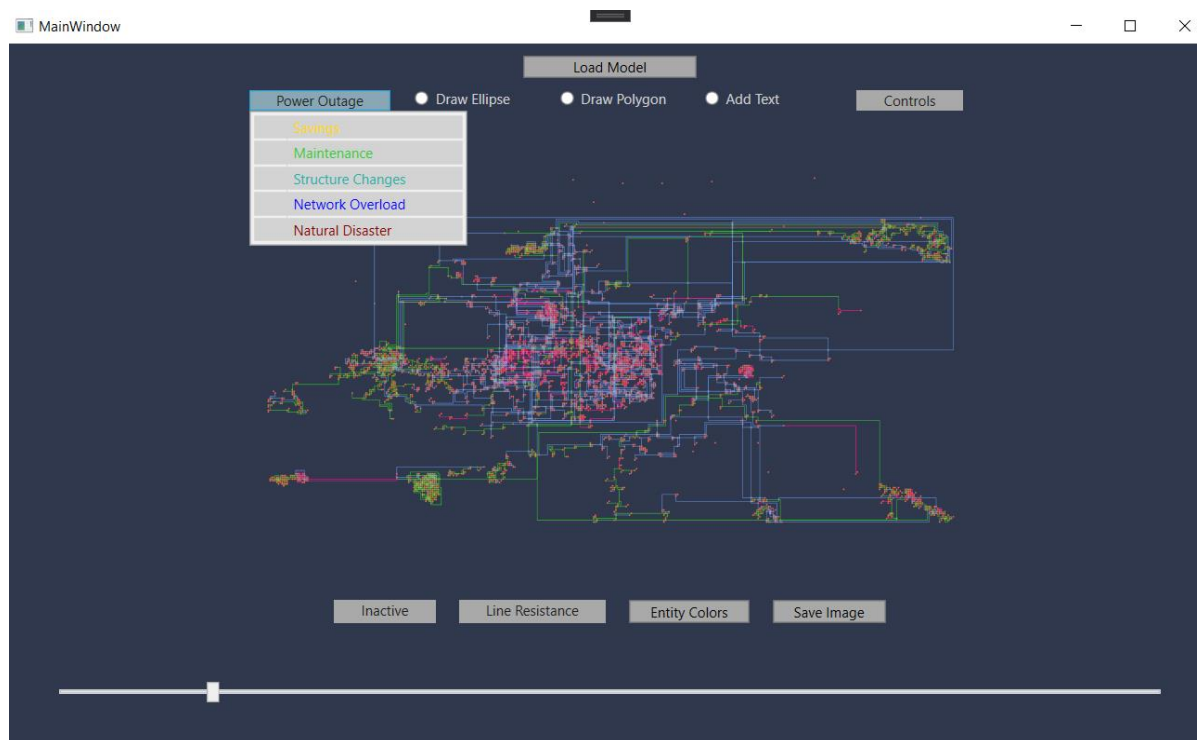
На Сlici 9 је приказан пример у којем су трафостанице постављене на црну боју, прекидачи на слику, док чворови нису измењени, приказани су иницијалном бојом.



Слика 9 – Измењен начин приказа ентитета

Уколико корисника интересује које су крајње тачке неког вода тј. који су ентитети повезани одређеним водом, а није их могуће лако препознати, десним кликом на вод ће се крајњи ентитети обојити у црно. Ова функционалност знатно олакшава рад у деловима мреже са великом густином ентитета и водова.

Апликација кориснику нуди могућност приказа насумично одабраних неактивних делова мреже кликом на опцију *Power Outage*. Селектовањем ове опције, кориснику се приказује мени са ставкама које представљају различите узроке губитка напона, што је приказано на Слици 10.



Слика 10 – Мени са узроцима губитка напона

Свака од опција у менију је обојена одређеном бојом. Том бојом ће бити обојени ентитети који су захваћени одређеним узроком губитка напона.

Пример безнапонског дела мреже услед преоптерећења мреже приказан је на Слици 11, где су захваћени ентитети обојени у плаво.





