Esa Salminen

Luotettavuuslaboratorion tieto- ja testiautomaatiojärjestelmän vaatimusmäärittely

Metropolia Ammattikorkeakoulu

Insinööri (AMK)

Automaatiotekniikka

Insinöörityö

27.9.2015

Metropolia

Tekijä(t) Otsikko Sivumäärä Aika	Esa Salminen Luotettavuuslaboratorion tieto- ja testiautomaatiojärjestelmän vaatimusmäärittely 41 sivua + 1 liite 27.9.2015
Tutkinto	Insinööri (AMK)
Koulutusohjelma	Automaatiotekniikan koulutusohjelma
Suuntautumisvaihtoehto	Automaation tietotekniikka
Ohjaaja(t)	Diplomi-insinööri Juha Sunio Insinööri Esa Harjunpää Lehtori Antti Liljaniemi

Insinöörityön aiheena oli tehdä luotettavuuslaboratorion tieto- ja testiautomaatiojärjestelmän vaatimusmäärittely. Työn tavoitteena oli verrata tehtyä vaatimusmäärittelyä järjestelmän käyttäjien todellisiin tarpeisiin ja näin tunnistaa järjestelmästä ne kohdat, jotka ovat kehittämisen tarpeessa.

Vaatimusmäärittely tehtiin kahdessa osassa. Ensin käytettiin tietojärjestelmän vaatimusmäärittelyyn hyväksi havaittuja toimintamalleja soveltuvin osin. Näihin toimintamalleihin lukeutuivat käyttäjäryhmien tunnistaminen ja käyttötarinoiden keräys. Käyttötarinoista tunnistettiin ne tietojärjestelmän toiminnot, joita käyttävät tarvitsevat hoitaakseen luotettavuuslaboratoriossa jokapäiväisiä työtehtäviään. Näitä toimintoja verrattiin olemassa olevaan tietojärjestelmään ja tuloksena saatiin selville, mitä puutteita siinä on tällä hetkellä. Kävi ilmi, että tietojärjestelmän tietoja lukevassa web-pohjaisessa käyttöliittymässä oli puutteita asiakkaalle tarjottavassa näkymässä sekä testisuunnitelmien ja -raporttien hyväksymiseen liittyvässä toiminnallisuudessa.

Vaatimusmäärittelyn toinen osa liittyi testiautomaatiojärjestelmän toimintojen parantamiseen. Testiautomaatiojärjestelmän käyttäjiltä kerättiin järjestelmään liittyviä parannusehdotuksia, jotka mahdollistaisivat sen käytön entistä tehokkaammin ja käytännöllisemmin. Parannusehdotukset priorisoitiin, jotta saataisiin hyvä yleisnäkemys ja keskustelua siitä, mitkä vaatimukset ovat toteuttamisen arvoisia. Tässä työssä ei tehty päätöstä vielä siitä, mitkä toiminnot tulevat uuteen järjestelmään sisältymään.

Työssä pohdittiin myös tulevan testiautomaatiojärjestelmän mahdollista arkkitehtuuria, josta esitettiin yksi mahdollinen toteutusmalli. Tässä mallissa automaatioverkon testilaitteita hallinnoitaisiin tietokoneella, joka toimii web-palvelimena. Näin testilaitteiden hallinnointi onnistuisi myös oman henkilökohtaisen tietokoneen kautta selaimella, eikä testilaitteiden hallinnointi olisi pelkästään valvomokoneen takana.

Tämä insinöörityö on osa alkanutta tieto- ja testiautomaatiojärjestelmän uudistushanketta ja järjestelmän kehitys tulee pohjautumaan tässä työssä esitettyihin kohtiin.

Avainsanat	vaatimusmäärittely, testiautomaatio, tietojärjestelmä

Author(s) Title Number of Pages Date	Esa Salminen Requirement specification of the reliability laboratory's information and test automation system 41 pages + 1 appendix 27.9.2015
Degree	Bachelor of Engineering
Degree Programme	Automation Technology
Specialisation option	Automation Information Technology
Instructor(s)	M.Sc. Juha Sunio B.Sc. Esa Harjunpää Lecturer Antti Liljaniemi

The purpose of this thesis was to make a requirements specification to the reliability laboratory's information and test automation system. This requirement specification was then compared to the true needs of the system users. As a result of this comparison, targets of the development actions were revealed.

The requirement specification was made in two parts. In the first part, requirement specification of the information system was made by using pre-approved design models. These design models included identification of user groups and collecting user stories. User stories were then analyzed and the needed functionality of the system was extracted from these stories. This functionality was then compared to the current system. As a result, the comparison revealed what functions the current system did not retain. It turned out that there was incomplete functionality in the web-based interface. This incomplete functionality included client side interactions and the process that concerned the approval of test plans and reports.

The second part of the requirement specification focused on improving the functionality of the test automation system. Reliability laboratory's personnel was interviewed to collect ideas and suggestions to improve the current systems efficiency and usability. Ideas and improvement suggestions were also prioritized to get a good general view and discussion about which ideas and suggestions are worth implementing. Decisions regarding those implementations was not in the scope of this thesis and therefore, it was not made.

This thesis includes also some thoughts about the new system and its architecture. One possible model of the architecture is also introduced. In this model, automation network's devices are controlled with computer that also acts as a web-server. This way, tester management could be achieved with everyone's personal laptop by using a web browser. This would be a more flexible method compared to a control computer with fixed position.

This thesis is a part of regeneration project of the reliability laboratory's information and test automation system. Futures development actions will be based on requirement specification presented in this thesis.

Keywords	requirement specification, test automation, information system

Sisällys

Lyhenteet

1 Johdanto 1

1.1 Työn vaiheet 2
1.2 Luotettavuuslaboratorio lyhyesti 2

2 Luotettavuus ja sen testaaminen 3

2.1 Luotettavuus käsitteenä 3
2.2 Luotettavuustestaus 3
2.2.1 Nostimen luotettavuustestaus 4
2.2.2 Komponenttien luotettavuustestaus 5

3 Testiautomaatiojärjestelmän kehityskaari 7

3.1 Automaattisen testauksen aloitus 7
3.2 Ensimmäiset ohjattavat logiikat 7
3.3 Päivitys automaatiojärjestelmäksi 8
3.4 Automaatiojärjestelmän jatkokehitys 10
3.5 Testiautomaatiojärjestelmän päivitys nykyiseen muotoon 11

4 Tietojärjestelmähankkeen valmistelu 13

4.1 Valmistelun yleiskuvaus 13
4.1.1 Valmistelun käynnistys 14
4.1.2 Järjestelmävaatimuksien määrittely 14
4.1.3 Perusarkkitehtuurin suunnittelu 15
4.1.4 Hankinnan mitoitus 16
4.1.5 Läpiviennin suunnittelu 16
4.1.6 Hankintasuunnitelman viimeistely 17
4.2 Tietojärjestelmän vaatimuksien määrittely 17
4.2.1 Käyttäjäryhmien kuvaus 18
4.2.2 Käyttötarinat 18
4.2.3 Kohdesanaston laatiminen 18
4.2.4 Käsitemalli 19
4.2.5 Toiminnallisten vaatimusten kuvaaminen 19

5 Luotettavuuslaboratorion tietojärjestelmän vaatimusmäärittely 20

5.1 Käyttäjäryhmät 20
5.2 Järjestelmän toiminnot 22
5.3 Vaatimusmäärittelyn vertaus nykyisen järjestelmän toimintoihin 24

6 Testiautomaatiojärjestelmän kehitys 26

6.1 Testiautomaatiojärjestelmän kehitystarpeiden määrittely 26
6.1.1 Käyttäjien haastattelut ja vaatimusten priorisointi 26
6.1.2 Vaatimusten kuvaus 28
6.2 Uuden testiautomaatiojärjestelmän kuvaus 32
6.2.1 Testiautomaatiojärjestelmän laitteistoarkkitehtuuri 32
6.2.2 Valvomosovelluksen toiminnallinen kuvaus 36
6.2.3 Testerin toiminnallinen kuvaus 37
6.3 Ajatuksia uuden testiautomaatiojärjestelmän kehittämisestä 38

7 Yhteenveto 39

Lähteet 41

Liitteet

Liite 1. Käyttäjähaastatteluista kerätyt vaatimukset

Lyhenteet

.NET	Microsoftin kehittämä ohjelmistokehys
IEC 61131	IEC:n (International Electrotechnical Commission) määrittämä standardi ohjattaville logiikoille
ASi	Actuator Sensor interface. Kenttäväyläprotokolla
ASP.NET	Active Server Pages. Microsoftin kehittämä web-ohjelmistokehys
HART	Highway Addressable Remote Transducer. Kenttäväyläprotokolla
HMI	Human-Machine Interface. Ihmisen ja koneen välinen käyttöliittymä
I/O	Input/Output. Ohjausta suorittavan laitteen tulot ja lähdöt
OLE	Object Linking and Embedding. Microsoftin kehittämä teknologia, joka mahdollistaa oliopohjaisten käsitteiden yhdistämisen.
OPC	OLE for Process Control. Avoimen tiedonsiirron standardi
PLC	Programmable logic controller. Ohjelmoitava logiikka
SCADA	Supervisory Control and Data Acquisition. Valvomo-ohjelmisto, jonka kautta ohjataan ja kerätään tietoa automaatiojärjestelmästä
SQL	Structured Query Language. IBM:n kehittämä standardoitu kyselykieli, jolla relaatiotietokantaan voi tehdä erilaisia hakuja, muutoksia ja lisäyksiä

1 Johdanto

Tässä insinöörityössä tehdään kohdeyrityksen luotettavuuslaboratorion tietojärjestelmään liittyvä vaatimusmäärittely, sekä verrataan sitä olemassa olevaan tietojärjestelmään. Vertailun tuloksena saadaan hyvä kokonaiskuva siitä, miten hyvin nykyinen tietojärjestelmä vastaa luotettavuuslaboratorion käyttäjien nykyisiä tarpeita. Tietojärjestelmän yhtenä osa-alueena on luotettavuustestejä ohjaava ja niistä tietoa keräävä testiautomaatiojärjestelmä. Myös tämän järjestelmän toiminnallisuutta tarkastellaan ja kehitetään nykyhetken käyttäjävaatimusten kautta.

Työssä käsiteltävä tietojärjestelmä pitää sisällään web-pohjaisen testi- ja resurssienhallintatyökalun, testiautomaatiojärjestelmän sekä näiden tietovarastona toimivan tietokannan. Testi- ja resurssienhallintatyökalun kautta hallinnoidaan suoritettavia testejä ja sitä käytetään apuvälineenä tietojen hakemiseen, kun testistä kirjoitetaan asiakkaalle raporttia.

Testiautomaatiojärjestelmä – joka on yksi tietojärjestelmän osa-alue – on alunperin suunniteltu nostinten ohjaukseen. Myöhemmin testaustarve on monipuolistunut ja samaa järjestelmää on alettu hyödyntää laajemmin esimerkiksi erilaisten komponenttitestien ohjaamiseen. Alkuperäisestä käyttötarkoituksestaan poikkeavien testien ohjaamisessa on kuitenkin se huono puoli, että järjestelmää tuntematon käyttäjä ei ilman kokeneen käyttäjän hiljaista tietoa tiedä minkälaisilla ehdoilla tai käskyillä testejä ohjataan. Tämä johtuu siitä, että testien ohjaus- ja vikaantumisasetukset on suunniteltu pääsääntöisesti vain nostintesteille. Työn yhtenä tavoitteena onkin selvittää millainen järjestelmä vastaisi paremmin erilaisia testaustarpeita. Testaustoiminnan kehittyessä, testiautomaatiojärjestelmän käyttäjät ovat tehneet lukuisia parannusehdotuksia. Näiden ehdotusten toteuttamista järjestelmään tutkitaan myös yhtenä osa-alueena tätä työtä.

1.1 Työn vaiheet

Luotettavuuden määrittelyä koskevaa teoriaosuutta käydään läpi kappaleessa 2. Kappaleessa 3 käydään läpi nykyisen testiautomaation kehityskaari. Tietojärjestelmän määrittelyn teoriaa on käyty kappaleessa 4, jossa Forseliuksen (2013) esittämästä tietojärjestelmän määrittelyyn liittyvästä mallista on tiivistetty oleellisimmat pääkohdat. Luotettavuuslaboratorion tietojärjestelmään liittyvää vaatimusmäärittelyä on toteutettu kappaleessa 5 käyttäen osittain hyväksi kappaleessa 4 läpikäytyjä periaatteita. Lisäksi, kappaleessa 5 verrataan olemassa olevan järjestelmän toimintoja tehtyyn vaatimusmäärittelyyn. Kappaleessa 6 pohditaan kuinka käyttäjien tekemiä parannusehdotuksia voitaisiin järjestelmään liittää ja minkälainen kokonaisuus uudesta järjestelmästä tulisi. Lopuksi esitetään yhteenveto kappaleessa 7 siitä, mitä työn aikana on tullut tehtyä.

1.2 Luotettavuuslaboratorio lyhyesti

Luotettavuuslaboratorion tärkein olemassaolon tarkoitus on tutkia kokonaisten laitteiden ja yksittäisten komponenttien luotettavuutta. Tämänhetkinen testattavien laitteiden skaala pitää sisällään isoimmillaan köysinostimen, joka kykenee nostamaan kymmenien tonnien kuorman. Skaalan toisessa päässä ovat yksittäiset pienet komponentit, kuten esimerkiksi releet ja painonapit. Näiden lisäksi, luotettavuuslaboratorion muut vastuualueet ovat moottoreiden ja moottorijarrujen testaukset, sekä materiaalitutkimus ja -analyysi.

2 Luotettavuus ja sen testaaminen

2.1 Luotettavuus käsitteenä

Luotettavuudesta on olemassa monenlaisia tulkintoja, mutta yleensä sillä tarkoitetaan systeemin kykyä toimia häiriöttömästi. O'Connorin ja Kleynerin (2012: 1) mukaan luotettavuus voidaan määritellä seuraavasti: todennäköisyys sille, että esine/asia toimii määrätyissä olosuhteissa, määrätyn aikaa, vaatimusten mukaisesti ja vikaantumatta.

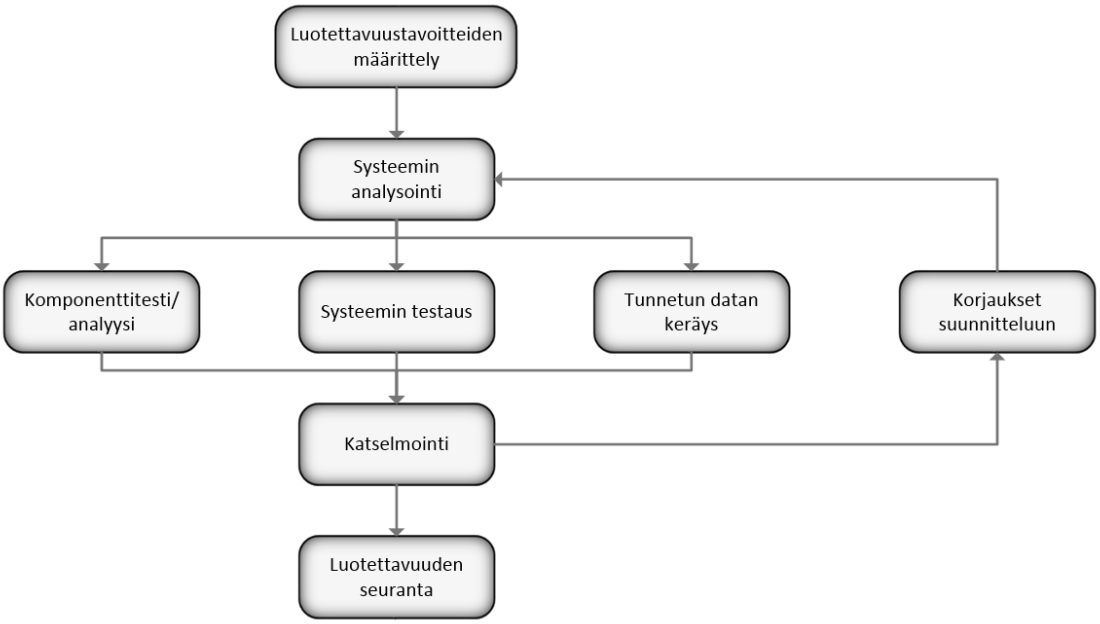
Luotettavuustekniikan kehitys alkoi voimakkaasti kasvaa 1950-luvulla, kun sotilastekniikan monimutkaisten elektroniikkajärjestelmien luotettavuutta selvitettiin. Tämän jälkeen luotettavuustekniikka on otettu käyttöön monella muullakin tekniikan alueella, kuten ilmailussa, ydinvoimatekniikassa, prosessiteollisuudessa ja tietoliikennetekniikassa. (Jokinen 2010: 127.)

