VAASAN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES

Markus Rintamäki

DIAGNOSTIIKKATYÖKALU JA SEN KÄYTTÖLIITTYMÄN VISUALISOINTI VERKKOSELAIMESSA

Tekniikka ja liikenne
2016

VAASAN AMMATTIKORKEAKOULU

Tietotekniikan koulutusohjelma

TIIVISTELMÄ

Tekijä	Markus Rintamäki
Opinnäytetyön nimi	Diagnostiikkatyökalu ja sen visualisointi verkkoselaimessa
Vuosi	2016
Kieli	suomi
Sivumäärä	42
Ohjaaja	Martti Mustonen

Tämän opinnäytetyön tarkoituksena oli suunnitella ja toteuttaa ABB Oy Power Grids Grid Automation -yksikölle selainpohjainen diagnostiikkatyökalu IEC61850 OPC-palvelimelle. Tällä hetkellä käytössä oleva työkalu sisältää paljon vähän käytettyjä ominaisuuksia ja lisäksi tärkeimmät niistä ovat hankalasti saatavilla. Näistä syistä johtuen haluttiin uusi työkalu, joka on mahdollisimman selkeä ja helppokäyttöinen.

Sovellus toteutettiin käyttämällä JavaScript-, HTML- ja CSS-ohjelmointikieliä. Lopputuloksena oli yksi iso XML-tiedosto, joka kuvaa miten näkymä on tarkoitus rakentaa. Sovellus pystyy ottamaan yhteyden OPC-palvelimiin ja hakemaan suojareleiden yhteystila-, diagnostiikka- ja prosessitietoja.

Opinnäytetyön tuloksena saatiin toimiva sovellus, joka täyttää sille asetetut tärkeimmät vaatimusmäärittelyt. Diagnostiikkatyökalua tullaan jatkossa edelleen kehittämään, siihen toteutetaan esimerkiksi tässä työssä toteuttamatta jääneet ominaisuudet.

Avainsanat IEC61850, OPC DA, MicroSCADA Pro, JavaScript, HTML

VAASAN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES
Tietotekniikan koulutusohjelma

ABSTRACT

Author	Markus Rintamäki
Title	A Diagnostic Tool and Visualizing its User Interface in a Web-browser
Year	2016
Language	Finnish
Pages	42
Name of Supervisor	Martti Mustonen

The aim of this thesis was to design and implement a browser-based Diagnostic Tool for an IEC61850 OPC server. The thesis was done for ABB Power Grids Grid Automation unit. The tool currently being used contains many rarely used features and the main ones are difficult to access. For these reasons a new tool is wanted that is as clear as possible and easy to use.

The application was implemented using JavaScript, HTML and CSS programming languages. The end result was one large XML file that describes how the view is to be built. The application is able to connect to OPC servers and retrieve protection relay status as well as diagnostic and process information.

The result of this thesis was a fully functional application that fulfills the most important requirement specifications set for it. The development of the Diagnostic Tool will be continued in the future. For example, features that were left outside of the scope of this thesis will be implemented.

Keywords IEC61850, OPC DA, MicroSCADA Pro, JavaScript, HTML

SISÄLLYS

KUVIO- JA TAULUKKOLUETTELO

TERMIT JA LYHENTEET

HVDC	High-voltage direct current, suurjännitetasavirta
SCADA	Supervisory Control And Data Acquisition, valvomo-ohjelmisto
IEC	International Electrotechnical Commission, kansainvälinen sähköalan standardointiorganisaatio
OPC	Open Platform Communications, sarja standardeja ja spesifikaatioita teollisuuden tietoliikenteelle
XML	Extensible Markup Language, merkintäkieli
CET	Communication Engineering Tool, ABB:n kehittämä työkalu
CEF	Chromium Embedded Framework, sovelluskehys upotetun selaimen käyttöön muissa sovelluksissa
Ping	TCP/IP-protokollan työkalu laitteen saavutettavuuden kokeilemiseen
AMQP	Advanced Message Queuing Protocol, protokolla hajautettujen järjestelmien väliseen liikenteeseen
IED	Intelligent Electronic Device, suojarele
IET	Integrated Engineering Tool, ABB:n kehittämä työkalu

1 JOHDANTO

Nykypäivänä selainpohjaisten sovellusten määrä kasvaa koko ajan. Se on suosittu sovellustyyppi, koska sovellusta pystytään käyttämään monilla eri alustoilla, esimerkiksi mobiililaitteilla. Mobiilikäyttö mahdollistaa sovelluksen käytön lähes missä tahansa.

Opinnäytetyön toimeksiantajana toimii ABB Oy Power Grids Grid Automation. Työn toteutus tehdään yksikön toimipisteessä Vaasassa heidän tarjoamilla laitteilla. Työn tavoitteena on luoda selainpohjainen diagnostiikkatyökalu IEC61850 OPC-palvelimelle nykyisen työpöytäsovelluksen CETin pohjalta. Opinnäytetyö on osa isompaa teknologiapäivitystä.

Tällä hetkellä käytössä oleva CET-työkalu sisältää paljon vähän käytettyjä ominaisuuksia ja tärkeimmät niistä ovat hankalasti saatavilla. Uudesta työkalusta halutaan mahdollisimman selkeä ja käyttäjäystävällinen.

Opinnäytetyön alussa esitellään työn kannalta oleellista teoriaa. Tämän jälkeen työssä kuvataan toteutettavaa sovellusta sekä käydään läpi käyttöliittymän suunnitteluprosessi. Seuraavaksi työssä käsitellään toteutuksen tärkeimmät vaiheet. Työn lopussa pohditaan jatkosuunnitelmia sovelluksen kehittämiseksi sekä opinnäytetyön onnistuvuutta.

2 ABB OY

ABB Oy on sähkö- ja automaatioteknologian alalla toimiva maailmanlaajuinen yhtymä, jonka pääkonttori sijaitsee Zürichissä. ABB työllistää Suomessa yli 5000 henkilöä ja lähes 140 000 henkilöä ympäri maailman 100 maassa. Suomessa yritys vaikuttaa noin 20 paikkakunnalla ja sen suurimmat tehdaskeskittymät ovat Vaasassa ja Helsingissä. Yrityksen liiketoiminta muodostuu neljästä eri divisioonasta, jotka ovat Electrification Products, Discrete Automation and Motion, Process Automation ja Power Grids. /2, 3, 5/

ABB Oy perustettiin vuonna 1988, kun ruotsalaisen Asean ja sveitsiläisen Brown Boverin sähköteknologiset liiketoiminnat yhdistettiin. ABB panostaa vahvasti tutkimukseen ja tuotekehitykseen, mikä on varmistanut yrityksen menestymisen eri markkinatilanteissa. Yritys on kehittänyt monia korkeatasoisia tekniikoita, kuten esimerkiksi pitkien välimatkojen HVDC-siirtolinjat ja ratkaisut laivojen sähköistykseen. Tällä hetkellä teollisuuden moottorit ja taajuusmuuttajat sekä sähköverkkojen toimitus ovat ABB:n liiketoiminnan kannalta tärkeimpiä tuoteryhmiä. /2, 4/

ABB Power Grids Grid Automation -yksikkö keskittyy verkonhallintajärjestelmiin, jotka valvovat ja ohjaavat siirto- ja jakeluverkkoja. Siirto- ja jakeluverkot tarvitsevat verkonhallintajärjestelmän, jotta ne toimisivat luotettavasti ja mahdollisimman tehokkaasti. Suomessa yksikössä työskentelee noin 40 henkilöä, jotka vastaavat MicroSCADA Pro -käytönvalvontajärjestelmän tutkimuksesta, kehityksestä, markkinoinnista ja myynnistä. /6/

3 KÄYTETYT TEKNOLOGIAT

3.1 MicroSCADA Pro

MicroSCADA Pro on sähköasema-automaatio- ja käytönhallintajärjestelmäsovelluksiin suunniteltu tuoteperhe, johon kuuluu kolme tuotetta: SYS600, SYS600C ja DMS600. /7/

Tämän opinnäytetyön kannalta oleellinen asia on SYS600-tuote, joka on SCADA-järjestelmä. Järjestelmä kerää reaaliaikaisia mittaustuloksia ja tilannetietoja verkkoon kytketyistä laitteista. Tieto kerätään prosessitietokantaan ja se voidaan esittää järjestelmän käyttöliittymässä, joka on konfiguroitavissa. Eniten käytettyjä ominaisuuksia käyttöliittymässä ovat tapahtumatietojen, hälytysten, mittaustulosten muutosten ja valvontakohteen yksiviivaesitysten näyttäminen sekä sähköprosessin laitteiden ohjaaminen. MicroSCADA Pro SYS600 -järjestelmän käyttäjiä ovat pääasiassa sähkönjakeluyhtiöt. /8/

3.2 Web-teknologiat

3.2.1 JavaScript

JavaScript, lyhennetään usein JS, on oliopohjainen komentosarjakieli, jonka avulla määritetään miten verkkosivut käyttäytyvät. JavaScript on parhaiten tunnettu ohjelmointikieli, jota käytetään verkkosivuilla. Verkkosivujen lisäksi JavaScriptiä voidaan käyttää myös ympäristöissä, jotka eivät ole selainpohjaisia. /9–10/

