Jaakko Reijasalo

# TM500-TESTIOHJELMISTO

# TM500-TESTIOHJELMISTO

Jaakko Reijasalo
TM500-testiohjelmisto
Kevät 2011
Tietotekniikka
Oulun seudun ammattikorkeakoulu

## ALKULAUSE

Tämä työ on tehty Nokia Siemens Networksille Oulussa 11.10.2010 – 6.4.2011 osana Oulun Seudun Ammattikorkeakoulun insinööritutkintoa.

Haluan kiittää Nokia Siemens Networks Oy:n Tommi Santasaarta sekä Jari Parkkista ja työn toimeksiantajia. Haluan myös kiittää ohjaavaa opettajaa Lauri Pirttiahoa tuesta ja ohjauksesta.

Jaakko Reijasalo

OULU 5.5.2011

**TIIVISTELMÄ**

Oulun seudun ammattikorkeakoulu

Tietotekniikka, Mobiiliteknologiat

---

Tekijä: Jaakko Reijasalo

Opinnäytetyön nimi: TM500-testiohjelmisto

Työn ohjaajat: Lauri Pirttiaho (OAMK), Tommi Santasaari (NSN Oy)

Työn valmistumislukukausi ja -vuosi: 2/2011 Sivumäärä: 52+44

---

TIIVISTELMÄ

Testaukseen kuluu merkittävä osa tuotekehitykseen käytetystä ajasta, joten sen tehostaminen voi säästää paljon resursseja. Nokia Siemens Networksilla LTE-matkapuhelinverkkojen testauksessa käytetään TM500 testilaitetta, joka luo useita virtuaalisia käyttäjälaitteita verkkoon ja ohjaa niiden toimintaa. Aiemmin laitetta ohjattiin Linux-koneilla ajamalla yksinkertaisia skriptejä. Tätä testausvaihetta haluttiin kehittää toteuttamalla toiminnot yhtenä työpöytäsovelluksena. Tällä haettiin sekä parempaa käytettävyyttä että mahdollisesti kevyempää toteutusta.

Tässä työssä toteutettu ohjelmistoprojekti seurasi ohjelmistoprosessin tavanomaisia työvaiheita. Sovellus toteutettiin C++ -kielellä Qt-kehitysympäristössäkäyttäen Qt:n kirjastoja. Toteutuksen yhteydessä tutustuttiin myös testilaitteiden ominaisuuksiin ja LTE–verkon toimintaan.

Toteutettu ohjelmisto toimii työn alussa asetettujen vaatimusten mukaisesti, mutta työn aikana kirjattiin myös monia jatkokehitysideoita. Aiemman järjestelmän tarjoamien toimintojen lisäksi ohjelmaan lisättiin muutamia uusia ominaisuuksia, joiden toivotaan helpottavan testien suorittamista.

---

Asiasanat:

testaus, testauslaitteet, ohjelmistokehitys, ohjelmistosuunnittelu

**ABSTRACT**

Oulu University of Applied Sciences

Degree Programme in Information technology, Mobile technologies

---

Author: Jaakko Reijasalo

Title of thesis: TM500 Test Software

Supervisors: Tommi Santasaari(NSN oy), Lauri Pirttiaho(OAMK)

Term and year of completion: Spring 2011 Number of pages: 52+44

---

This bachelor's thesis was commissioned by Nokia Siemens Networks.

Testing consumes a significant part of the time spent in product development,and therefore making it more efficient can save lots of resources. Nokia Siemens Networks develops LTE mobile networks and in one test process they use a TM500 testmobile device, which creates many user devices into the network and controls them.

Previously TM500 was controlled from Linux machines by running simple scripts. The scripts were cumbersome and there was a need to improve this part of the testing by implementing the control in one desktop application. This aimed at better usability and reduction in computer resources needed for the control.

This software development project followed the usual software process stages. The application was impmented in C++ language using the Qt Framework. During the project I also studied the propoerties of the test equipment and the operation of the LTE network.

The implemented software works according to the requirements set at the beginning, but during the work many new ideas were recorded that left plenty of room for further development. During the project some new features were added, which may make the testing more efficient in the future.

---

Keywords: testing, test equipment, software development, software design

# LYHENTEET

| | |
|---|---|
| 3GPP | Third Generation Partnership Project |
| DL | Downlink |
| eNode B | enhanced Node B |
| EPC | Evolved Packet Core |
| FDMA | Frequency Division Multiple Access |
| FTP | File Transfer Protocol |
| IP | Internet Protocol |
| IDE | Integrated Development Environment |
| LTE | Long-Term Evolution |
| OFDM | Orthogonal Frequency Division Multiplexing |
| SC-FDMA | Single Carrier Frequency Multiple Access |
| UE | User Equipment |
| UL | Uplink |

**SISÄLLYS**

# 1 JOHDANTO

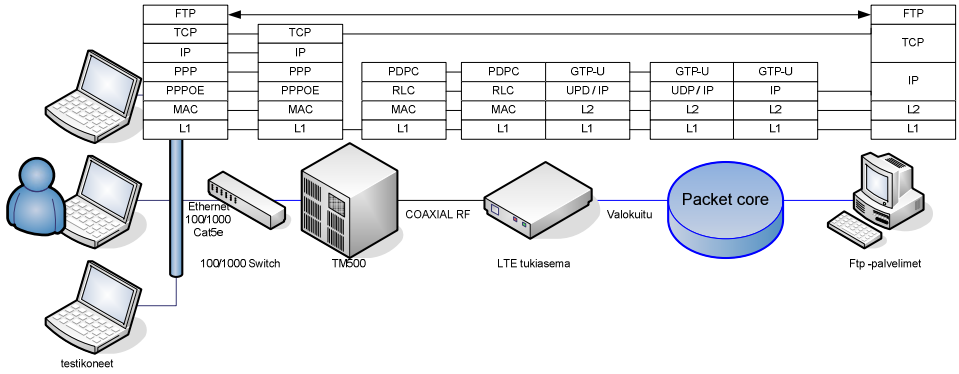
Matkapuhelinverkot kehittyvät nopeasti. Verkoilta vaaditaan enemmän tiedonsiirtonopeutta, pienempiä viiveitä, sekä häiriösietoisuutta. 3GPP-organisaation LTE–standardilla (3GPP, 2010; Long Term Evolution, 2010) pyritään vastaamaan näihin vaatimuksiin. LTE muuttaa koko verkon pakettikytkentäiseksi ja käyttää sille varatun taajuuskaistan tehokkaasti hyväkseen OFDMA- ja SC-FDMA-modulaatioiden avulla. Sillä saavutetaan jopa 170 Mbit/s DL ja 80 Mbit/s UL tiedonsiirtonopeudet.

Opinnäytetyö tehtiin Nokia Siemens Networks Oy:n tuotekehitysosastolle, jossa testataan LTE–tukiasemia sekä pakettiverkkoa eli EPC:tä. Työn aiheena oli ohjelmistoprojekti, jossa testauslaitteiston ohjausohjelmisto korvataan parannetulla versiolla. Toteuttu ohjelmisto muodostaa yhteyden TM500-testilaitteeseen (liite 1) käyttäen PPP–protokollaa. Ohjelmiston päävaatimuksina oli helpottaa tulosten lukemista sekä kasvattaa testikapasiteettia. Päävaatimusten toteutuksen lisäksi kehityksen aikana kerättiin uusia ideoita ohjelman edelleenkehittelyyn.

# 2 MÄÄRITELMÄ

LTE–tukiasema käy läpi pitkän testiketjun, ennen kuin se voidaan lähettää asiakkaalle. Tämä työ laaditaan vaiheeseen, jossa tukiaseman ohjelmiston toimintaa testataan kokonaisuutena muiden laitteiden kanssa. Tukiaseman ohjelmiston toimintaa testataan operatiivisella ja toiminnallisella tasolla. Operatiivinen testaus keskittyy käytettävyyteen sekä dokumentaatioon, toiminnallinen testaus taas tukiaseman suorituskykyyn ja toimintoihin. Työssä laadittu ohjelmisto keskittyy toiminnalliseen testaamiseen. Toimintavarmuutta testataan ajamalla tukiasemaa vähintään sellaisella liikennemäärällä, kuin normaaliolosuhteissa olisi syytä odottaa. Lisäksi ajetaan ylikuormitustestejä, joissa liikennemäärää lisätään reilusti. Tällä testillä tutkitaan ylikuormitusohjauksen toimivuutta sekä toimintavarmuutta.

Testaus suoritetaan käyttäen TM500-testilaitteita, joilla voidaan luoda virtuaalisia asiakaslaitteita verkkoon. Virtuaalilaitteet käyttäytyvät samalla tavalla kuin oikeat laitteet käyttäen samoja protokollapinoja sekä signalointia EPC:n kanssa. TM500 on yhdistettynä tukiaseman radiomoduuliin käyttäen koaksiaalista RF-kaapelia, tukiasemalta on kuituyhteys EPC:hen, josta taas on yhteys FTP–palvelimiin.



Kuva 1: Käyttöympäristö

### 2.1 Lähtökohta - aikaisempi toteutus

Opinnäytetyön idea lähti kesäharjoittelun aikana työksi saadusta testiohjelmiston kehityksestä. Testiohjelmiston alkuperäinen toteutus oli toteutettu komentokehotteessa ajettavilla shell–skripteillä, joita piti ajaa jokaista testiä varten kaksi. Ensimmäinen yhdisti neljä PPP-yhteyttä ja käynnisti ping-prosessit, seuraavalla käynnistettiin FTP–tiedonsiirto joko ylä- tai alakaistaan. Tällä tavalla voitiin ohjata neljää virtuaalilaitetta yhdessä testikoneessa. Tiedonsiirron seuraamiseen käytettiin iptraf–ohjelmaa, joka täytyi käynnistää erikseen. Testauksessa saatettiin vaatia useita kymmeniä virtuaalilaitteita, jonka takia työskentely hidastui sekä koneiden määrä alkoi kasvaa liian suureksi.

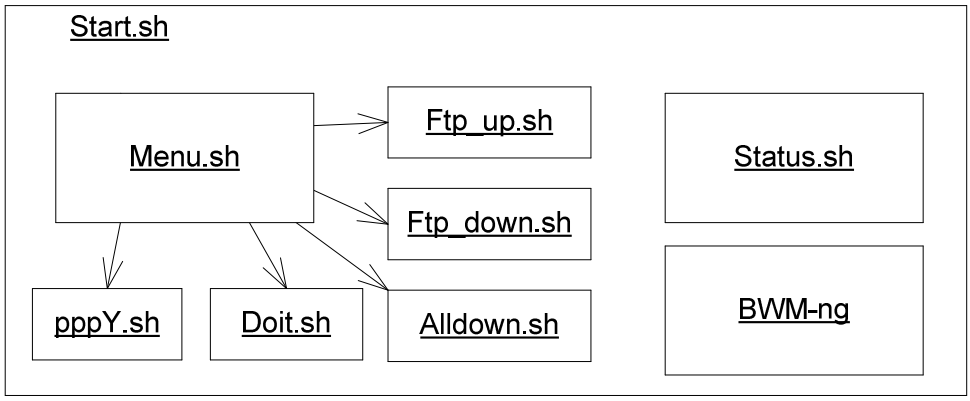
Uudelle ohjelmistolle annettiin vaatimusmääritelmäksi lisätä tuki mahdollisimman monelle yhteydelle samassa koneessa, lisätä ohjelman luettavuutta sekä helpottaa käytettävyyttä. Ohjelmaa lähdettiin laajentamaan vanhojen skriptien pohjalta. Yhteydenmuodostamisessa sekä FTP-siirrossa toimivia skriptejä muokattiin toistorakenteilla tukemaan useampaa yhteyttä. Lisäksi iptrafin korvaajaksi haettiin paremmin tarkoitukseen sopiva BWM-ng, joka näytti yhteyksien siirtonopeudet summattuna sekä erikseen.

Ohjelmalle luotiin skripteillä uusi käyttöliittymä, joka koostui kolmesta osasta. Näitä olivat tilaikkuna, valikko ja aikaisemmin mainittu BWM-ng-ohjelma. Tilaikkunassa näytetään yhteyksien tilat ja vasteajat. Valikon kautta käyttäjä pystyi hallitsemaa yhteyksiä, aloittamaan tiedonsiirrot sekä asettamaan PPP–asetukset.

Ohjelma oli käytettävyydeltään helpompi kuin aikaisempi toteutus, mutta silti hyvin rajoittunut. Esimerkiksi yhteyksiä pystyi käynnistämään neljä, kahdeksan tai 16. Yksittäisen yhteyden hallinta ei ollut mahdollista, ja tiedonsiirrot aloitettiin kaikille yhtä aikaa. Ohjelman lopullinen versio tuki 32 yhteyttä ilman tiedonsiirtoa sekä kahdeksaa yhteyttä siirron kanssa.

### 2.2 Vanhan toteutuksen moduulit ja niiden toiminnat

Ohjelmiston toiminnat jaettiin erillisiin skripteihin helpottamaan niiden kutsumista valikosta. Tässä käydään läpi ohjelmiston osista graafinen sekä sanallinen esitys.



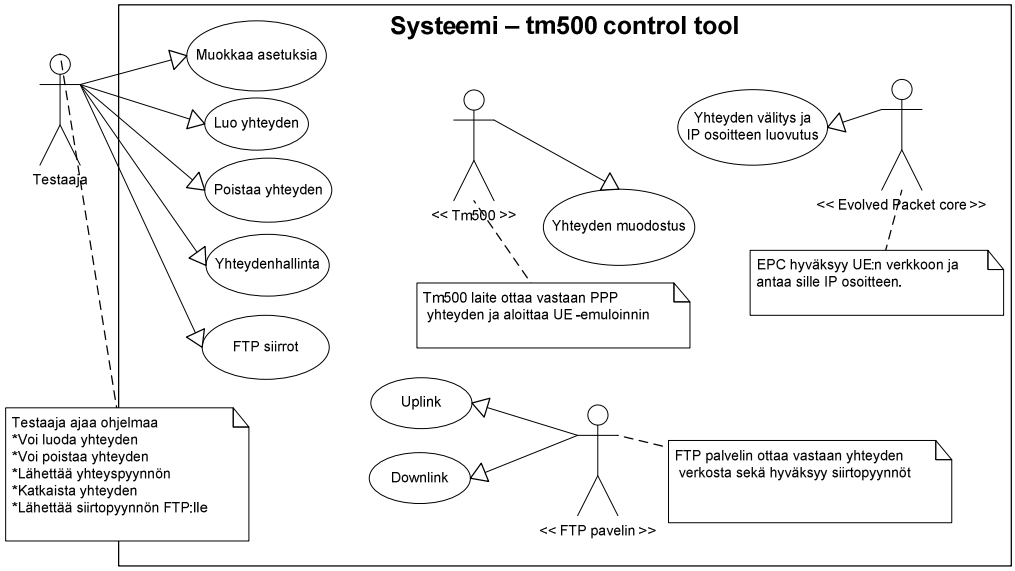

Kuva 2: Vanhan ohjelmiston rakenne

Start.sh käynnistää käyttöliittymän kokonaisuudessaan. Tähän kuuluvat ohjausikkuna, tilaikkuna sekä erillinen BWM-NG-kaistanseurausohjelma. Status.sh eli tilaikkuna tarkistaa muutaman sekunnin välein yhteyksien tilat sekä vasteajat FTP–palvelimille. Menu.sh-ohjausikkunasta käyttäjä pystyy numeroilla 1-9 antamaan komentoja ohjelmalle.

Jokainen komento kutsuu erillistä skriptiä: pppY.sh avaa PPP–yhteyden, jolle luodaan reitti tiettyyn ftp – palvelimeen. Seuraavaksi käynnistyy ping–ohjelma aikaisemmin luodun reitin IP-osoitteeseen ja kirjoittaa tulosteen erilliseen tiedostoon, josta tilaikkuna noutaa vasteajan. Pingin tarkoitus on saada tietoa vasteajasta sekä estää yhteyden katkeaminen, mikäli linjalla ei liiku muuta tietoa. Ftp_up.sh käynnistää FTP–siirron ncftp–ohjelmaa käyttäen. Sama periaate toimi myös ftp_down.sh:lla.

### 2.3 Uuden ohjelman määrittely

Ensimmäinen vaatimusmääritelmä pystyttiin laatimaan suoraan vanhan toteutuksen pohjalta. Vaatimusmääritelmän ja esitutkimuksen pohjalta aloitettiin määrittely, jonka pohjalta luotiin käyttötapauskaavio sekä käyttötapauskuvaukset ohjelmistolle. Kuvassa 3: Käyttötapauskaavio on kuvattu koko systeemi. Tarkemmat käyttötapauskuvaukset ovat liitteessä 2.




Kuva 3: Käyttötapauskaavio

# 3 ESITUTKIMUS, PROJEKTISUUNNITELMA JA PROSESSI

Työssä seurattiin tyypillisiä ohjelmistoprojektin vaiheita. Ensimmäisenä suoritettiin esisuunnittelu, jossa kartoitettiin käyttöympäristö ja sen asettamat reunaehdot sekä ohjelmiston perusvaatimukset. Esitutkimuksen jälkeen luotiin projektisuunnitelma, jossa kartoitettiin aikataulu ja projektissa käytettävät ohjelmistot ja laitteistot sekä muut resurssit. Projektisuunnitelman jälkeen pystyttiin aloittamaan ohjelmistoprosessi.

## 3.1 Esitutkimus

Esitutkimusvaihe aloitettiin ennen varsinaisen työn varmistumista. Sen pohjana toimivat hyvin aikaisemman toteutuksen reunaehdot sekä vaatimukset. Näiden lisäksi käytiin projektin alussa keskustelu siitä, mitä uusia ominaisuuksia ohjelmassa voisi olla sekä miten vanhoja voidaan mahdollisesti parantaa.

Järjestelmän osalta vaatimukset pysyivät samana. Ohjelman tulisi toimia vähintään yhtä hyvin samassa ympäristössä kuin vanhan toteutuksen. Järjestelmätason vaatimukset on käyty läpi kohdassa kappaleessa 4. Esitutkimuksesta luotiin lyhyt dokumentti, joka lähetettiin sekä Nokia Siemens Oy:lle sekä Oulun seudun ammattikorkeakoululle. Esitutkimuksen aikana kerätyt uudet vaatimukset kirjattiin vaatimusmääritelmään, jonka päivittämistä jatkettiin koko ohjelmistoprojektin ajan.

### 3.2 Projektisuunnitelma

Projektisuunnitelmassa päätettiin lopullinen työn tavoite, työaikataulu sekä käytettävät resurssit. Projekti aloitettiin virallisesti 11.10.2010, jolloin sopimus allekirjoitettiin. Suunniteltu aikataulu oli seuraava:

| Vaihe | Aika | Tulokset |
|---|---|---|
| Esitutkimus | 8-11.2010 | Esitutkimus.doc |
| Projektisuunnittelu | 11.10. – 25.11.2010 | Projektisuunnitema.doc |
| Määrittely ja analyysi | 25.11. – 6.4.2011 | Vaatimusmäärittely.doc |
| Suunnittelu | 1.1. – 6.4.2011 | Ohjelmiston kuvaus |
| Toteutus | 1.1. – 6.4.2011 | Ohjelmisto |

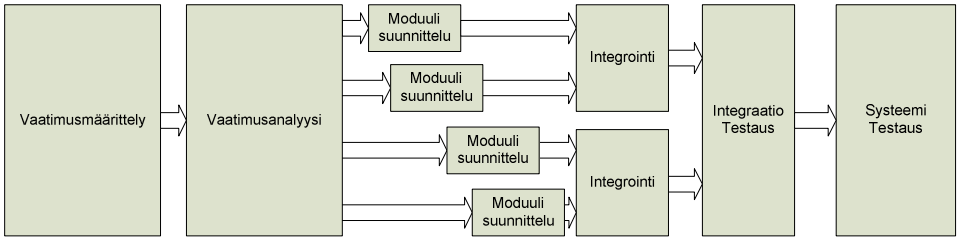
Projektin toteuttamista varten oli käytössä kaksi kannettavaa tietokonetta ja kytkin, jotka lainattiin NSN Oy:ltä. Laitteiden tarkemmat tiedot löytyvät liitteestä 1.

