VAASAN AMMATTIKORKEAKOULU
VASA YRKESHÖGSKOLA
UNIVERSITY OF APPLIED SCIENCES

Harri Juhani Hurtola

OMD EXPLORER OHJELMISTO ABB OMD800-LAITTEELLE

Tekniikka ja liikenne
2010

ALKUSANAT

Tämä opinnäytetyö on tehty Vaasan ammattikorkeakoulun tietotekniikan koulutusohjelman päättötyönä. Itse työ aloitettiin kevättalvella 2009 ja työn kirjallinen osuus kirjoitettiin syksyllä 2009.

Haluan kiittää opinnäytetyön ohjaajaa yliopettaja Ghodrat Moghadampouria yhteistyöstä, avusta ja tuesta opinnäytetyön eri vaiheissa. Kiitokset osoitan myös ABB Oy Pienjännitekojeiden Raimo Sillanpäälle sekä Antti Kullakselle opinnäytetyön aiheesta sekä yhteistyöstä.

Erityisesti haluan kiittää avovaimoani Sannaa tuesta ja kärsivällisyydestä.

Vaasassa 11.12.2009

Harri Hurtola

VAASAN AMMATTIKORKEAKOULU

Tietotekniikan koulutusohjelma

TIIVISTELMÄ

Tekijä	Harri Hurtola
Opinnäytetyön nimi	OMD Explorer ohjelmisto ABB OMD800–laitteelle
Vuosi	2009
Kieli	suomi
Sivumäärä	52 + 2 liitettä
Ohjaaja	Ghodrat Moghadampour

Tässä opinnäytetyössä käsitellään OMD Explorer-nimisen ohjelmiston toteutusta käyttäen ohjelmointikielenä Microsoftin C# .NET:iä. Ohjelmistolla on tarkoitus esittää ABB Oy Pienjännitekojeet-yhtiön valmistaman OMD800-laitteen toimintatiloja ja toimintamahdollisuuksia käyttäjälleen tietoliikenneyhteydellä, joka toteutetaan Modbus-protokollalla.

OMD-laitteiden avulla saadaan luotua tavallisesta sähkökytkimestä automaattisesti toimiva sähkökytkin. Automaattisesti toimiva sähkökytkin valitsee ensisijaisen ja toissijaisen sähkölähteen väliltä kuormalle sähköä välittävän sähkölähteen. OMD800-laite voi lähettää tilatietoja siihen liitetystä sähköverkosta ja vallitsevasta kytkentätilasta Modbus-protokollalla.

Tämän opinnäytetyön tarkoituksena oli kehittää ohjelmisto, jolla voidaan havainnollistaa graafisen kaavion avulla OMD800-laitteen toimintatiloja tietokoneohjelmassa. Tämä toimintojen esitys on mahdollista tehdä sekä ilman OMD800-laitetta että laitteen kanssa. Ohjelmisto on kehitetty OMD-laitteiden opetus- ja myyntitarkoituksiin.

Työn tuloksena syntynyt OMD Explorer-ohjelmisto kehitettiin käyttäen jatkokehitettäviä prototyyppejä. Kehityksessä otettiin huomioon projektin aikana muuttuneet ja lisääntyneet vaatimukset. Ohjelmisto täytti asiakkaan sille asettamat toiminnalliset vaatimukset, eli ohjelmisto kommunikoi Modbus-protokollalla OMD800-laitteen kanssa, ja kommunikoinnista saatu data esitetään käyttäjälle. Ohjelmisto toimii myös ilman fyysistä OMD800-laitetta, jolloin käyttäjälle esitetään tyypillinen verkon virhetilanne ja vaiheet jotka suoritetaan virhetilanteen sattuessa.

Asiasanat	OMD automaattinen kytkin, OMD Explorer, Modbus, MS C# .NET, Ohjelmistokehitys, Prototypointi

VAASAN AMMATTIKORKEAKOULU

UNIVERSITY OF APPLIED SCIENCES

Tietotekniikan koulutusohjelma

ABSTRACT

Author	Harri Hurtola
Title	OMD Explorer Software for ABB OMD800 Device
Year	2009
Language	Finnish
Pages	52 + 2 appendices
Name of Supervisor	Ghodrat Moghadampour

This thesis describes the developing process of OMD Explorer Windows application written in Microsoft C#.NET for communicating with an ABB switch controller called OMD800. The OMD-device family is manufactured by ABB Low Voltage Products. The application, which runs on a computer, communicates with the device via Modbus protocol.

Conventional power switches can be converted to automatic transfer switches by means of the OMD-devices. Automatic transfer switch monitors the primary power source and the secondary power source attached to the inputs, and chooses between these the one that powers the load on the output. The OMD800-device can send its status and power source information via Modbus protocol.

The aim of this thesis was to engineer an application that can demonstrate graphically the functionalities of the OMD800-device with and without real connection to the OMD800 device. The application has been developed mainly for sales and education purposes of the OMD-device family.

The objectives were met and the OMD Explorer application was developed successfully according to customer requirements. The application was developed through evolutionary prototyping model and changing product requirements were taken into account in iterative processes. The application can successfully communicate with the real OMD800-device via Modbus protocol and demonstrate its behaviour. It can also simulate the device behavior in off-line mode showing all the real steps in an imaginary situation with and without random switch failure cases.

Keywords	OMD automatic transfer switch, OMD Explorer, Modbus, MS C#.NET, Software development, Evolutionary prototyping

MERKIT JA LYHENTEET

USB	Universal Serial Bus, universaali sarjaportti. Nykyisissä tietokoneissa käytettävä sarjaliikenneportti.
RS-232	Tietoliikenneportti, jonka data on sarjamuotoista. Vanhemmissa tietokoneissa sekä teollisuuden laitteissa useimmin esiintyvä sarjaliikenneportti.
RS-482/RS-485	Tietoliikenneväylä, jonka data on sarjamuotoista, mutta asynkronista. RS-485-kommunikointiin riittää yksi parikaapeli.
OMD	ABB:n kehittämä automaattinen kytkinperhe.
OMD800	Kytkinperheen toiminnoiltaan kattavin malli, jota tässä työssä käytettiin.
PC	Personal Computer, tietokone.
CRC	Cyclic Redundancy Check, virheentarkistusalgoritmi

SISÄLLYS

1 JOHDANTO

Meitä ympäröivä maailma pyörii nykyään sähkön voimalla. Sähkön saatavuutta pidetäänkin keskuudessamme lähes itsestäänselvyytenä. Sähköä tarvitaan niin asumisessa, teollisuudessa kuin liikenteessäkin. Sähkön toimituksessa kotitalouksille on todella harvoin ongelmia varsinkin kaupunkialueilla, ja katkokset toimituksessa ovat lähinnä haittaavaa luonnetta. Sitäkin suuremman haitan sähkön toimitusvaikeudet aiheuttavat liikenteelle, teollisuudelle ja esimerkiksi sairaaloille. Yhteistä näille kaikille käyttökohteille on se, että jos ongelmia syntyy, on sekä asiakkaan että toimittajan etu saada sähköntoimitukset jatkumaan mahdollisimman pian vian ilmenemishetken jälkeen.

Sähköä voidaan syöttää kohteisiin eri reittejä pitkin. Tällaisiin reittien risteyskohtiin sähkökatkosten varalle tarvitaan laitteisto, joka tunnistaa sähkön toimituksen häiriöt sekä mahdollisuuksien mukaan vaihtaa kuormaan kytketyn sähköverkon toiseen, joka pystyy toimittamaan sähköä kuormalle. Teollisuudessa ja esimerkiksi sairaaloissa sähkönjakelun solmupisteessä, toinen sähkön tuottaja voi olla toisen sähköverkon sijaan varavoimala tai generaattori, joka täytyy käynnistää sähkökatkon sattuessa.

Joissakin tällaisissa sähkönkäyttökohteissa sähköverkon valinta kahden sisääntulon välillä vaatii manuaalista työtä, jolloin sähkökytkin täytyy käydä kääntämässä ensisijaiselta verkolta toissijaiselle verkolle tai päinvastoin. Myös generaattori saattaa vaatia manuaalisen käynnistyksen. Tällainen manuaalinen työ hidastaa verkon sähköistämistä uudelleen mahdollisen virhetilanteen jälkeen.

Nämä ongelmat silmälläpitäen ABB on valmistanut OMD-tuoteperheen. Tuoteperheen laitteet liitetään sähköverkon kytkimiin, jolloin kytkimistä tulee toiminnoiltaan automaattisia. OMD-laite tutkii jatkuvasti kahden sähkölähteen tiloja, ja jos ensisijaiseksi sähkölähteeksi määritetyssä sähköverkossa tapahtuu häiriö, kytketään toissijainen sähköverkko kytkimen perässä olevaan kuormaan kiinni. Tämän automatisoidun vaihtotapahtuman ansiosta laitteistoon kytketty kuorma on mahdollisimman vähän aikaa ilman sähköä, eikä tällainen vaihtotoimenpide tarvitse lainkaan manuaalista työtä. Osa OMD-tuoteperheen laitteista sisältää gene-

raattoriohjauksen, jolloin ne voivat ohjata suoraan laitteeseen kytketyn generaattorin käynnistystä ja sammutusta tarpeen vaatiessa.

Yksi OMD-tuoteperheen laitteista on OMD800, johon tässä opinnäytetyössä on keskitytty. Tämä laite on ominaisuuksiltaan tuoteperheensä monipuolisin, sisältäen mm. mahdollisuuden lähettää tilatietoja kytkimen ja siihen kytkettyjen sähköverkkojen sen hetkisestä tilasta Modbus-protokollalla eteenpäin halutulle kohteelle, esimerkiksi sähköverkon valvomoon.

Tässä opinnäytetyössä on kehitetty tietokoneella ajettava ohjelma, jolla voidaan esittää OMD-tuoteperheen toimintoja myynti- ja opetustarkoituksissa. Ohjelmalla voidaan esitellä laitteiden toimintoja erilaisissa tilanteissa joko itsenäisesti ilman oikeaa yhteyttä OMD-laitteeseen tai yhdistämällä tietokone OMD800-laitteeseen. Tällöin tietokoneella ajettava ohjelma saa tilatietonsa oikealta OMD800-laitteelta, joka lähettää tilatietonsa Modbus-protokollalla.

Ohjelman tarkoitus on havainnollistaa käyttäjälle OMD-tuoteperheen laitteiden toimintaa. Ohjelmalla näytetään käyttäjälle OMD-laitteen ohjaaman kytkimen toimintaa ja sitä miten verkon vaihto kahden sähköverkon välillä tapahtuu ongelmatilanteessa sekä kuinka generaattoriohjaus toimii.

1.1 ABB Oy:n esittely

ABB Oy on kansainvälinen sähkövoimatekniikkaan ja automaatiotekniikkaan erikoistunut konserni, jolla on toimintaa noin sadassa maassa. Suomessa toimintaa on yli 40:llä eri paikkakunnalla, joista yksi on Vaasa. ABB:n pääperiaatteita ovat muun muassa energiatehokkuuden parantaminen, ilmastonmuutoksen hidastaminen ja ympäristön huomioon ottaminen tuotannon eri vaiheissa. ABB:n ydinliiketoimintoihin kuuluvat sähkövoimatuotteet, sähkövoimajärjestelmät, automaatiotuotteet, prosessiautomaatio ja robotit.

Automaatiotuotteisiin kuuluva yksikkö, Pienjännitekojeet, valmistaa OMD-laiteperhettä, joka kytkimen kanssa muodostaa automaattisen sähkökytkimen. Tässä opinnäytetyössä on keskitytty OMD-tuoteperheen malliin 800, koska sen ominaisuuksiin kuuluu mahdollisuus välittää laitteen tilatietoja Modbus-protokollalla.

1.2 OMD-tuoteperhe ja OMD800-laite

ABB Oy:n valmistama OMD-tuoteperhe koostuu neljästä laitteesta: OMD100, OMD200, OMD300 ja OMD800 (kuva 1) /3/. Tuoteperheen tuotteet ovat ohjainlaitteita sähköverkon kytkimille. OMD-laite yhdistettynä sähkökytkimeen muodostaa automaattisia kytkimiä. Sähkökytkimessä on kaksi sisääntuloa sähkölähteille ja yksi ulostulo sähkölle. Niihin voidaan siis kytkeä kaksi eri sähkölähdettä, joista toinen on ensisijainen ja toinen toissijainen. Sähkökytkimen ulostuloon kytketään se kuorma, joka laitteiston avulla on tarkoitus sähköistää.

Kaikkien tuoteperheen laitteiden toimintaperiaate on samanlainen. Laite mittaa jatkuvasti siihen kytkettyjen sähkölähteiden arvoja ja tekee tämän tiedon pohjalta ratkaisun käytettävästä sähköverkosta. Jos ensisijaisen sähkölähteen sähkön parametrit (jännite, taajuus) muuttuvat OMD-laitteeseen esisäädettyjen parametrien ulkopuolelle tai sähkön saanti katkeaa, OMD-laite kytkee ensisijaisen verkon irti ja toissijaisen verkon kiinni kuormaan automaattisesti.

OMD200-laitteesta lähtien toinen sähkölähde voi olla myös generaattori, joka saa käynnistyskäskyn OMD-laitteelta jos toista sähkölähdettä tarvitaan. Molempia sähkön sisääntuloja analysoidaan jatkuvasti, ja kun ensisijaisen verkon sähkö on taas kriteerit täyttävää, toissijainen verkko kytketään irti ja ensisijainen kytketään ulostuloon. Juuri tämän toiminnan havainnollistamiseen ja esittämiseen tämän opinnäytetyön ohjelma on suunniteltu.

Kuva 1. OMD-tuoteperhe, OMD800-laite ja kytkin ylimpänä. /2/

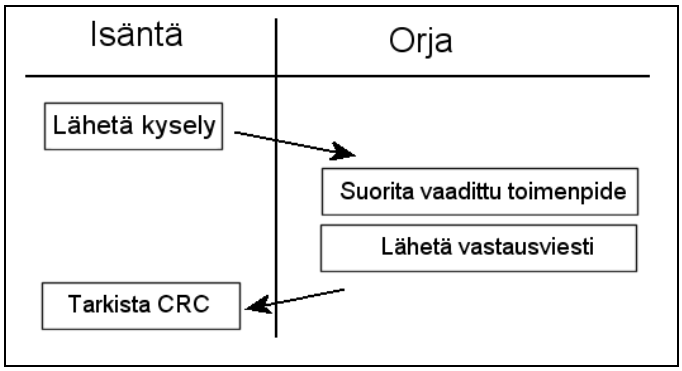
OMD-tuoteperheen laitteiden ominaisuudet kasvavat suurempaa mallinumeroa kohti, OMD800-laite on tuoteperheensä kattavin toiminnoiltaan. Tässä opinnäytetyössä käsitellään juuri OMD800-laitetta, koska sen toiminnasta ja verkon virhetilanteista saadaan välitettyä informaatiota Modbus-protokollan avulla, jota voidaan edelleen lukea vaikkapa laitteeseen kytketyn tietokoneen avulla. Tähän tietojen lukuun tarvitaan ohjelma, joka osaa tulkita Modbus-protokollan viestejä.

1.3 Modbus-protokolla

Modbus-protokolla on viestintätapa, jota käytetään yleisesti monissa asiakas – palvelin tai isäntä – orja–muotoisissa laitekokonaisuuksissa. Tätä protokollaa käytetään laajasti niin teollisuudessa kuin rakennuskohteissakin, energiajärjestelmissä ja pitkien etäisyyksien välisessä yksinkertaisessa tiedonsiirrossa.

Tyypillinen sovelluskohde Modbus-protokollasta on vähintään kahdesta laitteesta koostuva kokonaisuus, jossa toinen laite ohjaa jollain tavalla toista laitetta. Tällöin yksittäisistä laitteista saadaan tehtyä kokonaisuuksia, joiden toiminnot riippuvat saman verkon muista laitteista.

Esimerkki yksinkertaisesta kahden laitteen ratkaisusta voisi olla kuljetinhihna-yksikkö ja saha. Nämä kaksi laitetta kommunikoivat keskenään, toinen toimii isäntänä (saha) ja toinen asiakkaana (kuljetinhihna) jolloin saha ohjaa kuljetinhihnan toimintaa. Tarvittaessa tähän kuviteltuun laitteistoon voitaisiin liittää useampiakin laitteita, isäntä – asiakas -suhdetta voi myös muuttaa, jolloin lopputuloksena voisi olla esimerkiksi tietokoneella ohjattu sahakokonaisuus. Kuvassa 2 on havainnollistettu isäntä – asiakas –muotoista kommunikointia:



Kuva 2. Modbus-kommunikointi.

Modbus-verkko koostuu aina yhdestä isännästä, joka kysyy eri arvoja orjalaitteista tai asettaa niitä orjalaitteisiin. Orjalaitteita voi kytkentätavasta riippuen olla 1-31 kappaletta, ja orjalaitteet erotetaan toisistaan antamalla niille niin sanottu ”Slave ID”, eli yksilöity tunnus. Näitä numeromuotoisia tunnuksia on yksi jokaisella laitteella, ja mahdollisia tunnusarvoja ovat arvot väliltä 1-247.

Modbus-verkon kaapelointiin voidaan käyttää RS485-kaapelointia. Tällöin tietoliikenteen luomiseen riittää kierretty parikaapeli, jonka avulla verkkoon kytketyt laitteet voivat olla hyvinkin etäällä toisistaan, tietoliikenneyhteys saavutetaan vielä 1,2 kilometrin päästä.

Modbus-viestinnässä on kaksi tapaa välittää viestejä, Modbus ASCII-tila ja Modbus RTU-tila. Tässä opinnäytetyössä on käytetty RTU-tilaa (Remote Terminal Unit). Tässä tilassa jokainen 8-bittinen viesti sisältää kaksi heksadesimaaliarvoa (mahdolliset arvot ovat numerot 0-9 ja kirjaimet A-F) ja yksi heksadesimaaliarvo edustaa siis neljää bittiä per arvo. Suurimpana erona RTU ja ASCII-tiloissa on ASCII-tilan parempi virheensietokyky ja RTU-tilan suurempi tiedonsiirtonopeus.

Modbus-viestinnässä isäntä voi lähettää orjalle erilaisia toimintapyyntöjä. Toimintapyynnöt on numeroitu, jolloin jokainen numero tarkoittaa erilaista toimintoa. Toiminnot (esimerkiksi rekisterien luku tai kirjoitus) riippuvat käytettävissä olevasta laitteistosta, mutta mahdollisia toimintokoodeja ovat arvot väliltä 1-255.

Modbus-viestit ovat rakenteeltaan aina samanlaisia, koostuen taulukossa 1 esitetyistä osista:

Taulukko 1. Modbus-viestin rakenne. /7/

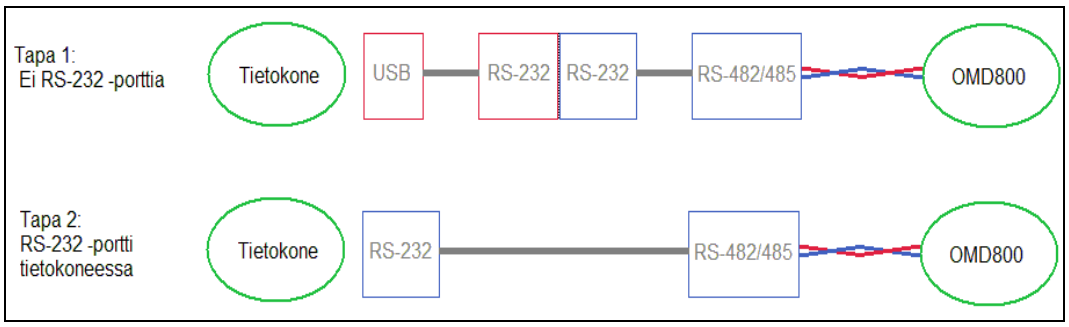
Kenttä	**Selite**	**Mahdolliset arvot**
Laiteosoite	Vastaanottajan osoite (slave)	1-247
Toiminnon koodi	Koodi määrittää kyselyn tyypin	1-255
Data	Siirrettävä data	00-FF
CRC	Tarkistesumma	-

Modbus-protokolla on vakiinnuttanut lähes standardiaseman teollisuudessa ja muissa käyttökohteissa pitkän ikänsä ja avoimuutensa vuoksi. Vaikka Modicon-niminen yritys kehitti sen jo vuonna 1979, on Modbus voimissaan yhä tänäkin päivänä, ja sen kehittämistä jatketaan edelleen. /4/, /1/, /5/.