JavaScriptin kehitti Brendan Eich vuonna 1995, kun hän työskenteli Netscapella. JavaScriptistä tuli ECMA-standardi vuonna 1997, jonka jälkeen JavaScript on tunnettu myös nimellä ECMAScript. JavaScript pohjautuu ECMA-262-standardiin. JavaScriptiä ei tule sekoittaa Javaan, sillä ne eroavat toisistaan hyvinkin paljon. Java on luokkapohjainen, kun taas JavaScript pohjautuu prototyyppeihin. /9–10/

JavaScriptin tärkein ominaisuus on dynaamisten toimintojen lisääminen verkkosivuille. Sillä voidaan luoda käyttäjän ja tietokoneen vuorovaikutusta parantavia toimintoja. Tällainen on esimerkiksi toiminto, jossa painike reagoi muuttamalla väriä

käyttäjän siirtäessä hiiren painikkeen päälle. Verkkoselain suorittaa JavaScript-tulkkauksen suoraan, joten sitä ei tarvitse kääntää erilliseksi tiedostoksi. /9/

JavaScript-ohjelmaa voidaan kirjoittaa HTML-dokumenttiin <body> ja <head>-osioihin. JavaScript ohjelma voidaan joko kirjoittaa suoraan HTML-dokumenttiin tunnisteiden <script> ja </script> väliin tai linkittää erillisenä tiedostona dokumenttiin, jolloin se kirjoitetaan seuraavalla tavalla: <script src=tiedoston_nimi.js”></script>. /10/

3.2.2 HTML

HTML on standardisoitu verkkodokumenttien eli verkkosivujen rakenteen ja sisällön kuvaamiseen luotu merkintäkieli. Merkintäkielen on tarkoitus järjestää raakateksti sellaiseen muotoon, että verkkoselain ymmärtää millaisessa muodossa teksti halutaan esittää. Raakateksti järjestellään lisäämällä tekstin joukkoon merkintätunnisteita. Tunniste koostuu avainsanasta ja sen ympärille lisättävistä kulmasulkeista. Tyypillisesti tunnisteet lisätään tekstin joukkoon pareina. Parin ensimmäistä tunnistetta kutsutaan aloitustunnisteeksi ja jälkimmäistä lopetustunnisteeksi. Lopetustunniste sisältää saman avainsanan kuin aloitustunniste, mutta sen avainsanan eteen on lisätty kauttaviiva. /11/

Aloitus- ja lopetustunnisteen ja niiden väliin jäävän sisällön yhdistelmää kutsutaan HTML-elementiksi. Näiden elementtien avulla muodostetaan HTML-dokumentti. Elementin tunnisteiden väliin jäävä osa sisältää tekstiä ja yleensä myös muita HTML-elementtejä. /11/

Esimerkki HTML-dokumentista:

```
<!-- Dokumentin tyyppi auttaa selainta näyttämään sivun oikein -->
<!DOCTYPE html>
<!-- Dokumentin sisältöä kuvaava elementti -->
<!-- Ensimmäisen elementin aloitustunniste -->
<html>
<!-- Selaimessa näkyvä sisältö määritetään body avainsanan avulla
-->
<!-- Sisältö-elementin aloitustunniste -->
<body>

<h1> Sisällön otsikko </h1>
<p> Sisällön kappale </p>

<!-- Sisältö-elementin lopetustunniste -->
</body>
<!-- Ensimmäisen elementin lopetustunniste -->
</html>
```

Esimerkistä ilmenee HTML-dokumentin rakenne. Dokumentin tyypin määrittämisen jälkeen aloitustunniste `<html>`, lopetustunniste `</html>` ja niiden väliin jäävät rivit muodostavat yhden HTML-elementin. Tämä elementti puolestaan sisältää selaimen sisältöä kuvaavan elementin, joka koostuu tunnisteista `<body>`, `</body>` ja niiden välisistä riveistä. Sisältö-elementti sisältää kaksi elementtiä `<h1> Sisällön otsikko </h1>` ja `<p> Sisällön kappale </p>`. Tehty esimerkki näyttäytyy verkkoselaimessa alla olevalla tavalla **(Kuvio 1.)**.

Sisällön otsikko

Sisällön kappale

Kuvio 1. Verkkoselaimen esittämä sisältö HTML-dokumentista.

3.2.3 HTML5

HTML5 on uusin versio standardisoidusta HTML-merkkikielestä. HTML5-version avulla verkkosivuille on helpompi lisätä multimediatiedostoja kuin aiemmissa HTML-versioissa. Uudessa versiossa esimerkiksi videoiden lisääminen verkkosi-

vulle voidaan tehdä `<video>` tunnisteen avulla. Tunniste on standardisoitu tapa lisätä video verkkosivulle ja sen avulla videon toistaminen voidaan tehdä ilman lisäosien lataamista. HTML5 määrittelee myös standardisoidun tavan lisätä äänitiedostoja `<audio>` tunnisteen avulla. Myös äänitiedostojen toistaminen onnistuu ilman lisäosia. /12–14/

Mediasisällön helpottamisen lisäksi HTML5 tarjoaa paremmat edellytykset käyttäjän ja verkkosivun vuorovaikutusta parantavien sovellusten luomiseen. Sovelluskehittäjät voivat tehdä verkkosivujen rakenteista selkeämpiä ja monipuolisempia, koska HTML5-versiossa esimerkiksi kappaleiden otsikoiden numerointi, ala- ja ylätunnisteiden lisääminen tai erillisen rakenteen luominen muusta tekstistä erillään olevalle objektille, kuten foorumiviestille tai uutisartikkelille onnistuu uusien tunnisteiden avulla. Näiden tunnisteiden lisäksi HTML5 sisältää monia muita tunnisteita, esimerkiksi kuvien asetteluihin ja ajan esittämiseen. /12/

3.2.4 CSS

CSS eli Cascading Style Sheets on verkkosivun ulkoasun muotoiluun käytettävä ohjelmointikieli. CSS-kielellä kuvataan miltä merkintäkielellä kirjoitettu dokumentti näyttää käyttäjälle verkkoselaimessa. Kun merkintäkielenä käytetään esimerkiksi HTML-kieltä, CSS kuvaa kuinka HTML-elementit tulisi kuvantaa näytölle. CSS-kielestä on olemassa standardisoitu versio W3C-spesifikaatio. CSS-standardista viimeisin versio on CSS3, joka on jaettu moduuleihin. Esimerkiksi valitsimista, laatikoiden muuntamisesta ja animoinnista on omat moduulinsa. /15, 17/

CSS-ohjelma voidaan sijoittaa HTML-dokumenttiin kolmella tapaa, joko suoraan elementtiin tai <head>-osioon, tai lisäämällä linkki luodusta CSS-tiedostosta <head>-osioon. Erillisellä CSS-tiedostolla on monia etuja, kuten esimerkiksi se, että se helpottaa päällekkäisyyksien välttämistä sekä tekee ulkoasun ylläpidosta helpompaa. CSS-tiedosto myös vähentää työmäärää huomattavasti, sillä se voi vastata usean verkkosivun ulkoasusta yhtä aikaa. /15–16/

CSS-syntaksi muodostetaan valitsimesta ja määrittelylohkosta. Valitsin osoittaa HTML-elementtiin, jota halutaan muotoilla. Määrittelylohko sisältää yhden tai useamman määrittelyn, jotka erotetaan toisistaan puolipisteellä. Jokainen määrittely sisältää CSS-ominaisuuden ja arvon kaksoispisteellä erotettuna. Määrittelylohko lopetetaan aina puolipisteeseen ja sitä ympäröi aaltosulkeet. /18/

Esimerkki CSS-syntaksista:

```
h1 { color:red; font-size:12px;}
```

Esimerkin tapauksessa muotoiltava elementti on h1, ja tälle elementille määrittelyjä on kaksi: väri on punainen ja fontin koko on 12.

3.2.5 WebUI for MicroSCADA Pro

MicroSCADA Pro WebUI on kehitteillä oleva uuden sukupolven käyttöliittymäalusta. WebUI-alustan tarkoitus on yhdistää useampi järjestelmä toimimaan samalla käyttöliittymällä, parantaa käyttäjäkokemusta sekä tehdä järjestelmien mobiilikäytöstä mahdollista. /19/

WebUI:n arkkitehtuuri on asiakas-palvelin pohjainen. Asiakasohjelmisto voi olla mikä tahansa selain, joka tukee HTML5:sta. Tämä mahdollistaa järjestelmien käytön monilla eri alustoilla, kuten esimerkiksi mobiililaitteilla. Mobiilikäytössä järjestelmien esitystapa saattaa kuitenkin poiketa perinteisestä näyttötyypistä. /19/

WebUI:n dialogimalli on seuraavanlainen: Kun asiakasohjelmisto pyytää palvelimelta verkkosivua, lataa palvelin *User Interface Definition Document*:n dialogivarastosta. *User Interface Definition Document* on XML-tiedosto, joka kuvaa miten näkymä on tarkoitus rakentaa. Palvelin suorittaa *User Interface Definition Document*:n ja lähettää sen asiakasohjelmistolle. Jos *User Interface Definition Document* tarvitsee arvoja ulkoisesta järjestelmästä, tilataan ne ja lähetetään edelleen asiakasohjelmistolle. Kun tilattu arvo ulkoisessa järjestelmässä muuttuu, lähettää palvelin päivitetyn arvon asiakasohjelmistolle ja näkymä päivittyy automaattisesti. /19/

Opinnäytetyön perinteisempien tehtävien lisäksi yhtenä tehtävänä oli toimia WebUI:n koekäyttäjänä ja raportoida mahdollisista uusista ideoista sekä vastaan tulleista ongelmista.