Järjestelmän luotettavuus voidaan määritellä sen yksittäisten komponenttien luotettavuuden perusteella. Järjestelmän rakenne määrittelee sen, miten yksittäisen komponentin vikaantuminen vaikuttaa koko järjestelmän toimintaan. Järjestelmän luotettavuutta selvitettäessä laaditaan siitä luotettavuusmalli, joka kertoo järjestelmän ja sen komponenttien välisen toiminnallisen riippuvuuden. Tästä mallista pyritään selvittämään koko järjestelmän luotettavuus pienempien osakokonaisuuksien tai yksittäisten komponenttien luotettavuuksien perusteella. Yksittäisten komponenttien luotettavuuden arviointivirheet aiheuttavat luotettavuuslaskuihin epätarkkuutta. Tämä johtuu yleensä siitä, että tilastomateriaali on puutteellista tai hajonta on suuri. (Jokinen 2010: 127.)

2.2 Luotettavuustestaus

Luotettavuustestaus on yksi yleisimmin käytetyistä luotettavuuden analysointimenetelmistä. Se voi sisältää yksittäisten komponenttien testaamista, mutta myös kokonaisten systeemien testaamista. Luotettavuustestaus voidaan luokitella kvalitatiiviseen ja kvantitatiiviseen testaukseen. Kvalitatiivisessa (laadullisella) testauksessa keskitytään lähinnä löytämään vikamekanismit ja heikoimmat lenkit. Vikamekanismi kuvaa vian juurisyyn ja siitä seuranneen tapahtumaketjun, joka johtaa lopulliseen vikaantumiseen. Kvantitatiivisella (määrällisellä) testauksella pääpaino on vikaantumisen tilastollisella analyysillä ja laitteen vikaantumishetken tiedostamisella.

Luotettavuuden testaamisen tulisi olla hallittu kokonaisuus, jossa testaustarve on hyvin suunniteltu. Kuvassa 1 on näytetty luotettavuusanalyysin prosessikaavio, jossa on kuvattu luotettavuusanalyysin päätoiminnot. Ennen kuin analyysia ruvetaan tekemään, on määriteltävä tavoitteet, jotka halutaan saavuttaa. Tämän jälkeen suoritetaan tarvittava testaus ja datan keräys, joista saatuja tietoja ja tuloksia verrataan asetettuihin tavoitteisiin. Mikäli tavoitetta ei saavuteta, on testattava kokonaisuus analysoitava heikkojen kohtien selvittämiseksi. Analyysin jälkeen voidaan palata piirustuspöydän ääreen parantamaan niitä. Kun heikot kohdat on saatu karsittua iteroimalla prosessikaavion testaus-analysointi-korjaus -kohtia, siirrytään luotettavuuden seurantaan, joka tapahtuu taustalla tuoteryhmän elinkaaren ajan.



Kuva 1. Luotettavuusanalyysin prosessikaavio (Reliability Design 2013).

2.2.1 Nostimen luotettavuustestaus

Nostimen luotettavuustestaus on kokonaisen systeemin testaamista. Kokonaisen nostimen testaaminen liittyy sen suunnittelun verifiointiin, eli toisin sanoen varmistetaan se, että tuote kestää suunnitellun elinikänsä suunnitellulla rasituksella. Tämä toteutetaan luotettavuuslaboratorion testeissä useimmiten siten, että nostimeen kiinnitetään nostimen kapasiteetin mukainen paino, jota ajetaan testipaikalla edestakaisin ylös ja alas. Tässä testissä luotettavuustavoitteiksi asetetaan yleensä tietty määrä ajotunteja tietyllä

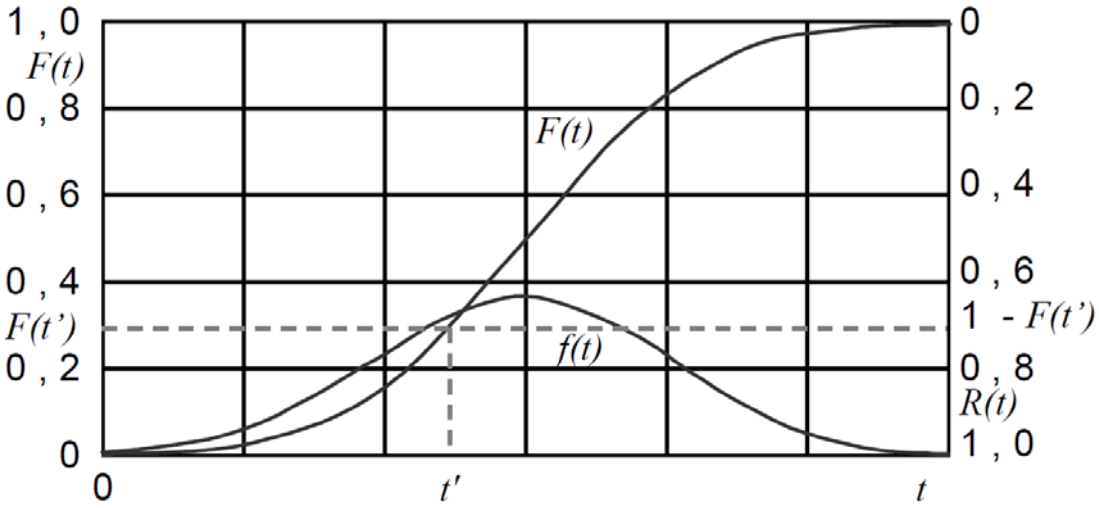
ajoittaiskäyttökertoimella (ED % -lukema). Esimerkiksi 40 %:in ajoittaiskäyttökertoimella ajava nostin ajaa 4 minuuttia ja pitää 6 minuuttia taukoa.

2.2.2 Komponenttien luotettavuustestaus

Yksittäisten komponenttien testaamisen tarkoituksena on tunnistaa luotettavat komponenttivalmistajat ja vertailla eri komponenttimallien luotettavuutta. Samalla on myös tarkoitus kerätä ja arkistoida tietoa eri komponenttien eliniästä ja vikaantumisominaisuuksista. Näitä tietoja käyttäen voidaan arvioida kokonaisen laitteen luotettavuutta kun tunnetaan sen yksittäisten komponenttien luotettavuus.

Komponenttien luotettavuustestauksen yhtenä tarkoituksena pyritään saamaan selville testattavan komponentin elinikä. Jos näitä tuloksia ei ole ennestään saatavilla, on järjestettävä testi, jossa joukko komponentteja ajetaan nimellisrasituksella elinikänsä loppuun asti. Testin tavoitteena on saada komponenttien vikaantumisen ajanhetket selville. Näistä tiedoista voidaan johtaa komponentin vikakertymä ja vikatiheys ajan funktiona.

Otetaan tästä esimerkkinä testi, jossa joukko N samanlaisia komponentteja laitetaan elinikätestiin nimellisrasituksella. Testiä jatketaan niin kauan, että kaikki komponentit ovat vikaantuneet. Testin jälkeen voidaan tulosten perusteella piirtää kuvan 2 mukaiset käyrät vikakertymästä $F(t)$ ja vikatiheydestä $f(t)$. Vikakertymä on siis integraali vikatiheyden funktiosta ajan suhteen ja sen arvo on maksimissaan 1 (=kaikki komponentit ovat vikaantuneet). Vikatiheyden funktiosta voidaan havaita vikaantumisen todennäköisyys valitulla ajanhetkellä. (Jokinen 2010: 128.)



Kuva 2. Vikakertymä $F(t)$ ja vikatiheys $f(t)$ ajan funktiona (Jokinen 2010: 128).

Tulosten perusteella voidaan arvioida testijoukkoon kuulumattoman samanlaisen komponentin vikaantumisen todennäköisyys. Todennäköisyys sille, että komponentti vikaantuu ennen ajanhetkeä t', on $F(t')$. Todennäköisyys sille, että komponentti ei vioitu ennen ajanhetkeä t', on $1 - F(t) = R(t)$. Funktio $R(t)$ on komponentin luotettavuuden funktio ajan suhteen. (Jokinen 2010: 128.)

3 Testiautomaatiojärjestelmän kehityskaari

Tässä kappaleessa käydään läpi olemassaolevan testiautomaatiojärjestelmän kehityskaari. Kovin yksityiskohtaisiin järjestelmäkuvauksiin ei tässä kappaleessa mennä vaan tarkoitus on enemmänkin antaa yleisluontoinen kuva järjestelmän kehityksestä.

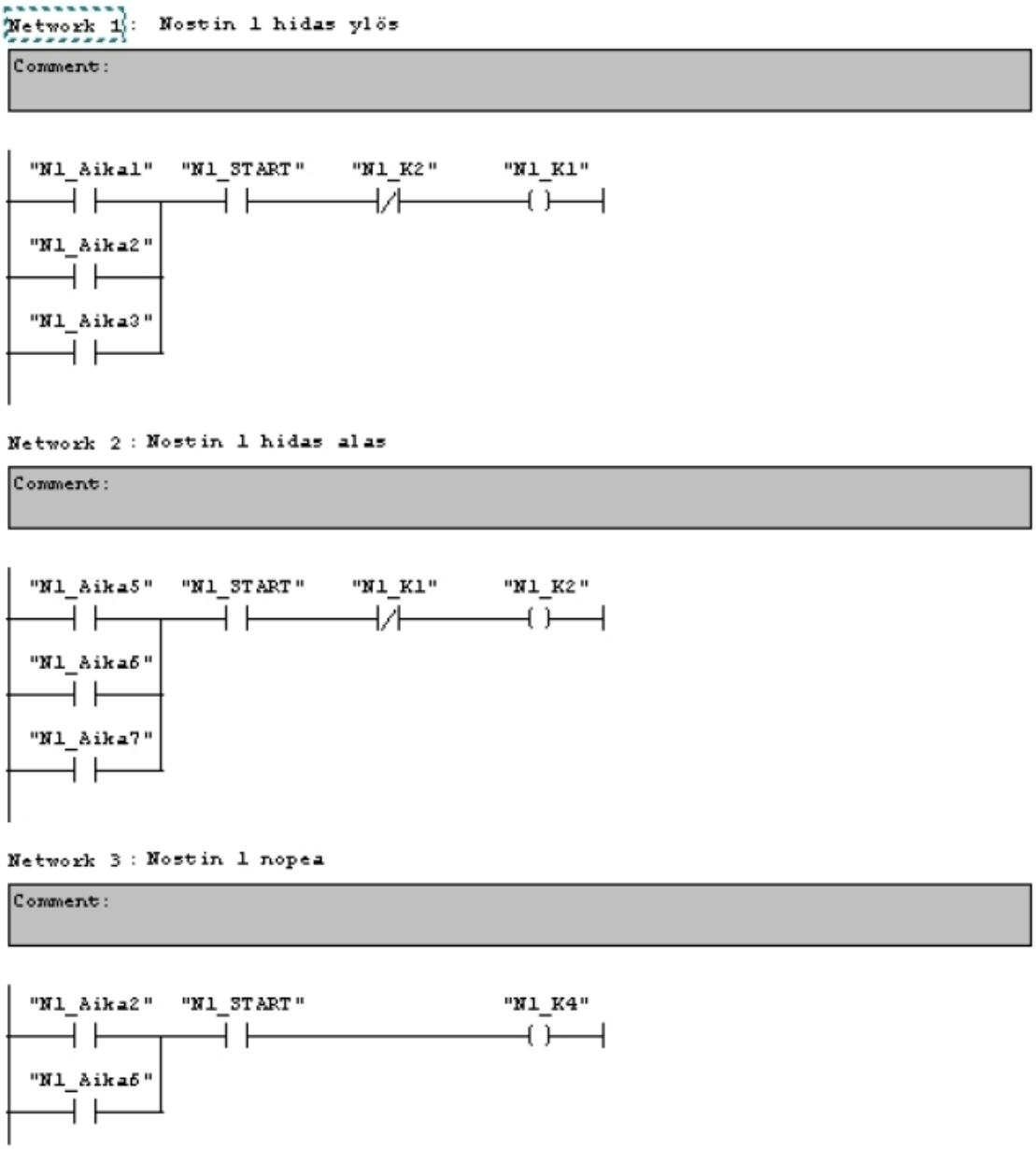
3.1 Automaattisen testauksen aloitus

Automaattinen testaus on alkanut luotettavuuslaboratoriossa 1980-luvulla, jolloin ensimmäisen nostintestin automaatio hoidettiin sähkömekaanisilla releillä ja ajastimilla. Testattavan nostimen ajoaikaa ja syklejä, jotka olivat testissä ainoat tallennettavat tiedot, laskettiin mekaanisilla tuntimittareilla ja laskureilla. Mekaaniset laitteet kuitenkin välillä vikaantuivat. Tätä ongelmaa yritettiin korjata lisäämällä testiin rinnakkaisia mittareita ja laskureita, joiden antamista arvoista pääteltiin tulokset, joiden uskottiin olevan lähellä oikeita. (Asiantuntijahaastattelu 2015a.)

3.2 Ensimmäiset ohjattavat logiikat

1990-luvun lopussa koeajettavien nostinten määrä kasvoi niin suureksi, että releillä toteutettua automaattitestausta jouduttiin laajentamaan. Samalla tehtiin päätös siirtyä releohjauksesta ohjelmoitaviin logiikoihin (PLC). Uusittuun järjestelmään hankittiin tuolloin kaksi PLC:tä, joilla pystyi ohjaamaan yhteensä kahta kymmenen nostimen testipenkkiä. Myös tiedonkeruussa tapahtui kehitystä päivityksen myötä. Uusitulla järjestelmällä voitiin eritellä nostimen ajamasta syklistä ajoajat (nopea, hidas ja lepo) sekä startit (hidas ja nopea ajo). Päivityksestä huolimatta tiedonkeruu jouduttiin tekemään silti manuaalisesti PLC:ltä tietokoneen ja sarjakaapelin avulla. (Heinonen 2003.)

Kuvassa 3 on näytetty pieni osa silloisen ohjauksen PLC-ohjelmasta, joka toteutettiin Siemensin ohjattavilla logiikoilla ja STEP 7 -ohjelmointiympäristöllä. Ohjelmassa on toteutettu yhden nostimen nosto ja laskuliikkeet.

