Ulla-Riitta Tervonen

WCDMA-TUKIASEMAN VIKADIAGNOSTIIKKA JA TESTAUS

Opinnäytetyö
Kajaanin ammattikorkeakoulu
Hallinnon- ja kaupan ala
Tietojenkäsittelynkoulutusohjelma
Syksy 2002
Pekka Heikkinen, Arto Karjalainen

ALKUSANAT

Tein opinnäytetyöni Elektrobit Group Ltd:n Kajaanin yksikössä. Työ liittyi Nokia Networks:n projektiin, jossa Elektrobit Group Ltd. oli alihankkijana, ja se on viisi vuotta luottamuksellinen tästä päivästä eteenpäin.

Kiitokset Pekka Heikkiselle Elektrobit Group Ltd:sta ja Arto Karjalaiselle Kajaanin ammattikorkeakoulusta. Kiitokset myös Saara Määtälle sekä kaikille Nokia Networks:n ja Kajaanin yksikön työntekijöille, joiden kanssa olen saanut työskennellä, ja joilta olen saanut paljon hyviä kommentteja ja kannustusta opinnäytetyöhöni liittyen.

Suuri kiitos kaikille ystävilleni ja sukulaisilleni, jotka ovat auttaneet pienen tyttäreni hoitojärjestelyissä ja näin suoneet minulle mahdollisuuden selvitä opinnoistani. Erityisesti haluan muistaa molempia vanhempiani sekä tätiäni Terttua.

Ja lopuksi…
Haluan omistaa tämän työn rakkaalle tyttärelleni Unnukalle.

Ulla-Riitta Tervonen
Kajaanissa lokakuun 31. päivä, 2002

Kajaanin ammattikorkeakoulu

OPINNÄYTETYÖ

TIIVISTELMÄ

Ala Kauppa- ja hallinto	Koulutusohjelma Tietojenkäsittely
Tekijä(t) Ulla-Riitta Tervonen	
Työn nimi WCDMA-tukiaseman vikadiagnostiikka ja testaus	
Vaihtoehtoiset ammattiopinnot	Ohjaaja(t) Arto Karjalainen, Pekka Heikkinen
Aika 31.10.2002	Sivumäärä 39 + liitteet

Tiivistelmä

Tämä opinnäytetyö liittyi Nokia Networks:n tukiasemien vikadiagnostiikan kehitysprojektiin, jossa Elektrobit Group Ltd oli alihankkijana. Vikadiagnostiikan tehtävänä on paikallistaa laitteistossa olevat viat, käsitellä ne sekä raportoida eli hälyttää niistä käyttäjälle ennen kuin ne vahingoittavat laitteiston muita osia.

Työssä testattiin vikadiagnostiikan ohjelmistoa ensin työasemaympäristössä ja sen jälkeen WCDMA-tukiasemissa. Löydetyt komponenttiviat käsitellään ja tallennetaan tietokantaan. Jos vika on jo aktiivinen, se poistetaan ja turhia hälytyksiä ei lähetetä operaattorille. Jokaisella hälytyksellä on ainakin yksi päähälytys tai seurannaishälytys. Jos päähälytys aktivoituu vain se lähetetään eteenpäin käsiteltäväksi. Säännöt hälytysten käsittelyyn on XML-muotoisessa tiedostossa. Käyttö- ja ylläpito-ohjelmiston huomattua vian yksikössä tai tiedonsiirtoväylässä tieto lähetetään vikadiagnostiikan käsiteltäväksi.

Työssä on esitelty vikadiagnostiikan perusteet WCDMA-tukiasemissa ja testaustyökalujen merkitys testausprosessissa sekä etsiä mahdollisia kehitysehdotuksia ohjelmistotestaukselle. Testaus keskittyi vikadiagnostiikan ja etenkin sääntötiedoston testaamiseen tukiasemassa. Testattava vika aiheutettiin, ja tarkistettiin ajonaikaisista tulosteista hälytystenkäsittely sekä hälytysten raportointi käyttö- ja ylläpitoliittymällä. Jos hälytystenkäsittely ei toiminut määritetyllä tavalla, sääntötiedostoa tai vikadiagnostiikan mallia muokattiin ennen uutta testausta.

Luottamuksellisuus	luottamuksellinen 31.10 2007 saakka
Hakusanat	vikadiagnostiikka, testaus, WCDMA-tukiasema
Säilytyspaikka	

Kajaanin ammattikorkeakoulu

ABSTRACT OF THE FINAL YEAR PAPER

Faculty Administration and Business	Degree programme Data Processing
Author(s) Ulla-Riitta Tervonen	
Title Testing of Fault Diagnostics at the WCDMA Base Stations	
Alternative professional studies	Instructor(s) Arto Karjalainen, Pekka Heikkinen
Date October 31, 2002	Total number of pages 39 + appendices

Abstract

The study was a part of a project to improve fault diagnostics at Nokia Networks´ WCDMA base stations. Elektrobit Group Ltd was the subcontractor in the project. The purpose of fault diagnostics is to find the fault in the base station, manage the fault handling and report the fault to the operator before it causes any failure in the system.

Firstly, alarm handling was tested in the workstation environment. Then the fault diagnostic software was linked and transferred to the base station and tested there. The purpose of the study was to introduce the testing tools that had been used and how the testing process of fault diagnostics was practically handled in laboratory conditions. The main testing subject was the XML format file consisting of the alarm handling rules. The software was transcribed into the base station units and the fault was caused according to the test plan. Finally, the alarm handling management was followed in the runtime prints as well as in the operation and maintenance interface, which is the tool the operator uses. If the test was refused the instructions or the model was modified before a retest.

The background information of testing and fault diagnostics was received from books and the Internet. In addition, the testing personnel was interviewed to discover any weaknesses in the testing procedure and to receive suggestions for improvement. The results of the survey indicated that detailed test cases were not available as they should be and the documentation was not of acceptable level. Thirdly, more training about the use of testing tools, the theory of fault diagnostics and hardware would make the testing process more efficient.

Confidentiality status	Confidental until October 31, 2007
Keywords	Fault diagnostics, alarm, WCDMA, base station, testing
Deposited at	

SISÄLLYSLUETTELO

LYHENNELUETTELO

Ascii	American Standard Code for Information Interchange.
Binary	Binäärijärjestelmä, kaksijärjestelmä.
black-box testaus	Mustalaatikkotestauksessa ohjelman toimintaa verrataan sen määrittelyihin ilman mitään tietoa ohjelman sisäisistä rakenteista.
DUT	Device Under Test. Testattavana oleva laite tai toteutus.
FD SW	Fault Diagnostic Software, Vikadiagnostiikan ohjelmisto.
Flash	Prosessorin muisti, jossa tieto säilyy, vaikka virta yksiköstä katkaistaan.
FMEA	Failure and Mode Effect Analysis, Vika-vaikutus analyysi (VVA), joka on keskeinen laadunsuunnittelun menetelmä niin tuote- kuin prosessisuunnittelussa.
FTP	File Transmission Protocol. Kahden laitteen/ohjelman välinen tiedostonsiirtoprotokolla.
NT-Tester	Rhapsody-mallilla luotu NT-simulointiympäristö, jolla voidaan tarkistaa myös sääntötiedoston rakenteen oikeellisuus sekä simuloida hälytyksiä.
Octopus	Nokian kehittämä oliosuuntautunut (UML-tyyppinen) mallinnuskieli, jota on käytetty esimerkiksi vikadiagnostiikan mallintamiseen.
O&M	Operation & Maintenance, käsitetään käyttö- ja huoltotoimenpiteinä tukiasematekniikassa
O&M SW	Tukiasemassa oleva käyttö- ja kunnossapidon käyttämä ohjelmisto. Ohjelmisto ohjaa tukiaseman toimintaa ja raportoi operaattorille tukiaseman tilaa.
PC	Personal Computer, mikrotietokone

ping	Internetin perusohjelma, jolla tutkitaan, onko tietty IP-osoite olemassa ja vastaako se palvelupyyntöihin.
Rhapsody	UML-mallinnusmenetelmää käyttävä työkalu, jolla voidaan muodostaa laajoja ohjelmistokokonaisuuksia ja generoida mallilla esimerkiksi C++-ohjelmointikoodia. Käytetään hyväksi erityisesti tietoliikennetekniikan sovelluksissa.
RNC	Radio Network Controller.
SQA	Software Quality Assurance, Ohjelmoinnissa yleisesti käytetty malli, jonka avulla pyritään takaamaan ohjelmiston hyvä laatu.
TCP/IP	Transport Control Protocol/Internet Protocol. Tiedonsiirtoprotokolla, jota voivat käyttää Internetiin liitetyt koneet. IP-protokolla reitittää siirrettävät paketit ja TCP-protokolla muodostaa yhteyden koneiden välille sekä vastaa pakettien pilkkomisesta.
UDP	User Datagram Protocol, TCP/IP-tiedonsiirtoprotokolla, jota käytetään kahden koneen välillä näyttämään sanomia. Osa sanomista voi hävitä; ei ole siis täysin luotettava esitystapa.
UDP-Log	UDP Datagram -protokollalla toimiva ohjelma, jolla saadaan ajoaikainen tiedosto PC:lle tukiasemaohjelmiston tulosteista.
UML	Unified Modeling Language, standard OO modelling language. Mallinnuskieli, jota käytetään paljon etenkin tietoliikennetekniikan sovelluksissa. Mallin avulla saadaan selkeä visuaalinen esitystapa ja sen avulla saadaan aikaan jo suunnitteluvaiheessa hyvä dokumentointiaineisto.
WCDMA	Wideband Code Division Multiple Access, Laajakaistainen radioliikennetekniikka runsaasti kapasiteettia vaativiin sovelluksiin. WCDMA-tekniikkaa käytetään mm. kolmannen sukupolven matkapuhelinjärjestelmissä.

white-box testaus	Lasilaatikkotestauksessa testataan ohjelmiston rakennetta ja sen toimintaa.
ws_ftp	Windows pohjainen ftp-protokollaa käyttävä tiedonsiirto-ohjelma PC:ltä palvelimelle.
XML	Extensible Markup Language, ohjelmoinnissa käytetty metakieli, jonka avulla luodaan XML-tiedostoja

KUVALUETTELO:

1 JOHDANTO

Tietoliikennetekniikka kehittyy ja uusia tuotteita tuodaan markkinoille nopeaa vauhtia. Laitteiden tulisi olla helppokäyttöisiä ja luotettavia. Virhetilanteissa laitteiden tulisi pystyä ilmaisemaan käyttäjälle, missä on vikaa ja millaisia toimintoja käyttäjän tulisi tehdä, jotta vika saataisiin poistettua ja saattaa laite normaaliin toimintakuntoon. Laitteen hyvä toimintavarmuus ja pitkän huoltovälin tuoma huolettomuus vaativat hyvän vikadiagnostiikan.

Tämä opinnäytetyö liittyi Nokia Networks:n projektiin, jossa Elektrobit Group Ltd oli Nokia Networks:n alihankkijana. Projektissa kehitettiin WCDMA (Wideband Code Division Multiple Access) tukiasemiin ohjelmistoa, jonka tarkoituksena olisi olla tukiasemasukupolvista riippumaton. Olennaisena osana tukiaseman ohjelmistoon liittyy Octopus-menetelmällä kehitetty vikadiagnostiikka, jonka tavoitteena on löytää tukiaseman yksiköissä tai tiedonsiirtoväylissä mahdolliset viat ennen kuin ne ehtivät aiheuttaa lisävahinkoa tukiaseman muille yksiköille ja siten heikentää tukiaseman toimintakykyä. VTT Electronics ja Nokia Networks ovat kehittäneet yhdessä vikadiagnostiikkaa vuodesta 1995 alkaen useissa projekteissa. Näiden projektien tavoitteena oli tutkia tehtyjä tukiasemasukupolvien toiminnallisia malleja ja tuottaa geneerinen palvelumalli, joka on järjestelmäriippumaton, laajennettavissa ja konfiguroitavissa.

Työssä on esitetty, kuinka WCDMA-tukiaseman vikadiagnostiikan testaus suoritettiin sekä esitelty siinä käytetyt työkalut. Testaus tarkentui samalla, kun vikadiagnostiikka-

ohjelmiston toteutus eteni. Olennaisena osana testaukseen kuului myös versiopäivitykset sekä vikatietokannan käyttäminen virheenkorjauksessa. WCDMA-tukiasemien kalliin hinnan vuoksi niiden sijoituspaikkoina tulee olemaan suuret kasvukeskukset ja niitä käytetään yhdessä toisen sukupolven eli GSM-tukiasemien rinnalla. WCDMA-tukiaseman etäisyys operaattorin ohjainkeskuksesta on vain muutama kilometri. WCDMA-tukiasemien puhelunvälityskyky vaihtelee tukiasematyypeittäin ja kalustuksen eli yksiköiden lukumäärän mukaan. Tässä työssä testattiin useaa erilaista WCDMA-tukiasemaa. Puhelujenvälityskapasiteetti näiden tukiasemien välillä vaihtelee paljon. Suurimmat näistä tukiasemista täydellä kalustuksella varustettuina voivat välittää useita satoja puheluja samanaikaisesti.

WCDMA-tukiasemien vikadiagnostiikan ensimmäinen testausvaihe suoritettiin NT-ympäristössä ja sen jälkeen lopullisessa toimintaympäristössään eli tukiasemassa (ns. target-testaus). Tässä työssä keskityttiin vikadiagnostiikan target-testaukseen.

Hyvin suunniteltu ja toimiva vikadiagnostiikka helpottaa asiakkaan toimintaa. Kunnossapito- ja huoltotyöt vähenevät ja operaattorin käyttämä tukiasemaohjaimen käyttö helpottuu ja selkeytyy. Turhat vikahälytykset jäävät pois ja vain olennainen tieto tukiaseman toiminnasta välittyy operaattorille. Vikadiagnostiikan avulla tarjotaan operaattorille valmiiksi ajateltu malli, jolla hälytysten informaatio tuotetaan sellaiseen muotoon, että mahdollinen vika voidaan korjata.

Luvussa 2 käsitellään vikadiagnostiikkaa yleisesti; mitä sillä tarkoitetaan ja miksi se on tärkeää. Vikadiagnostiikkaa on tutkittu paljon ja sitä on sovellettu monilla eri tekniikan aloilla prosessitekniikasta tietoliikennetekniikkaan. Luvussa 3 käsitellään vikadiagnostiikan teoriaa ja periaatteita WCDMA-tukiasemissa ja luvussa 4 vikadiagnostiikan testausta. Lukuun 5 keräsin WCDMA-tukiasematestauksessa mukana olleilta testaajilta heidän mielipiteitään ja kokemuksiaan testausprosessista. Yhteenveto-lukuun kokosin omien kokemusteni ja toisilta testaajilta saamien mielipiteiden pohjalta testaukseen liittyviä parannusehdotuksia vikadiagnostiikan jatkokehitystyötä ajatellen.

