Sami Eskola

Testausohjelmiston toteutus ja validointi

Metropolia Ammattikorkeakoulu

Insinööri (AMK)

Elektroniikka

Insinöörityö

5.5.2015

Metropolia

Tekijä Otsikko Sivumäärä Aika	Sami Eskola Testausohjelmiston toteutus ja validointi 34 sivua + 1 liite 5.5.2015
Tutkinto	Insinööri (AMK)
Koulutusohjelma	Elektroniikka
Ohjaaja	Vanhempi testaussuunnittelija Tero Leskinen Lehtori Janne Mäntykoski

Tässä insinöörityössä suunniteltiin ja toteutettiin automaattinen testausasema osaksi Vaisalan painelaitetuotannon valmistusprosessia. Testiaseman tarkoitus on parantaa Vaisalan painelaitteiden luotettavuutta ja havaita vikaantuneet laitteet mahdollisimman aikaisessa vaiheessa tuotantoketjua. Samalla tavoitteena oli parantaa Vaisalan painetuotteiden sarjanumeroseurattavuutta ja jäljitettävyyttä.

Testiaseman suunnittelu toteutettiin osana Vaisalan testauskehitys-tiimiä ennalta määrätyn testaussuunnitteluprosessin mukaisesti. Ohjelmallinen toteutus suunniteltiin Microsoft Visual Studio-ohjelmalla C#-ohjelmointikielellä ja National Instruments Test Stand testauskehitysohjelmistolla.

Työn tuloksena saatiin valmistettua testiasema, joka vastasi kaikkia sille määriteltyjä vaatimuksia. Testiasema otettiin käyttöön Vaisalan tuotantoon huhtikuussa 2015.

Työssä käsitellään myös testauksen merkitystä osana ohjelmisto- ja testauskehitystä sekä käydään läpi erilaisia testausmenetelmiä. Lopuksi työssä käydään vaiheittain läpi suunnitellun testerin testaus ja ohjelmistovalidointi.

Avainsanat	Ohjelmistosuunnittelu, testaus, validointi

Author Title Number of Pages Date	Sami Eskola Designing and Validating Test Station Software 34 pages + 1 appendix 5 May 2015
Degree	Bachelor of Engineering
Degree Programme	Electronics
Instructor	Tero Leskinen, Senior Test Engineer Janne Mäntykoski, Senior Lecturer

In this Bachelor's thesis the aim was to design and develop an automatic test station which will be implemented as part of Vaisala's pressure devices manufacturing process. The purpose of the new test station was to improve pressure devices' reliability and catch the faulty device individuals in the manufacturing process as early as possible. At the same time the goal was to make Vaisala's pressure devices' serial number tracking and traceability better.

Designing the test station was executed by being a part of Vaisala's test development team. Designing process advanced with pre-specified development process. Test software development was executed with Microsoft Visual Studio software development program with C# programming language and with National Instruments Test Stand test management software.

As a result a working test station was produced. Test station fulfilled every assigned requirements. Test station was implemented to Vaisala's production in April 2015.

This thesis also discusses the significance of testing in software and test development. Also different software development models and testing methods are covered. At the end of this thesis the testing and validation of the manufactured test station are reviewed step by step.

Keywords	Software design, testing, validation

Sisällys

Lyhenteet ja sanasto

BAROCAP	Vaisalan valmistama paineanturi
BARO-1	Painelaitteiden käyttämä painelähetin
PTB330	Digitaalinen barometri
PTU300	Digitaalinen yhdistetty paine-, kosteus- ja lämpömittari
PTB210	Pääosin ulkokäyttöön tarkoitettu digitaalinen barometri
Operaattori	Tehtaan elektroniikka-asentaja/kokoonpanija, testerin käyttäjä
GPIB	Hewlett-Packardin kehittämä IEEE-488 standardin liitäntä mittalaitteille
RS-232	(Recommended Standard 232) Sarjamuotoinen tietoliikennestandardi
C#	Microsoftin kehittämä C-ohjelmointikielestä polveutunut ohjelmointikieli
UUT	Unit Under Test, testattava laite

1 Johdanto

Tämä insinöörityö käsittelee Vaisala Oyj:n toimeksiantamaa painelähettimien esitestausaseman suunnittelua ja testausaseman ohjelmistovalidointia. Testausaseman on tarkoitus havaita lyhyellä testillä mahdollisesti vikaantuneet painelähettimet ennen laitteen lämpötilariippuvuuden määrittämistä. Testerin tulee samalla syöttää testattavalle laitteelle jäljitettävyystietoja, kuten laitekohtainen sarjanumero, ja kirjoittaa anturin eränumero talteen Vaisalan laitejäljitettävyyskantaan. Testiasema tulee osaksi Vaisalan paineenmittauslaitteiden valmistusprosessia.

Tämän työn teoriaosiossa käydään läpi, mitä testaus käsitteenä tarkoittaa. Työssä selvitetään testausohjelmistojen vaatimusmäärittelyn ja validoinnin käsitteitä sekä tutkitaan niiden tärkeyttä suunnitteluprosessissa. Käytännönosiossa keskitytään Vaisalan painemoduulien esitestausaseman suunnitteluprosessiin; vaatimusmäärittelyyn sekä ohjelmistovalidointiin.

Työssä käytettyjä mitta-arvoja ja tuloksia ei kaikkia voida julkistaa opinnäytetyössä Vaisalan vaitiolovelvollisuuden nojalla. Tämän vuoksi osasta mittaustuloksia käsittelevistä kappaleista on jouduttu jättämään tarkemmat lukuarvot pois.

2 Lähtötilanne ja tavoitteet

Vaisala Oyj pyrkii jatkuvasti parantamaan tuotantonsa tehokkuutta ja sujuvuutta. Tämän vuoksi Vaisalassa on otettu käyttöön lean-tuotantomalli.

> Lean on asiakaslähtöinen prosessijohtamisen malli. Se perustuu virtauksen (exit rate) maksimointiin ja hukan (menetetty aika) poistamiseen. Se on siis toiminta ja ajattelutapa, jossa virtausta ja jalostusarvon osuutta maksimoidaan poistamalla hukkaa. [1.]

Jatkuvasti kiristyvän markkinakilpailun vuoksi Vaisalan painetuotteiden tuotantolinjaa kehitetään parhaillaan vastaamaan nykyisiä vaatimuksia lopputuotteiden laadun ja tuotantoputken läpimenoajan suhteen. Laatua valvotaan testaamalla valmistettuja laitteita ja laitteiden eri osia mahdollisimman monipuolisesti ja tarkasti ennen lopputuotteen lähettämistä asiakkaalle. Tuotannon läpimenoaikaa pyritään lyhentämään löytämällä mahdolliset tuoteviat ja materiaaliongelmat heti niiden ilmaantumisen jälkeen. Näin säästytään hukkatyöltä ja turhilta kustannuksilta.

Laitteiden osille tarvitaan myös parempi jäljitettävyys, jotta lopullinen tuote voidaan yhdistää siihen käytettyjen osien eri tuotantoeriin ja tuoteyksilökohtaisin sarjanumeroihin.

Tehtävänä oli suunnitella ja toteuttaa testausohjelmisto, jonka tarkoituksena on varmistaa painemoduulien juotoslaatu, BAROCAP-paineanturin toiminta ja kirjata anturikohtainen eränumero ja piirilevykohtainen sarjanumero Vaisalan jäljitettävyystietokantaan. Testerin tulisi myös tulostaa tuotetarrat, joista ilmenee tuotteen nimi, sarjanumero ja mitattava painealue.

3 Testaussuunnittelun ja ohjelmistovalidoinnin teoriaa

> Ohjelmistotestaus on prosessi, tai sarja prosesseja, joiden tarkoitus on varmistaa, että ohjelmisto tekee juuri sen mitä sen on suunniteltu tekevän ja ettei se tee mitään tarpeetonta. Ohjelmiston tulisi olla ennakoitava ja johdonmukainen, eikä sen pitäisi tuottaa yllätyksiä käyttäjälleen. – Glenford Myers [2.]