Näiden lisäksi kehityksessä käytettiin hetkellisesti systeemitestauksessa oikeaa ympäristöä hyväksi, jolloin tietokone kytkettiin TM500-laitteeseen. Dokumentoinnissa käytettiin suurimmaksi osaksi Office-tuoteperhettä erillisellä tietokoneella, jossa toimi Windows-käyttöjärjestelmä.

| Ohjelma: | Käyttötarkoitus: |
|---|---|
| Linux, Fedora 12 | Käyttöjärjestelmä jossa ohjelma tulisi toimimaan |
| Qt SDK | Ohjelmointiympäristö |
| PPPd | PPP–yhteyden kontrollointi |
| TOP | Järjestelmänseuranta ohjelma |
| BWM-ng | Kaistanseurantaohjelmisto |
| Doxygen | Koodin dokumentointi |
| vsftp | FTP–palvelin |
| MS office | Dokumentointi |

### 3.3 Ohjelmistoprosessi

Ohjelmistoprosessina käytettiin lisäävää mallia. Ohjelmistosta kehitettiin ensin testiympäristössä (liite 1) toimiva versio, jonka jälkeen sitä testattiin myös oikeassa käyttöympäristössä. Samalla päivitettiin vaatimusmäärittelyä, mikäli uusia ideoita ilmeni. Tämän jälkeen aloitettiin kierros alusta.




kuva 4: Ohjelmistoprosessin kulku.

Työssä seurattiin tyypillisiä ohjelmistotuotannon vaiheita: Ensin määriteltiin ominaisuudet, jotka suunniteltiin ja toteutettiin. Lopuksi ne yhdistettiin. Testaus oli jatkuvaa koko ohjelmointiprosessin ajan.

# 4 VAATIMUSMÄÄRITTELY JA ANALYYSI

Ensimmäinen vaatimusmääritelmä laadittiin vanhan ohjelmiston pohjalta. Sen jälkeen vaatimuksia kerättiin lisää ohjelmiston käyttäjiltä palavereissa.

Rajoittavia vaatimuksia olivat ohjelmiston käyttöjärjestelmä sekä ohjelmiston vaatimat resurssit. Ohjelmisto tultaisiin laatimaan pelkästään Linux-ympäristöön, ja sen tulisi olla siirrettävissä distribuutiosta toiseen. Resurssien käytön osalta ohjelmiston tulisi olla tehokkaampi kuin vanhan toteutuksen. Tavoitteeksi asetettiin vanhan käyttäjämäärän tuplaaminen. Suurimmaksi käyttäjämääräksi asetettiin silti 64. Tämän käyttäjämäärän voisi tulevaisuudessa saavuttaa jatkokehittelyllä tai päivittämällä laitteita.

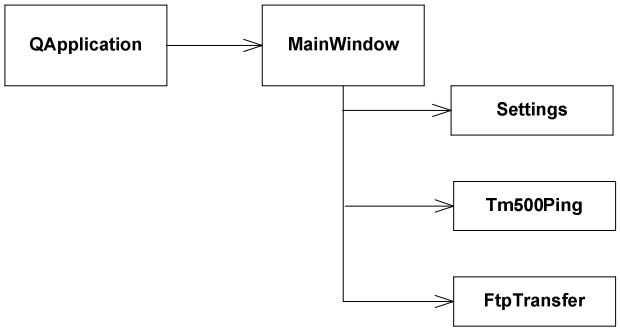
Muut vaatimukset liittyivät käytettävyyteen, kuten käyttöliittymään ja sen toimintoihin. Käyttöliittymään haluttiin näkymään kaikki yhteyden tiedot sekä summattuna kaikkien yhteyksien siirtonopeudet. Asennuksen oli oltava yksikertainen ja kaikki ohjelmiston tarvitsemat asetukset tulisi pystyä asettamaan ohjelman sisältä.

Vaatimuksia kerättiin vaatimusmäärittelydokumenttiin. Dokumentissa ilmoitettiin vaatimukset luokittain, kuten ”Käyttövaatimukset” ja ”rajoittavat vaatimukset”. Vaatimusmäärittelydokumentti löytyy liitteenä.

# 5 SUUNNITTELU JA TOTEUTUS

Määrittelyvaiheen jälkeen aloitettiin moduulisuunnittelu, jossa toteutettiin tarvittavat moduulit sekä testattiin niiden toimintaa.

Luokkia lopullisessa toteutuksessa oli neljä: käyttöliittymä on toteutettu MainWindow–luokkassa, yhteys TM500ping-luokkassa, tiedonsiirto FtpTransfer-luokassa sekä asetukset Settings-luokassa. Nämä toteuttivat nimensä mukaisia tehtäviä. Koska kaikki muut luokat välittivät viestejä, joita esitettiin käyttöliittymässä, valittiin käyttöliittymä pääluokaksi.




Kuva 5: Ohjelman rakenne

Moduulit integroitiin järjestyksessä niiden valmistuttua. Tällä pyrittiin selvittämään moduulien resurssien käyttö sekä tuomaan esille mahdolliset viat. Moduulit luotiin suunnitelman mukaan järjestelmällisesti siten, että toiminnallisuutta pystyttiin jo aikaisessa vaiheessa testaamaan kokonaisuuksina. Moduulit laadittiin tässä järjestyksessä: käyttöliittymä-, yhteys-, tiedonsiirto- ja asetukset-moduuli. Testauksessa käytettiin whitebox-testausta (McConnell 2004, 544) Jokainen metodi testattiin sekä yksin että erikseen heti laatimisen yhteydessä.

### 5.1 Rakenne

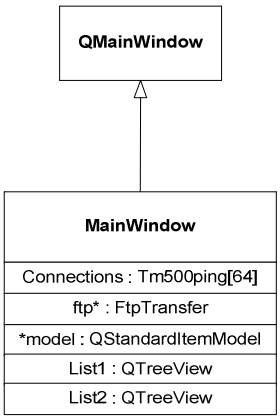
Kappaleessa kuvataan moduulien rakenne luokkakaaviona (Rumbaugh, Jacobson & Booch 2005, 67) sekä sekvenssikaavioilla (Rumbaug ym. 2005, 123). Lisäksi moduulien toimintaa on esitetty käyttäjän silmin aktiviteettikaavioilla (Rumbaugh ym. 2005, 116.). Rakenne toteutettiin kuvan 5: Ohjelmistoprosessin kulku osoittamalla tavalla.

Rakenteen kuvauksissa on viitattu suoraa Qt:n luokkiin, jotka tyypillisesti alkavat Q–kirjaimella. Kaikki tekstissä esitetyt metodit ja luokat on kirjoitettu eri kirjasintyypillä lukemisen helpottamiseksi.

#### 5.1.1 Käyttöliittymä-moduuli - MainWindow

Käyttöliittymäluokan tehtävät perustuivat käytettävyyteen liittyviin vaatimuksiin. Sen kautta käyttäjä pystyisi ohjailemaan muiden luokkien toimintaa. Sen tuli myös esittää tietoa yhteyden tilasta, siirtonopeuksista sekä vasteajasta. Käyttöliittymän tuli olla helppokäyttöinen ja yksinkertainen.

Käyttöliittymäluokasta muodostui luokka, joka luo ja ohjaa muita luokkia. Se luo tiedonsiirto-objektin sekä käyttäjän lisätessä yhteyksiä jokaiselle oman yhteys-objektin.




Kuva 7: MainWindow-luokkakaavio

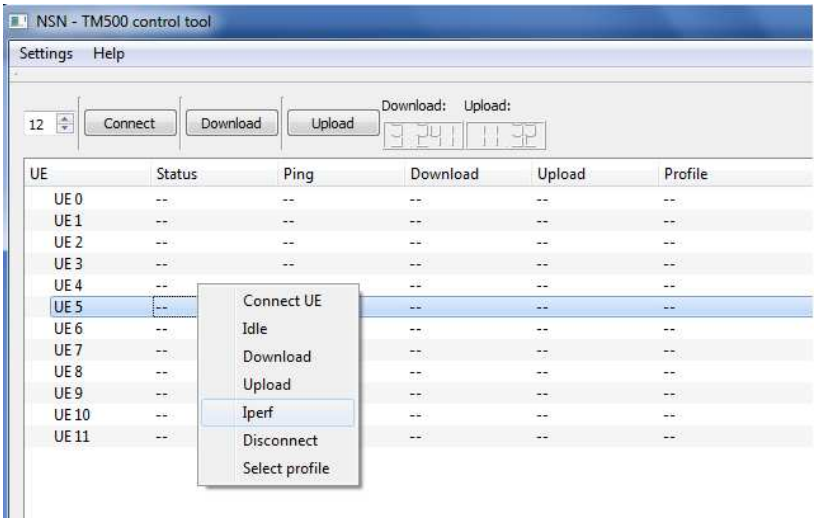
Käyttöliittymän toteuttaminen aloitettiin luomalla pohja Qt:n editorilla( Blanchette & Summerfield 2008, 3. luku). Pohjaan lisättiin pääikkunan valikkorakenne sekä sen toiminnot, ”Connect”-, ”Download”-, ”Upload”–napit ja tiedonsiirron summat näyttävät numerokentät.

Loput käyttöliittymästä luotiin käsin. Tähän kuului kaksi listaa, joista kummassakin esitetään 32 yhteyttä, sekä listojen valikkorakenteet. Listat olivat tyypiltään `QTreeview`, ja ne käyttivät tiedostomallinaan `QStandardModel`:ia, joka jaettiin kahteen osaan `QSortFilterProxyModel`:in avulla.

Yhteyksien hallinta toteutettiin kahdella tavalla. Käyttäjä pystyi käynnistämään kaikki lisätyt yhteydet kerrallaan sekä niille tiedonsiirrot. Toinen tapa oli valita yhteydet listalta joko maalaamalla tai yksitellen ja avaamalla oikeanpuoleisella hiiren painikkeella toimintavalikon, joka näkyy kuvassa 8: Käyttöliittymä.

Toimintavalikon vaihtoehdot sekä toiminnot:

- Connect – Yhdistää valitut yhteydet ja käynnistää pingauksen
- Idle – Yhdistää valitut yhteydet, mutta ei käynnistä pingiä.
- Download – Aloittaa tiedonsiirron downlinkkiin.
- Upload – Aloittaa tiedonsiirron uplinkkiin.
- Stop DL/UL – Katkaisee tiedonsiirron, mutta ei yhteyttä
- Iperf – Käynnistää Iperf ohjelman
- Disconnect – Katkaisee yhteyden.




kuva 8: Käyttöliittymä

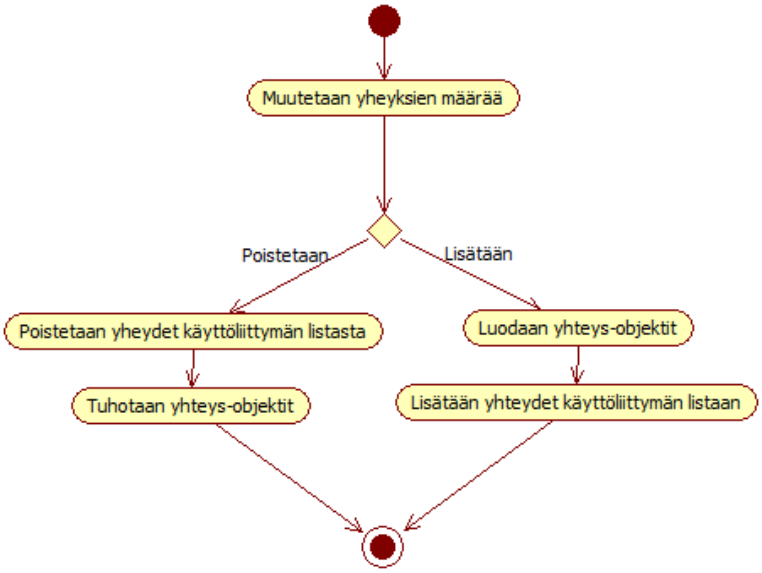
Luokalla oli paljon erilaisia slotteja, joista suurinta osaa kutsuttiin luokan sisältä, käyttöliittymän napeista. Jokaisella napilla oli oma slotti, jonka sisällä hoidettiin tehtävä, kuten yhteyden avaus. Nappien lisäksi luokka otti vastaan signaaleita yhteys- sekä tiedonsiirto-objekteilta. Vastaanotettujen signaalien mukana tuli tietoa yhteyden tilasta ja siirtonopeudesta, sekä jokaisen yhteyden oma numero. Numeron perusteella tieto osattiin sijoittaa käyttöliittymässä oikean yhteyden kohdalle.

Yhteyden tilasta esitetään yhteyslistassa seuraavat tiedot:

- IP-osoite jonka PPP-yhteys saa isäntäkoneelta
- vasteaika millisekunteina ftp -palvelimelta
- downlink siirtonopeus
- uplink siirtonopeus
- valittu profiili kyseiselle yhteydelle.

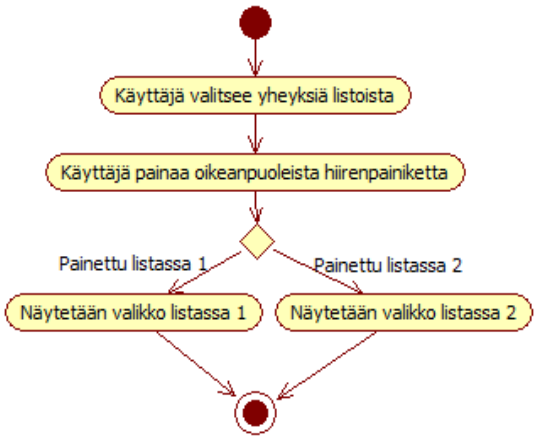
IP-osoite haetaan yhteyden muodostuksen yhteydessä. IP-osoitteella halutaan varmistaa, että laite on varmasti kirjautunut EPC:n. Yhteys-objekti lähettää jokaisen yhdistysvaiheen jälkeen uuden signaalin yhteyden tilasta. Näiden avulla voidaan päätellä, missä kohtaa yhteydessä on vikaa. Vasteaikaa tarvitaan harvemmin testeissä, mutta sen avulla tiedetään, onko yhteys FTP-palvelimeen muodostunut oikein. Tämäkin tieto tulee yhteys-objektilta.

Tiedonsiirtonopeudet saadaan tiedonsiirto-objektilta. Signaalin käsittelyn yhteydessä talletetaan tiedonsiirtonopeudet ylä- sekä alakaistaan erillisiin taulukoihin. Taulukoiden solut summataan ja esitetään käyttöliittymässä.




Kuva 9: Aktiviteettikaavio yhteyden lisäämisestä

Käyttäjä voi joko lisätä tai poistaa yhteyksiä muuttamalla numerovalitsinta käyttöliittymän ylälaidassa. Ohjelma lukee numerovalinnan muutokset ja joko poistaa tai lisää yhteydet listaan. Ennen lisäystä tarkastetaan, ettei yhteyksien lukumäärä ylitä syötettyjen FTP–palvelimien osoitteiden lukumäärää. Mikäli näin käy, käyttäjälle näytetään virheilmoitus. Jos yhteyksiä lisätään enemmän kuin 32, näytetään käyttöliittymässä toinen lista, johon loput yhteydet sijoitetaan.

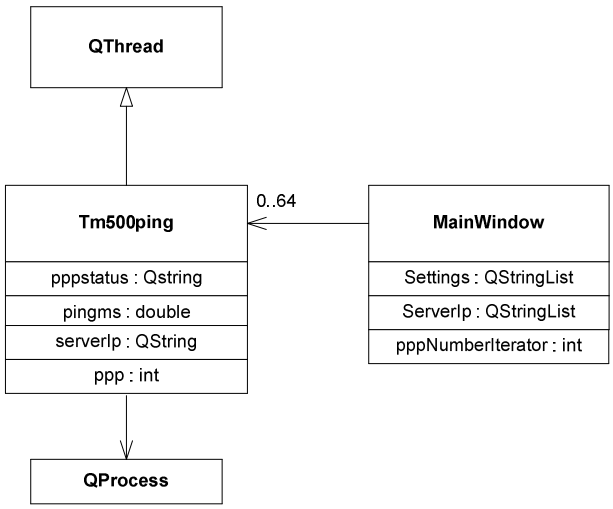



Kuva 10: Aktiviteettikaavio listan toiminnasta

Kun käyttäjä on lisännyt yli 32 yhteyttä, näytetään käyttöliittymässä toinen lista. Käyttäjä pystyy valitsemaan kummastakin listasta yhteyksiä sekä avaamaan toimintavalikon. Toimintavalikko aukeaa siihen listaan, missä käyttäjä painaa oikeanpuoleista hiiren painiketta. Toiminnon valinnassa ohjelma tarkastaa, mitkä yhteydet ovat valittuina ja välittää ne eteenpäin objektille tarvittavien tietojen kanssa.

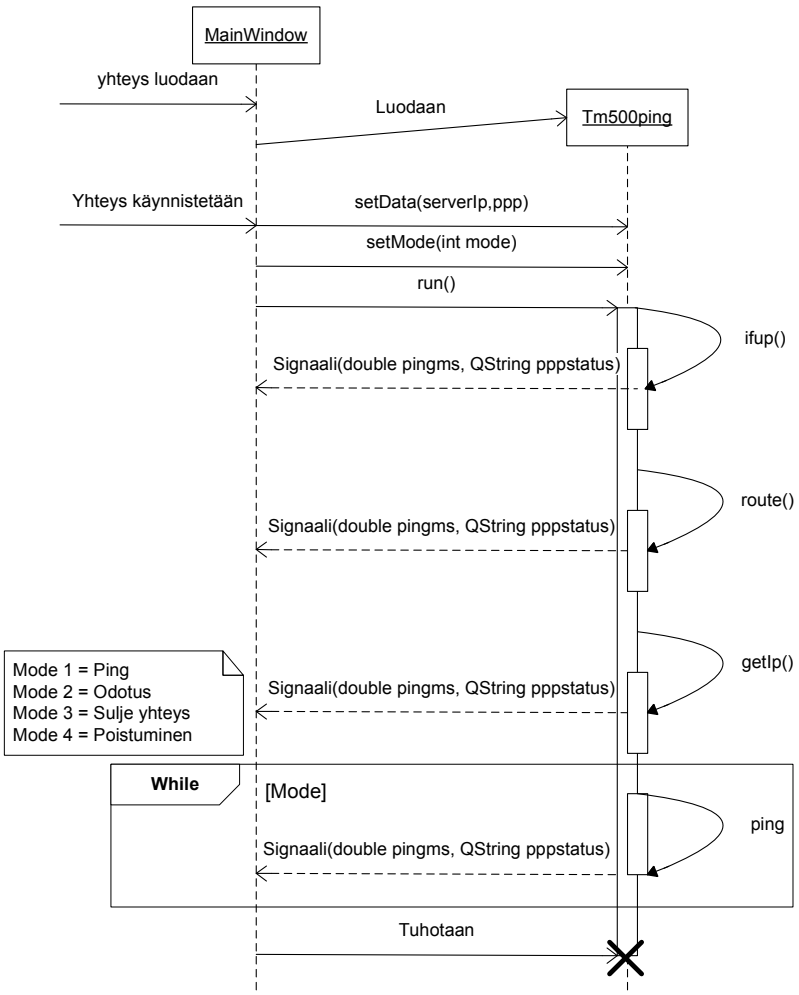
### 5.1.2 Yhteys-moduuli – TM500ping

Yhteys-moduulin päätehtävät ovat yhteyden käynnistys, yhteyden tilatietojen välitys sekä vasteajan mittaaminen ping-prosessia käyttäen. Yhteyden käynnistäminen onnistuu sekä ilman että pingin kanssa.




Kuva 11: TM500ping-luokkakaavio

Luokka perii `QThread:in`, koska luokalla tulee olemaan paljon toimintoja, jotka voisivat lukita käyttöliittymän. Luokan toiminta perustui ulkoisiin prosesseihin, joita ajettiin Qt `QProcess` (Qt Reference Documentation, 2010)-luokan avulla. Prosessien kulusta saatiin tietoa ainakin seuraavista vaiheista: oliko prosessi käynnistynyt, oliko prosessi suorittamassa tehtävää sekä onko prosessi lopettanut. Lisäksi standardituloste pystyttiin ottamaan talteen ja käsittelemään. Nämä tiedot lähetettiin signaalina käyttöliittymäluokalle esitettäväksi.