2 VIKADIAGNOSTIIKKA

Vikadiagnostiikalla tarkoitetaan laitteessa ilmenneen vian löytämistä ja siitä ilmoittamista käyttäjälle ennen kuin se ehtii vahingoittamaan laitteiston muita osia. Se varmistaa laitteiden toimintavarmuuden ja helpottaa käyttäjän kunnossapito- ja ylläpitotoimenpiteitä. Vian löytäminen, sen mallintaminen ja käsittely ovat vikadiagnostiikan tärkeimmät tehtävät.

Vikadiagnostiikan mallinnuksessa otetaan huomioon mallin rakenne ja mallin uudelleenkäytettävyys. Koska tekniset järjestelmät kehittyvät nopeasti ja entistä monimutkaisimmiksi, on erityisesti vikadiagnostiikan suunnitteluun kiinnitettävä huomiota. Yleistä vikadiagnostiikan teoriatietoa tarvitaan useilla eri teknologian aloilla. (Guckenbiehl, Malik, Milde, Neumann & Struss 2000).

2.1 Vian mallinnus

Vikadiagnostiikalle asetetaan yleensä seuraavia vaatimuksia. Vikadiagnostiikan suunnittelun ja toteutuksen tulisi olla taloudellisesti kannattavaa ja se tulisi suunnitella siten, että se olisi jatkossa helposti sovellettavissa ja muutettavissa sekä ohjelmistopäivitys olisi joustavaa.

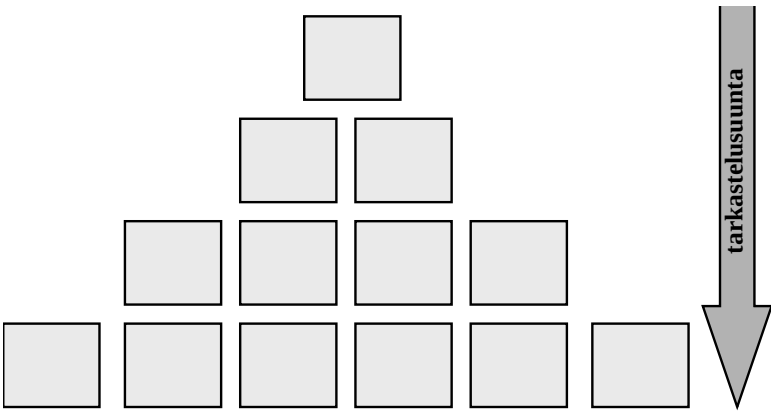
Vikadiagnostiikan mallinnuksessa esitetään laitteissa esiintyvät mahdolliset viat (engl. fault) sekä niiden vaikutus prosessiin eli laitteen toimintaan. Vika komponentissa

(yksikössä) tarkoittaa sitä, että sen toimintatila poikkeaa normaalista eikä komponentti ole kokonaan toimintakyvytön. Häiriöllä (engl. failure) puolestaan tarkoitetaan, että komponentti on toimintakyvytön. Vikadiagnostiikan tärkein tehtävä on löytää ilmenneet viat ennen kuin ne ehtivät aiheuttaa häiriöitä muussa toimintaympäristössään. (Frisk 2001, 8).

Vikadiagnostiikan vikojen määrittelyssä vaikeinta on ns. kriittisyysrajan asettaminen oikealle tasolle. Jos taso määritetään liian alas, tulee turhia hälytyksiä ja liian korkealle asetettu kriittisyysraja puolestaan estää oikeiden hälytysten lähettämisen. Vikadiagnostiikassa käytetään useita eri mallinnusmenetelmiä. Vikadiagnostiikan mallinnuksen yleisimmät FMEA (Failure and Mode Effect Analysis) mallinnusmenetelmät ovat vikapuuanalyysi sekä vika-vaikutus –analyysi. Nämä ovat ns. ennaltaehkäiseviä laatumenetelmiä.

Vikapuuanalyysi (VA)

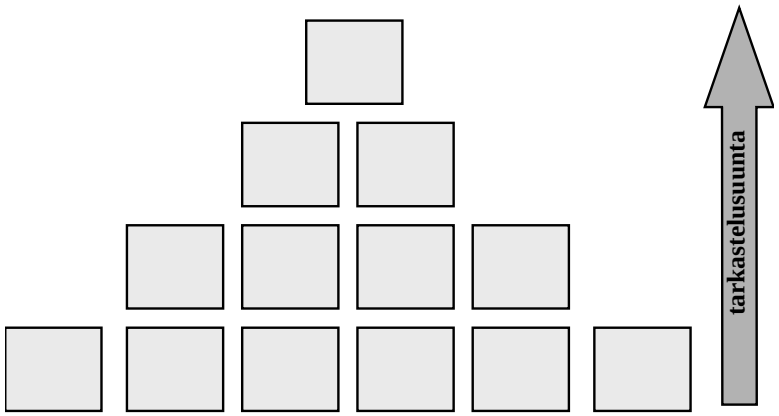
Vikapuun avulla lasketaan vian todennäköinen esiintyminen sen komponenteissa esiintyvien todennäköisten vikojen perusteella. Vikapuulla voidaan määrittää monenlaisten järjestelmien toimintaa ja sitä käytetään erityisesti vikojen paikantamiseen. Vikapuusta muodostetaan graafinen esitys, jossa ns. huipputapahtuma eli vaarallisin olosuhde, joka laitteessa tai yksikössä esiintyy, on vikapuun juuri ja oksat kuvaavat vian syitä eli aiheuttajia. Vikapuu laaditaan siten, että siihen otetaan mukaan todennäköisimmät ja merkittävimmät viat. Vikapuuta luetaan ylhäältä alaspäin (kuva 1), jolloin tarkastelussa etsitään vika, joka aiheuttaa päävian ja sen jälkeen siirrytään vikapuun yläosista yhä pienempiin osiin. Vikapuuanalyysia käytetään hyödyksi erityisesti vikojen kriittisyyden arvioinnissa. Tilasta toiseen siirrytään joko TAI-porttien tai JA-porttien kautta. Osatapahtumien todennäköisyyden perusteella voidaan laskea lopputapahtuman todennäköisyys. Vikapuu-analyysissa tarkastellaan ensin suuremmasta osasta pienempään päin, eli ensin koko ohjelmistoa ja sen jälkeen moduulitasoja. (TTKK 2000), (Tuotekehitys ja laatu 1999), (Kärki & Karjalainen 1999).



Kuva 1. Vikapuuanalyysin tarkastelusuunta.

Vika-vaikutus –analyysi (VVA)

Vika-vaikutus –analyysissa tutkitaan vian vaikutusta järjestelmään eli pyritään selvittämään vikojen aiheuttamat seuraukset. Järjestelmä jaetaan osiin, esimerkiksi pienempiin kokonaisuuksiin ja komponentteihin. Vikojen aiheuttamat syyt kuvataan mahdollisimman tarkasti. Koska vika-vaikutus –analyysia käytetään vikojen paikantamiseen, on tärkeää päättää, millä tasolla toimintaa tarkastellaan. Vioittumista tutkitaan alhaalta ylöspäin (kuva 2). Häiriöiden vakavuusasteiden arvioinnin avulla saadaan kokonaiskuva järjestelmästä (Tuotekehitys ja laatu 1999). Vian löytyessä sen vaikutuksia voidaan arvioida ja mahdollisesti estää haitallisten vaikutusten syntyminen. (Harju 1999), (TTKK 2000), (Kärki & Karjalainen 1999).



Kuva 2. Vika-vaikutus -analyysin tarkastelusuunta.

2.2 Vikojen luokittelu

Häiriötilanteet luokitellaan usein vikadiagnostiikassa niiden vakavuuden perusteella. Jos esimerkiksi samasta viasta aiheutuu useita eritasoisia häiriötilanteita, vain vakavimmat hälytykset raportoidaan ylöspäin. Vika voi olla kriittinen (engl. critical), vakava (engl. major) tai vähäinen (engl. minor). Hälytys voi olla myös tiedotusluonteinen (info). Kriittinen vika on silloin, kun sen vaarana on heikentää koko laitteiston toiminta normaalista toimintakyvystä toimimattomaksi. Esimerkiksi tukiaseman ainoan kellon vikaantuminen on kriittinen vika. Reaaliaikaisuus on oleellista tukiasemaohjelmiston toiminnalle, sillä kello ohjaa tukiaseman muita yksiköitä toimimaan oikeassa järjestyksessä.

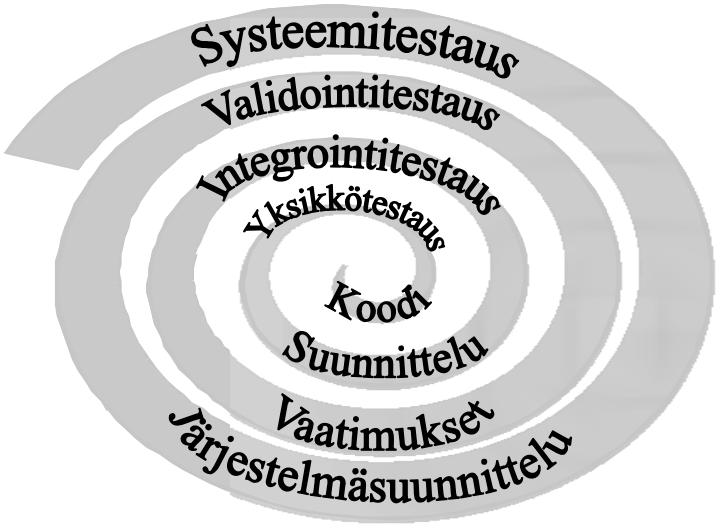
2.3 Ohjelmistoviat

SQA:n (Software Quality Assurance) mukaan testauksen tarkoitus on varmistaa mahdollisimman aikaisessa vaiheessa ohjelmiston viat ja samalla varmistaa ohjelmiston hyvä laatu. SQA perustuu projektin määrittelyihin eli ohjelmiston pitäisi toimia sille asetettujen määrittelyjen mukaisesti. Ohjelmiston virheenä voidaan pitää kaikkia niitä tilanteita ja toimintoja, joissa ohjelman toiminta poikkeaa oletetusta. (Kautto 1996). Ohjelmiston hyvä laatu varmistetaan riittävällä testaamisella ja katselmoinneilla. Laadunvarmistus on testaamisen lisäksi myös ohjelmiston analysointia, auditointia sekä raportointia. (Forsell 2001).

2.4 Testaus

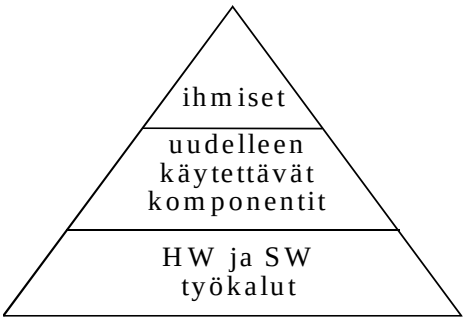
Ohjelmistotestauksen tärkeimmät asiat ovat tarkka suunnitelma testauksesta ja testausten dokumentointi. Suunnittelulla määritetään sellaiset systemaattiset käytännöt, että ajankäyttö olisi tehokasta ja viat löytyisivät ajoissa. Testauksen strategia voidaan esittää spiraalimallilla (kuva 3) (Pressman 2000, 469). Testausprosessi alkaa spiraalin sisältä yksikkötesteistä, jolloin testattavana on itse ohjelmistokoodi ja spiraalin ulommaisena rinkinä on systeemitestaus, joka pohjautuu kokonaisjärjestelmän toimivuuteen. Validointi-testauksessa testataan ohjelmiston ja vaatimusmäärittelyn vastaavuus. Testaus suoritetaan

aina pienemmästä osasta laajempaan kokonaisuuteen eli yksikkötestauksesta systeemitestaukseen ja white-box -testauksesta black-box -testaukseen. White-box -testaukseen katsotaan yleisesti kuuluvaksi yksikkötestaus ja black-box –testaukseen integrointi-, validointi- ja systeemitestaukset. On kuitenkin muistettava, että riittävällä testauksella voidaan löytää ohjelmiston viat, mutta ohjelmiston oikeellisuutta sillä ei pystytä varmistamaan.



Kuva 3. Testausstrategia

Testaukseen käyttävissä olevat resurssit on kartoitettava jo ennen varsinaista testausvaihetta. Kuvassa 4 on esitetty kehitysprosessin resurssit pyramidin muodossa. Kehitysympäristö eli laitteisto ja suunnittelussa käytetyt ohjelmistot pyramidin pohjalla luovat perustan kehitystyölle. Uudelleenkäytettävät komponentit ovat pyramidin toisella tasolla. Niitä ovat edellisistä projekteista saadut valmiit ohjelmistot, määritykset, suunnitelmat, ohjelmistokoodit ja testaustiedot, jotka vastaavat kokonaan tai osittain käynnissä olevalle projektille suunniteltavia ohjelmistoja. Pyramidin ylimmällä tasolla on projektin (testauksen) ensisijainen voimavara eli projektissa mukana olevat ihmiset. Henkilöstön lukumäärä vahvistuu vasta sitten, kun tiedossa on koko projektin laajuus ja aikataulu. Pienissä projekteissa käytetään usein hyväksi konsultaatiota erikoisosaamisessa (Pressman 2000, 122-124).



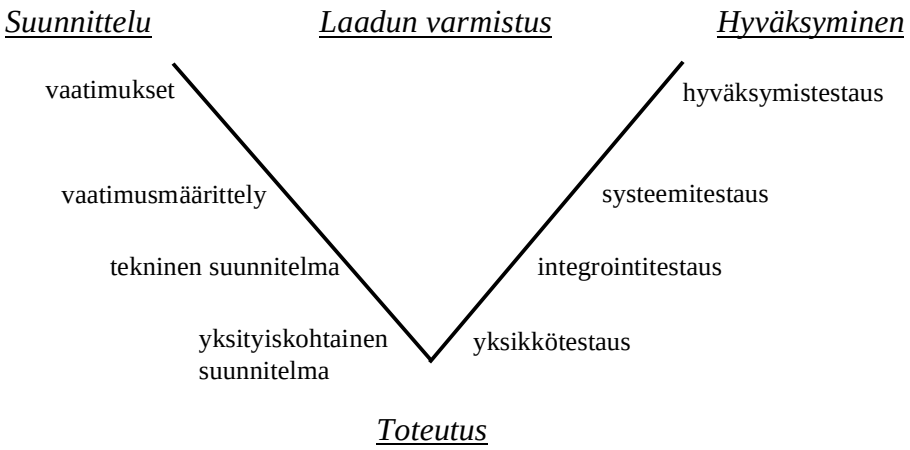
Kuva 4. Projektin resurssit eli voimavarat.