3.3 IEC61850

IEC61850 on sähköverkkojen tietoliikennestandardi, jonka tärkeimpänä tavoitteena on ollut varmistaa, että eri valmistajien suojarelelaitteet ovat keskenään yhteensopivia. Standardi esittelee puurakenteisen tietomallin suojareleelle. /20/ Tietomallitasot koostuvat alenevassa hierarkiajärjestyksessä korkeimmasta lähtien seuraavasti:

1. Fyysisistä laitteista/palvelimista (Physical Device, PD)
2. Loogisista laitteista (Logical Device, LD)
3. Loogisista solmuista (Logical Node, LN)
4. Data-objekteista (Data Object, DO)
5. Data-attribuuteista (Data Attribute, DA)

Kuviossa 2 on diagnostiikkatyökalun puunäkymän sisältö numeroitu tietomallitasojen mukaan **(Kuvio 2.)**.

Kuvio 2. Tietomallitasot puunäkymässä

3.4 OPC DA

OPC Data Access spesifikaatio, joka tunnetaan myös nimellä OPC DA, on ensimmäinen klassisista OPC-spesifikaatioista. OPC DA sisältää asiakas-palvelin standardeja, jotka tarjoavat spesifikaatioita reaaliaikaiseen tiedonsiirtoon tiedonkeruu- ja valvontalaitteen välillä. Opinnäytetyössä spesifikaatiota käytettiin suojareleen ja valvomo-ohjelmisto SCADAn väliseen tiedonsiirtoon. /1/

OPC DA määrittelee kolme ominaisuutta, jotka tiedonkeruulaitteen pitää pystyä palauttamaan asiakkaan niitä pyytäessä. Nämä kolme OPC DA -tiedon ominaisuutta ovat arvo, laatu ja aikaleima. Jos tiedonkeruulaite ei pysty tarjoamaan näitä kolmea ominaisuutta, OPC DA-palvelin palauttaa pyydetyt tiedot. /1/

4 SOVELLUKSEN KUVAUS

4.1 Lähtökohta

Opinnäytetyön tavoitteena on siis luoda selainpohjainen diagnostiikkatyökalu IEC61850 OPC -palvelimelle nykyisen työpöytäsovelluksen CETin pohjalta. Työkalua tullaan pääasiallisesti käyttämään upotettuna selaimena (CEF), mutta suora selainkäyttökin on mahdollista.

Diagnostiikkatyökalun täytyy pystyä näyttämään kommunikaatiokanavien ja kytkettyjen suojareleiden yhteystila- ja diagnostiikkatiedot. Sillä täytyy pystyä tarkkailemaan kytkettyjen suojareleiden prosessitietoja, kuten mittauksia ja kytkimien asentoja. Lisäksi sen täytyy automaattisesti mukautua OPC-palvelimen konfiguraatioon selaamalla OPC-nimiavaruutta.

4.2 Vaatimusten määrittely

Tässä osiossa käydään tarkemmin läpi diagnostiikkatyökalun toimintavaatimuksia. Taulukkoon on listattu diagnostiikkatyökaluun haluttuja toimintoja ja siitä pystytään näkemään, mitkä toiminnot tulevat olemaan tärkeitä toteutuksessa **(Taulukko 1.)**. Prioriteetit ovat 1 (täytyy olla), 2 (pitäisi olla) ja 3 (kiva olla).

Taulukko 1. Vaatimusmäärittely

Toiminto	**Priori-teetti**
Aliverkon suojareleiden yleistietonäkymä	1
Yksittäisen suojareleen tarkempi yhteystilatieto- ja diagnostiikka-laskurinäkymä	1
Puunäkymä, johon haetaan automaattisesti IEC61850-tietomallin mukainen tietorakenne käymällä läpi OPC-nimiavaruutta	1
Data-attribuuttien arvojen lukunäkymä	1
Työkalun tapahtumaloki	1
Yhteystilatietoikoni puunäkymässä	2
OPC-tapahtuma- ja hälytysnäkymä	3
Data-attribuuttien arvojen kirjoittaminen	3
Suojareleen saavutettavuuden kokeileminen ping-toiminnolla suoraan työkalusta	3

Opinnäytetyön aihealuetta tarkasteltiin myöhemmin toteutuksen aikana ja päädyttiin siihen, että prioriteetti kolmosen toiminnot toteutetaan myöhemmin eivätkä näin ollen kuulu opinnäytetyön sisältöön.

4.3 Käyttötapauskaavio

Käyttötapauskaavion avulla pystytään selkeästi esittämään mitä eri mahdollisuuksia käyttäjällä on.

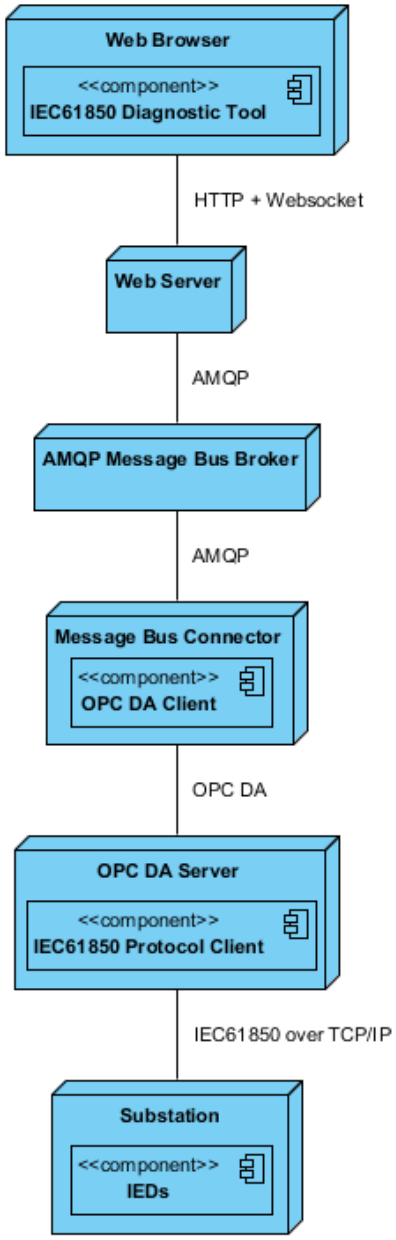
Käyttäjällä on mahdollisuus tutkia työkalun tapahtumalokia ja tarkistaa suojareleiden yhteystilatieto suoraan puunäkymästä. Lisäksi käyttäjä voi tarkastella yhden aliverkon suojareleiden yleistietoja, data-attribuuttien arvoja ja suojareleen tarkempia yhteystilatietoja ja diagnostiikkalaskureita. Käyttäjällä on myös mahdollisuus nollata diagnostiikkalaskurit **(Kuvio 3.)**.

Kuvio 3. Diagnostiikkatyökalun käyttötapauskaavio

4.4 Sijoittelukaavio

Sijoittelukaavion avulla saadaan selkeä yleiskuva koko järjestelmästä, sen eri komponenteista ja niiden välisestä liikenteestä.

Ala-aseman suojareleiden IEC61850-kommunikaatio tuodaan MicroSCADA-palveluväylään (AMQP) OPC DA -palvelimen ja -asiakasohjelmiston kautta **(Kuvio 4.)**.