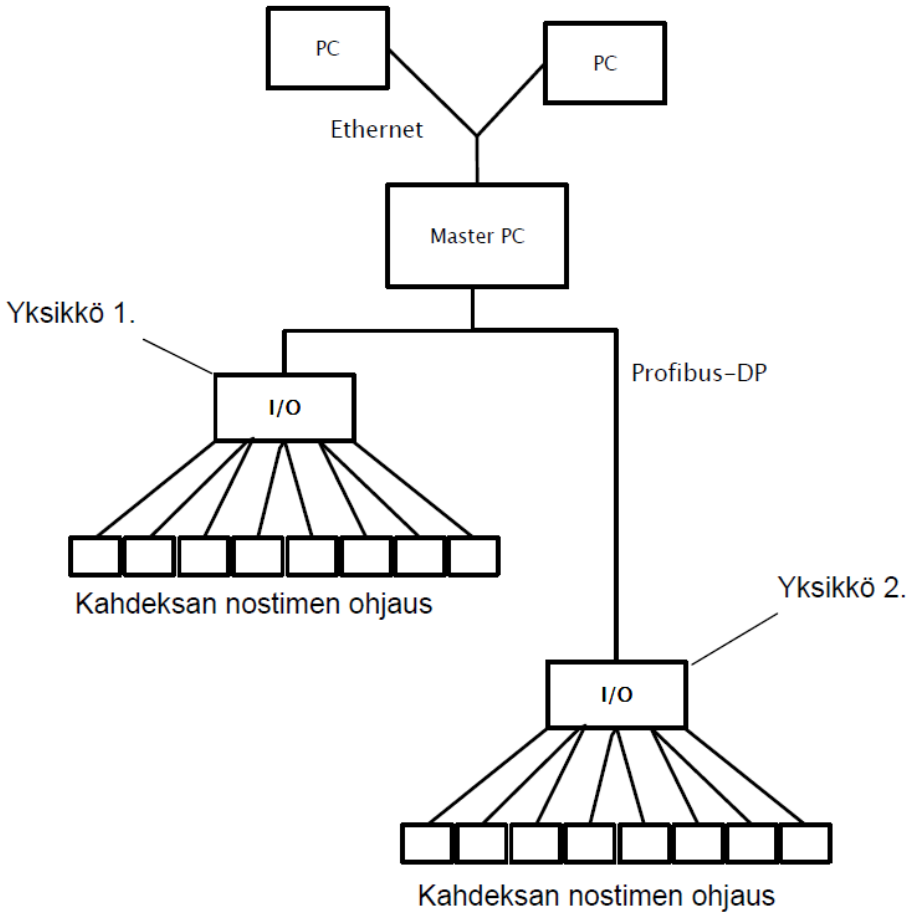


Kuva 3. Siemens STEP 7:lla tehty PLC-ohjelman osa (Heinonen 2003: 9).

3.3 Päivitys automaatiojärjestelmäksi

2000-luvun alkupuolella järjestelmä uusittiin täysin insinöörityön yhteydessä (Heinonen 2003). Vasta tämän päivityksen jälkeen voitiin puhua aidosta automaatiojärjestelmästä. Kuvaan astuivat tuolloin Wonderware:n FactorySuite-ohjelmisto (joka koostui InControl-, InTouch- ja InSQL-ohjelmista), Profibus-DP -väylä ja väylään kytkettävät Beckhoff:n

I/O-moduulit. Järjestelmän rakenne on esitetty kuvassa 4. Järjestelmässä olevien koeajopaikkojen määrä oli uudistetussa järjestelmässä 8 + 8, eli kaksi testipenkkiä, joissa kummassakin oli 8 koeajopaikkaa.



Kuva 4. Järjestelmän arkkitehtuuri 2000-luvun alun päivityksen jälkeen (Heinonen 2003: 10).

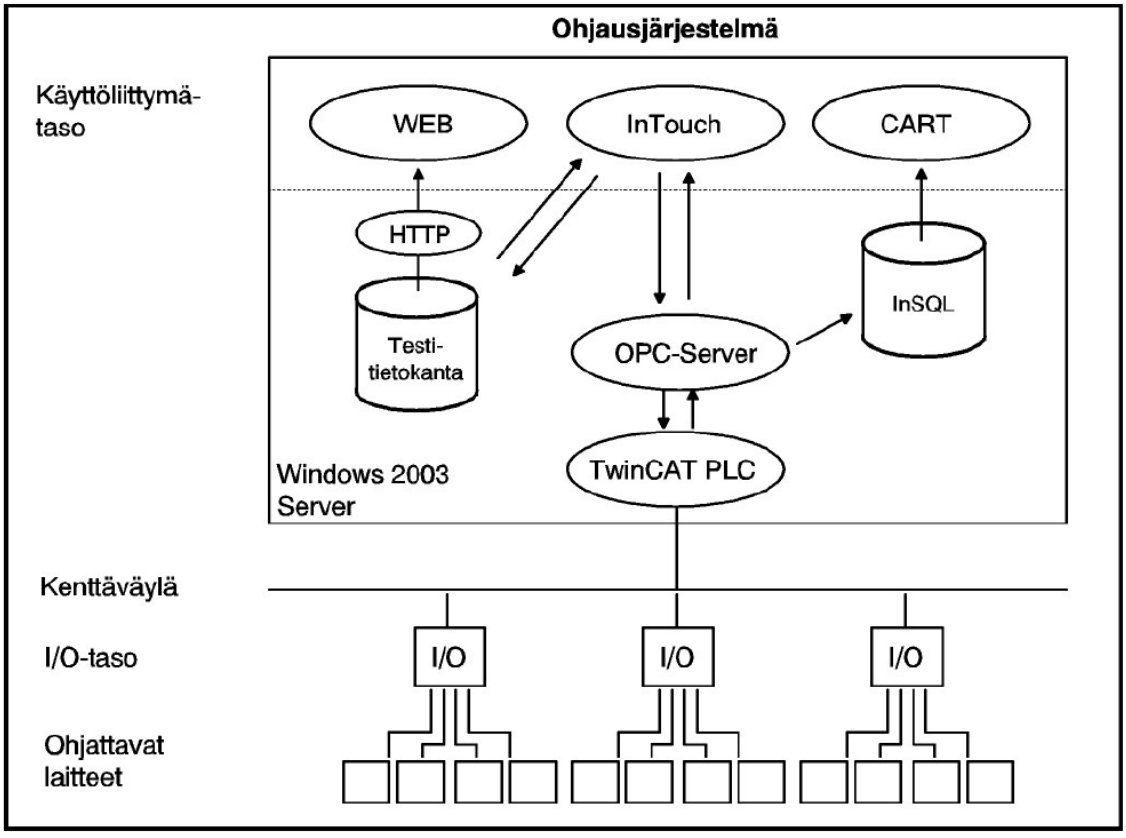
Järjestelmän pääkomponentteina olivat Master PC, Profibus DP -kenttäväylä ja I/O-moduulit. Master PC:ssä olevista ohjelmistoista InControl suoritti IEC 61131-3 -ohjelmointistandardin mukaista logiikkaohjelmaa, InTouch:ia käytettiin käyttöliittymänä ja InSQL tallensi ajonaikaiset tiedot omaan tiedostojärjestelmäänsä, josta tiedot tallennettiin edelleen toiseen tietokantaan jatkokäsiteltäväksi. Kenttäväylän kautta Master PC ohjasi I/O-moduuleita, jotka oli kytketty testattaviin nostimiin.

Tietojen analysointia varten kehitettiin Microsoft Excel -ohjelmaan perustuva työkalu, jolla voitiin hakea ja esittää tietokantaan tallennettua koeajodataa automaatiojärjestelmän kanssa samaan Ethernet-verkkoon kytketyillä tietokoneilla (Kuva 4. PC).

3.4 Automaatiojärjestelmän jatkokehitys

Seuraava järjestelmäpäivitys tehtiin vuonna 2006, myöskin liittyen insinöörityöhön (Vuoristo 2006). Insinöörityössä rakennettiin erillinen testilaitteisto (arkkitehtuuri on esitetty kuvassa 5), jossa käytettiin osittain samaa ohjaustekniikkaa kuin testiautomaatiojärjestelmässäkin. Uutena lisänä testeriin tuli mahdollisuus selata tietokantaan tallentuneita tietoja www-pohjaisella käyttöliittymällä yrityksen intranetin kautta.

Testerin käyttöliittymänä toimi InTouch-sovellus, joka standardoidun rajapinnan (OPC) kautta ohjasi PLC:tä. Tiedonkeruuta suoritettiin siten, että ajonaikaiset tiedot testipaikan muuttujista tallennettiin InSQL-tietokantaan, josta tietoja analysoitiin organisaation oman CART-ohjelmiston avulla. Testitietokanta piti sisällään tiedot testeistä, testatuista laitteista ja käyttöhistoriasta. Testitietokanta muodostettiin käyttämällä alustana Microsoftin SQL Serveriä, josta luettiin tietoja www-pohjaisella käyttöliittymällä.



Kuva 5. Vuonna 2006 rakennetun testerin arkkitehtuuri (Vuoristo 2006: 32).

Tämän testerin kehityksen jälkeen laboratorion koko testiautomaatiojärjestelmä suunniteltiin uusiksi käyttämään samanlaista arkkitehtuuria. Testiautomaatiojärjestelmän päivityksestä tulikin erittäin onnistunut, mutta siinä tuli ajan myötä vastaan seuraavia ongelmia:

- uuden testerin käyttöönotto oli todella työlästä InTouch:sta johtuen
- ongelmia oli mm. lisenssien, versioiden ja OPC-yhteyksien kanssa
- InTouch-ohjelmasta jouduttiin tekemään epäoptimaalista esimerkiksi taulukoiden puuttuessa. Taulukoita käytettiin PLC-ohjelmassa tehokkaasti hyväksi. (Asiantuntijahaastattelu 2015b.)

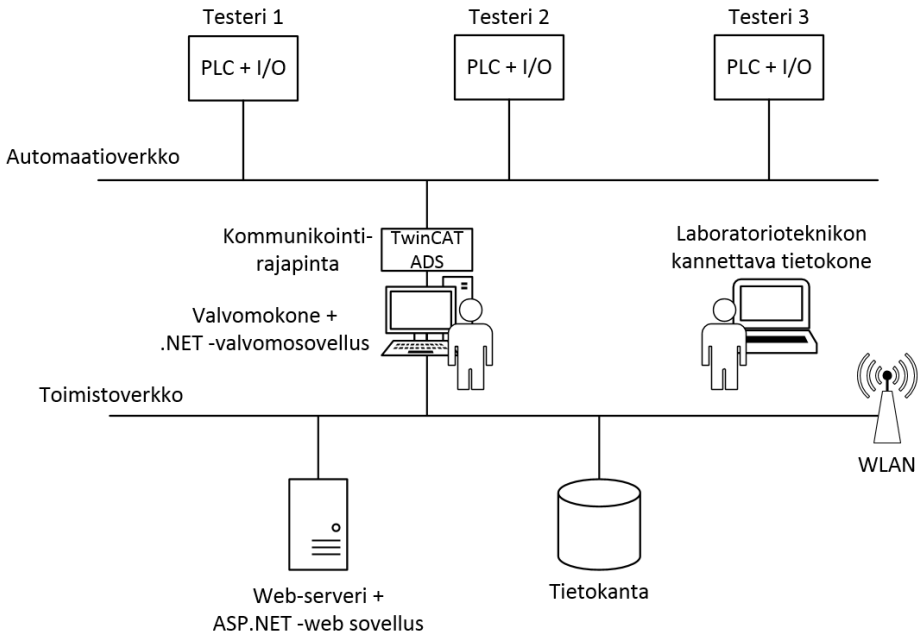
Testaustoiminnan laajentuessa huomattiinkin, että InTouch:in ominaisuudet eivät riittäneet dynaamiseen ja skaalautuvaan järjestelmään, mikä asettikin taas uuden muutostarpeen järjestelmään (Asiantuntijahaastattelu 2015b). Dynaamisella ja skaalautuvalla järjestelmällä tarkoitetaan tässä tapauksessa sitä, että järjestelmässä olisi valmius erilaisille testityypeille ja että se olisi riippumaton testipaikkojen lukumäärästä.

3.5 Testiautomaatiojärjestelmän päivitys nykyiseen muotoon

Viimeisin järjestelmäpäivitys tehtiin vuonna 2011. Oikeastaan voidaan puhua tässä tapauksessa koko järjestelmän uusimisesta, koska vanhasta järjestelmästä ei otettu uusiokäyttöön mitään ohjelmistokomponentteja, PLC-ohjelmaa lukuun ottamatta.

Uusitun järjestelmän arkkitehtuuri on esitetty kuvassa 6. InTouchin tilalle ohjelmoitiin oma Visual Basic .NET -kielinen valvomosovellus, joka Beckhoff:n TwinCAT ADS -kirjastoa käyttäen kommunikoi suoraan ohjelmoitavien logiikoiden kanssa.

Järjestelmän uusimisen jälkeen luotettavuuslaboratorion testaustoimintaa laajennettiin uusiin laboratorioihin. Tämän takia päätettiin myös uusia vanha www-pohjainen tietokannan käyttöliittymä, koska siitä puuttui uutena vaatimuksena tulleet käyttäjänhallinta ominaisuudet. Näin alettiin kehittää uutta web-käyttöliittymää, joka toteutettiin .NET-alustaan perustuvalla ASP.NET-teknologialla.



Kuva 6. Vuonna 2011 uusitun järjestelmän arkkitehtuuri (Asiantuntijahaastattelu 2015c).

Tutkitaan nykyistä järjestelmää hieman arkipäivän esimerkin kautta. Kun luotettavuuslaboratoriossa halutaan suorittaa testausta jollain testerillä, avataan web-käyttöliittymän kautta työ, jossa on suoritettavan testin tarkemmat tiedot. Tämä työ tallentuu tietokantaan, josta valvomokoneen valvomosovelluksella voidaan linkittää kyseinen työ johonkin tiettyyn testeriin ja sen testipaikkaan.

Tämän jälkeen valvomosovelluksesta määritetään työssä testaukseen liittyvät asetukset, kuten ajosykli ja vikaantumisehdot, jotka valvomosovellus lataa testerin PLC:lle. Tämän jälkeen käynnistetään testi valvomosta. Testin etenemistä voi seurata web-sovelluksen kautta, koska valvomosovellus päivittää testin tietoja jatkuvasti tietokantaan. Web-sovelluksen käyttö onnistuu myös langattomasti kannettavalla tietokoneella, koska Ethernet-verkossa on WLAN-lähetin/vastaanotin.

Jos testissä tapahtuu vikaantuminen, joka liipaisee ennalta asetetun vikaantumisehdon, kirjautuu valvomokoneen testipaikan kommenttikenttään aikaleimalla varustettu viesti, joka kertoo tarkemmat tiedot vikaantumisesta. Tämän lisäksi testipaikka lopettaa ajamisen, koska vikaantumista halutaan yleensä tutkia ennen kuin testipaikka laitetaan uudestaan ajoon.