Monet elävät käsityksessä, että testaamalla osoitetaan testattavan tietokoneohjelman virheettömyys. On kuitenkin erittäin epätodennäköistä, että ohjelmistot olisivat täysin virheettömiä, kun ne päätyvät testattavaksi. Tämän yleisen harhaluulon vuoksi Glenford Myers määritteli jo vuonna 1979 julkaistussa kirjassaan testauksen olevan ohjelmiston suorittamista ennemminkin virheiden löytämiseksi kuin virheiden puuttumisen todistamiseksi. Edellä mainitun vuoksi testaus kannattaa liittää osaksi ohjelmistosuunnittelua jo sen alkuvaiheista saakka ja sen tulisi olla mukana koko suunnittelu-, toteutus- ja käyttöönottoprosessin ajan.

Tämän luvun ensimmäisessä osiossa käydään läpi ohjelmistotestauksen määrittelyä ja termistöä. Toisessa osassa selvitetään erilaisia testausmenetelmiä ja tapoja suunnitteluprosessissa.

3.1 Mitä on testaus?

Ohjelmistotestaus on yksi tärkeimpiä, ellei tärkein vaihe ohjelmistosuunnittelussa ohjelman loppukäyttäjän ja testattavan tuotteen toiminnan kannalta. Vaatimusmäärittely ja hyvä suunnittelu ovat toki tärkeitä, mutta jos testaus on jätetty liian pienelle huomiolle tai on kiireellä tehty, saattaa huomaamatta jäänyt virhe tehdä joitakin ohjelmiston toimintoja täysin turhaksi tai toimimattomaksi.

Testaus on myös aikaa vievimpiä ja siksi myös suunnitteluprojektin kannalta kalleimpia työvaiheita ohjelmistosuunnittelussa. Ideaalitilanteessa lopullisessa ohjelmassa ei suunnittelun ja toteutuksen jälkeen olisi yhtään virhettä, ristiriitaa tai toimimatonta osaa. Teollisuuden yleissääntönä voidaan sanoa, että yksi viidesosa lähdekoodista sisältää neljä viidesosaa virheistä ja että kymmenen prosenttia löydetyistä virheistä vie yhdeksänkymmentä prosenttia korjauksille varatusta työajasta. Tietokoneohjelmiston

täydellinen testaaminen on kuitenkin täysin mahdotonta ja testaaminen onkin siksi loputon prosessi ohjelmiston koko elinkaaren ajan. [3.] Tästä johtuen jokaisen testaussuunnitteluprojektin viimeinen työvaihe on ohjelmiston ylläpito.

On olemassa kahdenlaista testausta: staattista ja dynaamista. Staattinen testaaminen on jonkin testattavan laitteen tai ohjelmakoodin testaamista automaattisesti jonkin testausohjelmiston tai lähdekoodianalysaattorin avulla. Dynaaminen testaaminen on käsin tehtävää testausta ilman automatiikkaa. Kattava ohjelmistotestaus pitää sisällään sekä staattista että dynaamista testausta.

Testauksen voi käsitteenä jakaa kahteen pääryhmään: verifiointi ja validointi.

3.1.1 Verifiointi

Verifioinnin määritys: *Todentaa, vahvistaa, todistaa oikeaksi.* [4.] Verifiointi tutkii, onko tuote tehty oikein. Onko tuote sen mukainen, mitä on suunniteltu tehdä?

Pitkäaikaisissa ja laajoissa ohjelmistokehitysprojekteissa ohjelmiston verifiointi alkaa heti projektin alkuvaiheessa ja kestää määrittelyn, suunnittelun, toteutuksen ja osittain testauksen ajan.

3.1.2 Validointi

Validioinnin määritys: *Luotettavuus, paikkansapitävyys, (oikeudellinen) pätevyys, voimassaolo* [4] Validointi tutkii, onko tehty oikea tuote. Onko ohjelma rakennettu oikein ja sen mukainen, kuin suunnitteluvaiheessa on määritelty ja vaatimuksien mukainen?

On valitettavan yleistä, että validoinnin merkitystä vähätellään. Ohjelmiston validointi mielletään useimmiten ohjelmistokehityksen viimeiseksi vaiheeksi, joka päättyy ohjelmiston valmistumiseen. Ohjelmiston validointi alkaa kuitenkin vasta ohjelmiston suunnitteluvaiheen loppupuolella ja jatkuu suunnitteluprosessin jälkeenkin lopullisen ohjelmiston koko elinkaaren loppuun asti. Jos testattavaan ohjelmistoon tehdään minkäänlaisia muutoksia, joilla on vaikutusta ohjelmakoodin suoritettavaan toimintoon, täytyy se huomioida validoinnissa ja testata tarkoin läpi uudestaan. Jatkuva validointi saattaa tuntua mitättömältä tai jopa täysin turhalta, mutta ohjelmiston mahdollisen

pitkän elinkaaren vuoksi ja niin kutsutun ohjelmavirheiden lumipalloefektin välttämiseksi validointia ei sovi jättää liian vähälle huomiolle. Pahimmassa tapauksessa monen vähäpätöiseksi mielletyn validoimattoman ohjelmistomuutoksen seurauksena koko ohjelmisto voi pirstaloitua täysin toimimattomaksi ja mahdottomaksi ylläpitää.

Eräs validoinnin merkittävä osa, joka vaatii erityisen maininnan on reliabiliteetti, eli mittauksen toistettavuuden todentaminen. Testataan, että saman tuotteen toistuvissa mittauksissa ei tule liian suurta mittausheittelyä. [5.]

Erityisesti ohjelmistoissa, jotka suorittavat toistuvaa automatisoitua sekvenssiä, on ohjelman toistettavuuden todentaminen erittäin merkittävässä roolissa. Tehokas tapa testata esimerkiksi automaattisen testerin toistettavuutta on toteuttaa mahdollisimman laajamittainen Gage R&R (Gage Repeatability and Reproducibility).

Gage R&R tehdään esimerkiksi siten, että kolme eri operaattoria testaa kymmentä eri tuoteyksilöä. Jokainen operaattori testaa yksittäisen laitteen kolmeen otteeseen [6.]. Testien tulokset syötetään tulosten analyysiohjelmaan kuten Minitabiin, jolla tutkitaan, antaako testeri luotettavia ja toistuvia testaustuloksia samoille laitteille riippumatta ympäristömuuttujista, kuten eri operaattoreista, mittauspaikasta ja testausajankohdasta.

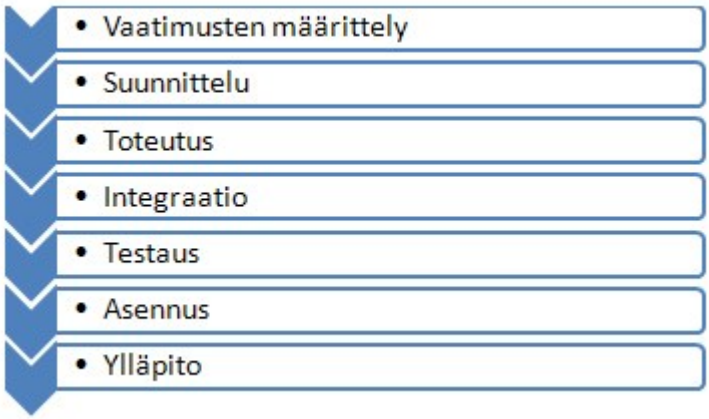
3.2 Suunnitteluprojektimallit testauksen näkökulmasta

Projektitöissä käytetään lähes aina jotain ennalta sovittua projektimallia ja ennakkoon määriteltyä aikataulua. Yleisesti eri yrityksillä on yksi ennalta sovittu prosessimalli, jonka mukaan kaikki projektit etenevät. Suurilla organisaatioilla, joissa projektitöitä tehdään paljon, on useimmiten tarkkaan yritykselle yksilöllisesti suunniteltu prosessi, jonka mukaan kaikki yrityksen omat projektit etenevät. Vaikka yrityksen käytössä oleva projektimalli olisi yritykselle yksilöllisesti laadittu, perustuu suurin osa malleista neljään yleisimpään projektimalliin. Yleisimpiä projektimalleja ovat vesiputous-, V-, RUP-mallit. Testauksen näkökulmasta kaikki projektimallit eivät sovellu paljon testausta sisältävään työhön ja käytettävää projektimalli täytyy valita tarkkaan ennen projektin aloittamista.