Kuvio 4. Järjestelmän sijoittelukaavio

4.5 Sekvenssikaaviot

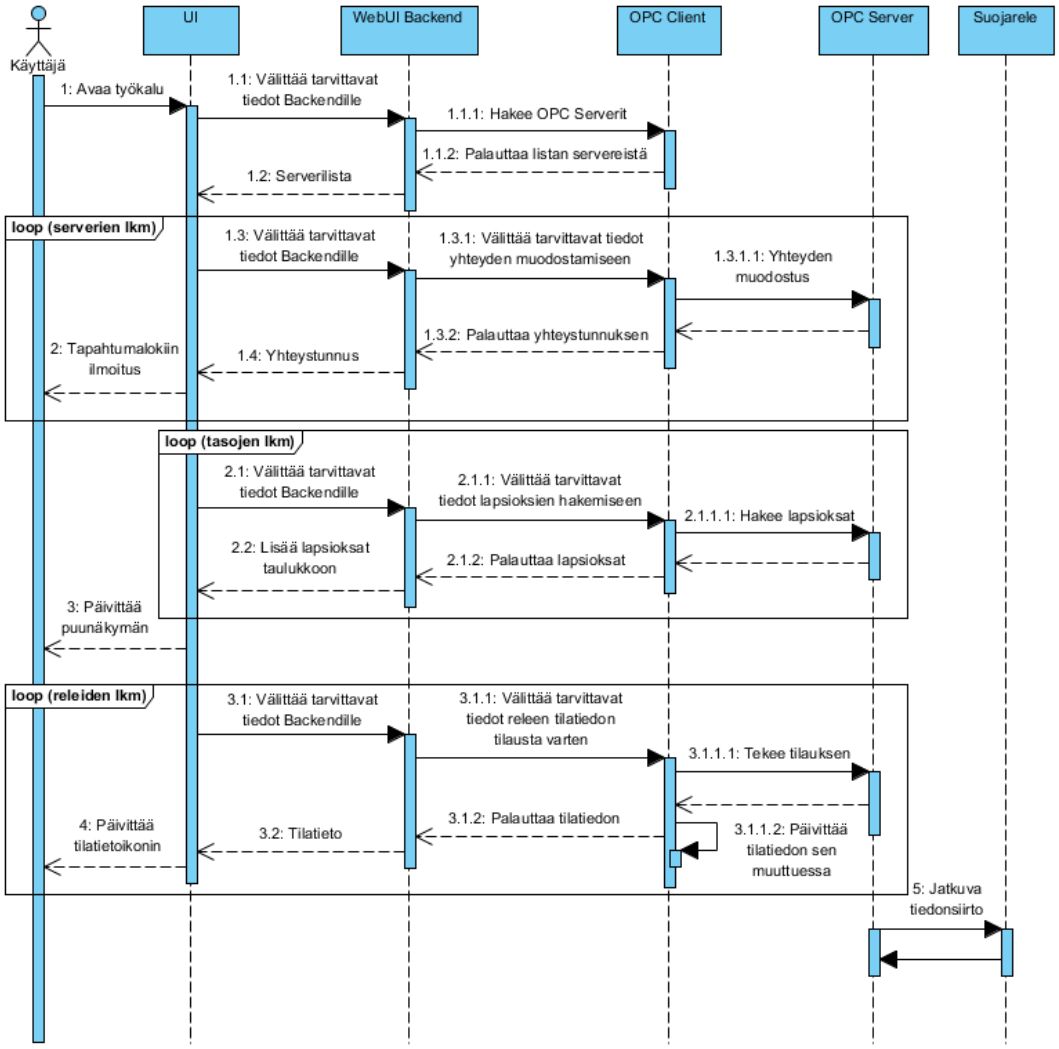
Sekvenssikaavion avulla voidaan kuvata sovelluksen eri toimintavaiheita. Lisäksi se antaa selkeän kuvan käyttäjän ja järjestelmän välisestä vuorovaikutuksesta.

4.5.1 Työkalun avaaminen

Kun käyttäjä avaa työkalun, haetaan saatavilla olevat OPC-palvelimet. OPC-asiakasohjelmisto palauttaa listan palvelimista ja käyttöliittymä tallettaa listan taulukkoon. Listan OPC-palvelimet käydään yksitellen läpi. OPC-asiakasohjelmistolle välitetään tarvittavat tiedot yhteyden muodostamiseen, joka tämän jälkeen muodostaa yhteyden OPC-palvelimeen. OPC-asiakasohjelmisto palauttaa yhteystunnuksen, jolla erotetaan palvelimet toisistaan. Käyttöliittymä antaa käyttäjälle tapahtumalokiin ilmoituksen yhteyden muodostamisesta.

Jokaiselle OPC-palvelimelle haetaan lisäksi IEC61850-tietomallin mukainen tietorakenne. Käyttöliittymä välittää tarvittavat tiedot lapsioksien hakemiseen ja OPC-asiakasohjelmisto palauttaa ne käyttöliittymälle, joka tallentaa lapsioksat taulukkoon. Tämä toistetaan niin monta kertaa, että IEC61850-tietomalli on käyty läpi tasolta aliverkko, tasolle data-objekti. Data-objekti-tasolla haetaan vielä sen alla olevat lehdet eli data-attribuutit. Kun kaikki tasot on käyty läpi, päivitetään puunäkymä käyttäjälle.

Jokaiselle suojareleelle haetaan yhteystilatieto. OPC-asiakasohjelmistolle välitetään tarvittavat tiedot tilauksen tekemistä varten. OPC-asiakasohjelmisto tekee tilauksen ja palauttaa yhteystilatiedon käyttöliittymälle. Käyttöliittymä päivittää käyttäjälle suojareleen yhteystilatietoikonin. Tilaus jää voimaan ja aina kun yhteystilatieto muuttuu, lähettää OPC-asiakasohjelmisto uuden yhteystilatiedon käyttöliittymälle. **(Kuvio 5.)**.

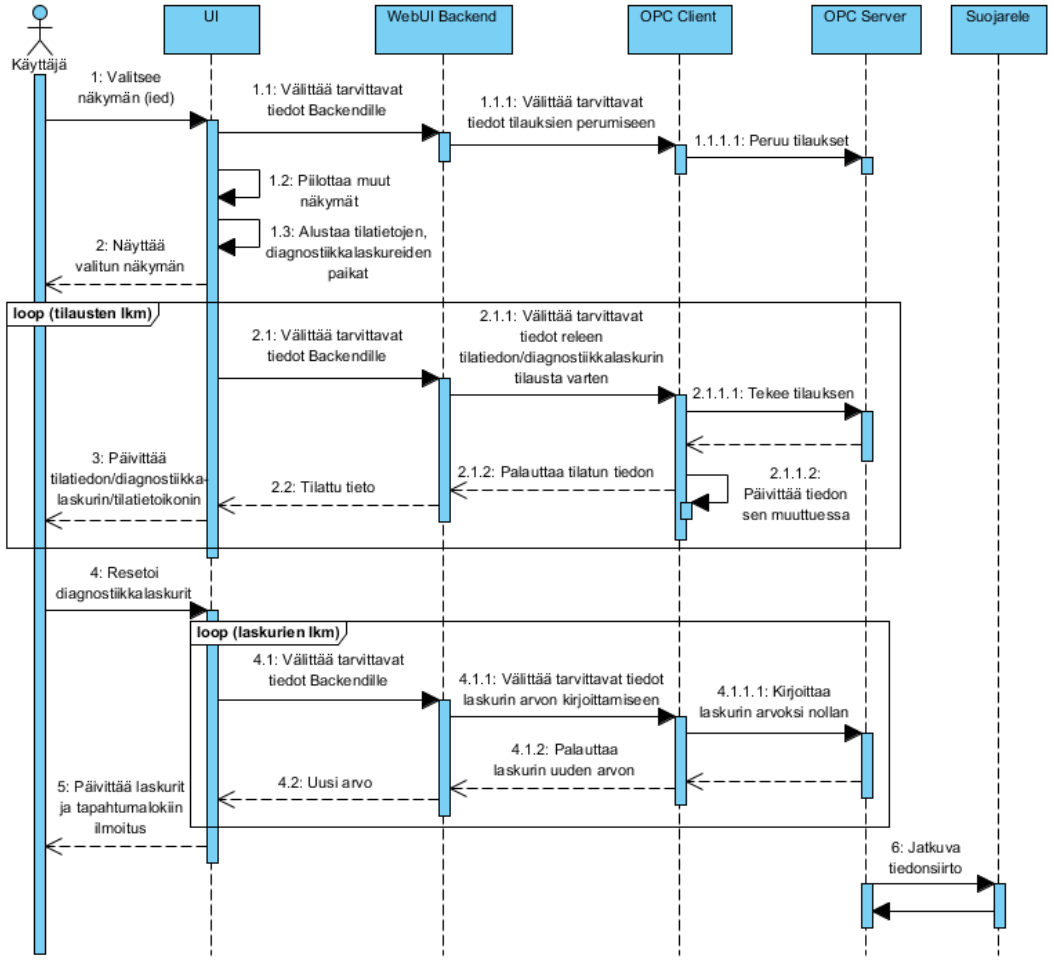


Kuvio 5. Sekvenssikaavio työkalun avaamisesta

4.5.2 Näkymän vaihtaminen

Käyttäjän vaihtaessa näkymää, perutaan edellisen näkymän tilaukset virheellisten tietojen välttämiseksi. Samaan aikaan käyttöliittymä piilottaa muut näkymät ja alustaa yhteystilatietojen ja diagnostiikkalaskureiden paikat. Tämän jälkeen välitetään OPC-asiakasohjelmistolle tarvittavat tiedot tilauksen tekemistä varten, joka tekee tilauksen. OPC-asiakasohjelmisto palauttaa tilatun tiedon ja käyttöliittymä päivittää sen käyttäjälle. Tämä toistetaan niin monta kertaa kuin haluttuja yhteystilatieto- tai diagnostiikkalaskuritilauksia on. Niin kauan kuin näkymä on valittuna, päivittää OPC-asiakasohjelmisto minkä tahansa tilatun tiedon muuttuessa uuden tiedon käyttöliittymälle.

Tässä esimerkissä käyttäjällä on lisäksi mahdollisuus nollata diagnostiikkalaskurit. Käyttäjän painaessa Reset counters -painiketta välitetään OPC-asiakasohjelmistolle tarvittavat tiedot laskurin arvon kirjoittamiseen. OPC-asiakasohjelmisto kirjoittaa nollan laskurin uudeksi arvoksi ja palauttaa käyttöliittymälle uuden arvon. Tämä toistetaan niin monta kertaa kuin on laskureita. Lopuksi käyttöliittymä päivittää nollatut arvot käyttäjälle. **(Kuvio 6.)**.