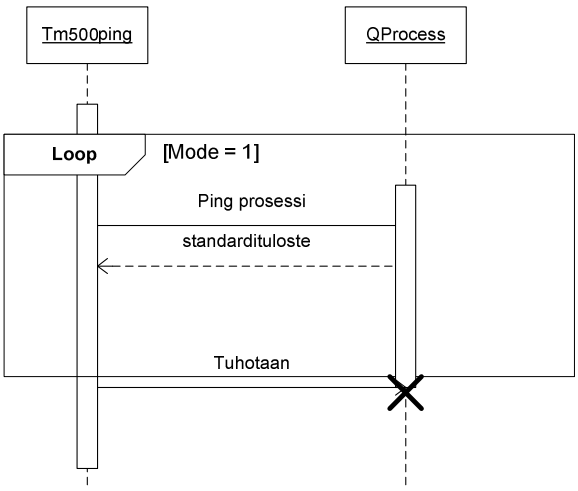



Kuva 12: Sekvenssikaavio yhteys-moduulin toiminnasta

Kun käyttäjä luo uuden yhteyden, luodaan samalla TM500ping-objekti. Luomisen yhteydessä objektille asetetaan `setData`–metodin avulla ppp-numero sekä serverIp-taulukosta luettava FTP–palvelimen osoite.

Kun käyttäjä valitsee käyttöliittymän listan toimintavalikosta (kuva 8: Käyttöliittymä) ”Connect”, ”Idle” tai ”Disconnect”, muutetaan ensin `setMode`-metodilla mode–muuttujan arvo. Tämän jälkeen kutsutaan objektin säikeen käynnistys metodia `run()`, joka luo ping-prosessin sekä käynnistää while–silmukan, joka pyörii kunnes säie halutaan tuhota. Silmukan sisällä on yhteysluokan toiminnallisuus, joka on rakennettu switch–rakenteen sisälle.

Switch toimii mode-muuttujan arvon mukaan. Mode-arvon ollessa yksi, käynnistetään yhteys (kuva 12) ja ajetaan ping-prosessia, kunnes käyttäjä vaihtaa mode-arvoa. Mode-arvolla kaksi yhteys muodostetaan samalla tavalla, mutta ping-prosessia ei käynnistetä. Yhteyden tilaa päivitetään pelkästään kymmenen sekunnin välein kutsumalla `getIp()`-metodia. Mikäli ping-prosessi on päällä, se sammutetaan. Ping-prosessia ei kuitenkaan tuhota. Mode-arvon ollessa kolme, kutsutaan `ifDown()`-metodia, jossa ajetaan ”ifdown”-prosessi, joka sulkee yhteyden. Yhteyden tilan sulkemisesta lähetetään tilatietoja käyttöliittymään, jonka jälkeen lisätään pidempi viive ennen seuraavaa mode-tarkastusta. Mode-arvolla neljä poistutaan While-silmukasta ja samalla säikeen eventloopista. Tämän jälkeen säie tuhotaan.




Kuva 13: Sekvenssikaavio ping-prosessin toiminnasta

Luokka käytti kolmea eri prosessia yhteyden tarkastamisessa ja muodostamisessa:

Ifup ja ifdown käynnistivät yhteyden. Syntaksi on prosessia ajettaessa komento + yhteyden nimi, esimerkiksi ”ifup ppp0”.

Yhteyden muodostamisen jälkeen tarkastettiin onko yhteys saanut oikean IP–osoitteen EPC:lta. Tähän käytettiin ifconfig-komentoa, jonka avulla voidaan tarkastaa sekä asettaa yhteyden asetuksia. Ohjelman syntaksi on ifconfig + yhteys, jota haluttiin tarkastella. Tämän prosessin standardituloste luettiin muistiin sekä käsiteltiin säännöllisiä lausekkeita käyttäen. Säännöllisillä lausekkeilla voidaan esimerkiksi etsiä tiettyjä kirjain- tai numeroyhdistelmiä. (Friedl, J. 2006.)

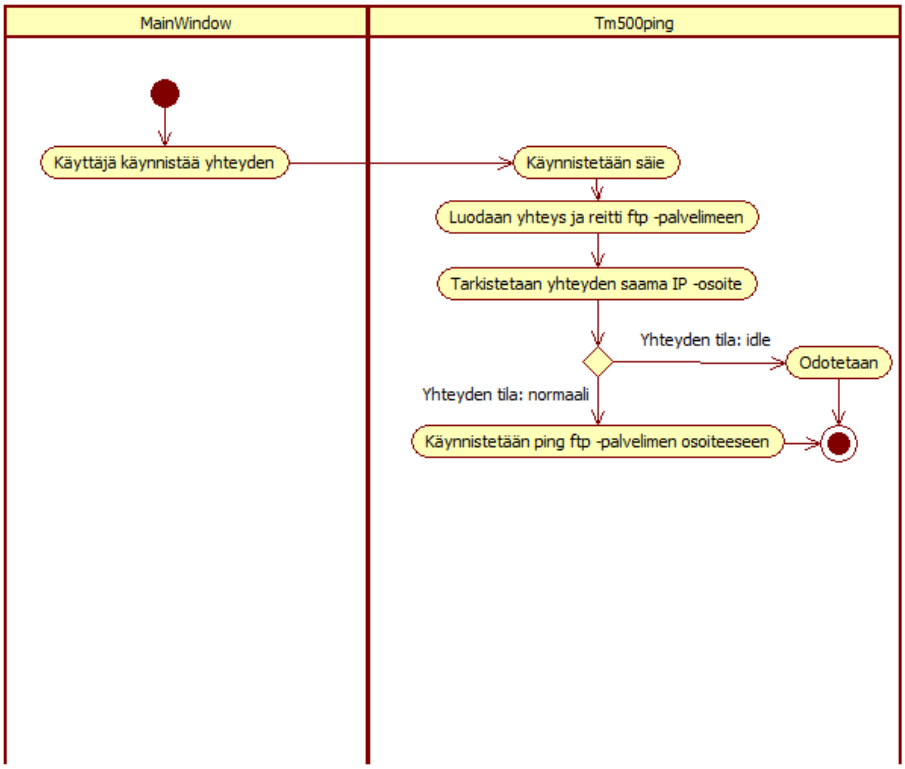
Tässä tapauksessa etsittiin neljä kertaa kolmen numeron sarja, ja nämä sarjat ovat erotettuna pisteellä. Jos tätä sarjaa ei löytynyt, signaloitiin virheilmoitus. IP-osoitteen löydyttyä sitä ei pystytty tarkastamaan varmuudella EPC:n jakamaksi osoitteeksi. Tämä jätettiin käyttäjän vastuulle.

Kun yhteys oli saanut onnistuneesti IP-osoitteen, siirryttiin luomaan reittiä FTP–palvelimelle. Jokaiselle yhteydelle täytyi olla oma osoitteensa, johon luotiin reitti käyttämällä route ohjelmaa.

Ohjelman syntaksi oli route add –net <ftp -palvelin> netmask <netmask> dev <reititettävä yhteys>, esimerkiksi ”route add –net 192.168.3.10 netmask 255.255.255.0 dev ppp0.

Reitityksen onnistumista seurattiin prosessin palauttamista arvoista. Joissakin tapauksissa reititys epäonnistui, mutta ei palauttanut virheellistä ilmoitusta. Tämä oli kuitenkin vain tapauksissa, joissa IP-osoite ei ollut oikea. Tämä tarkoitti sitä, että yhteys EPC:en oli epäonnistunut.

Viimeinen luokan käyttämä prosessi oli ping, jonka käynnistäminen oli valinnainen. Sen standarditulosteesta otettiin talteen vasteaika säännöllisillä lausekkeilla. Ping-ohjelman syntaksi oli ping <Ftp –palvelimen osoite>.

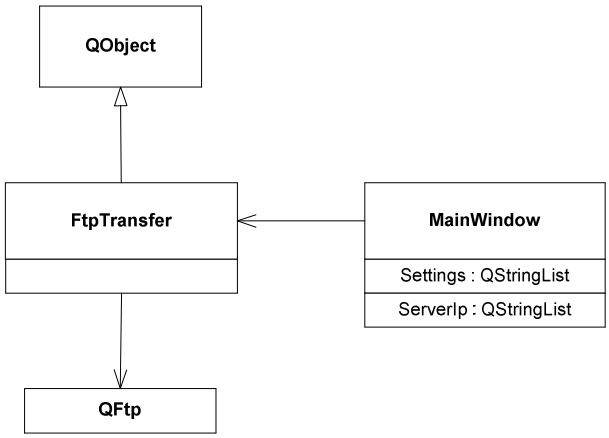



Kuva 14: Aktiviteettikaavio yhteyden muodostamisesta:

Käyttäjä valitsee halutun määrän yhteyksiä listoilta ja painaa oikeaa hiiren painiketta avatakseen toimintavalikon (Kuva 8: käyttöliittymä ). Käyttäjä valitsee ”Connect”, jolloin ohjelma tarkastaa mitkä yhteydet on valittuina. Valittujen yhteyksien kohdalla kutsutaan yhteys–objektia, jolloin sen säie käynnistetään. Tämä säie aloittaa yhteyden muodostuksen, jonka jälkeen siirrytään silmukkarakenteeseen. Silmukassa on sisäänrakennettu vaihtoehtolauseke joko ajamaan ping-prosessia, odottamaan tai sulkemaan yhteyden.

### 5.1.3 Tiedonsiirto-moduuli - FtpTransfer

Luokan päätehtävät ovat aloittaa tiedonsiirto, katkaista se tarvittaessa sekä välittää tietoa tiedonsiirtonopeudesta.

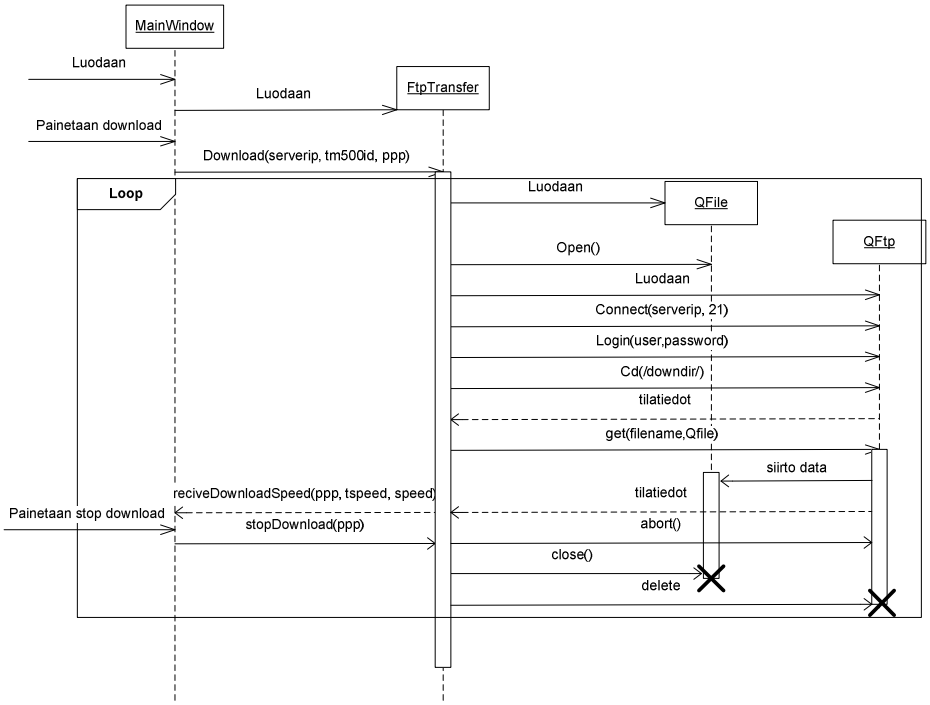



Kuva 15: FtpTransfer-luokkakaavio

Luokka laadittiin aluksi käyttäen Qt:n `QNetworkAccessManager`-moduulia. Qt:n dokumentaation mukaan tällä olisi pitänyt pystyä hallitsemaan ohjelmassa tarvittavia tiedonsiirtoja. Se kuitenkin osoittautui pahasti keskeneräiseksi ja varasi liikaa resursseja. Siirtojen keskeyttäminen jätti lisäksi tiedonsiirron päälle, mutta katkaisi kaiken viestinnän käyttäjälle.

Tiedonsiirron katkaisu oli hyvin tärkeä toiminto, joten `QNetworkAccessManager` oli korvattava jollakin toisella toteutuksella.

Sen tilalle otettiin vanhempi moduuli `QFtp`, jossa oli yhteyden katkaisun kanssa sama ongelma kuin `QNetworkAccessManager`:issa. `QFtp`:n toiminta erosi `QNetworkAccessManager`:ista kuitenkin siten, että jokaiselle siirrolle täytyi luoda oma `QFtp`-objekti. Kun objekti siirron keskeytyksen yhteydessä tuhottiin, loppui myös tiedonsiirto. Tämä toteutus ei ollut oikeaoppinen, ja se pyrittiin ottamaan huomioon testauksessa.



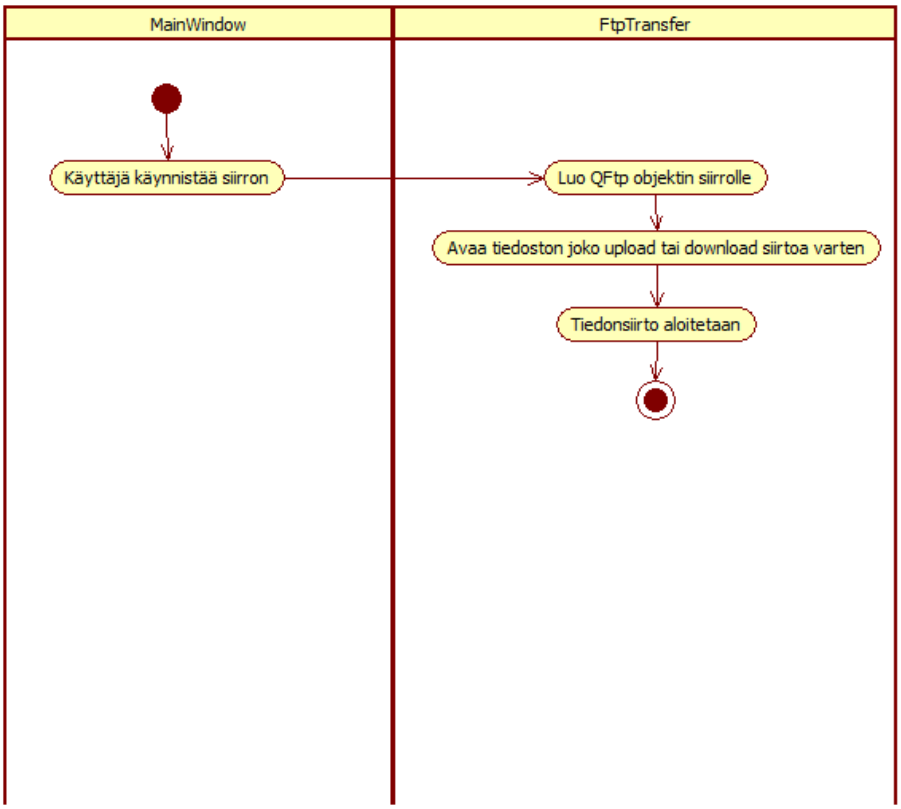

Kuva 16: Sekvenssikaavio tiedonsiirto-moduulin toiminnasta

Kuvassa on esitetty tiedonsiirto-objektin toiminta alakaistaan tapahtuvassa siirrossa. Tiedonsiirto-objekti, eli `FtpTransfer` luodaan käyttöliittymän muodostimessa.

Kun käyttäjä haluaa aloittaa tiedonsiirron, kutsutaan `download`–metodia. `Download`-metodi välittää attribuutteina yhteydelle annetut asetukset. Näitä asetuksia ovat seuraavat: FTP–palvelimen IP-osoite, TM500id, ja ppp–numero. Metodi luo ensimmäisenä `QFile`–objektin, jota Qt käyttää tiedostojen käsittelyyn. Alakaistaan siirrettäessä avattiin `QFile` suoraan /dev/null-tiedostoon, joka poistaa suoraa kaiken siihen tallennetun tiedon. Yläkaistaan siirrettäessä käytettiin määriteltyjä tiedostoja, joita oli kolmea eri kokoa. TM500id-attribuuttia käytettiin yläkaistaan siirrettäessä tiedoston nimenä. Kun `QFile` oli onnistuneesti avattu, luodaan `QFtp`–objekti.

QFtp–objektilla avataan yhteys palvelimeen kutsumalla connect–metodia, jonka mukana välitetään serverIp eli palvelimen osoite sekä FTP-portti. Tämän jälkeen kutsutaan QFtp–objektin login–metodia jonka attribuutteina välitettiin käyttäjätunnus sekä salasana. Kun kirjautuminen palvelimelle oli suoritettu, vaihdettiin kansiota cd–metodilla.

Nyt voidaan aloittaa tiedonsiirto alakaistaan get- ja yläkaistaan put –metodeilla. Get–metodi tarvitsee parametreina FTP–palvelimella sijaitsevan tiedoston, joka halutaan ladata, sekä aiemmin avatun QFile:n, johon ladattu tieto haluttiin sijoittaa. Put–metodi toimi samalla tavalla, mutta QFile:n pitää avata tiedosto, joka halutaan siirtää, toinen attribuutti on siirrettävän tiedoston nimi palvelimella.



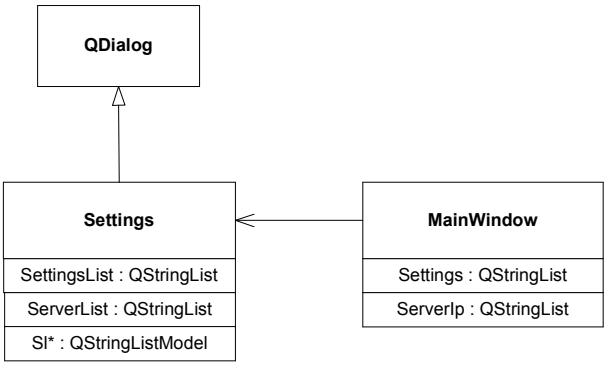

Kuva 17: Aktiviteettikaavio FTP-siirron aloittamisesta

Käyttäjä on lisännyt listaan vähintään yhden yhteyden ja yhdistänyt sen onnistuneesti. Käyttäjä valitsee yhteyden listalta ja painaa oikeanpuoleista hiiren painiketta. Tällöin käyttäjälle esitetään valikko (Kuva 8: Käyttöliittymä), joka näytetään siinä listassa missä käyttäjä on nappia painanut. Listasta käyttäjä valitsee joko ”Upload”- tai ”Download”-valinnan, jolloin ohjelma tarkastaa, mitkä yhteydet ovat valittuina sekä kutsuu tiedonsiirto-objektia.

### 5.1.4 Asetukset-moduuli - Settings

Luokan tehtävänä oli pyytää ohjelman toimintojen kannalta pakollisia asetuksia käyttäjältä. Näitä olivat PPP:n käyttämät yhteysasetukset sekä FTP–palvelimien osoitteet.

Asetuksetluokka on toinen ohjelmassa käytetty käyttöliittymäluokka. Käyttöliittymä koostuu leikelehdestä, jossa on omat sivunsa PPP–asetuksille, FTP–asetuksille ja aikataulutukselle. Käyttöliittymässä käytettiin tekstikenttiä sekä yhtä listaa, joka näytti käyttäjän lisäämät FTP–palvelimien osoitteet.