1.4 Tietoliikenneyhteys OMD800-laitteen ja tietokoneen välillä

Tietoliikenneyhteys OMD-laitteen ja tietokoneen välille luodaan käyttäen tarvittavia adaptereita eli muuntimia. Tarvittavien muuntimien tyypit riippuvat käytettävästä kokoonpanosta tietokoneessa. Jos tietokoneessa on USB-portti, tarvitaan USB – RS232- ja RS232 – RS485 -muuntimet. Jos tietokoneessa on jo valmiiksi RS232–portti, tarvitaan ainoastaan RS232 – RS485-muunnin. Viimeksi mainitusta

tietoliikenneyhteys toimitetaan kierretyllä parikaapelilla OMD800-laitteen päällä olevaan liitäntään X51 /3/. Kuvassa 3 on esitetty laitteiston kytkentätavat.



Kuva 3. Tietokoneen ja OMD800-laitteen kytkentämahdollisuudet.

Kommunikointi OMD-laitteen ja tietokoneen välillä tapahtuu käyttäen Modbus-protokollaa. Modbus-protokollan viestinnässä isäntä (master) lähettää kyselyn, johon kyselyssä osoitettu orjalaite (slave) vastaa. Opinnäytetyön ohjelma toimii tässä tapauksessa isäntänä, joka aloittaa kyselyn, johon OMD-orjalaite lähettää vastauksen.

2 PROJEKTIN KUVAUS

Ohjelmiston oli, tarkoituksensa puolesta, oltava niin yksinkertainen kuin mahdollista, jotta sitä voisi käyttää satunnainen laitteiston myyntihenkilö, joka ei olisi ennen ohjelmaa käyttänytkään. Ohjelman tuli myös olla mahdollisimman selkeä ja informatiivinen opetustilanteita ja myyntitilanteita silmälläpitäen, ja tähän tarkoitukseen kaaviopohjainen graafinen esitys laitteiston toiminnasta oli selkein ratkaisu. Koska ohjelman käyttöympäristö on kansainvälinen, valittiin ohjelman käyttökieleksi englanti. Projektin edetessä ilmeni tarve saada ohjelman tekstit myös saksaksi. Ohjelmaan kehitettiin kielenvalintamahdollisuus, mutta oletuksena ohjelman kieli on englanti.

Ohjelmisto tuli olla käytettävissä millä tahansa työasemalla ABB:llä. Vaatimus oli myös, että ohjelma toimisi suoraan niin, ettei sitä tarvitsisi asentaa tietokoneeseen vaan pelkkä kopiointi riittäisi. Ohjelman tuli myös toimia ilman minkään lisätyökalun asentamista tietokoneeseen.

Ohjelman toteutus valittiin tapahtuvaksi Microsoftin kehittämällä C# -kielellä. Tämä valinta oli looginen, koska opinnäytetyön alkuvaiheissa oli käynnissä samanaikaisesti opintoihin kuuluva C# -kurssi. Siten ohjelmointikieli ja ohjelmointiympäristö olivat entuudestaan tuttuja. Ohjelman toteutus tapahtui käyttäen Microsoftin Visual Studio 2005-ohjelmointiympäristöä.

Visual Studio 2005-ohjelmointiympäristö käyttää Microsoftin .NET Framework-ohjelmistokomponenttikirjastoa, josta tulee olla asennettuna tietokoneeseen vähintään versio 2.0. Komponenttikirjasto ei sisälly Windows XP-käyttöjärjestelmään valmiina, vaan se tulee asentaa erikseen. Windows Vistan mukana tulee .NET Frameworkista versio 3.0, ja Windows 7:n mukana tulee versio 3.5, joka on tällä hetkellä viimeisin vakaa julkaistu versio.

Projektin edetessä ilmeni, että ABB:n tietokoneissa tätä .NET Framework-laajennusta ei ollut asennettuna, mutta se saatiin kuitenkin hankittua ja asennettua nopeasti. Tämän pohjalta vaatimuksia lisäosien asentamiselle hieman helpotettiin ja projektia voitiin siten kehittää eteenpäin.

Ohjelma suunniteltiin resoluutiolle 1024 x 768. Muita vaatimuksia ohjelmalle oli tietoliikenneyhteys OMD-laitteeseen Modbus-protokollaa käyttäen. Tämä tietoliikenneyhteys on mahdollista toteuttaa OMD-laitteessa olevasta RS-485-portista. Jotta tietoliikenneyhteys saadaan toimitettua tietokoneeseen saakka, tarvitaan erilaisia muuntimia kaapelille. Tähän tarkoitukseen ABB tarjosi ohjelmiston kehitysvaiheessa USB – RS-232-muuntimen ja edelleen RS-232 – RS-485-muuntimen.

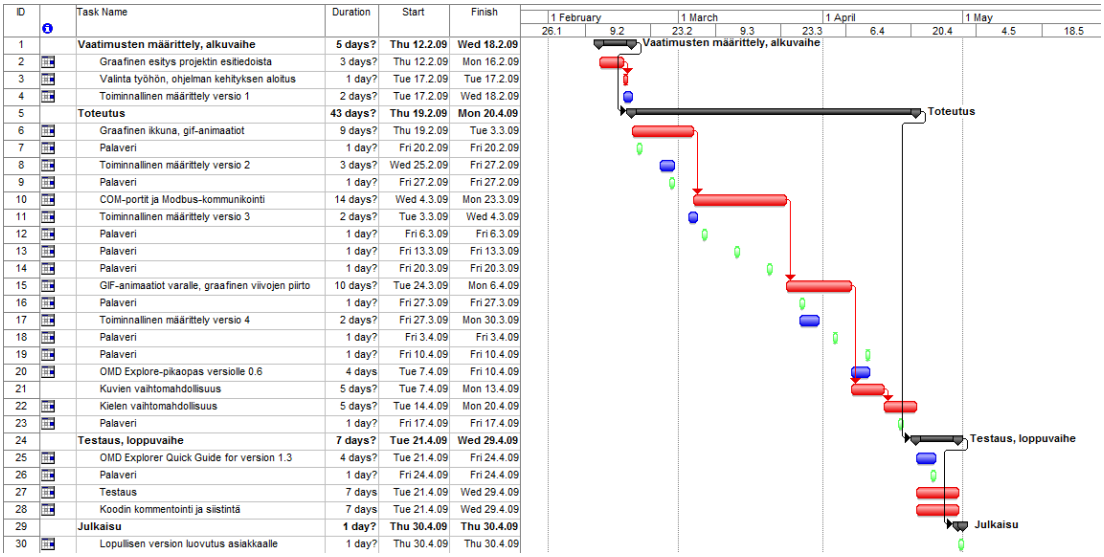
Ohjelmassa tuli olla kaksi toimintatilaa. Ensimmäinen tila kuvaa OMD-laitteen toimintaa itsenäisesti, ilman fyysistä kytkentää ja tietoliikenneyhteyttä itse laitteeseen. Toinen tila kuvaa oikean OMD–laitteen toimintaa: laite kytketään tarvittavilla adaptereilla tietokoneen COM- tai USB-porttiin, jonka jälkeen OMD-laitteen sen hetkinen tila ja kytkentä ovat luettavissa ohjelmasta tietoliikenneyhteyden välityksellä sekä kaavio- että tekstimuodossa.

2.1 Aikataulu

Projekti käynnistyi 12. helmikuuta 2009. Ensimmäinen vaihe projektissa oli demo-ohjelman tekeminen annettujen lähtötietojen pohjalta. Tämä demo-ohjelma koostui alkeellisesta animaatiosta, jossa kuvattiin OMD-laitteiden toimintaa. Samanaikaisesti kehitettiin ensimmäistä versiota toiminnallisesta määrittelystä, joka valmistui 19. helmikuuta. Projekti jatkui Modbus-protokollan tutkimisella ja ohjelman edelleen kehittämisellä.

Projektia tehtiin aina huhtikuun loppuun saakka. Karkea arvio koko ohjelmiston tekemiseen kuluneesta ajasta on noin 300 tuntia. Ohjelmiston tuli olla valmis huhtikuun keskivaiheilla. Tämä suunniteltu valmistumisajankohta kuitenkin ylitettiin uusien ja muuttuneiden vaatimusten ja toimintojen vuoksi. Perustoiminnoiltaan toimiva versio saatiin asiakkaan käyttöön toivottuna ajankohtana 18. huhtikuuta. Tämän jälkeenkin suoritettiin vielä lopullista testausta ja virheiden korjausta. Viimeinen tapaaminen ABB:n kanssa oli 30. huhtikuuta, jolloin ohjelmistosta luovutettiin viimeinen versio lähdekoodeineen.

Projektin aikataulua on havainnollistettu kuvassa 4.



ID	Task Name	Duration	Start	Finish
1	**Vaatimusten määrittely, alkuvaihe**	**5 days?**	**Thu 12.2.09**	**Wed 18.2.09**
2	Graafinen esitys projektin esitiedoista	3 days?	Thu 12.2.09	Mon 16.2.09
3	Valinta työhön, ohjelman kehityksen aloitus	1 day?	Tue 17.2.09	Tue 17.2.09
4	Toiminnallinen määrittely versio 1	2 days?	Tue 17.2.09	Wed 18.2.09
5	**Toteutus**	**43 days?**	**Thu 19.2.09**	**Mon 20.4.09**
6	Graafinen ikkuna, gif-animaatiot	9 days?	Thu 19.2.09	Tue 3.3.09
7	Palaveri	1 day?	Fri 20.2.09	Fri 20.2.09
8	Toiminnallinen määrittely versio 2	3 days?	Wed 25.2.09	Fri 27.2.09
9	Palaveri	1 day?	Fri 27.2.09	Fri 27.2.09
10	COM-portit ja Modbus-kommunikointi	14 days?	Wed 4.3.09	Mon 23.3.09
11	Toiminnallinen määrittely versio 3	2 days?	Tue 3.3.09	Wed 4.3.09
12	Palaveri	1 day?	Fri 6.3.09	Fri 6.3.09
13	Palaveri	1 day?	Fri 13.3.09	Fri 13.3.09
14	Palaveri	1 day?	Fri 20.3.09	Fri 20.3.09
15	GIF-animaatiot varalle, graafinen viivojen piirto	10 days?	Tue 24.3.09	Mon 6.4.09
16	Palaveri	1 day?	Fri 27.3.09	Fri 27.3.09
17	Toiminnallinen määrittely versio 4	2 days?	Fri 27.3.09	Mon 30.3.09
18	Palaveri	1 day?	Fri 3.4.09	Fri 3.4.09
19	Palaveri	1 day?	Fri 10.4.09	Fri 10.4.09
20	OMD Explore-pikaopas versiolle 0.6	4 days	Tue 7.4.09	Fri 10.4.09
21	Kuvien vaihtomahdollisuus	5 days?	Tue 7.4.09	Mon 13.4.09
22	Kielen vaihtomahdollisuus	5 days?	Tue 14.4.09	Mon 20.4.09
23	Palaveri	1 day?	Fri 17.4.09	Fri 17.4.09
24	**Testaus, loppuvaihe**	**7 days?**	**Tue 21.4.09**	**Wed 29.4.09**
25	OMD Explorer Quick Guide for version 1.3	4 days?	Tue 21.4.09	Fri 24.4.09
26	Palaveri	1 day?	Fri 24.4.09	Fri 24.4.09
27	Testaus	7 days	Tue 21.4.09	Wed 29.4.09
28	Koodin kommentointi ja siistintä	7 days	Tue 21.4.09	Wed 29.4.09
29	**Julkaisu**	**1 day?**	**Thu 30.4.09**	**Thu 30.4.09**
30	Lopullisen version luovutus asiakkaalle	1 day?	Thu 30.4.09	Thu 30.4.09

Kuva 4. Projektin aikataulu.

Aikataulukaaviossa projektin eri vaiheet on kuvattu mustalla, ohjelman kehitys-kohdat punaisella, dokumentointi sinisellä ja pidetyt palaverit vihreällä.

2.2 Kehitysympäristön laitteet

Opinnäytetyöprojektin keskeiset laitteet (tietokone poislukien) on esitetty alla olevassa kuvassa 5. Laitteet on numeroitu kuvaan havainnollistamisen vuoksi. Kuvan numeroidut laitteet ovat:

1 OMD800-laite

2 Sähköverkon kytkin, jota OMD-laite ohjaa

3 Sähkökytkin, jolla voitiin simuloida ensisijaisen verkon sammuminen ja gene-raattorin käynnistys

4 ADAM 4520 RS-232 – RS-485 –muunnin (kuva 6)

5 Erillinen käyttösähkölähde OMD-laitteelle

Kuva 5. Tietokoneeseen kytketty laitteisto.

Kuva 6. Työssä käytetty ADAM-4520 RS-232 – RS-485 -muunnin.

2.3 Vaatimusten keruu

Projektin alkuvaiheessa ABB:n vaatimusten pohjalta oli syntynyt vaatimusmäärittelyn ensimmäinen versio. Tätä vaatimusmäärittelyä kehitettiin projektin alussa ja hieman projektin aikanakin. Kehitettävän ohjelmiston vaatimukset luokiteltiin numeroasteikolla 1-3, jossa numero 1 vastaa tärkeintä ominaisuusluokkaa. Tämän luokan ominaisuudet pitäisi ohjelmaan kehittää. Luokka 2 kuvaa ominaisuusluokkaa, joka ohjelmaan tulisi kehittää ja luokka 3 kuvaa ominaisuusluokkaa, joka olisi hyvä ohjelmassa olla (must have, should have, nice to have). Viimeisimmässä toiminnallisen määrittelyn versiossa on vaatimuksina seuraavat taulukon 2 seikat:

Taulukko 2. Toiminnallisen määrittelyn vaatimukset lajiteltuna tärkeyden mukaan.

Viittaus	**Kuvaus**	**Tärkeys (1 = tärkein)**
T1	COM sarjaporttiyhteys, pariteetti, stop, nopeus (USB)	1
T2	Lue OMD-laitteen rekisteri Modbus-protokollalla	1
T3	Lue OMD-laitteen konfigurointitiedosto (XML)	1
T4	Näytä laitteen arvot Textbox-kentässä	1
T9	Näytä laitteen ID	1
L1	Yhteys laitteeseen käyttäen USB-RS485 -kaapelia	1
L2	Modbus Analog Devices ADM483	1
T5	Näytä rekisterien arvot graafisessa käyttöliittymässä	2
T6	Näytä hälytysarvot koodilla ja selväkielisenä	2
T8	Tallenna tilatiedot lokitiedostoon	2
T7	Kysele hälytysarvot määräajoin	3
L3	Modbus etäyhteys TCP/IP:llä	3

Toiminnallisen määrittelyn (Functional Specification) viimeisin versio on tämän opinnäytetyön liitteenä (LIITE 1).

2.4 Vaatimusten analysointi

Ohjelmalle annetut vaatimukset eivät olleet turhan korkealla projektin alkuvaiheessa, niinpä suurin osa vaatimuksista saatiin toteutettua ohjelmaan. Muutama vaatimus osoittautui tarpeettomaksi. Ohjelma ei esimerkiksi mitenkään pysty lukemaan Modus-väylän läpi OMD-laitteen asetusrekisteriä.

Vaatimusmäärittelyyn tuli projektin edetessä lisäominaisuuksia, jotka luokiteltiin tärkeydelle 2. Nämä projektin edetessä ilmi tulleet lisävaatimukset implementoitiin ohjelmaan mukaan. Lisävaatimukset on esitetty taulukossa 3.

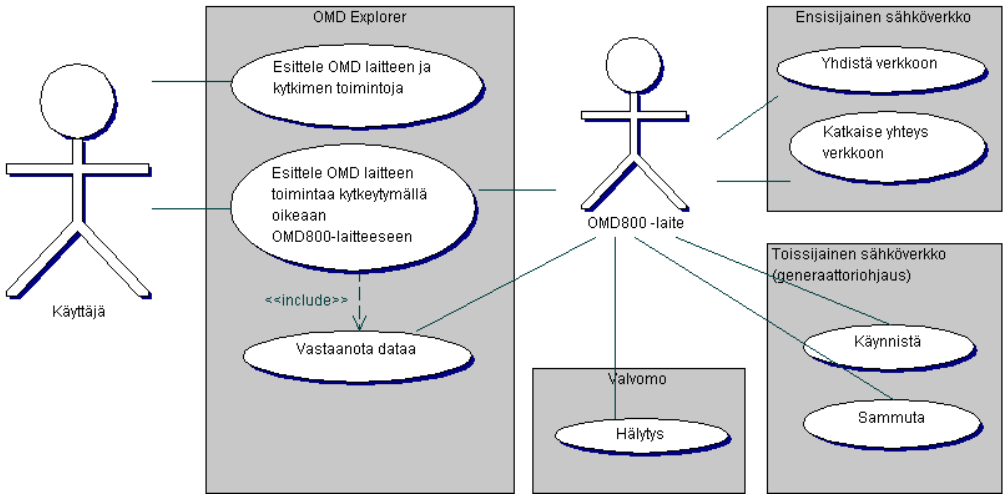
Taulukko 3. Lisävaatimukset ohjelmalle.

Viittaus	**Kuvaus**	**Tärkeys**
L4	Kaavion piirto ohjelmallisesti	2
L5	Sähkölähteiden ja kuorman kuvat vaihdettaviksi	2
L6	Taustakuva vaihdettavaksi	2
L7	Kielivaihtoehtovalikko: englanti ja saksa	2

Lopullisesta ohjelmasta löytyvät kaikki yllämainitut vaatimukset, paitsi taulukon 2 kohdat T3, T8 ja L3.

2.5 Mallinnus

Ohjelman käyttötapakaavio (kuva 7) on suhteellisen yksinkertainen, koska ohjelmakin on suunniteltu yksinkertaiseksi käyttää.



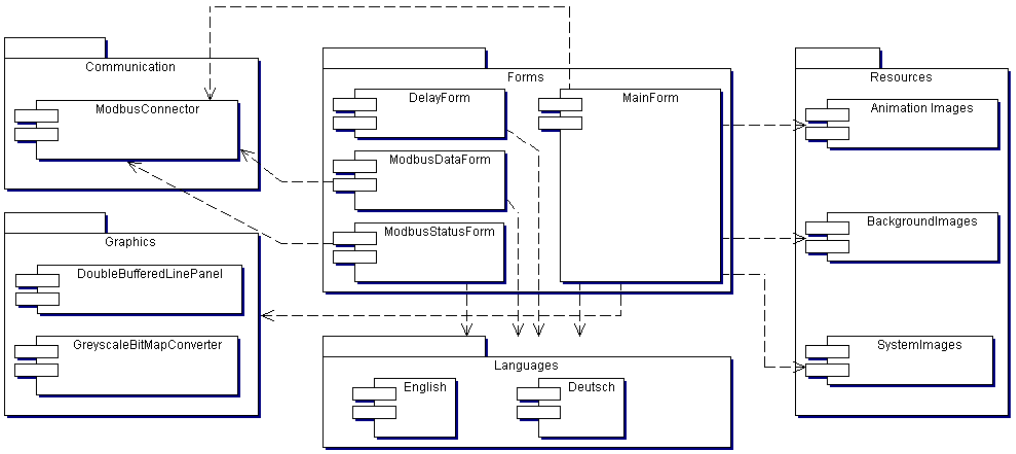
Kuva 7. Järjestelmän toiminnan kuvaus.