Слика 11 – Приказ безнапонског дела мреже услед преоптерећења

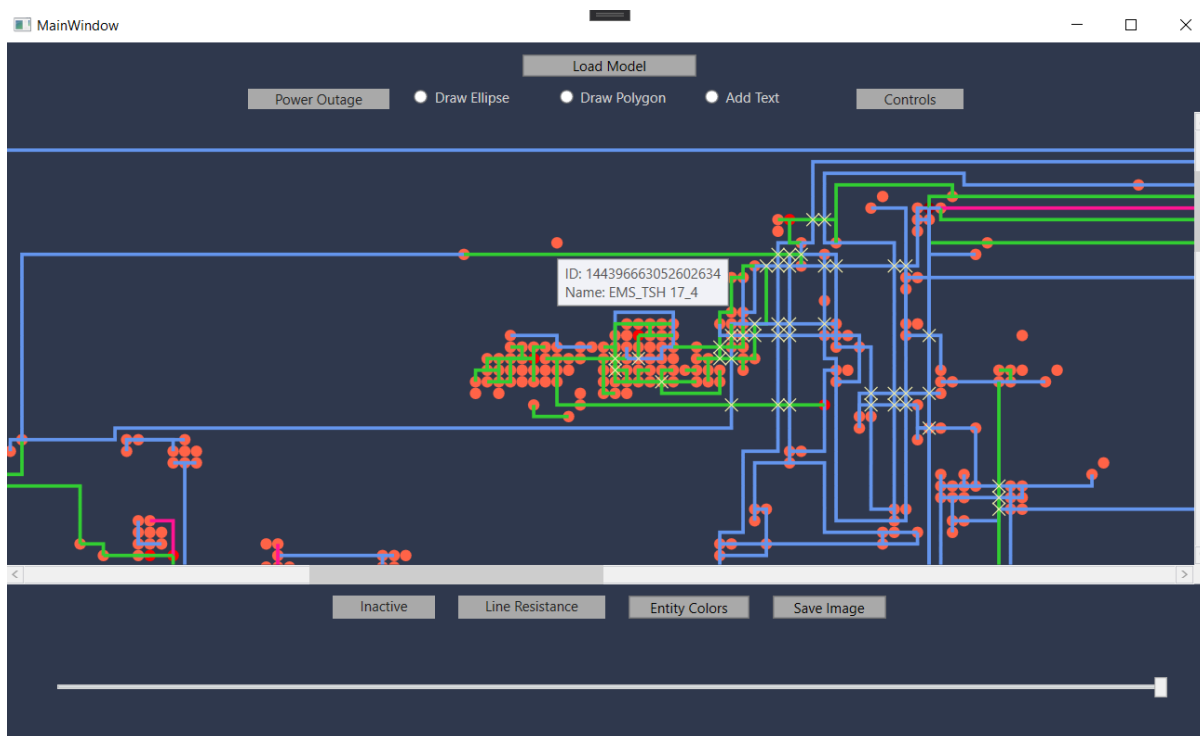
Такође, корисник може и да уклони приказ безнапонског дела мреже кликом на узрок губитка напона у менију за који више не жели да се приказује.

Омогућен је и приказ више безнапонских делова мреже услед различитих одабраних фактора, што је и уоквирено на Слици 12.



Слика 12 – Приказ више безнапонских делова мреже

Уколико корисника интересују информације о ентитетима и водовима на приказаној мрежи, довољно је да само постави курсор на ентитет/вод од интереса. Информације о ентитету/воду ће се приказати као на слици 13. Након склањања курсора са позиције, информације више неће бити видљиве.



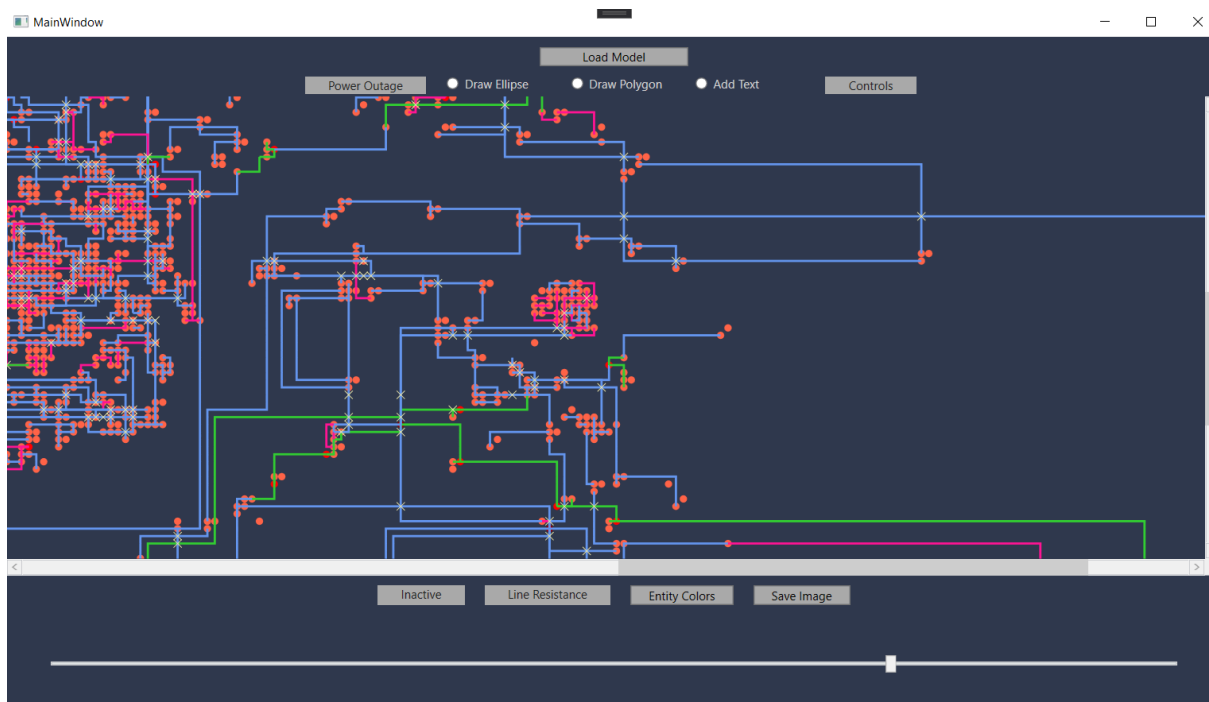
Слика 13 – Приказ информација о ентитету

Одабиром опције *Inactive*, кориснику се преко менија нуде опције *Hide* и *Show*, које омогућавају уклањање неактивног дела мреже и његово враћање. Наведене опције су приказане на Сlici 14.