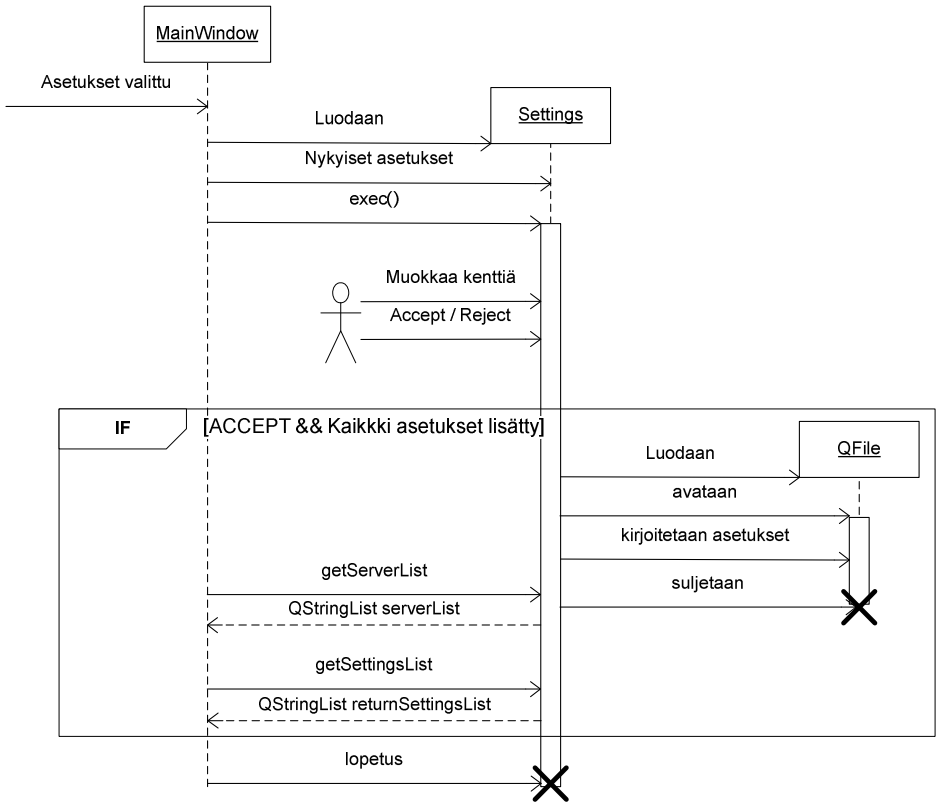



Kuva 18: Settings-luokkakaavio

Luokka perii `QDialog`:in, jonka avulla Qt:ssa voi helposti hoitaa esimerkiksi asetuksien sekä muiden tietojen kyselyn (Blanchette & Summerfield 2005, 2. luku). Sen toiminta perustuu tapaan, jolla objektia kutsutaan.

Se voidaan tehdä joko estävänä tai vapaana. Asetus-objektia kutsutaan estävänä, jolloin käyttäjän täytyy hyväksyä tai sulkea ikkuna päästäkseen takaisin käyttöliittymään.

Asetukset-moduulin käyttöliittymä luotiin Qt:n editorilla. Se koostuu tekstikentistä, joihin asetukset syötetään, sekä yhdestä `QListView`:tä. `QListView` näyttää `QStringListModel`:in, johon luetaan ftp –palvelimien osoitteet.

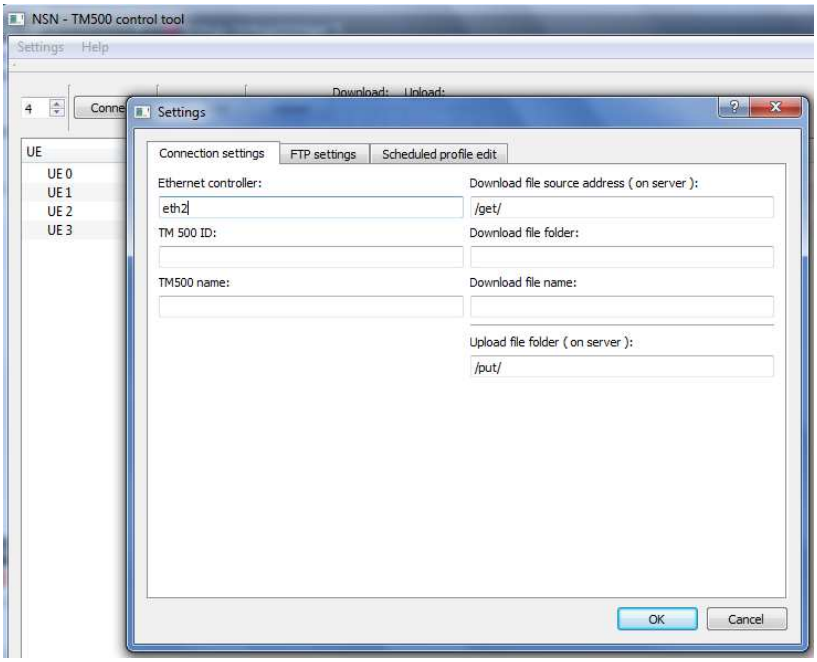



Kuva 19: Sekvenssikaavio settings-moduulin toiminnasta

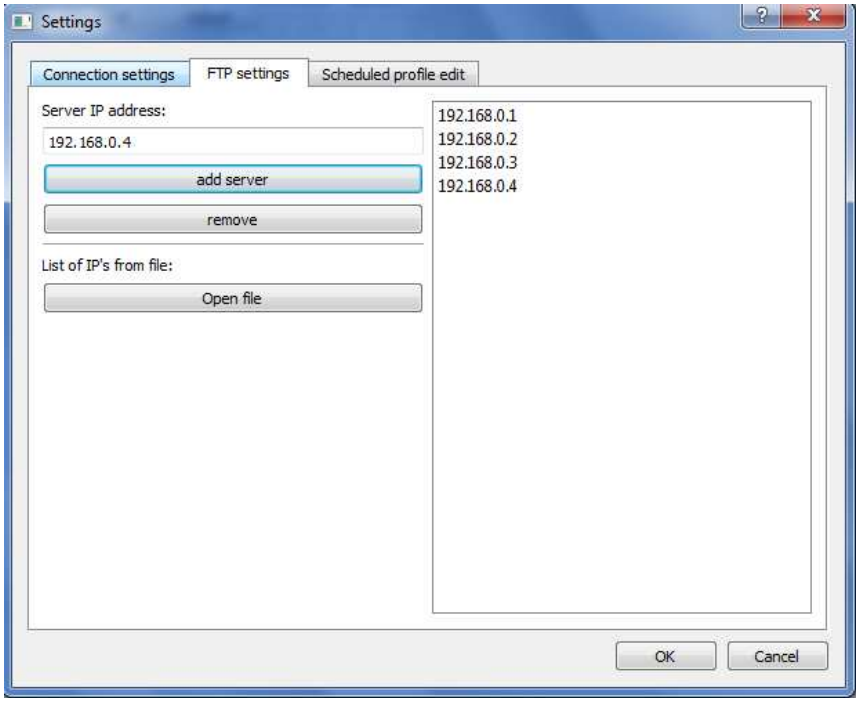
Asetukset-objekti luodaan vain hetkellisesti, kun sitä tarvitaan. Luonnin jälkeen sille asetetaan sen hetkiset asetukset `setSettings(QStringList)` ja `setServerList(QStringList)`-metodeilla. Tämän jälkeen kutsutaan `exec()`-metodia, jolla esitetään `QDialog`-tyyppinen(Kuva 20, 21) ikkuna. `Exec()`-metodilla `QDialog`-ikkuna on estävä, joten käyttäjän on hyväksyttävä tai suljettava se, ennen kuin voi palata käyttämään ohjelman muita toimintoja.

Käyttäjän painaessa ”Ok” tarkastetaan ensin, onko kaikki kentät täytetty. Jokainen kenttä täytyy täyttää ohjelman toiminnan takaamiseksi, mikäli jokin kenttä uupuu, ilmoitetaan siitä käyttäjälle virheilmoituksella.

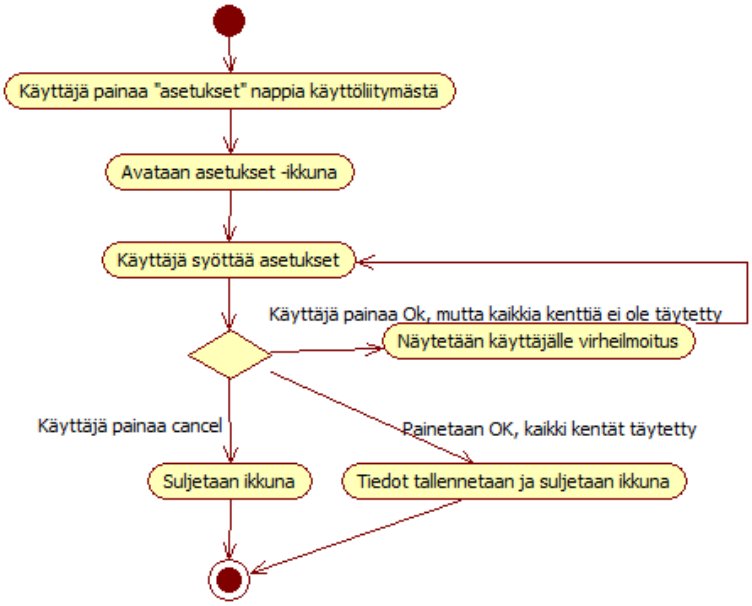
Kun kaikki kentät on täytetty, palataan käyttöliittymään. `Exec()`-metodi palauttaa käyttöliittymään muuttujan 1, mikäli asetukset-objekti lopetettiin hyväksymällä, tai 0, jos se suljettiin muulla tavalla. Jos palautusarvo oli 1, haettiin asetukset `getServerlList-` sekä `getSettingsList`-metodeilla. Tämän jälkeen asetukset-objekti tuhottiin.




Kuva 20: PPP-asetukset




Kuva 21: FTP-asetukset




Kuva 22: Aktiviteettikaavio asetusten syöttämisestä

Käyttäjä valitsee ”asetukset” eli ”settings” valikon alta käyttöliittymästä. Käyttäjälle avataan asetukset-ikkuna. Käyttäjä syöttää kenttiin tiedot yhteydestä sekä FTP–servereistä. Tiedot luetaan poistuttaessa ensin listaan, joka sen jälkeen palautetaan pääluokkaan. Jos käyttäjä jättää kenttiä tyhjiksi, esitetään siitä virheilmoitus, joka ohjeistaa täyttämään kaikki kentät.

### 5.2 Moduulitestaus

Moduulitestaus suoritettiin kääntämällä ohjelmaa usein ja testaamalla metodit heti laatimisen jälkeen. Testaaminen hoidettiin suurelta osalta oikean käyttöympäristön ulkopuolella käyttäen ohjelman kannalta lähes identtistä testiympäristöä, koska käyttöympäristön TM500-laitteet olivat jatkuvassa käytössä, ympäristö oli helppo pystyttää pienellä vaivalla.

Testaus suoritettiin niin kutsuttuna white-box-testauksena (McConnell 2004, 542.), jossa metodien toiminta tunnettiin. Metodien toiminta tutkittiin arvoilla, joita niihin tultaisiin syöttämään. Vikatilanteissa ja virheenjäljityksessä (McConnell 2004, 577) käytettiin hyväksi debug–tilaa sekä debug–viestejä.

### 5.2.1 Käyttöliittymä-moduuli

Moduulin toiminnan testaaminen keskittyi käyttäjän ohjaamiin metodeihin. Näitä olivat yhteyden luominen sekä sen ohjaaminen. Näiden lisäksi se otti vastaan tietoa luomiltaan luokilta.

Käyttöliittymän toimintoja testattiin tulostamalla debug-viestejä metodien suorituksen yhteydessä. Jokainen metodi kirjoitti konsoliin metodin aloituksesta sekä lopetuksesta. Myös attribuuttien tulo ja lähtöarvot tulostettiin. Nämä viestit poistettiin aina metodin toiminnan pitkän testauksen jälkeen, mikäli niissä ei ollut riippuvuuksia muihin metodeihin.

Esimerkkinä yhteyden lisäyksen yhteydessä kutsuttava metodi:

```
if( pppNumberIterator > value){
    while(pppNumberIterator > value){
        pppNumberIterator--;
        removeListItem();
        qDebug() << pppNumberIterator; //tulostaa pppNumberIteraattorin
    }
} else {
    while(pppNumberIterator < value &serverIp.length()>pppNumberIterator){
        createListItem();
        pppNumberIterator++;
        qDebug() << pppNumberIterator; //tulostaa pppNumberIteraattorin
        }
    }
```

### 5.2.2 Yhteys-moduuli

Yhteysmoduulin testauksessa käytettiin apuna sekä käyttöliittymämoduulia sekä pientä testiohjelmaa. `IfUp()`- ja `ifDown()`-metodit luotiin ensimmäisenä. Niiden testaamiseen käytettiin testiohjelmaa, jossa ensin luotiin yhteys-moduuli. Sen jälkeen se ajoi yhteyden ylös muutaman sekunnin päästä alas.

Lopulliset testaukset suoritettiin yhdistettynä käyttöliittymä-moduuliin. Näin voitiin kokeilla suoraa käyttöliittymä-moduulin lähettämillä raja-arvoilla moduulin toimintaa. Samalla pystyttiin testaamaan käyttöliittymä-moduulin ominaisuuksia, kuten tietojen esittämistä. Tästä kerrotaan enemmän integraatiotestauksen yhteydessä.

### 5.2.3 Tiedonsiirto-moduuli

Tiedonsiirtomoduulin testauksessa käytettiin jo integroituja käyttöliittymä sekä yhteys-moduuleja. Koska Asetukset-moduuli tehtiin viimeisenä, moduulin testauksessa käytettiin pakotettuja asetuksia. Moduulin toimintaa tutkittiin kutsumalla tiedonsiirron `download()` -ja `upload()`-metodeja käyttöliittymästä. Toimintaa seurattiin `QFtp`-luokan lähettämillä signaaleilla. Signaalit yhdistettiin slotteihin, joista toiminnan lisäksi kirjoitettiin `qDebug()`-metodin avulla konsoliviestejä. Tällä tavalla pystyttiin helposti seuraamaan moduulin toimintaa.

### 5.2.4 Asetukset-moduuli

Asetukset-moduulin testit keskittyivät asetusten välitykseen sekä asetus-tiedostojen oikeanlaiseen muokkaamiseen. Koska asetukset haluttiin käyttöliittymä-moduuliin, joka oli jo valmis, integroitiin moduulit heti. Moduulin testaus suoritettiin muiden moduulien tapaan käyttämällä konsoliviestejä sekä debug–toimintoa sekä seuraamalla palautusarvoja käyttöliittymä-moduulista. Tiedostoon kirjoittamisen testaamisessa käytettiin ensin tyhjää tiedostoa, jolloin sen sisällön tarkistamisella voitiin todeta, että ohjelma kirjoitti asetukset.

### 5.3 Integraatiotestaus

Integrointivaihe suoritettiin iteratiivisesti: aina kun yksittäinen moduuli oli valmis, se yhdistettiin kokonaisuuteen. Näin päästiin kokeilemaan ohjelman toimintaa kokonaisuutena aikaisessa vaiheessa. Tässä vaiheessa korostui ohjelman suorituskyvyn mittaus. Ohjelman varaaman muistin ja prosessoriajan määrää pystyttiin näin kohdentamaan helposti tiettyyn ohjelman osaan.

Ensimmäinen integrointi tehtiin käyttöliittymän sekä yhteys-moduulin välillä. Tässä vaiheessa keskityttiin testaamaan yhteyden muodostumista sekä vasteajan mittausta. Yhteyden muodostumista seurattiin käyttämällä tcpdump–ohjelmaa palvelinkoneella, joka esitti yhteyden neuvottelun tarkasti. Ping–prosessin lähettämiä paketteja seurattiin BWM-ng:llä, joka esittää kaikkien yhteyksien tiedonsiirron eriteltynä. Tätä käytettiin myös myöhemmin hyväksi seuraamaan yhteydenmuodostumista.

Viimeisessä vaiheessa ohjelmaa testattiin myös FTP– siirtojen osalta, joita myös seurattiin BWM-ng:llä. Koko ohjelman kehityksen ajan mitattiin myös sen käyttämää prosessoriaikaa sekä muistinkulutusta. Nämä tiedot saatiin käyttämällä konsolipohjaista TOP–ohjelmaa sekä graafista System-monitoria. Näitä tuloksia kirjattiin ylös ja ne löytyvät testiraporteista.

# 6 SYSTEEMITESTAUS

Systeemitestaus alkoi heti kun kolme tärkeintä moduulia, käyttöliittymä, yhteydenhallinta sekä ftp olivat valmiit. Testausta suoritettiin ajamalla testejä käyttötapauskuvauksien mukaan, sekä suorittamalla toimintoja satunnaisesti. Käyttötestien lisäksi käytettiin hyväksi testimetodeja, joilla pyrittiin rasittamaan ohjelmaa normaalin käytön äärirajoille. Ohjelmisto jätettiin myös testattavaksi käyttöympäristöön kehityksen loppupuolella.

## 6.1 Testaus kehitysympäristössä

Ohjelmaa pyrittiin rasittamaan mahdollisimman laajamittaisesti testiympäristössä (liite 1). Eniten testauksessa kului aikaa FTP–toteutuksen testaamiseen, joka aiheutti ongelmia suurella resurssienkulutuksella. Ftp:n suorituskykyä sekä muistivuotoja tutkittiin ajamalla erikseen sen testaukseen toteutettuja skriptejä jotka käynnistivät siirron ja sulkivat sen viiden sekunnin kuluttua. Skripti kirjasi ylös montako onnistunutta siirtoa testin aikana suoritettiin. Tiedonsiirron lopetusta ja uudelleenaloitusta testattiin lisäksi myös käyttämällä pientä tiedostoa, joka siirtyi nopeasti. Ohjelma aloitti normaalin toiminnan mukaan uuden siirron entisen loputtua. Näin saatiin rasitusta aikaan ilman erillisiä metodeja. Testien aikana seurattiin prosessorikuormaa, muistinvarausta ja kaistankäyttöä.

Testauksessa ongelmana oli, että testiympäristöä ei voitu rakentaa täysin vastaavaksi oikean ympäristön kanssa. Yksi eroavaisuus oli suuremmat viiveet yhteyden muodostumisessa. Viiveet johtuivat siitä, että oikea ympäristö PPP–yhteyden lisäksi neuvotteli EPC:n kanssa yhteydelle IP:n. Tiedonsiirron osalta testit vastasivat oikean ympäristön toimintaa.

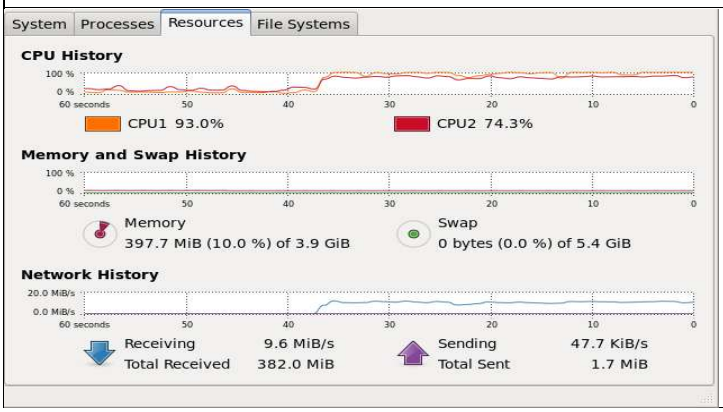
### 6.2 Testaus käyttöympäristössä

Ohjelmiston testaus käyttöympäristössä (Kuva 1) jäi suhteellisen vähäiseksi, koska toteutukseen kului suunniteltua enemmän aikaa edellä kuvattujen odottamattomien ongelmien vuoksi. Monia testejä kuitenkin ehdittiin suorittaa tämän opinnäytetyön teon aikana ja testausta jatketaan myös tämän jälkeen. Testien tulokset olivat lupaavia ja ohjelmisto toimi vaatimusten mukaisesti.

Ohjelmasta saatiin oikeassa ympäristössä paljon pieniä vikoja korjattua, kuten yhteydenmuodostamisessa ilmenneet odotettua isommat viiveet. Ohjelmaa testattiin samalla tavalla kuin testiympäristössä suorittamalla käyttötapauskuvauksien mukaisella tavalla yhteyden ylösajoa, lopetusta sekä tiedonsiirtoa. Näiden lisäksi ajettiin rasitustestejä, joilla voidaan varmistua, ettei ohjelman toiminta häiritse TM500-laitetta. Tämä osoittautui isoksi ongelmaksi ohjelman toiminnassa. Yhteydet piti muodostaa yksitellen eikä liian nopeassa tahdissa. Jos kaksi yhteyttä yritettiin avata samaan aikaan, oli tuloksena hyvin useasti laitteen uudelleenkäynnistäminen.