Ohjelmassa on kaksi toimintatilaa, tietojen esitystila sekä oikean laitteen käyttötila. Ohjelma käynnistyy ensin tietojen esitystilaan, josta käyttäjä voi vaihtaa tilan oikean laitteen käyttötilaan. Näin varmistutaan siitä, että ohjelmaa voidaan käyttää tietokoneessa joka ei ole kytketty OMD800-laitteeseen, eikä täten yritetäkään luoda Modbus-yhteyttä jo ohjelmaa käynnistettäessä.

OMD-laite osaltaan käsittelee siihen kytkettyjä sähkölähteitä automaattisesti. Ainoastaan tieto OMD-laitteen tilasta saadaan tuotua OMD Explorer-ohjelmaan. OMD-laitteen rekistereihin ei kirjoiteta, eikä sen asetuksia vaihdeta.

Ohjelmoinnissa on hyvä jakaa tuotettua koodia luokkiin, jotta pystytään pilkkomaan isoa kokonaisuutta pienempiin kokonaisuuksiin. Luokat on hyvä jakaa edelleen moduuleihin koodin toiminnan mukaan, jotta esimerkiksi grafiikkaan liittyvät luokat sijaitsevat yhdessä moduulissa ja kommunikointiin liittyvät omassaan.

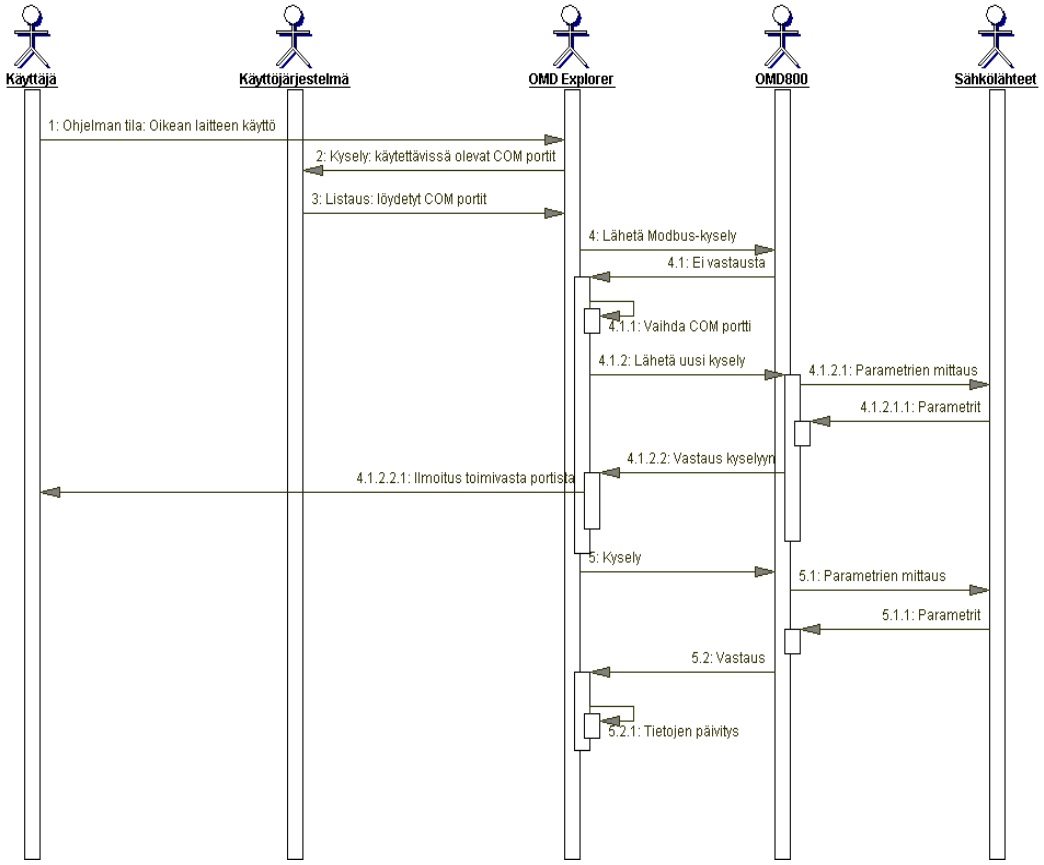
Ohjelman komponenttidiagrammi sisältää viisi moduulia, joihin ohjelman toimintoja on jaettu. Toimintojen jako eri moduuleihin selventää projektin hallintaa ohjelman kehitys- ja ylläpitovaiheessa runsaasti. Moduulit koostuvat seuraavista osista: resurssit, kielet, grafiikka, kommunikointi ja ikkunat. Alla olevaan kuvaan 8 on havainnollistettu ohjelman moduulijako, luokkajako sekä niiden väliset riippuvuudet.



Kuva 8. Ohjelman komponenttidiagrammi.

Jos ohjelmaa käytetään yhdessä oikean OMD-laitteen kanssa, ohjelma suorittaa yhteyden avauksen automaattisesti sekä ylläpitää yhteyttä itsestään niin, että käyt-

täjän ei tarvitse huolehtia Modbus-viestinnästä. Yhteyden muodostus on esitetty kuvassa 9.

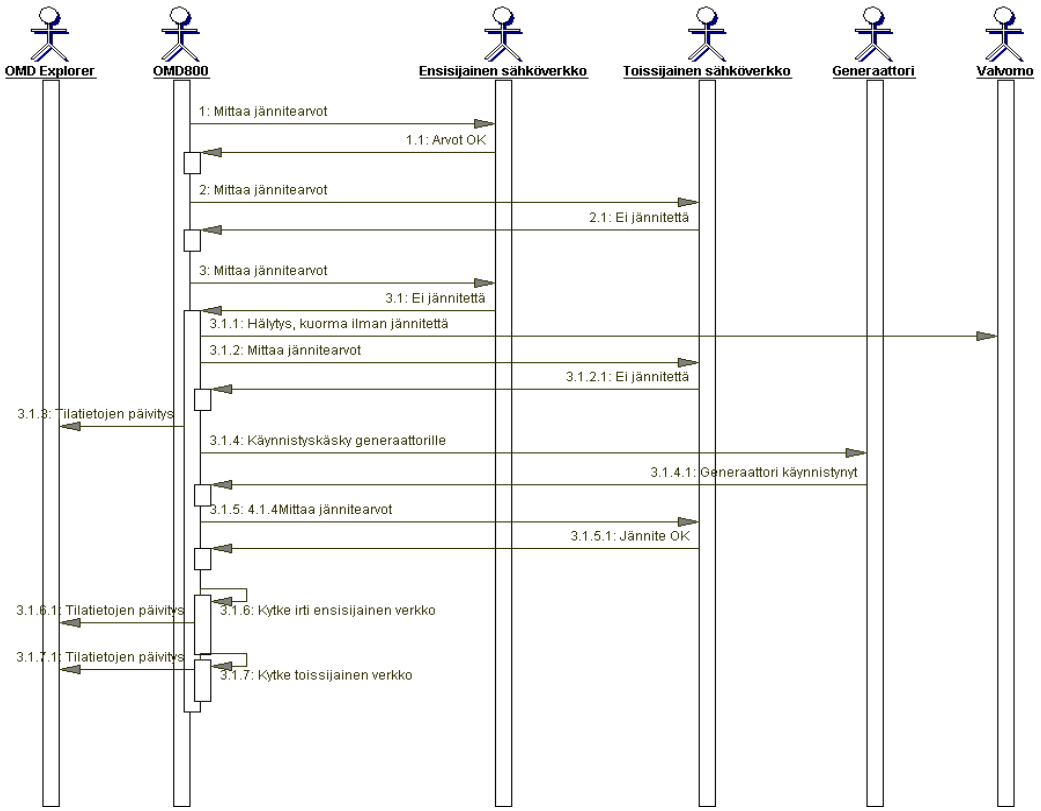


Kuva 9. Yhteyden muodostus ja ylläpito kaaviona.

Kun käyttäjä valitsee ohjelman käyttötilaksi oikean laitteen tilan, ohjelma kysyy käyttöjärjestelmältä tietokoneessa olevat COM-tietoliikenneportit. Tietoliikenneportteihin lähetetään portti kerrallaan yksi Modbus-kysely ja odotetaan vastausta. Jos vastausta ei saada, vaihdetaan tietoliikenneportti seuraavaan ja lähetetään uusi kysely. Tätä toistetaan niin kauan kunnes kaikki käyttöjärjestelmältä saadut tietoliikenneportit on käyty läpi. Jos vastaus Modbus-kyselyyn saadaan, on OMD800-laitteeseen kytketty tietoliikenneportti löydetty ja kyselyiden lähetystä porttiin jatketaan. Jos toimivaa tietoliikenneporttia ei löydy, käyttäjä saa siitä ilmoituksen.

Tyypillinen OMD-laitteilla käytännössä esiin tuleva virhetilanne on sellainen, jossa sähkön saanti ensisijaisesta verkosta katkeaa. Tällöin käynnistetään sarja tapahtumia, joilla sähköistetään toinen sähkölähde (jos toissijainen sähkölähde on

generaattori) sekä kytketään ensisijainen sähkölähde irti ja toissijainen sähkölähde kuormaan kiinni. Tämä toimenpide on esitetty kuvassa 10 sekvenssikaaviona.

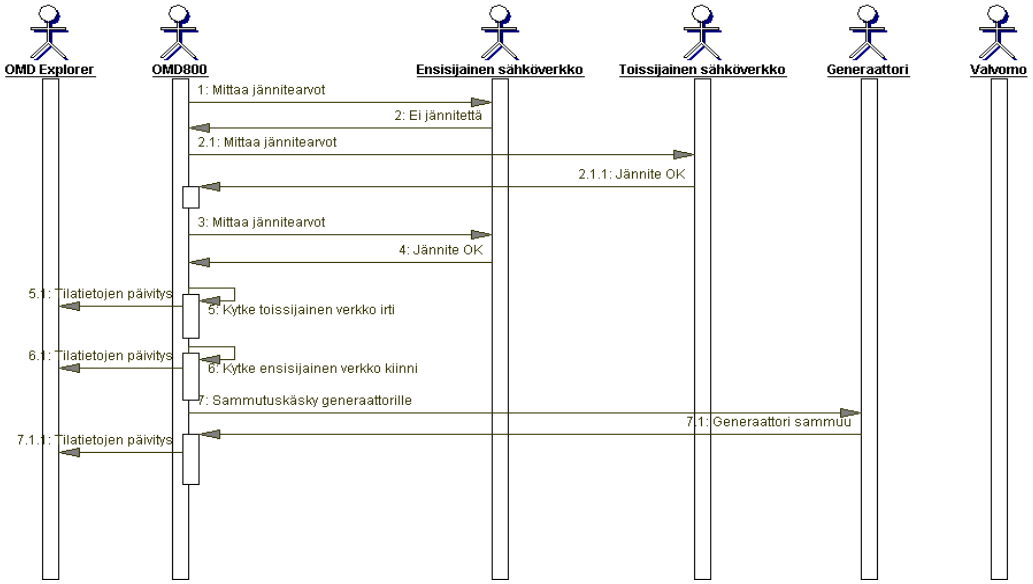


Kuva 10. Ensisijainen sähkölähde vikaantuu, generaattori käynnistetään ja kytketään kuormaan.

Sähköverkkojen parametreja mitataan jatkuvasti. Virhetilanteiden korjaantuessa, ensisijaisen verkon uudelleen sähköistyttyä, suoritetaan takaisinkytkentä ensisijaiseen verkkoon.

Tällöin toissijainen verkko kytketään kuormasta irti ja ensisijainen verkko kytketään takaisin kiinni kuormaan. Jos toissijaisena sähkölähteenä on generaattori, sille lähetetään sammutuskäsky. Vaihtosekvenssi on näin ollen käynyt läpi täyden kierroksen ja palannut alkuperäiseen luonnolliseen tilaansa, jossa ensisijainen sähköverkko on kytketty kuormaan.

Tämä tapahtumasarja on esitetty kuvassa 11.



Kuva 11. Ensisijainen verkko sähköistyy, toissijainen verkko kytketään irti ja generaattori sammutetaan.

3 TOTEUTUS

Ohjelmiston toteutus alkoi helmikuussa 2009, kun halutusta ohjelmasta tuli toteuttaa demonstraatioversio. Tässä vaiheessa ohjelma koostui vain ikkunasta, jossa oli yksi käynnistyspainike ja tila animaatiota varten. Itse animaationa toimi PowerPoint-dokumentista saatu animaatio.

Seuraavaksi ohjelmaan kehitettiin funktiot, joilla saatiin käyttöjärjestelmältä tiedot käytettävissä olevista tietoliikenneporteista. Samalla aloitettiin haastavaksi osoittautunut Modbus–protokollan sovellus ohjelmaan. Myöhemmin tietoliikenneporttien luku käyttöjärjestelmältä hoidettiin niin, että käyttäjän ei tarvinnut valita porttia itse, vaan ohjelma hoiti yhteyden käynnistyksen ja portin etsinnän automaattisesti.

Animaatioita hiottiin projektin edetessä paremmiksi. Gif-animaatioihin liittyneet ongelmat ja animaation nykiminen johtivat siihen, että koko animaatio tehtiin myöhemmin puhtaasti ohjelmallisesti piirtämällä viivat paneelielementtiin vaihe kerrallaan. Täten esityksestä ja itse kaaviosta saatiin selkeämpi ja uusia vaatimuksia paremmin palveleva. Valmiiseen ohjelmaan jätettiin myös alkuperäinen Gif–animaatio varalle, mutta oletuksena ohjelmassa toimi uusi versio animoinnista. Käytetyn animaatiotavan voi tarvittaessa vaihtaa ohjelman ylävalikosta.

Ohjelmaan kehitettiin projektin edetessä esiin tulleita uusia vaatimuksia, kuten ohjelman kielivaihtoehdoksi englannin lisäksi saksa. Animointitilassa tarvittava OMD-tuoteperheen ominaisuuksia paremmin esiin tuova animoinnin viiveasetus, kehitettiin niin ikään ohjelmaan.

Ohjelman kehityksen yhteydessä ilmeni tarve kehittää ohjelmalle käyttöohje. Tämä englanninkielinen, 15-sivuinen dokumentti, on liitetty tämän opinnäytetyön liitteeksi (LIITE 2). Ohjelman kaikki tärkeimmät toiminnot on esitetty tässä käyttöoppaassa ohjelman käynnistämisestä lähtien.

3.1 Kehitystyökalut

Ohjelmiston suunnittelussa ja toteutuksessa käytettiin kehitysympäristönä Microsoftin Visual Studio 2005-ympäristöä. Kyseinen kehitysympäristö tarjoaa kattavat työkalut seuraaviin ohjelmointikieliin: Visual Basic, Visual J#, Visual C++ sekä tässä projektissa käytettyyn Visual C# -kieleen.

Apuna kaavioiden piirrossa käytettiin Borlandin valmistamaa Together-suunnitteluympäristöä, jolla piirrettiin tässä dokumentissa olevia diagrammeja. Projektin aikataulukaavio toteutettiin käyttäen Microsoftin Projectia. Modbus-viestinnän ja rekisterien toimintaa tutkittiin käyttäen Modbus Poll-nimistä ohjelmaa /6/.

3.2 Luokkien toteutus ja jako

Ohjelman koodia pyrittiin jakamaan luokkiin sitämukaa kun koodia projektiin kertyi. Luokkiin jako selkeyttää ohjelman rakennetta luontivaiheessa kun yksittäisistä luokista ei tule fyysisesti liian laajoja. On myös järkevää jäsennellä ja yhtenäistää koodia niin, että esimerkiksi kommunikointiin liittyät asiat ovat samassa luokassa. Myös ohjelman mahdollinen jatkokehitys helpottuu kun olemassa oleva koodi on jäsennelty loogisesti.

Luokkien sisältämä koodi pyrittiin kommentoimaan mahdollisimman hyvin, jotta koodin tulkinta olisi mahdollisimman helppoa jopa sellaiselle henkilölle, joka on projektiin ulkopuolinen. Koodi on kommentoitu muutaman rivin välein. Kommenttien kielenä koodissa on käytetty englantia.

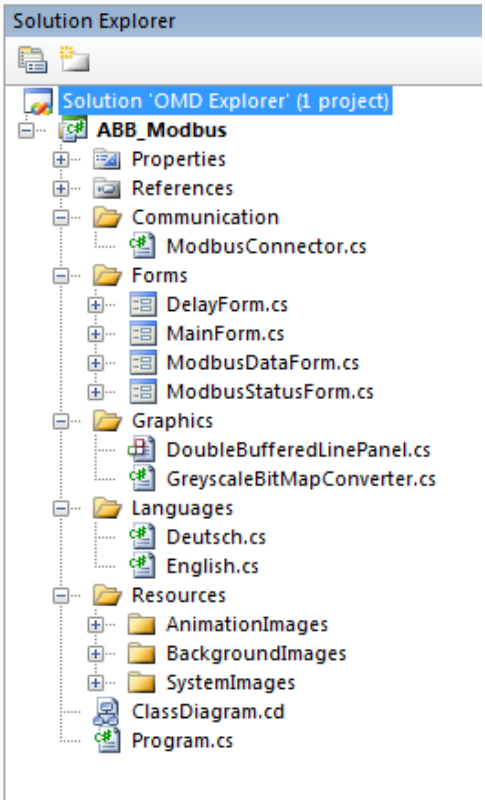
OMD Explorer-projekti koostuu moduuleista, joista jokaisella on oma tehtävänsä projektissa. Moduuleihin jako selkeyttää ohjelman ylläpitoa ja luokkajakoa.

Communication–moduuli sisältää ModbusConnector-luokan, joka sisältää aliohjelmat Modbus viestintään: vapaiden COM porttien etsintä, yhteyden avaus ja katkaisu, viestin luonti, lähetys, vastaanotto ja virheentarkistus.

Forms-moduuli sisältää kaikki ohjelmassa käytetyt formit, joista pääformi on MainForm. Ohjelman käynnistyessä luodaan myös ModbusDataForm, joka sisältää verbaalisen selityksen OMD-laitteen rekisterien arvoista. Ohjelman käyttötavasta riippuen luodaan myös DelayForm, jolla asetetaan viivetiloja sähkökytkennän eri kohdissa (animaatiotila) tai ModbusStatusForm (oikean laitteen käyttötila), johon tulee verbaalinen selitys laitteen sen hetkisestä tilasta.

Graphics-moduuli sisältää luokat, joilla käsitellään ohjelman grafiikkaa. DoubleBufferedLinePanel on muokattu versio, .NET ympäristön Panel-oliosta. Muokattua versiota ohjelmassa käytetään kaavioiden pohjana, johon piirretään erivärisiä viivoja kuvaamaan OMD-laitteen tilaa ja sähköliitosten sähköistystä. Alkuperäinen Panel ei kelpaa siitä syystä, että alkuperäiselle Panel-oliolle ei saa asetettua Double buffered-piirtotilaa. Tämän johdosta paneeliin piirretty viiva välkkyy piirtotilanteessa ikävästi, haitaten oleellisesti animaation näkyvyyttä. Ongelmasta päästään eroon tekemällä väkisin uusi Panel-olio, jonka piirtotilaksi asetetaan Double buffered. GreyscaleBitMapConverter taas, nimensä mukaisesti, muuttaa sisään syötetyn kuvan mustavalkoiseksi. Tätä luokkaa tarvitaan, jos käyttäjä vaihtaa ohjelman kaavion kuvia. Käyttäjän antamasta yhdestä kuvasta tulee käyttöön näin kaksi kuvaa, eikä käyttäjän tarvitse etsiä käyttötarpeisiinsa sopivaa kuvaparia. Mustavalkoista kuvaa käytetään, jos kyseinen laite tai sähkölähde on sähkötön, ja alkuperäistä kuvaa käytetään kun kyseinen laite tai sähkölähde on sähköistetyssä tilassa.