3.2.1 Vesiputous

Käytännössä vesiputousmalli etenee hyvin selkeästi vaihe vaiheelta ja on siksi helposti seurattavissa. Projektin aikatauluttaminen on helppoa verrattuna muihin malleihin sen selkeän tehtävänjaon vuoksi. Vesiputousmallissa projekti alkaa aina vaatimusten määrittelyllä. Kun määrittely on tehty, siirrytään seuraavaan vaiheeseen, jossa suunnitellaan projektin etenemisen kaikki tarvittavat vaiheet. Suunnittelua seuraa työn toteutus, integraatio, testaus, asennus ja ylläpito (kuvio 1). Malli etenee nimensä mukaisesti putoavassa järjestyksessä ylhäältä alaspäin. Edeltävään työvaiheeseen on mahdollista palata vielä seuraavasta vaiheesta, mutta projektinhallinnallisesti se ei ole kovinkaan toimiva ratkaisu. Alkuperäinen projektiaikataulu muuttuu helposti eikä projektin etenemistä voi ennustaa riittävällä tarkkuudella. Testauksen kannalta vesiputousmalli on todella taipumaton sen ehdottoman ja karkean työvaihe-erottelun vuoksi, eikä siksi sovellu erityisen paljon testausta vaativiin projekteihin. [7.]

Vesiputousmalli on vanhin ja yksinkertaisin nykyisin tunnettu projektimalli ja siksi sen käyttö projektityössä on ehkä vielä tänä päivänäkin yleisintä, vaikka se ei sellaisenaan välttämättä soveltuisikaan nykyisiin laajoihin projektitöihin.

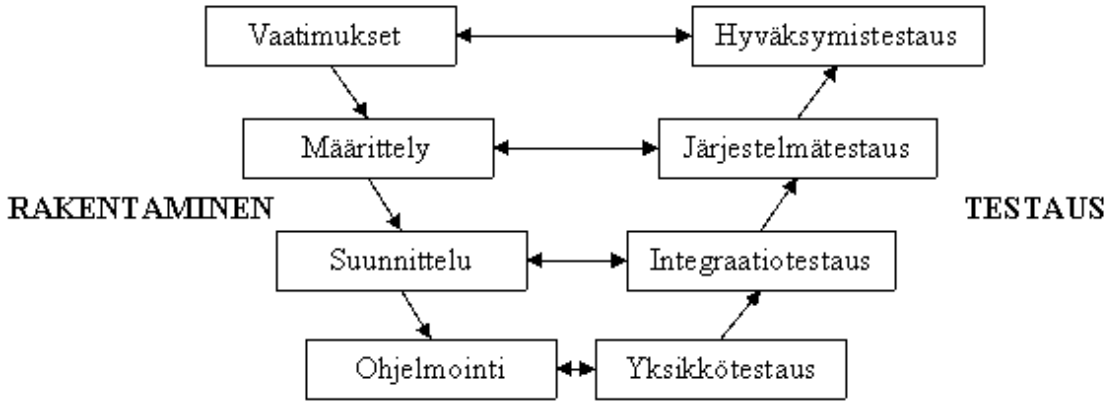


Kuvio 1. Vesiputousmallin kulku [8.]

3.2.2 V-projektimalli

V-malli on projektimalli joka on kehittynyt vesiputousmallin pohjalta. Kuten vesiputousmallissa, projekti etenee työn vaatimusmäärittelyllä, suunnittelulla ja toteutuksella. Tämän jälkeen testaus toteutetaan neljässä vaiheessa jokaiselle

testausta edeltäneelle työvaiheelle käänteisessä järjestyksessä. Testaus aloitetaan toteutuksen testauksella yksittäisten tekijöiden osalta, josta siirrytään yleiseen kokonaisvaltaisen toiminnan testaukseen. Lopuksi varmennetaan kokonaisuuden toiminta lopullisessa käyttöympäristössä (kuvio 2). [9.]



Kuvio 2. V-projektimallin kulku [10.]

V-projektimalli soveltuu erityisesti ohjelmistokehitykseen merkittävästi vesiputousmallia tehokkaammin. Projektimalli etenee V-muodossa kuvaajan vasenta sakaraa alaspäin rakentamisen osalta ja nousee oikeanpuoleista sakaraa ylöspäin testauksen osalta. Palaaminen testauksesta takaisin rakentamisvaiheeseen on myös mahdollista. Testaukseen on kiinnitetty enemmän huomiota, ja suunnittelussa tapahtuneiden virheiden löytäminen on huomattavasti todennäköisempää. Mallin heikkona puolena on sen kertakäyttöisyys projektissa. Kun kokonaisuus on kertaalleen testattu ja projektissa luotuun ohjelmistoon tehdään muutos, ei saman projektimallin käyttäminen enää toimi. Tällöin muutokset vaativat kokonaan uuden projektin aloittamisen, jossa käytetään kulloinkin sopivaksi määriteltyä projektimallia. Ylläpidettävyyden ja pitkän tuotteen elinkaaren suhteen V-mallin käyttö ei ole kannattavin vaihtoehto.

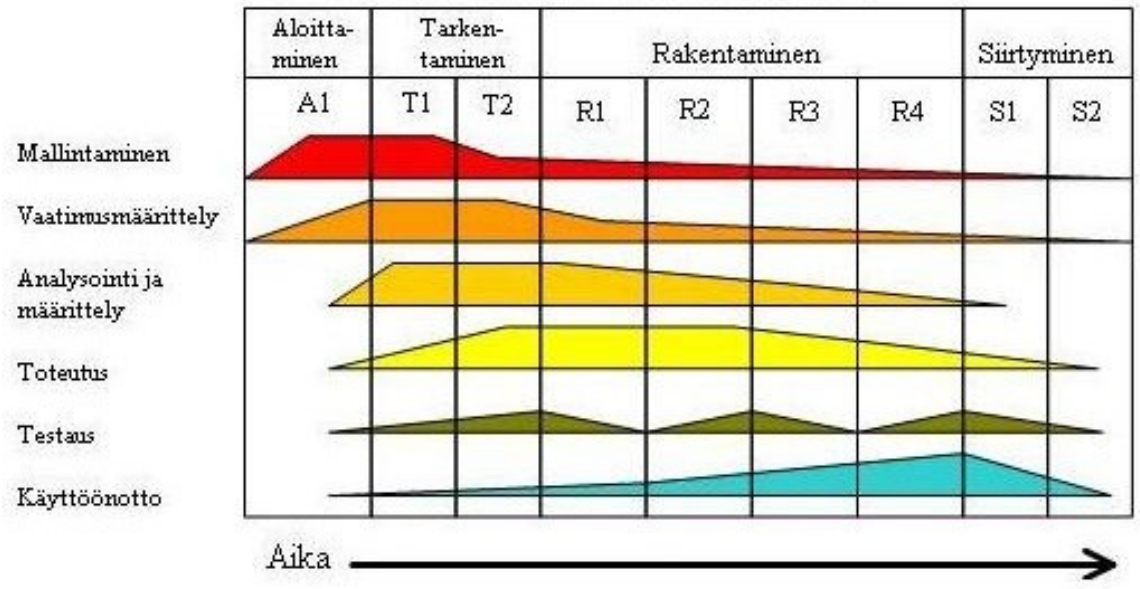
3.2.3 RUP

RUP lyhenne tulee sanoista Rational Unified Process. Alun perin International Business Machines (IBM) loi menetelmän omaan käyttöönsä parantaakseen ohjelmistokehitystänsä vuonna 1999. RUP-menetelmä on luotu vesiputousmallin ja V-projektimallin vahvoja puolia silmälläpitäen vastaamaan juuri ohjelmistokehityksen vaatimuksia. Ajatuksena oli luoda projektimalli, joka rakentuu useista yksinään

toimivista prosesseista, joista iteratiivisesti rakennetaan suurempi kokonaisuus. Lähtökohtaisesti RUP keskittyy kuuteen ohjelmistotuotannon parhaaksi havaittuun käytäntöön:

- Kehitystyö pitää tehdä vaiheittain osa kerrallaan, aloittaen tärkeimmästä osasta.
- Vaatimuksia pitää hallita, päivittää ja valvoa koko kehitysprojektin ajan.
- Ohjelmat pitää rakentaa siten, että ne muodostuvat joukosta itsenäisesti toimivia komponentteja.
- Mallien tekemiseen pitää käyttää visuaalista suunnittelua.
- Laatua pitää valvoa jatkuvasti.
- Muutokset on tehtävä hallitusti. [7; 11.]

RUP-malli jakautuu neljään pääryhmään: aloittaminen, tarkentaminen, rakentaminen ja siirtyminen. Pääryhmien sisään lukeutuvat kaikki vesiputousmallin vaiheet. Tuotteen kehitys etenee tarkkaan hallinnoiduissa sykleissä jokaisen pääryhmän läpi ja jokaiselle vaiheelle on luotu oma vaatimusmäärittelynsä. Testaus kulkee koko prosessin ajan mukana jokaisessa syklissä sen merkityksen lisääntyessä projektin edetessä (kuvio 3).



Kuvio 3. RUP-projektimallin kulku [12.]

Työn aloittamisvaihe pitää sisällään vaatimusmäärittelyyn ja projektinhallintaan liittyvät suunnittelut. Tässä vaiheessa päätetään projektin aloittamisesta ja määritellään projektin aikataulu ja eteneminen.

Tarkentaminen pitää sisällään teknisen suunnittelun, kuten vaatimusmäärittelyn ja tarkan kuvauksen toteutuksen etenemisestä. Tarkennusvaiheessa kerätään kaikki projektin kannalta hyödylliset entuudestaan olemassa olevat ja muissa projekteissa tehdyt ohjelmamoduulit ja valmiit ohjelmistot, joita tarvitaan työn toteutukseen. Testaus aloitetaan heti, kun projektissa tulee eteen jotain testausta vaativaa, kuten entuudestaan olemassa olevien ohjelmamoduulien toimivuus ja yhteensopivuus meneillään olevaan projektiin.

Rakentamisvaiheessa luodaan itse ohjelmisto. Kehitys aloitetaan niistä osista, jotka ovat kriittisimpiä koko projektin kannalta. Kun osat ovat valmiita, aloitetaan niiden testaus niin pian kuin mahdollista. Näin vältytään ikäviltä yllätyksiltä projektin myöhemmissä vaiheissa ja ylimääräiset kustannukset ja aikataulun muuttumiset tulevat esille mahdollisimman aikaisessa vaiheessa. Rakentamisvaiheen lopussa ja siirtymävaiheeseen siirryttäessä testaamisen merkitys on suurimmillaan. Tässä vaiheessa testaus pitää sisällään tuotteen validoinnin ja verifioinnin.

Siirtymävaihe pitää sisällään valmiin tuotteen siirtämisen tuotantoon. Tuotteen ei ole pakko olla lopullisessa muodossaa vielä siirtymän alkuvaiheessa, vaan siihen voi tehdä vielä muutoksia, joiden tarve ilmenee vasta käyttöönoton yhteydessä. Juuri ennen lopullista käyttöönottoa tehdään vielä viimeinen pienimuotoinen validointi, jolla varmistetaan lopullisen tuotteen toiminta. Kun kehitysprojekti saatetaan kokonaisuudessaan päätökseen, alkaa viimeinen työvaihe, ylläpito, joka kestää koko ohjelmiston loppuelinkaaren.

Menetelmän hyvänä puolena on sen monipuolisuus ja testaamisen läsnäolo koko projektin ajan. Projekti etenee useissa pienissä osioissa, jolloin projektin seuranta ja aikataulun arviointi on helppoa. Suurilta yllätyksiltä ja kustannuksilta on mahdollista välttyä kokonaan tai niistä saadaan tieto hyvissä ajoin ennen mahdollisten ongelmakohtien paisumista liian suuriksi.

3.3 Testausmenetelmät

Testausohjelmiston testaaminen on jaoteltu erilaisiin testausmenetelmiin. Yksittäisten osien yksityiskohtaisen testaamisen lisäksi on useita menetelmiä testata ohjelmiston toimintaa kokonaisuutena. Huolellisesti tehty ohjelmistotestaus kattaa kaikki seuraavat testivaiheet, jotta virheellisen ja arvaamattoman toiminnan minimointi olisi tehokkaimmillaan.

Savutestaus: Testivaihe ennen varsinaista syvällisempää ohjelmistotestausta, jossa varmistetaan kaikkien perustoimintojen oikea toiminta. Savutestit ovat pieniä testejä, joilla varmistetaan itsestäänselvyytenä pidettävien asioiden toiminta, kuten käynnistyykö testattava ohjelma ja toimivatko kaikki ohjelmiston näppäimet. Savutesteihin luokitellaan myös ennalta arvaamattomien tilanteiden testaus, kuten sähkökatkot tai mahdollisten oheislaitteiden hajoaminen.

Musta laatikko -testaus: Testissä varmistetaan, että testattava ohjelmisto tekee juuri sen, mitä sen odotetaankin tekevän.

Esimerkki musta laatikko -testauksesta: Ohjelmistolle syötetään kolme numeroa: 2, 4 ja 6. Ohjelmaa pyydetään laskemaan luvut yhteen ja antamaan tulos. Testaaja varmistaa, että annettu tulos on 12.

Lasilaatikkotestaus: Testissä varmistetaan, että testattava ohjelmisto tekee juuri sen, mitä sen odotetaankin tekevän juuri sillä tavalla kuin on tarkoitus.

Esimerkki lasilaatikkotestauksesta: Ohjelmistolle syötetään kolme numeroa: 2, 4 ja 6. Testaaja varmistaa, että testeri laskee juuri nämä luvut yhteen, eikä tee mitään muuta ylimääräistä. Tämän jälkeen testaaja varmistaa, että annettu tulos on 12.

Regressiotestaus: Testissä varmistetaan että jo entuudestaan aikaisemmassa testivaiheessa testattu toiminnallisuus on edelleen voimassa oleva, kun testiin lisätään jokin uusi ominaisuus.

Uusi ohjelmakoodi sisältää todennäköisemmin virheitä kuin vanha jo entuudestaan validoitu ohjelmakoodi. Vanha koodi, jota on muutettu tai paranneltu, on toiminnallisesti katsottuna kokonaan uutta koodia. Vanha ohjelmakoodi, johon on lisätty

ominaisuuksia, on myös käytännössä uutta koodia. Uusi oheislaite, kuten uusi jigi, voi aiheuttaa muutoksia ohjelman toimintaan. Kaikissa edellä mainituissa tapauksissa regressiotestaus on välttämätön. Jopa uudelleen validointi saattaa olla tarpeen, jos ohjelmakoodiin tehdyt muutokset ovat olleet laajoja tai koskettaneet usean muun koodimoduulin toimintaa.

Kuormitustestaus: Jos testattavaa ohjelmistoa pystyy käyttämään useampi käyttäjä samanaikaisesti, testataan, että ohjelmisto pystyy suoriutumaan odotetuista toiminnoista ilman ongelmia usean käyttäjän käyttäessä ohjelmaa samaan aikaan.