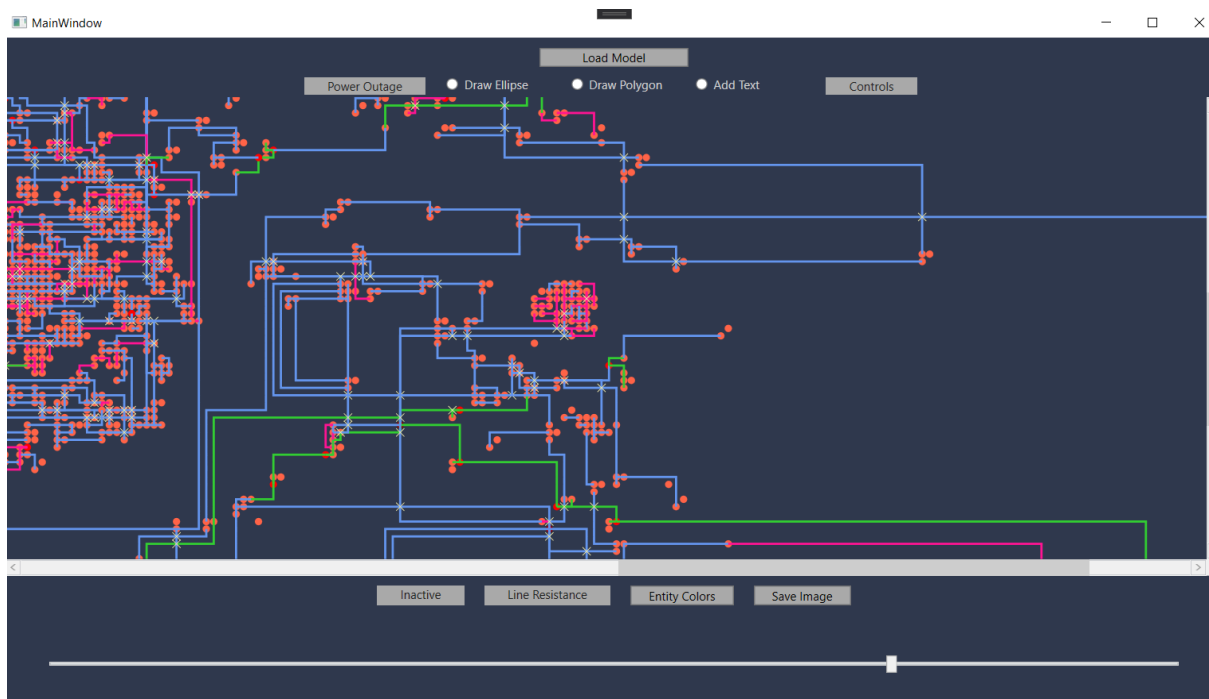


Слика 14 – *Inactive* опције

На Слици 15 је приказан део модела без неактивног дела мреже, док је на Слици 16 приказан део модела са свим ентитетима.



Слика 15 – Део модела без неактивног дела мреже



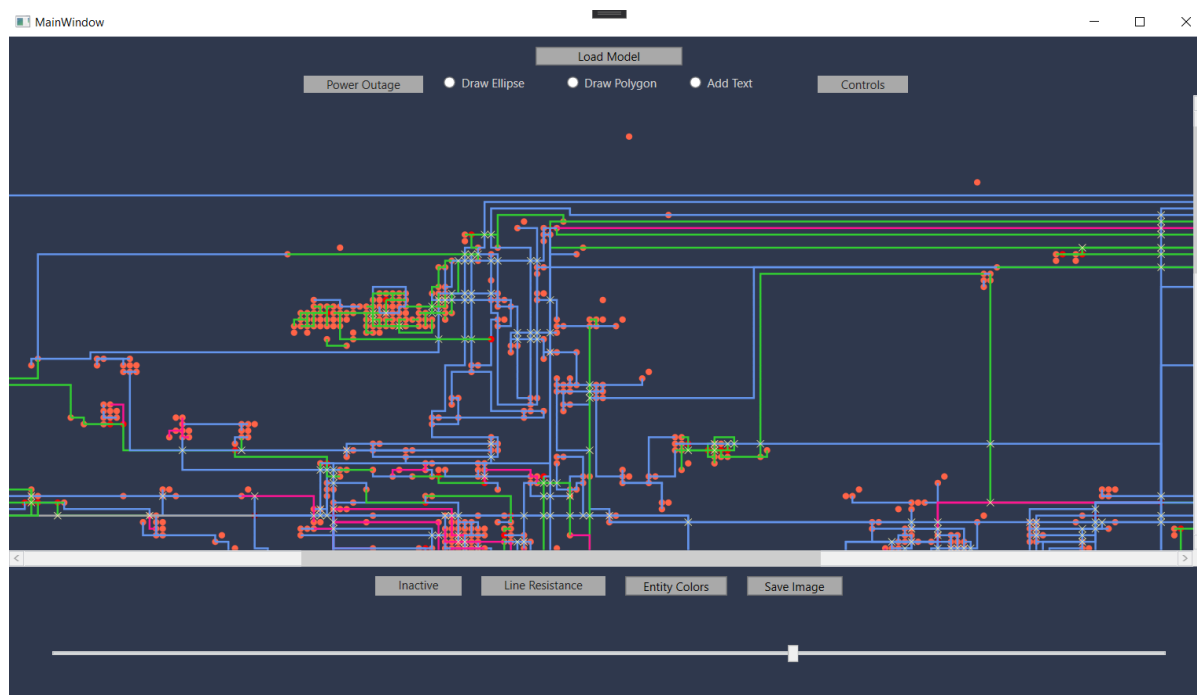
Слика 16 – Део модела са свим ентитетима