Languages-moduuli sisältää luokat ohjelman kielille. Kielien muuttujat ja niiden arvot sisältyvät luokkiin, joten tarvittaessa ohjelman kielivalikoimaa voidaan lisätä suhteellisen helposti. Kielen lisäys tapahtuu luomalla uusi, samat muuttujat sisältävä luokka, mutta jonka muuttujiin tallennetaan uutta kieltä vastaava informaatio. Tämän lisäksi on tehtävä menu-valikkoon valintamahdollisuus kyseiselle kielelle. Ohjelman luokka- sekä moduulijako on esitetty kuvassa 12.



Kuva 12. Projektin luokkajako sekä moduulijako.

3.3 Aliohjelmien toteutus

Ohjelman kehityksessä koodia pyrittiin jäsentämään toimintojen mukaan aliohjelmiin. Aliohjelmilla koodin rakennetta voitiin yksinkertaistaa ja selventää. Vaikka ohjelmakoodia jaettiin luokkiin ja erilaisiin aliohjelmiin, projektissa on yksi luokka jossa selkeästi on eniten koodia, formi MainForm.

3.3.1 Modbus-kommunikointi

Modbus-kysely luodaan ModbusConnector-luokassa, jossa on yhteyden luontiin ja ylläpitoon liittyviä aliohjelmia. Kun käyttäjä valitsee käytettäväksi oikean laitteen tilan ohjelmassa, käynnistyy prosessi, jossa selvitetään käyttöjärjestelmältä tietokoneesta löytyvät COM-tietoliikenneportit. Jokaiseen näistä käytettävissä olevista porteista lähetetään yksi Modbus-viesti, johon odotetaan vastausta. Jos vastaus saadaan, on toimiva portti löytynyt ja tätä porttia käytetään siitä hetkestä alkaen Modbus-kommunikointiin.

Itse Modbus-viesti, jossa kysytään tietyt rekisterit tietyltä Modbus-laitteelta, luodaan seuraavalla aliohjelmalla GenerateModbusQuery:

```
private void GenerateModbusQuery
(ushort registers,ref byte[] message)
{
   // Address of the device is 1
      byte address = 1;

   // Starting register is always 2000
      ushort start = 2000;

   // Array to receive CRC bytes
      byte[] CRC = new byte[2];

   // Modbus query is created here. message[1] is funct. code
      message[0] = address;
      message[1] = 3;
      message[2] = (byte)(start >> 8);
      message[3] = (byte)start;
      message[4] = (byte)(registers >> 8);
      message[5] = (byte)registers;

   // After query created, calculate it's CRC value
      CalculateCRCvalues(message, ref CRC);
      message[message.Length - 2] = CRC[0];
      message[message.Length - 1] = CRC[1];
}
```

Koska OMD Explorer-ohjelman tarkoitus on olla yhteydessä ainoastaan yhteen Modbus-laitteeseen ja tiedot luetaan ennalta määritetyistä rekistereistä, on nämä alkuehdot määritetty jo koodissa kiinteiksi.

Modbus-kysely osoitetaan orjalaitteelle, jonka ID-tunnus on 1 ja käyttäen toiminnon koodia numero 3, joka tarkoittaa rekisterien lukua /7/. Rekistereitä luetaan laitteesta kahdeksan kappaletta, alkaen rekisteristä 2000 /3/. Näistä tiedoista luodaan Modbus-viesti, johon vielä lopuksi lasketaan vaadittu CRC-tarkistesumma käyttäen CalculateCRCValues-aliohjelmaa.

Kysely lähetetään aiemmin käyttöjärjestelmältä saadusta COM-porttien listasta selvitettyyn toimivaan tietoliikenneporttiin käyttäen aliohjelmaa SendQuery.

```
try
{
      // Keep track of sent messages
      sentQuery++;

      // Write message to serial port
```

```
            modbusSerialPort.Write(message, 0, message.Length);

            // Get the response
            GetResponse(ref response);
      }
      catch (Exception)
      {
            // If error received, set communication error to true
            communicationError = true;
            return false;
      }
```

Lähetetyistä viesteistä pidetään lukua, samoin kuin vastaanotetuista viesteistä. Jos nämä lukumäärät eroavat toisistaan, yhteydessä on häiriö ja siitä ilmoitetaan käyttäjälle. Yhteyshäiriön sattuessa odotetaan noin 10 sekuntia yhteyden toipumista. Jos yhteys toipuu tässä ajassa, Modbus-kyselyitä jatketaan normaalisti.

Vastaus lähetettyyn Modbus-kyselyyn tallennetaan aliohjelmalla GetResponse:

```
private void GetResponse(ref byte[] response)
{
   // Get data from serial port
      for (int i = 0; i < response.Length; i++)
      {
        // and save it to response -array
           response[i] = (byte)(modbusSerialPort.ReadByte());
      }
}
```

Sarjaportista saapunut vastaus luetaan tavu kerrallaan tavutaulukkoon kunnes vastaanotetun viestin pituus täyttää taulukon. Vastaanotettava viesti on aina vakiokokoinen riippumatta sen sisällöstä, jos kyselytyyppi pysyy samana. Tämän jälkeen saadulle viestille suoritetaan CRC-tarkistus, jossa varmistetaan vastaanotetun paketin eheys. Vastaanotetun viestin CRC-tarkiste tarkistetaan ja jos se on oikein, viestin purkua jatketaan.

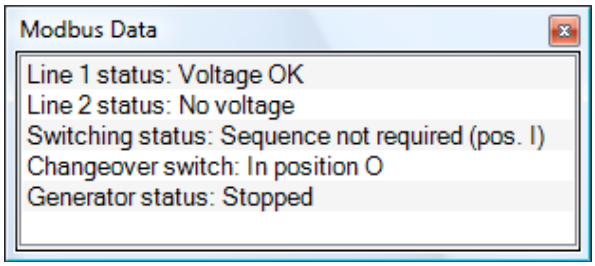
Vastaanotetusta viestistä poistetaan vastauksen lähettäneen laitteen tunnus, tarkistussumma ja muut tiedot, jotka eivät ole itse rekistereiden sisältöä jota tarvitaan. Lopuksi jäljelle jää lähettäneen laitteen rekisterien arvot väliltä 2000-2007. Vastaanotettujen, järkevää dataa sisältävien viestien lukumäärästä pidetään kirjaa muuttujilla, jotta mahdollinen tietoliikennekatkos havaitaan ja yhteys voidaan käynnistää uudelleen.
Modbus-paketin purku suoritetaan seuraavaksi:

```
if (CheckResponse(response))
{
   // Response OK
   // Return requested register values:
   for (int i = 0; i < (response.Length - 5) / 2; i++)
   {
      values[i] = response[2 * i + 3];
      values[i] <<= 8;
      values[i] += response[2 * i + 4];
   }

   // Keep track of successfully received messages
   receivedQuery++;

   // If no error is present then reset the counter
   communicationErrorCounter = 0;
   communicationError = false;
   return true;
}
```

Vastaanotettuja uusia arvoja verrataan edellisen lukukerran arvoihin. Jos juuri luetut arvot ovat muuttuneet edellisistä, päivitetään uudet arvot ohjelmaan käyttäjän nähtäväksi.

3.3.2 Datan esitys sanallisesti

ModbusData-formi sisältää Modbus-kyselyyn vastauksen, eli se sisältää tiedot siitä mitkä arvot OMD-laitteen rekisterin kyselystä saatiin. Nämä rekisterien 2000–2007 arvot sekä niiden sisältävät arvot (esimerkiksi 1), on esitetty formissa sanallisesti, jotta käyttäjän ei tarvitse tietää mitä mikin rekisterin arvo missäkin rekisterissä tarkoitti. Formin alin rivi sisältää tiedon virherekisteristä 2007 ja tämän rivin sisältöä ei näytetä käyttäjälle, jos virhettä ei ole sillä hetkellä päällä eli rekisterissä 2007 arvo on 0. Tällä kuvassa 13 olevalla formilla esitetään käyttäjälle ohjelman tärkein anti yhdessä graafisen kaavion kanssa.



Kuva 13. Modbus-rekisterien 2000-2002 ja 2005-2007, sekä rekisterien sisältämien arvojen sanallinen selitys käyttäjälle (rekisterissä 2007 ei hälytystä).

Vaikka tämän formin arvot saadaankin suoraan Modbus-väylästä, voidaan sen sisään syöttää ”väärennettyä” tietoa. Tästä on se hyöty, että ohjelman käyttö esitystilassa ei estä formin näyttämistä sekä rekisterien sanallista esitystä käyttäjälle, vaan esitystilassa voidaan pakottaa ikkunaan vastaavat arvot kuten ne tulisivat oikeasta OMD-laitteesta.

Aliohjelman kutsu ja halutut rekisteriarvot asetetaan seuraavasti:

```
ModbusDataInjector(0,1,0,0,0,2,2,0);
```

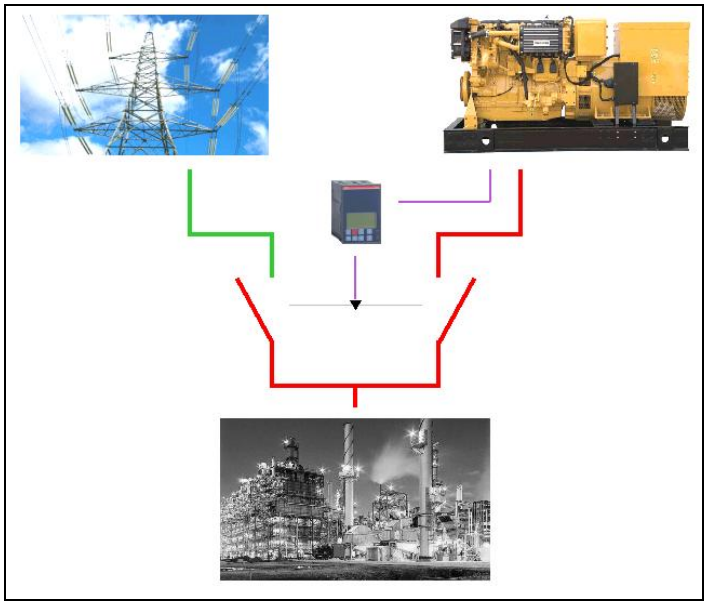
Aliohjelma asettaa ModbusData-formin rekisterien arvoiksi annetut arvot:

```
private void ModbusDataInjector(int reg0, int reg1, int reg2,
int reg3, int reg4, int reg5, int reg6, int reg7)
{
   // This method injects false data to modbus data array
   // forcing the modbus data form to read 'correct' values,
   // meaning the values the user sees on the screen
      modbusData[0] = reg0;    // Register 2000
      modbusData[1] = reg1;    // Register 2001
      modbusData[2] = reg2;    // Register 2002
      modbusData[3] = reg3;    // Register 2003
      modbusData[4] = reg4;    // Register 2004
      modbusData[5] = reg5;    // Register 2005
      modbusData[6] = reg6;    // Register 2006
      modbusData[7] = reg7;    // Register 2007
}
```

OMD800-laitteen välittämien rekisterien nimet ja arvot on tallennettu ohjelmassa erillisiin taulukoihin, joista sanalliset selitykset saadaan esitettyä käyttäjälle valitun kielen mukaisesti.

3.3.3 Datan esitys graafisesti

Ensisijaisen ja toissijaisen sähkölähteen sekä kuorman sähköistys on kuvattu ohjelmassa myös graafisesti (kuva 14). Käyttäjälle näytetään myös generaattoriohjauksen tila, kumpi sähkölähde on kytkettynä kuormaan sekä se mitkä johtimet ovat jännitteellisiä. Nämä tiedot saadaan Modbus-väylältä ja näiden tietojen perusteella kaavio päivitetään vastaamaan oikeaa tilannetta.



Kuva 14. Graafinen esitys laitteiston sähköistyksestä sekä kytkennän tilasta. Ensisijainen verkko (vasen yläkulma) tuottaa sähköä (vihreä johdin), mutta sitä ei ole kytketty kuormaan (alareuna, mustavalkoinen kuva).

Jos ohjelman käyttötilana on animointi, eli ohjelmaa käytetään ilman oikeaa OMD-laitetta, kaavion grafiikka piirretään silti ikään kuin Modbus-väylän tiedoista käyttäen kohdassa 3.3.2 kuvattua ModbusDataInjector-aliohjelmaa.

Viivojen piirto ohjelmallisesti kaavioon osoittautui työlääksi ohjelmoida. Itse laitteen animointitilan 12-vaiheinen animaatio on jatkumo, jossa seuraavan kohdan piirto aloitetaan vasta edellisen valmistuttua. Lisäksi animaatio noudattaa tiettyä järjestystä ja viivat piirretään animaationomaisesti eteneviksi. Tähän viivojen animointiin syntyi koodia viidesosa koko ohjelmasta, noin 950 riviä. Oikean laitteen tilassa viivoille ei piirretä animointia ja linjojen sähköistys (eli piirtoväri)

saadaan suoraan Modbus-kyselyn tiedoista (onko ensisijainen tai toissijainen sähkölähde sähköistetty). Animointia ei piirretä oikean laitteen toimintatilassa siksi, että vallitseva tilanne olisi käyttäjän nähtävissä välittömästi kun muutos sähköverkossa tapahtuu.

Seuraavassa koodissa on voimassa eräs tavallinen tilanne, jossa ensisijainen sähkölähde on kytketty kuormaan ja generaattori ei ole käynnissä. Grafiikan viivojen piirto tapahtuu niiden tietojen perusteella, jotka Modbus-rekistereistä 2000-2007 saadaan. Viivojen piirto vaatii paneelin päivityksen.

Koska Modbus-väylän arvot ovat muuttuneet edellisestä rekisterien lukukerrasta, myös ModbusStatusForm-ikkuna päivitetään vastaamaan uutta tilannetta.

```
if (modbusData[0]==0 && modbusData[1]==1 && modbusData[5]==1
&& modbusData[6] == 2)
{   // Device status 1
    // LINE1:               Voltage OK    AND
    // LINE2:               No voltage    AND
    // Generator:           OFF           AND
    // Changeover Switch:   In position I

    // drawLineParams: parameters used in line draw
    // [0] = is primary network producing power
    drawLineParams[0] = true;

    // [1] = is backup network producing power
    drawLineParams[1] = false;

    // [2] = is primary network SWITCHED
    drawLineParams[2] = true;

    // [3] = is backup network SWITCHED
    drawLineParams[3] = false;

    // [4] = is LOAD on
    drawLineParams[4] = true;

    // Force refresh of the panel
    centerPanel.Refresh();

    // Boolean for determin. is the load ON or OFF. This is
    // used in the modbus status form, in drawing the
    // coloured box next to the text
    systemIsRunning = true;
    modbusStatusForm.Refresh();

    // In this stage, Grid is on and backup is off
    primaryPowerPictureBox.Image = gridOnImage;
    backupPowerPictureBox.Image = backupOffImage;
```

```
        // Read text from stages array to modbusstatusform
        modbusStatusFormText = stages[8];
        modbusStatusFormStage = 8;
        modbusStatusForm.ModbusStatusFormLabel.Text = stages[8];
        systemIsRunning = true;

        // If modbusstatusform is showing, update previous data
        if (modbusStatusFormShowing)
            updateModbusStatusForm();

        // Draw load depending if the power is really on or not
        if (modbusData[0] == 0 && modbusData[5] == 1)
            loadPictureBox.Image = loadOnImage;
        else if (modbusData[1] == 0 && modbusData[5] == 3)
            loadPictureBox.Image = loadOnImage;
        else
            loadPictureBox.Image = loadOffImage;

        // Draw current picture to picturebox and show it
        Image image = Image.FromFile(animationImageDirPath +
        "01_PrimaryUp_SecondaryDown_PowerOn.gif");
        centerPictureBox.Image = image;
    }
```

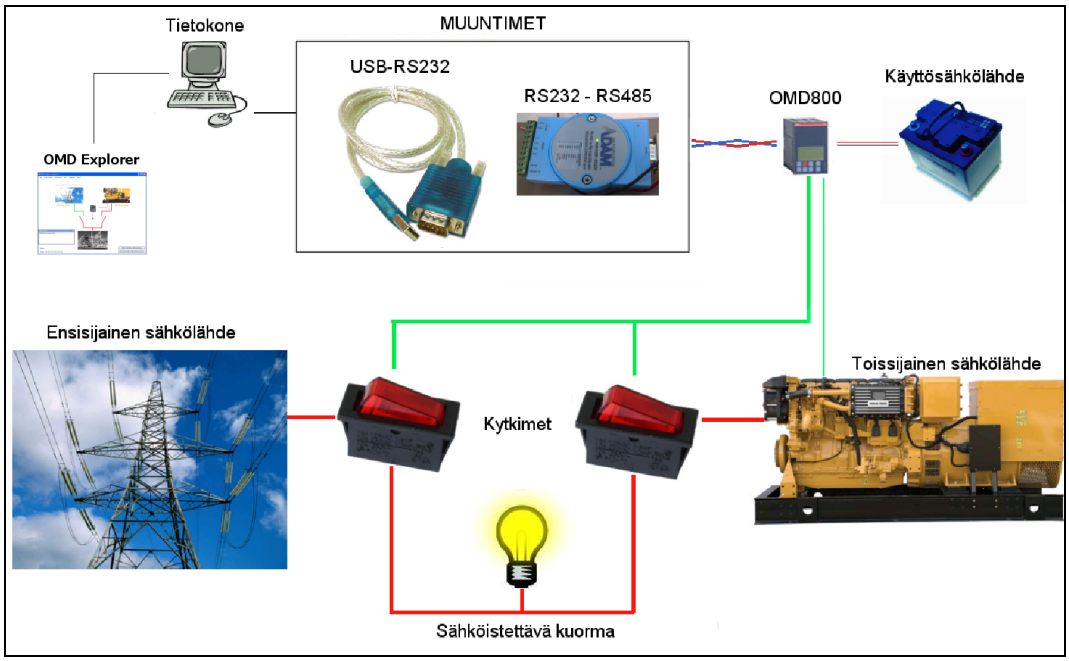
Ohjelman kaaviossa olevat kolme kuvaa (jotka ovat vaihdettavissa) ilmaisevat käyttäjälle kirkkaudellaan tai mustavalkoisuudellaan sen, onko kyseinen laite sähköistetty. Kuvaksi voi käyttäjä halutessaan vaihtaa minkä tahansa värillisen kuvan. Annetusta kuvasta tehdään kopio, joka on mustavalkoinen.

Kuva käsitellään mustavalkoiseksi GreyscaleBitMapConverter-luokalla pikseli kerrallaan, säilyttäen mahdollinen läpinäkyvyys (jos käyttäjä tuo läpinäkyvän gif-kuvan):

```
for (currentPixelHeight = 0;
currentPixelHeight < convertedBitmap.Height;
currentPixelHeight++)
{
    for (currentPixelWidth = 0; currentPixelWidth <
    convertedBitmap.Width; currentPixelWidth++)
    {
        Color color = sourceBitmap.GetPixel
        (currentPixelWidth, currentPixelHeight);

        lumaValue = (int)
        (color.R * 0.3 + color.G * 0.59 + color.B * 0.11);

        convertedBitmap.SetPixel
        (currentPixelWidth, currentPixelHeight, Color.FromArgb
        (color.A, lumaValue, lumaValue, lumaValue));
    }
}
```

3.4 Kommunikointi ja kytkennät

OMD800-laitteeseen saadaan luotua tietoliikenneyhteys kuten kohdassa 1.4 todettiin. Seuraavassa kuvassa 15 on selitetty se miten laitteisto toimii ja miten projektin laitekokonaisuus muodostuu erilaisista laitteista:



Kuva 15. OMD Explorerin kytkentäarkkitehtuuri.