Lisäksi ohjelmiston kokoa täytyy mitata, jotta saadaan selville mahdollisten virheiden lukumäärä. Testauksen tehtävänä on löytää mahdollisimman paljon virheitä jo tuotteen suunnitteluvaiheessa. Virheenkorjaus suunnittelun ja toteutuksen aikana maksaa vain noin 1 % ja systeemitestauksen aikana noin 10 % siitä, mitä se tulisi maksamaan tuotteen käyttöönoton jälkeen. (Forsell 2001), (Pressman 2000, 465).

Ohjelmistotestauksissa yleensä käytetty ns. V-malli esittää optimaalisen testausprosessin, jossa testaukseen kiinnitetään huomiota jo suunnitteluvaiheessa, vaatimusmäärittelyn muodostuessa. Systeemitestaus on mukana kiinteästi koko ohjelmistoprojektin elinkaaren ajan laadunvarmistuksen takaamiseksi. Kuvassa 5 on esitetty V-malli, josta selviää testauksen ja suunnittelun samanaikaisuus. Kussakin testausvaiheessa ohjelmiston toimintaa verrataan määrittelyihin, jotka ohjelmistolle (laitteistolle) on asetettu.

SQA:n (Software Quality Assurance) mukaan ohjelmistotestauksen vastuu jakaantuu seuraavasti: yksikkötestauksesta vastaa suunnittelijat, laadun takaa systeemitestaus ja hyväksymistestauksesta vastaa hyväksymistestausryhmä, joka on yleensä asiakas.

Teknisen suunnittelun testauksesta vastaa suunnittelijat ja teknisen määrittelyn toteuttajat. (Batzman 2000.)



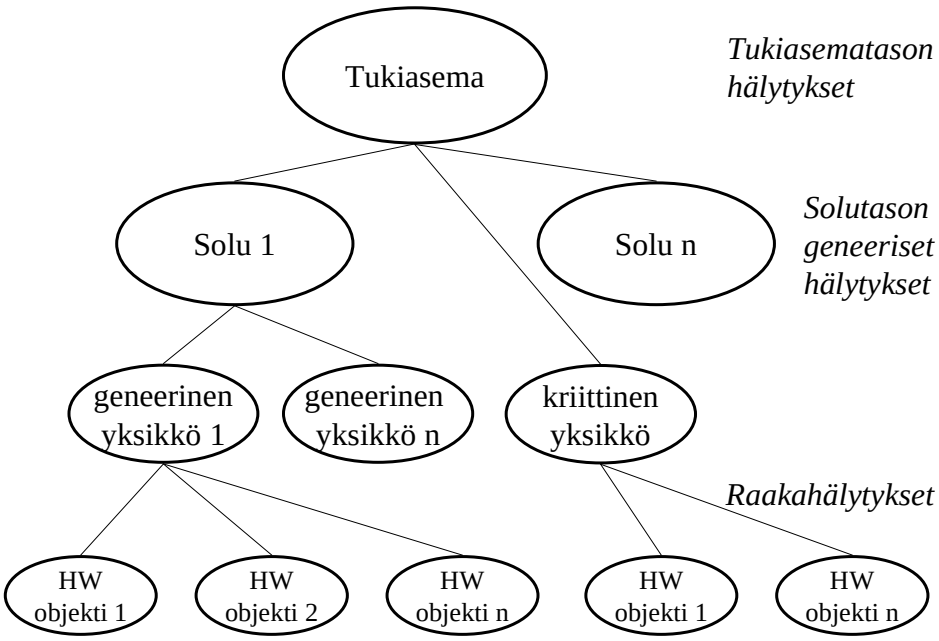
Kuva 5. Testausprosessi kuvattuna ns. V-mallilla.

3 TUKIASEMAN VIKADIAGNOSTIIKKA

Hankalaksi WCDMA-tukiaseman vianmallinuksen suunnittelun on tehnyt tukiasemaympäristön monimutkaisuus ja aikaisempien kokemusten puutuminen. Suunnittelun alkuvaiheessa ei ole ollut tietoa tarkasta asiakkaan tarpeesta vikadiagnostiikan suhteen; minkälaisia vaatimuksia asiakas asettaa tukiaseman vikadiagnostiikalle ja minkälaiset vaatimukset asiakkaalla on esimerkiksi huolto- ja kunnossapitotyötä ajatellen.

Tässä WCDMA projektissa hälytysten käsittely perustuu geneerisiin yksiköihin ja palvelumallien luokitteluun. Tukiasema koostuu itsenäisistä soluista, jotka tarvitsevat tiettyjen yksiköiden palveluja toimiakseen. Tukiasemassa voi olla yhdestä useampaan soluun tukiaseman tyypistä riippuen, ja solussa voi olla monta samanlaista yksikköä.

Kuvassa 6 on esitetty periaate, johon hälytysten luokittelu WCDMA-tukiasemien vikadiagnostiikassa perustuu. Hälytyksiä on kolmen tasoisia: tukiasematasoisia, solutasoisia ja yksikkötasoisia ns. raakahälytyksiä. Tukiasema on toimintavalmiudessa, jos sillä on vähintään yksi solu ja vaadittavat kriittiset yksiköt toimintakunnossa. Jos kriittinen yksikkö kuten kello, antenniyksikkö, signaalinsuodatin, lämmitin tai tuuletin on epäkunnossa, operaattorille lähetetään vakavimman tasoinen hälytys (tukiasema viallinen). Raakahälytyksiä voi olla useita, mutta vikadiagnostiikan avulla viat pyritään korjaamaan ennen kuin ne aiheuttavat solutasoisen hälytystilanteen. (Puurunen 2000).

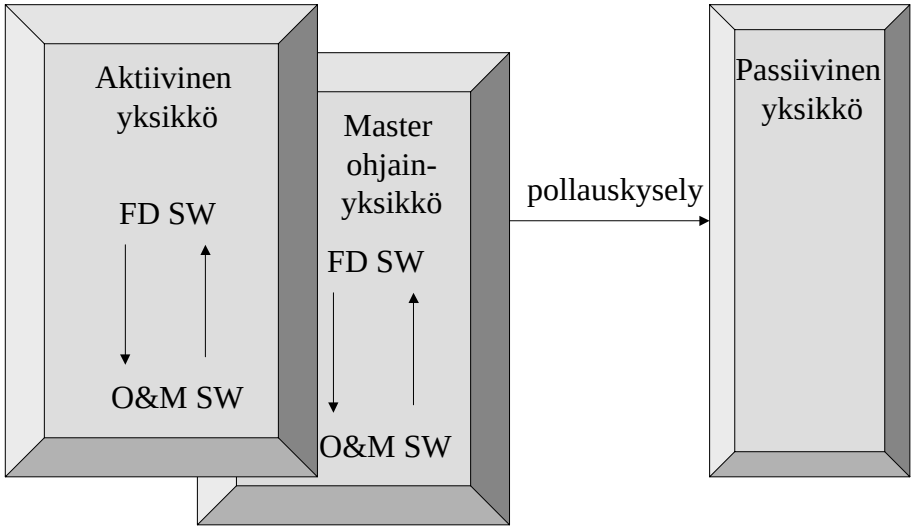


Kuva 6. Hälytysten luokittelu geneeristen yksiköiden avulla.

3.1 Hälytysten käsittely

Laite paikallistaa hälytyksen ja kohdistaa sen vialliseen yksikköön tai tiedonsiirtoväylään. Havaittu vika aiheuttaa hälytyksen, joka välitetään laitteen käyttäjälle. Vika voi aiheuttaa monta uutta hälytystä, mutta ainoastaan alkuperäinen raportoidaan eteenpäin. Hälytykset havaitaan automaattisesti ja laite määrittää yksityiskohtaisesti sen viallisen yksikön tai väylän, mistä hälytys on tullut.

Aktiiviset yksiköt eli ne yksiköt, joilla on oma kontrolliyksikkönsä, käyttävät hälytystenkäsittelyssä sekä käyttö- ja kunnossapito-ohjelmistoa (O&M SW) että vikadiagnostiikan ohjelmistoa (FD SW). Aktiiviset yksiköt lähettävät viestin hälytyksestä O&M SW:lle, joka kerää tietoja vioista ja lähettää ne edelleen vikadiagnostiikan käsiteltäväksi. Vikadiagnostiikka paikallistaa hälytyksen aiheuttajan, huolehtii hälytysten luokittelusta, tiedonkeruusta, toimintojen toipumisesta sekä hälytysten raportoinnista. Passiivisilta yksiköiltä puuttuu kontrolliyksiköt ja siksi solun ohjainyksikkö kysyy niiltä säännöllisin väliajoin, että ovatko ne toimintakunnossa (ns. pollauskysely) (kuva 7).

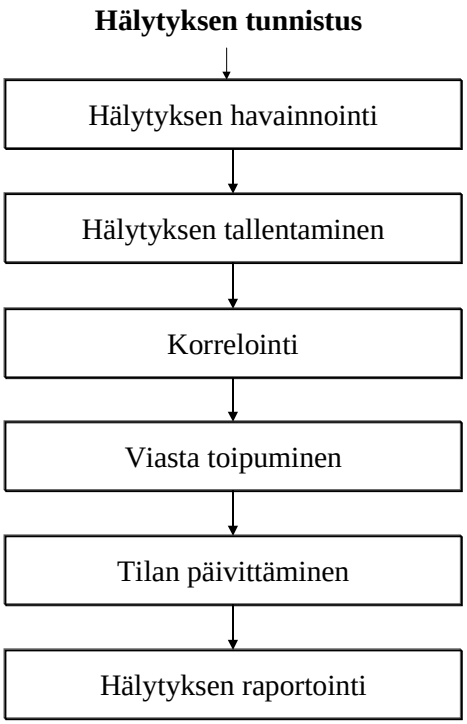


Kuva 7. Aktiivisen ja passiivisen yksikön suhde.

Koska hälytysten käsittely perustuu tietoon yksikön tilasta, ei turhaa hälytysten lähettämistä ja yksiköiden tilan päivittämistä ole. Jos sama hälytys on tullut jo aikaisemmin ja häly on edelleen aktiivinen, uusia hälytyksiä ei välitetä. Mutta jos yksikön tila muuttuu uuden hälytyksen johdosta toimintakyvyttömästä vialliseksi, hälytys lähetetään ylöspäin ja yksikön tila päivitetään uudestaan.

Perinteisiltä GSM tukiasemalta saattaa tulla vikaantuneista yksiköistä ja tiedonsiirto-ongelmista operaattorille noin 6500 hälytystä päivässä. Vain pieni osa näistä hälytyksistä ovat vikaantuneiden yksiköiden aiheuttamia, vaan suurin osa hälytyksistä on ns. seurannaishälytyksiä. Vikadiagnostiikan kehittämisellä pyritään vähentämään hälytyksiä sekä näyttämään operaattorin tukiasemaohjaimelle vain ne hälytykset, jotka ovat varsinaisia hälytysten aiheuttajia. (Puurunen 2000, 21).

Kuvassa 8 on esitetty vikadiagnostiikan eteneminen funktiolta toiselle (Puurunen 2000, 19).



Kuva 8. Hälytyksen käsittelyfunktiot.

3.1.1 Hälytyksen havaitseminen ja tallentaminen

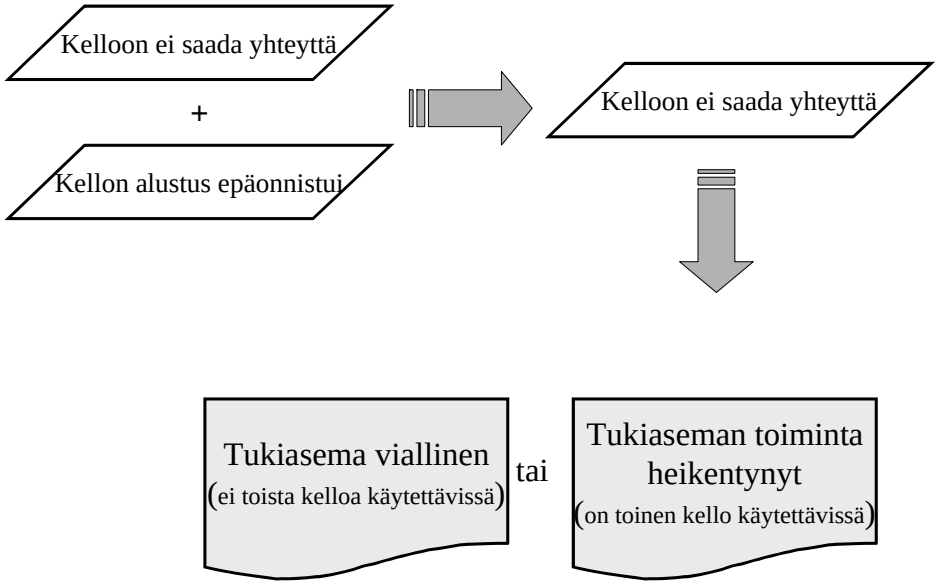
Jokainen hälytys poimitaan ja tallennetaan hälytyshistoriaan, jossa jokaisella hälytyksestä tiedetään aika, hälytysnumero ja –tunniste, nimi, hälytyksen aiheuttanut yksikkö tai väylä sekä tilatieto. Jos uusi hälytys ei löydy hälytystietokannasta, se tallennetaan ja aloitetaan sen käsittely.

3.1.2 Hälytyksen korrelointi

Korreloinnin tavoitteena on löytää hälytyksen todellinen syy ja aiheuttaja sekä raportoida siitä eteenpäin. Korrelointisäännöt, jotka määritellään hälytysketjuina, on tallennettu ns. sääntötiedostoon. Säännöissä määritellään hälytysten seurannaisvaikutus ja mistä hälytys on mahdollisesti aiheutunut eli generoitunut. Hälytykset ovat joko päähälytyksiä tai seurannaishälytyksiä ja jokaisella hälytyksellä on joko päähälytys tai useampi

seurannaishälytys. Jos hälytyksellä on päähälytys, mikä ilmenee yleensä noin 0,5 – 1-minuutin kuluessa, niin ainoastaan päähälytys lähetetään tukiasemaohjaimelle.

Useimmilla tukiaseman yksiköillä on ns. redundanttiyksikkönsä eli ylimääräinen, samanlainen yksikkö, joka otetaan automaattisesti käyttöön toisen vikaantuessa. Jos vikadiagnostiikalle tulee hälytys ”*Kellon alustus epäonnistui*” ja samalle kellolle on jo aktiivinen hälytys ”*Kelloon ei saada yhteyttä*”, seurauksena on, että ”*Kellon alustus epäonnistui*” –hälytys poistetaan automaattisesti. ”*Kelloon ei saada yhteyttä*” hälytys lähetetään ylöspäin ja operaattori saa ilmoituksen ”Tukiaseman toiminta heikentynyt”, jos redundanttinen kello voidaan ottaa käyttöön. Jos toista kelloa ei ole käytettävissä, operaattori saa ilmoituksen ”Tukiaseman viallinen” (kuva 9).