Esimerkki testitapauksesta:

| |
|---|
| top - 21:17:59 up  2:07,  4 users,  load average: 4.20, 2.34, 2.64 |
| Tasks: 262 total,  14 running, 232 sleeping,  0 stopped,  16 zombie |
| Cpu(s): 29.6%us,53.7%sy,0.0%ni,9.7%id,0.0%wa,0.2%hi,6.8%si,0.0%s |
| Mem:  4080772k total, 1504044k used,  2576728k free, 78040k buffers |
| Swap:  5668856k total, 0k used,  5668856k free,  1018888k cached |
| TM500app 17759 nobody  20 0 1936 632 552 R  8.6  0.0  0:04.48 |




testissä oli 16 yhteyttä, joista kaikissa päällä tiedonsiirto alakaistaan.

# 7 JATKOKEHITYSMAHDOLLISUUDET

Työn aikana kirjattiin useita hyviä Ohjelman jatkokehitysideoita, joista osa on jo tätä raporttia kirjoittaessa osittain toteutettu. Suurimmat jatkokehityksen tarpeet ovat ohjelman päivittäminen Qt:n uuden version jälkeen, sillä `QNetworkManager`-moduuli, joka hoitaa tiedonsiirron, oli pahasti kesken eikä soveltunut ohjelman käyttöön ollenkaan. Itse ohjelman koodia tulisi myös siistiä sekä ohjelman arkkitehtuuria katsoa uudestaan. Tehtäviä tulisi jakaa paremmin luokkien kesken.

Ohjelman yksi keskeinen toiminta, pingaus, tulisi hoitaa Qt:lla itsellään. Tällä hetkellä käytössä on ulkoinen ping-prosessi. Pingin korvaaminen Qt:lla toteutetulla vasteajanmittauksella voisi vapauttaa pienen määrän resursseja esimerkiksi FTP:n käyttöön.

Tiedonsiirron sujuvuutta voitaisiin parantaa korvaamalla FTP:n toisella siirtomuodolla, kuten esimerkiksi käyttämällä raakasocketteja. Qt:n luokkakirjasto sisältää `QtcpSocket`- luokan, jolla voidaan muodostaa raaka TCP-yhteys serverin ja testikoneen välille. Siirtotiellä voidaan sitten siirtää satunnaisesti generoitua dataa.

Ohjelmiston alkuperäisvaatimuksissa oli kaistankäytön mukaan piirtyvä graafi, jonka toteutus aloitettiin, mutta koska sen ei nähty olevan tärkeimpiä ominaisuuksia, se tiputettiin taka-alalle ja se ei ehtinyt tässä raportissa kuvattuun versioon. Graafin ohjelmointi kokonaan itse ei välttämättä ole tarpeen, koska sen toteuttamiseen on valmiita Qt-ympäristöön tarkoitettuja kirjastoja, sekä useita Qt:n ulkopuolisia kirjastoja ja sovelluksia. Koska ohjelmisto kirjaa ylös kaistankäyttöä lokitiedostoon, voisi siitä testauksen jälkeen piirtää graafin, josta voidaan tutkia onko siirtokaistalla tapahtunut suuria poikkeamia.

Windows-käyttöjärjestelmän tuki saattaa olla tulevaisuudessa aiheellinen muutos. Sen tekemiseen ohjelmiston yhteydenotto- sekä asetus–moduulit joudutaan suurilta osin kirjoittamaan uudelleen.

Vaatimusten mukaan käyttökieleksi jäi englanti. Käyttökieliä voisi tulevaisuudessa lisätä kielipaketeilla. Tämä on toteutettavissa suhteellisen helposti Qt Linguist-ohjelman avulla.

Ohjelman lisäksi parannettavaa olisi käyttöympäristössä, jossa nykyiset kannettavat tietokoneet voitaisiin korvata tehokkaammalla pöytäkoneella, joka voisi hoitaa ylimääräisten verkkokorttien avulla isomman määrän yhteyksiä. Kone voisi olla etähallittava ja sijoitettuna esimerkiksi serveritelineeseen.

# 8 YHTEENVETO

Tässä opinnäytetyössä laadittiin ohjelmistoprojektina työkalu LTE–testiympäristöön. Työn ohella tutustuttiin sekä LTE–verkkoihin että niiden testaukseen. Työn idea lähti aikaisemmasta vaikeakäyttöisestä sekä hitaasta Linuxin shell–skripteillä toteutetusta kokonaisuudesta, joka nyt päivitettiin oikeaksi työpöytäsovellukseksi.

Ohjelmisto ei toteutunut täydellisesti aikataulussa. Se oli raportin kirjoitushetkellä vielä testikäytössä, mutta alkuperäisen vaatimusmääritelmän tavoitteet saavutettiin sekä osa kehityksen varrella syntyneistä ideoista toteutettiin. Ohjelmisto toimii siedettävästi vielä kaksinkertaisella käyttäjämäärällä vanhaan toteutukseen verrattuna, ja sen käyttö on sujuvampaa. Ohjelmaan lisättiin myös ylimääräisiä ominaisuuksia, kuten yhteyden käynnistäminen ilman pingiä, sekä Iperf:n käynnistäminen ohjelmasta käsin. Lisäksi raportin kirjoitusvaiheessa oli työn alla aikataulutusmahdollisuus, jolla voidaan hallita yhteyksiä luomalla niille omat aikatauluprofiilit. Suunnitteluvaiheeseen jääneet ideat lisättiin jatkokehitysmahdollisuuksiin.

Työ oli ensimmäinen toteuttamani isompi ohjelmistoprojekti. Kokemusta karttui riskien hallinnasta, kuten odottamattomien ongelmien huomioimisesta, aikataulutuksesta ja työmäärän arvioimisesta. Suurin osa työstä oli ohjelmistokehitystä, kun aiempi kokemukseni oli vain pienten ohjelmien tekemistä. Tämä vei selkeästi eniten aikaa. Ohjelmointitaitoni karttuivat työn aikana, mutta ohjelmiston arkkitehtuurisessa sekä toiminnallisessa toteutuksessa on parantamisen varaa. Suurin ongelma toteutusvaiheessa oli toimimattomaksi todettu Qt:n QNetworkAccessManager-moduuli, joka esti tiedonsiirron toteutuksen suunnitellulla tavalla. Tämä osittain aiheutti myös sen, että ohjelmisto kaatuili ja kulutti paljon resursseja. Ongelmat ratkaistiin käyttämällä vanhempaa QFtp-moduulia, joka toimii, mutta on kyseenalaisesti toteutettu. Muistivuotoja tai suurempia ongelmia ei testeissä kuitenkaan ilmennyt.

Koko työ oli erittäin mielenkiintoinen. Ohjelma tullaan saattamaan loppuun ja sen jatkokehitysideat toteutetaan.

**Muotoiltu:** englanti (Iso-Britannia)

## LÄHDELUETTELO


Blanchette, J. & Summerfield, M. 2008. C++ GUI programming with Qt4. Trolltech Press

Friedl, J. 2006 Mastering regular expressions. O'Reilly Media. USA: Sebastopol,California

McConnell, S. 2004. Code Complete: A Practical Handbook Of Software. Microsoft Press

Moray, R. 2009. LTE and evolution to 4G wireless. Kiina: Agilent Technologies

Rumbaugh, J., Jacobson, I. & Booch, G. 2005. The Modelling Language Reference Manual. Boston: Pearson Education, Inc

Qt Reference Documentation (4.6), Nokia. 2010. Hakupäivä 10.4.2011 http://doc.trolltech.com/4.6/index.html

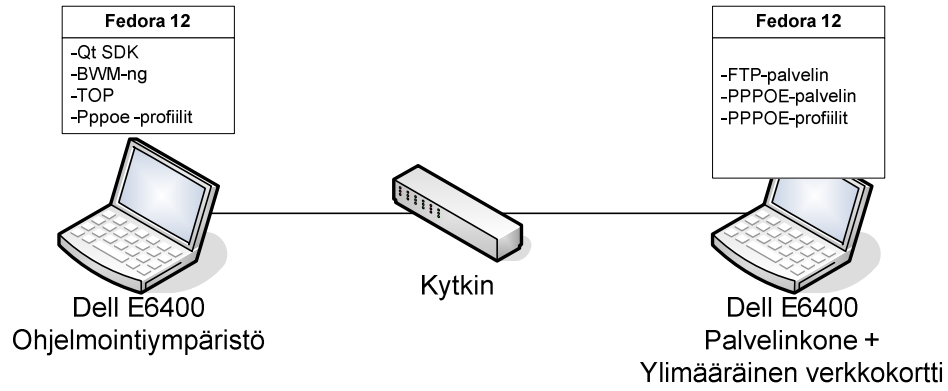
Qt A cross-platform Application and UI Framework, Nokia. 2010. Hakupäivä 24.2.2011 http://qt.nokia.com/

3GPP. 2010. Hakupäivä 20.11.2010 http://www.3gpp.org/

Long Term Evolution, 3GPP. 2010. Hakupäivä 20.11.2010 http://www.3gpp.org/LTE.html

**LIITTEET**

# 1 YMPÄRISTÖN ASENNUS JA KÄYTETYT LAITTET




Kuva: Testi ja ohjelmointiympäristö

Ohjelmointiympäristön ohjelmat ja asetukset:

- käyttöympäristössä käytössä olevat PPPoE –profiilit
- BWM-ng kaistanseuranta ohjelma
- TOP sekä graafinen system-monitor muistin sekä prosessorin kuormituksen seurantaa varten.

Palvelinkoneen ohjelmat ja asetukset:

- Kaksi verkkokorttia, toinen ottamaan vastaan PPP –yhteyksiä ja toiselle luotu useita IP–osoitetta emuloimaan EPC:n jakamia osoitteita.
- FTP–pavelin jossa identtinen hakemistorakenne ja käyttäjäasetukset kuin oikeassa käyttöympäristössä.
- PPPoE-palvelin, joka ottaa vastaan vain ohjelmistoympäristön tilin asetuksilla tulevat yhteydet.

### 1.1 Qt Framework

Qt asennettiin käyttämällä  yum –paketinhallinta ohjelmalla.

yum –install qt-devel

yum –instal qt-assistant

Asennusten jälkeen Qt framework sekä Qt assistant sekä Liguest olivat valmiita käyttöön.

### 1.2 Testiympäristön asennus

Testausta varten toisesta käytössä olleesta kannettavasta luotiin testiympäristö, johon asennettiin FTP– ja PPP-palvelimet. Palvelimen asennuksessa ei tarvinnut ottaa huomioon tietoturvaa, sillä laitteet eivät ole yhteydessä verkkoon vaan ovat kytkettynä pelkästään toisiinsa. FTP ohjelmaksi valikoitui vsftpd sen helppokäyttöisyyden takia.

Asennus suoritettiin seuraavasti:

Otetaan superuser oikeudet:

Admin@NSN:~$ sudo su -

```
[sudo] password for Admin:
root@NSN:~#
```

Asennettiin vsftpd:

```
root@NSN:~# yum –install vsftpd
```

Kun asennus oli valmis, oli syytä tarkastaa asetukset. Serverin asetukset löytyivät seuraavasta tiedostosta. Asetusten muutoksien jälkeen täytyi käynnistää palvelin uudelleen.

```
root@NSN:~# nano /etc/vsftpd/vsftpd.conf
```

seuraavaksi lisättiin ryhmä testuser FTP–tiliä varten:

```
root@NSN ~# groupadd testuser
```

Luodaan kansiot siirtoja varten ja annetaan käyttöoikeudet testuser ryhmälle:

```
root@NSN ~# mkdir /usr/ftp/upload
root@NSN ~# chmod 750 /usr/ftp/upload
root@NSN ~# chown root:testuser /usr/ftp/upload

root@NSN ~# mkdir /usr/ftp/download
root@NSN ~# chmod 750 /usr/ftp/download
root@NSN ~# chown root:testuser /usr/ftp/download
```

lisätään käyttäjä testuser ryhmään:

```
root@NSN ~# useradd -g testuser -d /usr/ftp/ tester
root@NSN ~# passwd tester
```

Seuraavilla komennoilla voidaan käynnistää/sulkea/uudelleen käynnistää tai tarkastaa palvelimen tila:

```
root@NSN ~# service vsftpd start
root@NSN ~# service vsftpd stop
root@NSN ~# service vsftpd restart
root@NSN ~# service vsftpd status
```

PPPoE palvelimen asennus suoritettiin asentamlla rp-pppoe –paketti, joka mahdollistaa myös kevyen palvelimen teon PPPoE-palvelinohjelmalla. PPPoE-palvelin on tarkoitettu vain kevyeen testikäyttöön ja siitä puuttui helppo tapa ajaa useita tilejä eikä se välttämättä ollut pitkässä käytössä vakaa, testikäyttöön tämä silti riitti hyvin.

Asennus aloitettiin hakemalla super-user oikeudet:

```
Admin@NSN:~$ sudo su -
[sudo] password for Admin:
root@NSN:~#
```

Seuraavaksi haettiin paketti rp-pppoe

```
root@NSN:~# yum –install rp-pppoe
```

Kun asennus oli valmis, asetettiin PPPoE-palvelimen asetukset:

```
root@NSN:~# nano /etc/ppp/ppp-server.conf
```

Tiedostoon lisättiin seuraavat rivit:

```
require-pap
refuse-chap
```

Seuraavaksi avattiin pap-secrets:

```
root@NSN:~# nano /etc/ppp/pap-secrets
```

Tiedostoon lisättiin salasanan tarkastus. Käyttäjän tili syötettäisiin suoraa serveriä käynnistäessä.

```
# client        server  secret                  IP addresses
*                    *  "salasana"              *
```

Tämän jälkeen serveri oli vain asetettava kuuntelemaan verkkoa:

```
root@NSN:~# pppoe-server –I eth3 –s <tili>
```

Palvelin hyväksyi vain tietyllä tilillä ja salasanalla yhteydet sisään, tämä asetus ei olisi ollut pakollinen, mutta se helpotti testausta. Koska jokaiselle yhteydelle oli palvelinta käynnistettäessä annettava tili erikseen, luotiin erillinen skripti ajamaan serveri ylös tarvittaessa. Skriptiin lisättiin myös IP-aliaksien luominen, joita tarvittiin emuloimaan eri FTP–palvelimia.

```
root@NSN:~# nano pppoeserv.sh
```

ja tiedostoon kirjoitettiin rivit:

pppoe-server –I eth3 –s <tili> -s <tili2> -s <tili3>

ifconfig eth2 192.168.1.1

ifconfig eth2:1 192.168.1.2

ifconfig eth2:2 192.168.1.3

Kun tiedosto oli valmis, sille annettiin ajo-oikeudet.

```
root@NSN:~# chmod +x pppoeserv.sh
```

### 1.3 Käytetyt PC -laitteet

Ohjelmistokehitys tapahtui pääosin Dell E5400 kannettavalla tietokoneella. Kannettava on identtinen lopullisessa käyttöympäristössä olevien laitteiden kanssa. Kannettavia käytössäni oli kaksi joista toinen tuli testikäyttöön.

Dell E5400 laitetiedot:

- Käyttöliittymä: LINUX - Fedora 12 32bit
- Prosessori: Intel Core 2 Duo Processor P9500 (2.53GHz)
- Keskusmuisti: 4GB 800MHz DDR2 SDRAM
- Kiintolevy: 160GB Hitachi HDD (7200rpm)
- Näyttö: 14.1" TFT LCD näyttö 1440x 900 (WXGA+, matte)
- Näytönohjain: 256MB nVidia Quadro
- Levy-asema: 8x DVD (+/-R double layer) drive
- Verkko: Intel Wireless WiFi Link 4965AGN (802.11a/g/n) ja 1Gb Lan
- Muuta: Bluetooth 2.1

Ohjelmiston tulisi toimia ensisijaisesti tällä kokoonpanolla, tukien vähintään samaa käyttäjämäärää kuin aikaisempi toteutus.

### 1.4 TM500

Tukiasemien testaukseen käytetään usein testilaitteistoja, koska hyvin usein tukiasemat ovat edellä tuotekehityksessä, eikä asiakaslaitteita saada testikäyttöön tarpeeksi. Vaatimukset testilaitteita kohtaan kasvavat jatkuvasti, niiden täytyy pystyä käsittelemään enemmän dataa, toimimaan usealla taajuudella sekä hallita aina kehittyneempiä modulaatiotekniikoita. Aeroflex on tuonut yhden tällaisen testilaitteen markkinoille.

TM500 LTE–MultiUE testimobiilin ominaisuuksia:

- Yksityiskohtainen asiakaslaitehallinta
- Yksityiskohtaiset ja kattavat tiedonkeräysmahdollisuudet.
- FDD & TDD LTE testimobiili – yhteensopivuus 3GPP:n standardeilla.
- Kykenee 300 Mbp/s downlink ja 75Mbp/s uplink tiedonsiirtonopeuksiin.
- Layer 1 ja 2 Mittaus ja statistiikka.
- Sisäinen ja protokolla ohjattu viestintä.
- Tukee layer1 ja layer 2 analyysi –moodia.
- tukee layer 3:a ja NAS(Non-access stratum):ia

Testilaiteella voidaan emuloida 32 yhtäaikaista asiakaslaitetta, taikka useampaa mikäli halutaan vain tavoitella maksimi kapasiteettia. Kun yhden asiakaslaitteen testaus on onnistuneesti suoritettu loppuun, voidaan lisätä uusia laitteita verkkoon yksitellen. Näin voidaan helposti havaita missä vaiheessa ongelmia alkaa ilmetä. Tässä vaiheessa laitteen tuomat ominaisuudet helpottavat tukiaseman toiminnan seurausta ja debuggausta. Jokaista asiakaslaitetta voidaan testin vaatimusten mukaan muokata. Ongelmien korjausta varten TM500 pystyy tallentamaan ja esittämään hyvin suuren määrän vikadiagnostiikkaa. Käyttäjä pystyy itse valitsemaan mitä tietoa tallennetaan ja esitetään.