Одабиром опције *Line Resistance*, кориснику се преко менија нуде опције *Red/Orange/Yellow* и *Initial* путем којих је омогућена промена боја водова у односу на вредност отпорности вода и враћање на иницијалне боје. Наведене опције су приказане на Слици 17.

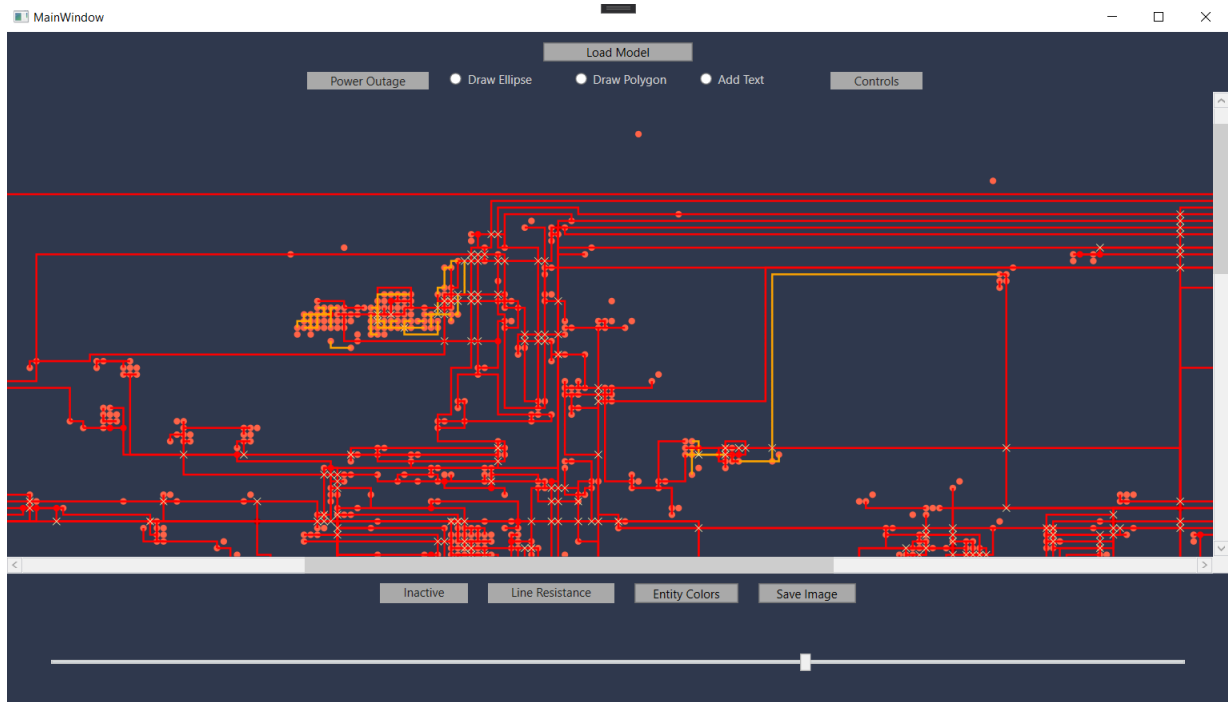


Слика 17 - *Line Resistance* опције

Пример разлике иницијалних боја водова и боја водова на основу отпорности се може видети на Слици 18 и на Слици 19.



Слика 18 – Иницијалне боје водова

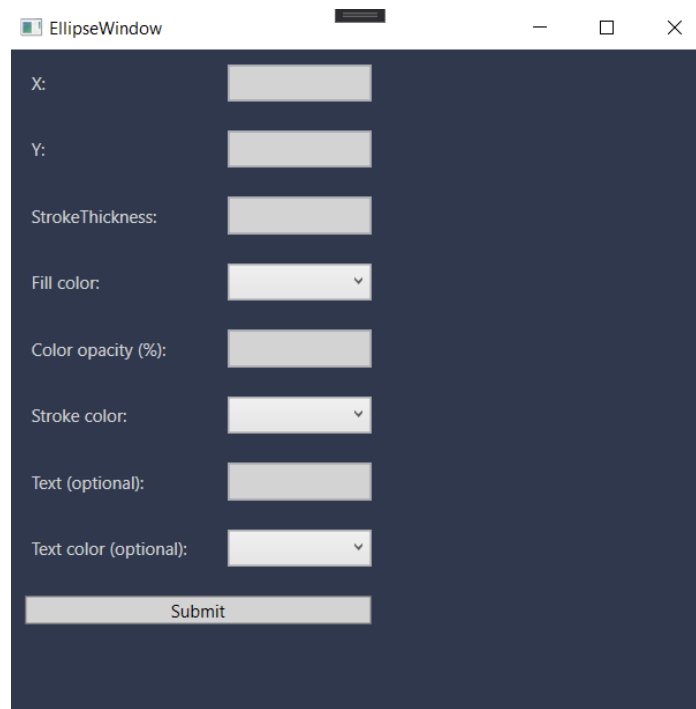


Слика 19 – Боје водова на основу отпорности

Кликом на дугме *Save Image* кориснику је омогућено трајно чување тренутног приказа на платну као слике. Слика ће бити сачувана у самом фолдеру апликације.