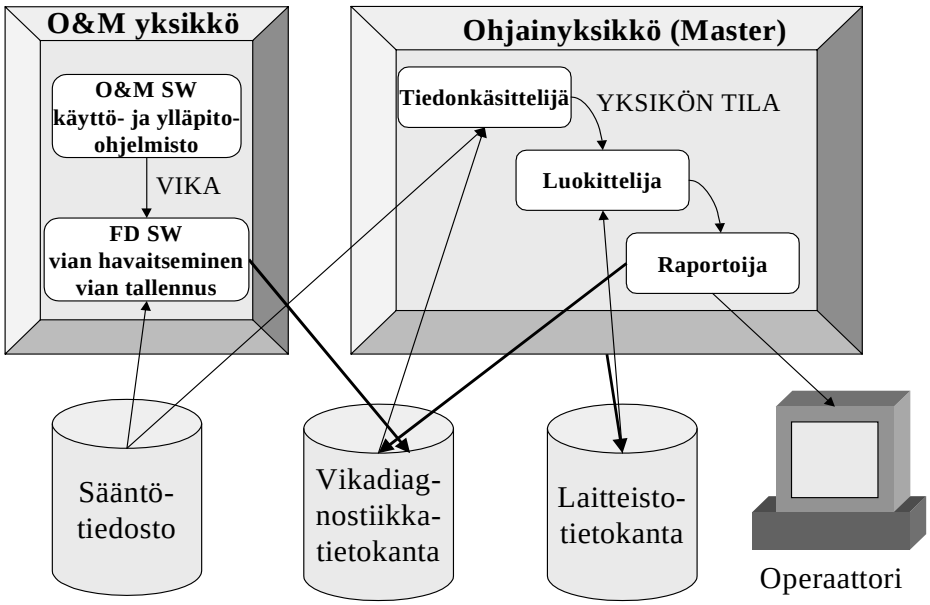
Kuva 9. Esimerkki kellohälytyksen korreloinnista.

3.1.3 Viasta toipuminen

Vian ilmetessä siitä pyritään toipumaan mahdollisimman nopeasti joko korjaamalla tai eristämällä (engl. block) vikaantunut yksikkö tai solu siten, ettei se aiheuta lisävikoja tukiaseman muille yksiköille tai soluille. Yksiköitä ja soluja voidaan palauttaa ns. reset-toiminnoilla, jolloin yksiköstä tai valitusta solusta käytetään virrat pois. Resetoinnin jälkeen tarkistetaan ohjelmistojen lataantuminen ja jos resetointi suoritettiin jonkin vian vuoksi, tarkistetaan vian häviäminen. Redundanttinen yksikkö antaa mahdollisuuden vikaantuneen yksikön vaihtamiseen tai korjaamiseen tukiaseman toiminnan siitä häiriintymättä. Vikaantunut yksikkö eristetään ennen irtiottamista. Esimerkiksi signaalinsuodatinyksiköt voidaan vaihtaa ilman eristämistä (ns. hot-insert and remove – toiminto). Tukiaseman resetoinnissa yksiköille suoritetaan parametrien ja ohjelmistojen uudelleenlataus.

3.1.4 Yksikön tilan päivittäminen

Yksikön tila voidaan havaita yksikön etupaneelissa olevan valon väristä, operaattorin käyttämän ylläpito- ja hallintakäyttöliittymän laitteistonäkymästä tai käyttöliittymän yksikkötiedoista. Vihreä valo tarkoittaa, että yksikkö on toimintakunnossa (engl. working). Keltainen valo ilmoittaa, että yksikön toimintakyky on heikentynyt (engl. degraded). Punainen valo osoittaa yksikön olevan viallinen (engl. faulty). Ohjelmiston lataus on käynnissä, jos etupaneelin valo vilkuttaa keltaista valoa. Jos vikadiagnostiikka on käsitellyt vian ja tallentanut sääntöjen perusteella yksikön tilan tietokantaan, yksikön tilatieto välitetään ylläpito- ja hallintakäyttöliittymälle. Kuvassa 10 on esitetty yksikön tilan päivitys sekä viankäsittelyn eteneminen tukiasemaympäristössä.



Kuva 10. Hälytystenkäsittely tukiasemasysteemissä.

3.1.5 Hälytyksen raportointi

Hälytys raportoidaan operaattorin ylläpito- ja hallintakäyttöliittymälle ja tallennetaan vikadiagnostiikka-tietokantaan. Hälytyksistä raportoidaan vakavuus, numero, ajanhetki, hälytyksen nimi sekä hälytyksen aiheuttanut yksikkö ja solu. Taulukossa 1 on hälytysten vakavuusasteet, jotka ilmoitetaan operaattorille sekä esimerkki tilanteesta, jolloin kyseinen hälytys lähetetään.

Taulukko 1. Hälytyksen vakavuus

Hälytyksen vakavuus	Esimerkki hälytyksen aiheuttajasta
Tukiasema viallinen	Radiolinkkiyhteyksissä on vikaa
Tukiaseman toiminta heikentynyt	Tukiaseman toinen kello on viallinen
Ilmoitus tukiaseman toiminnasta	Tukiaseman ovi on auki
Solu viallinen	Solun ainut lähetin- ja vastaanottoyksikkö on viallinen
Solun toiminta heikentynyt	Solun lähetin- ja vastaanottoyksikkö on viallinen (varayksikkö toimintakunnossa)
Ilmoitus solun toiminnasta	Lähetin- ja vastaanottoyksikön lämpötila on noussut

3.2 Viankäsittelyn komponentit

Hälytysten käsittely on osa Rhapsody-mallilla kehitetyssä tukiasemaohjelmistoa. Jokaisesta viankäsittelyn komponenteista on muodostettu oma alijärjestelmänsä malliin eli jokainen vian käsittelyn komponentti muodostaa mallin yhden pakettitasoisen osan. Komponentit ovat tiedonkerääjä, esikäsittelijä, viankäsittelijä, luokittelija, raportoija sekä tietokantakäsittelijä. Jokainen paketti käsittää useita luokkia, joista jokaisella on oma tehtävänsä vikadiagnostiikan käsittelyssä. Komponentit toimivat yhdessä muun tukiasemaohjelmiston kanssa. Tukiasemaohjelmisto käsittää useita paketteja, joita ovat esimerkiksi käynnistys, yksiköiden havaitseminen ja tallentaminen tietokantaan, alustus, konfigurointi ja yksikön lämpötilojen havaitseminen.

3.2.1 Tiedonkerääjä

Tiedonkerääjä huolehtii järjestelmän yksiköiden raakahälytysten keräämisestä järjestelmän eri komponenteilta. Se odottaa viestejä ja ohjaa ne oikeille yksiköille. Yksikkö voi olla joko normaalissa toimintatilassa (engl. working), sen toiminta voi olla heikentynyt (engl. degraded) tai se on kokonaan toimintakyvytön (engl. out of order). Tiedonkerääjä vastaanottaa raakahälytyksen ja pyytää tietokantakäsittelijältä hälytyksen toimintasäännöt.

3.2.2 Esikäsittelijä

Koska jokaisesta hälytyksestä on tiedossa yksilöidyt tiedot, esikäsittelijä tarkistaa sääntöjen avulla onko vastaava hälytys tullut jo aikaisemmin. Jos hälytys on tullut jo aikaisemmin ja se on aktiivinen, puskuroidaan samat hälytykset pois. Tämän jälkeen tiedonkerääjä lähettää tietokantakomponentille tiedot aktiivisista hälytyksistä. (Puurunen 2000, 27). Sääntötiedostosta esikäsittelijä lukee hälytyksen viiveen ja tarkistaa onko hälytys ennestään aktiivinen vai onko hälytys cancel-tyyppinen (eli aktiivinen hälytys peruutetaan) sekä päivittää tiedon tietokantaan.

3.2.3 Viankäsittelijä

Vikadiagnostiikka eli viankäsittelijä selvittää hälytyksen aiheuttajan ja poistaa samalla hälytyksestä aiheutuneet seurannaishälytykset. Vikadiagnostiikan alussa tarkistetaan, mikä hälytys oli kyseessä ja mikä on yksikön tila. Jos sama vian numero, sama hälytyksen aiheuttaja ja sama vian vakavuus on jo tietokannassa, hälytys poistetaan. Viat luokitellaan viiteen tasoon (ks. taulukko 1) ja käsittelijä kerää vikatietokannasta vain korkeimman tason vikailmoitukset ja suorittaa kyselyn tietokantakäsittelijälle ja käsittelee vikailmoituksen saamiensa sääntöjen perusteella. Kun vian aiheuttaja on selvinnyt sääntöjen avulla, suoritetaan sääntöjen mukaiset toiminnot kuten resetoinnit tai redundanttisen yksikön ottaminen käyttöön ja päivitetään yksikön tai solun tila tietokantaan. (Puurunen 2000, 27)

3.2.4 Luokittelija

Luokittelijan tehtävänä on seurata ja päivittää tukiaseman yksiköiden tiloja. Kun luokittelija saa tiedon yksikön tilan muuttumisesta, se lähettää tiedon hälytysten raportoinnille. Luokittelija ei ollut käytössä vielä tämän työn aikana, mutta se tullaan ottamaan tukiaseman vikadiagnostiikkaan jatkossa. Hälytykset luokitellaan sääntötiedostoon ns. painotuskertoimilla.

3.2.5 Hälytysten raportoija

Raportoija vastaanottaa hälytyksen ja vertaa sitä hälytyshistoriaan. Se tarkistaa saapuneen hälytysnumeron ja yksikön numeron perusteella, onko hälytys jo aktiivinen ja sen perusteella päättää lähetetäänkö hälytys eteenpäin. Jos hälytys on uusi, raportoija lähettää sen ylemmälle tasolle tukiaseman ylläpito- ja hallintakäyttöliittymälle näytettäväksi. Myös siinä tapauksessa, että yksikön tila muuttuisi esimerkiksi degraded tilasta faulty tilaan, lähetys lähetetään eteenpäin. Hälytystieto sisältää numeron, nimen, ajan ja viallisen yksikön tai solun nimen. (Puurunen 2000, 28).

Jos vika on saatu korjattua viankäsittelyn toiminnoilla, esimerkiksi resetoinnilla, tieto välitetään tietokantaan ja aktiivinen hälytys poistetaan tietokannasta sekä ylläpito- ja hallintakäyttöliittymältä.

3.2.6 Tietokantakäsittelijä

Tietokantakäsittelijä toimii tiedon muuntajana vikadiagnostiikkatietokannan ja komponenttien välillä. Tietokantakäsittelijässä tieto muokataan sellaiseen muotoon, että se on komponenttien hyödynnettävissä ja oikeat tiedot menevät oikeille komponenteille. Käsittelijä vastaanottaa viestin ja suorittaa vaaditun toiminnon, jonka tuloksena saadaan tietoa tietokannasta. Kyselyn tuloksesta rakennetaan ja lähetetään viesti vastaanottajalle. (Puurunen 2000, 28-29)

3.3 Vikadiagnostiikkatietokanta

Tietokantakomponentti sisältää tietoa, joita muut komponentit käyttävät tietokantakäsittelijäkomponentin kautta. Tietokantaan tallennettavia tietoja ovat: raakahälytykset ja geneeriset hälytykset, raportissa oleva hälytystekstit, hälytysraportit sekä jokaisen komponentin omat päättelysäännöt. (Puurunen 2000, 29)

Laitteistotietokanta sisältää puolestaan tiedot tukiaseman yksiköistä, yksiköiden välisistä kaapeloinneista sekä solutiedoista (eli mitkä yksiköt kuuluvat mihinkin soluun). Ns. master ohjainyksikkö pollaa eli tutkii, mitä yksiköitä tukiasemassa on ja välittää tiedon 10 sekunnin välein tietokantaan. Jokaisessa tukiasemassa on yksi master ohjainyksikkö sekä muita aktiivisia yksiköitä kuten lähetys- ja vastaanottoyksiköt, antenniyksiköt, vahvistinyksiköt sekä signaalinsuodatinyksiköt. Passiivisia yksiköitä, joilla ei ole omaa ”älyä” ovat tuuletin, jotka jäähdyttää tukiaseman yksiköitä, kello, signaalinsuodatin, virtalähde, lämmitin ulos sijoitettavassa tukiasemassa sekä kortti, jolta luetaan tukiaseman tyyppi tukiaseman käynnistysvaiheessa. Jos master ohjainyksikkö vikaantuu toinen ohjainyksikkö (slave) ottaa master-aseman haltuunsa.

4 HÄLYTYSTEN TESTAUS TUKIASEMAYMPÄRISTÖSSÄ

WCDMA-tukiaseman vikadiagnostiikan testaus suoritettiin pääasiallisesti lopullisessa ympäristössään eli tukiasemassa. Tällaista testausta kutsutaan target-testaukseksi eli testattavana oli sekä tukiaseman ohjelmistot että itse tuote kaikkine komponentteineen. Järjestelmää, jossa teknologioita kuten elektroniikkaa ja komponentteja on yhdistetty kokonaisuudeksi, ja jota ohjaa mikroprosessori sekä siinä toimiva ohjelmisto, kutsutaan sulautetuksi järjestelmäksi.

Hälytysten testaus perustui määrittelydokumenttiin, joka oli noin 200 sivuinen dokumentti hälytyksistä. Jokainen hälytys on selostettu dokumentissa siten, että siitä selviää mistä, hälytys on aiheutunut, mitä uusia hälytyksiä kyseinen hälytys voi aiheuttaa, mahdolliset muut hälytykset, joista hälytys on voinut generoitua, millä yksiköillä vika ilmenee, mikä ilmoitus lähetetään operaattorille ja miten operaattori voi korjata vian. Operaattorille toimitettava dokumentti hälytysten käsittelystä on hieman suppeampi kuin testeissä käytetty määrittelydokumentti.

Vikadiagnostiikkaa kehitettiin nopeassa tahdissa. Sääntötiedostoa, vikadiagnostiikka-ohjelmistoa, ylläpito-ohjelmistoa sekä ylläpito- ja hallintakäyttöliittymää kehitettiin koko testausvaiheen ajan. Sääntötiedostoa päivitettiin versionhallintaan miltei päivittäin, joinain päivinä päivityksiä saattoi tapahtua useita. Ennen muutoksen päivittämistä ja siirtämistä versionhallintaan, muutokset oli testattava ja todistettava toimiviksi. Testausympäristön laatiminen ja itse testaustapahtuman todentaminen vaati useiden työkalujen käytön hallintaa.

4.1 Työkalut testauksessa

Sääntötiedoston loogisuusvirheet voitiin tarkistaa ajamalla tiedosto NT-ympäristössä, *NT-testerilla*. *Ftp-työkalua* käytettiin tiedostojen siirtämiseen PC:n hakemistosta yksikön flash-muistiin. Varsinaisessa target-testauksessa käytettiin *järjestelmän analysaattori-simulaattoria* viestien lähettämiseen ja ohjelmistojen lataamiseen sekä *tukiaseman ylläpito- ja hallintakäyttöliittymää* tukiaseman kuvan ja hälytysten näyttämiseen, komissioinnin tekemiseen sekä reset-toimintoihin. *UDP-Log työkalulla* tallennettiin testauksen ajonaikaiset tulosteet. *Vikatietokanta-työkaluun* kerättiin löydetyt ohjelmistoviat sekä kirjattiin korjaukset sekä testaustapahtuma suoritetun korjauksen jälkeen. Seuraavaksi esitellään testaustyökalut, niiden käyttö ja merkitys prosessissa.