Kuvio 6. Sekvenssikaavio näkymän vaihtamisesta

5 KÄYTTÖLIITTYMÄN SUUNNITTELU

5.1 Suunnittelun lähtökohdat

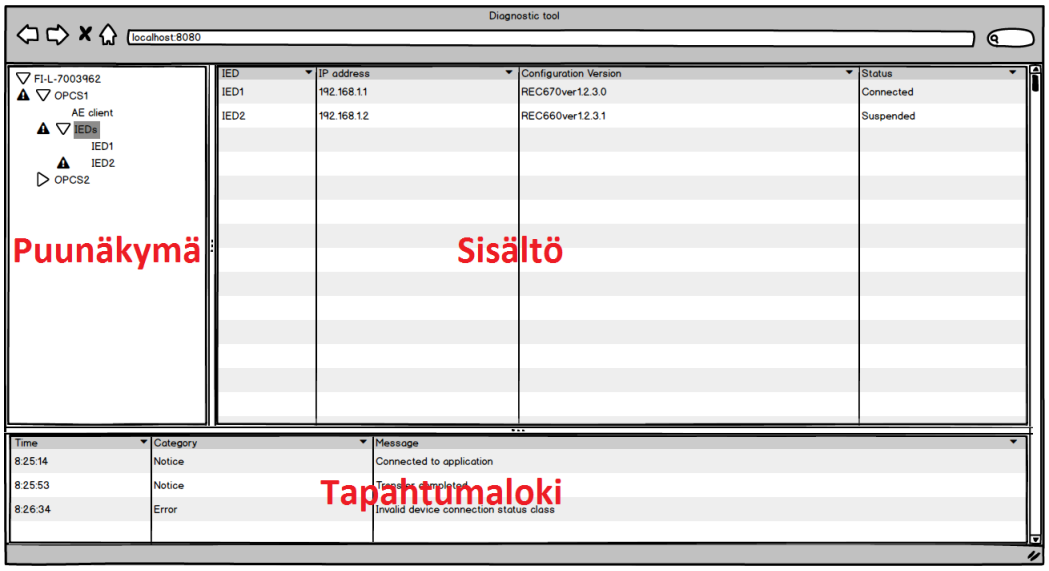
Ennen käyttöliittymän suunnittelua tutkittiin nykyistä CET-työkalua. Pidin palavereja CET-työkalun käyttäjien kanssa ja pyrin saamaan selville mikä työkalussa on hyvää ja mikä huonoa. Sain muutamia parannusehdotuksia, kuten esimerkiksi aliverkon suojareleiden yleistietonäkymä ja suojareleen saavutettavuuden kokeileminen ping-toiminnolla suoraan työkalusta. Otin vaikutteita toisesta jo olemassa olevasta uuden sukupolven selainpohjaisesta työkalusta nimeltä MicroSCADA IET Data Loader Tool. Diagnostiikkatyökalusta halutaan samankaltainen edellä mainitun työkalun kanssa, sillä ne ovat molemmat osa samaa teknologiapäivitystä.

Näiden asioiden pohjalta tein luonnostelmat, jotka suunnittelin Balsamiq Mockups -työkalulla. Balsamiq Mockups on nopea ja helppokäyttöinen työkalu luonnosten tekoon.

5.2 Suunnittelun toteutus

5.2.1 Käyttöliittymän yleisluonnostelma

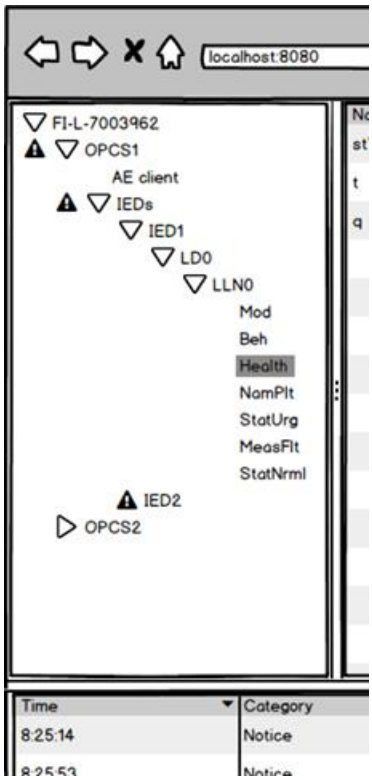
Käyttöliittymän yleisluonnostelma on jaettu kolmeen osaan **(Kuvio 7.)**. Vasemmalla on puunäkymä, josta käyttäjä voi valita haluamansa näkymän. Sisältö vaihtuu sen mukaan mitä käyttäjä valitsee puunäkymästä. Luonnostelman alareunassa on työkalun tapahtumaloki. Tapahtumalokista käyttäjä näkee nopeasti mitä työkalussa sillä hetkellä tapahtuu, kuten esimerkiksi huomautuksia ja virheilmoituksia.



Kuvio 7. Käyttöliittymän yleisluonnostelma

5.2.2 Puunäkymä

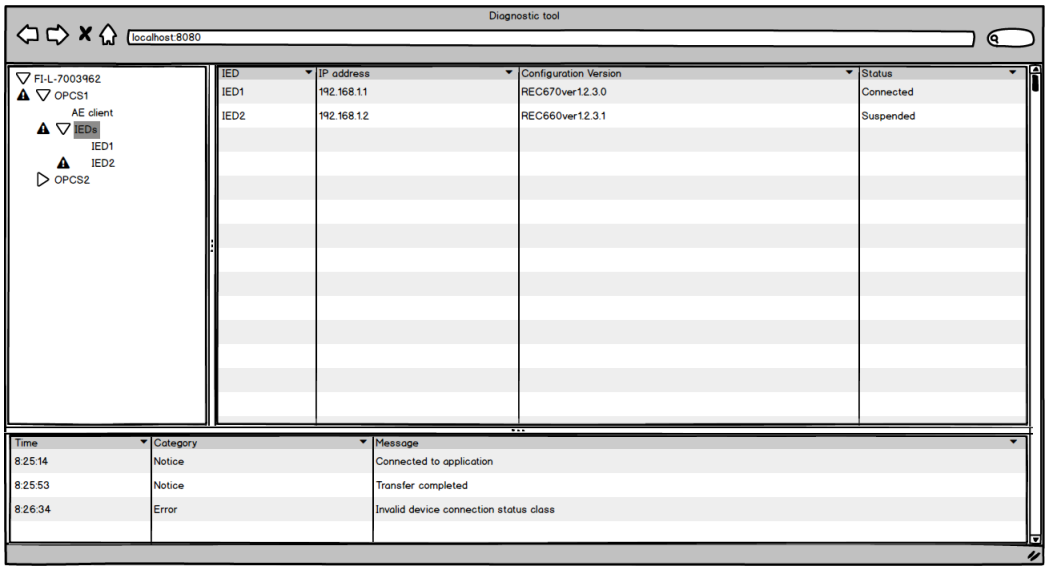
Käyttöliittymä hakee automaattisesti IEC61850-tietomallin mukaisen tietorakenteen ja näyttää sen puunäkymässä **(Kuvio 8.)**. Puunäkymän eri tasot ovat OPC-palvelin (OPCS1), aliverkko (IEDs), fyysinen laite (IED1), looginen laite (LD0), looginen solmu (LLN0) ja data-objekti (Health). Kun käyttäjä valitsee puunäkymästä aliverkon, vaihtuu näkymäksi aliverkon suojareleiden yleistietonäkymä. Vastaavasti kun käyttäjä valitsee puunäkymästä fyysisen laitteen, vaihtuu näkymäksi yksittäisen suojareleen tarkempi yhteystilatieto- ja diagnostiikkalaskurinäkymä. Käyttäjän valitessa puunäkymästä data-objektin, vaihtuu näkymäksi data-attribuuttien arvojen lukunäkymä. Puunäkymässä näkyy huomautusikoni, mikäli suojarele on vikatilassa tai siihen ei saada luotua yhteyttä. AE client jätettiin opinnäytetyön sisällön ulkopuolelle.