Koska ohjelmisto on suunniteltu yksinkertaiseksi, siihen ei tehty käyttäjän asetettavissa olevia asetuksia sarjaportille vaan nämä asetukset asetettiin ohjelmaan kiinteästi. Vastaavat taulukon 4 asetukset tulisi siis olla myös kytkettävässä OMD800-laitteessa.

Taulukko 4. Ohjelman kiinteät asetukset sarjaportille

Selite	**Arvo**
Slave ID johon kysely osoitetaan	1
Bittinopeus	9600bps
Databitit	8
Pariteetti	Pois
Pysäytysbitti	1
Aikakatkaisu tiedon luvulle	450ms
Aikakatkaisu tiedon lähetykselle	450ms

OMD800-laite analysoi jatkuvasti siihen liitettyjen sähkölähteiden tiloja. Nämä tiedot tallennetaan laitteen sisäisiin rekistereihin, joista nämä tiedot voidaan edelleen lukea Modbus-protokollan avulla. Näin vallitsevat arvot saadaan näytettyä edelleen, esimerkiksi tietokoneella.

Modbus-kommunikoinnissa käytetään erilaisia koodeja kyselyille, joita halutaan orjalaitteelle lähettää. OMD Explorer-ohjelman Modbus-kysely on kuitenkin aina samanlainen, koska samasta orjalaitteesta luetaan joka kyselyssä samojen rekisterien arvot.

Tässä ohjelmassa käytetty koodi usean rekisterin Modbus-kyselylle on 03 (Read Holding Registers) /7/, ja kyselyssä luetaan arvot rekistereistä väliltä 2000-2007 /3/, tosin rekisterien 2003 ja 2004 sisältöön ei kiinnitetä huomiota. Tämän koodin, 03, kyselyssä rekisterin eteen sijoitetaan luku 4, joten ensimmäinen luettava rekisteri on arvoltaan 40001. Siispä rekisteri 2000 on Modbus-kyselyssä rekisteri 42001. Ohjelman Modbus-kysely on esitetty taulukoissa 5 ja 6.

Taulukko 5. Modbus-kysely selväkielisenä.

Kenttä	**Data**
Orjan laiteosoite	1
Toiminnon koodi	3 – Lue rekisterit
Aloitusrekisteri	2000
Rekisterien lukumäärä	8
CRC summa	-

OMD Explorer-ohjelman suorittama Modbus-kysely on aina samanlainen.

Taulukko 6. Ohjelman suorittama Modbus-kysely viestinä.

Kenttä	**Data desimaalina**	**Data heksalukuna**	**Data binäärinä**
Orjan laiteosoite	1	01	00000001
Toiminnon koodi	3	03	00000011
Aloitusrekisteri (ylimmät)	7	07	00000111
Aloitusrekisteri (alimmat)	208	D0	11010000
Rekisterien lukumäärä (ylimmät)	0	00	00000000
Rekisterien lukumäärä (alimmat)	8	08	00001000
CRC summa	68 129	44 81	01000100 10000001

Vastaanotetun viestin sisältö riippuu siitä, mikä tilanne OMD-laitteeseen liitetyssä sähkökytkimessä sillä hetkellä sattuu olemaan. Esimerkkinä taulukossa 7 vastausviesti kyselyyn, kun voimassa on tilanne, jossa ensisijainen sähköverkko on kytketty irti sähkökatkon vuoksi ja generaattori on käynnistetty onnistuneesti.

Taulukko 7. Modbus-kyselyn vastaus, kun generaattori on onnistuneesti kytketty.

Kenttä	**Data desimaalina**	**Data heksalukuna**	**Data binäärinä**
Orjan laiteosoite	1	01	00000001
Toiminnon koodi	3	03	00000011
Datan pituus (rekisterien lkm *2)	16	10	00010000
Data (ylimmät, 42001)	0	00	00000000
Data (alimmat, 42001)	1	01	00000001
Data (ylimmät, 42002)	0	00	00000000
Data (alimmat, 42002)	0	00	00000000
Data (ylimmät, 42003)	0	00	00000000
Data (alimmat, 42003)	2	02	00000010
Data (ylimmät, 42004)	*ei huomioida*		
Data (alimmat, 42004)	*ei huomioida*		
Data (ylimmät, 42005)	*ei huomioida*		
Data (alimmat, 42005)	*ei huomioida*		
Data (ylimmät, 42006)	0	00	00000000
Data (alimmat, 42006)	3	03	00000011
Data (ylimmät, 42007)	0	00	00000000
Data (alimmat, 42007)	1	01	00000001
Data (ylimmät, 42008)	0	00	00000000
Data (alimmat, 42008)	0	00	00000000
CRC summa	21 113	15 71	00010101 01110001

OMD Explorer-ohjelma kertoo sanallisen selityksen rekisterin ja siinä olevan arvon mukaan, joten käyttäjän ei tarvitse tietää mitä jokaisen rekisterin jokainen mahdollinen arvo tarkoittaa.

Seuraavassa taulukossa 8 on jokaisen rekisterin jokainen mahdollinen paluuarvo.

Taulukko 8. OMD800-laitteen rekisterit ja niiden mahdolliset arvot. /3/

Selite	**Rekisterin osoite**	**Arvot**
LN1 – ensisijainen sähkölähde	2000	0 = Jännite OK 1 = Ei jännitettä 2 = Alijännite 3 = Ylijännite 4 = Vaihe puuttuu 5 = Epäsymmetria 6 = ? 7 = Taajuus ei sallituissa rajoissa
LN2 – toissijainen sähkölähde	2001	0 = Jännite OK 1 = Ei jännitettä 2 = Alijännite 3 = Ylijännite 4 = Vaihe puuttuu 5 = Epäsymmetria 6 = ? 7 = Taajuus ei sallituissa rajoissa
Kytkennän tila	2002	0 = Kytkentäsekvenssiä ei tarvita (ensisijainen linja käytössä) 1 = Sekvenssi kesken (ensisijaisesta toissijaiseen) 2 = Sekvenssi suoritettu (toissijainen linja käytössä) 3 = Käänteinen sekvenssi kesken (toissijaisesta ensisijaiseen) 4 = Sekvenssihäiriö
Automaattisen vaihtokytkimen tila	2005	1 = tila I 2 = tila O 3 = tila II
Generaattorin tila	2006	1 = Päällä 2 = Pois päältä 3 = Virhe
Hälytysrekisteri	2007	0 = Ei virhettä 1 = Kytkin jumissa (kohta I) 2 = Kytkin jumissa (kohta II) 3 = Toissijaisen lähdön ohjaushäiriö 8 = Kytkentä vaiheelle I, häiriö 16 = Kytkentä vaiheelle II, häiriö 32 = Toissijaista lähtöä ei voitu kytkeä 64 = Generaattori ei käynnisty 128 = Generaattori pysähtyy ilman pysäytyskäskyä 256 = Kytkimen kahva asennettu 512 = Ulkoinen virhe 4096 = Generaattorihäiriö 8192 = Sähkölähteet puuttuvat

Kun käyttäjä valitsee ohjelman toimintatilaksi oikean OMD-laitteen tilan, käynnistyy yhteydenluontiprosessi. Tämä prosessi on monivaiheinen. Ensin OMD Explorer kysyy käyttöjärjestelmältä käytettävissä olevat COM-tietoliikenneportit, minkä jälkeen portit käydään läpi yksitellen lähettämällä kyseiseen porttiin yksi Modbus-kysely. Jos kyselyyn saadaan vastaus, on toimiva portti löytynyt ja Modbus-kyselyaliohjelma käynnistetään. Tämä aliohjelma kysyy uudet arvot OMD-laitteesta 500 millisekunnin välein.

Jos OMD-laitteelta saadaan samat arvot rekistereistä 2000–2002 ja 2005–2007 kuin edellisellä lukukerralla, ei samoja arvoja piirretä ohjelmaan uudelleen, eikä Modbus Data-ikkunan sisältöä muokata mitenkään. Tällä estetään ohjelman turhaa kuormitusta sekä Modbus Data-ikkunan välkkymistä piirtotiheydestä johtuen. Modbus Data-ikkunassa on kuitenkin laskuri, jolla käyttäjä näkee lähetettyjen ja vastaanotettujen viestien lukumäärän eli sen, onko tietoliikenneyhteys yhä olemassa laitteiden välillä vaikka ruudun päivitystä ei tapahdukaan.

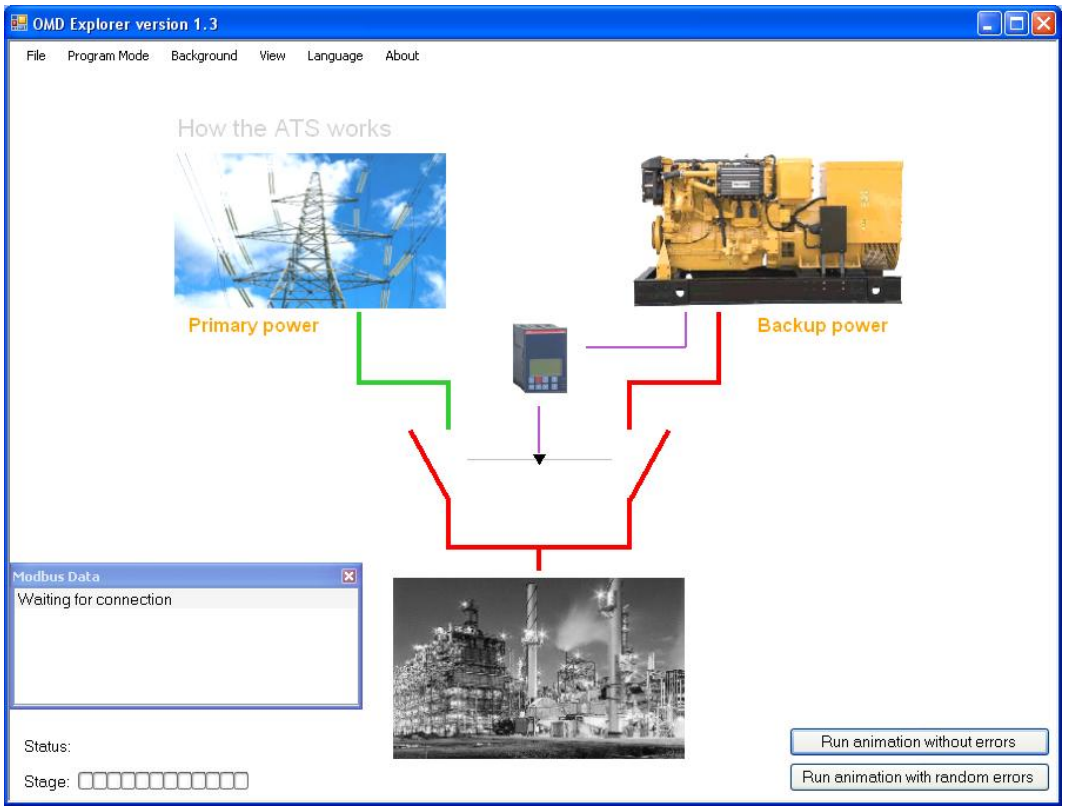
Tietoliikenneyhteys OMD Explorer-ohjelman ja OMD-laitteen välillä on automaattinen, eikä se vaadi käyttäjältä mitään toimia uusien arvojen päivittämiseen ohjelmaan.

4 TOIMINTOJEN KUVAUS

Ohjelman yleisilme on selkeä, mutta havainnollinen. Ohjelma koostuu kolmesta kuvasta, joiden keskelle jää kaavio. Kaavio kuvaa sähköjohtimien jännitteitä, sähköverkon kytkimen tiloja sekä OMD-laitteelta lähteviä käskyjä generaattorille ja sitä, millä perusteilla sähköverkon kytkintä käytetään. Ohjelman perusnäkymä on esitetty kuvassa 16.

Ympärillä olevat kolme kuvaa kuvaavat ensisijaisen sekä toissijaisen sähköverkon tiloja sekä kuormaan kytketyn laitteiston sähköistystä. Ohjelman toimintatilasta riippuen ohjelman pääikkunassa voi esiintyä muitakin komponentteja.

Ohjelman yläreunassa on valikko josta ohjelman tilaa, kieltä, taustakuvaa ja näkyviä elementtejä voi vaihtaa ja ohjelman sulkea. Ohjelman kieli on oletuksena englanti.

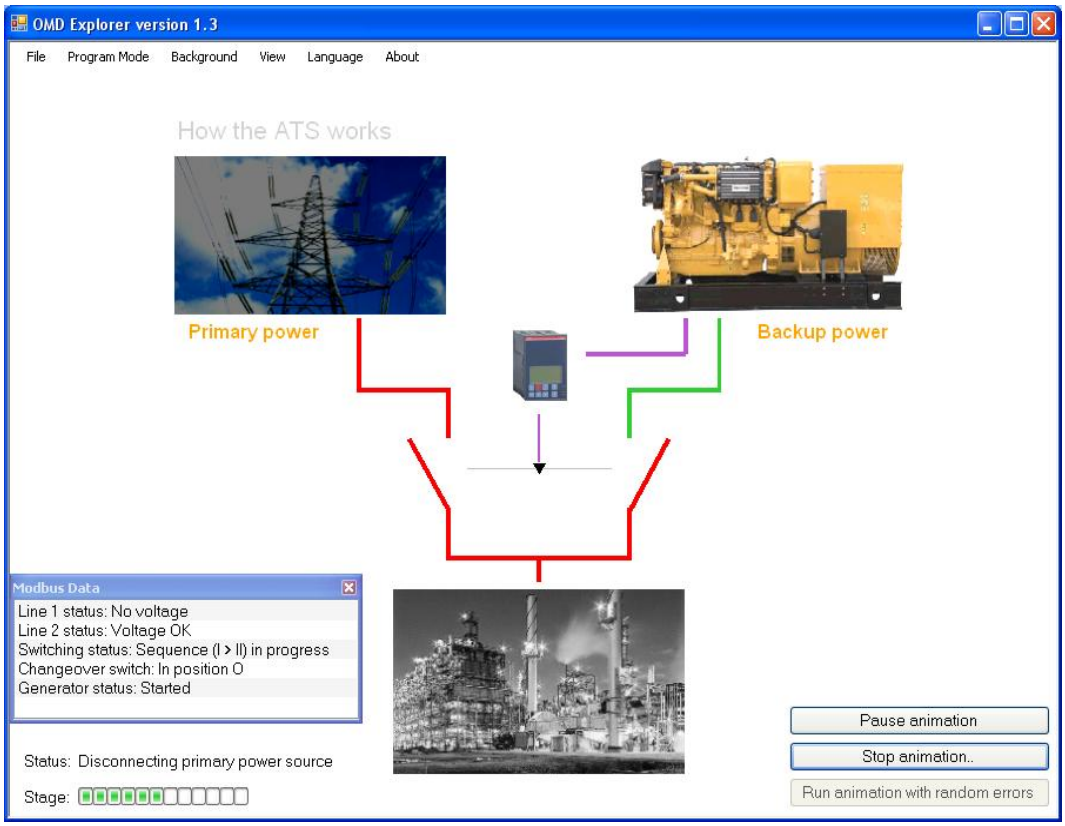


Kuva 16. OMD Explorer-ohjelman perusnäkymä. Kaaviopohjainen esitystapa värillisine viivoineen selkeyttää ohjelman välittämää informaatiota.

4.1 Toimintojen esitys -käyttötila (Use for Demonstration)

Kun sovellus käynnistetään, se on automaattisesti toimintojen esitystilassa, jolloin sitä voidaan käyttää itsenäisesti ilman OMD-laitetta. Tässä tilassa ohjelmalla voidaan esittää oikean OMD-laitteen tiloja ja toimintaa erilaisissa tilanteissa.

Ohjelman grafiikka koostuu sähkökaaviosta (kuvassa punainen / vihreä), OMD-laitteesta (keskellä ohjelmaa) sekä kytkimestä, jota OMD-laite ohjaa. OMD-laite ohjaa myös toisen sähkölähteen generaattoria (paksu violetti viiva). Lisäksi ohjelmassa on kolme kuvaa, jotka kuvaavat ensisijaisen sähköverkon sähköistystä (vasen yläreuna), toissijaisen sähköverkon sähköistystä (oikea yläreuna) sekä sähkölähteisiin kytkettyä kuormaa (keskellä alhaalla). Kuvien värit ja kirkkaus vaihtelevat riippuen siitä, onko laite tai kuvan kohde aktiivinen vai ei. Ohjelman yleisnäkymä tässä toimintatilassa on kuvattu kuvassa 17.



Kuva 17. OMD Explorer-ohjelma toimintojen esitystilassa.

Tässä toimintatilassa on mahdollista käynnistää animaatio. Animaation tarkoitus on kertoa käyttäjälleen OMD-laitteen toiminnasta sähkökytkimen ohjauksessa, ja siitä miten sähköverkon vaihtoprosessi etenee OMD-laitteen ohjaamana. Ohjelman graafisten elementtien tehtävänä animoinnissa on havainnollistaa käyttäjälle seuraavat asiat: tuleeko kuormalle sähköä, kummasta sähkölähteestä sähköä tulee, ja mikä on OMD-laitteen ohjaaman generaattorin tilanne.

Keskellä ikkunaa esitettävä animaatio voidaan käynnistää, pysäyttää hetkeksi ja pysäyttää kokonaan painikkeilla, jotka ovat ohjelman oikeassa alareunassa. Valittavissa on myös animaation esittäminen ilman virhetilanteita sekä satunnaisilla virhetilanteilla. Animaatio ilman virhetilanteita tapahtuu seuraavan taulukon 9 mukaisesti:

Taulukko 9. Animoinnin kierto (ei virhetilanteita).