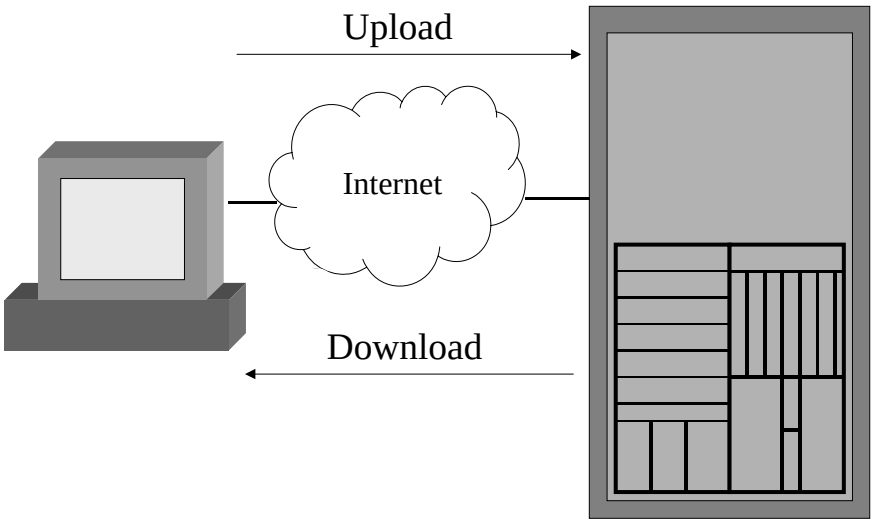
4.1.1 NT-testeri

Rhapsody-mallilla kehitetyllä NT-testausympäristöllä hälytyksiä voidaan simuloida ja suorittaa tarvittavat komponenttitasoiset testaukset. Malliin on kehitetty erillinen testausluokka, jonka avulla voidaan testata määritettyjen testitapausten toimivuus. NT-ympäristössä testataan testitapausten toimintaa simuloimalla sanomia ja seuraamalla tilakaavioista, miten hälytysten käsittely etenee. Ensimmäinen hälytysten testausvaihe oli suoritettu Nokia Networks:lla simuloimalla hälytyksiä tässä ympäristössä. Testitapauksia oli mallissa noin viisikymmentä. NT-testeria käytettiin myös XML-muotoisen sääntötiedoston loogisuusvirheiden kuten pilkkuvirheiden ja sulkuvirheiden etsimiseen.

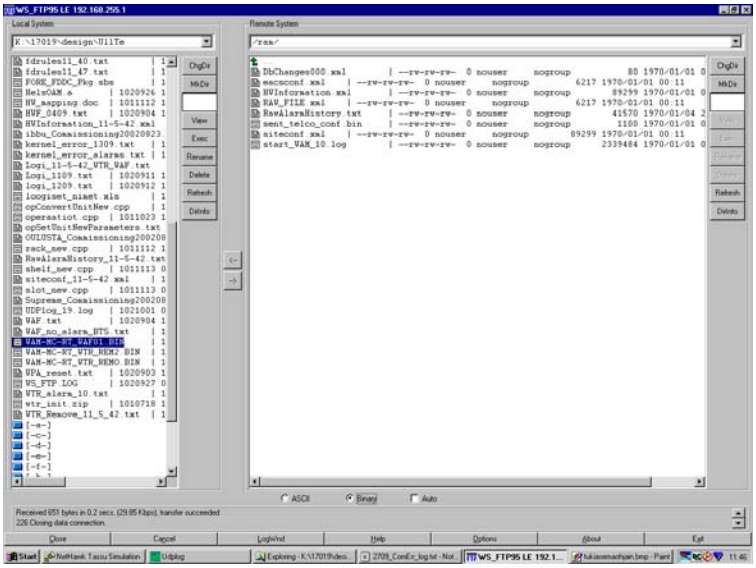
4.1.2 Ftp-työkalu

Tiedostojen siirtämiseen Windows PC:n ja tukiaseman välillä käytetään ftp-työkalua ws_ftp (Winsock File Transfer Protocol). Tiedonsiirto tapahtuu Internet-verkon kautta (Kuva 11). Ohjelmatiedostot ladataan PC:n hakemistolta tukiaseman yksiköille (upload). Tukiaseman yksiköiltä voidaan siirtää tiedostoja kuten ajonaikaisia tietokantatiedostoja PC:lle (download).

Käyttöliittymä on selkeä ja helppokäyttöinen (kuva 12). Ohjelmalle määritetään palvelimen nimi, käyttäjätunnus ja salasana. Nämä tiedot voidaan tallentaa muistiin, jottei niitä tarvitse määrittää joka kerta uudestaan. On vain varmistettava, ettei kukaan ulkopuolinen käytä ohjelmaa. Vasemman puoleisissa ikkunoissa (local system) näkyy PC, jolta otetaan yhteys tukiasemaan. Oikealla olevat ikkunat kuvaavat sen yksikön hakemistoja, joille on otettu yhteys (remote system). Ikkunoiden alla olevat valinnat *ascii, binary* ja *auto* tarkoittavat, missä muodossa tiedostoja siirretään: ascii tarkoittaa tekstitiedostoa (esimerkiksi xml-tiedostot) ja binary yleistä datatiedostoa (bin-tiedostot). Tiedoston uudelleennimeäminen ja näyttäminen sekä poistaminen onnistuu helposti ws_ftp –ohjelmalla. Tiedostojen siirtämisen jälkeen hakemiston näkymä päivitetään *refresh*-painikkeella.



Kuva 11. Tiedonsiirto Internetin kautta PC:n ja tukiaseman välillä.



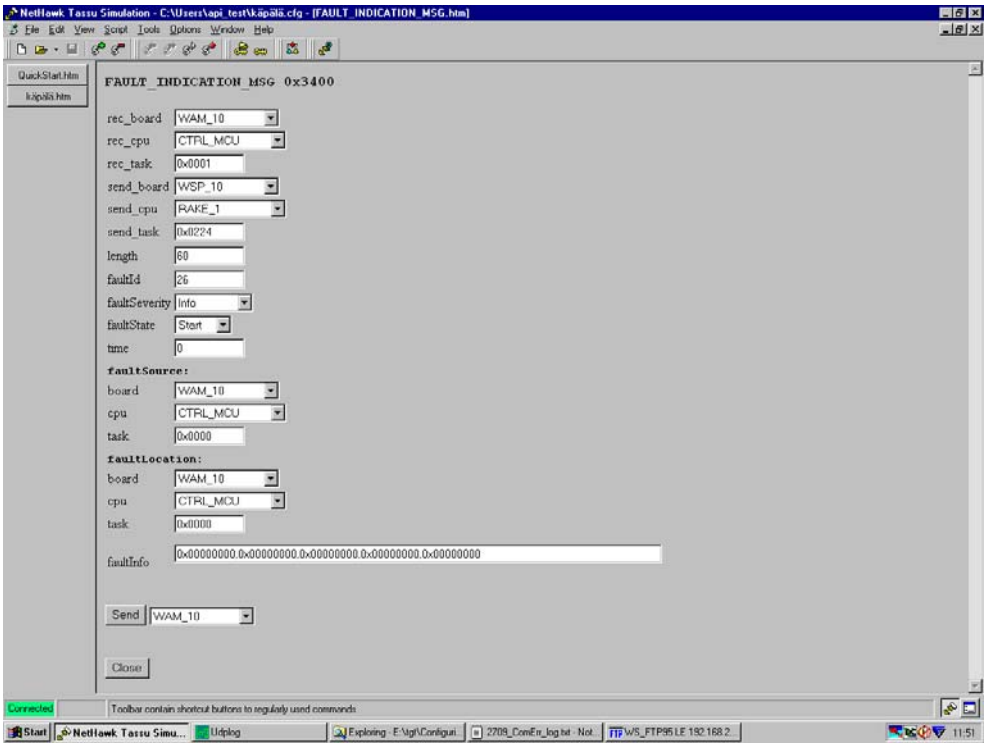
Kuva 12. Ws_ftp -ohjelman käyttöliittymä.

4.1.3 Järjestelmän analysaattori ja simulaattori

Järjestelmän analysaattori ja simulaattoriohjelmaa käytetään PC:lta. Ohjelmalla kommunikoidaan TCP/IP-, ftp- ja UDP-protokollien välityksellä tukiasemaan (DUT, Device Under Test) 100 Mbit/s Ethernet-kaapelilla.

Ohjelmistojen lataus yksiköiden flash-muistiin suoritetaan ftp-protokollaa käyttäen. Ohjelmalla voidaan luoda omia skripteja, joilla viestejä lähetetään tukiasemalle sekä ohjelmistoja ladataan. Hälytyksen testauksessa käytettiin yli kahtakymmentä skriptia, jotka oli jaoteltu vian aiheuttavan yksikön mukaan. Skripti on HTM-muotoinen käyttöliittymäikkuna (kuva 13). Kuvassa on raportoinnin nopeutusviestin lähetys. Viesti lähetettiin tukiasemalle sen ollessa toimintakunnossa. Viestin lähetys nopeutti hälytysten raportointia ylläpito- ja hallintakäyttöliittymälle.

Ohjelmalle voidaan määrittää useita erilaisia konfiguraatioita. Esimerkiksi ohjelmiston lataamiseen ja hälytysviestien lähetykseen käytetään omia konfiguraatioita. Työkalusta tuli projektin aikana käyttöön uusia verisoita ja etenkin konfiguraation muuttaminen ajon aikana, yhteyttä katkaisematta, oli suuri testausta helpottava ja nopeuttava ominaisuus.

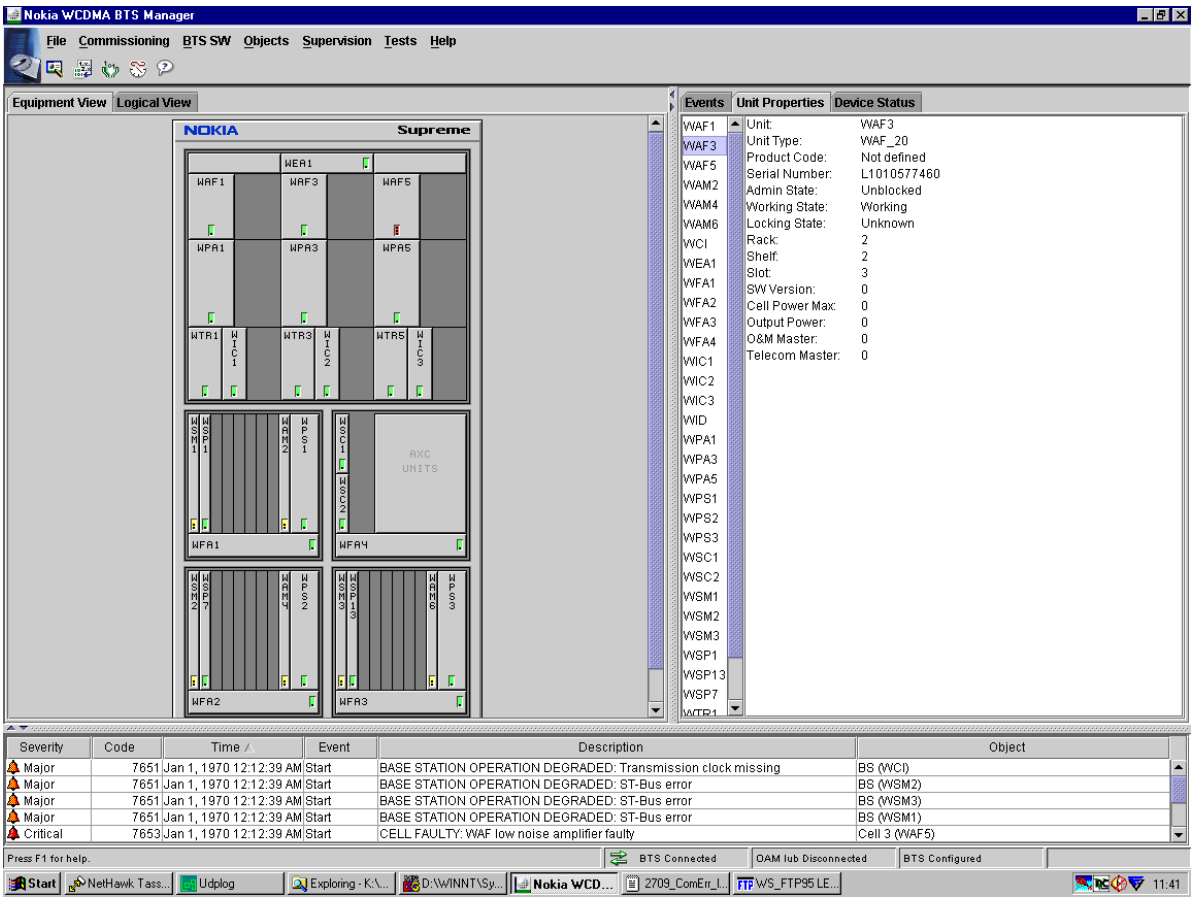


Kuva 13. Ylläpito- ja hallintakäyttöliittymä.

4.1.4 Tukiaseman ylläpito- ja hallintakäyttöliittymä

Tukiaseman ylläpito- ja hallintakäyttöliittymä (kuva 14) on operaattorin käyttöön kehitetty työkalu. Operaattori havaitsee käyttöliittymän kautta hälytykset, jotka on raportoitu tukiasemalta. Operaattorin käytettävissä on dokumentti, jossa kerrotaan miten kunkin hälytyksen tultua on toimittava. Kullekin hälytykselle on selvitetty vian aiheuttajan syy sekä mahdolliset huolto-ohjeet: yksikön vaihto, resetointi, eristys tai esimerkiksi ohjelmiston uudelleenlataus.

Ohjelmalla näytetään tukiaseman kuva, yksiköiden nimet ja niiden tilat (working, degraded tai faulty) sekä tunnistetiedot kuten sarjanumerot sekä hälytyshistoria. Aktiiviset hälytykset näkyvät käyttöliittymän alareunassa siinä järjestyksessä, kun ne ovat syntyneet. Jos hälytys peruutetaan (vika korjaantunut), hälytys häviää näytöltä. Kaikki hälytystiedot tallennetaan hälytyshistoriatietoihin, josta ne voidaan tarvittaessa lukea ja tallentaa tiedostoon.