Kuvio 8. Puunäkymän luonnostelma

5.2.3 Aliverkon suojareleiden yleistietonäkymä

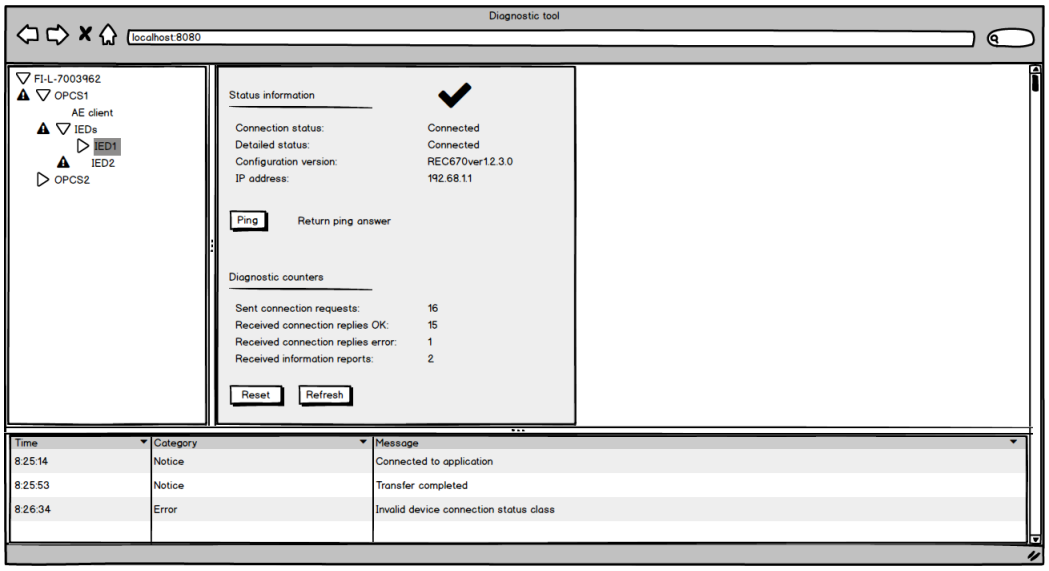
Yleistietonäkymässä käyttäjä näkee nopeasti ja kootusti aliverkon kaikkien suojareleiden IP-osoitteet, konfiguraatioversiot ja yhteyden tilan **(Kuvio 9.)**.



Kuvio 9. Aliverkon suojareleiden yleistietonäkymän luonnostelma

5.2.4 Yksittäisen suojareleen tarkempi yhteystilatieto- ja diagnostiikkalaskurinäkymä

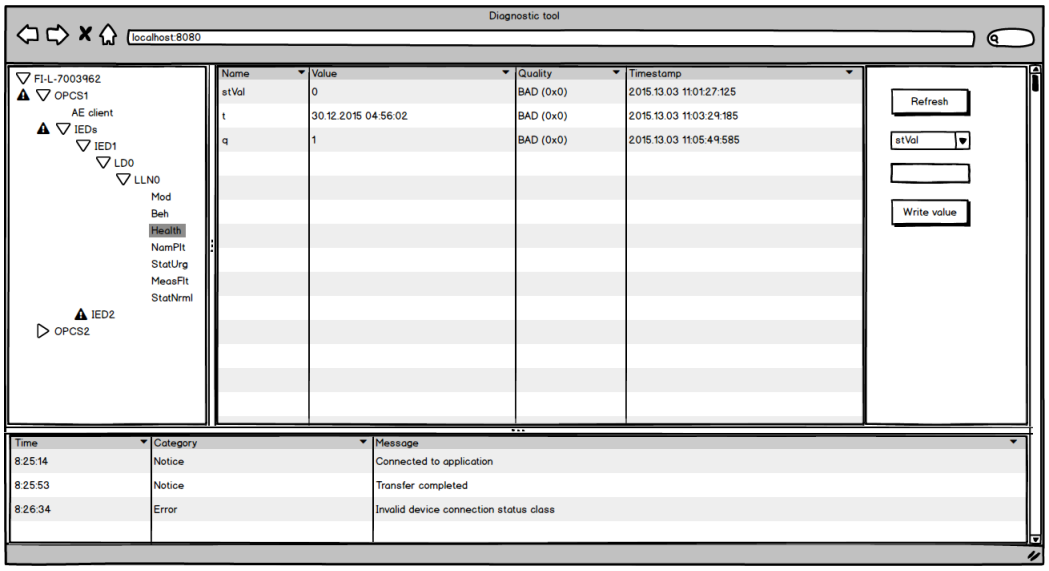
Tässä näkymässä käyttäjä näkee edellisen näkymän lisäksi suojareleen yksityiskohtaisemman yhteystilatiedon ja diagnostiikkalaskurit **(Kuvio 10.)**. Lisäksi yhteystilatietoikoni päivittyy yhteystilatiedon mukaan. Reset-painikkeesta käyttäjä voi nollata diagnostiikkalaskurit. Refresh-painiketta ei tarvittu toteutuksessa, sillä laskurit päivittyvät automaattisesti. Ping-toiminto jätettiin opinnäytetyön sisällön ulkopuolelle.



Kuvio 10. Yksittäisen suojareleen tarkempi yhteystilatieto- ja diagnostiikkalaskurinäkymän luonnostelma

5.2.5 Data-attribuuttien arvojen lukunäkymä

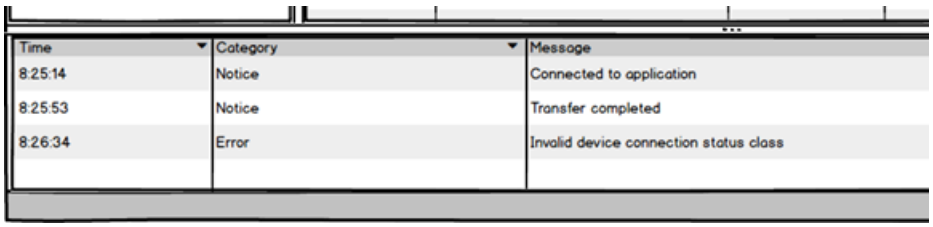
Data-attribuuttien arvojen lukunäkymässä käyttäjä pystyy data-attribuuttien arvojen lukemisen lisäksi lukemaan myös arvon laadun ja aikaleiman milloin arvo on viimeksi muuttunut **(Kuvio 11.)**. Data-attribuuttien arvojen kirjoitus jätettiin opinnäytetyön sisällön ulkopuolelle.



Kuvio 11. Data-attribuuttien arvojen lukunäkymän luonnostelma

5.2.6 Tapahtumaloki

Tapahtumalokista käyttäjä pystyy seuraamaan mitä työkalussa tapahtuu **(Kuvio 12.)**. Käyttäjälle tulevia ilmoituksia ovat yhteyden muodostus OPC-palvelimeen ja suojareleen yhteystilatiedon muuttuminen.



Kuvio 12. Tapahtumalokin luonnostelma

6 DIAGNOSTIIKKATYÖKALUN TOTEUTUS

6.1 Yleistä toteutuksesta

Apuna toteutuksessa oli yksinkertaisia koodiesimerkkejä muun muassa OPC-palvelimien hausta ja korkeamman tason käyttöliittymäkomponenttien käytöstä. Niitä jouduttiin kuitenkin muokkaamaan käyttötarkoituksiin sopiviksi. Lähdekoodieditorina toteutuksessa käytettiin Notepad++:aa.

Toteutuksen tuloksena oli yksi WebUI-näkymä, iso XML-tiedosto, joka sisältää JavaScript:ä, HTML:ää, CSS:ää ja erilaisia palvelurajapintoja. Toteutus on sen verran laaja, että seuraavissa kolmessa kappaleessa käydään läpi toteutukselle oleellisemmat toiminnot. Luvun viimeisessä kappaleessa esitellään toteutettu sovellus.

6.2 OPC-palvelimien haku

OPC-palvelimien haku toteutettiin getServers-funktiolla **(Kuvio 13.)**. Kun käyttäjä avaa työkalun, kutsuu käyttöliittymä getServers-funktiota. Funktio valmistelee GenericOpc.NamespaceBrowsing.GetServers-palvelukutsun ja suorittaa sen. Palvelu palauttaa listan palvelimista, jonka käyttöliittymä tallettaa taulukkoon. Tämän jälkeen kutsutaan connectOpcServers-funktiota, joka hoitaa yhteyden muodostamisen OPC-palvelimiin. Sekunnin viiveen jälkeen puunäkymä päivitetään olettaen, että OPC-palvelimiin saadaan yhteys sen aikana.

```
self.getServers = function() {
    var host = 'localhost'; //isäntämuuttujan sijoitus
    var request = { //palvelukutsun valmistelu
        session: {sessionId: self.get("sessionId")},
        host: host
    };
    self.invokeService( //palvelukutsun suoritus
        "GenericOpc.NamespaceBrowsing",
        "GetServers",
        request,
        function(result, final) {
            self.servers = result.servers.filter(function(s) {
                return s.indexOf("IEC61850") > -1;
            }); //lisätään palautetut palvelimet taulukkoon
            self.connectOpcServers(); //funktiokutsu
            setTimeout(function() {
                self.set("tree_data", self.updated_tree_data);
            }, 1000);   //näytetään sekunnin jälkeen
                        //tietorakenne puunäkymässä
        }
    );
}
```