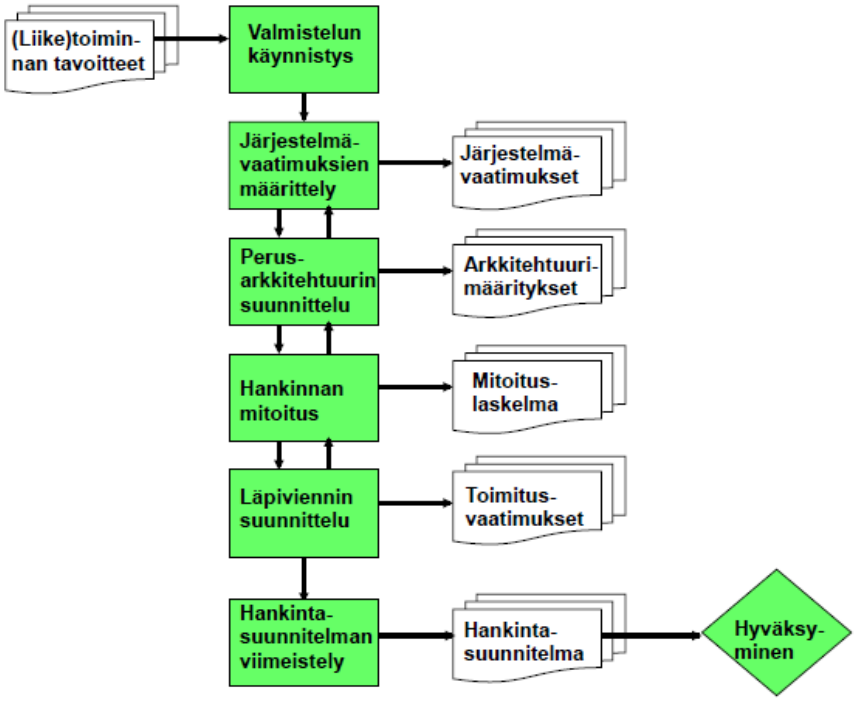
4 Tietojärjestelmähankkeen valmistelu

Mitä isompaa tietojärjestelmää ollaan hankkimassa, sitä tärkeämpää on järjestelmän huolellinen suunnittelu (Forselius 2013: 25).

4.1 Valmistelun yleiskuvaus

Hankkeen valmistelu voi turhauttaa kokematonta suunnittelijaa, koska aikaa kuluu ja mitään näkyvää ei tunnu valmistuvan. Tästä turhautumisesta on kuitenkin hyvä olla tietoisesti välittämättä, koska huolellisen valmistelun vaikutus hankinnan onnistuneeseen läpivientiin on erittäin suuri. (Forselius 2013: 29.)

Valmisteluvaiheen tarkastelun on myös syytä olla oikealla tasolla. Liian ylimalkainen määrittely vain siirtää tarpeellisen selvitystyön tekoa myöhäisempiin työvaiheisiin, joissa keskittyminen pitäisi olla kokonaan ihan muissa asioissa. Toisaalta, liian tarkka määrittely alkuvaiheessa saattaa rajata tarkoituksettomasti pois ne ratkaisut, jotka myöhemmin olisivatkin osoittautuneet oikeiksi. Kuvassa 7 on esitetty valmisteluprosessin tehtävät ja tehtävistä saatavat lopputulokset. (Forselius 2013: 26.)



Kuva 7. Valmisteluprosessin työvaiheet ja niistä saatavat tulokset (Forselius 2013: 25).

4.1.1 Valmistelun käynnistys

Valmisteluprosessin osatehtävässä ***Valmistelun käynnistys*** kuvataan hankkeen motiivi ja lähtökohdat. Tietojärjestelmähankkeen käynnistämisen tarkoituksena tulisi olla (liike)toiminnan kehittämistarve. Kehittämistarve voi olla esimerkiksi olemassa olevan toiminnan laajentamista tai parantamista. Tietojärjestelmää tulisi ajatella (liike)toiminnan yhtenä investoinnin kohteena, joka maksaa itsensä jossain määritetyssä ajassa takaisin. (Forselius 2013: 26.)

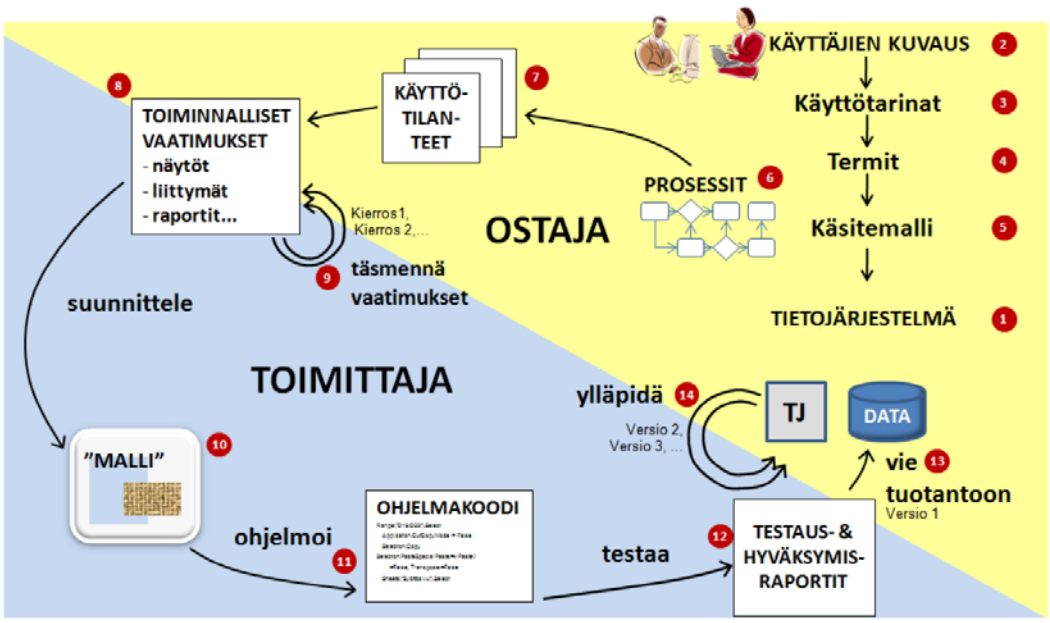
4.1.2 Järjestelmävaatimuksien määrittely

Järjestelmävaatimuksien määrittely pitää sisällään hankkeen eri osapuolten yhteisen näkemyksen siitä, mitä tuleva järjestelmä tulee pitämään sisällään. Näihin lukeutuvat järjestelmän toiminnallisuus, tekniset reunaehdot ja laadulliset seikat. Teknisissä reunaehdoissa voidaan esimerkiksi tuoda esille käytettävä ohjelmointikieli, käyttöjärjestelmä tai haluttu tietokantaratkaisu. Laadulliset seikat pitävät sisällään lähinnä järjestelmän luotettavuus-, käytettävyys- ja tehokkuuskriteerit. Tämä vaihe usein toteutetaan kokonaan erillisenä esitutkimusprojektina ja sen käytännön toteutuksessa löytyvät yleensä seuraavat päätehtävät:

- järjestelmän hankkijan tarpeiden keruu
- tarpeiden analysointi
- tarpeiden täsmentäminen vaatimuksiksi
- vaatimuksien priorisointi ja hyväksyminen.

Näillä työvaiheilla pyritään tuomaan esille se, miten hankinnan kohteena oleva tietojärjestelmä tulee asettumaan osaksi käyttäjien työtehtäviä päivittäisessä toiminnassa. (Forselius 2013: 26-46.)

Tietojärjestelmän määrittely- ja hallintaprosessin kuvaus on esitetty kuvassa 8. Pääperiaatteiltaan se vastaa perinteistä määrittely-suunnittelu-toteutus-testaus -työnkulkua, jota käytetään monessa muussakin ohjelmistoon liittyvässä projektiluontoisessa suunnittelutyössä.



Kuva 8. Tietojärjestelmän määrittely- ja hallintaprosessi (Forselius 2013: 31).

4.1.3 Perusarkkitehtuurin suunnittelu

Perusarkkitehtuurin suunnittelu siirtää suunnitteluprosessin jo hieman konkreettisemmalle tasolle. Tässä työvaiheessa on tarkoitus määrittää järjestelmän teknisten ratkaisujen päälinjat. Näihin kuuluvat yleensä seuraavat osa-alueet:

- järjestelmän keskitys- ja hajautusratkaisut
- työasema- ja palvelinalustojen valinta
- käytettävien kehitystyövälineiden valinta
- avoimen lähdekoodin käyttömahdollisuudet
- käyttöliittymävaihtoehdot
- tietoliikenneratkaisut. (Forselius 2013: 26.)

Jos tietojärjestelmän hankkivaan organisaatioon kuuluu ennestään IT-osasto, joka vastaa tietoteknisten verkkojen ja laitteiden toiminnasta, on sillä todennäköisesti olemassa

valmiita toimintamalleja, jotka liittyvät perusarkkitehtuurin määrittelyyn. On järkevää myös tiedustella organisaation nykyisten tietoteknisten ratkaisujen liittämistä hankittavaan tietojärjestelmään, koska näin voidaan vähentää ylläpidettäviä kohteita. Parhaassa tapauksessa voidaan koko ylläpito sopia IT:n tai jonkun muun tahon hoidettavaksi. Esimerkki tästä voisi olla se, että jos yrityksellä on olemassa jo jonkun valmistajan tietokantaratkaisu käytössä, kannattaa yrittää liittää myös hankittava tietojärjestelmä käyttämään sitä.

4.1.4 Hankinnan mitoitus

Hankinnan mitoituksen tavoitteena on saada realistinen kuva hankkeen laajuudesta, jota voidaan käyttää alustavassa työmäärä-, kustannus- ja aikataulusuunnittelussa. Kustannusarviota tarvitaan investoinnin kannattavuuden arviointiin sekä mahdollisen investointiehdotuksen luontiin. Jos tietojärjestelmä rakentuu useista eri teknologiaa käyttävistä osista ja pitää sisällään erilaisia ohjelmisto- ja laitteistohankintoja, monimutkaistuu hankinnan mitoitus ja on vaikeaa jopa mitoituksen ammattilaiselle. Mitoitus perustuu edellisissä työvaiheissa tehtyihin järjestelmävaatimuksiin sekä tehtyihin arkkitehtuurivalintoihin. Siksi, onkin tärkeää saattaa nämä edellisten työvaiheiden lopputulokset oikealle tarkkuustasolle, jotta realistinen arvio saa riittävän varman pohjan. (Forselius 2013: 26.)

4.1.5 Läpiviennin suunnittelu

Läpiviennin suunnittelun tarkoitus on lisätä hankesuunnitelmaan toimitukselle asetettavat vaatimukset. Näissä vaatimuksissa on hyvä huomioida järjestelmää hankkivan organisaation olemassa olevat hankinta- ja kilpailutuskäytännöt. Hankinnan aikataulu laaditaan sellaiseksi, että kaikilla hankkeeseen liittyvillä osapuolilla on realistinen mahdollisuus noudattaa sitä. Jos järjestelmää hankkiva organisaatio on osaamattomuuttaan laatinut liian tiukan aikataulun, se voi pudottaa kilpailutettavista toimittajista osaavimmat ja rehellisimmät toimittajaehdokkaat. Läpiviennin suunnittelu ottaa myös kantaa toteutettavan projektin valvontakäytäntöihin. (Forselius 2013: 27.)

4.1.6 Hankintasuunnitelman viimeistely

Hankintasuunnitelman viimeistelyssä kootaan yhteen edellisissä työvaiheissa tehdyt määritykset sekä täydennetään hankintasuunnitelma hankkeen perusteluiden ja investointilaskelmien osalta. Valmiin hankintasuunnitelman on tarkoitus tukea ostajaorganisaation tekemää päätöstä. Hankintasuunnitelma voidaan katselmoinnin lopputuloksena hyväksyä, hylätä tai palauttaa täydennettäväksi. Hankintasuunnitelman viimeistelyä edeltävissä työvaiheissa on vaihekohtaisesti hyvä miettiä välikatselmuksien tarvetta, jotta voidaan korjata havaittuja puutteita jo kyseisen vaiheen aikana. (Forselius 2013: 27.)

4.2 Tietojärjestelmän vaatimuksien määrittely

Koska tämä työ on rajattu ainoastaan luotettavuuslaboratorion tietojärjestelmän vaatimusmäärittelyn tekemiseen, käsitellään kuvan 7 hankintakaaviosta ainoastaan kohta *Järjestelmävaatimuksien määrittely* lähemmin tarkasteltuna.

Yksi onnistuneiden tietojärjestelmähankkeiden yhteinen tekijä on se, että järjestelmävaatimusten tuottamiseen on käytetty suuri työmäärä. Huolellisesti tehtynä se on hankinnan valmisteluvaiheen suurin tehtävä. Fred Brooks on yksi ohjelmistoalan tunnetuimpia asiantuntijoita. Hän julkaisi vuonna 1987 paljon huomiota saaneen artikkelin *No Silver Bullet: Essence and Accidents of Software Engineering*. Tässä artikkelissa hän toteaa seuraavaa:

> The hardest single part of building a software system is deciding what to build. No other part of the conceptual work is as difficult in establishing the detailed technical requirements, including the interfaces to people, to machines, and to other software systems. No other part of the work so cripples the results if done wrong. No other part is more difficult to rectify later. (Brooks 1987.)

Moni asiantuntija ja tutkimuskin osoittavat, että ohjelmistoprojektien epäonnistumiset johtuvat pääosin huonoista vaatimuksenkäsittelyistä (Haikala & Mikkonen 2011: 61). Tämä tuntuu maalaisjärjelläkin ajateltuna ihan pätevältä syyltä. Kukaan meistä tuskin osaa sanoa minkälainen näkemys muilla hankkeen jäsenillä on tietojärjestelmästä, jos emme muotoile ja dokumentoi näkemystämme helposti ymmärrettävään muotoon.

Pelkkään suulliseen kommunikaatioon ei kannata luottaa, koska siinä sattuu paljon väärinymmärryksiä. Puheella viestityt asiat myös tahtovat vääristyä ja unohtua ajan myötä,

eikä kukaan enää muista tarkasti mistä viikkoja sitten on sovittu. Tästä johtuen, kannattaakin kuvata ja dokumentoida tietojärjestelmä ja sen käyttötilanteet kattavasti yhteisen ymmärryksen saavuttamiseksi. (Forselius 2013: 30.)

Järjestelmän vaatimusten muuttumista kesken määrittelyn ei ole syytä pelästyä. Määrittelyprojektin edetessä, myös osallisten tietämys ja näkemys tietojärjestelmästä kehittyy, ja järjestelmää tarkastellaan viisaampana uusin silmin. Vaatimusten dokumentteihin tulisikin tehdä jatkuvasti päivityksiä jos asiat niissä muuttuvat. Hyvin ylläpidettyä vaatimuskirjoa voidaan myöhemmin hyödyntää järjestelmän lopputestauksessa, ylläpidossa ja käyttäjien koulutuksessa. (Forselius 2013: 30.)

4.2.1 Käyttäjäryhmien kuvaus

Kun tiedetään mitä tietojärjestelmällä on tarkoitus tehdä, on hyvä pohtia ketkä järjestelmää tulevat käyttämään. Käyttäjiksi on nimetty ne henkilöt tai muut toimijat (esimerkiksi toinen tietojärjestelmä) jotka viestivät järjestelmän kanssa. Näistä käyttäjistä voidaan muodostaa käyttäjäryhmiä. Saman käyttäjäryhmän jäsenille on yhteistä se, että he tarvitsevat järjestelmää samanlaisten toimintojen suorittamiseen. Käyttäjäryhmän liian tarkkaa vuorovaikutusta järjestelmän kanssa tulisi vielä tässä vaiheessa välttää. Ryhmän kuvaukseksi riittää sellainen tarkkuustaso, josta käy ilmi ryhmän luonne ja erottuminen muista ryhmistä. (Forselius 2013: 31-32.)