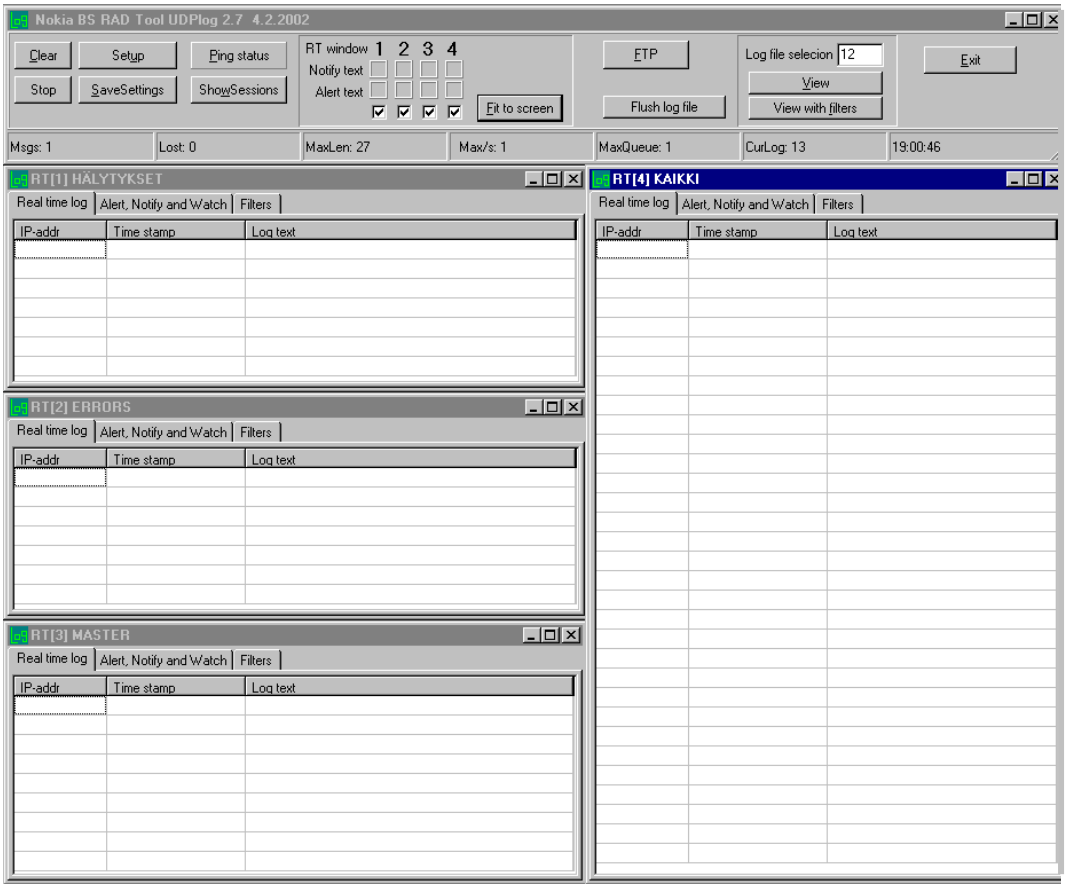


Kuva 14. Tukiaseman ylläpito- ja hallintakäyttöliittymä.

Tukiaseman ylläpito- ja hallintakäyttöliittymän avulla luodaan uusi ns. komissiointitiedosto tai ladataan vanha tiedosto hakemistosta. Tiedostossa määritellään tukiaseman kokoonpanotiedot. Operaattori suorittaa ns. reset-toiminnot sekä ohjelmistolataukset ja –päivitykset tällä työkalulla. Yksittäinen resetoitava yksikkö valitaan työkalun valikosta tai suoraan laitekuvasta hiiren oikeanpuoleisella painikkeella. Solutason resetointi tehdään valitsemalla valikosta resetoitava solu. Tukiaseman resetointi eli virtojen käyttäminen pois suoritetaan tällä käyttöliittymällä.

4.1.5 Ajonaikainen työkalu

Kuvassa 15 on esitetty UDP-protokollaa käyttävä UDP-Log, jonka avulla seurataan tukiasemaohjelmiston toimintaa ajon aikana. Työkalun avulla PC:lle voidaan näyttää ja tallentaa tulostuneet tiedot sekä seurata kontrolliyksiköiden toimintatilaa “pingaamalla” yksiköiden IP-osoitteita (*ping status*).



Kuva 15. Ajonaikainen työkalu.

Vaikka UDP-protokolla ei olekaan aivan luotettava tulosteiden näyttämisessä, osoittautui se tärkeäksi apuvälineeksi testauksessa. Osa tiedoista voi jäädä pois. Työkalun avulla voidaan suodattaa tulosteesta tarpeettomia tietoja pois ja käyttää jopa neljää tulosteikkunaa samanaikaisesti. Kussakin tulosteikkunassa on erinomaiset suodatustoiminnot eli yhteen ikkunaan voidaan ajaa esimerkiksi kaikki tulosteet, toiseen ikkunaan kaikki hälytykset, kolmanteen virhetilanteet tai varoitukset ja neljänteen esimerkiksi master ohjainyksikön tulostamat tiedot.

4.1.6 Vikatietokanta

Testaajien tehtävänä on löytää ohjelmistosta mahdollisimman paljon virheitä. Etenkin systeemitestausryhmä testaa ohjelmistoja systemaattisesti löytääkseen ohjelmistosta virheitä. Virheet kirjataan vikatietokantaan. Virhetilanne dokumentoidaan ja tietokantaan liitetään ajonaikaiset tulosteet, tiedot käytetystä tukiasemasta ja ohjelmistoversiosta, testaajan yhteystiedot sekä vian löytymispäivämäärä. Vikatietokanta toimii niin, että se lähettää automaattisesti sähköpostin välityksellä tiedon uuden ohjelmistovirheen kirjaamisen jälkeen. Virheen korjaaja sovitaan ja vikatietokantaan päivitetään analyysi tilanteesta, korjatut tiedostot versionumeroineen sekä lopuksi kirjataan testaustapahtuma. Jokaisesta viankorjauksen analyysista ja testausraportin kirjaamisesta lähetetään automaattinen sähköpostiviesti muille suunnittelijoille ja testaajille. Sähköpostista käy ilmi vian tila (uusi, vika löydetty, korjattu, testausraporttia muutettu jne.). Yksikkötestauksen suorittaa korjaaja itse, mutta tietyissä tilanteissa systeemitestausryhmä testaa ja todentaa vian poistuneen. Jos systeemitestaus ei voi todeta vian korjaantuneen, tapauksesta tehdään uusi vikaraportti.

4.2 Testausympäristön laatiminen

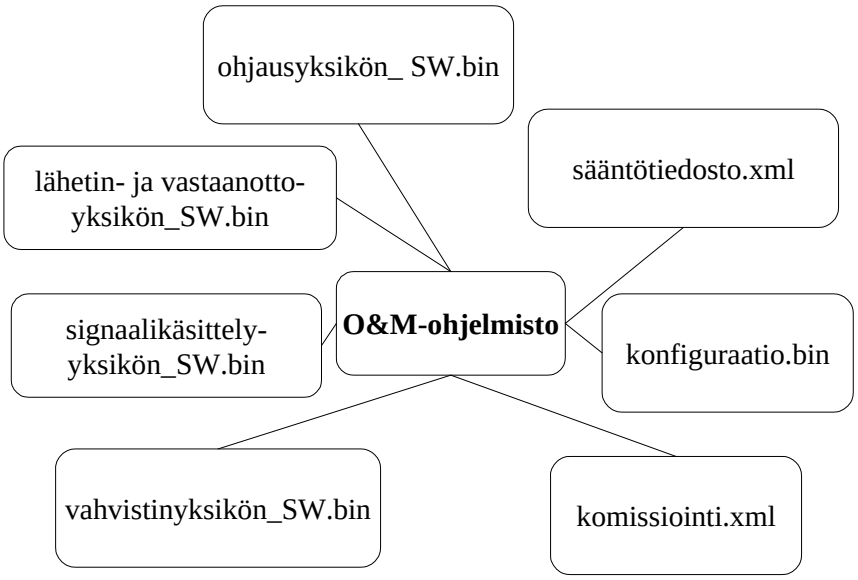
Testaus oli suoritettava aina uusimmalla **ohjelmistokokonaisuudella**. Jos Rhapsody-mallin jotain vikadiagnostiikkakomponenttia, joka on mallissa omana pakettinaan, oli muokattu siten, että se vaati tietyn muotoisen sääntötiedoston, nämä molemmat muutokset oli otettava mukaan testaukseen. Malli käännettiin ja linkattiin, jolloin syntyi tarvittavat O&M SW käsittävät binaaritiedostot. Ohjelmiston binaaritiedosto siirrettiin ensin ftp-

työkalulla yksikön ram-hakemistoon ja sen jälkeen ohjelmistonlataus skriptilla yksikön rom-hakemistoon.

Lisäksi oli valittava oikea tukiaseman **ylläpito- ja hallintakäyttöliittymän versio**, sillä sitä kehitettiin samanaikaisesti O&M SW:n kanssa. Varsinkin testauksen alkuvaiheessa käyttöliittymästä tuli uusi versio miltei jokaisen O&M ohjelmistopaketin mukana ja O&M SW vaati oikean ylläpito- ja hallintakäyttöliittymän version. Kehitystyön alussa O&M SW päivityksiä tuli noin kerran viikossa tai harvemmin, mutta ohjelmistokehityksen siinä vaiheessa, kun ohjelmistoja alettiin toimittamaan asiakkaalle, uusia ohjelmistoversioita tuli kolme kertaa viikossa. Oikean **konfigurointitiedoston** ja **komissiointitiedoston** valitsemisella oli suuri vaikutus testaustapahtumaan. Esimerkiksi väärän komissiointitiedoston ollessa flashilla tukiasema ei käynnistynyt kunnolla ja testaus epäonnistui täysin. Hälytystestaus voitiin aloittaa vasta, kun kaikki edellä mainitut seikat oli otettu huomioon ja ajonaikaisesta tiedostosta tarkistettu, että tukiasema on normaalissa toimintakunnossa.

Tukiasema oli **resetoitava** tai käynnistettävä uudestaan ohjelmistolatauksen jälkeen, jotta ladatut tiedostot latautuivat ajonaikaiseksi ohjelmistoksi. Tukiaseman käyttämät ohjelmistot on esitetty kuvassa 16. Ajon aikana flashille muodostui useita tärkeitä tiedostoja kuten hälytystiedostoja ja tietokantatiedostoja, joita tallennettiin ohjausyksikön flashilta ftp-työkalulla.

Ohjelmistovallinnan lisäksi oli tarkistettava, että kabinetin **kalustus** oli oikein. Joissakin testeissä tukiaseman kaikki solut täytyi olla toiminnassa. Pieniä testauksia voitiin suorittaa myös ns. vajaakalustuksella eli esimerkiksi yhden solun yksiköillä.



Kuva 16. Flashille ladattavat tiedostot.

4.2.1 O&M ohjelmisto

O&M ohjelmistot ladattiin ohjainyksiköille sekä lähetys- ja vastaanottoyksiköille ftp-työkalulla sekä tukiaseman analysaattori-simulaattori ohjelmalla tai suoraan ylläpito- ja hallintakäyttöliittymällä. Ohjelmistot, jotka käännettiin ja linkattiin Rhapsody-mallista binaarimuotoiseksi tiedostoksi, sisältää ohjainyksiköiden, lähetys- ja vastaanottoyksiköiden, vahvistinyksiköiden ja signaalinkäsittely-yksiköiden tarvitsemat tiedostot**.**

Vikadiagnostiikan ottaminen mukaan O&M SW ohjelmiston mukaan tapahtui vaiheittain. Samalla, kun tukiaseman ylläpito- ja hallintakäyttöliittymää kehitettiin, myös hälytysten testaaminen tarkentui ja yksilöityi.

4.2.2 Sääntötiedosto

XML-muotoinen sääntötiedosto, joka on yli 2500 riviä pitkä, käsittää vian käsittelyn ja vian raportoinnin säännöt. Sen muokkaaminen tapahtui Notepad tekstinmuokkausohjelmalla. Yksi yleisin virhe sääntötiedoston muodossa oli, että sulku oli jäänyt pois tai että tiedostossa oli ensimmäinen rivi jäänyt tyhjäksi. Sääntötiedoston

tarkistettiin NT-testerilla. Jos tiedosto oli virheellinen, XML-parserointi ei onnistunut, eikä ohjelmiston hälytysjärjestelmä toiminut lainkaan.

Sääntötiedostossa määritellään jokaiselle hälytykselle viiveet sekä säännöt niiden käsittelyä varten: start ja mahdolliset cancel toiminnot, reset ja palautustoiminnot (engl. recovery) sekä hälytyksen nimi, joka välitetään operaattorin tukiasemaohjaimelle. Joissakin testaustapauksissa haluttua tilannetta ei saatu aikaiseksi ilman sääntötiedoston muokkaamista. Tällöin sääntötiedostosta oli poistettava testauksen ajaksi toisia hälytyksiä, jotka estivät tilanteen muodostumisen. Sääntötiedosto ladattiin ohjainyksikön flashille ftp-työkalulla. Liitteellä 1 on malli sääntötiedostosta.

4.2.3 Komissiointitiedosto

Komissiointitiedosto on XML-muotoinen tiedosto, jossa määritellään tukiaseman konfigurointitiedot kuten kaapelointitiedot. Kullekin tukiasematyypille on oma komissiointitiedosto ja siksi oli käytettävä oikeaa tiedostoa. Tiedostoa muokattiin XML-Notepad ohjelmalla tai se luotiin suoraan tukiaseman ylläpito- ja hallintakäyttöliittymällä. Liitteessä 2 on näkymä XML-Notepad –työkalusta ja esimerkki XML-tiedoston muokkaamisesta.