## 2. KÄYTTÖTAPAUSKUVAUKSET

| KT 1 : TM500 Ohjaustyökalu - Asetuksien syöttäminen | |
|---|---|
| Yleiskuvaus: | Käyttäjä asettaa tarvittavat asetukset TM500 ohjaustyökaluun. |
| Laatija: | Jaakko Reijasalo |
| Käyttäjäroolit: | Testaaja on ainut järjestelmän käyttäjä |
| Esitiedot: | Käyttäjä on käynnistänyt ohjelman superuser oikeuksilla. |
| Käyttötapauskuvaus: | |
| T1 | Käyttäjä valitsee valikosta ”settings” |
| T2 | Käyttäjälle avataan asetukset dialogi, josta on valittuna välilehti ”settings” |
| T3 | Käyttäjä syöttää tiedot: Ethernet sovitin, TM500 id, TM500 käyttäjätili, ftp upload ja download tiedostot ja hakemistot |
| T4 | Käyttäjä hyväksyy tiedot painamalla OK |
| Poikkeukset: | |
| P4 | Käyttäjä on jättänyt kenttiä tyhjäksi. Käyttäjälle annetaan virheilmoitus. |
| Lopputulos: | |
| Käyttäjä on syöttänyt tiedot onnistuneesti ja voi edetä testaukseen | |
| Vaatimukset: | |
| V1 | |
| | |
| Muuta: Asetukset ovat laite/palvelin riippuvaisia, joten niitä ei voida tarkistaa. | |

| KT 2 : TM500 Ohjaustyökalu – FTP palvelimen tiedot | |
|---|---|
| Yleiskuvaus: | Käyttäjä lisää FTP –palvelimen tiedot asetuksiin |
| Laatija: | Jaakko Reijasalo |
| Käyttäjäroolit: | Testaaja on ainut järjestelmän käyttäjä |
| Esitiedot: | Käyttäjä on käynnistänyt ohjelman. |
| Käyttötapauskuvaus: | |
| T1 | Käyttäjä valitsee valikosta ”ftp servers” |
| T2 | Käyttäjälle avataan asetukset dialogi, josta on valittuna välilehti ”ftp servers” |
| T3 | Käyttäjä kirjoittaa serverin IP osoitteen kenttään |
| T4 | Käyttäjä painaa ”add server” -nappia |
| T5 | Serveri lisätään listaan. |
| T6 | Käyttäjä valitsee painaa OK |
| Poikkeukset: | |
| P4 | Käyttäjä on jättänyt IP kentän tyhjäksi. Käyttäjälle annetaan virheilmoitus. |
| Lopputulos: | |
| Käyttäjä on syöttänyt tiedot onnistuneesti ja voi edetä testaukseen | |
| Vaatimukset: | |
| V1 | |
| | |
| Muuta: Asetukset ovat laite/palvelin riippuvaisia, joten niitä ei voida tarkistaa. | |

| KT 3 : TM500 Ohjaustyökalu – Yhteyden lisäys | | |
|---|---|---|
| Yleiskuvaus: | | Käyttäjä lisää neljä yhteyttä. |
| Laatija: | | Jaakko Reijasalo |
| Käyttäjäroolit: | | Testaaja on ainut järjestelmän käyttäjä |
| Esitiedot: | | Käyttäjä on käynnistänyt ohjelman sekä asettanut vähintään 4 FTP – palvelimen osoitetta. |
| Käyttötapauskuvaus: | | |
| T1 | Käyttäjä painaa käyttöliittymässä ”lisää yhteydet”(+) nappia. | |
| T2 | Neljä yhteyttä luodaan käyttäjän asetuksilla. | |
| T3 | | |
| Poikkeukset: | | |
| P1 | Käyttäjä ei ole lisännyt tarpeeksi ftp – palvelimia asetuksiin. Yhteyksiä ei luoda. | |
| Lopputulos: | | |
| Käyttäjä on luonut yhteydet, jotka voidaan seuraavaksi yhdistää. | | |
| Vaatimukset: | | |
| V1 | | |
| | | |
| Muuta: Asetuksia ei voida tarkastaa, yhteydet luodaan silti. Toiminta varmistuu vasta kun yhteyksiä yritetään yhdistää. | | |

| KT 4 : TM500 Ohjaustyökalu – Yhdistäminen TM500 laitteeseen | |
|---|---|
| Yleiskuvaus: | Käyttäjä yrittää yhdistää ohjelmalla TM500 laitteeseen käyttäen PPP –protokollaa. |
| Laatija: | Jaakko Reijasalo |
| Käyttäjäroolit: | Testaaja on ainut järjestelmän käyttäjä |
| Esitiedot: | Käyttäjä on käynnistänyt ohjelman sekä asettanut TM500 asetukset sekä vähintään yhden FTP –serverin ip:n. Sekä lisännyt yhden yhteyden käyttöliittymään. |
| Käyttötapauskuvaus: | |
| T1 | Käyttäjä valitsee yhteyden jonka haluaa yhdistää ja painaa oikeaa hiiren-nappia. |
| T2 | Käyttäjälle avataan valikko josta valitaan ”Connect” |
| T3 | Ohjelma käynnistää prosessin ifup käyttäjän syöttämillä asetuksilla |
| T4 | Ohjelma käynnistää prosessin route käyttäjän syöttämillä asetuksilla |
| T5 | Ohjelma käynnistää prosessin ping käyttäjän syöttämillä asetuksilla |
| T6 | Ohjelma alkaa näyttää vasteaikaa yhteyden tiedoissa. |
| T7 | |
| Poikkeukset: | |
| P3-4-5 | Käyttäjä on syöttänyt väärät tiedot, ohjelma näyttää virhetilan yhteyden tiedoissa |
| Lopputulos: | |
| Yhteys on muodostunut ja sitä pingataan. Vasteaika tulostuu yhteyden tietoihin. | |
| Vaatimukset: | |
| V1 | |
| | |
| Muuta: . | |

| KT 5 : TM500 Ohjaustyökalu – FTP Download | |
|---|---|
| Yleiskuvaus: | Käyttäjä lataa tiedoston FTP –palvelimelta. |
| Laatija: | Jaakko Reijasalo |
| Käyttäjäroolit: | Testaaja on ainut järjestelmän käyttäjä |
| Esitiedot: | Käyttäjä on käynnistänyt ohjelman sekä asettanut TM500 asetukset sekä vähintään yhden FTP –serverin ip:n. Sekä lisännyt yhden yhteyden käyttöliittymään, sekä yhdistänyt sen onnistuneesti EPC:n ja luonut reitin FTP –serverin IP osoitteeseen. |

| Käyttötapauskuvaus: | |
|---|---|
| T1 | Käyttäjä valitsee yhdistetyn yhteyden, jonka kohdalla painaa oikeaa hiiren-nappia |
| T2 | Käyttäjälle avataan lista, josta valitaan ”Download” |
| T3 | Ohjelma yhdistää FTP serverille, käyttäen asetuksia jotka käyttäjä on syöttänyt |
| T4 | Serveri hyväksyy pyynnön |
| T5 | Tiedosto siirretään käyttäjän asetuksissa määrittelemään hakemistoon |
| T6 | Käyttäjälle esitetään tiedonsiirtonopeus käyttöliittymässä ( yhteyskohtainen) |
| T7 | Käyttäjälle esitetään summattuna kaikki tiedonsiirrot käyttöliittymässä |
| T8 | Kun tiedosto on siirretty aloitetaan uusi, kunnes käyttäjä keskeyttää prosessin. |

| Poikkeukset: | |
|---|---|
| P3-4-5 | Käyttäjä on syöttänyt väärät tiedot, ohjelma näyttää virhetilan yhteyden tiedoissa |

| Lopputulos: |
|---|
| Yhteys on muodostunut ja sitä pingataan. Vasteaika tulostuu yhteyden tietoihin. |

| Vaatimukset: | |
|---|---|
| V1 | |

| |
|---|
| Muuta: . |

## 3. TESTITAPAUKSET

Testattuja tilanteita sekä niiden tuloksia.

| Testitilanne - Yhteyden lisäämine ja poistaminen | |
|---|---|
| Laatija: | Jaakko Reijasalo |
| Esitiedot: | Käyttöliittymä avattuna, ohjelma käynnistetty superuser oikeuksilla. |
| Testikuvaus: | |
| Lisätään yhteyksiä sekä poistetaan niitä käyttöliittymän kautta. | |
| Haluttu tulos: | |
| Jokainen lisätty yhteys tulee näkyviin listassa sekä poistuu siitä. | |
| tulos: | |
| Yhteyden lisäys ja poisto toimii hyvin. | |

| Testitilanne – Yhteyden käynnistäminen | |
|---|---|
| Laatija: | Jaakko Reijasalo |
| Esitiedot: | Käyttöliittymä avattuna ja ohjelma käynnistetty superuser oikeuksilla. vähintään yksi yhteys lisättynä. Palvelimet ovat toiminnassa. |
| Testikuvaus: | |
| Yhteyden muodostaminen PPP –serverille, reitin luominen Ftp-palvelimelle. Yhteyden tilatietojen ilmoittaminen. | |
| Haluttu tulos: | |
| Yhteyden muodostuminen tapahtuu onnistuneesti, ja yhteyden eri tilojen kulusta ilmoitetaan käyttäjälle siinä järjestyksessä kuin ne etenevät. | |
| tulos: | |
| Yhteyden muodostuminen toimi halutulla tavalla, yhteyden muodostamisesta ei tullut<br>halutulla tavalla viestejä, koska osa operaatioista kulkee ohi liian nopeasti. Viivettä tietojen näyttämiseen lisätty. | |

| Testitilanne – Yhteyden käynnistäminen, epäonnistuminen | |
|---|---|
| Laatija: | Jaakko Reijasalo |
| Esitiedot: | Käyttöliittymä avattuna ja ohjelma käynnistetty superuser oikeuksilla. vähintään yksi yhteys lisättynä. Palvelimet ovat poissa käytöstä. |
| Testikuvaus: | |
| Yhteyden muodostamisen yrittäminen ilman toimivaa yhteyttä. Yhteyden tilatietojen ilmoittaminen. | |
| Haluttu tulos: | |
| Yhteyden luominen ei onnistu ja käyttäjälle ilmoitetaan tapahtumista. Yhteyden luominen<br>lakkaa ensimmäiseen epäonnistuneeseen yritykseen. | |
| tulos: | |
| Yhteyden tiloista ilmoitetaan, toiminta ei lakkaa ensimmäisen vaiheen epäonnistuttua.<br>Yhteyden tilasta ilmoitetaan käyttäjälle oikein. | |

| Testitilanne – FTP palvelimelta lataus | |
|---|---|
| Laatija: | Jaakko Reijasalo |
| Esitiedot: | Käyttöliittymä avattuna, ohjelma käynnistetty superuser oikeuksilla. vähintään yksi yhteys lisättynä. Yhteys saatu palvelimelle. |
| Testikuvaus: | |
| Käynnistetään yksi tiedonsiirto alakaistaan. | |
| Haluttu tulos: | |
| Tiedonsiirto käynnistyy ja välittää käyttäjälle tiedonsiirtonopeuden. | |
| tulos: | |
| Lopputulos oli halutun tuloksen mukainen. | |

| Testitilanne – FTP rasitustesti | |
|---|---|
| Laatija: | Jaakko Reijasalo |
| Esitiedot: | Käyttöliittymä avattuna, ohjelma käynnistetty superuser oikeuksilla.16 yhteyttä lisättynä. Yhteys saatu palvelimelle. |
| Testikuvaus: | |
| Käynnistetään 16 siirtoa alakaistaan, jokainen aloittaa uuden siirron kunnes käyttäjä keskeyttää toiminnan. Tiedosto on pieni jolloin yhteyksiä avautuu tiheään tahtiin. Testiä ajetaan yli tunti, samalla seurataan prosessori sekä muistikuormaa, joka dokumentoidaan. | |
| Haluttu tulos: | |
| Kuorma ei saa olla liian suuri, joka voisi vaikuttaa käyttöliittymän tai muiden ohjelmien ja<br>prosessien toimintaan, muistin käytön oltava kohtuullista. | |
| tulos: | |
| Lopputulos oli tyydyttävä. Prosessorikuormat sekä muistinkulutus näkyvät dokumentaatiosta. Testiä ajettiin noin 1-2 tuntia. Ohjelma ei kaatunut ja muistia ei vuotanut, muistinkäyttö oli 20-30Mb normaalia käyttöä isompi. | |
| Kerätty tieto: | |
| | |

| Testitilanne – FTP rasitustesti | |
|---|---|
| Laatija: | Jaakko Reijasalo |
| Esitiedot: | Käyttöliittymä avattuna, ohjelma käynnistetty superuser oikeuksilla.16 yhteyttä lisättynä. Yhteys saatu palvelimelle. |
| Testikuvaus: | |
| Käynnistetään 16 siirtoa alakaistaan joka keskeytetään 3s kuluttua aloituksesta käyttäjän stop transfer –komennolla. Tällä tutkitaan onko keskeytystoimi joka toteutettiin Qt:n toimintatapojen vastaisesti muistia vuotava. | |
| Haluttu tulos: | |
| Kuorma ei saa olla liian suuri, joka voisi vaikuttaa käyttöliittymän tai muiden ohjelmien ja<br>prosessien toimintaan, muistia ei saa vuotaa. | |
| tulos: | |
| Lopputulos oli tyydyttävä. Prosessorikuormat sekä muistinkulutus näkyvät dokumentaatiosta. Testiä ajettiin noin 1 tunti. Ohjelma ei kaatunut ja muistia ei vuotanut, muistinkäyttö oli 20-30Mb normaalia käyttöä isompi. | |

Kerätty tieto:

**Testin alussa:**

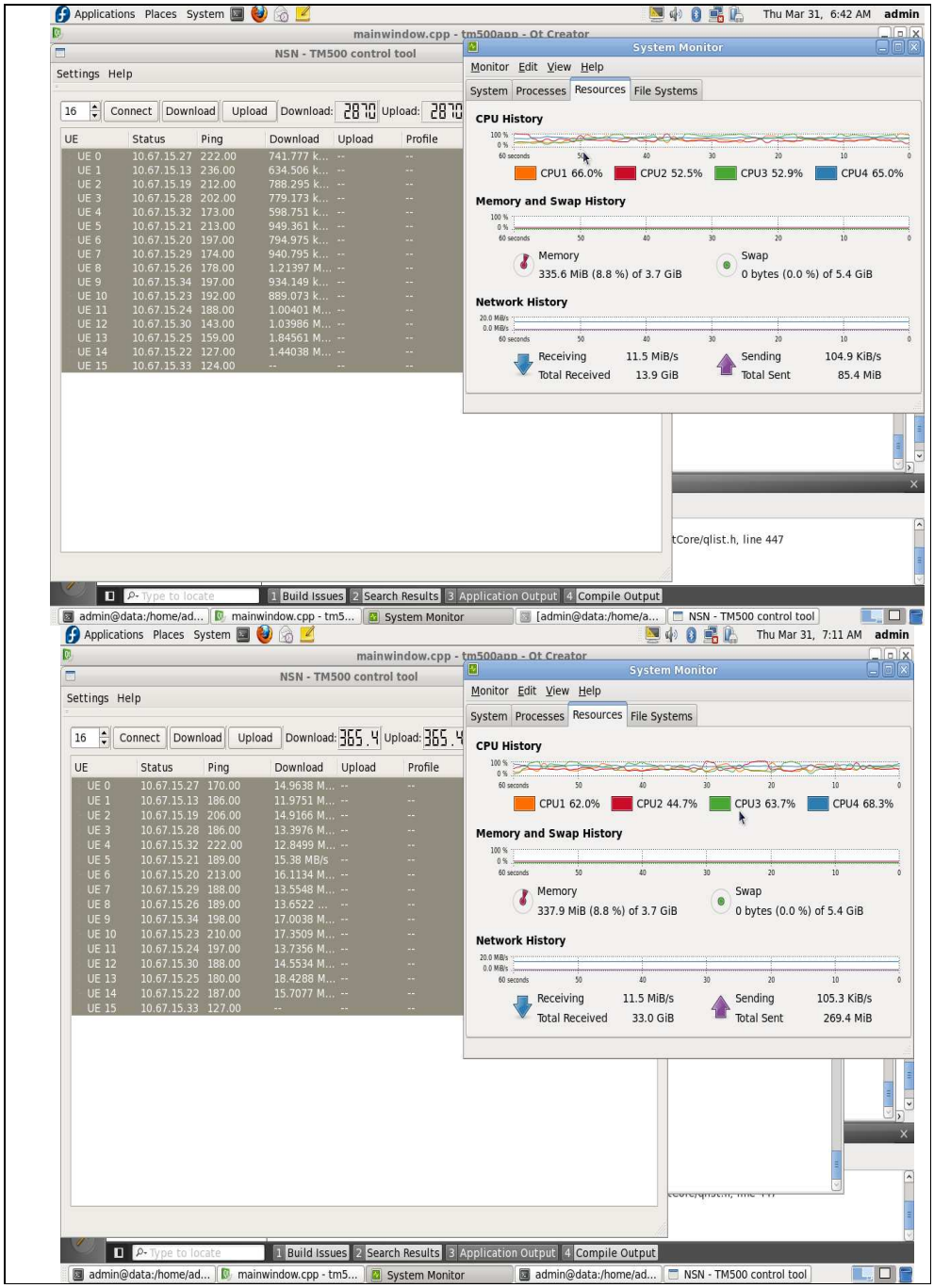
```
top - 06:40:22 up  8:52, 3 users,  load average: 1.13, 1.42, 1.10
Tasks:   1 total, 1 running, 0 sleeping, 0 stopped, 0 zombie
Cpu(s): 32.7%us, 26.1%sy,  0.0%ni, 34.2%id,  0.0%wa,  0.2%hi,
6.7%si,  0.0%st
Mem:   3921104k total, 799052k used, 3122052k free, 158336k buffers
Swap:  5652472k total, 0k used,  5652472k free,   308436k cached

PID USER   PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
2402 root   20   0  227m  23m  11m R 88.9  0.6  15:44.20 TM500app
```

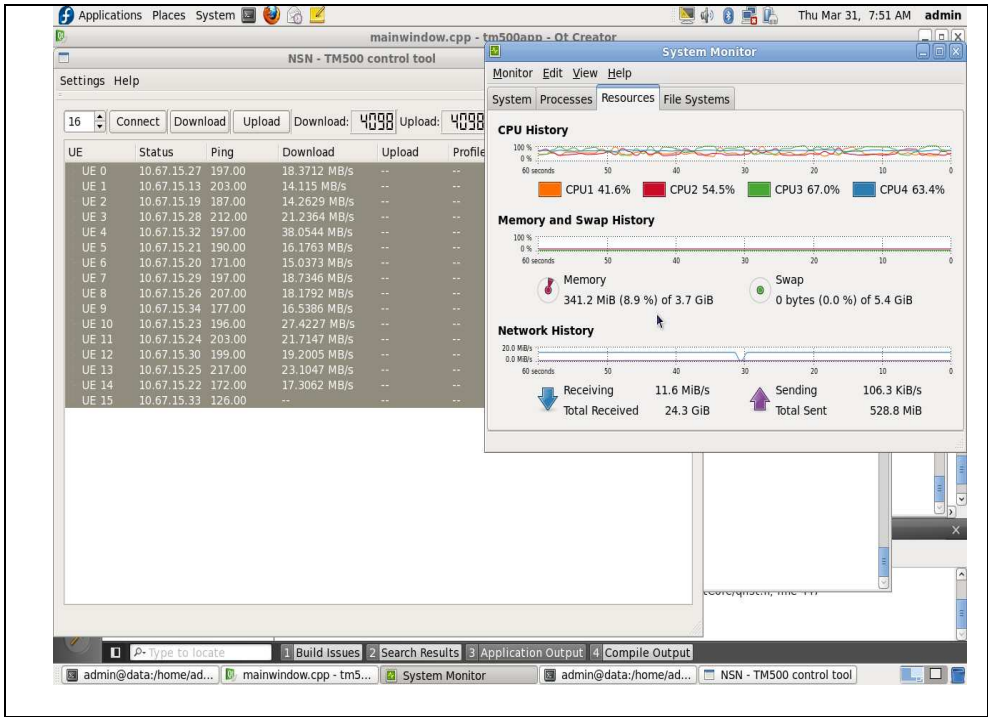
**30min kuluttua:**

```
op - 07:10:09 up  9:22,  3 users,  load average: 0.96, 1.28, 1.34
Tasks:   1 total,   1 running,   0 sleeping,   0 stopped,   0 zombie
Cpu(s): 32.9%us, 25.7%sy,  0.0%ni, 34.1%id,  0.0%wa,  0.3%hi,
7.0%si,  0.0%st
Mem:   3921104k total, 820892k used, 3100212k free, 158400k buffers
Swap:  5652472k total, 0k used,  5652472k free,   312600k cached

PID USER    PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
2402 root   20   0  239m  34m  11m R 90.2  0.9  42:10.04 TM500app
```

**1t 10min kuluttua:**

```
top - 07:51:42 up 10:03,  3 users,  load average: 1.16, 1.38, 1.42
Tasks: 1 total,   1 running,   0 sleeping,   0 stopped,   0 zombie
Cpu(s): 32.7%us, 25.2%sy,  0.0%ni, 35.4%id,  0.3%wa,  0.2%hi,
6.2%si,  0.0%st
Mem:   3921104k total, 822396k used, 3098708k free, 158404k buffers
Swap:  5652472k total, 0k used, 5652472k free, 312832k cached

PID USER  PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
2402 root  20   0  239m  34m  11m R 91.6  0.9  79:32.21 TM500app
```