Animaation kohta	**Kyseisen kohdan kuvaus**
0	Laskurien nollaus, valmistelut
1	OMD-laite kytkeytyy sähkökytkimeen
2	Ensisijainen sähköverkko kytketään kuormaan
3	Ensisijainen sähköverkko tulee sähköttömäksi
4	OMD-laite lähettää käynnistyskäskyn generaattorille
5	Generaattori tuottaa sähköä
6	Ensisijainen sähköverkko kytketään irti
7	Toissijainen sähköverkko (generaattori) kytketään kuormaan
8	Ensisijainen sähköverkko sähköistyy uudelleen
9	Toissijainen sähköverkko kytketään irti
10	Ensisijainen sähköverkko kytketään kuormaan
11	Generaattorille lähetetään sammutuskäsky
12	Generaattori sammuu

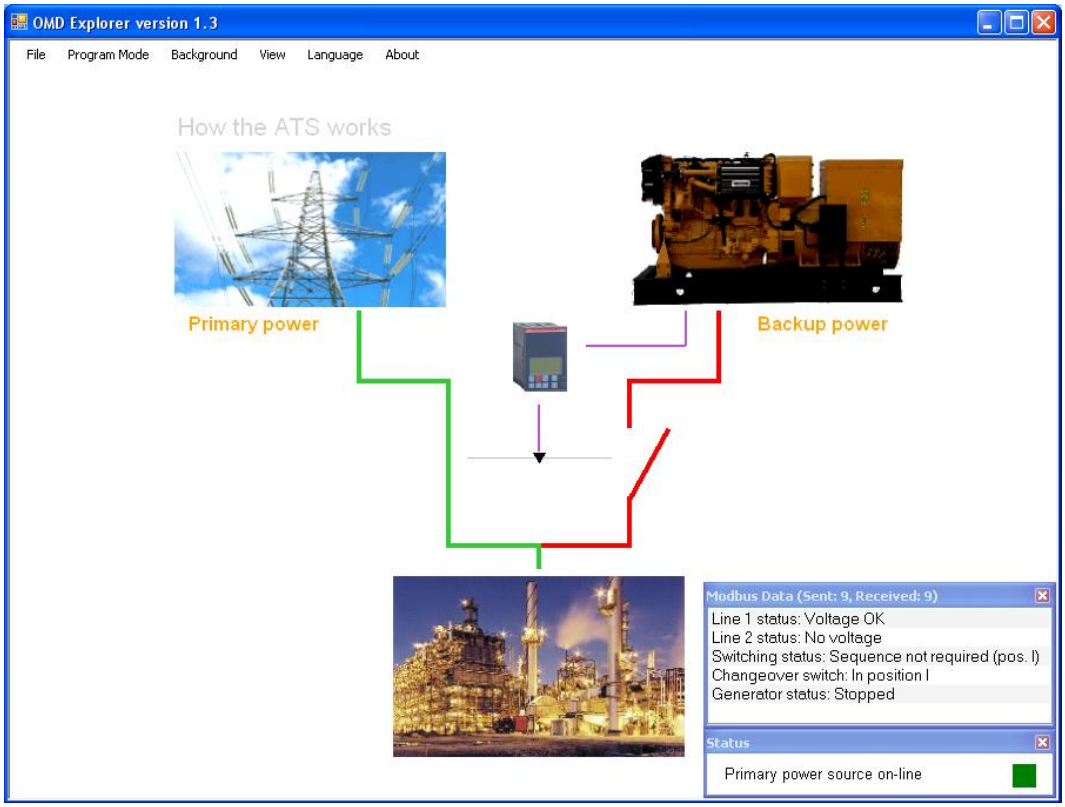
Ohjelman vasemmassa alareunassa on 12-kohtainen palkisto, jonka tarkoitus on kuvata mikä kohta animaatiosta on esitysvuorossa kyseisellä hetkellä. Tämän yläpuolella on verbaalinen selitys siitä mitä animaatiossa esitetään käyttäjälle. Vielä tämän yläpuolella on erikseen suljettavissa oleva Modbus Data-ikkuna, johon tässä esitystilassa luodaan dataa vastaamaan esityksessä olevaa tilannetta.

Koska OMD-laitetta voidaan käyttää erilaisissa kohteissa, joissa kuorma on erilainen ja sähkölähteet voivat olla mitkä tahansa sähkölähteet, käyttäjä voi vaihtaa

havainnollistamisen vuoksi kaikki kolme kuvaa mieleisekseen. Kuvan vaihto tapahtuu napsauttamalla kuvaa hiirellä. Tarvittaessa ohjelman taustakuva ja kieli voidaan valita käyttötarkoituksen mukaan. Kielen ja taustakuvan vaihto tapahtuu ylävalikosta. Versiossa 1.3 kielivaihtoehtoja ovat englanti (oletus) ja saksa.

4.2 Oikea laite -käyttötila (Use with real OMD Device)

Oikean laitteen tila ei eroa graafisesti kovinkaan paljon esitystilasta. Kun käyttäjä valitsee oikean laitteen tilan käyttöönsä ylävalikosta, käyttöliittymästä poistuvat kaikki painikkeet oikeasta alareunasta. Vasemmasta alareunasta katoavat animaation vaihetta kuvaavat elementit. Modbus Data–ikkuna siirtyy pääohjelman oikeaan reunaan ja pääohjelmaan luodaan toinenkin ikkuna, nimeltään Modbus Status. Tähän ikkunaan tulee verbaalinen selitys siitä mikä tila kytkimessä on päällä ja merkkivalo sille tuleeko kuormaan sähköä vai ei. Vertaa kuvaa 17 seuraavaan kuvaan 18:



Kuva 18. OMD Explorer oikean laitteen esitystilassa.

Oikean laitteen esitystila ei vaadi käyttäjältä mitään toimia. Tietoliikenne OMD800-laitteeseen muodostetaan automaattisesti, eikä käyttäjän tarvitse asettaa mitään tietoliikenneasetuksia ohjelmaan. Reaaliaikainen tieto sähkölähteiden tilasta, kuorman sähköistyksestä, sähkökytkimen asennosta ja generaattorin tilasta esitetään ohjelman grafiikalla kuten animaatiotilassakin.

Ainoa käyttäjän näkemä ero animaatiotilaan, kaavion osalta, on sähkökaavion viivojen piirto välittömästi kun OMD-laite havaitsee siihen kytketyissä sähkölähteissä parametrien muutoksen. Kaavion piirto tapahtuu ilman animointia siksi, että tässä tilassa halutaan tuoda käyttäjän tietoisuuteen laitteiston reagointinopeus muuttuviin tilanteisiin, mahdollisimman vähäisellä viiveellä. Tilan muutokset piirretään ohjelmaan korkeintaan puolen sekunnin viiveellä.

5 TESTAUS

Ohjelman testausta suoritettiin kahdella eri tasolla. Projektin edetessä suoritettiin jatkuvaa testausta ohjelmiston kehityksen ohessa sekä projektin loppupuolella keskityttiin ohjelman kehittämisen sijasta testaukseen ja muihin projektin loppuvaiheen töihin.

Testausta suoritettiin ajamalla ohjelmaa ja tekemällä samalla joko OMD-laitteella erilaisia virhetilanteita tai käyttämällä ohjelmaa vastoin sen suunniteltua käyttölogiikkaa. Tämä tarkoittaa lähinnä kokeilemalla kaikkia mahdollisia vaihtoehtoja joita käyttäjän on mahdollista tehdä eri tilanteissa.

Laitteistoon kuului kytkin, jolla normaali ensisijaisen sähköverkon vikatilanne saatiin generoitua. Tällaisessa tapauksessa ensisijaisen sähköverkon sähköntuotto katkesi, jolloin OMD800-laite lähetti toisen sähkölähteen oletetulle generaattorille käynnistyskäskyn. Kun toissijaisesta verkosta saatiin sähköä (eli generaattori oli toiminnassa), siirryttiin syöttämään kuormaan sähköä toissijaisesta sähkölähteestä. Kun kytkintä käännettiin uudelleen, ensisijaisen sähköverkon jännite palasi normaaliksi, jonka jälkeen OMD800-laite kytki kuorman ensisijaiseen verkkoon ja lähetti generaattorille sammutuskäskyn.

5.1 Virhetilanteiden generointi

Mahdollisia virhetilanteita on kahdenlaisia, joko käyttäjän syöte on viallinen (väärää asiaa yritetään suorittaa kun se ei ole sallittua, esimerkiksi ohjelman ajomoodista johtuen) tai OMD-laitteessa on päällä jokin epänormaali tilanne. Virhetilanteita ohjelman kehitys- ja testausvaiheessa suoritettiin OMD-laitteelle ja sen kytkennöille seuraavasti:

Yksi virhetilanteen simulointitavoista oli seuraava: Poistettiin fyysinen ohjausjohto OMD-laitteen ja sähkökytkimen väliltä, jonka jälkeen kytkettiin ensisijainen sähköverkko pois päältä. Tämä virhetilanne edustaa tilannetta, jossa ensisijaiseen sähköverkkoon tulee häiriö, jonka vuoksi toinen sähkölähde täytyisi kytkeä kuormaan kiinni. OMD-laite lähettää toissijaiselle sähkölähteelle (generaattorille) käynnistyskäskyn. Kun toissijainen sähkölähde tuottaa sähköä, vaihto verkkojen

välillä tulisi tapahtua. Koska vaihtoprosessi ei etene, OMD-laite lähettää Modbus-protokollalla tilakyselyyn vastauksen, josta OMD Explorer-ohjelma osaa antaa käyttäjälle oikean virheilmoituksen tilanteesta. Tämä virhetilanne simuloitiin verkonvaihtosekvenssin molemmissa vaihdoissa eli sekä vaihdoissa ensisijainen verkko → toissijainen että toissijainen verkko → ensisijainen.

Toinen virhetilanteen simulointitavoista oli seuraava: kommunikaatio OMD-laitteen ja tietokoneen välillä katkaistiin ottamalla tietoliikennejohto irti tietokoneesta kesken Modbus-kyselyiden. Tämän virhetilanteen seurauksena OMD Explorer-ohjelma antaa ilmoituksen käyttäjälleen siitä, että yhteys on poikki ja että vastausta kyselyyn odotetaan edelleen. Tämän jälkeen tietoliikennejohto joko kytkettiin takaisin paikoilleen, jolloin yhteys muodostuu automaattisesti uudelleen ja kyselyitä jatketaan tai jätettiin irti, jolloin tapahtuu seuraavaa: jos yhteys ei ole kymmenen sekunnin aikana korjautunut, käyttäjä saa siitä vielä erikseen ilmoituksen, jossa kysytään haluaako käyttäjä yrittää yhteyden muodostusta uudelleen vai pysäyttää Modbus-kyselyt kokonaan.

Käyttäjän luomia virhetilanteita, itse OMD Explorer-ohjelmassa, luotiin lähinnä pysäyttämällä ja jatkamalla animaatiota toiminnon esitystilassa sekä vaihtamalla ohjelman kieltä kesken animoinnin. Myös ohjelman taustakuvaa sekä kaavion vaihdettavia kuvia vaihdettiin kesken animoinnin. Näiden toimien lisäksi yritettiin keksiä mahdollisia tilanteita, joissa käyttäjä voisi toimia vastoin ohjelman loogista toimintaa.

5.2 Testauksen vaiheet

Ohjelman testausta suoritettiin käytännössä koko ajan rinnakkain ohjelman kehittämisen kanssa. Viikoittain, perjantaisin, pidetyt kokoukset asiakkaan kanssa paljastivat mahdollisia vikoja ja virhetilanteita, kun ohjelman toimintaa käytiin yhdessä ABB Pienjännitekojeiden edustajien kanssa läpi.

Kun huhtikuun lopun sovittu valmistumisajankohta lähestyi, ohjelman kehitys lopetettiin vähitellen ja painotusta siirrettiin enemmän ohjelman testaukselle. Testausta suoritettiin tekemällä oikean laitteen toimintatilassa erilaisia tilanteita

OMD-laitteella, sisältäen tahallisesti luotuja virhetilanteita laitteen sähköverkon kytkimeen ja sähkökytkentöihin.

Testausta jatkettiin projektin loppuun saakka lähdekoodin järjestämisen ja kommentoinnin ohessa aina toukokuun alkuun saakka. Silloin ABB:lle luovutettiin ohjelmasta viimeisin versio, versionumeroltaan 1.3.

5.3 Virhetilanteet tietoliikenneyhteydessä

Jos ohjelman käyttötapa valitaan oikean OMD-laitteen tilaksi, voi mahdollisia yhteyden virhetilanteita olla kahdenlaisia, joko OMD-laitteeseen yhdistettyä porttia ei löydy tai toimiva tietoliikenneyhteys katkeaa kesken ohjelman käytön.

Kun oikean laitteen tila käynnistetään, ohjelma kysyy käyttöjärjestelmältä tietokoneesta löytyvät COM-tietoliikenneportit. Tämän jälkeen löydetyt portit käydään läpi yksi kerrallaan ja porttiin lähetetään yksi Modbus-kysely. Jos kyselyyn saadaan järkevä vastaus, on toimiva portti löytynyt. Käyttäjää informoidaan löytyneestä portista ja yhteys käynnistetään. Jos porttia ei löydy, käyttäjälle ilmoitetaan, että sopivia portteja ei löytynyt eikä yhteys käynnisty.

Jos toimiva yhteys katkeaa kesken tietojen luvun OMD-laitteesta, yhteyttä odotetaan 10 sekunnin ajan. Tämä siksi, että oikeassa verkon virhetilanteessa sähköt katoavat hetkeksi myös OMD-laitteesta, ellei sitä ole akkuvarmistettu. Tämä taas tarkoittaa sitä, että OMD-laite ei voi kommunikoida Modbus-väylällä ennen kuin toinen siihen kytketyistä sähkölähteistä on sähköistynyt.

Jos yhteys ei ole toipunut, tuona kymmenen sekunnin aikana, käyttäjälle ilmoitetaan yhteyden katkenneen. Samassa ilmoitusikkunassa on painikkeet, joilla käyttäjä voi yrittää käynnistää yhteyden uudelleen tai lopettaa yhteyden muodostuksen kokonaan. Yhteyden muodostusta voi myöhemmin yrittää uudelleen valitsemalla ylävalikosta uudelleen ohjelman käyttötilaksi oikean OMD-laitteen tilan.

6 TYÖN TULOKSET

Lopullinen versio OMD Explorer–ohjelmistosta valmistui 30. huhtikuuta 2009, hieman sovitusta aikataulusta myöhässä. Tähän myöhästymiseen vaikuttivat mm. ohjelmaan kehitetyt lisäominaisuudet sekä testaus. Ohjelma lähdekoodeineen ja Quick Guide -pikaoppaineen luovutettiin ABB:lle viimeisessä tapaamisessa. Ohjelman lopulliseksi versionumeroksi muodostui 1.3.

Projektin tarkoituksena oli toteuttaa OMD-tuoteperheen toimintoja ja ominaisuuksia kuvaava tietokoneella ajettava ohjelma. Ohjelman tuli olla mahdollisimman yksinkertainen ja siten havainnollinen ohjelma sekä markkinointi- että opetuskäyttöä silmälläpitäen. Valmiissa ohjelmassa oli kaksi toimintatilaa, joilla havainnollistettiin OMD-tuoteperheen laitteiden toimintoja sekä ilman OMD-laitetta että laitteen kanssa. Ohjelman grafiikalla esitettiin käyttäjälle OMD-laitteiden toimintoja ja ominaisuuksia. Lisäksi ohjelmassa oli lisäominaisuuksina mm. animaatiokuvien ja taustakuvan vaihto sekä kielen vaihtomahdollisuus englannista saksaksi.

Ohjelma vaadittuine ominaisuuksineen ja muutamine lisäominaisuuksineen sekä käyttöoppaineen toteutettiin projektin aikana onnistuneesti.

7 YHTEENVETO

Projekti onnistui kireähköstä aikataulusta huolimatta hyvin. Säännöllisesti pidetyt kokoukset asiakkaan kanssa edesauttoivat ohjelman kehitystä ja muutosta vaatineet seikat tulivat erittäin nopeasti ilmi. Tämä edesauttoi tahollaan projektin valmistumista, sillä jatkuva kehityksen seuranta ja asiakkaan kanssa ohjelman läpikäynti pitivät molemmat osapuolet tietoisina projektin etenemisestä. Näin toimien projektin lopussa ei päässyt muodostumaan yllätyksiä, puolin eikä toisin, ohjelman ominaisuuksista.

OMD Explorer-ohjelmaan kehitettiin vaatimusten mukaisina tietoliikenneyhteys OMD800-laitteeseen käyttäen Modbus-protokollaa. Protokollan välittämien tietojen perusteella voitiin senhetkisiä OMD-laitteen tilatietoja esittää ohjelmassa sekä graafisesti että sanallisesti. Ohjelmaan kehitettiin myös animointitila, jolla OMD-laitteen toimintoja voidaan esitellä ilman fyysistä yhteyttä laitteeseen.

Asiakkaan toiveita ohjelman toiminnasta ja ulkoasusta sekä kielestä kuunneltiin ja pyrittiin toteuttamaan aikataulun ja osaamisen puitteissa. Projektin tuloksena syntynyt ohjelma oli molempien osapuolien mielestä käyttötarkoitukseen nähden onnistunut.

LÄHTEET

/1/ ABB Oy, 2009. Modbus RTU Communication [online]. Viitattu 28.8.2009. Saatavilla www-muodossa: <URL:http://www.abb.fi/product/seitp329/50f98f657be2a17dc1257070002b4315.aspx?productLanguage=fi&country=FI&tabKey=7 >.

/2/ Automatic transfer switches [online]. Viitattu 25.8.2009. Saatavilla www-muodossa: <URL: http://library.abb.com/GLOBAL/SCOT/scot209.nsf/VerityDisplay/4F042276BE7C07A5C12575AC003D57CB/$File/1SCC303001B0201.pdf >.

/3/ Automatic transfer switches, OTM_C_D_, Installation and operating instructions, ABB Oy, 1SCC303003M0202, rev. B

/4/ Introduction to MODBUS 2002 [online]. Viitattu 28.8.2009. Saatavilla www-muodossa: <URL:http://www.sena.com/download/tutorial/tech_Modbus_v1r0c0.pdf >.

/5/ Modbus-IDA, 2006. Modbus Application Protocol Specification V1.1b [online]. Viitattu 11.9.2009. Saatavilla www-muodossa: <URL: http://www.modbus.org/docs/Modbus_Application_Protocol_V1_1b.pdf >.

/6/ Modbus Poll-ohjelma [online]. Viitattu 20.11.2009. Saatavilla www-muodossa: <URL:http://www.modbustools.com/>.

/7/ Modbus protocol, specifications and in depth tutorial [online]. Viitattu ohjeista ja taulukoista 14.8.2009. Saatavilla www-muodossa: <URL:http://www.lammertbies.nl/comm/info/modbus.html>.

LIITTEET

LIITE 1 Functional Specification

LIITE 2 OMD Explorer - Quick Guide

VAASAN AMMATTIKORKEAKOULU
Tekniikka ja Liikenne
Tietotekniikan osasto

Tekijä: Harri Hurtola
LUOTTAMUKSELLINEN

Functional Specification

ABB OMD Explorer

Table of Contents:

Version History:

Version	Description	Creator	Date
1.0	Ensimmäinen versio asiakkaan sähköpostien perusteella	Harri Hurtola	19.2.2009
1.1	Language changed to english, new Sequence & Use Case diagrams	Harri Hurtola	22.2.2009
1.2	Corrected typos, sequence and use case diagrams	Harri Hurtola	4.3.2009
1.3	Release version	Harri Hurtola	27.4.2009

1 INTRODUCTION

This project is about creating a demonstrative application for ABB. The application should be able to read information from OMD device through Modbus, using USB cable. This Windows application reads the values from the OMD device and updates them on the graphical user interface. Application will be used in Hannover 20. – 24.4.2009. In future, application can be used in sale demonstrations and in education. Parts of Timo Kankaanpää's Requirements Specification (Vaatimusmäärittely, abb_switch, Kankaanpää Timo, Versio 1.0) has been referenced and quoted in this document.

Automatic transfer switch
Modbus
RS-485
SACE

2 OVERVIEW

2.1 Environment

Windows-application ”OMD Explorer” is connected with USB-cable and necessary converters to RS-485 –bus which the OMD device uses. Application reads OMD device’s status information via Modbus. Application will be designed to run with Microsoft Windows XP with Service Pack 3 installed. In addition, .NET Framework 2.0 or later needs to be installed. Application will be created using C# language and Microsoft Visual Studio 2005 –environment.

2.2 Behavior

Application is needed to be operational within 15 minutes. No installation of the application is required, other than uncompressing the ZIP archive if the program is distributed as ZIP archive. User interface is simple: There are only three buttons on the main program form, and depending on the selected mode of the program, buttons may not exist at all. This does not include buttons on the message boxes which may be displayed during the program run. Connection to the RS-485 deivce is created without input from the user; Program automatically requests COM port table from the operating system and tries to open the communication itself.

There is menu bar on the top of the main window, in which will be following items: File: (Exit), Program mode (Use for demonstration, Use with real OMD device), Background, View (Show Modbus data, Show status, Draw GIF images, Draw Lines), Language (English, Deutsch) and About.

Main screen of the program shows the current status of the OMD device: Power source currently in use, the status of the power sources, and values from the Modbus which are displayed as text messages in textbox. Animated pictures describe visually which power source is currently in use and if the transfer from one power source to another occurs, the transfer with related messages to generator will be animated. Animation is either done by drawing lines within the program or by displaying existing GIF animation images.