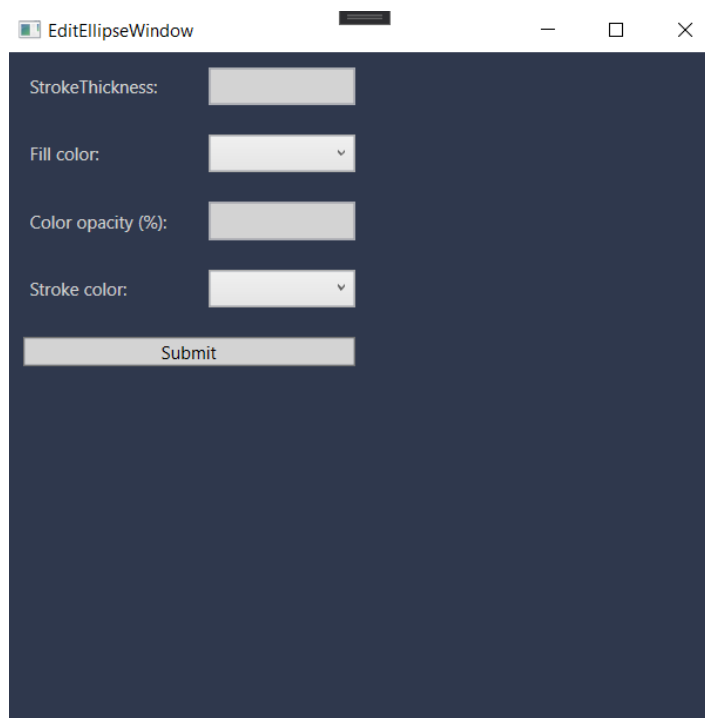
Софтвер имплементира могућност додавања елипси, полигона и текста преко исцртаног модела. Корисник преко радио-дугмади (*Draw Ellipse*, *Draw Polygon*, *Add Text*) бира који ће се облик исцртати.

У случају одабира исцртавања елипсе, корисник бира позицију на платну на којој ће се елипса исцртати, а затим, приликом притиска десног клика миша на одређену позицију, отвара се нови прозор (Слика 20) у којем се уносе вредности потребне за исцртавање елипсе. Те вредности су: ширина, висина, дебљина линије, боја, провидност боје, боја линије, текст унутар елипсе (опционо) и боја текста (опционо), респективно. Кликом на дугме *Submit*, исцртаће се елипса на одређеној позицији на платну.

The screenshot shows a window titled "EllipseWindow" with a dark blue background. On the left side, there are eight labels with corresponding input fields: "X:" with a text box, "Y:" with a text box, "StrokeThickness:" with a slider, "Fill color:" with a color picker, "Color opacity (%):" with a slider, "Stroke color:" with a color picker, "Text (optional):" with a text box, and "Text color (optional):" with a color picker. At the bottom of these controls is a "Submit" button.

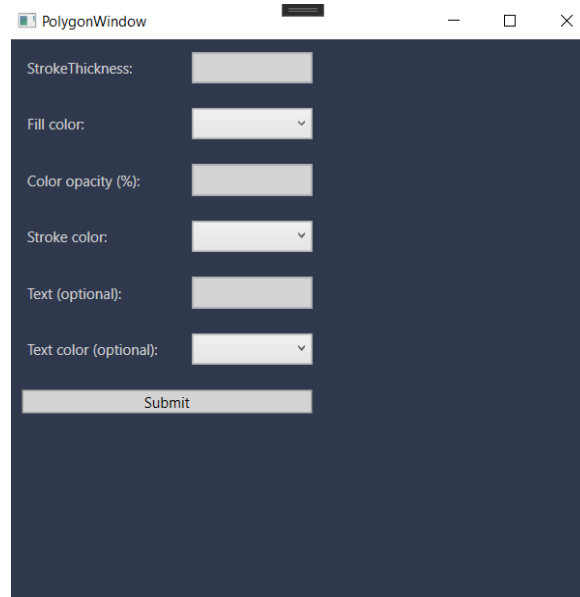
Слика 20 – Прозор за креирање елипсе

Такође, омогућена је измена атрибута приказаних елипси. Левим кликом на елипсу, отвара се нови прозор (Слика 21) у којем се уносе нове вредности за дебљину линије, боју, провидност боје и боју линије. Кликот на дугме *Submit*, примениће се унесене промене елипсе.

The screenshot shows a window titled "EditEllipseWindow" with a dark blue background. It contains four labels with input fields: "StrokeThickness:" with a slider, "Fill color:" with a color picker, "Color opacity (%):" with a slider, and "Stroke color:" with a color picker. At the bottom of these controls is a "Submit" button.