Kuormitustestauksessa testataan myös pitkäaikaista kuormitusta ennalta määritellyn ajan. Tällä testauksella varmistetaan, että jos ohjelmisto esimerkiksi varaa muistia testitietokoneelta jokaisella toiminnan suorituksella tietyn määrän, se myös vapauttaa varatun muistin toiminnon päätyttyä myöhempää käyttöä varten. Muussa tapauksessa ohjelmisto täyttää tietokoneen muistin, joka lopulta johtaa mahdollisesti koko laitteiston jumiutumiseen.

4 Testiaseman suunnittelu ja toteutus

Tässä luvussa käydään läpi Vaisala Oyj:lle toteutetun painelähettimen esitestausaseman suunnittelu, toteutus, verifionti ja validointi ja käyttöönotto.

Työn suunnitteluosa Vaisalassa toteutettiin kolmessa vaiheessa. Aluksi tutustuttiin testattaviin painelähettimiin, niiden teknisiin ominaisuuksiin ja laatuvaatimuksiin ja luoda esitietojen ja toimeksiannon pohjalta testin vaatimusmäärittely (*Test Specification*). Seuraavaksi tehtiin testauksen ohjelmointi ja muu ohjelmallinen toteutus. Tuotekohtaiset testausmäärittelyt, käyttöliittymä ja laiteajurit toteutetaan C#-ohjelmointikielellä. Viimeisenä tehtiin testerin ohjelmistovalidointi ja verifiointi (*Software validation*). Testaukseen sovellettiin tässä työssä käsiteltyä teoriaa sekä Vaisalan testaussuunnitteluprosessiin määriteltyjä validointivaatimuksia.

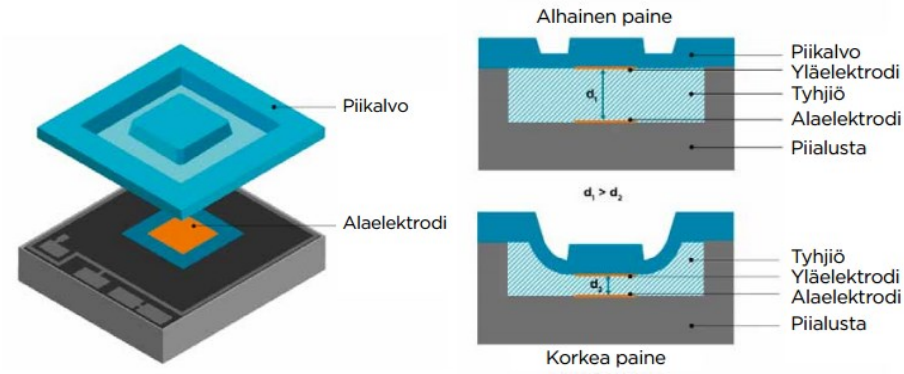
4.1 Insinöörityön aiheen valinta sekä aiheeseen perehtyminen

Tammikuussa 2014 ilmeni mahdollisuus insinöörityöhön Vaisalassa. Painelähettimien valmistusprosessia kehitettiin parhaillaan uuteen suuntaan ja prosessiin tarvittiin kokonaan uusi työvaihe, painelähettimien esitestaus.

Testaussuunnittelu ja ohjelmallinen toteutus oli määrä toteuttaa kolmen kuukauden aikana huhtikuun 2014 loppuun mennessä. Insinöörityössä suunnitteluprosessia lähestytään RUP-metodia soveltaen. (ks. luku 3.2.3).

4.2 Testattavat tuotteet

Testattaviin tuotteisiin kuului kolme erilaista Vaisalan valmistamaa painelähetintä: BARO-1, BARO-1A ja PTB210SUB. Kaikkien kolmen tuotteen paineanturina toimii Vaisala BAROCAP. BAROCAP on mikromekaaninen anturi, joka mittaa painetta piikalvon mittojen muuttumisen perusteella. Ympäröivän paineen kasvaessa tai vähentyessä kalvo taipuu, jolloin anturin sisällä olevan tyhjiön korkeus kasvaa tai pienenee. Tyhjiön vastakkaiset puolet toimivat elektrodeina (kuva 1). Kun näiden elektrodien välinen etäisyys muuttuu, myös anturin kapasitanssi muuttuu. Kapasitanssi mitataan ja muunnetaan painelukemaksi. [13.]



Kuva 1. BAROCAP-anturin rakenne [13.]

BARO-1- ja BARO-1A-moduulit ovat sekä toiminnaltaan että mitoiltaan hyvin samantapaiset. BARO-1 on täysin digitaalisella kommunikoinnilla toimiva painemoduuli, kun taas BARO-1A:lla on mahdollisuus lukea laitetta myös analogisena signaalina. Koska testauksessa on tarkoitus todentaa vain paineanturin toimintaa ja syöttää laitteelle tunnistetietoja, käytetään BARO-1- ja BARO-1A-moduulien kommunikaatioon kuitenkin vain digitaalista väylää. Laitteen digitaalisella kommunikaatiolla on käytössä neljä linjaa: V+, GND, RxD ja TxD. Koska väylät ovat samat kuin USB-kommunikaatiossa, oli järkevintä tehdä testiaseman liitännät USB-TTL-kaapelilla. Virransyöttö BARO-1- ja BARO-1A-laitteille tapahtuu erillisen liittimen kautta. BARO-1-moduuleja käytetään Vaisalan tuotteissa mm. PTB330 digitaalisessa barometrissa (kuva 2) ja PTU300, jossa yhdistetty ilmanpaine-, kosteus- ja lämpötilalähetin. BARO-1A-lähettimiä käytetään PTB110-barometrissa (kuva 3).

Kuva 2. Vaisala PTB330 digitaalinen barometri [14.]

Kuva 3. Vaisala PTB110-barometri [14.]

PTB210SUB:n kommunikointi testerin kanssa kulkee RS-232 väylää pitkin, josta käytössä ovat linjat RxD, TxD ja GND. PTB210SUB-virransyöttö tapahtuu samasta liittimestä kuin kommunikointi. Tämän vuoksi testerin suunnitteluvaihetta varten täytyi valmistaa erillinen testiliitin, jossa yhdistyy virransyöttö ja RS-232-kommunikointi. Vaisala PTB210 digitaalinen barometri (kuva 4) on pääosin ulkokäyttöön tarkoitettu barometri.

Kuva 4. Vaisala PTB210 digitaalinen barometri [14.]

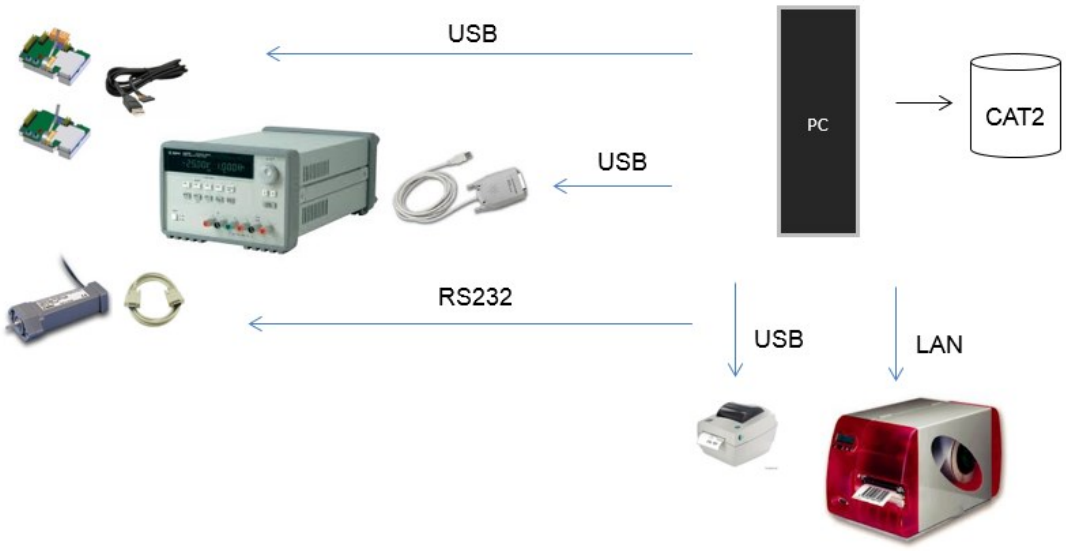
Koska BARO-1 ja BARO-1A ovat kaikin puolin hyvin samantapaiset, tehtiin laitteille yksi yhteinen testipenkki. Sekä BARO:n että PTB210SUB-testipenkin mekaanisen suunnittelun toteutti Otto Vainikka, ja sen suunnitteluosio on jätetty tästä opinnäytetyöstä sen vuoksi pois.