2.3 Users and further development

The main functionality for the program is demonstrating the OMD device and its status. The graphical user interface of the application is designed to be as simple as possible, so it would be easy enough to use, even for first timers and those who use the application rarely. Complete application is aimed to be released by 30.4.2009. Complete source code is commented and delivered so future development would be possible.

2.4 Capacity Requirements

The application is designed so that it will not send anything to the OMD device other than the required values of which register to poll. Reading the values from the device is designed to be executed every 500ms (Status of the power grid, status of the generator, and values from the OMD device).

2.5 General files and configuration files

The application itself is designed to be one EXE file only. However additional folder is required, featuring background images and the images for the primary / backup / load images. These can be changed during the program run if needed.

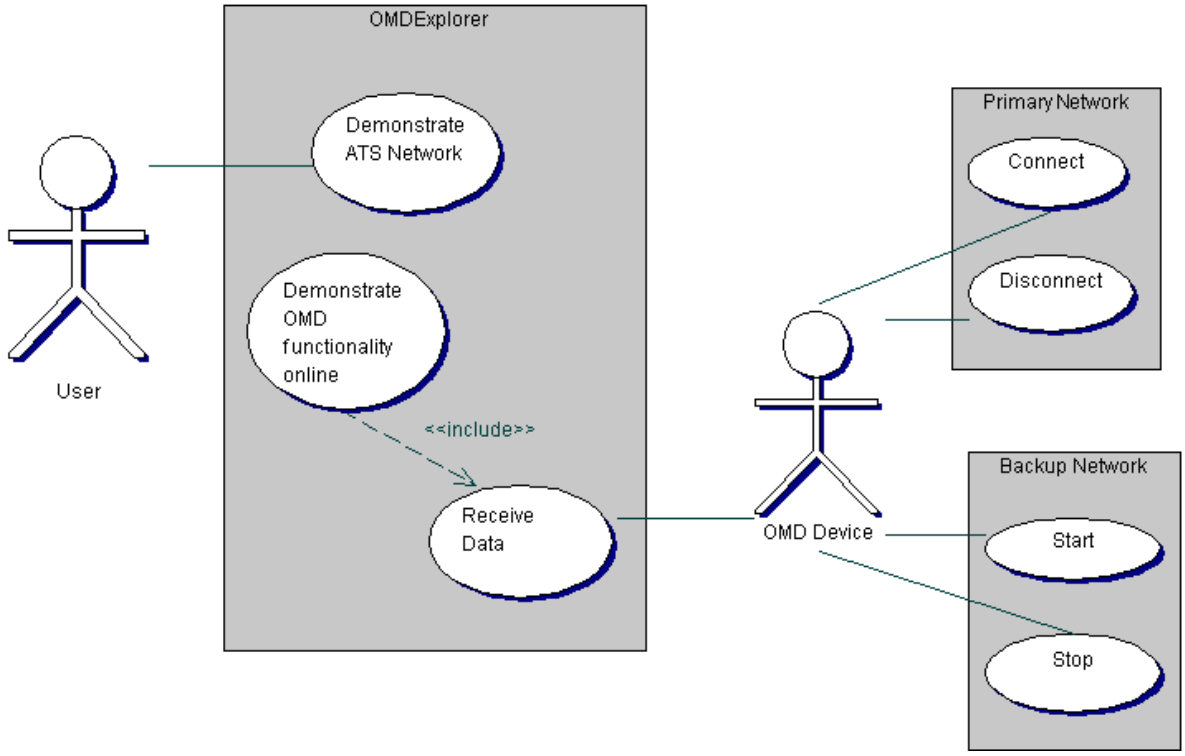
2.6 Quality Function Deployment

Reference	Description	Importance	Accomplished
T1	COM Serial port connection, parity, stop, speed (USB)	1	X
T2	Read OMD device registry via Modbus	1	X
T3	Read OMD device configuration file (XML)	1	-
T4	Show values from device in textbox	1	X
T9	Show device ID	1	-
L1	Connection to device using USB-RS485 cable	1	X
L2	Modbus Analog Devices ADM483	1	?
T5	Show registry values in graphical user interface	2	X
T6	Show alert values with code and in plain text	2	X
T8	Save status information to log file	2	-
T7	Poll alert values with pre-set intervals	3	X
L3	Modbus TCP/IP remote connection	3	-

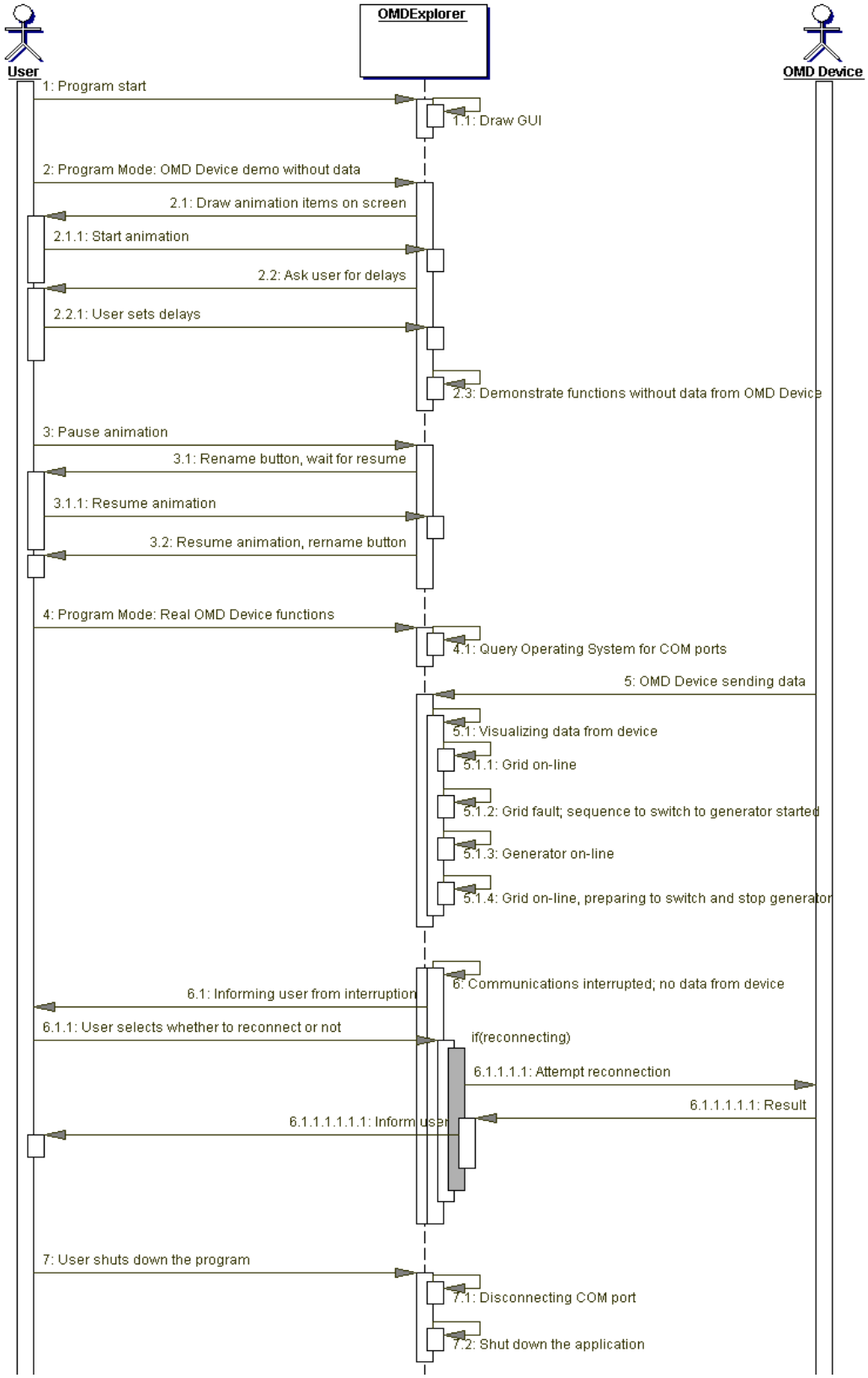
3. ACTION

3.1 Describing the program: Use Case

Because the application is simple as possible, the Use Case –diagram is also quite simple:



3.2 Describing the program: Sequence Diagram



3.3 Usage Intensity

Usage intensity of the program is designed to be demonstrations only. It is not designed to be full time operational. Application is able to monitor only one USB connected OMD device at a time.

4. EXTERNAL CONNECTIONS

4.1 Hardware Connections

Used hardware consists of one PC computer and one ABB OMD device.

4.2 Communications

Communications link between computer and OMD device is created with USB-RS-485 cable. The hardware in the end of the cable is Modbus Analog Devices ADM483.

5. OTHER PROPERTIES

5.1 Performance and response times

Response time when reading data from the OMD device to the application must be below 500ms. The Modbus read is designed so that polling the device occurs every 500ms periods. The polled values are compared to previously polled ones, and if no change has happened, the window is not updated (with the same data again).

5.2 Usability and recovering from errors

User must be able to read the information of what kind of OMD device is in use, in the main screen of the application. After the COM port setup in the application start, input from the user is no required: The data from the OMD device will be shown in the main window of the application. If

communication between the OMD device and the computer is interrupted, the user is informed and asked if restoring the communication is attempted or not.

OMD Explorer – Quick guide

Harri Hurtola

26.4.2009

For version 1.3

Table of Contents:

1. Information

1.1 About this program

This program, OMD Explorer, was made for ABB Low Power Products as demonstration tool for OMD 800 –device. OMD Explorer demonstrates OMD800 –device's functionality, both with real data from device via modbus-RS232 –connection, and without real device's data. This program was made at Vaasa University of Applied Sciences by student Harri Hurtola with supervision and guidance of Dr. Ghodrat Moghadampour during spring 2009.

1.2 Requirements

The following minimum requirements must be met:

- Operating system: Windows XP
- Microsoft .NET Framework 2.0 or later
- Program is designed to be used with resolution 1024 x 768

Additional requirements for full functionality:

- RS-232 –port or
- USB to RS-232 –converter

2. Installation

2.1 Files & Folders

OMD Explorer might be available with two different formats:

- One ZIP compressed file or
- Single Executable file and Resources –folder

If the version you have contains .exe file (OMD Explorer.exe) and Resources – folder, you can run the program by double clicking the .exe file.

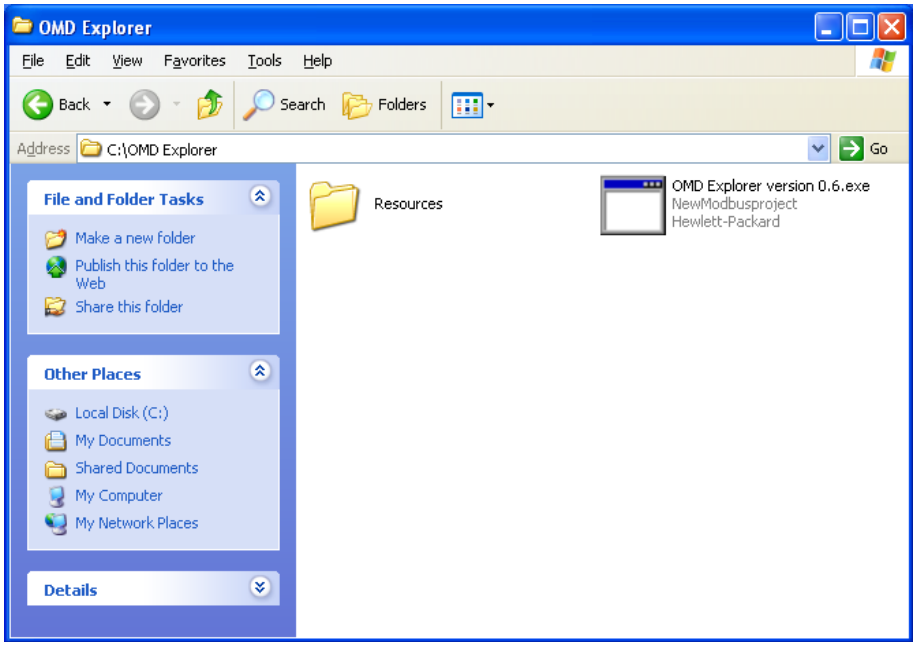


Figure 1: Executable file and image folder called Resources.

Otherwise, if you have just one ZIP compressed file, read next chapter.

2.2 Uncompressing ZIP file

If you have only one ZIP file, uncompress it to continue. Select file by clicking it once, then press mouse right button, and then select “Extract all..” :

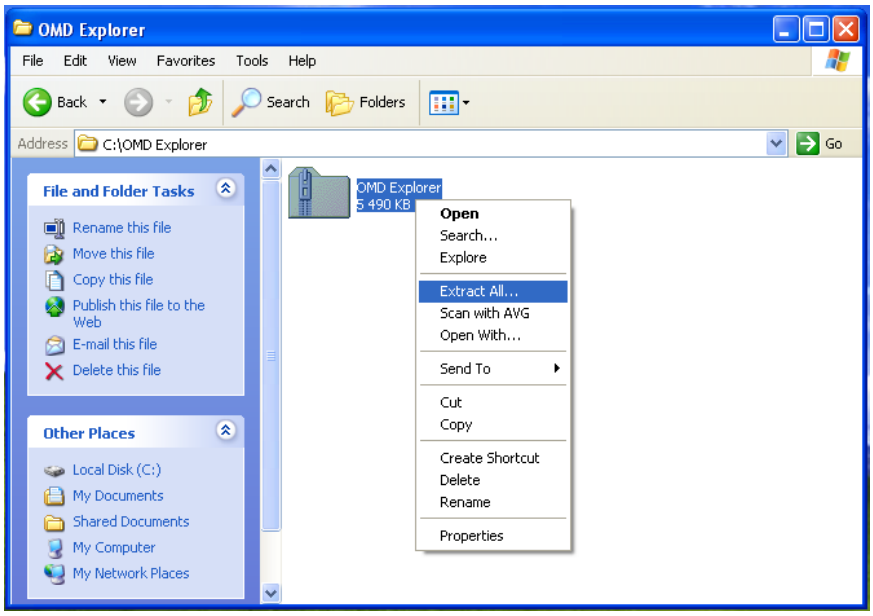


Figure 2: Uncompressing the ZIP archive.

Press ‘Next’ and after that you must choose desired folder to uncompress the ZIP file. When you are ready, press ‘Next’:

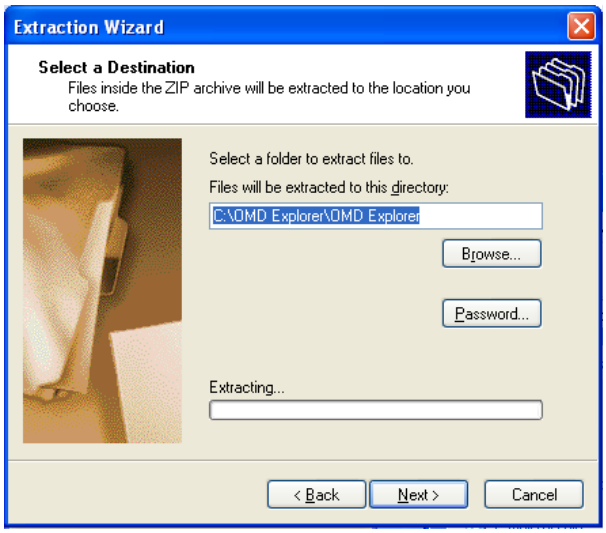


Figure 3: Uncompressing the ZIP archive with Windows XP built in tool.

After you have uncompressed the ZIP file, you can run the program. Go to the folder to which you uncompressed the ZIP file and you should now see the same that in picture 1. To run the program, double click the OMD Explorer.exe –file.

3. Usage

3.1 Menu explained

Menu bar includes different functions which are described here starting from left.

FILE:	
- EXIT	Disconnects RS-232 connections if any and exit the program.
PROGRAM MODE:	
- Use for demonstration	Use demonstration mode. No RS-232 connection to OMD800.
- Use with real device	Use with real data from real OMD800 device through Modbus.
BACKGROUND:	Choose background image to use. Default is white colour.
VIEW:	
- Modbus Data	Opens Modbus data form to view the data transmitted by OMD800 device through Modbus. This form is opened in both modes of the program, but user can close it and reopen from this menu.
- Modbus Status	Opens Modbus status form in real device mode to view the current status of the OMD800 device in short sentence. This form is not visible by default in Use with real device –mode.
- Draw GIF images	User can choose whether the OMD device's surroundings are drawn as GIF images or
- Draw Lines	Draw surrounding lines with the program. This gives few advantages, one of which is the improved visibility on dark backgrounds. This way of drawing is selected by default.
LANGUAGE:	User can change the language of main items of the program. The languages that are currently built in to the program are English (default) and Deutsch. Language selection does not alter the texts in menu bar or any of the message boxes in the program. Language selection alters the texts in the Modbus Data –form and Modbus Status –form. The labels on the main screen changes also.

3.2 Overview

When program is run, user should see the window below. This is the main screen that is first displayed when the program is run. Depending on the program mode described later, some buttons and visual elements disappear from the window.

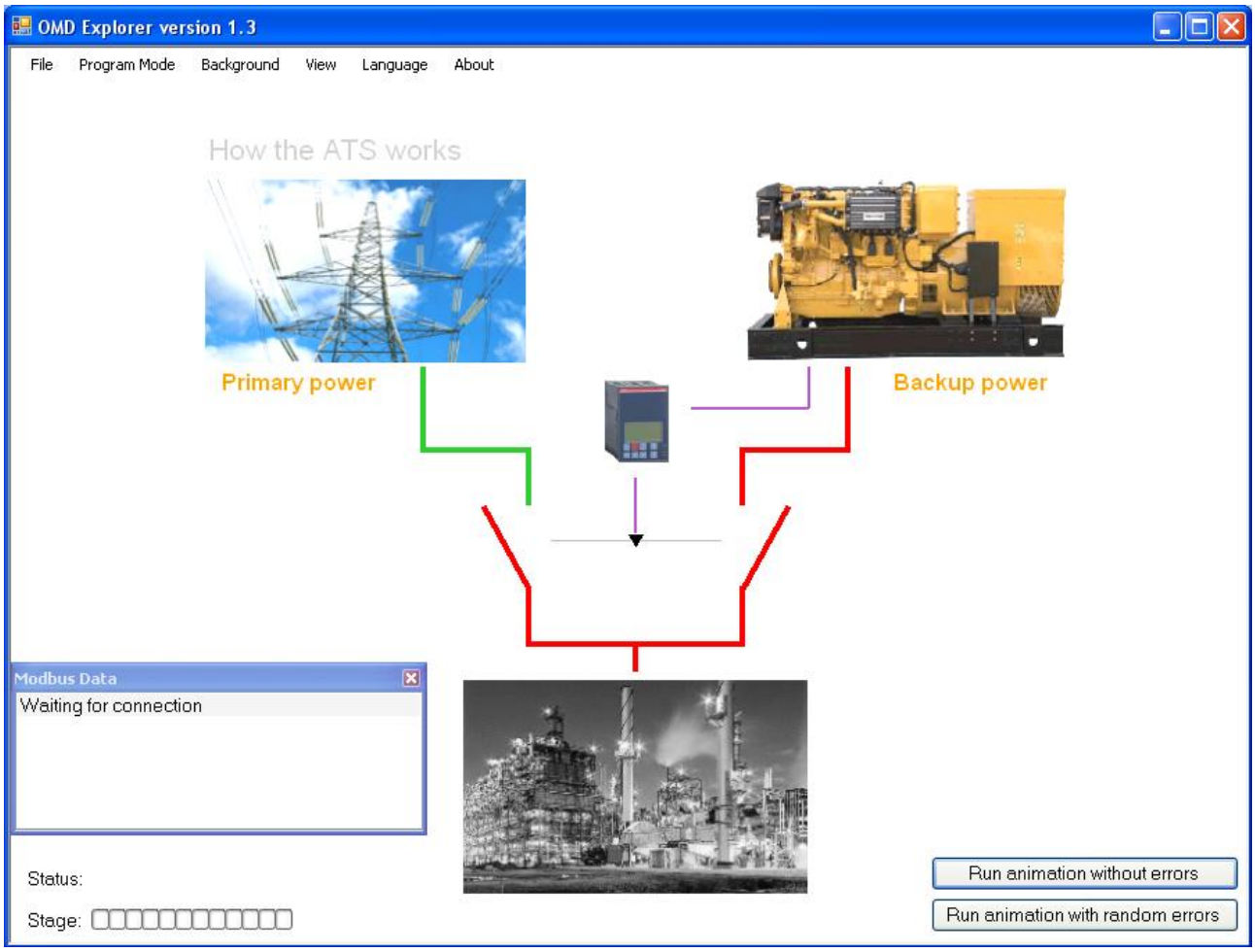


Figure 4: OMD Explorer's appearance when program started.