Kuvio 13. getServers-funktio

6.3 Puunäkymän rakentaminen

Opinnäytetyön ehdottomasti haastavin vaihe oli puunäkymän rakentaminen. Se toteutettiin getNamespace-funktiolla **(Kuvio 14.)**. getNamespace-funktiota kutsutaan jokaisen OPC-palvelimen yhteyden muodostamisen jälkeen. Funktiolle tuodaan parametrina i, joka on OPC-palvelimen järjestysnumero, tree, joka sisältää sen hetkisen tietorakenteen taulukossa ja depth, joka ilmaisee tietorakenteen sen hetkistä tasoa. getNamespace on rekursiivinen funktio, eli se kutsuu funktion lopussa aina itse itseään uusilla parametriarvoilla. Tästä johtuen funktion alussa on toistoehto. Tässä tapauksessa se on määritelty niin, että jos toistoehto on yli viisi, hyppää ohjelma ulos funktiosta. Toistoehdon tarkastuksen jälkeen funktio valmistelee GenericOpc.NamespaceBrowsing.GetChildren-palvelukutsun tarvittavilla tiedoilla ja suorittaa sen. Palvelu palauttaa haetut lapsioksat. Lapsioksat käydään yksitellen läpi ja niiden tiedot sijoitetaan tietorakennetaulukkoon. Funktion lopussa funktio kutsuu

itseään uusilla parametriarvoilla. Kun IEC61850-tietomallin mukaiset tasot on käyty läpi tasolta aliverkko, tasolle data-objekti, päivitetään puunäkymä käyttäjälle.

```
self.getNamespace = function(i, tree, depth) { //parametrit
    if (depth > 5) { //funktio toistoehto
        return;
    }
    var path = ''; //polun muuttujan alustus
    if (depth > 1) {
        path = tree.path;
    }       //muuttujaan sijoitetaan parametrina tuotu polkutieto
    var connection = self.connection; //yhteysid:n talletus muuttujaan
    var request = { //palvelukutsun valmistelu
        session: {sessionId: self.get("sessionId")},
        connection: connection,
        path: path
    };
    var subTree = tree; //muuttujaan sijoitetaan parametrina tuotu taulukko
                        //joka sisältää sen hetkisen tietorakenteen
    self.invokeService( //palvelukutsun suoritus
        "GenericOpc.NamespaceBrowsing",
        "GetChildren",
        request,
        function(result, final) {
            for (var b = 0; b < result.childBranches.length; b++) {
                        //käydään läpi palautetut lapsioksat
                        //jotka eivät ole "Attributes"
                if (result.childBranches[b] != "Attributes") {
```

```
                if (!subTree.items) {
                    subTree.items = []; //alustetaan taulukko lapsioksille
                }                       //mikäli sitä ei ole vielä tehty
                var parentPath = ''; //alustetaan vanhemman polkumuuttuja
                if (depth > 1) {
                    parentPath = subTree.path + '\\';
                } //sijoitetaan muuttujaan polkutieto
                if (depth == 1) { //tehdään ainoastaan aliverkkotasolla
                    subTree.items[b] = {
                        label: result.childBranches[b], expanded: true,
                        path: parentPath + result.childBranches[b],
                        value: {opc_server: self.servers[i],
                        level: self.levels[depth],
                        parent: subTree.path,
                        path: parentPath + result.childBranches[b]},
                        icon: self.icons[depth]
                    };
                } else { //tehdään muilla tasoilla
                    subTree.items[b] = { //sijoitetaan lapsioksan tiedot taulukkoon
/*nimitieto*/               label: result.childBranches[b],
/*polkutieto*/              path: parentPath + result.childBranches[b],
/*opc-palvelintieto*/       value: {opc_server: self.servers[i],
/*tasotieto*/               level: self.levels[depth],
/*vanhempitieto*/           parent: subTree.path,
/*polkutieto*/              path: parentPath + result.childBranches[b]},
/*ikonitieto*/              icon: self.icons[depth]
                    };
                }
                self.getNamespace(i, subTree.items[b], depth+1);
                //rekursiivinen funktio, kutsu uusilla parametreilla
```

Kuvio 14. getNamespace-funktio

6.4 Tilauksen tekeminen

Tilauksen tekeminen toteutettiin startSubs-funktiolla **(Kuvio 15.)**. Käyttäjän vaihtaessa näkymää, kutsuu käyttöliittymä startSubs-funktiota. Funktiolle tuodaan parametrina id, joka on tilattavan tiedon polku ja iedName, joka on puunäkymästä valitun kohteen nimi. Lisäksi funktiolle tuodaan parametrina displayName, joka piti kierrättää funktion kautta, jotta yhteystilatieto saatiin oikealle suojareleelle ja callback, jolla varmistetaan tilauksen teko ennen ohjelman muiden osien suorittamista. Funktio valmistelee tilauksen määrittämällä tarvittavat tiedot ja tekee sitten tilauskutsun. Mikäli data-muuttuja ei ole tyhjä, kutsutaan callback-funktiota. Callback-funktio välittää tilatun tiedon eteenpäin haluttuun kohtaan ohjelmaa. Ohjelman suoritus jatkuu eteenpäin. startSubs-funktio tallettaa lopuksi tilaustunnuksen taulukkoon. Tilaus on voimassa niin kauan kunnes käyttäjä vaihtaa näkymää. Siinä tapauksessa taulukko käydään läpi ja kaikki tilaukset perutaan.

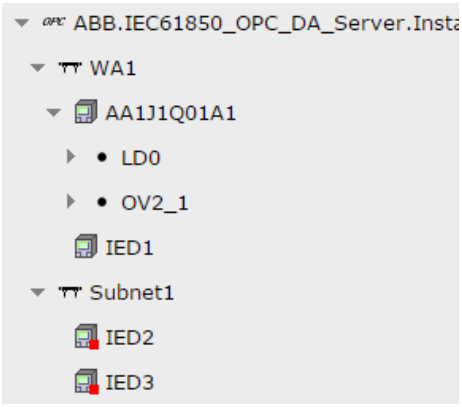
```
self.startSubs = function(id, iedName, displayName, callback) { //parametrit
    displayName = displayName || ''; //näyttönimen kierrättäminen
                                     //funktion kautta tarvittaessa
    var connection = self.connection;   //yhteysid:n talletus muuttujaan
    var path = "`GenericOpc::" + connection + ":" + id + "`";
                                        //tilauksen polun talletus muuttujaan
    var subscriptionId = 0;             //tilausid:n alustus
    subscriptionId = self.bindSingle(path, function(data) {
                                        //tilausfunktion kutsu
        if (iedName && self.currentlySelectedItem !== iedName) {
            return;     //näkymää vaihtaessa estetään virheellisen arvon palautus
        }
        if (typeof data !== "undefined" && data !== null) {
            callback(data.toString(), displayName);
        }              //callback-funktion kutsu mikäli data ei ole tyhjä
    });

    if (subscriptionId === 0) { //tilausid:n lisääminen taulukkoon
        self.pageSubscriptions.push(subscriptionId);
        self.currentSubscriptionId = subscriptionId;
    }
}
```

Kuvio 15. startSubs-funktio

6.5 Valmiin sovelluksen esittely

6.5.1 Puunäkymä

Valmis puunäkymä hakee IEC61850-tietomallin mukaisen tietorakenteen automaattisesti (**Kuvio 16.)**. Käyttäjä pystyy selaamaan tietorakennetta ja valitsemaan eri näkymiä. Tietorakenteen eri tasoille on annettu omat ikonit. Suojareleen ikoniin on lisäksi sisällytetty suojareleen yhteystilatieto. Punainen laatikko ilmoittaa suojareleen vikatilasta tai yhteyden katkeamisesta.



Kuvio 16. Toteutettu puunäkymä

6.5.2 Aliverkon suojareleiden yleistietonäkymä

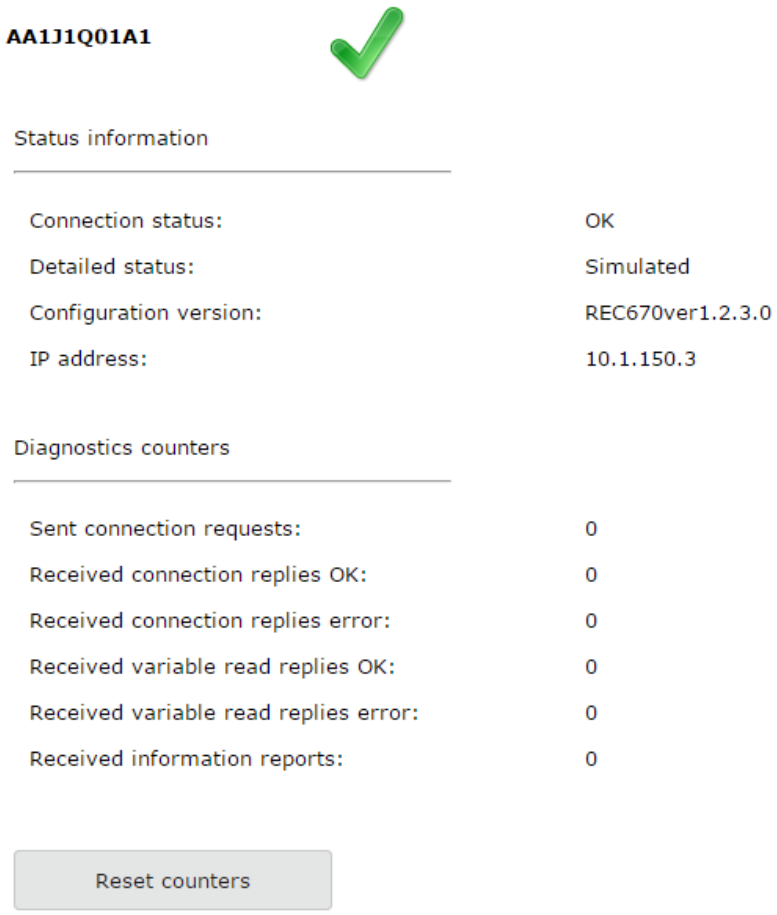
Valmis yleistietonäkymä näyttää käyttäjälle nopeasti ja kootusti kaikkien aliverkon suojareleiden tärkeimmät konfiguraatio- ja yhteystilatiedot **(Kuvio 17.)**. Palautteen perusteella näkymään lisättiin suunnittelun jälkeen suojareleen tarkempi yhteystilatieto.