Virransyöttö molemmilla testipenkeillä tapahtuu Agilent E3632A -jännitelähteellä, joka on yhdistetty testitietokoneeseen GPIB-väylällä. GPIB-väylä mahdollistaa jännitelähteen ohjauksen etänä testeritietokoneelta käsin ja on siten ohjelmoitavissa testisekvenssiin automaattisesti ohjattavaksi ilman, että operaattorin tarvitsee käyttää jännitelähdettä manuaalisesti testiä suorittaessa.

Testattavien laitteiden datalehdistä kävi ilmi laitteiden käyttöjännitealueet sekä virrankulutuksen rajat laitteen toimiessa normaalisti. Datalehdet ovat vain Vaisalan sisäiseen käyttöön ja jätetty siksi tämän työn ulkopuolelle. BARO-1- sekä BARO-1A-moduulit pitävät sisällään myös lämpötila-anturin, sillä molemmat laitteet mittaavat painetta lämpötilan suhteen. Siksi painemoduulin lämpötila-anturin toiminta pitäisi myös varmistaa. PTB210SUB ei pidä sisällään lämpöanturia.

4.3 Testauksen vaatimusmäärittely

Varsinainen testaussuunnittelu alkoi testerin vaatimusmäärittelyn dokumentoimisella. Aluksi määriteltiin, mitä tuotteita testataan ja mitä tuotteista halutaan testata. Lisäksi määriteltiin, mitä muita toimintoja testerin tulisi toiminnan todentamisen lisäksi tehdä. Dokumenttiin lisättiin testerin lohkokaavio ja suunniteltu kokoonpano (kuva 5).



Kuva 5. Testerin lohkokaavio testausmäärittelydokumentista [15.]

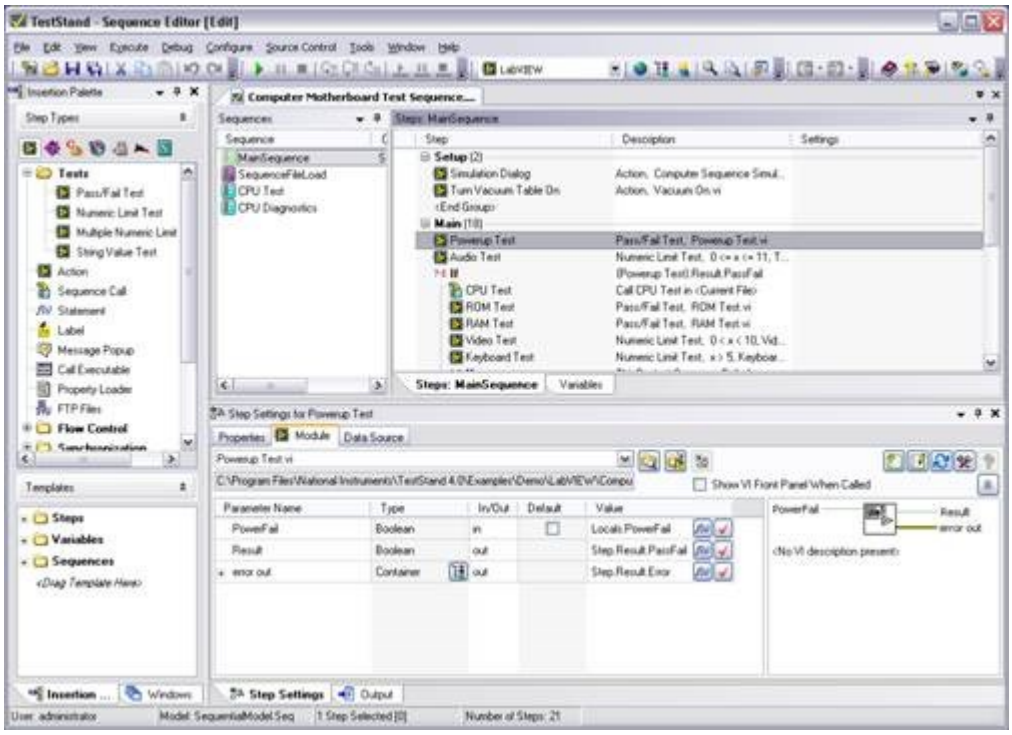
Vaatimusmäärittelyyn liitettiin testattavien tuotteiden piirustukset laitteiden fyysisistä mitoista sekä laitteiden piirikaaviot. Seuraavaksi kirjattiin ylös tuotekohtaiset testipisteet ja muut liitännät, kommunikointiasetukset ja kommunikointitapa. BARO-1- ja BARO-1A-kommunikointi toteutettiin 3,3 V TTL-sarjakaapelilla ja PTB210SUB kommunikointi RS-232-sarjakaapelilla.

Vaatimusmäärittelyyn laaditaan yhteenveto kaikista testerin suorittamista testeistä ja toiminnoista tuotekohtaisesti. Suoritettavat testit ja toiminnot määriteltiin lopullisessa järjestyksessään seuraavasti:

- virran syöttö ja virrankulutuksen mittaus
- kommunikointitesti
- ohjelmistoversion tarkistus ja kirjaaminen muistiin
- laitteen sarjanumeron syöttö ja varmistus, että numero on luettavissa laitteelta
- piirilevyn sarjanumeron tarkistus ja kirjaaminen jäljitettävyystietokantaan
- anturin eränumeron yhdistäminen laitteen sarjanumerolle ja syöttö jäljitettävyystietokantaan
- laitteen mitattavan painealueen syöttö CAT2-jäljitettävyystietokantaan
- BARO-1 ja BARO-1A laitteiden lämpötila-anturin toiminnan testaus ja luetun arvon vertaaminen tehtaan vallitsevaan lämpötilaan.
- BAROCAP anturin juotoslaadun varmistus ja anturin toiminta
- laitetarrojen tulostaminen testin päätyttyä.

4.4 Testausprosessin ohjelmointi

Testausprosessin testisekvenssin ohjelmointi tehtiin TestStand–ohjelmistolla. TestStand on National Instrumentsin kehittämä automaattisten testausten kehitystyökalu (kuva 6). TestStand tukee lähes kaikkia tunnettuja ohjelmointikieliä ja on visuaalisuutensa vuoksi helppokäyttöinen. Testisekvenssit rakennetaan moduuleista, jotka ohjaavat ennalta, vapaavalintaisella ohjelmointikielellä kirjoitettuja koodeja. [16.] Vaisalan nykyisenä uusien testereiden ohjelmointikielenä toimii C#.



Kuva 6. TestStand käyttöliittymän kuvakaappaus. [16.]

Laiteajurit ja ohjelmakoodi on tehty Microsoft Visual Studio 2010 –ohjelmistolla C# -kielellä. C#-ohjelmointikieli pohjautuu vahvasti C-ohjelmointikieleen ja on siksi C#-kieltä osaamattomallekin helposti omaksuttavissa. TestStand on loogisesti toteutettu ohjelma, sillä lähes ilman minkäänlaista kokemusta, ohjelmoija saa toteutettua koko testerin automaation vain pienehköllä avustuksella. Laiteajurit olivat jo tehtynä suunnittelutyötä aloittaessa, joten C# luonti jäi työn osalta hyvin vähäiseksi. C#:n ymmärrys tuli tarpeellisemmaksi vasta testerin virheenkäsittelyssä ja testauksessa.