Applications Places System
Thu Mar 31, 6:42 AM admin
mainwindow.cpp - tm500app - Qt Creator
NSN - TM500 control tool
Settings Help
16 Connect Download Upload Download: 2870 Upload: 2870
UE Status Ping Download Upload Profile
UE 0 10.67.15.27 222.00 741.777 k... -- --
UE 1 10.67.15.13 236.00 634.506 k... -- --
UE 2 10.67.15.19 212.00 788.295 k... -- --
UE 3 10.67.15.28 202.00 779.173 k... -- --
UE 4 10.67.15.32 173.00 598.751 k... -- --
UE 5 10.67.15.21 213.00 949.361 k... -- --
UE 6 10.67.15.20 197.00 794.975 k... -- --
UE 7 10.67.15.29 174.00 940.795 k... -- --
UE 8 10.67.15.26 178.00 1.21397 M... -- --
UE 9 10.67.15.34 197.00 934.149 k... -- --
UE 10 10.67.15.23 192.00 889.073 k... -- --
UE 11 10.67.15.24 188.00 1.00401 M... -- --
UE 12 10.67.15.30 143.00 1.03986 M... -- --
UE 13 10.67.15.25 159.00 1.84561 M... -- --
UE 14 10.67.15.22 127.00 1.44038 M... -- --
UE 15 10.67.15.33 124.00 -- --
System Monitor
Monitor Edit View Help
System Processes Resources File Systems
CPU History
CPU1 66.0% CPU2 52.5% CPU3 52.9% CPU4 65.0%
Memory and Swap History
Memory
335.6 MiB (8.8 %) of 3.7 GiB
Swap
0 bytes (0.0 %) of 5.4 GiB
Network History
Receiving 11.5 MiB/s
Total Received 13.9 GiB
Sending 104.9 KiB/s
Total Sent 85.4 MiB
tCore/qlist.h, line 447
Type to locate
1 Build Issues 2 Search Results 3 Application Output 4 Compile Output
admin@data:/home/ad... mainwindow.cpp - tm5... System Monitor [admin@data:/home/a... NSN - TM500 control tool
Applications Places System
Thu Mar 31, 7:11 AM admin
mainwindow.cpp - tm500app - Qt Creator
NSN - TM500 control tool
Settings Help
16 Connect Download Upload Download: 365.4 Upload: 365.4
UE Status Ping Download Upload Profile
UE 0 10.67.15.27 170.00 14.9638 M... -- --
UE 1 10.67.15.13 186.00 11.9751 M... -- --
UE 2 10.67.15.19 206.00 14.9166 M... -- --
UE 3 10.67.15.28 186.00 13.3976 M... -- --
UE 4 10.67.15.32 222.00 12.8499 M... -- --
UE 5 10.67.15.21 189.00 15.38 MB/s -- --
UE 6 10.67.15.20 213.00 16.1134 M... -- --
UE 7 10.67.15.29 188.00 13.5548 M... -- --
UE 8 10.67.15.26 189.00 13.6522 ... -- --
UE 9 10.67.15.34 198.00 17.0038 M... -- --
UE 10 10.67.15.23 210.00 17.3509 M... -- --
UE 11 10.67.15.24 197.00 13.7356 M... -- --
UE 12 10.67.15.30 188.00 14.5534 M... -- --
UE 13 10.67.15.25 180.00 18.4288 M... -- --
UE 14 10.67.15.22 187.00 15.7077 M... -- --
UE 15 10.67.15.33 127.00 -- --
System Monitor
Monitor Edit View Help
System Processes Resources File Systems
CPU History
CPU1 62.0% CPU2 44.7% CPU3 63.7% CPU4 68.3%
Memory and Swap History
Memory
337.9 MiB (8.8 %) of 3.7 GiB
Swap
0 bytes (0.0 %) of 5.4 GiB
Network History
Receiving 11.5 MiB/s
Total Received 33.0 GiB
Sending 105.3 KiB/s
Total Sent 269.4 MiB
Type to locate
1 Build Issues 2 Search Results 3 Application Output 4 Compile Output
admin@data:/home/ad... mainwindow.cpp - tm5... System Monitor admin@data:/home/ad... NSN - TM500 control tool


Applications Places System
Thu Mar 31, 7:51 AM admin
mainwindow.cpp - tm500app - Qt Creator
NSN - TM500 control tool
Settings Help
16 Connect Download Upload Download: Upload:
UE Status Ping Download Upload Profile
UE 0 10.67.15.27 197.00 18.3712 MB/s -- --
UE 1 10.67.15.13 203.00 14.115 MB/s -- --
UE 2 10.67.15.19 187.00 14.2629 MB/s -- --
UE 3 10.67.15.28 212.00 21.2364 MB/s -- --
UE 4 10.67.15.32 197.00 38.0544 MB/s -- --
UE 5 10.67.15.21 190.00 16.1763 MB/s -- --
UE 6 10.67.15.20 171.00 15.0373 MB/s -- --
UE 7 10.67.15.29 197.00 18.7346 MB/s -- --
UE 8 10.67.15.26 207.00 18.1792 MB/s -- --
UE 9 10.67.15.34 177.00 16.5386 MB/s -- --
UE 10 10.67.15.23 196.00 27.4227 MB/s -- --
UE 11 10.67.15.24 203.00 21.7147 MB/s -- --
UE 12 10.67.15.30 199.00 19.2005 MB/s -- --
UE 13 10.67.15.25 217.00 23.1047 MB/s -- --
UE 14 10.67.15.22 172.00 17.3062 MB/s -- --
UE 15 10.67.15.33 126.00 -- -- --
System Monitor
Monitor Edit View Help
System Processes Resources File Systems
CPU History
100 %
0 %
60 seconds 50 40 30 20 10 0
CPU1 41.6% CPU2 54.5% CPU3 67.0% CPU4 63.4%
Memory and Swap History
100 %
0 %
60 seconds 50 40 30 20 10 0
Memory
341.2 MiB (8.9 %) of 3.7 GiB
Swap
0 bytes (0.0 %) of 5.4 GiB
Network History
20.0 MiB/s
0.0 MiB/s
60 seconds 50 40 30 20 10 0
Receiving 11.6 MiB/s
Total Received 24.3 GiB
Sending 106.3 KiB/s
Total Sent 528.8 MiB
Type to locate
1 Build Issues 2 Search Results 3 Application Output 4 Compile Output
admin@data:/home/ad...
mainwindow.cpp - tm5...
System Monitor
admin@data:/home/ad...
NSN - TM500 control tool

# 4 NELJÄNNEN SUKUPOLVEN MATKAPUHELINVERKOT

4G (Generation) on nimensä mukaisesti neljäs sukupolvi matkapuhelin verkko standardissa. 4G tekniikoita ovat tällä hetkellä ainakin LTE ja Mobile-Wimax, joista kumpaakin kehitellään edelleen. Tekniikoiden kuulumisesta 4G–luokkaan on epäselvyyksiä, sillä ne eivät täytä IMT-Advanced standardin ehtoja. Usein niitä kuulee kutsuttavan 3.9G:si. LTE:n seuraaja, LTE-A(Advanced) tulee täyttämään nämä vaatimukset ja on täten aito neljännen sukupolven mobiili-verkko. Standardit määrittelee 3rd Generation Partnership Project, eli 3GPP. 3GGP on usean tietoverkko–yrityksen yhteistyöllä toteutettu organisaatio. 3GPP lajittelee standardit eri julkaisuiksi (Release), joista LTE on kahdeksas.

4G Tekniikassa 3G:n kaistantaajuus tekniikat on korvattu taajuusjakoisilla (OFDMA/SC-FDMA) sekä hyväksi käytetään MIMO antenni-järjestelmää, Dynaamista kanavan määritystä ja kanavakohtaista aikataulutusta. Myös verkon rakenne on muutettu kokonaan piirikytkentäisestä pakettikytkentäiseen.

| Versio | Julkaisu | Ominaisuudet |
|---|---|---|
| Release 98 | 1998 | GSM, AMR, EDGE, GPRS |
| Release 99 | 2000 | Ensimmäinen UMTS 3G verkko, käytti CDMA-Modulaatiota |
| Release 4 (alk. 2000) | 2001 | Muutti runkoverkon kokonaan IP –pojautuvaksi. |
| Release 5 | 2002 | IMS ja HSDPA |
| Release 6 | 2004 | Yhdistetty toimivuus Wlanin kanssa. HSUPA ja MBMS muutokset IMS:än. |
| Release 7 | 2007 | Keskittyi tiputtamaan latensseja, parantamaan QoS:ä ja reaaliaikaisia sovellutuksia kuten VoIP:a. lisäsi myös tuen mobiilimaksuille sekä EDGE-E:lle. |
| Release 8 | 2008 | Ensimmäinen LTE julkaisu, kokonaan pakettikytkentäinen. OFDMA, FDE ja MIMO pohjainen radioverkko. |
| Release 9 | 2009 | SAE muutoksia, Wimax, LTE/UMTS yhteensopivuus. |
| Release 10 | tulossa | LTE-A, joka täyttää IMT-A 4G vaatimukset. Yhteensopiva release 8:n kanssa. |

### 4.1 LTE

LTE ( Long Term Evolution ):n on siis 3GPP release 8. Sen kehitys aloitettiin vuonna 2004, projektissa päätettiin keskittyä asiakkaan ja runkoverkon välisen tekniikan sekä radiotien kehittämiseen. Tavoitteeksi otettiin myös 3-4 kertainen(100Mbp/s) siirtonopeus asiakkaalle release 6 (HSDPA):n verrattuna, ja 2-3 kertainen asiakkaalta verkkoon (50 Mbp/s). 2007 LTE:n, teoreettinen määrittely saatiin valmiiksi ja ensimmäiset tekniset tiedot hyväksyttiin. 2008 lopussa tekniset spesifikaatiot olivat tarpeeksi kattavat tuotteen viemiseksi lopulliseksi tuotteeksi. OFDMA valittiin modulaatiotavaksi downlink liikennöintiin ja SC-FDMA uplink:iin. Datamodulaatio tapoja downlinkissä tulee olemaan QPSK, 16QAM ja 64QAM ja uplinkissä BPSK, QPSK, 8PSK ja 16QAM.

### 4.1.1 LTE tekniset tavoitteet

- Korkea spektritehokkuus:
    - OFDM ( Orthogonal Frequency-Division Multiplexing) , tukiasemalta asiakkaalle tapahtuvassa liikenteessä. Modulaatio kestää hyvin radiotien häiriöitä sekä mahdollistaa useamman käyttäjän ajon samalla taajuudella.
    - SC-FDMA (Single Carrier Frequency-Division Multiplexing) Asiakkaalta tukiasemalle tapahtuvassa liikenteessä. Tämän etuna alhainen PAPR (Peak to Average Power Ratio ).
    - Usean antennin hyväksikäyttö(MIMO).
- Matala verkkoviive:
    - Lyhyt lähetysviive
    - Lyhyt HO (Hand over) viive ja keskeytysaika.
- Tukea usealle taajuuskaistalle:
    - Käytössä 1.4, 3, 5, 10 ja 20 MHz:n taajuuskaistat.
- Yksinkertainen protokolla-arkkitehtuuri:
    - Pohjautuu jaettuihin kanaviin.
    - pakettikytkentäinen, puhelut hoidettu VOIP( voice over ip ) kautta.
- Yksinkertainen verkko-arkkitehtuuri:
    - eNodeB toimii ainoana E-UTRAN(Evolved Universal Terrestial Radio Network) osana.
    - Pienempi määrä RAN ( Radio Access Network ) kytkentöjä, eNodeB kytköksissä suoraa MME/SAE-Gateway:in(S1), tai eNodeB suoraa eNodeB:en ( X2 ).
- Yhteentoimivuus aikaisempien 3GPP julkaisujen (release) kanssa.
- FDD ja TDD samalla radioyhteys teknologialla.
- Tehokas multicast/broadcast
    - Yhden taajuuden verkko OFDMA:lla
- Tuki SON:lle ( Self organizing network )

| | | |
|---|---|---|
| Access Scheme | UL | DFTS-OFDM |
| | DL | OFDMA |
| Bandwidth | | 1.4, 3, 5, 10, 15, 20MHz |
| Minimum TTI | | 1msec |
| Sub-carrier spacing | | 15kHz |
| Cyclic prefix length | Short | 4.7µsec |
| | Long | 16.7µsec |
| Modulation | | QPSK, 16QAM, 64QAM |
| Spatial multiplexing | | Single layer for UL per UE<br>Up to 4 layers for DL per UE<br>MU-MIMO supported for UL and DL |

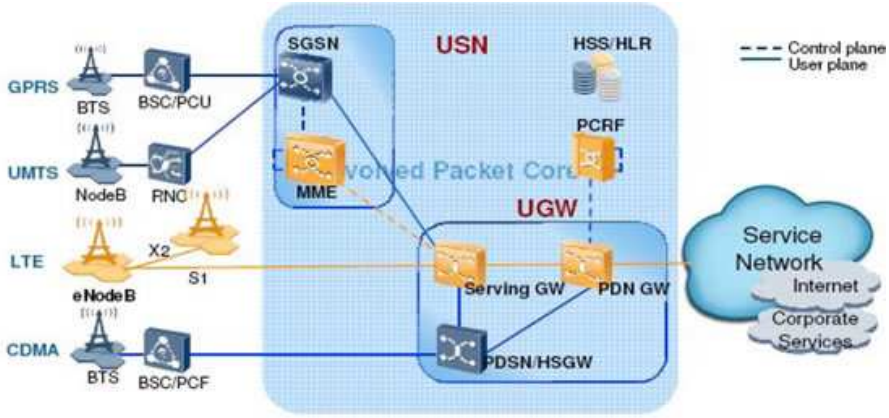
Kuva 1: LTE spesifikaatiot (http://www.3gpp.org/LTE)

| Category | | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| Peak rate Mbps | DL | 10 | 50 | 100 | 150 | 300 |
| | UL | 5 | 25 | 50 | 50 | 75 |
| Capability for physical functionalities | | | | | | |
| RF bandwidth | | 20MHz | | | | |
| Modulation | DL | QPSK, 16QAM, 64QAM | | | | |
| | UL | QPSK, 16QAM | | | | QPSK, 16QAM, 64QAM |
| Multi-antenna | | | | | | |
| 2 Rx diversity | | Assumed in performance requirements. | | | | |
| 2x2 MIMO | | Not supported | Mandatory | | | |
| 4x4 MIMO | | Not supported | | | | Mandatory |

Kuva 2: LTE:n UE spesifikaatiot

### 4.1.2 LTE Verkkorakenne

LTE verkkorakenne koostuu E-UTRAN:ista, sekä SAE/EPC:sta, eli pakettiverkon osasta. Tekstissä on kerrottu yksinkertaistettuna eri elementtien toiminnot.




*Source: 3GPP.org*

Kuva 3: LTE-verkko

**eNodeB**

eNodeB on tukiasema, joka hoitaa radioliikenteen käyttäjälaitteelta verkkoon. Se eroaa aikaisemmista tekniikoista hoitamalla BSC (base station controller) ja RNC (Radio Network Controller) laitteiden ominaisuudet itse. Se pystyy myös olemaan yhteydessä toisiin eNb tukiasemiin, jolloin verkon rakenteesta tulee joustavampi.

**HSS/HLR**

HSS säilyttää tilaajatiedot, se hoitaa myös tunnistuksen sekä sallii asiakkaan verkkoon MME:n kanssa.

**MME ( Mobile Managment Entity)**

MME hoitaa verkon signaloinnin, ja toimii täten keskuspisteenä SAE –verkossa. Kun käyttäjälaite yrittää liittyä verkkoon, MME pyytää HSS:ltä asiakastiedot ja hoitaa SGW:n kanssa vapautuksen verkkoon. HSS:ltä saama asiakastieto käytetään luomaan salausavaimet joilla turvataan NAS:lla käyty hallintatason signalointi. MME:llä hoidetaan myös signalointi vanhempien 2/3G–verkkojen kanssa, jolloin se on yhteydessä SGSN:een (Serving GPRS Support node).

**Serving Gateway (SGW)**

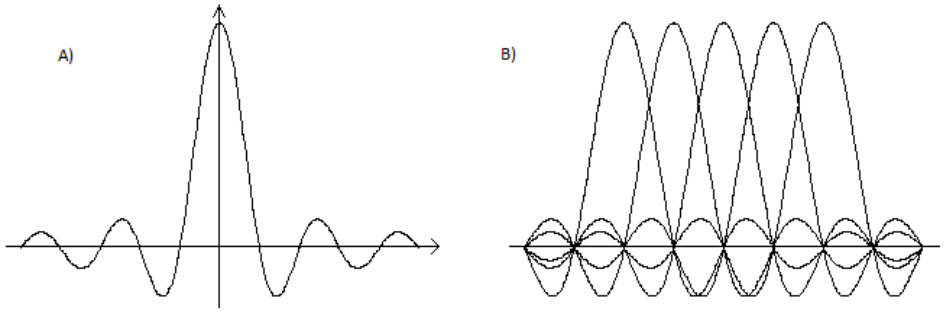
Hoitaa käyttäjätason uudelleen reitityksen ja uplink ja downlin pakettien uudelleen ohjauksen PDN Gateway:n ja radioverkon välillä. Jokainen käyttäjä voi olla yhteydessä vain yhteen SGW:en joka taas voi olla yhteydessä erilaisiin PDN GW:in, riippuen käyttäjälaitteen vaatimuksista.

**Packet Data Network Gateway (PDN GW)**

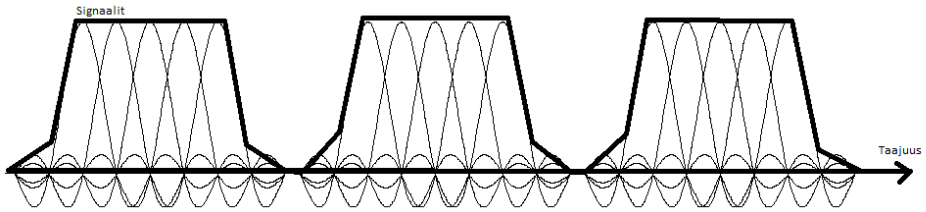
PDN GW huolehtii pakettiyhteyksistä ulkopuolisiin verkkoihin ja antaa käyttäjälaitteelle IP–osoitteen sen liittyessä verkkoon. Muita PDN GW:n tehtäviä on määrittää siirtonopeus sekä toimia linkkinä muille kuin 3GPP:n teknologialle.

## 5 OFDM JA OFDMA JA SC-FDMA

OFDM (Orthogonal Frequency-Division Multiplexing) on siirtotekniikka joka käyttää suurta määrää lähekkäin asetettuja kantoaaltoja, jotka eivät häiritse toisiaan siirron aikana. Tämä saavutetaan tekemällä signaaleista ortogonaalisia, eli kohtisuoria toisiinsa nähden. Jokaisella kaistalla voidaan siirtää modulaatiotavasta riippuen tietty määrä bittejä. Kanavia voi olla yhtä aikaa tuhansia.




Kuva 4: OFDM apukantoaallot: A) Yksittäinen apukantoaalto B) Spektri useista apukantoaalloista.




kuva 5: OFDM signaalit, useita lähekkäin asetettuja apukantoaaltoja lähekkäin aseteltuina.

| | CDMA | OFDMA |
|---|---|---|
| Siirtokaista | Käyttää koko kaistan | Riippuva – mahdollisuus käyttää koko kaista |
| taajuuskohtainen aikataulutus | Ei mahdollista | Yksi tärkeimmistä ominaisuuksista, tarvitsee kuitenkin jatkuvaa tietoa kaistan tilasta vastaanottajalta. |
| Symbolin aika | Hyvin lyhyt, vastakohtainen systeemin kaistalle. | Hyvin pitkä, määrittyy kantoaaltojen tilanvarauksesta ja ei ole riippuvainen systeemin kaistasta. |
| Taajuudenkorjaus | Vaikeaa yli 5 MHz | Helppoa mille tahansa taajuuskaistalle |
| Monikuuluvuuden sietokyky | Vaikeaa yli 5 Mhz | Sietää hyvin monikuuluvuutta |
| MIMO mahdollisuus | Vaatii paljon laskentaa, signaalin ollessa aikatasossa. | Soveltuu hyvin MIMO:n, sillä signaali esitellään taajuustasossa. |
| Taajuusalueen vääristymisen sietokyky | Vääristyminen jakautuu koko kanavalle tasaisesti | Herkkä mahdollisille virheille, varsinkin solun reunoilla jossa signaalien vaimeneminen on vaikeaa ottaa huomioon. |

OFDMA (Orthogonal Frequency-Division Multiple Access) on variaatio OFDM tekniikasta, se mahdollistaa aikatasossa apukantoaaltojen jaon käyttäjien kesken. OFDMA myös sietää kaistalla tapahtuvia virheitä sekä vaimenemista hieman paremmin, sekä kuluttaa vähemmän tehoa.