4.2.4 Konfigurointitiedosto

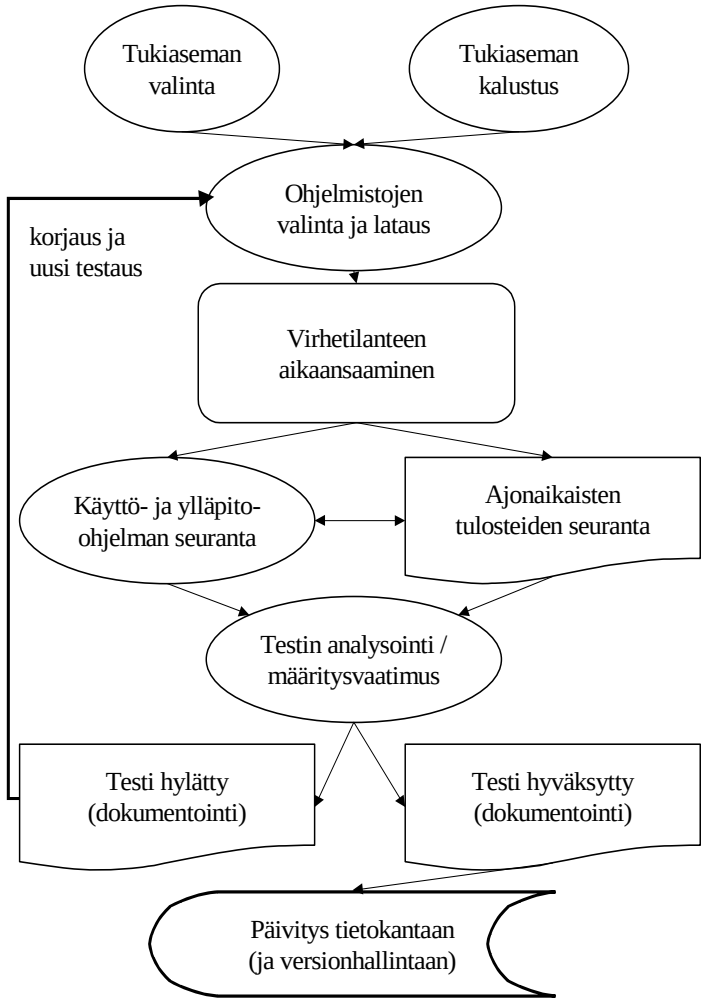
Binaarimuotoisen ohjelmiston konfigurointitiedoston avulla määritettiin, mitä ajonaikaisia tulosteita haluttiin näyttää. Koska UDP-Log –ohjelmalla määritetyn tulosteen maksimikoko oli määritetty, oli tarkkaan harkittava, mitä konfiguraatiota aikoi käyttää. Tiedoston muokkaamiseen oli oma työkalunsa, jolla tiedosto voitiin ottaa ftp-protokollalla muokkaamista varten ohjainyksikön rom-hakemistosta ja siirtää tallennettu tiedosto takaisin rom-hakemistoon. Liitteellä 3 on näkymä konfigurointitiedoston muokkauksesta.

4.3 Hälytystestauksen eteneminen

Tukiasemaohjelmiston testaus vaatii huolellisuutta ja järjestelmällisyyttä. Osa hälytyksistä oli suunniteltu vain tietyille tukiasemille, joten tukiasema valittiin testattavien hälytysten perusteella. Tukiasema (kabinetti) oli kalustettava siten, että haluttu vikatilanne saatiin aikaiseksi. Prosessin eteneminen on esitetty kuvassa 17. Jos testi meni läpi, testi dokumentoitiin ja kirjattiin testitietokantaan. Jos sääntötiedostoa päivitettiin, versiopäivityksestä ilmoitettiin sähköpostitse kaikille tukiasemaohjelmiston suunnittelijoille, ei ainoastaan vikadiagnostiikassa mukana olleille henkilöille.

Testin hylkäämiseen saattoi olla monia syitä; hälytys kohdistui väärään yksikköön, yksikön tila ei päivittynyt, sääntötiedostossa esitetyt viankäsittelyt eivät toimineet määritetyllä tavalla, tiedot näytettiin väärin tai puutteellisesti ylläpito- ja hallintakäyttöliittymällä tai hälytys ei tullut lainkaan käyttöliittymälle saakka.

UDP-protokollalla toimivalla UDP-Log työkalulla voitiin näyttää tukiasemaohjelmiston ajonaikaiset tulosteet, joita malliin oli määritetty. Testausta varten jouduttiin Rhapsody mallia usein muokkaamaan niin, että halutut tulosteet saatiin UDP-Logissa näkyviin ja voitiin seurata testin etenemistä näiden tulosteiden avulla. UDP-Logissa on määritetty maksimitulosteiden määrä, ja jos konfigurointitiedostosta oli valittu liian paljon tulostettavia asioita, osa jäi täysin tulostumatta.



Kuva 17. Tukiaseman hälytystestauksen eteneminen.

Hälytyksiä testattiin järjestelmällisesti. Ensimmäisessä target-testausvaiheessa tarkistettiin järjestelmän analysaattori ja simulaattori –työkalua käyttäen, että hälytykset oli määritelty kullekin yksikölle oikein ja niiden käsittely käynnistyi ja raportointi eteni oikein. Työkalun käyttö oli helppoa ja kätevää, tosin alussa oli epäselvää, mitkä kaikki skriptien kentät oli täytettävä ennen viestin lähetystä, sillä kaikki kentät eivät olleet aktiivisia. Alkuvaikeuksien jälkeen hälytysten lähettäminen tukiasemalle oli helppoa ja jos hälytyksen lähetys jostain syystä epäonnistui, se oli helposti nähtävissä ajonaikaisesta tulostetyökalusta, sillä onnistunut hälytyksen lähetys tallentui UDP-Logiin.

Operaattorin käyttämä ylläpito- ja hallintakäyttöliittymä oli tärkeä työväline hälytysten testaamisessa. Tukiaseman ylläpito- ja hallintakäyttöliittymän avulla tarkistettiin, että hälytykset näkyivät oikein ja yksiköiden ja solujen tilat päivittyivät hälytysten tultua aktiivisiksi. Jos hälytys peruutettiin oli todennettava, että yksikön tila palasi

toimintakuntoon ja hälytys poistui aktiivisten hälytysten listalta. Taulukossa 2 on esimerkkejä ohjainyksikön, kabinetin oven ja kellon vikaantumisesta. Yksikkö, hälytys ja hälytysilmoitus raportoidaan operaattorin käyttöliittymälle. Huolto-ohje selviää operaattorille toimitetusta hälytysdokumentista.

Taulukko 2. Esimerkki todellisesta viasta.

Yksikkö	Hälytys	Vian ilmeneminen ja näyttö käyttöliittymällä	Huolto-ohje
Ohjainyksikkö	Ohjelmiston lataus epäonnistui	Tukiaseman toiminta heikentynyt Valo punainen	• Resetoi yksikkö • Tarkista tiedostot • Vaihda yksikkö
Ovi	Tukiaseman ovi auki	Ilmoitus tukiaseman toiminnasta	• Sulje ovi
Kello (ei redundattista yksikköä)	O&M ei löydä yksikköä tietokannasta	Tukiaseman viallinen	• Tarkista liitynnät • Vaihda yksikkö

5 TESTAAJIEN KOKEMUKSIA

Vikadiagnostiikkaa ja lähemmin hälytysten käsittelyä tukiasemassa testattiin Nokia Networks:lla Oulussa sekä Elektrobit Oy:n Kajaanin yksikössä. Hälytysten testaajia oli yhteensä kuusi henkilöä, joista kaksi kuului systeemitestausryhmään. Esitin muille testaajille kysymyksiä testausprosessiin liittyen. Kyselin heiltä yleisellä tasolla heidän kokemuksiaan hälytystestauksesta, mahdollisia ongelmatilanteista, joihin he olivat testaamisen aikana joutuneet sekä mahdollisia kehitysehdotuksia ja tulevaisuudennäkymiä liittyen vikadiagnostiikan testaamiseen. Yhteydenpito testaajien ja suunnittelijoiden kesken käytiin sähköpostitse.

Opastus testaustyöhön

Suurin osa testaajista oli sitä mieltä, että opastus oli ollut riittävää. Koska kaikissa testauspaikoissa ei ollut vikadiagnostiikan asiantuntijoita, neuvoja jouduttiin kysymään sähköpostitse. Alussa testaajat eivät osanneet kysyä apua riittävän ajoissa ja testit saattoivat sen vuoksi viivästyä. Vastaukset kysymyksiin ja tarvittavat testausohjeet saatiin kuitenkin nopeasti.

Dokumentointi

Suurimpana puutteena pidettiin yksityiskohtaisten testitapausten puuttumista. Testitapausten avulla testaaminen olisi ollut joustavampaa ja tehokkaampaa. Myös testitulosten kirjaaminen testitietokantaan ei kaikilta testauspaikkakunnilta onnistunut teknisten puutteiden vuoksi. Testitulokset kirjattiin Word-dokumenttiin, jotka lähetettiin sähköpostitse Nokia Networks:lle testitietokantaan tallennettavaksi. Hälytystenkäsittely-

dokumentin päivityksestä tiedotettiin varsinkin projektin alkuvaiheessa puutteellisesti, mikä aiheutti joskus epäselvyyksiä testien suorittamisessa. Testaaja ei tiennyt vian vaatimusmäärittelyjen muuttuneen lukiessaan vanhentunutta dokumenttia. Testaajien mielestä sääntötiedoston katselmoinnit olivat puutteellisia.

Testauksessa ilmenneet vaikeudet

Ongelmaksi oli koettiin ajoittainen testipaikkojen (tukiasemien ja PC:den) vähyys, mikä viivästytti testausta. Projektin aikana myös täysin vialliset tukiasemat estivät testauksen suorittamisen. Vialliset (tai vanhantyyppiset) yksiköt aiheuttivat myös ongelmia.

Testitapausten puuttuminen aiheutti sen, että sääntötiedostoa oli muokattava, jotta vika saatiin syntymään ja hälytys lähtemään eteenpäin. Koska RNC-yhteyttä (Radio Network Controller) ei ollut käytettävissä, testaaja ei voinut tarkistaa menikö hälytys RNC:lle asti.

Joidenkin työkalujen käyttökoulutus olisi ollut tarpellista, sillä varsinkin alussa niitä ei osannut hyödyntää tarpeeksi tehokkaasti. Esimerkiksi järjestelmän analysaattori- ja simulaattori -työkalun käyttökoulutus projektin alkuvaiheessa olisi helpottanut käytännön toimissa.

Testaajien mukaan uusien asioiden ja teknologioiden opettelu kiireisen aikataulun vuoksi ei ollut aina mahdollista. Samalla tämä vaikeutti ja hidasti testauksen suorittamista ja samalla tärkeiden perustietojen omaksuminen saattoi viivästyi.

6 JOHTOPÄÄTÖKSET

Tarkkojen *testausohjeiden* olemassaolo sekä niiden säännöllinen päivitys tehostaisi testaamista; turhat epäselvyydet eivät haittaisi testaamista, testaaminen nopeutuisi ja epätietoisuus siitä, että oliko testaustapahtuma riittävä, jäisi pois. Myös tarkat testitapaukset, joiden mukaan testaus suoritettaisiin ja vika saataisiin aikaan, helpottaisivat ja tehostaisivat testaamista. Näiden *testitapaus-dokumenttien ja hälytystenmäärittely-dokumentin päivittäminen* on työlästä, mutta saatava hyöty panoksiin nähden olisi huomattavaa. Joidenkin *työkalujen käyttökoulutus* olisi ollut hyödyllistä jo hälytystestauksen alkuvaiheessa. Monet epäselvyydet olisivat jääneet pois ja työskentely olisi ollut rutiininomaisempaa ja tehokkaampaa; testaaja olisi voinut keskittyä olennaiseen eli itse testaamiseen. Myös hälytysteorian ja hälytystenkäsittelyn teoriatietojen opiskelu oli aikaa vievää ja kiireellisen aikataulun vuoksi se ei ollut riittävää; testaaja ei aina ehtinyt omaksua tarvittavaa teoriatietoa. Vikadiagnostiikan koulutus olisi tehostanut omalta osaltaan käytännön testaamista.

Myös hälytysdokumentin päivityksestä versionhallintaan tulisi ilmoittaa sähköpostitse kaikille. Jokaisella testaajalla pitäisi olla mahdollisuus testitietokannan käyttöön ja dokumenttien lukemiseen sekä testien kirjaamiseen. Dokumentointi helpottuisi ja testauksesta vastaisi testaaja alusta loppuun saakka. Mahdolliset virhekirjaukset vähenisivät.

7 POHDINTAA

Vikadiagnostiikka on aivan oma osa-alueensa ohjelmistokehityksessä ja tulevaisuudessa sen merkitys kasvaa. Teknologian kehittyessä laitteiden toimintavarmuutta ja –helppoutta arvostetaan ja siksi laitevalmistajat kiinnittävät entistä enemmän huomiota vikadiagnostiikan kehittämiseen jo kaupallisten näkökohtien vuoksi. Myös asiakkaiden toiveet laitteiden kunnossapidon helppouteen asettaa omat vaatimukset vikadiagnostiikan kehittämiselle. Vian löytäminen ja siitä ilmoittaminen ennen kuin se ehtii vahingoittamaan muuta laitteistoa, on vikadiagnostiikan tärkein tehtävä.

Vikadiagnostiikka on monelle käsitteenä tuntematon, niin kuin minullekin aloittaessani WCDMA-tukiasemien hälytysten testaamisen. Kiinnostuin hälytysten käsittelystä hälytysten testauksen yhteydessä. Vikadiagnostiikka on tutkittu paljon ja WCDMA-tukiasemien vikadiagnostiikka tutkimus on kehittynyt 1990-alupuolelta GSM-tukiasemien hälytyskäsittelystä kohti tukiasemasukupolvesta riippumatonta suuntaa. WCDMA-tukiaseman vikadiagnostiikkaa on kehitelty tiiviisti 1990-luvun lopusta lähtien Nokia Networks:lla ja VTT Electronics:lla. On syntynyt vikadiagnostiikan malli, joka on helposti muokattavissa.

Tukiaseman vikadiagnostiikan (hälytysten) käsittelyssä on otettava huomioon monia käytännön seikkoja. Ohjelmistovalinnat, tietämys tukiaseman toiminnasta, hälytysten käsittelystä ja sen teoriasta sekä testaustyökaluista vaikuttavat testauksen onnistumiseen. Vikadiagnostiikan toiminnat on sääntötiedostossa, josta vikadiagnostiikka ne lukee ja suorittaa tarvittavat toimenpiteet.

Itse testausprosessi on rutiininomainen toimenpide, jos ohjelmistot ja tukiasema ovat toimintakunnossa. Testaus suoritettiin hälytysdokumenttiin nojautuen uusinta sääntötiedostoa käyttäen. Sääntötiedostoa jouduttiin usein muokkaamaan testattavien hälytysten osalta siten, että testitilanne saatiin aikaiseksi. Testaustilanne ja testitapahtumat olisi hyvä olla kaikkien testaajien tiedossa. Testin onnistuessa testitapahtuma kirjattiin tietokantaan. Kaikilla testaajilla ei ollut kuitenkaan pääsyä testitietokantaan ja dokumentoinnin suoritti joku toinen. Dokumentoinnin tulisi olla järjestelmällistä, jolloin testaajat voisivat tarkistaa, mitkä testitapaukset on testattu ja mitkä on testaamatta. Turhilta epäselvyyksiltä vältyttäisiin. Testitietokantaa käytettiin projektin alussa. Projektin edetessä testausprosessi on osa käyttö- ja ylläpito-ohjelmiston testaamista ja kaikki vikaraportit kirjataan samaan vikatietokantaan, jonne myös testaustiedot lisätään tehdyn korjauksen jälkeen.