4.2.2 Käyttötarinat

Kun järjestelmän käyttäjäryhmät on saatu tunnistettua, on seuraavaksi aika kuvata kunkin ryhmän tarpeita käyttötarinoiden muodossa. Käyttötarinoilla saadaan järjestelmän käyttäjät luonnollisesti kommunikoimaan järjestelmälle asetettavista tarpeista. Samalla kun käyttötilanteita kuvataan, tulee myös esille usein erilaisia työtapoja, joiden yhtenäistämiselle tarjoutuu hyvä mahdollisuus. (Forselius 2013: 32-33.)

4.2.3 Kohdesanaston laatiminen

Kun käyttötarinoita luodaan eri käyttäjäryhmille, saattaa esiin tulla ongelmia terminologian kanssa. Kaksi eri käyttäjää voi käyttää samasta asiasta erilaisia nimityksiä tai sama termi tarkoittaa eri käyttäjille eri asioita. Tätä varten on hyvä luoda sanasto kohteesta,

jossa on määritelty termit, niiden kuvaukset ja mahdolliset synonyymit. (Forselius 2013: 33.)

4.2.4 Käsitemalli

Jos hankittavan järjestelmän avulla tallennetaan tai hallinnoidaan tietoa, pitää tallennettavista tietoryhmistä selvittää ja kuvata niiden väliset keskinäiset suhteet. Tätä kuvausta tarvitaan tietojärjestelmän taustalla olevien tietokantojen suunnittelussa. Yksi suosittu työkalu tähän työhön on käsiteanalyysi. Käsiteanalyysissä luodaan järjestelmän pysyvien tietorakenteiden välisiä suhteita kuvaava käsitemalli. Käsitemallin laatiminen voi olla vaativaa kun tietojärjestelmää ollaan hankkimassa automatisoimattomalle kohdealueelle, eikä organisaatiolla ole ennestään kuvattua tietoarkkitehtuuria. Käsiteanalyysi kannattaa aloittaa kun järjestelmän ensimmäiset määritykset on tehty. Käytännössä tämä tarkoittaa riittävää määrää käyttötarinoita sekä terminologian selvitystä. (Forselius 2013: 33-34.)

4.2.5 Toiminnallisten vaatimusten kuvaaminen

Tässä kohtaa määrittelyä keskitytään niihin (liike)toimintaprosesseihin, joissa tuleva tietojärjestelmä on osallisena. Hankkivalla organisaatiolla voi olla myös muitakin prosesseja toiminnassaan, mutta niitä ei ole syytä kuvata tietojärjestelmän kuvausdokumenteissa. Tämä siksi, että keskityttäisiin kuvaamaan olennaisia asioita tietojärjestelmän kannalta. (Forselius 2013: 35-36.)

Toimintojen kuvaaminen aloitetaan karkean tason prosessikaavioita käyttäen. Tämän kaavion päätarkoitus on selkeästi näyttää, miten joku tehtävä, alkuherätteen saatuaan, saadaan onnistuneesti toteutettua. Suosittu tapa kuvata prosesseja on ns. uimaratakaavio, jossa jokainen toimija sijaitsee omalla uimaradallaan. Yhdellä radalla on siis myös hankittava tietojärjestelmä. Muita toimijoita voidaan kuvata kokonaisina käyttäjäryhminä, mikäli se sopii kuvaustilanteeseen. (Forselius 2013: 35-36.)

Tässä kohdassa tietojärjestelmää hankkivalla organisaatiolla on erinomainen mahdollisuus jalostaa omia toimintaprosessejaan. Mikäli vanhat toimintatavat vain muunnetaan suoraan prosessikuvauksiksi, on vaarana se, että kangistutaan vanhoihin kaavoihin.

5 Luotettavuuslaboratorion tietojärjestelmän vaatimusmäärittely

Tässä kappaleessa sovelletaan kappaleen 4 teoriaosuutta luotettavuuslaboratorion vaatimusmäärittelyyn soveltuvin osin. Koska täydellinen järjestelmän määrittely ei mahdu tämän insinöörityön puitteisiin, tehdään kevyempi määrittely järjestelmän pääkohdista. Näihin lukeutuvat käyttäjäryhmien, käyttötarinoiden ja järjestelmän toimintojen määrittäminen.

5.1 Käyttäjäryhmät

Luotettavuuslaboratorion tietojärjestelmän vaatimusmäärittely aloitettiin käyttäjäryhmien tunnistamisella. Seuraavaksi on esitelty nämä käyttäjäryhmät, sekä lyhyesti selostettu miten he ovat järjestelmän kanssa tekemisissä.

Luotettavuuspäällikkö

Luotettavuuspäällikkö käyttää järjestelmää nähdäkseen kokonaiskuvan luotettavuuslaboratorioiden työkuormasta. Tarvittaessa hän voi tarkentaa näkymää myös tehtävätasolle, josta on nähtävissä yksittäiseen projektiin liittyvät asiat. Luotettavuuspäällikkö käyttää järjestelmää edellisten lisäksi seuraaviin tehtäviin:

- Hyväksyy palvelupyynnöt asiakkailta.
- Määrittelee työtehtäviin resurssit (testipaikan ja projektipäällikön).
- Hyväksyy testiin liittyvät testisuunnitelmat ja loppuraportit.
- Seuraa kapasiteetin ja palvelun laadun tunnuslukuja (ks. laatupäällikön kuvaus).

Laatupäällikkö

Laatupäällikkö seuraa järjestelmästä avoimia ja myöhässä olevia työtehtäviä. Lisäksi häntä kiinnostaa testauspaikkojen käyttöaste ja testien sujuvuus. Laatupäällikkö hankkii myös palautetta testaustoiminnan ja luotettavuuslaboratorion palvelun laadusta, järjestämällä asiakkaille asiakastyytyväisyyshaastatteluita.

Testauspäällikkö

Testauspäällikkö vastaa yksittäisen luotettavuuslaboratorion toiminnasta. Hän näkee järjestelmästä luotettavuuslaboratorioiden, projektipäälliköiden ja laboratorioteknikoiden työkuorman. Hän voi määrittää järjestelmästä alaistensa työtehtävät ja niiden prioriteetin.

Tiiminvetäjä

Tiiminvetäjän vastuut ovat samankaltaiset testauspäällikön kanssa sillä erolla, että hän vastaa luotettavuuslaboratorion yksittäisen tiimin työtehtävistä, kun taas testauspäällikkö vastaa kokonaisen testilaboratorion työtehtävistä.

Projektipäällikkö

Projektipäällikkö on yksittäisen projektin vastuuhenkilö ja on täten vastuussa testisuunnitelmasta, toteutuksesta ja loppuraportista. Hän käyttää järjestelmää avatakseen sinne uusia projekteja, seuratakseen käynnissä olevia ja sulkeakseen valmiita. Lisäksi hän hyväksyttää testisuunnitelmat ja loppuraportit järjestelmän välityksellä luotettavuuspäälliköllä. Hän määrittelee myös, mitä tietoja asiakas voi testeistään seurata järjestelmän kautta.

Laboratorioteknikko

Laboratorioteknikko näkee järjestelmästä omat työtehtävänsä ja niiden prioriteetin. Tehdyt työtehtävät merkitään järjestelmässä valmiiksi. Hän asettaa testit päälle ja valvoo niitä järjestelmän kautta. Hän myös kirjaa testeihin liittyvät mielenkiintoiset tapahtumat järjestelmään.

Asiakas

Asiakas näkee järjestelmästä omiin testeihin liittyvät tiedot ja voi avata uusia palvelupyyntöjä järjestelmän kautta.

Järjestelmän ylläpitäjä

Järjestelmän ylläpitäjällä on mahdollisuus käyttää järjestelmän kaikkia toimintoja. Hänen tehtävänä on ylläpitää ja päivittää järjestelmää, sekä ratkaista järjestelmän käyttöön liittyvät ongelmatilanteet.

5.2 Järjestelmän toiminnot

Edellisessä kappaleessa tunnistettiin käyttäjäryhmät ja kerrottiin sanallisesti miten he suoriutuvat päivittäisistä työtehtävistään järjestelmän avulla. Tässä kappaleessa tunnistetaan kertomuksista ne toiminnot, joita järjestelmältä vaaditaan työtehtävistä suoriutumiseksi. Toiminnot käydään läpi loogisessa järjestyksessä asiakkaan palvelupyynnöstä aina testiraportin lähettämiseen asiakkaalle.

Palvelupyynnön avaus

Asiakas kirjautuu tietojärjestelmään käyttäen omaa henkilökohtaista tunnustaan. Tämän jälkeen hän kirjaa palvelupyynnön luotettavuuslaboratoriolle. Palvelupyyntöön täytyy kirjoittaa työtehtävän yleisluontoinen kuvaus. Halutessaan asiakas voi lisätä palvelupyyntöön liitetiedostoja. Asiakas voi myös seurata järjestelmästä aiemmin avattujen palvelupyyntöjen tilaa nähdäkseen onko niihin reagoitu. Asiakkaan puolesta palvelupyynnön voi myös avata kuka tahansa luotettavuuslaboratorion henkilöstöstä.

Palvelupyynnön hyväksyminen

Luotettavuuspäällikkö, testauspäällikkö tai tiiminvetäjä kirjautuu järjestelmään ja näkee, että hänelle on tullut uusi palvelupyyntö. Hän avaa palvelupyynnön ja hyväksyy tai hylkää sen. Hän voi myös lähettää sen asiakkaalle takaisin täydennystä varten jos tietoja on liian niukasti. Hyväksymisen yhteydessä luotettavuuspäällikkö, testauspäällikkö tai tiiminvetäjä määrittää palvelupyynnön toteutuksesta vastaavan projektipäällikön.

Testisuunnitelman hyväksyminen

Palvelupyyntöön liitetty projektipäällikkö lähettää työstä testisuunnitelman järjestelmän kautta luotettavuuspäällikölle tai hänen valtuuttamalleen henkilölle. Testisuunnitelma hyväksytään tai palautetaan projektipäällikölle korjattavaksi, mikäli siihen on tarvetta. Projektipäällikkö näkee järjestelmästä hyväksymispyynnön tilan.

Työtehtävän avaus

Hyväksytylle palvelupyynnölle voidaan avata yksi tai useampia työtehtäviä. Jokainen testattava tuote kirjataan omaksi työtehtäväkseen. Työtehtävien avaus tapahtuu yleensä projektipäällikön toimesta. Työtehtävä voidaan jakaa tarvittaessa useampaan osatehtävään. Työtehtäville määritellään järjestelmän kautta tavoitepäivämäärä ja vastuullinen tekijä.

Resurssien hallinta

Resurssien hallinta antaa mahdollisuuden nähdä ja hallinnoida työtehtäviä luotettavuuslaboratorioiden henkilöille ja testipaikoille. Testipaikkoja hallinnoidessa, järjestelmä antaa yleisluontoisen näkymän vapaista ja varatuista testipaikoista. Varatun testipaikan kohdalla näkyy nykyisen testin ennustettu lopetuspäivämäärä.

Testipaikan ja mittausten ohjaaminen

Järjestelmän kautta voidaan luotettavuuslaboratorioiden testereiden yksittäisille testipaikoille määrittää ajosyklit, vikaantumisehdot ja suoritettavat mittaukset. Testien edetessä järjestelmään voi testipaikkakohtaisesti kirjoittaa mielenkiintoisista tapahtumista kommentteja.

Testiraportin tulostus

Yksittäiseltä testipaikalta voi järjestelmän kautta tulostaa raportin, jossa näkyy testiin liittyvät laskuritiedot (esimerkiksi ajosykli, ajoaika, startit), testipaikan rekisteröimät vikailmoitukset sekä laboratoriohenkilökunnan kirjoittamat kommentit. Tämä raportti yleensä liitetään varsinaiseen asiakkaalle lähetettävään raporttiin.

Testiraportin hyväksyntä

Projektipäällikkö hyväksyttää järjestelmän välityksellä luotettavuuspäälliköllä tai hänen valtuuttamallaan henkilöllä testin tulokset kokoavan raportin. Hyväksytty raportti lähetetään tämän jälkeen asiakkaalle. Asiakas validoi ja hyväksyy raportin. Asiakas voi myös lähettää raportin takaisin korjauspyynnön kera, mikäli hän kokee, että raportti on puutteellinen.

5.3 Vaatimusmäärittelyn vertaus nykyisen järjestelmän toimintoihin

Tällä hetkellä luotettavuuslaboratorion tietojärjestelmä voidaan jakaa kolmeen osa-alueeseen, jotka ovat:

1. web-pohjainen testien ja resurssien hallintatyökalu
2. testiautomaation ohjaus
3. mittaustenhallintasovellus.

Kappaleen 5.2 toiminto *testipaikan ja mittausten ohjaaminen* käsitellään tarkemmin luvussa 6, joten tässä kappaleessa tarkastellaan miten käytössä olevan testien ja resurssien hallintatyökalun toiminnot vastaavat tehtyä vaatimusmäärittelyä.

Asiakas pääsee tällä hetkellä kirjautumaan järjestelmään ja näkee ne testit, joihin hänelle on annettu lukuoikeudet. Tämä ei tapahdu automaattisesti, joten yleensä projektipäällikkö antaa asiakkaalle tarvittavat oikeudet. Asiakas ei voi avata tällä hetkellä järjestelmään uusia palvelupyyntöjä, vaan joutuu lähettämään sen esimerkiksi sähköpostin muodossa jollekin luotettavuuslaboratorion henkilölle. Kaikki luotettavuuslaboratorion henkilöt ovat velvoitettuja vastaanottamaan asiakkaan palvelupyynnön ja ohjaamaan sen eteenpäin luotettavuuslaboratorion projekteja hallinnoivalle henkilökunnalle.

Testisuunnitelman hyväksyminen onnistuu nykyisen järjestelmän kautta. Käytännössä tämä tarkoittaa sitä, että projektipäällikkö avaa järjestelmään työn ja liittää testisuunnitelman sille tarkoitettuun paikkaan. Kun työ on avattu ja testisuunnitelma lisätty, lähettää

järjestelmä tiedon sähköpostilla luotettavuuspäällikölle. Luotettavuuspäällikkö voi hyväksyä testisuunnitelman tai jättää sen hyväksymättä. Testisuunnitelmaa ei voi hylätä ja palauttaa korjattavaksi käyttäen pelkästään järjestelmän toiminnallisuutta. Hyväksymisen yhteydessä järjestelmä tallentaa aikaleiman, josta myöhemmin on nähtävissä, koska suunnitelma on hyväksytty.

Työtehtävän avaus tehdään testisuunnitelman hyväksyttämisen yhteydessä. Työtehtävälle annetaan yksilöllinen tunnusnumero ja tavoitepäivämäärä. Työtehtävä voidaan purkaa useammaksi osatehtäväksi. Yksittäisille osatehtäville voidaan määrätä vastuullinen teknikko, jonka jälkeen tämä osatehtävä näkyy myös teknikon henkilökohtaisessa tehtävälistassa.

Resursseja hallinnoivat henkilöt näkevät teknikoiden työtehtävät sekä testipaikkojen varauksen voi tarkistaa järjestelmän kautta. Yksittäisen testipaikan näkymässä on mahdollista tarkistaa monta tuntia testi vielä ajaa, ennen kuin se pysähtyy. Tämä edellyttää, että ajosyklien määrälle on annettu raja-arvo, jonka jälkeen testi pysähtyy.

Testiraportin tulostus onnistuu vaatimusmäärittelyn mukaisesti, mutta testiraportin hyväksymiseen liittyviä toimintoja ei nykyisestä järjestelmästä löydy. Asiakkaalle toimitettavaa loppuraporttia ei myöskään voi järjestelmän kautta lähettää.