On top of the screen there is the menu which has different possible selections available again depending on the program mode selected.

On top left corner is the primary power image which displays the status of the main power source for the OMD device's controlled switch.

On the right top corner is the backup power image which displays the status of the backup power source for the switch.

On the bottom of the screen is load image which displays the status of the load connected to the switch.

Centre of the screen displays the status of the switch, and whether the OMD device communicates with the backup power source and the switch.

On the left side of the screen shows Modbus Data –form. This form shows the data and messages delivered by the OMD800 device. Depending on the selected mode of the program, these messages are either real messages from real OMD800 device, or messages that would be seen by the user if the program would be in real device mode with real OMD800 –device. Also the location of this form is changed depending on the program mode currently selected.

3.3 Program modes

Program has two main modes:

- Functionality for demonstration only, and
- Functionality to be used with real OMD device online

The mode for the program can be changed from the menu bar in the top of the application.

3.3.1 Demonstration mode without real device data

By default, the program starts in demonstration only –mode. In this mode, user can run the program without real OMD800 device attached. When this program mode is selected, OMD Explorer looks like the same than in the figure 4 on page 7.

Buttons in right corner controls the demonstration of the OMD device. Demonstration can be run with or without random errors on the stages. When desired demonstration is run, its button changes so that user can stop the animation from the same button it started.

When either animation's button is pressed to start the animation, new form will open to ask the user for pre-set delay times of the OMD device. This illustrates the OMD800 –device's capabilities on

power change situations, and stopping the generator. Selecting these delay values are mandatory to start the animation, although user can select the delays to be 0 seconds if no delays are needed.

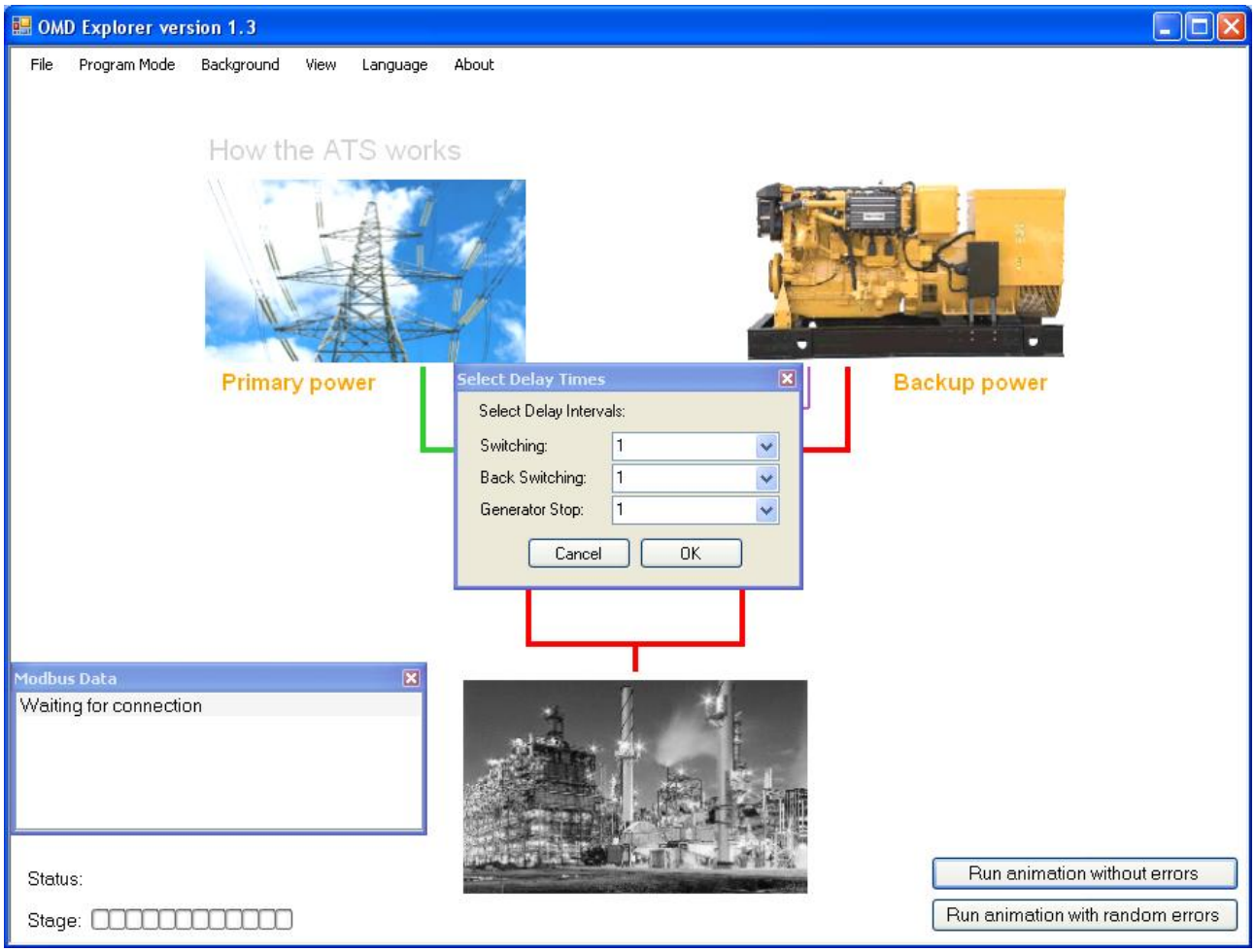


Figure 5: Delay form opened

After selecting the delays from the combo boxes, and after pressing the OK button, selected animation is run.

If pausing the demonstration is needed, there is one button also for that purpose. This button exists only if animation is running. If animation is paused by this button, it can be resumed from the same position by clicking the same button which now has “Resume animation” text on it.

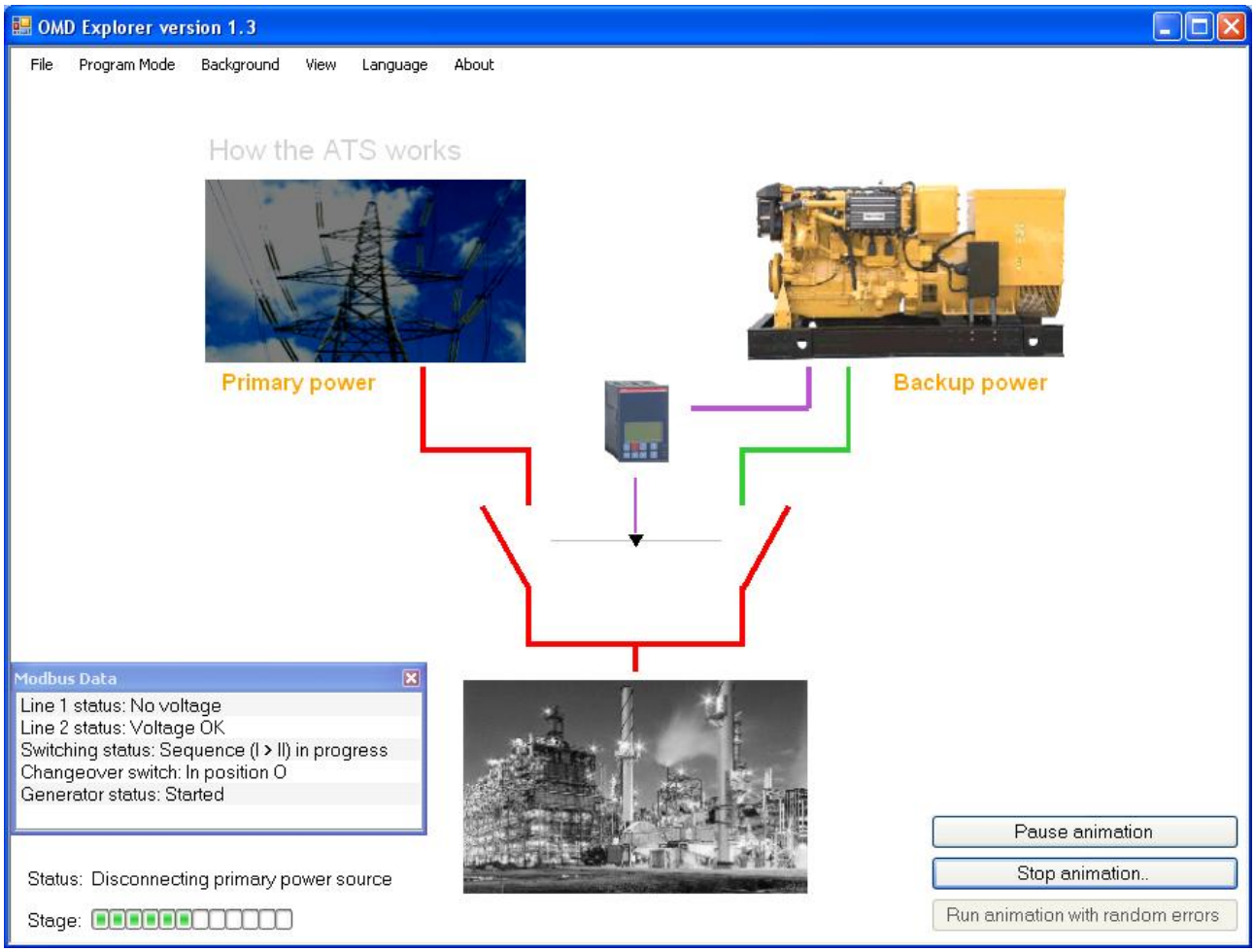


Figure 6: OMD Explorer while running animation of OMD device.

On the left bottom corner is status explained verbally and also what stage is currently being demonstrated. Above this is Modbus Data –form which shows what data would be received from the real OMD800 –device. This data is generated in the program depending on the animation stage.

3.3.2 Demonstration mode with real device data

If user selects ‘Use with real OMD Device’ from the menu, communication to OMD device is attempted automatically through computer’s RS-232 ports, or USB-RS-232 converter. If this connection attempt is successful, user gets information about in which COM port the OMD device is found.

If the attempt fails, user gets information that no OMD device was found. This connection sequence can be initialized again by selecting ‘Use with real OMD Device’ again from the menu.

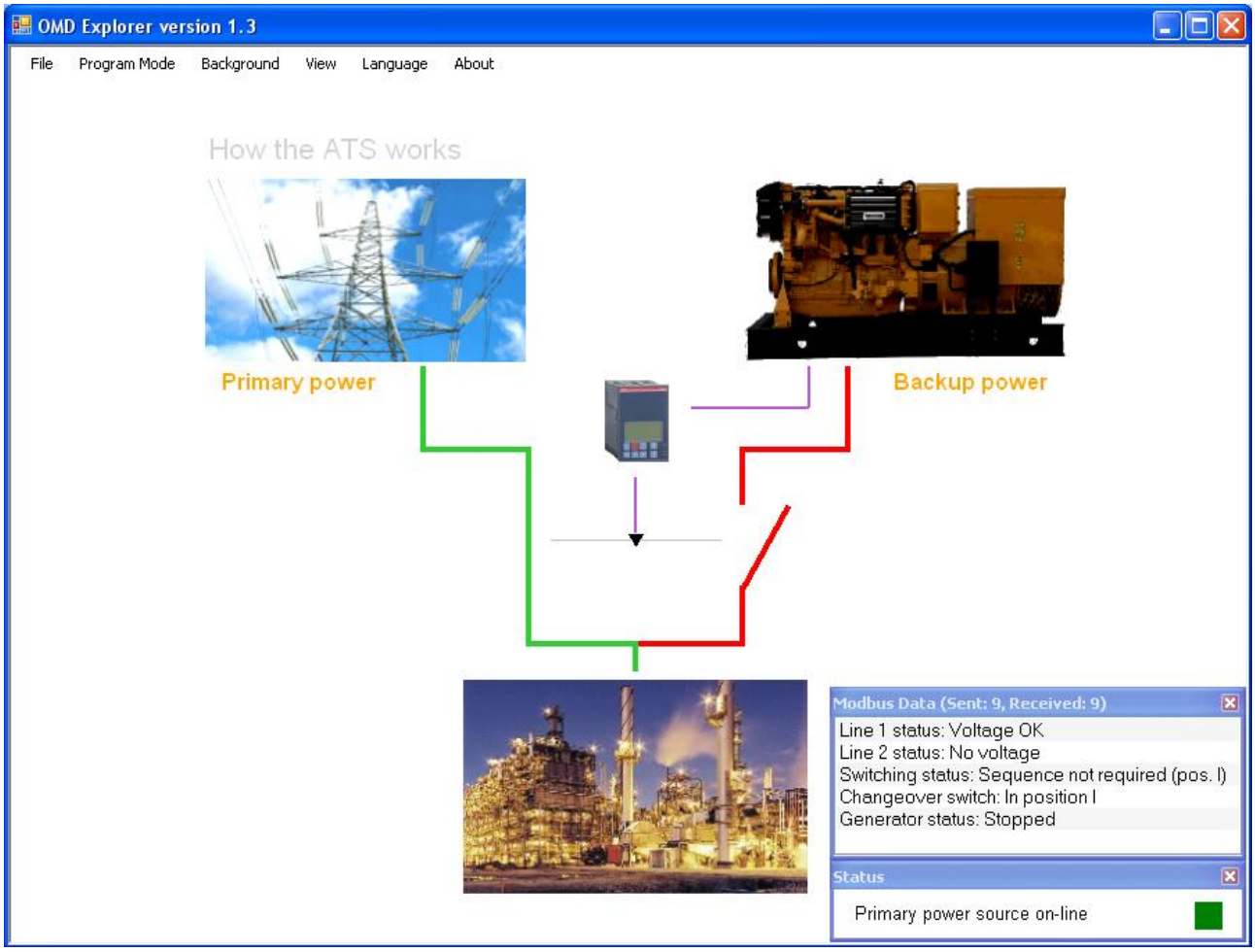


Figure 7: OMD Explorer while communicating with real OMD device.

Program functionality is the same by many aspects as in the demonstration mode. Switch status is displayed on the center of the window, and primary / backup / load –pictures on the same spots as those in without real device data mode. On the bottom right corner, Modbus Data form is displayed and it contains the data that OMD device sends via Modbus – translated to plain English. This form contains a box of rows, total of 6 in which the last one is visible only if OMD device sends alarm. User can close and reopen this form if needed. Reopening the form can be done from menu on the top of the screen.

A form which name is Status, is displayed beneath the Modbus Data form automatically in this program mode. This form shows the status of the OMD device in short sentence. This form can also be closed and reopened if needed. Reopening the form is done from menu also.

In this mode there is no buttons visible. OMD Device's functionality with the power switch is

displayed on the center of the program.

User can change the background as well as the images displaying primary network, backup and load. These changes apply immediately. How to change these pictures is described next in this document.

3.4 Changing primary network / backup / load images

To change the image of primary network / backup / load, click the image you wish to change, once. Following dialog will open:

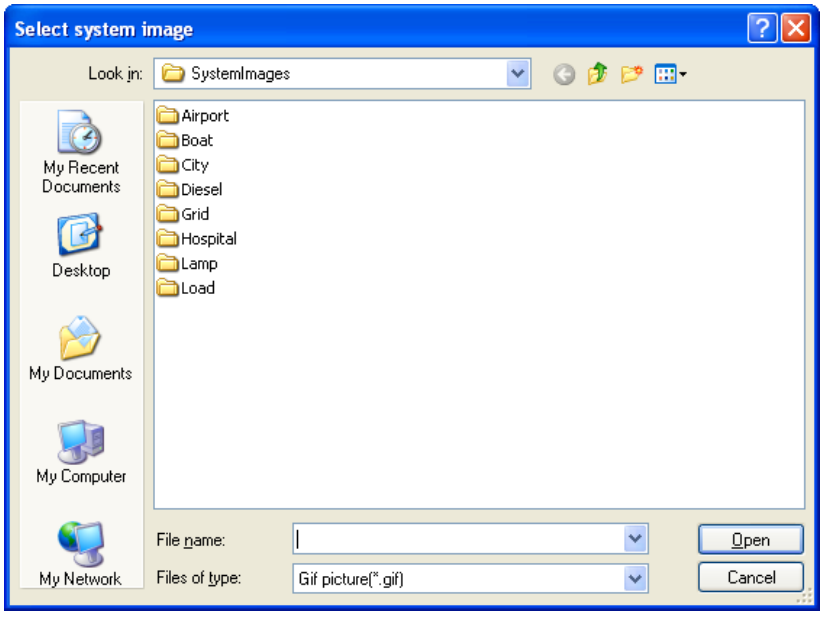


Figure 8: Changing the image of selected picture.

Choose the folder which describes your needs best and open it. From there, choose the image you wish and press OK. Image should be changed now.

3.5 Features explained

OMD Explorer connects automatically to OMD device if a real device function is selected. Communication is attempted on every COM port found in the system, and first one found should be

used. If no suitable COM ports are found from the system, an error message will be displayed. OMD Explorer tries to connect using following settings: Data rate 9600bit/s, No parity, 1 stop bit.

If the connection breaks up while communicating with the device for more than 10 seconds, user is informed and asked whether to attempt reconnection or not to.

Modbus queries take place on every 500ms. Modbus query is attempted on registers 2000-2007 although values from registers 2004-2005 are discarded.

In real device mode, register 2007 should have value 0 which means no error is present in the OMD800 –device. If register 2007 has some error on it, a label will be shown on the left bottom corner of the window with a message of the error. The same error is showed on the Modbus Data form. The program will give a little audible alarm for 10 times to make the alarm known to the user.

4. Adding images to program

4.1 Adding background images to program

Depending where OMD Explorer has been installed, you must first navigate to that folder on your computer. Then go to ‘Resources’ –folder, and there you will see ‘BackgroundImages’ –folder which contains all the background images that can be used with the program.

Few recommendations for images:

Image format:	Jpeg or Gif image
Resolution:	1024 x 768 (Images will be resized to fit the screen)

After adding images to correct folder, they should be selectable in the Background Image change form.

4.2 Adding primary / backup / load images to program

Depending where OMD Explorer has been installed, user must first navigate to that folder on computer. Then user must go to ‘Resources’ –folder, and there will be ‘SystemImages’ –folder. Go there and then choose the folder that describes your needs best. Images within those folders can be selected to any location (Primary – Backup – Load). Best results you get with Gif image which has background colour set to transparent.

User can create new folders if you find it useful.

User needs to add only one image per location to get both two status images: From the image added (and later choose in the program), the program creates second image, black and white coloured as ‘Off-line’ image. This must be kept in mind, when adding new image to program. The image added, should be bright, colourful and describing the device as in ‘On-line’ status.

Few recommendations for images:

Image format:	Jpeg or Gif image (Transparency works only with gifs)
Resolution:	230 x 200 (Images will be resized to fit the screen)

After adding images to correct folder, they should be selectable in the Background Image change form.