IED name	IP address	Configuration version	Connection status	Detailed status
Filter	Filter	Filter	Filter	Filter
AA1J1Q01A1	10.1.150.3	REC670ver1.2.3.0	OK	Simulated
IED1	127.0.0.1	-	OK	Simulated

Kuvio 17. Toteutettu aliverkon suojareleiden yleistietonäkymä

6.5.3 Yksittäisen suojareleen tarkempi yhteystilatieto- ja diagnostiikkalaskurinäkymä

Valmis yksittäisen suojareleen näkymä näyttää käyttäjälle konfiguraatio- ja yhteystilatietojen lisäksi diagnostiikkalaskurit **(Kuvio 18.)**. Palautteen perusteella näkymään lisättiin suunnittelun jälkeen diagnostiikkalaskurit *Received variable read replies OK* ja *Received variable read replies error*. Yhteystilatietoikoni päivittyy yhteystilatiedon mukaan, ja käyttäjä voi halutessaan nollata diagnostiikkalaskurit.



Kuvio 18. Toteutettu yksittäisen suojareleen tarkempi yhteystilatieto- ja diagnostiikkalaskurinäkymä

6.5.4 Data-attribuuttien arvojen lukunäkymä

Valmis lukunäkymä näyttää käyttäjälle data-attribuuttien arvojen lisäksi arvon laadun ja aikaleiman siitä milloin arvo on viimeksi muuttunut **(Kuvio 19.)**. Testauksen aikana todettiin, että arvojen hakeminen näkymään on todella hidasta. Ohjelmointivirhettä ei pystytty toteutuksen aikana paikantamaan, mutta se tullaan korjaamaan tulevaisuudessa.

Name	Value	Quality	Timestamp
Filter	Filter	Filter	Filter
ctlVal	0	GOOD	6.6.2016 20:32:51
lastApplError	0	BAD	1.1.1980 14:00:00
cmdTermination	false	BAD	1.1.1980 14:00:00
stVal	0	BAD	6.6.2016 20:32:51
t	Mon Jun 06 2016 17..	GOOD	6.6.2016 20:32:51
q	0	GOOD	6.6.2016 20:32:51
ctlModel	1	BAD	1.1.1980 14:00:00
Oper\ctlNum	0	BAD	1.1.1980 14:00:00
Oper\Test	false	BAD	1.1.1980 14:00:00
Oper\Check	0	BAD	1.1.1980 14:00:00
Oper\T	Sat Dec 30 1899 00...	BAD	1.1.1980 14:00:00
Oper\ctlVal	0	BAD	1.1.1980 14:00:00

Kuvio 19. Toteutettu data-attribuuttien arvojen lukunäkymä

6.5.5 Tapahtumaloki

Valmiista tapahtumalokista käyttäjä pystyy seuraamaan työkalun eri tapahtumia **(Kuvio 20.)**. Tällä hetkellä ilmoituksia tulee OPC-palvelimien yhteyden muodostamisesta ja suojareleiden yhteystilatiedon muutoksesta.

Time	Level	Category	My logger
	Notice	Connection	Successfully connected to ABB.IEC61850_OPC_DA_Server.Instance[1]
6.50.05	Notice	IED status	AA1J1Q01A1: OK
6.50.05	Notice	IED status	IED1: OK
6.50.05	Notice	IED status	IED2: Suspended
6.50.05	**Notice**	**IED status**	**IED3: Suspended**

Kuvio 20. Toteutettu tapahtumaloki

7 TESTAUS

Diagnostiikkatyökalun testaaminen tapahtui pääasiassa toteutuksen aikana. Kun jokin uusi toiminto saatiin valmiiksi, testattiin se välittömästi. Testaamisessa käytettiin apuna työkalua nimeltä Softing Toolkit. Työkalulla pystyttiin simuloimaan suojareleiden käyttäytymistä halutulla tavalla. Toteutuksen valmistuttua, toteutettiin kolme testitapausta työkalun tärkeimpien toimintojen testaamiseksi **(Taulukko 2.)**.

Taulukko 2. Testitapaukset

Testitapaus	**Odotettu tulos**	**Tulos**
Työkalun avaaminen	OPC-palvelimiin saadaan yhteys, ja puunäkymä muodostuu oikein	OK
Näkymän vaihtaminen	Näkymä vaihtuu ilman haamukuvia ja tilatut tiedot näkyvät oikein	OK
Diagnostiikkalaskureiden nollaus	Diagnostiikkalaskurit nollaantuvat ja tapahtumalokiin tulee ilmoitus	OK

8 JOHTOPÄÄTÖKSET

Opinnäytetyön tuloksena saatiin toimiva sovellus, joka on hyvä prototyyppi jatkokehitykselle. Toteutettu sovellus täyttää sille asetetut tärkeimmät vaatimusmäärittelyt, jotka olivat aliverkon suojareleiden yleistietonäkymä, yksittäisen suojareleen tarkempi yhteystilatieto- ja diagnostiikkalaskurinäkymä, IEC61850-tietomallin mukaisen tietorakenteen sisältävä puunäkymä, data-attribuuttien arvojen lukunäkymä sekä työkalun tapahtumaloki.

Diagnostiikkatyökalua tullaan edelleen kehittämään, siihen toteutetaan tässä työssä toteuttamatta jääneet ominaisuudet ja käyttöliittymää tullaan viemään haluttuun suuntaan. Tuotteistamisen aikana työkalua tullaan testaamaan perusteellisemmin ja tiedossa oleva ohjelmointivirhe korjataan. Diagnostiikkatyökalu on mahdollisesti mukana seuraavassa julkaisussa.

Opinnäytetyö oli haastava ja odotettua laajempi. Työn alkuvaiheessa meni liian paljon aikaa uusien asioiden opetteluun ja omaksumiseen. Työlle alun perin suunniteltu aikataulu ei pitänyt, siitä huolimatta työn lopputuloksena saatuun sovellukseen voi olla tyytyväinen. Haasteellisuudestaan huolimatta, työ oli erittäin mielenkiintoinen ja opetti paljon, josta tulee varmasti olemaan hyötyä tulevaisuudessa.

LÄHTEET

Kirjallisuuslähteet

/1/ Iwanitz, F. & Lange, J. 2001. OLE for Process Control: Fundamentals, Implementation, and Application. Saksa. Hüthig Verlag Heidelberg.

Verkkolähteet

/2/ ABB Oy. ABB-yhtymä. Viitattu 5.6.2016.
http://new.abb.com/fi/abb-lyhyesti/yhtyma

/3/ ABB Oy. ABB Suomessa. Viitattu 5.6.2016.
http://new.abb.com/fi/abb-lyhyesti/suomessa

/4/ ABB Oy. ABB:n historia. Viitattu 5.6.2016.
http://new.abb.com/fi/abb-lyhyesti/historia

/5/ ABB Oy. ABB Group structure. Viitattu 6.6.2016.
http://new.abb.com/about/abb-in-brief/group-structure

/6/ ABB Oy. Network Management. Viitattu 6.6.2016
http://new.abb.com/fi/abb-lyhyesti/suomessa/yksikot/network-management

/7/ ABB Oy. MicroSCADA Pro. Viitattu 6.6.2016
http://new.abb.com/substation-automation/products/software/microscada-pro

/8/ ABB Oy. MicroSCADA Pro SYS600. Viitattu 6.6.2016
http://new.abb.com/substation-automation/products/software/microscada-pro/sys600

/9/ Mozilla. JavaScript. Viitattu 6.6.2016
https://developer.mozilla.org/en-US/docs/Web/JavaScript

/10/ w3schools. JavaScript. Viitattu 6.6.2016
http://www.w3schools.com/js

/11/ w3schools. HTML. Viitattu 6.6.2016
http://www.w3schools.com/html/

/12/ Mozilla. HTML5. Viitattu 6.6.2016
https://developer.mozilla.org/en-US/docs/Web/Guide/HTML/HTML5

/13/ w3schools. HTML5 Video. Viitattu 6.6.2016
http://www.w3schools.com/html/html5_video.asp

/14 w3schools. HTML5 Audio. Viitattu 6.6.2016
http://www.w3schools.com/html/html5_audio.asp

/15/ Mozilla. CSS. Viitattu 6.6.2016.
https://developer.mozilla.org/en-US/docs/Web/CSS

/16/ w3schools. CSS. Viitattu 6.6.2016.
http://www.w3schools.com/css

/17/ w3schools. CSS3. Viitattu 6.6.2016
http://www.w3schools.com/css/css3_intro.asp

/18/ w3schools. CSS Syntax. Viitattu 6.6.2016
http://www.w3schools.com/css/css_syntax.asp

/19/ ABB Oy. ABB MicroSCADA Pro WebUI Architecture Specification. 2013. ABB:n sisäinen dokumentti.

/20/ Illinois Security Lab. Prototype of IEC 61850. Viitattu 6.6.2016
http://seclab.illinois.edu/wp-content/uploads/2011/03/iec61850-intro.pdf