6 Testiautomaatiojärjestelmän kehitys

6.1 Testiautomaatiojärjestelmän kehitystarpeiden määrittely

Tässä kappaleessa keskitytään testiautomaatiojärjestelmän kehittämiseen, joka liittyy tietojärjestelmän toimintoon *testipaikan ja mittausten ohjaaminen*. Tämä kohta valittiin sen takia, koska tällä tietojärjestelmän osa-alueella ei ole varsinaista ylläpitäjää, mutta tarve kehittämiselle on. Kappaleessa 5.1.1 on kerrottu lyhyesti, kuinka kehitystarpeet määriteltiin. Kappaleessa 5.1.2 on kerrottu vaatimuksen tuomasta hyödystä sekä samalla pohditaan lyhyesti kuinka kyseisen vaatimuksen voisi käytännössä toteuttaa.

Tässä kappaleessa ei esitellä niitä kehitysehdotuksia, jotka eivät lukijalle avaudu ilman laajempaa ymmärrystä testiautomaatiojärjestelmästä ja sen käytöstä. Nämä pois jätettävät ehdotukset liittyvät lähinnä järjestelmän yksityiskohtaiseen toimintaan, eivätkä täten jätä oleellista tietoa pois järjestelmän päätoiminnoista.

6.1.1 Käyttäjien haastattelut ja vaatimusten priorisointi

Järjestelmän käyttäjien omakohtaiset kokemukset ovat ensisijaisen tärkeitä järjestelmää kehitettäessä, koska he tulevat järjestelmää jatkossakin käyttämään. Luotettavuuslaboratorion käyttäjistä muodostettiin pienet ryhmät sen mukaan miten he järjestelmää käyttivät. Ajatuksena tässä oli saada keskustelua ryhmän sisällä siitä, minkälaisia toimintoja he järjestelmältä toivoivat.

Kerätyt vaatimukset koottiin yhteen ja priorisoitiin sen mukaan, miten hyödyllinen kukin vaatimuksen toteuttava ominaisuus olisi. Luotettavuuslaboratorion henkilökunnasta valittiin avainhenkilöt suorittamaan priorisointi. Heille annettiin lista näistä vaatimuksista ja jokainen priorisoi ne ensin itse. Tämän jälkeen tulokset kerättiin yhteen ja niitä katselmoitiin ryhmässä. Katselmoinnin yhteydessä kävi ilmi, että osa henkilöistä oli ymmärtänyt jotkut vaatimukset hieman eri tavalla kuin toiset, ja tästä johtui osittaiset erimielisyydet. Nämä virheelliset käsitykset oikaistiin katselmoinnin yhteydessä, eikä niiden vaikutusta ole nähtävissä puhtaaksi kirjoitetussa vaatimuslistassa.

Priorisoinnissa käytettiin arvosteluasteikkoa 1 (ei ollenkaan tärkeä) - 5 (erittäin tärkeä). Priorisoinnin tarkoituksena oli tunnistaa sellaiset ehdotukset, joista ei välttämättä olisi

suurta hyötyä käytäntöä ajatellen, tai joiden toteuttamiseen tarvittava työmäärä olisi liian suuri suhteessa saatavaan hyötyyn. Priorisoitu lista on esitetty liitteessä 1, jossa selkeyttämisen vuoksi on rivit järjestetty arvosteluista johdetun keskiarvon mukaan laskevaan järjestykseen. Vaatimusten arvostelijat on merkattu liitteeseen kirjaimin A, B, C, D, E, F ja G.

Vaatimusten taustavärillä on osoitettu mahdollisesti eriäviä mielipiteitä vaatimuksen tarpeellisuudesta. Vihreällä taustavärillä merkityt vaatimukset herättivät arvostelijoissa suunnilleen samanlaisen mielipiteen tärkeyden suhteen. Keltaisella merkityissä vaatimuksissa oli pientä erimielisyyttä, kun taas punaisissa kohdissa saattoi olla täysin vastakkaisia mielipiteitä.

Suurin käyttäjille näkyvä kehitysehdotus liittyi testiautomaatiojärjestelmän arkkitehtuurin muuttamiseen. Tämän seurauksena testiautomaation käyttö helpottuisi huomattavasti. Nykyisessä järjestelmässä testin käynnistys, syklin asetukset ja vikakuittaukset täytyy tehdä valvomo PC:ltä, ja tämän lisäksi itse testerin luona täytyy käydä laittamassa testerin ajokäsky päälle (turvallisuussyistä). Pisimmillään valvomokoneen ja testerin välinen kävelymatka saattaa olla satoja metrejä, joten on järkevää välttää turhaa edestakaista kulkemista. Parannusehdotuksen toteutuksen jälkeen, testin käynnistys ja muut toiminnot voitaisiin suorittaa joko valvomokoneen, henkilökohtaisen tietokoneen tai testerin käyttöliittymän kautta.

Kehitysehdotuksia tuli myös mittaus- ja ohjausjärjestelmän yhdistämiseen, jotta voitaisiin esimerkiksi lämpötilamittauksen perusteella pysäyttää testi. Toisaalta, ohjauspuoli voisi myös käynnistää mittauksen kun tieto jostain mielenkiintoisesta tapahtumasta tulisi järjestelmään. Lisäksi, mittausdatan synkronointi suoritettavan testin aikajanaan olisi tärkeää testitulosten analysoinnin kannalta.

Paljon tuli myös sellaisia ehdotuksia, joita ei voi oikeastaan lukea uusiksi ominaisuuksiksi vaan paremminkin vanhan toiminnallisuuden ja käytettävyyden hiomiseksi. Osa ehdotuksista käsitteli osittain päällekkäisiä asioita. Tämän takia olisi hyvä miettiä, kannattaako kaikkia ehdotuksia toteuttaa. Esimerkiksi mahdollisuus testipaikan hallintaan testipaikalta saattaisi poistaa tarpeen, jossa valvomosovellusta käytettäisiin omalta kannettavalta tietokoneelta.

6.1.2 Vaatimusten kuvaus

Testipaikan ajopyynti, käynnistäminen ja vikojen kuittaaminen voidaan hoitaa testipaikan luona

Tämän toiminnon tarkoituksena on helpottaa testin hallintaa siten, että testiä voidaan ohjata suoraan itse testipaikalta. Tämän toiminnon toteuttaminen vaatisi vikojen, kuittauksen ja testipyynnin lisäämistä testipaikan käyttöliittymäpaneeliin. Mikäli paneelia ei testipaikalla ole, voitaisiin testi käynnistää henkilökohtaisen kannettavan tietokoneen välityksellä, käyttämällä selaimen kautta käytettävää valvomosovellusta.

Mittausdata on oltava yhdistettävissä tiettyyn testattavan laitteen käyttöhetkeen

Tällä toiminnolla on tarkoitus synkronoida testattavan laitteen mittaus testin aikajanaan. Aikajanaan sidotulla mittauksella voidaan testin aikana seurata esimerkiksi kuinka laitteen lämpötila muuttuu testaustapahtuman aikana. Kun testin jälkeen analysoidaan laitteen käyttäytymistä, voi tästä tiedosta olla hyötyä esimerkiksi sellaisessa tapauksessa, jossa testattava laite on hajonnut testin aikana. Käytännön toteutus voisi olla esimerkiksi tietokantaan kerättävä mittaus. Tietokannassa mittausdatan keräävä taulu voisi esimerkiksi sisältää kentät syklinumerolle, testipaikalle ja aikaleimalle. Tämän perusteella voisi tarkasti määritellä, että missä kohtaa aikajanaa mittaukset on tehty.

Testerissä on napit testien alasajoon ja näiden samojen testien ylösajoon

Testien aikana täytyy välillä tehdä testattaville laitteille erilaisia huoltoja ja/tai mittauksia. Luotettavuuslaboratorion turvallisuusohjeet määräävät, että ympärillä mahdollisesti vaaraa aiheuttavat testit täytyy ajaa alas toimenpiteiden ajaksi. Jos pysäytettäviä testejä on paljon, ei välttämättä ole mielekästä yrittää muistaa, mitkä testit tuli pysäytettyä. Tätä varten olisi hyvä olla toiminto, joka käynnistää ne samat testit uudestaan, jotka tuli alasajo-toiminnolla pysäytettyä.

Testerissä mahdollisuus ladata sama testisykli halutuille testipaikoille. Myös kaikkien testipaikkojen käynnistäminen/sammuttaminen yhdellä napin painalluksella pitää olla mahdollista.

Tässä kohdassa esitetyt toiminnot helpottaisivat erityisesti komponenttitestien hallintaa. Nykyisessä järjestelmässä testisykli on ladattava jokaiselle testipaikalle erikseen ja käynnistys on suoritettava yksi testipaikka kerrallaan. Komponenttitesteissä saattaa olla useita kymmeniä testipaikkoja, joten toiminnallisuuden parantaminen nopeuttaisi testien ajoonlaittoa sekä myös vähentäisi inhimillisiä virheitä testisykliä ladatessa. On huomioitava, että joissain tapauksissa testien yhtäaikainen käynnistäminen kuormittaa esimerkiksi testerin ohjausjännitemuuntajaa tai testeriä syöttävää sähköverkkoa liikaa. Tämän takia on käytettävä porrastavaa toimintoa testien käynnistykseen, joka jakaisi testipaikkojen käynnistyksen tasaisesti halutulle ajanjaksolle. Käytännön toteutus tarkoittaisi sitä, että testerin logiikkaohjelmaan lisätään kyseinen toiminto. Lisäksi, logiikkaohjelman ohjelmistorajapintaan pitäisi luoda muuttuja, jolla kyseistä toimintoa ohjataan valvomosovelluksesta käsin.

Testipaikalla tulee olla mahdollista asettaa useampi ohjauskäsky rinnakkain. Myös useamman ehdon käyttö täytyy olla mahdollista.

Nykyisessä järjestelmässä on mahdollista ajaa vain yhtä käskyä kerrallaan. Tämä on todettu rajoittavaksi tekijäksi testaustoiminnan monipuolistuessa, joten tulevassa järjestelmässä tulisi olla mahdollisuus tehdä useita rinnakkaisia ohjauksia testattavalle laitteelle. Nykyisessä järjestelmässä voi myös asettaa testisyklin askeleille ehtoja, joiden täytyy täyttyä, jotta testi jatkuu. Esimerkiksi nostimen testipaikalla voidaan nostoajon päätteeksi laittaa tauko-askeleen kohdalle ehto ”yläraja laukaistu”. Mikäli ehto ei ole tosi kyseisen askeleen kohdalla (esimerkin tapauksessa yläraja jäänyt laukaisematta), menee testi vikatilaan ja kommentti kirjautuu automaattisesti kommenttikenttään. Näitä kyseisiä ehtoja tulee voida myös asettaa useita rinnakkain uudessa järjestelmässä.

Testattavan laitteen vikaantumishetken ympäriltä tallennetaan mitattavista muuttujista, ohjauksesta ja takaisinkytkennöistä tilatiedot

Tämä ominaisuus helpottaisi laitteen vikaantumisen analysointia. Käytännön testauksessa on ollut monesti hyödyllistä tietää, miten testattavaa laitetta on ohjattu ja millaisia takaisinkytkentätietoja järjestelmään on tullut vikaantumisen aikana. Käytännössä tämä

tarkoittaa jonkinlaisen rengaspuskurityyppisen nauhoituksen lisäämistä testipaikoille. Tämä tarkoittaa sitä, että kun muisti on täyttynyt siihen kirjatuista arvoista, alkaa muistin päällekirjoittaminen vanhimmista arvoista lähtien.

Nostintestissä sillansiirto pitää olla mahdollinen

Joissain testeissä täytyy olla nostimen lisäksi mahdollista ohjata nostinta liikuttavaa vaunua. Tulevassa järjestelmässä on hyvä myös varautua siihen, että myös vaunua liikuttavaa siltaa halutaan ohjata. Koko nosturi tulisi olla siis mahdollista kytkeä automaattiohjauksen piiriin. Nosturin nostoelimessä saattaa myös olla joku ohjausta vaativa toimilaite, joka olisi myös syytä pitää mielessä ohjausta suunniteltaessa.

Testipaikkaa pitää pystyä ohjaamaan mittauksen perusteella (esimerkiksi lämpötilan nouseminen aiheuttaa vian ja pysäyttää testin). Testipaikan pitää myös pystyä käynnistämään mittaus jollain herätteellä

Testattavista laitteista voidaan mitata erilaisia fyysisiä suureita (esimerkiksi lämpötila tai värinä). Järjestelmässä pitäisi olla mahdollisuus ennalta asettaa näille suureille raja-arvoja, joiden ylittyessä, tapahtuu joku käyttäjän määrittämä toimenpide. Tämä voisi olla testin pysäytys tai vaikka kommentin kirjaus testipaikan kommenttikenttään.

Testipaikalla tulee olla mahdollista asettaa n määrä ajosyklejä, joita voi ajaa haluamassaan järjestyksessä

Testauksen monipuolistamiseksi, järjestelmässä tulisi olla mahdollista käyttää testin aikana erilaisia ajosyklejä. Tämä siis voisi olla eräänlainen sykli, joka sisältää testisyklejä.

Testipaikalta tulee saada näkyviin trendi, jossa mitattavat muuttujat esitetty graafisesti

Suoritettavan testauksen aikana voi olla tarve seurata esimerkiksi nostimen lämpötilan kehittymistä. Tämän takia järjestelmän olisi tarjottava mahdollisuus näyttää halutut muuttujat graafisesti ajan suhteen.

Testipainon paikantaminen

Nostintesteissä kuormana käytettävän painon sijainnin mittaaminen antaisi uusia mahdollisuuksia testien ohjaamiseen, kuten esimerkiksi nostonopeuden mittaaminen ja hidastus- ja pysäytysrajojen opettamisen. Nykyisessä järjestelmässä katkaistaan nostimen ajokäsky mekaanisella pyörivällä rajakytkimellä tai vipurajalla. Paikoitusjärjestelmän avulla voitaisiin sitoa pysäytys- ja hidastusrajat testipainon sijaintiin, jotka voitaisiin esimerkiksi ajamalla opettaa kyseiselle testipaikalle. Toteutusta varten pitäisi ottaa selvää erilaisista etäisyydenmittausantureista ja testata, mitkä näistä voisivat soveltua painon paikoitukseen.

Testipaikalla tehdyt toimenpiteet tallentuvat järjestelmän lokikirjaan, josta käy ilmi, että kuka on tehnyt muutoksia ja milloin.

Testeihin tapahtuvat muutokset (esimerkiksi testisyklin muutokset) on tärkeää jäljittää niiden tekijään, jotta pystytään tarvittaessa selvittämään muutosten tarkoitus. Muutosta tehtäessä, järjestelmä voisi tarjota samalla kommentointimahdollisuutta, jotta lokikirjasta nähdään suoraan syy muutokselle.

Valvomoon pitää olla mahdollisuus päästä henkilökohtaisella tietokoneella

Nostintestin aloituksessa on tarve yleensä hienosäätää ajosykliä, jotta testiajo saadaan sujumaan onnistuneesti. Vanhassa järjestelmässä on täytynyt ajosyklin muutokset tehdä valvomokoneen kautta, joten kaukana valvomokoneesta sijaitsevien testipaikkojen hienosäätäminen on vaatinut edestakaista siirtymistä valvomon ja testipaikan välillä. Tämä voitaisiin välttää mahdollistamalla valvomon käyttäminen henkilökohtaiselta tietokoneelta, jonka voisi ottaa mukaan testipaikalle hienosäätöä varten.