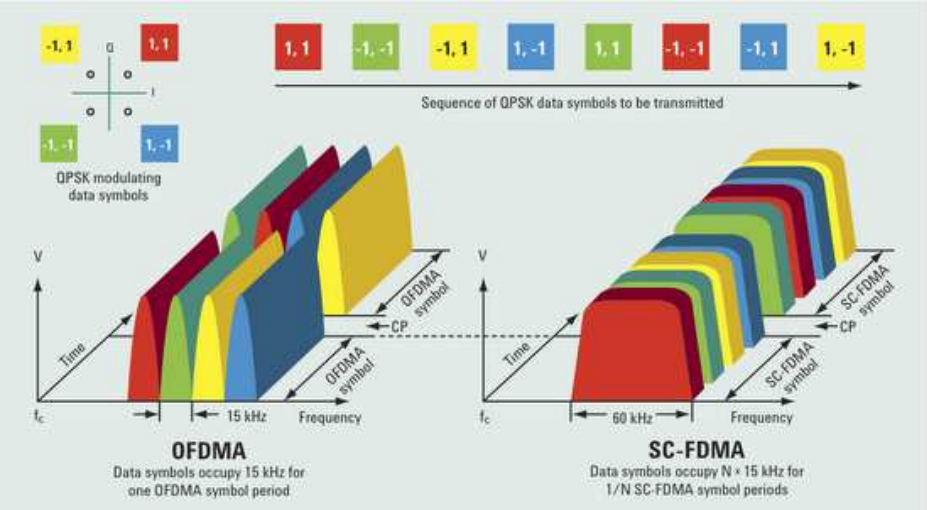
OFDM/A on käytössä ainakin seuraavissa:

- Digi-tv: DVB-T
- ADSL
- Wlan: 802.11a,11g,11n
- Wimax

Valinta OFDMA:n käyttöön LTE tekniikassa johtui sen hyvästä kyvystä sietää virheitä. Tekniikasta oli myös jo aikaisemmin saatu hyviä käyttökokemuksia Wimax sekä Wi-F radioverkoissa. OFDMA on myös modulointitekniikka joka kykenee hyvin suuren datamäärän välittämiseen, mikä on LTE:n tärkeimpiä vaatimuksia. Kaiken lisäksi OFDMA tukee sekä aika ja taajuusjakoista dupleksisuutta.

SC-FDMA (Single-Carrier Frequency Division Multiple Access) modulaatiotapa valittiin uplink tiedonsiirtoon. Se on hyvin samankaltainen tekniikka kuin OFDMA. Erona on yksi diskreetti Fourier muunnos enemmän sekä pelkästään yksi kantoaalto, usean apukantoaallon sijaan.

Muotoiltu: suomi




kuva 6: OFDMA ja SC-FDMA vertailussa

Kuvassa 6. on vierekkäin aseteltuna OFDMA ja SC-FDMA siirtotapahtuma QPSK – modulaatiota käyttäen, joten pelkästään jokaisen apukantoaallon vaihetta muutetaan ja lähetysteho pysyy samana. Kuvassa vasemmalla OFDM siirrossa on selvyyden vuoksi näytetty vain 4 apukantoaaltoa, joista jokainen varaa 15kHz tilan. Yhdessä nämä luovat yhden symbolin. Symboleiden välissä on cyclic prefix, joka toimii välimerkkinä (”guard interval”).

Kuvasta poiketen cyclic prefix ei ole tyhjä aukko, vaan se on täytetty seuraavan symbolin lopulla. Tämä takaa lähetystehon jatkuvuuden ja vaihe-eron symbolin lopussa, sekä estämään monikuuluvuutta(Multipath). SC-FDMA joka on kuvassa oikealla, siirtää QPSK modulaatiotapahtumat sarjassa, joista jokainen varaa kuvassa esitetyn 60kHz kaistan. Data symbolit aikatasossa muutetaan taajuustasoon käyttämällä diskriittiä Fourier muunnosta, kun ne ovat taajuustasossa, ne sijoitetaan kanavalle haluttuun kohtaan ja muutetaan jälleen käänteistä Fourier taajuusmuunnosta käyttäen takaisin aikatasoon. Tämän jälkeen lisätään Cyclic prefix.

# 5 QT

Qt:n kehitys alkoi kahden norjalaisen, Haavard Nordin ja Eirik Chambe-Engin toimesta vuonna 1991. Kolmen vuoden kuluttua Qt yhdistyi Quasar Technologiesin kanssa ja tällöin yrityksen nimeksi tuli Troll tech, joka yksinkertaistettiin myöhemmin Trolltechiksi. Kehitysympäristön nimeksi jäi Qt. Qt tuki aluksi pelkästään kahta eri alustaa, Qt/X11 sekä Qt/Windows. Vuoden 2001 lopulla julkaistiin versio 3.0 joka toi tuen myös Mac OS X – alustalle. Seuraava iso muutos tapahtui kesäkuussa 2008 kun Nokia osti Trolltech ASA:n.

Oston jälkeen Nokia alkoi rakentaa kehitysympäristöä tukemaan omia tuotteitaan, Symbian OS:ia sekä silloin kehitteillä olevaa Maemoa. Vuoden 2011 alussa Nokia ilmoitti vähentävänsä merkittävästi Qt:n kehittämiseen käyttämiään resursseja sen keskittyessä pelkästään Windows Phone 7 alustaan, tämän ei silti uskota vaikuttavan merkittävästi Qt:n kehitykseen, sen ollessa vapaan lähdekoodin alusta.


Kuva 1: Qt SDK

Qt kehitysympäristö tukee useaa alustaa sekä kieltä. Tällä hetkellä tuettuja alustoja on Linux/X11, Mac OS X, Windows, sulautettu linux, Symbian sekä Maemo/MeeGo ja näiden lisäksi useita muita Trolltechin ulkopuolella kehitettyjä portteja. Kieliä Qt tukee useita, esimerkkeinä C++, Python, Java, C# & .NET. Qt:lla kehittäminen hoidetaan pääsääntöisesti Qt Creator:lla joka on Qt:n oma IDE (Integrated Developement Environment), mutta kehittäminen onnistuu myös Visual studiolla ja Eclipse:llä.

Qt Creator ympäristö on helposti omaksuttava ja siinä on paljon ominaisuuksia, kuten integroitu GUI työkalu, Versiohallinnan tuki, useiden projekti ja käännösmäärittelyjen mahdollisuus, GDB ja CDB debuggerit ja tuki sekä Desktop että mobiilisovelluksille, joista jälkimmäistä voidaan ajaa IDE:n omassa simulaattorissa. Qt Creator tukee kielinä c++:a ja javascriptiä. Lisäksi Trolltech on luonut paljon oheisohjelmia hoitamaan erilaisia tehtäviä kuten Qt assistant,Qt Desinger sekä Qt Linguest.

Qt Assistant on apuohjelma, jonka avulla suunnittelija pääsee nopeasti käsiksi muiden ohjelmien käyttöohjeisiin sekä Qt:n referenssimanuaaliin. QtAssistant on HTML pohjainen ja sen voi sulauttaa toimimaan oman ohjelman käyttöoppaana. Qt Desinger on apuohjelma jolla voidaan luoda käyttöliittymä pohjia, joita voidaan käyttää ohjelmissa. Qt Linguist helpottaa ohjelmistojen kielipakettien laatimista. Käännetyksi halutut tekstit kirjoitetaan käännösfunktion sisään ja Linguist auttaa niiden käännösprosessissa.

Useamman alustan tuki Qt:ssa on hoidettu Qmake:lla, joka automatisoi makefile:n luonnin. Qt käyttää Meta-objekti compiler:a käännösoperaatiossa, joka tulkitsee käyttäjän koodista tietyt makrot ja luo niistä meta-informaatiota. Tätä metainformaatiota käytetään mahdollistamaan Qt:n erikoisominaisuuksia kuten:

- Signaalit ja slotit: Qt:n tehokas tapa hoitaa GUI ohjelmassa elementtien ja objektien välinen tiedonvälitys.
- Itsehavaitsevaisuutta(introspection): Objektien tyyppi voidaan selvittää ajettaessa koodia
- Asynkroniset funktiokutsut.

Qt on lisenssin alainen ohjelmisto. Aluksi käytössä oli kaksi eri lisenssiä, lähdekoodin saatavuutta säännöstelevä GPL lisenssi ei kaupalliseen käyttöön sekä kaupallinen lisenssi suljetun koodin sovelluksia valmistaville yrityksille. Version 4.5 julkaisun myötä Nokia muutti lisensointia LGPL(Lesser General Public License) joka mahdollistaa suljetun lähdekoodin julkaisemisen kaupallisesti ilman erillistä maksullista lisenssiä. Tunnetuimpia Qt:lla ohjelmoituja sovelluksia ovat Skype, Google Earth, VirtualBox, VLC media player ja KDE SC.

## 6 NOKIA SIEMENS NETWORKS

Nokia Siemens Networks B.V on Nokia Oyj:n ja Siemens AG:n jaetussa omistuksessa oleva yhtiö joka perustettiin huhtikuussa 2007. Nokia Siemens Networks Oy on Nokia Siemens Networks B.V:n tytäryhtiö. Se muodostui Nokian Network Business Groupin ja Siemensin COM – divisioonan yhdistyksellä. Yrityksen liiketaloudellinen johtaja on tällä hetkellä Intialainen Rajeev Suri ja hallituksen puheenjohtajana entinen Nokian toimitusjohtaja Olli-Pekka Kallasvuo. Nokia Siemens Networks Oy:n pääkonttori sijaitsee Espoossa ja Euroopan konttori Münchenissä, Saksassa. NSN oy valmistaa sekä kehittää tietoliikenneverkoissa käytettäviä laitteita. Esimerkkeinä 3G, WIMAX sekä LTE tietoverkkolaitteet.

Kesäkuussa 2010 yhtiö osti amerikkalaisen Motorolan langattomien laitteiden valmistuksesta vastaavan osaston joka nosti sen tuotealueellaan maailman toiseksi tai kolmanneksi suurimmaksi, kauppa toteutui huhtikuussa 2011. Tällä hetkellä yritys toimii 150 maassa ja sen palveluksessa on noin 66 000 henkeä. Suomessa yrityksellä on toimipisteitä Helsingissä (~4500), Oulussa (~2000) ja Tampereella (~2000). Tuotantolaitokset sijaitsevat Suomessa, Saksassa, Kiinassa ja Intiassa. Tuotekehitystä yhtiöllä on Saksassa Münchenissä ja Ulmissa, Kiinassa Pekingissä, Yhdysvalloissa Piilaaksossa sekä Suomessa Helsingissä, Tampereella ja Oulussa.

# Projektisuunnitelma – TM500 testausohjelmisto.

| **Projektisuunnitelma** | | |
|---|---|---|
| **Tilaaja:** | Nokia Siemens Networks | |
| **Laatija:** | Jaakko Reijasalo, OAMK, Raahen tekniikan ja talouden kampus | |
| **Dokumentin muutokset** | | |
| **Pvm.** | **Versio** | **Muutokset** |
| 5.11.2010 | **0** | Ensimmäinen versio |
| 18.3.2011 | **1** | Rakenne uusittu |

# SISÄLLYS

# 1. PROJEKTIN IDEA JA TAVOITE

Ohjelman idea lähti kesätyönä toteutuneesta Linux skriptistä, jonka toiminnallisuutta olisi mahdollista parantaa toteuttamalla se uudestaan sovelluksena. Uuden toteutuksen etuina olisi paremmin toteutettu käyttöliittymä ja mahdolliset uudet ominaisuudet olisi toteutettavissa helposti ja resursseja säästäen.

Ohjelman tavoitteena on luoda vanhan toteutuksen tilalle opinnäytetyönä uusi työpöytäsovellus. Ohjelmisto tulee vastaamaan vähintään vanhan skripteillä toteutetun järjestelmän ominaisuuksia, mutta parantaa käyttöliittymää sekä tulosten luettavuutta. Lisäksi ohjelmistoon voidaan lisätä uusia ominaisuuksia paljon helpommin kuin vanhalla toteutuksella, ilman että suorituskyky sekä käytettävyys kärsivät. Ohjelman jatkokehittelyä varten ohjelma tulee dokumentoida ja luoda siten että mahdollisimman pienellä vaivalla tämä voidaan toteuttaa. Projektin työvaiheessa, eli ohjelmistoprosessissa käytetään lisäävää mallia, eli yhden version valmistuttua se otetaan käyttöön ja aloitetaan kierros alusta.

## 2. Projektin organisointi

| Rooli | Nimi |
|---|---|
| Ohjaaja | Tommi Santasaari |
| Ohjaaja | Lauri Pirttiaho |
| Suunnittelija/toteuttaja | Jaakko Reijasalo |

Tommi Santasaari toimi Nokia Siemens Networksilla työn ohjaajana. Oulun ammattikorkeakoulun nimettynä ohjaajana toimi Lauri Pirttiaho. Työn toteutuksesta vastaa Jaakko Reijasalo, ryhmästä TIT8KN.

## 3. Projektin aikataulu

| **Vaihe** | **Aikajana** | **Tulokset** |
|---|---|---|
| Esitutkimus | 8-11/2010 | Esitutkimus.doc |
| Projektisuunnittelu | 11.10. – 25.11.2010 | Projektisuunnitema.doc |
| Määrittely | 25.11. – 6.4.2011 | Vaatimusmäärittely.doc |
| Suunnittelu | 1.1. – 6.4.2011 | käyttötapauskaavio.doc |
| Toteutus | 1.1. – 6.4.2011 | - |

Projekti aloitettiin virallisesti 11.10.2010 jolloin sopimus kirjoitettiin NSN Oy:n Ruskon tehtaalla Oulussa. Projektin määrittelyssä sekä opiskeltiin että toteutettiin työtä. Vaikka ohjelmointialusta sekä käyttöjärjestelmä olivat jo ennestään tuttuja, vaati työ niiden syvällisempää tutkimista.

## 4. Projektin rajaukset

- Projektin ohjelma ei tule käsittelemään kuin PPP yhteyttä TM500 laitteelle.
- Käyttökielenä tulee olemaan englanti.
- Ohjelmisto laaditaan Linux käyttöjärjestelmälle
- Käyttöympäristön muutoksia ei oteta huomioon (muuttuneet yhteystavat)

## 5. Projektin resurssit

Työssä käytettävä laitteisto koostui kannettavasta tietokoneesta jossa on asennettuna Linux – käyttöjärjestelmä. Tällä ympäristöllä hoidetaan ohjelmointi sekä esitestaus. Kun ohjelma on edennyt siihen vaiheeseen jossa vaaditaan oikean järjestelmän testausta, tulisi hetkellisesti saada käyttöön TM500 laite sekä yhteys EPC:n. Projektin toteutuksesta ei synny sovitun palkkion lisäksi muita kustannuksia.

Ohjelmistot joita työssä tarvitaan:

| | |
|---|---|
| -Linux, Fedora 12 | Käyttöjärjestelmä |
| -Qt SDK | Ohjelmointiympäristö |
| -rp-pppoe | Yhteys-sovellus |
| -TOP | Järjestelmänseuranta ohjelma |
| -BWM-ng | Kaistanseuranta ohjelma |
| -vsftp | Palvelin ohjelmisto |
| -MS Office | Työkalut projektin dokumentointiin |

# Vaatimusmääritelmä – TM500 testiohjelmisto

| | | | |
|---|---|---|---|
| **TM500 Ohjaustyökalu – vaatimusmäärittely** | | | |
| **Tilaaja:** | Nokia Siemens Networks | | |
| **Laatija:** | Jaakko Reijasalo, OAMK, Raahen tekniikan ja talouden kampus | | |
| **Dokumentin muutokset** | | | |
| **Pvm.** | | **Versio** | **Muutokset** |
| 5.11.2010 | | 0 | Ensimmäinen versio |
| 20.11.2010 | | 1 | Lisätty alustavat tiedot |
| 12.12.2010 | | 2 | Korjauksia |
| 23.2.2011 | | 3 | Uudet ideat päivitetty |
| 18.3.2011 | | 4 | Korjauksia rakenteeseen |

SISÄLLYS

# 1. JOHDANTO

Tässä dokumentissa käsitellään ohjelmistoon liittyviä vaatimuksia. Dokumenttia päivitetään työn ohessa mikäli uusia vaatimuksia tai ominaisuuksia ehdotetaan. Ohjelma tehdään Nokia Siemens Networksille opinnäytetyönä. Työn aihe on ohjelmisto LTE-testaukseen. Sen tarkoitus on parantaa vanhan ohjelmiston vaikeaa käyttöliittymää sekä pyrkiä vapauttamaan resursseja.

# 2. YLEISTÄ

Uusi toteutus tulee korvaamaan LTE–testauksessa käytettävän vanhan ohjelmiston, joka on laadittu Linux shell–skriptejä käyttäen. Ohjelmalla hallitaan TM500-testilaitetta jolla voidaan ottaa 32 yhteyttä käyttäen Point-to-Point protokollaa. Kun PPP – yhteys on avattu, pitää käynnistää ping–ohjelmisto jolla saadaan yhteyden vasteaika. Jokaiselle yhteydelle on myös pystyttävä käynnistämään oma FTP–yhteys ja sen tiedonsiirtonopeutta on pystyttävä mahdollisimman luotettavasti seuraamaan.

## 2.1 Yhteydet muihin sovelluksiin

Vanha toteutus käytti paljon ulkoisia prosesseja skriptien läpi ajettuna, tämä teki ohjelmistosta raskaan ja käytöltään kankean. Uusi ohjelmisto on tarkoitus laatia siten, että mahdollisimman vähän ohjelman ulkopuolisia prosesseja käytetään. Poikkeuksena on tällä hetkellä ping, jonka uudelleen toteutus on mietinnässä.

## 2.2 Käytettävät alustat

Ohjelma luodaan Linux käyttöjärjestelmään käyttäen Qt –framework ohjelmointialustaa.

### 2.3 Ohjelmiston rajoitteet

Rajoitteet liittyvät laitteiston tehoon. Tarkoitus on saada yhdelle tietokoneelle mahdollisimman monta yhteyttä jotka vaativat laitteistoresursseja.

## 3. VAATIMUKSET

Vaatimuksien keräykseen käytettiin yksinkertaista luettelotyyppistä menetelmää. Suurin osa vaatimuksista oli saatu jo vanhan toteutuksen yhteydessä.

### 3.1 Toimintavaatimukset

– mahdollisuus kytkeä päälle tiedonsiirto erikseen downlink ja uplink kaistoille
– tiedonsiirtonopeus summattuna erikseen downlin sekä uplink siirroille
– mahdollisuus kytkeä päälle ainakin 32 yhteyttä, tavoitteena 64
– tiedonsiirtonopeus käyttöliittymässä jokaiselle yhteydelle
– yhteydentilan seuranta käyttöliittymästä
– vasteajan näkyminen käyttöliittymässä (päivittyminen 1-3min välein)
– ohjelmiston asennuksen oltava helppo
– käyttöliittymän kieli englanti

Toissijaiset toimintavaatimukset:

– tulosten automaattinen kirjaaminen erilliseen CVS tiedostoon
– Erilaisten profiilien luominen
– Yhteyden tilan esittäminen väreillä
– kaistan kokonaiskäytöstä piirtyvä graafi

### 3.2 Yhteensopivuus ja siirrettävyys vaatimukset

Ohjelma tullaan kehittämään Linux käyttöjärjestelmään. Windows käyttöjärjestelmään mahdollinen siirtyminen yritetään ottaa huomioon mahdollisimman hyvin suunnitteluvaiheessa. Ohjelmisto pitää voida siirtää eri laitekokoonpanojen välillä.

### 3.2 Suorituskykyvaatimukset

Suorituskykyyn liittyvät vaatimukset ovat avoimet, sillä työn yksi tarkoitus on tutkia kuinka monta yhteyttä saadaan toimimaan yhdessä laitekokokoonpanossa.

### 3.3 Luotettavuusvaatimukset

Ohjelman on oltava suhteellisen vakaa, sen on toimittava pitkienkin testiskenaarioiden ajan luotettavasti.

### 3.4 Tietoturvavaatimukset

Tietoturvan suhteen ei ole vaatimuksia sillä ohjelmisto tulee sisäiseen suljettuun verkkoon.

### 3.5 Ylläpidettävyyteen ja jatkokehittelyyn liittyvät vaatimukset

Ohjelman ylläpidon tarve tulisi olla pieni, mahdollisissa virhetilanteissa tulisi pystyä käyttöohjeiden avulla selviämään ongelmista. Ohjelman tulee olla jatkokehittelyä varten mahdollisimman helposti muunneltavissa. Dokumentointi tulee suorittaa huolella, jotta jatkokehittely tai viankorjaus olisi mahdollisimman helppoa.