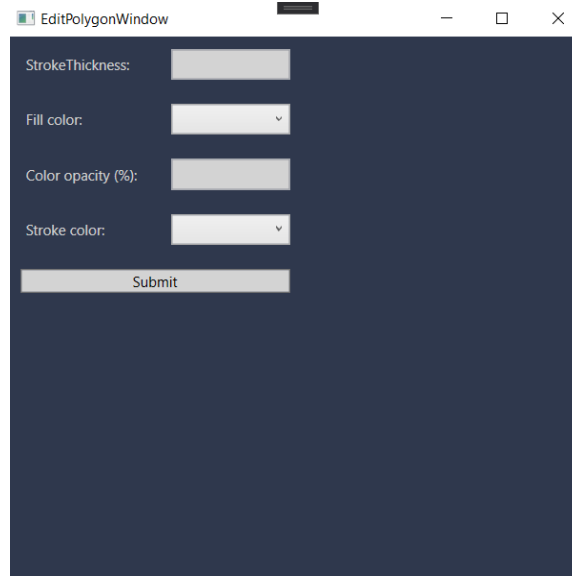
Слика 21 – Прозор за измену елипсе

У случају одабира исцртавања полигона, корисник бира позицију на платну на којој ће се полигон исцртати, а затим, приликом притиска десног клика миша на одређене позиције, одређују се тачке које ће ограничити површину полигона. Након што су сва темена полигона одређена, левим кликом на било коју позицију на платну отвара се нови прозор (Слика 22) у којем се уносе вредности потребне за исцртавање полигона. Те вредности су: дебљина линије, боја, провидност боје, боја линије, текст унутар полигона (опционо) и боја текста (опционо), респективно. Кликом на дугме *Submit*, исцртаће се полигон на одређеним позицијама на платну.



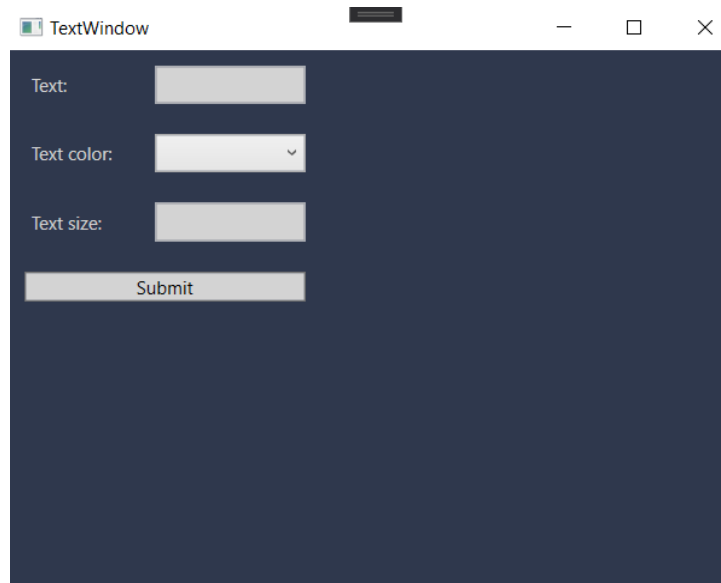
Слика 22 – Прозор за креирање полигона

Такође, омогућена је измена атрибута приказаних полигона. Левим кликом на полигон, отвара се нови прозор (Слика 23) у којем се уносе нове вредности за дебљину линије, боју, провидност боје и боју линије. Кликом на дугме *Submit*, примениће се унесене промене полигона.



Слика 23 – Прозор за измену полигона

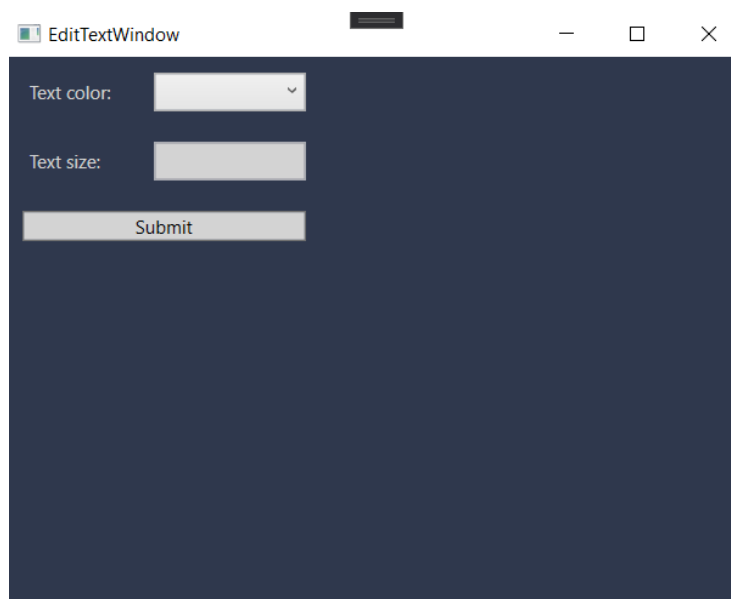
У случају одабира уноса текста, корисник бира позицију на платну на којој ће се текст приказати, а затим, приликом притиска десног клика миша на одређену позицију, отвара се нови прозор (Слика 24) у којем се уносе вредности потребне за приказивање текста. Те вредности су редом сам текст, његова боја и величина. Кликом на дугме *Submit*, исписаће се текст на одређеној позицији на платну.