Testattavien laitteiden testirajamäärittelyt laadittiin Vaisalan SpecEditorilla. Sovelluksella luodaan jokaiselle testattavalle tuotteelle oma .XML-tiedosto, jossa

määritellään kaikki testerin suorittamat testit ja niiden raja-arvot. Testirajamäärittelyyn kirjataan myös, mihin jäljitettävyystietokantaan testitulokset syötetään.

Jos testi on numeerinen, syötetään yhdelle testille kaksi erillistä ylä- ja alarajaa. Kapeammat raja-arvot määrittävät, mikäli testattavassa vaiheessa on vain pientä heittoa odotettuun mitta-arvoon. Raja-arvojen rikkoutuessa vaihetta ei kuitenkaan hyväksytä testin läpäisyyn. Testinkäyttöliittymä antaa tuolloin testin tulokseksi ”Fail”. Laajemmat raja-arvot taas määrittävät, että testattavassa vaiheessa on suurempaa vikaa, eikä se toimi ollenkaan esimerkiksi puuttuvan tai rikkoutuneen johtimen vuoksi. Testikäyttöliittymä antaa siinä tapauksessa testin tulokseksi ”Bogus”.

Esimerkki .XML-tiedoston BARO-1 virranmittauksesta:

```
<Test Name="Current Consumption" Optional="false">
<Comment>Current Consumption</Comment>
<Metadata DoesNotCauseFailure="false" Format="0.00" Unit="mA" />
<NumericCriteria LowLimit="0.0100000000" HighLimit="15.0000000000" BogusLow="-
1.0000000000" BogusHigh="60.0000000000" />
</Test>
```

TestStandissa määritellään testisekvenssissä jokaiselle testille sitä vastaava rivi .XML-tiedostosta, johon testituloksia verrataan.

Kaikki testit eivät välttämättä ole tutkittavissa numeerisena, vaan testirajamäärittelyssä on numeeristen testien lisäksi testejä, joissa verrataan testattavalta laitteelta saatua paluuarvoa boolean funktiona tosi-epätosi–muotoisena. Työn aikana tehdyssä testerissä tarkistetaan esimerkiksi, onko testeri havainnut virheitä testin aikana. Mikäli testit menevät läpi, antavat ne ”Error” muuttujalle arvon ”NO ERROR”, mutta jos testit eivät mene läpi, saa muuttuja ”Error” arvon ”ERROR” Jos luettu arvo on sama kuin odotettu arvo, saa testeri boolean funktiona arvon tosi.

Esimerkki .XML-tiedoston Error tarkastuksesta:

```
<Test Name="Error" Optional="false">
<StringCriteria Pattern="NO ERROR" Comparison="Equal" />
</Test>
```

4.5 Sarjanumerointi

Laitteen sarjanumerointi syötetään Vaisalan sarjanumeromäärittelyn mukaisesti.

Sarjanumero on kahdeksan merkkiä pitkä. Merkkijono alkaa kirjaimella, joka vastaa laitteen valmistusvuotta. Seuraavat kaksi numeroa kertovat viikon numeron ja kolmas luku kertoo, monesko viikonpäivä laitteen tekohetkellä on. Viimeiset neljä numeroa ovat päiväkohtainen juokseva järjestysluku.

Ensimmäinen merkki määräytyy siten, että esimerkiksi vuosi 2013 = J, 2014 = K ja 2015 = L …

Vuonna 2014 tekemiin validointeihin käytettiin sarjanumeroita K1500001 – K1500003.

Päivää osoittavan merkin kohdalla käytetään numeroa 0, jotta testit eivät jäljitettävyystietokannassa sekoittuisi myöhemmin tuotannossa testattavien laitteiden kanssa.

Sarjanumeron syöttö testerin käyttöliittymässä toteutettiin siten, ettei käyttäjä voi syöttää virheellistä sarjanumeroa (ks. luku 4.8.3).

4.6 Antureiden jäljitettävyys

Yksi testerin merkittävimmistä ominaisuuksista on saada Vaisalan tuotteille entistä parempi jäljitettävyys ja seurattavuus. Lopputuotteen sarjanumeron perusteella pystyy selvittämään jäljitettävyystietokannan avulla kaikki siihen käytetyt alikokoonpanot sarjanumeroittain, kuten moduulit ja anturit.

Testerin käyttöliittymä kysyy laitteeseen juotetun BAROCAP–anturin eränumeroa, joka on kirjattu anturin koteloon. Kun anturit saapuvat kokoonpanosoluun, tulee niiden mukana anturierän numero ja viivakoodi. Operaattori syöttää kunkin laitteen anturin eränumeron joko käsin sille kuuluvaan syöttökenttään tai vaihtoehtoisesti automaattisesti lukemalla anturiviivakoodin, jolloin käyttöliittymä täyttää anturieränumeron automaattisesti. Eränumero on muutoin samaa muotoa, kuin

sarjanumerointi, mutta käytössä on vain sarjanumeroperiaatteen neljä ensimmäistä merkkiä.

Kuten testerikäyttöliittymän sarjanumeron syötössä, myös anturieränumeron syöttö on toteutettu niin, ettei käyttäjä voi syöttää numeroa virheellisesti.

4.7 Juotoslaadunvalvonta

Testerin merkittävin testi ja pääsyy koko uuden testausprosessin lisäämiselle on BAROCAP-anturin juotoslaadun ja toiminnan todentaminen. Testivaiheen on tarkoitus lean-periaatteen nojalla estää viallisen tuotteen pääsy tuotantoputkessa eteenpäin seuraavaan tuotannon työvaiheeseen, lämpötilamäärittelyyn, turhaan ja aiheuta siten ylimääräistä työtä ja pidennä muiden valmistettavien tuoteyksilöiden tuotantoputken läpimenoaikaa.

Koska BAROCAP-anturi on toimintaperiaatteeltaan kapasitiivinen anturi, saa testattavan paineanturin juotoksen toiminnan testattua lukemalla anturin antaman kapasitanssiluvun. Moduulilta saatua kapasitanssiarvoa verrataan kahteen eri raja-arvoon (ks. luku 4.4), jotka ovat suhteutettuna testausympäristössä vallitsevaa painetta vastaavaan kapasitanssiarvoon. Taulukosta 1 nähdään, että Suomessa kautta aikain kirjattujen ilmanpaineenmittauksen havainnot sijoittuvat 940 hPa:n ja 1066 hPa:n välille. Testausympäristön määriteltynä painearvona käytetään 900–1100 hPa:n painealuetta. Vaikka keskimäärin ilmanpaine merenpinnan korkeudella on noin 1013 hPa, tarkempaan painearvon kritisointiin ei ole tarvetta, sillä anturia ei ole vielä tässä tuotantovaiheessa viritetty vastaamaan tarkkaa mittalukemaa.

Taulukko 1. Suomessa tehdyt ilmanpaineen havainnot merenpinnantasossa [17.]

minimi	maksimi	keskim.	yksikkö
940	1066	1013	hPa

Jos anturilta luettu painearvo ylittää suppeammat raja-arvot, antaa testauskäyttöliittymä testausvaiheen tulokseksi ”Fail”. Jos tulos on ”Fail”, on oletettavaa, että BAROCAP-anturi on ehjä, mutta anturin juotoksessa anturin ja piirilevyn välillä saattaa olla heikko juotos. Jos taas anturilta luettu painearvo ylittää laajemmat raja-arvot, antaa

testauskäyttöliittymä tulokseksi ”Bogus”. Tuloksen ollessa ”Bogus” on joko anturin juotos unohtunut kokonaan, tai anturi ei saa yhteyttä moduulin piirilevyyn. Vika saattaa olla myös anturin virheellisessä toiminnassa tai anturi on vikaantunut kuljetuksessa jo ennen juotosta tai mahdollisesti juotoksen aikana.