Testipaikka muistuttaa lähestyvästä huollosta

Testattaville laitteille tulee usein tarve suorittaa erilaisia määräaikaishuoltoja. Nykyisessä järjestelmässä on mahdollisuus asettaa raja-arvo syklilaskuriin, joka pysäyttää testin tietyn syklimäärän toteutuessa. Joskus testin pysähtyminen tapahtuu viikonlopun aikana, jolloin testi on pysähdyksissä pitkään. Tämä toiminto muistuttaisi testistä vastuussa olevaa teknikkoa tekemään huollon aiemmin, esimerkiksi pysähdystä edeltävänä arkipäivänä.

Testerin testipaikka pitää pystyä lukitsemaan

Testauksessa voi tulla eteen tilanteita, joissa testattavan laitteen ajaminen halutaan lopettaa tutkimuksia varten. Yksi syy tähän voi olla se, että laitteessa on joku alkava vaurio, eikä sitä haluta ajaa hajoamispisteeseen saakka. Tästä syystä olisi hyvä, että yksittäiset testipaikat voisi lukita pysäytystilaan ja näin suojata vahinkokäynnistyksiltä.

Testeri tallentaa ympäristöstään tietoa, kuten esimerkiksi lämpötila ja kosteus

Testauspaikan ympäristöstä voisi olla hyödyllistä tallentaa tietoa, jotta olosuhdevaikutukset voitaisiin tiedostaa samalla kun tutkitaan testattavan laitteen käyttäytymistä mittausdatasta.

6.2 Uuden testiautomaatiojärjestelmän kuvaus

Tässä kappaleessa on käyty läpi nykyisen testiautomaatiojärjestelmän rakennetta. Samalla on myös pohdittu, miten nykyisestä järjestelmän rakenteesta saisi mahdollisimman vähällä työllä uuteen testiautomaatiojärjestelmään soveltuvan rakenteen.

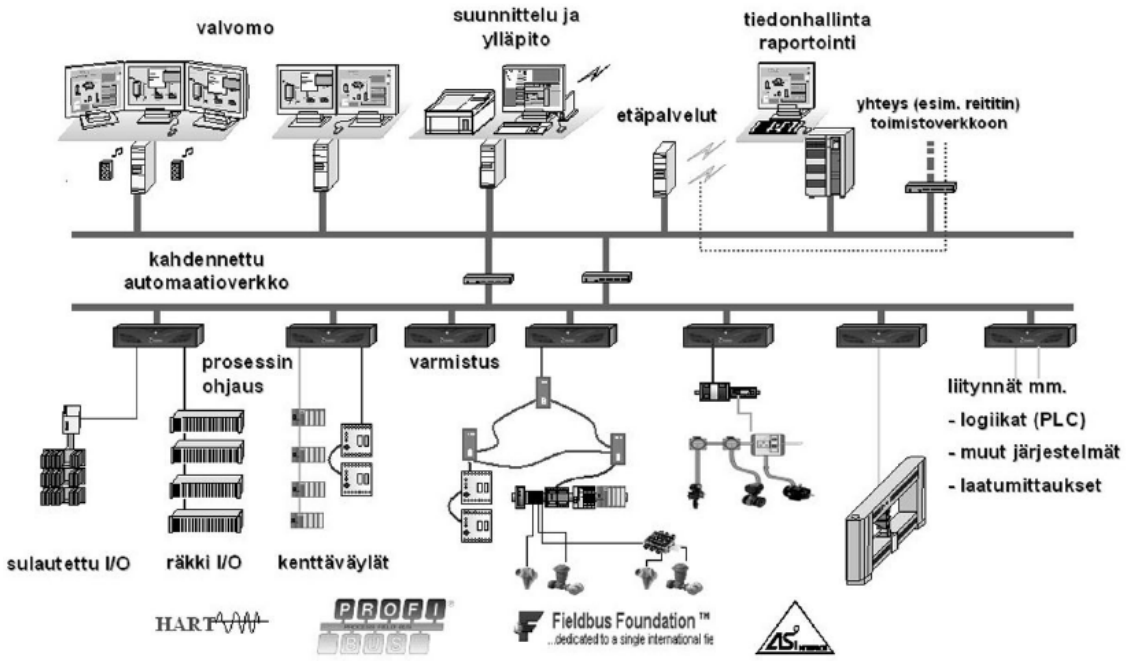
6.2.1 Testiautomaatiojärjestelmän laitteistoarkkitehtuuri

Tietoverkot

Testiautomaatiojärjestelmässä (Kuva 6 s.12) on tällä hetkellä käytössä aika yleinen automaatioon liittyvä verkkoratkaisu, jossa automaatiolaitteet ovat omassa Ethernet-verkossaan erotettuna toimiston ja yrityksen Ethernet-verkosta. Näiden verkkojen solmupisteessä on valvomokone, joka hoitaa tiedonkeruun automaatioverkon laitteista ja kirjoittaa tiedot toimistoverkon kautta tietokantaan.

Kuvassa 9 on esitetty yksi yleinen malli nykyaikaisen automaatiojärjestelmän arkkitehtuurista. Mallissa on ylimmällä tasolla toimistoverkko, josta luodaan yhteys reitittimen kautta valvomon tietoverkkoon. Valvomon tietoverkosta on luotu kahdennettu yhteys kenttätason tietoverkkoon, johon on kytketty automaation ohjaus. Näistä ohjauslaitteista

taas lähtee omat kenttäväylätyyppinsä (esimerkiksi HART, Profibus, Fieldbus, ASi) ohjattaville laitteille. Testiautomaation verkko on hieman yksinkertaisempi kuin kuvan 9 malli, mutta samaa pääperiaatetta käytetään.



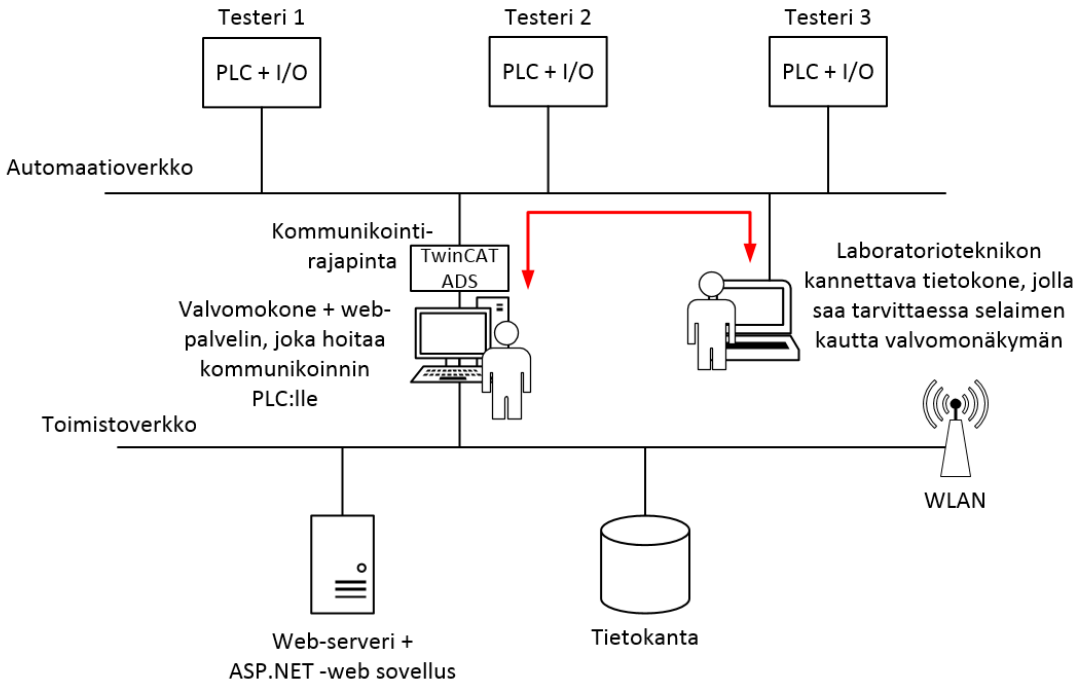
Kuva 9. Nykyaikaisen automaatiojärjestelmän malli (Suomen Automaatioseura 2010: 52).

Nykyisen testiautomaation verkkoratkaisu on käytössä toiminut hyvin, eikä sitä ole syytä ilman hyvää perustelua muuttaa. Nykyajan trendi suosii langattomuutta, mutta esimerkiksi komponenttitestien ympäristössä on usein paljon häiriötekijöitä (esimerkiksi kontaktoreiden päästöstä aiheutuvat jännitepiikit), jotka voivat syödä langattoman verkon luotettavuutta. Jos langattomia verkkoja haluttaisiin käyttää, olisi syytä ensin kokeilla verkon häiriönsietokykyä testiympäristössä jollain pienellä koelaitteistolla.

Valvomon toimintaperiaate

Nykyinen PC-pohjainen valvomosovellus on palvellut hyvin tarkoitustaan, mutta testaamisen helpottamiseksi olisi syytä miettiä sellaista ratkaisua, joka mahdollistaa valvomoon pääsyn myös omalta henkilökohtaiselta kannettavalta tietokoneelta. On olemassa myös erilaisia etäkäyttöohjelmia, joiden avulla valvomokoneeseen voitaisiin ottaa yhteys, mutta näiden ongelma se, että usea käyttäjä ei pääse samaan aikaan käyttämään valvomoa.

Yksi vaihtoehto voisi olla selainpohjainen sovellus, joka sallisi pääsyn testereiden asetuksiin automaatioverkosta käsin. Web-palvelin voisi sijaita tietokoneessa, joka on kytketty automaatioverkon ja toimistoverkon välille, koska näin tämä sama kone voisi myös hoitaa tietojen keräämisen testereiltä ja mittauslaitteilta automaatioverkon kautta, ja tämän jälkeen kirjoittaa ne toimistoverkkoa käyttäen tietokantaan. Yksi malli tämän kaltaisesta arkkitehtuurista on esitetty kuvassa 10. Tässä mallissa ei vanhaan järjestelmään tarvitsisi tehdä muutoksia laitteiston suhteen, vaan tarvittavat parannukset voitaisiin tehdä ohjelmallisesti.



Kuva 10. Yksi mahdollinen arkkitehtuurin toteutusmalli

Testereiden ohjaus

Testien ohjaus on hoidettu pääsääntöisesti Beckhoffin ohjattavien logiikoiden avulla. Ne ovat osoittautuneet toimiviksi ratkaisuiksi pienen koon ja edullisten hankintakustannusten vuoksi. Myös testiautomaatiota ylläpitävät henkilöt ovat saaneet paljon harjoitusta Beckhoffin laitteiden kanssa.

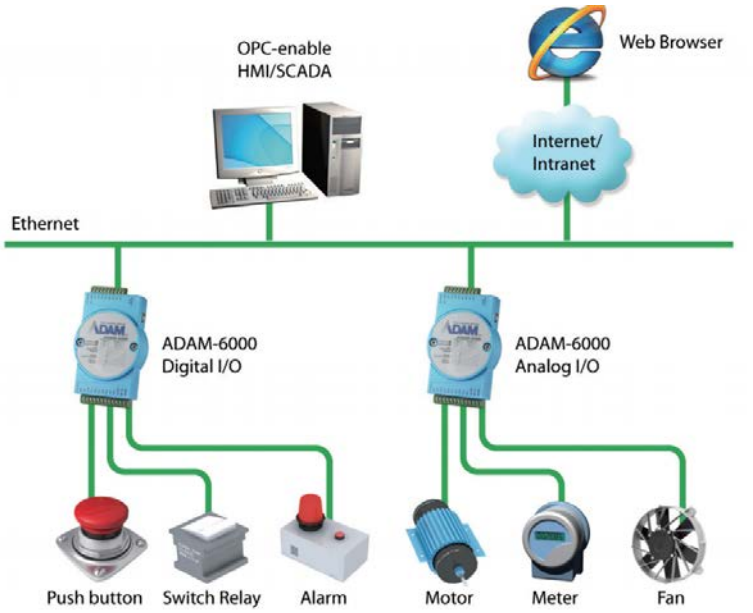
Testereissä käytettävät sovellusohjelmat perustuvat perustoiminnallisuuden sisältävään logiikkaohjelmaan, johon tarvittaessa lisätään testin vaatimaa lisätoiminallisuutta. Perusohjelmistossa on vakioidut ohjelmistorajapinnat, jonka ansiosta minkä tahansa perusohjelman sisältävän testerin voi kytkeä nykyiseen valvomoon. Tällainen ratkaisumalli voisi palvella tulevaisuudessakin testaustarpeita.

Käytössä oleva TwinCAT 2 -ohjelmointiympäristö alkaa olemaan elinkaarensa päässä (erittäin toimiva ratkaisu silti), eikä tue logiikkaohjelmoinnin kielistandardiin (IEC 61131-3) 2013 uutena lisänä tullutta olio-pohjaista ohjelmointityyliä kuten uudempi TwinCAT 3. Taitavissa käsissä oliopohjainen PLC-ohjelma synnyttää parhaimmillaan helpommin ylläpidettävän ja selkeämmän ohjelmasovelluksen.

Ylläpidettävyyden ja selkeyden parantuminen johtuu siitä, että proseduraaliseen ohjelmointityyliin verrattuna, oliotyylinen ohjelmointi sitoo datan ja toiminnot yhdeksi kokonaisuudeksi (olio), kun taas proseduraalisessa ohjelmointityylissä toiminnot käsittelevät globaalisti saatavilla olevaa dataa. Myös monet muut olio-ohjelmoinnin ominaisuudet kuten perintä, rajapintojen käyttö ja monimuotoisuus tekevät sovellusohjelmasta taitavissa käsissä tehokkaan ja selkeän.

Mittausjärjestelmä

Nykyisen mittausjärjestelmän muodostavat Advantechin ADAM Ethernet I/O -moduulit, joista löytyy paljon erilaisia malleja tarvittavien mittausten suorittamiseen. Kuvassa 11 on esitetty malliarkkitehtuuri, jolla kyseisiä mittausmoduuleita voidaan hyödyntää. Mittausmoduulit kytketään Ethernet-verkkoon, jonka välityksellä käyttöliittymästä (HMI/SCADA tai web-selain) luetaan mittauksia ja tehdään konfigurointi. Mittausmoduuleiden lukemiseen käytetään kohdeyrityksessä ohjelmistoalihankinnalla tehtyä sovellusta, joka on vielä kehitysasteella, mutta joiltain osin käytössä.



Kuva 11. Advantechin ADAM Ethernet I/O -moduulit ja niiden toimintaympäristö (Advantech ADAM-6017).

Jotta ohjauksen ja mittauksen yhteensovittaminen onnistuisi vaivattomasti, tulisi niiden toimia saman sovelluksen ohjaamana. Tätä varten web-pohjaiseen valvomosovellukseen voitaisiin integroida myös mittausten hallinta. Laitteisto voisi toimia niin, että halutulle testipaikalle voidaan konfiguroida tietty mittausmoduuli, jonka I/O-kanavien toiminta määriteltäisiin samalla tavalla selaimen kautta kuin esimerkiksi testipaikan ajosyklit.

6.2.2 Valvomosovelluksen toiminnallinen kuvaus

Valvomosovellukseen kirjaudutaan sisälle käyttäen omaa henkilökohtaista tunnusta ja salasanaa. Käyttäjän tekemistä toimenpiteistä jää aikaleimalla varustettu merkintä sovelluksen lokikirjaan. Tämä toiminto parantaa testeihin tehtyjen muutosten jäljitettävyyttä.