Слика 24 – Прозор за креирање текста

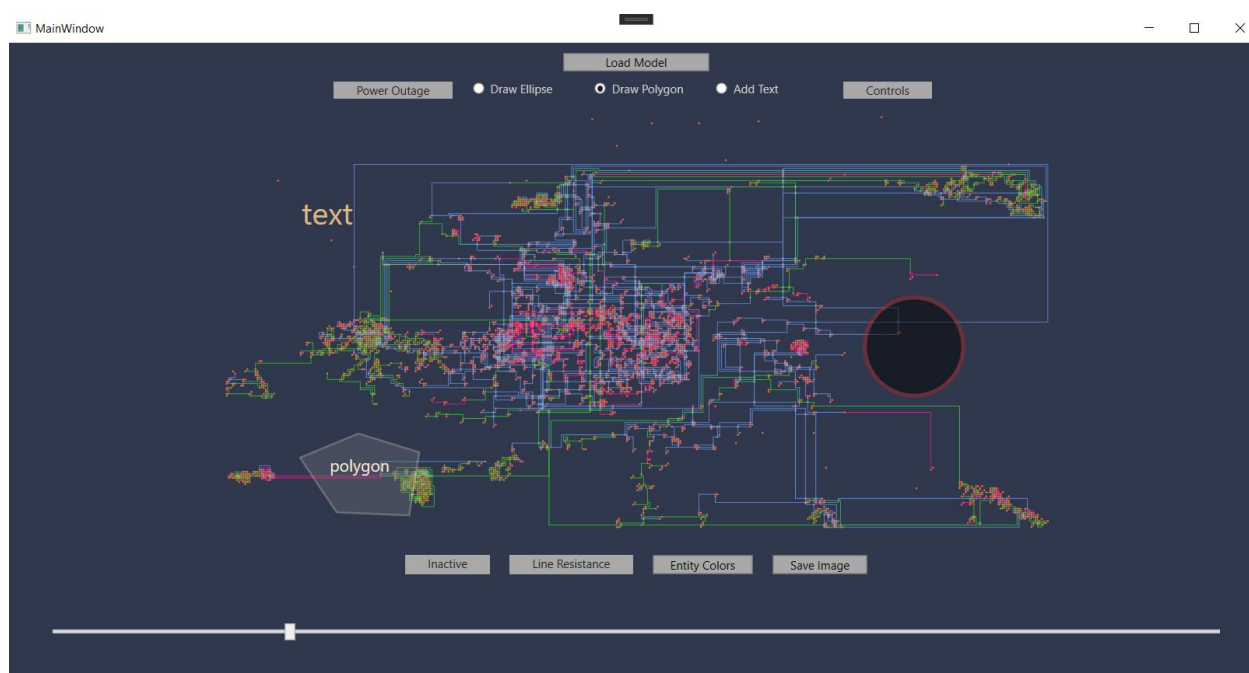
Такође, омогућена је измена атрибута приказаних текстова. Левим кликом на текст, отвара се нови прозор (Слика 25) у којем се уносе нове вредности за боју и величину текста. Кликом на дугме *Submit*, примениће се унесене промене текста.





Слика 25 – Прозор за измену текста

Пример исцртаних облика и исписаног текста преко модела мреже приказан је на Слици 26.



Слика 26 - Пример исцртаних облика и исписаног текста преко мреже

Кликом на опцију *Controls*, кориснику су понуђене три опције – *Undo*, *Redo* и *Clear*, као што је приказано на Слици 27.



Слика 27 – Controls опције

Кликом на опцију *Undo*, уклониће се последње исцртани елемент са платна.

Кликом на опцију *Redo*, кориснику је омогућено да се врати корак уназад тј. да врати последње уклоњени елемент са платна.

Кликом на опцију *Clear*, уклањају се сви исцртани елементи са платна и остаје само модел електродистрибутивне мреже. Међутим, поновним кликом на опцију *Redo*, корисник може да поврати све елементе који су нестали приликом обраде функционалности *Clear*.

## 6. Закључак

Као резултат рада, добијена је функционална апликација која омогућава увид у модел мреже дистрибуције електричне енергије Новог Сада. Циљ апликације јесте надзор и управљање поменутом мрежом. Кориснику је омогућено учитавање модела, цртање облика (елипсе и полигона) и додавање текста преко модела, измена наведених облика и текстова, сакривање и поновно приказивање неактивног дела мреже, промена боја водова у односу на вредност отпорности и враћање на иницијалну боју, опциони одабир боје/слике за одређени тип ентитета, чување слике тренутног приказа и додатна функционалност приказа безнапонских делова мреже.

Као евентуални додатак раду, зарад даљег усавршавања, могла би се имплементирати реална мапа испод приказа модела мреже помоћу коришћења библиотеке *GMap.NET*. Такође, као још једна од додатних функционалности, могла би се додати функција означавања одређеног дела мреже и приказивања само уоквиреног дела модела, зарад прецизнијег увида у детаље модела, док се ентитети ван тих граница не би приказивали. Корисно би било имплементирати *BFS* алгоритам на другачији начин, такав да извршавање буде брже.