Hälytyksiä on testattu tämän työn valmistuessa kymmenen kuukauden ajan. Työkalut ovat kehittyneet ja niistä on tullut useita toisistaan poikkeavia versioita useita. Työkaluihin perehtymiseen ei ole jäänyt tarpeeksi aikaa kiireisen aikataulun vuoksi. Tehokkaalla työkalujen käyttökoulutuksella työskentely tehostuisi ja paineet vähenisivät. Vaikka työ oli ajoittain raskasta, niin etenkin vikadiagnostiikan testaamisessa iloisin asia on ollut, kun on saanut aiheutettua tukiasemalle todellisen vian ja sen jälkeen on voinut todentaa viankäsittelyn toimineen toivotulla tavalla.

LÄHTEET

Batzman, 2000. Sample software system test plan for a new application. www-dokumentti. <http://members.tripod.com/~bazman/planframe.html?button3=Sample+Software+Test+Plan > (Luettu 26.8.2002).

Forsell, M. 2001. Ohjelmistotuotanto TIE330. Jyväskylän Yliopisto. www-dokumentti. <http://193.167.110.132/marko.forsell/S2001_Luku_08.pdf> (Luettu 29.8.2002).

Frisk, E., 2001. Residual generation for fault diagnosis. Linköping : Linköping University.

Guckenbiehl, T., Malik, A., Milde, H., Neumann, B. & Struss, P., 2000. Integrating model-based diagnosis techniques into current work processes. www-dokumentti. <http://www.radig.informatik.tu-muenchen.de/forschung/qreason/Publications/2000/content.pdf > (luettu 30.9.2002).

Harju, H. 1999. Ohjelmiston luotettavuusarvioinnit. VTT. www-dokumentti. <http://www.pcuf.fi/sytyke/lehti/kirj/st19992/15.pdf> (Luettu 30.9.2002)

Kautto, T. 1996. Ohjelmistotestaus ja siinä käytettävät työkalut. www-dokumentti. <http://www.mit.jyu.fi/opiskelu/seminaarit/ohjelmistotekniikka/testaus/ > (Luettu 26.8.2002).

Kärki, S. & Karjalainen, S. 1999. VTT Rakennustekniikka, Ilmastointijärjestelmän vikadiagnostiikka ja menetelmät ja sovellukset. www-dokumentti. <http://www.inf.vtt.fi/pdf/tiedotteet/1999/T1967.pdf> (Luettu 29.08.2002)

Laine, H. 1998. Ohjelmistotuotanto. Testaus 1. www-dokumentti. <http://www.cs.helsinki.fi/u/laine/ot/s98/testaus1.pdf > (Luettu 15.10.2002)

Pressman, R. 2000. Software Engineer, A Partioner’s Approach. Viides painos. McGraw-Hill International (UK) Limited.

Puurunen, M. 2000. Hälytysten luokittelu geneerisessä vikadiagnostiikka-arkkitehtuurissa. Diplomityö. Luottamuksellinen vuoteen 2005 saakka. Oulun yliopisto. Prosessi- ja sähkötekniikanosasto.

Salminen, J-P. 1996. Mallipohjainen hälytystenkäsittely matkapuhelinverkon tukiasemassa. Diplomityö. Oulun yliopisto. Sähkötekniikanosasto.

TTKK, 2000. Vikojen huomioon ottaminen. (Luettu 8.10.2002). <http://www.fhpa.fi/fpf/arkisto/seminaarit/viat/viat.html>

Tuotekehitys ja laatu, 1999. Opetusmateriaali. www-dokumentti. <http://ylivieska.cop.fi/amp99/opiskelumateriaali/Tuotekehitys/Opintomateriaali/t18.ppt>(Luettu 8.10.2002).

LIITTEET

Liite 1: Kuvitteellinen malli sääntötiedostosta.

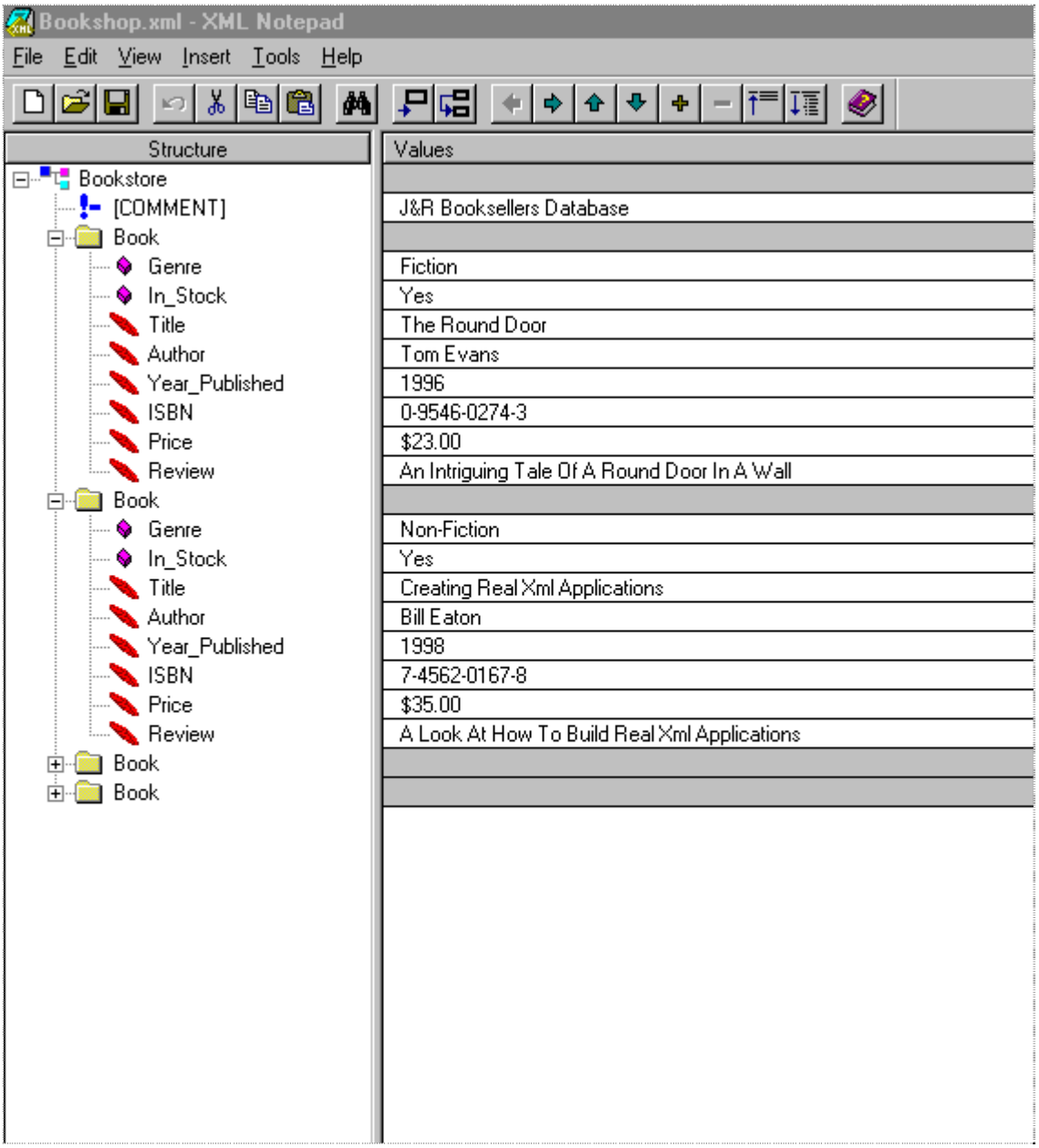
Liite 2: XML-tiedoston muokkaamiseen käytetty XML-Notepad ohjelma. Esimerkki kirjakaupan tietokannasta.

Liite 3: Tukiaseman konfigurointitiedoston muokkaus.

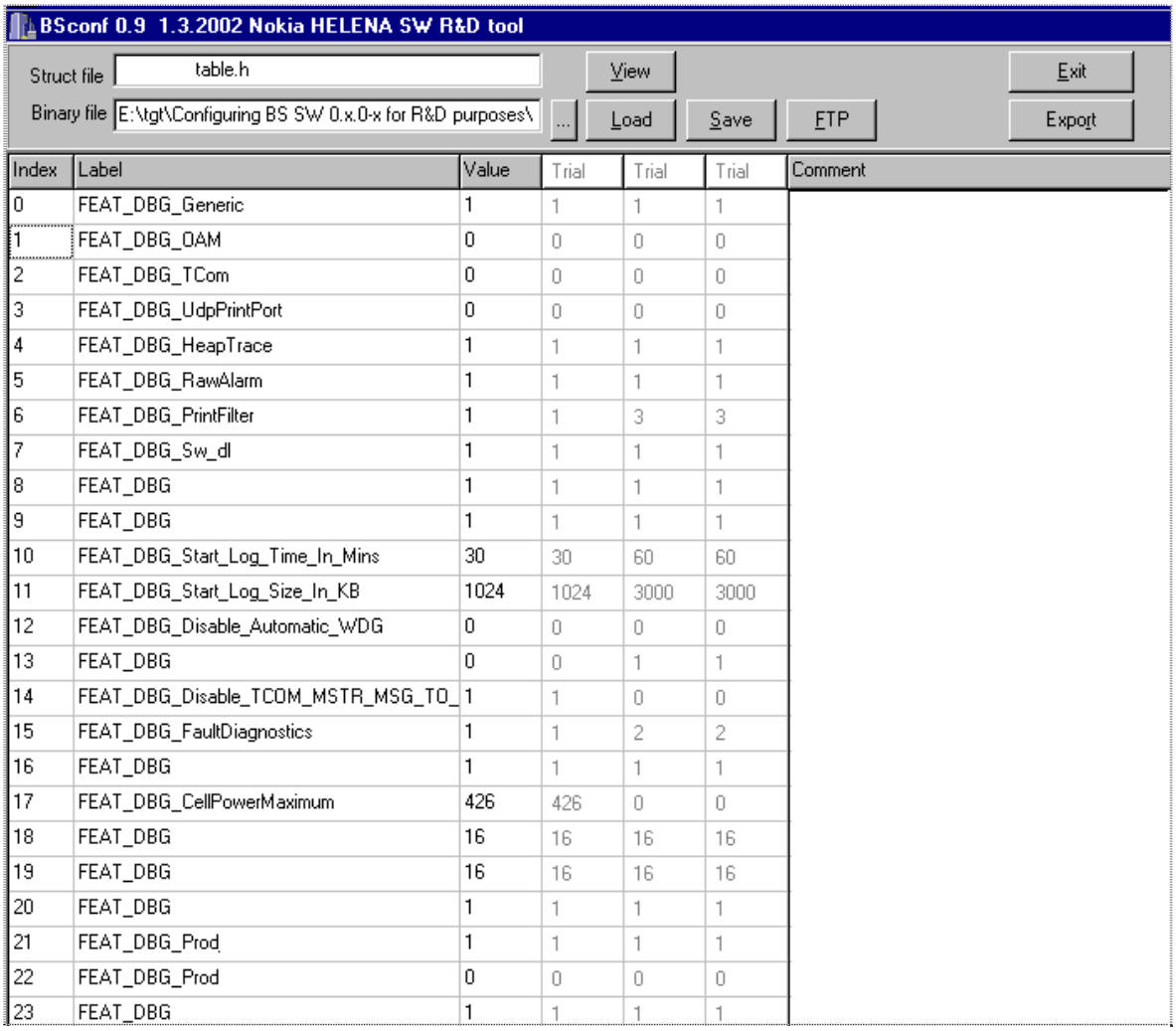
Kuvitteellinen malli sääntötiedostosta.

```
<?xml version="1.0"?>

<VIAN_KASITTELIJA>

 <OPINNAYTETYO="TEKIJA" >
  <VERSIO>1</VERSIO>
 </OPINNAYTETYO>

 <OPINNAYTETYO="ESIKASITTELIJA" >
  <SAANNOT>
  (SAANTO viive
     (0; paivita; (); -1; 1)
     (1; peruutus; (); 4; 2)
     (2; odota; 3; 3)
     (3; peruutettu; (); -1; 4)
     (4; laheta; (); -1; -1)
  )
  </SAANNOT>
 </OPINNAYTETYO >

 <OPINNAYTETYO="KASITTELIJA" >
  <SAANNOT>
  (SAANTO no_out_of_order_start
     (0; yksikon_tila; (@syy; FAULTY); -1; 1)
     (1; paivita; (@syy; FAULTY); 2; -1)
     (2; laheta; (no; OUT_OF_ORDER; @FaultActivity; @syy; @numb); -1; -1)
  )
  </SAANNOT>
 </OPINNAYTETYO >

 <OPINNAYTETYO="MAPPAYS" >
  <SAANNOT>
  (MAPPAYS viive saannoille
  1 5 10 )
 </SAANNOT>
 </OPINNAYTETYO >

 <OPINNAYTETYO="RAPORTOIJA" >
  <TEKSTI>
  (0,"Kellon alustus epäonnistui")
  (1,"Ohjelmiston epäonnistui")
  (2,"Yksikön lämpötila kohonnut")
  (3,"Ovi auki")
  </TEKSTI>
 </OPINNAYTETYO >
</VIAN_KASITTELIJA>
```

XML-tiedoston muokkaamiseen käytetty XML-Notepad ohjelma. Esimerkki kirjakaupan tietokannasta.



Tukiaseman konfigurointitiedoston muokkaus.



BSconf 0.9 1.3.2002 Nokia HELENA SW R&D tool

Struct file: table.h — View — Exit

Binary file: E:\tgt\Configuring BS SW 0.x.0-x for R&D purposes\ — ... — Load — Save — FTP — Export

Index	Label	Value	Trial	Trial	Trial	Comment
0	FEAT_DBG_Generic	1	1	1	1	
1	FEAT_DBG_OAM	0	0	0	0	
2	FEAT_DBG_TCom	0	0	0	0	
3	FEAT_DBG_UdpPrintPort	0	0	0	0	
4	FEAT_DBG_HeapTrace	1	1	1	1	
5	FEAT_DBG_RawAlarm	1	1	1	1	
6	FEAT_DBG_PrintFilter	1	1	3	3	
7	FEAT_DBG_Sw_dl	1	1	1	1	
8	FEAT_DBG	1	1	1	1	
9	FEAT_DBG	1	1	1	1	
10	FEAT_DBG_Start_Log_Time_In_Mins	30	30	60	60	
11	FEAT_DBG_Start_Log_Size_In_KB	1024	1024	3000	3000	
12	FEAT_DBG_Disable_Automatic_WDG	0	0	0	0	
13	FEAT_DBG	0	0	1	1	
14	FEAT_DBG_Disable_TCOM_MSTR_MSG_TO_	1	1	0	0	
15	FEAT_DBG_FaultDiagnostics	1	1	2	2	
16	FEAT_DBG	1	1	1	1	
17	FEAT_DBG_CellPowerMaximum	426	426	0	0	
18	FEAT_DBG	16	16	16	16	
19	FEAT_DBG	16	16	16	16	
20	FEAT_DBG	1	1	1	1	
21	FEAT_DBG_Prod	1	1	1	1	
22	FEAT_DBG_Prod	0	0	0	0	
23	FEAT_DBG	1	1	1	1	