Valvomosovelluksen etusivulla avautuu näkymä valvomokoneeseen sidotuista testilaitteistoista. Näistä käyttäjä voi valita testilaitteiston, jolle aikoo tehdä toimenpiteitä. Käyttäjä voi muokata sovelluksen asetuksista, mitkä testerit etusivun näkymässä ovat esillä. Uusien testereiden lisääminen ja vanhojen poistaminen on myös mahdollista sovelluksen kautta.

Testin yhdistäminen halutulle testipaikalle edellyttää, että testi kuuluu johonkin projektipäällikön aikaisemmin avaamaan työhön. Tässä työssä on määritetty automaattisesti ajettavat testit, joista jokaisesta on avattu oma osatehtävä. Mikäli projektipäällikkö on testit määrittäessä laatinut myös käytettävän testisyklin, yhdistetään myös se testipaikalle.

Kun käyttäjä valitsee etusivulta yksittäisen testerin, tarkentuu näkymä testerin sisältämiin yksittäisiin testipaikkoihin. Yksittäisen testipaikan näkymässä on tiedot testistä, laskuriarvoista sekä viimeisimmät kirjatut kommentit. Tästä näkymästä voidaan myös asettaa testipaikan ajosykli sekä mittauksiin liittyvät toimenpiteet.

Testipaikan ajosykli määritetään lisäämällä ajosykliin yksittäisiä ajoaskeleita, joille määritetään kesto, ohjattavat lähdöt sekä takaisinkytkentäehdot. Näiden ajoaskeleiden yhdistelmillä voidaan testipaikalle luoda haluttu ohjaus. Yksittäisiä ajosyklejä voi myös ketjuttaa peräkkäin, jolloin saadaan testipaikalle vaihtelevaa ohjausta.

Mittauksia voi lisätä testipaikalle linkittämällä ja konfiguroimalla siihen haluttu mittausmoduuli. Testipaikan asetuksista määritetään mitattavan suureen tyyppi, mittausintervalli ja skaalausasetukset. Mitattavaan suureeseen voidaan lisätä raja-arvoihin liittyviä toimintoja, kuten testin ajaminen vikatilaan tai automaattisen kommentin kirjaaminen testipaikalle.

6.2.3 Testerin toiminnallinen kuvaus

Testerin käyttöliittymänä toimii graafinen paneeli tai tarvittavat painonapit. Kaikkiin testereihin ei ole mahdollista asentaa järkevällä tavalla graafisia käyttöliittymäpaneeleja, joten tämän takia on käyttöliittymätoiminnallisuus toteutettava myös painonapein. Paneelin kautta voidaan käynnistää ja sammuttaa testipaikka. Myös testisyklin ohjelmointi ja vikojen kuittaus on mahdollista paneelin kautta. Paneelin kautta voi katsoa yksittäisen testipaikan tarkemmat tiedot.

Painonapein toteutettu käyttöliittymä soveltuu yksinkertaisempiin testeihin, missä testisyklin muokkaus ei kuulu testipaikan hienosäätöön käyttöönottoa tehdessä. Esimerkiksi komponenttitesteissä ei yleensä ole tarvetta hienosäätää testisykliä. Jos testisykliä on tarpeen muuttaa, se voidaan tehdä tietokoneella valvomosovelluksen kautta. Painonapeilla voidaan käynnistää ja pysäyttää testit.

Joissain testilaitteistoissa on tarpeen ajaa testipaikkaa myös käsin radio-ohjaimen välityksellä. Tällaisia testilaitteistoja ovat esimerkiksi nostintesterit. Radio-ohjaimen välityksellä on myös mahdollista suorittaa testattavalle nostimelle opetusajo, jossa määritetään sen ylä- ja alarajan paikka. Tämä opetusajomahdollisuus vaatii nostimeen kytkettävän painon paikoituksen laser-etäisyysmittauksella tai jollain muulla tavoin.

6.3 Ajatuksia uuden testiautomaatiojärjestelmän kehittämisestä

Kehitettävää uuteen testienohjausjärjestelmään tulee paljon, eikä kaikkea toiminnallisuutta voida toteuttaa kerralla. Sopiva tapa uuden järjestelmän kehittämiseen, olisi kokeilla protoilemalla mahdollisia toteutustapoja. Hyväksi havaitut tavat voitaisiin lisätä inkrementaalisesti kehitettävään järjestelmään, jota rakennettaisiin vanhan järjestelmän rinnalle. Jossain vaiheessa kun uusi järjestelmä on riittävän vakaa ja sisältää riittävästi toiminnallisuutta, voitaisiin tehdä järjestelmän vaihto vanhasta uuteen. Tästä alkaisi uuden järjestelmän käyttö- ja ylläpitokausi. Uuteenkin järjestelmään voitaisiin kehittää uusia toimintoja protomallissa, mutta ne liitettäisiin suoraan käytössä olevaan järjestelmään, kunhan ne on ensin testattu ja todettu toimiviksi.

Edellisen kappaleen sisältö on melko epätarkka kuvaus konkreettisen kehitystyön aloittamiseksi, joten sitä täytyy tarkentaa lisää projektisuunnitelmassa. Myös kehitystyön purkaminen yksittäisiksi, pienemmiksi askeliksi auttaa hallitsemaan tapahtuvaa kehitystyötä. Nämä askeleet voitaisiin kuvata esimerkiksi seuraavalla tavalla:

- työhön liittyvän projektisuunnitelman teko
- vaatimusten tarkentaminen sille tasolle, että toteuttamiskelpoisuuden voi protoillessa selvittää
- toteutettavien toimintojen liittäminen uuteen järjestelmärakenteeseen
- kokonaisuuden katselmointi, testaus ja dokumentoinnin täydentäminen
- järjestelmän käyttäjien koulutus.

Testiautomaatiojärjestelmän kehitys on suurimmilta osin ohjelmistokehitystyötä, joka jakaantuu kahteen osaan: ohjelmoitavan logiikan ja valvomon sovellusohjelmaan. Hyvä lähestymistapa ohjelmistokehitykseen olisi aloittaa kehitys näiden kahden kokonaisuuden rajapintojen suunnittelusta. Toisin sanoen, mitä viestejä näiden sovellusten tulee vastaanottaa ja lähettää, ja minkälaisten muiden järjestelmien tai käyttäjien kanssa ne kommunikoivat saadakseen aikaan halutun toiminallisuuden. Samalla tulisi selvitettyä se, mitä I/O-liitäntöjä ohjelmoitavan logiikan ja mittausjärjestelmän tulisi pitää sisällään.

Toteutettava järjestelmä tulee olemaan edelleen osa isompaa kokonaisuutta, eli luotettavuuslaboratorion tietojärjestelmää. Tämän takia on pidettävä jatkuvasti mielessä, kuinka uusi testiautomaatiojärjestelmä tulee johdonmukaisesti liittymään tähän isompaan kokonaisuuteen.

7 Yhteenveto

Insinöörityössä selvitettiin luotettavuuslaboratorion tietojärjestelmän käyttäjien tarpeet tietojärjestelmää kohtaan. Näitä tarpeita verrattiin olemassa olevaan tietojärjestelmään, jotta voitiin tunnistaa kehityksen kohteita. Kävi ilmi, että web-pohjaisen hallintatyökalun toiminnallisuus kattoi suurimman osan käyttäjävaatimuksista, mutta myös kehitettävää löytyi, kuten asiakkaan web-näkymän toimintojen kehittäminen ja testisuunnitelmien ja -raporttien hyväksymiseen liittyvä toiminallisuus. Ennen kuin näitä toimintoja ruvetaan kehittämään olisi syytä arvioida luotettavuuslaboratorion omien testaus- ja työprosessien kulku. Tämä sen takia, ettei tyydyttäisi tekemään asioita niin kuin ollaan aina ennenkin tehty, vaan etsittäisiin tämänhetkiset parhaat mahdolliset toimintatavat.

Työssä tehtiin myös testiautomaation kehitykseen liittyvä vaatimusmäärittely, jossa järjestelmän käyttäjiltä kerättiin parannusehdotuksia. Nämä ehdotukset priorisoitiin luotettavuuslaboratorion henkilöistä koostuvan pienen ryhmän kanssa. Näiden ehdotusten pohjalta voidaan testiautomaatiojärjestelmästä kehittää testien ohjaukseen nykyistä monipuolisempi ja joustavampi järjestelmä. Työssä esiteltiin myös yksi mahdollinen malli siitä, minkälainen uusi tuleva testiautomaatiojärjestelmä voisi olla. Valmiita pakettiratkaisuja on vaikea löytää luotettavuuslaboratorion kaltaiselle toimintaympäristölle, etenkin jos järjestelmään täytyy räätälöidä sovellusalakohtaisia toimintoja. Tämän takia esitettyä mallia ei pystytty työhön käytetyn ajan puitteissa tarkentamaan enempää.

Tämä työ erosi aikaisemmista järjestelmäpäivitysten vaatimusmäärittelyistä siten, että huomioon otettavia asioita oli enemmän kuin aikaisemmissa päivityksissä. Tämä johtui siitä, että käytössä olevaa järjestelmää on kehitetty jatkuvasti, ja sen koko on kasvanut paljon vuosien saatossa. Työssä käytettiin tietojärjestelmän kehittämiseen tarkoitettuja hyväksi havaittuja toimintamalleja soveltuvin osin. Nämä toimintamallit olivat järjestelmän käyttäjäryhmien pohdinta ja käyttäjäryhmille kohdennetut järjestelmätoiminnot. Laaja käyttäjien haastattelu testiautomaation parannusehdotusten suhteen oli tuottoisa ja uusi työvaihe edellisiin järjestelmäpäivityksiin verrattuna.

Suurin haaste työn tekemisessä oli rajata työssä käsiteltävät aiheet riittävän tarkasti, jotta niihin pystyttiin perehtymään riittävällä tarkkuudella. Työhön käytettävä aika ei mahdollistanut niin tarkkaa vaatimusten tarkennusta, jotta sen pohjalta voitaisiin aloittaa konkreettinen kehitystyö. Vaatimusten tarkentaminen tulee olemaan tästä insinöörityöstä alkaneen kehitysprojektin seuraava askel.

Haasteita työn aikana tuotti myös vaatimusmäärittely kokonaisuudessaan, koska työn luonne oli melko uutta allekirjoittaneelle sekä muulle luotettavuuslaboratorion henkilökunnalle. Tästä johtuen työssä piti edetä varovasti. Työstä tuli kuitenkin arvokasta kokemusta ja insinöörityön tavoite saatiin täytettyä.

Lähteet

Asiantuntijahaastattelu. 2015a. Luotettavuusteknikko. Kohdeyritys. 3/2015

Asiantuntijahaastattelu. 2015b. Beckhoff Automation Oy. Sähköpostiviesti 23.4.2015.

Asiantuntijahaastattelu. 2015c. Projektipäällikkö. Kohdeyritys. Sähköpostiviesti 5.8.2015

Advantech ADAM-6017. Verkkodokumentti. <http://www.advantech.com>. Luettu 22.7.2015.

Brooks, F.P. 1987. No Silver Bullet: Essence and Accidents of Software Engineering. IEEE Computer, Vol. 20, No. 4 (April 1987) s. 10-19

Forselius, Pekka. 2013. Onnistunut tietojärjestelmän hankinta. Helsinki: Talentum

Haikala, Ilkka & Mikkonen, Tommi. 2011. Ohjelmistotuotannon käytännöt. Helsinki: Talentum

Heinonen, Vilho. 2003. Koeajettavien nostinten ohjausjärjestelmä. Insinöörityö. Helsingin Ammattikorkeakoulu

Jokinen, Tapani. 2010. Tuotekehitys. Verkkodokumentti. <http://lib.tkk.fi/Reports/2010/isbn9789526033204.pdf>. Luettu 20.4.2015

O'Connor, P., Kleyner, A. 2012. Practical Reliability Engineering. United Kingdom: John Wiley & Sons, Ltd.

Reliability Design. 2013. Sisäinen dokumentti. Kohdeyrityksen intranet. Luettu 22.5.2015.

Suomen Automaatioseura. 2010. Teollisuusautomaation tietoturva. Verkkodokumentti. <https://www.viestintavirasto.fi/attachments/tietoturva/TeollisuusautomaationTietoturva.pdf>. Luettu 20.7.2015

Vuoristo, Aapo. 2006. Taajuusmuuttajien kiihdytetyn testausjärjestelmän suunnittelu ja rakentaminen. Insinöörityö. EVTEK-ammattikorkeakoulu.

Käyttäjähaastatteluista kerätyt vaatimukset

Järjestelmään lisättävät toiminnot

Jos priorisoinnin suurimman ja pienimmän numeron ero on isompi tai yhtäsuuri kuin 3 -> punainen, jos 2 -> keltainen, muuten vihreä

ID	Vaatimus	Priorisointi 1 (ei ollenkaan tärkeä) - 5 (erittäin tärkeä) A	B	C	D	E	F	G	Keskiarvo
1	Testipaikan ajopyynti, käynnistäminen ja vikojen kuittaaminen voidaan hoitaa testipaikan luona	5	5	5	4	5	5	5	4,9
2	Mittausdata on oltava yhdistettävissä testattavan laitteen aikajanaan	5	5	4	5	5	5	4	4,7
3	Testerissä on napit testien alasajoon ja näiden samojen testien ylösajoon	5	4	4	5	5	5	5	4,7
4	Testerissä mahdollisuus ladata sama sykli halutuille testipaikoille. Myös kaikkien testipaikkojen käynnistäminen/sammuttaminen yhdellä napin painalluksella pitää olla mahdollista	5	4,5	5	5	4	5	4	4,6
5	Testipaikalla tulee olla mahdollista asettaa useampi ohjauskäsky rinnakkain. Myös usemman feedbackin käyttö täytyy olla mahdollista	5	5	4	5	4	4	5	4,6
6	Testattavan laitteen vikaantumishetken ympäriltä tallennetaan mitattavista muuttujista arvot ja ohjauksesta/feedbackeista tilat	5	5	5	5	4	3	5	4,6
7	Nostintestissä sillansiirto pitää olla mahdollinen	4	5	5	5	4		4	4,5
8	Testipaikkaa pitää pystyä ohjaamaan mittauksen perusteella (esim. lämpötilan nousu aiheuttaa vian). Testipaikan pitää myös pystyä käynnistämään mittaus jollakin herätteellä	4	5	5	5	5	3	3	4,3
9	Testerin mittausdata saadaan halutessa ulos standardimuotoisessa tiedostossa	4	4	4	4	5	5	4	4,3
10	Testipaikalla tulee olla mahdollista asettaa n määrä syklejä, joita voi ajaa haluamassaaan järjestyksessä	4	5	5	5	4	2	5	4,3
11	Testipaikalta tulee saada graafinen kuvaaja mittauksista	5	5	3	4	4	4	5	4,3
12	Testipainon sijainnin mittaaminen. Mahdollistaa nostonopeuden mittaamisen ja uudentyyppiset rajat	5	5	3	3	4	3	5	4,0
13	Testipaikalla tehdyt muutokset tallentuvat aina loggeriin. Loggerista selviää, että kuka on tehnyt muutoksia ja milloin. Muutosten kommentointi myös mahdollista	3	4	3	3	5	2	5	3,6
14	Valvomoon pitää olla mahdollisuus päästä henkilökohtaisella kannettavalla tietokoneella	4	2	4	3	4	3	3	3,3
15	Testipaikka muistuttaa lähestyvästä huollosta	3	5	2	3	3	3	4	3,3
16	Testerin testipaikka pitää pystyä "lukitsemaan"	4	2	2	5	2	2	5	3,1
17	Testeri tallentaa ympäristöstään tietoa (lämpötila, kosteus)	2	2	3	3	3	3	3	2,7