## Литература

- [1] J. Hughes, A. Van Dam, M. McGuire, D. Sklar, J. Foley, *Computer Graphics: Principles and Practice*, 3rd ed., 2013
- [2] A. Weil, *Learn WPF MVVM - XAML, C# and the MVVM pattern*, 2016
- [3] A. Nathan, *WPF 4.5 Unleashed*, 1st ed., 2010
- [4] R. Landa, R. Gonella, S. Brower, *2D: Visual Basics For Designers (Design Concepts)*, 1st ed., 2006
- [5] J. Duncan Glover, T. Overbye, S. Mulukutla, *Power System Analysis and Design*, 6th ed., 2016
- [6] GeeksforGeeks, „GeeksforGeeks“, <https://www.geeksforgeeks.org/breadth-first-search-or-bfs-for-a-graph/>, Breadth First Search algoritam.

## Додатак А

```

public Queue<Element> BFS(Element start, Element end)
{
    Queue<Element> copy = new Queue<Element>();
    Queue<Element> queue = new Queue<Element>();

    divs[start.X, start.Y].Visited = true;
    queue.Enqueue(start);
    copy.Enqueue(start);

    bool found = false;

    Element current = new Element();

    while (queue.Any())
    {
        current = queue.Dequeue();

        if (current.X == end.X && current.Y == end.Y)
        {
            found = true;
            break;
        }
        if (current.X + 1 < divCountX)
        {
            if (!divs[current.X + 1, current.Y].Visited)
            {
                divs[current.X + 1, current.Y].Visited = true;
                divs[current.X + 1, current.Y].ParentX = current.X;
                divs[current.X + 1, current.Y].ParentY = current.Y;
                queue.Enqueue(divs[current.X + 1, current.Y]);
                copy.Enqueue(divs[current.X + 1, current.Y]);
            }
        }
        if (current.X - 1 >= 0)
        {
            if (!divs[current.X - 1, current.Y].Visited)
            {
                divs[current.X - 1, current.Y].Visited = true;
                divs[current.X - 1, current.Y].ParentX = current.X;
                divs[current.X - 1, current.Y].ParentY = current.Y;
                queue.Enqueue(divs[current.X - 1, current.Y]);
                copy.Enqueue(divs[current.X - 1, current.Y]);
            }
        }
        if (current.Y + 1 < divCountY)
        {
            if (!divs[current.X, current.Y + 1].Visited)
            {
                divs[current.X, current.Y + 1].Visited = true;
                divs[current.X, current.Y + 1].ParentX = current.X;
                divs[current.X, current.Y + 1].ParentY = current.Y;
                queue.Enqueue(divs[current.X, current.Y + 1]);
                copy.Enqueue(divs[current.X, current.Y + 1]);
            }
        }
        if (current.Y - 1 >= 0)
        {
            if (!divs[current.X, current.Y - 1].Visited)
            {
                divs[current.X, current.Y - 1].Visited = true;
            }
        }
    }
}

```

```

        divs[current.X, current.Y - 1].ParentX = current.X;
        divs[current.X, current.Y - 1].ParentY = current.Y;
        queue.Enqueue(divs[current.X, current.Y - 1]);
        copy.Enqueue(divs[current.X, current.Y - 1]);
    }
}

if (!found)
{
    while (copy.Any())
    {
        Element temp = copy.Dequeue();

        divs[temp.X, temp.Y].Visited = false;
        divs[temp.X, temp.Y].ParentX = -1;
        divs[temp.X, temp.Y].ParentY = -1;
    }

    return null;
}

Stack<Element> stack = new Stack<Element>();
while (copy.Count > 0)
{
    stack.Push(copy.Dequeue());
}
while (stack.Count > 0)
{
    copy.Enqueue(stack.Pop());
}

Element last = new Element();
while (copy.Any())
{
    Element temp = copy.Dequeue();
    if (temp.X == end.X && temp.Y == end.Y)
    {
        last = temp;
        break;
    }

    divs[temp.X, temp.Y].Visited = false;
    divs[temp.X, temp.Y].ParentX = -1;
    divs[temp.X, temp.Y].ParentY = -1;
}

Queue<Element> ret = new Queue<Element>();
ret.Enqueue(last);

while (copy.Any())
{
    Element next = copy.Dequeue();

    if (last.ParentX == next.X && last.ParentY == next.Y)
    {
        ret.Enqueue(next);
        last = next;
    }
    else
    {
        divs[next.X, next.Y].Visited = false;
        divs[next.X, next.Y].ParentX = -1;
        divs[next.X, next.Y].ParentY = -1;
    }
}
}

```

```
divs[start.X, start.Y].Visited = false;  
divs[start.X, start.Y].ParentX = -1;  
divs[start.X, start.Y].ParentY = -1;  
divs[end.X, end.Y].Visited = false;  
divs[end.X, end.Y].ParentX = -1;  
divs[end.X, end.Y].ParentY = -1;  
  
return ret;  
}
```

## Подаци о кандидату



Кандидат Наташа Шћекић је рођена 6. 7. 1999. године у Новом Саду. Завршила је Основну школу „Јован Поповић“ у Новом Саду са скроз одличним успехом као носилац Вукове дипломе. Похађала је Гимназију „Јован Јовановић Змај“ у Новом Саду. Факултет техничких наука у Новом Саду је уписала школске 2018/2019. године. У школској 2021/2022. години добила је Доситејеву диплому за успех на претходним годинама студија. Испунила је све обавезе и положила је све испите предвиђене студијским програмом Примењено софтверско инжењерство у року са просечном оценом 9.30.