4.8 Testiohjelmiston validointi ja validoinnin dokumentointi

Virallinen ohjelmistovalidointi toteutettiin 24-osaisena testien summana ja dokumentoitiin vaihe vaiheelta. Validointidokumentti pitää sisällään validoinnissa olevan ohjelman tiedot, yhteenvedon validoinnista, validointisuunnitelman, ja validoinnin tulosten ja toteutuksen tarkastelun vaiheittain eriteltynä ja selitettynä. Vaisalan virallinen dokumentointikieli on englanti, jonka vuoksi osat tässä työssä esitetyistä validoinnin otteista ovat englanniksi.

4.8.1 Validoidun testerin yhteenveto

Yhteenvedon alussa on lyhyt kuvaus, mitä validoitava testeri on suunniteltu tekemään.

Suurimmassa osassa ohjelmistokehitystä harjoittavissa yrityksissä, kuten Vaisalassa, on käytössä jokin versionhallintaohjelmisto ohjelmakoodien ylläpitämisen helpottamiseksi. Validointidokumentissa on tultava selkeästi ilmi, mikä ohjelmaversio on ollut kyseessä validoinnin hetkellä ja mistä versiohallinnan hakemistopolusta viimeisin validoitu ohjelmaversio löytyy. Yhteenvedon loppuun kasataan lista validoinnissa tehdyistä validointitoiminpiteistä, validoinnin tulos, päivämäärä ja validoinnin tekijä. Vaisalassa tekijä ilmoitetaan henkilökohtaisella nimimerkillä, initiaaleilla. (ks. Liite 1).

4.8.2 Validointisuunnitelma

Validointisuunnitelmassa kerrotaan lyhyesti jokaisen validointivaiheen tarkasteltavan osan nimike ja se mitä siitä aiotaan validoida. Painemoduulien esitestausaseman validioinnissa tehtiin seuraavat validointitestit:

1. Make of executable: Saako ohjelmakoodin käännettyä ajettavaan muotoon?

2. Latest available source coded from valid location: Löytyykö ohjelmakoodin viimeisin käyttöversio siille osoitetusta hakemistosta?

3. NI TestStand Analyze: Löytääkö National Instruments TestStand-ohjelman oma koodianalyysi-ominaisuus ohjelmakoodista virheitä?

4. Settings, Serial number: Toimiiko sarjanumeron syöttö laitteelle?

5. Settings, Sensor batch number: Toimiiko anturin eränumeron syöttö laitteelle?

6. Settings, Pressure range: Tuleeko laitteelle oikea painemoduulin paineenmittausalue?

7. Settings, Specification file(s): Ovatko laitteiden testirajamäärittelyt oikeassa ohjelmapolussa ja ajantasaiset?

8. Measurements, Measurement accuracy settings of measurement equipment: Toimivatko testerin oheislaitteet oikealla tarkkuudella?

9. Measurements, Measurement results measured correctly: Mittaako testeri mitattavat suureet oikein?

10. Measurements, Instrument calibration check: Tarkastaako testeri käynnistyessään oheislaitteiden kalibrointipäivämäärän oikein ja ilmoittaako ohjelma, kun kalibrointi on vanhentunut tai vanhentumassa?

11. Result file, Product type: Tuleeko testitulostiedostoon oikean tuotteen nimi?

12. Result file, Serial number: Tuleeko testitulostiedostoon oikea painemoduulin sarjanumero?

13. Result file, Sensor batch: Tuleeko testitulostiedostoon oikea anturin eränumero?

14. Result file, CB number: Tuleeko testitulostiedostoon oikea piirilevyn eränumero?

15. Result file, Pressure range: Tuleeko testitulostiedostoon oikea painemoduulin paineenmittausalue?

16. Result file, Measurements results: Tuleeko testitulostiedostoon oikeat mittaustulokset?

17. Result file, Measurements units: Tuleeko testitulostiedostoon oikeat mittatulosyksiköt?

18. Label, Product type: Onko laitetarrassa oikean tuotteen nimi?

19. Label, Pressure range: Onko laitetarrassa oikea painemoduulin paineenmittausalue?

20. Label, Serial number: Onko laitetarrassa oikea painemoduulin sarjanumero?

21. Database, Measurements results: Tuleeko jäljitettävyystietokantaan oikea testitulostiedosto?

22. Database, Settings read from UUTs and end settings of UUT: Ovatko testatulle painemoduulille syötetyt tiedot oikein ja luettavissa moduulilta?

23. Initialization settings and end settings of station: Näyttääkö testauskäyttöliittymä väliaikatiedot testin etenemisestä oikein ja asettuuko testeri testin jälkeen alkutilaan seuraavaa testiä varten?

24. Special cases, error handling: Virheenkäsittelyn toiminnan tarkastus.

Testit 1 ja 3 ovat hyviä esimerkkejä staattisesta testauksesta, jossa testaaja hyödyntää jotain testaussovellusta analysoimaan oman ohjelmistokoodin toimintaa. Kaikki loput 22 vaihetta ovat dynaamisia testejä, joita ei voi tai kannata antaa automatiikan testattavaksi testin luonteen vuoksi. Tämä dynaamisten ja staattisten testien suhdeluku antaa hyvän kuvan siitä, että vaikka erilaisten koodianalysaattoreiden käyttäminen olisikin helppoa ja hyödyllistä, eivät ne yksinomaan ilman dynaamista testaamista riitä. Staattisten testien tuloksiin nojaaminen ei ole lähellekään riittävää kokonaisvaltaisen testauksen kannalta.

4.8.3 Validoinnin toteutus

Validoinnin toteutusosiossa käydään läpi askel kerrallaan jokainen validointisuunnitelmassa esitetty validointivaihe. Jokainen vaihe pyritään todentamaan mahdollisimman tarkasti huomioiden kaikki mahdolliset muuttujat ja lopulliseen tulokseen vaikuttavat tekijät.

1. Make of executable: Vaihe todennettiin Microsoft Visual Studion sisäänrakennetulla kääntäjällä. Ohjelmakoodin kääntäminen suoritettavaksi exe-tiedostoksi onnistui ongelmitta. Yhteenvetona koodin kääntämisestä saatu tulos tallennettiin sellaisenaan validointidokumenttiin seuraavasti.

```
Compile complete -- 0 errors, 2 warnings
  VTX.Main -> C:\Work\BARO-1 PTB210 Pretest\VTX.Main\Release\VTX.Main.exe
========== Build: 13 succeeded or up-to-date, 0 failed, 0 skipped ==========
```

Analysaattorin löytämät kaksi varoitusta liittyivät käyttöliittymään loppukäyttäjältä tarkoituksella piilotettujen ominaisuuksien näyttämättä jättämiseen, eivätkä siksi aiheuttaneet mitään haittaa toiminnan kannalta.

2. Latest available source coded from valid location: Testerin lähdekoodin validoitu versio löytyy Vaisalan versionhallinnasta revisiolla 16105. Testerin käyttöliittymän versio on versioitu revisiolla 16108. Validoinnissa varmistettiin, että molemmat versiot ovat viimeisimmät käytössä olevat revisiot ja ne löytyivät versionhallinnasta niille osoitetulla numeroilla.

3. NI TestStand Analyze: National Instrumentsin oma koodianalyysiohjelma käy läpi koko testisekvenssin ja tarkistaa, löytyykö lähdekoodista ristiriitoja, ylimääräisiä tai puuttuvia muuttujia. Analysaattori varmistaa myös, että kaikki testivaiheet toimivat koodin puolesta, eikä mitään testivaiheita ole hypätty yli tai pakotettu antamaan ennaltamäärättyjä testituloksia. Analysaattorilla saadaan helposti pois suljettua käyttäjän inhimilliset virheet, kuten koodiin unohtuneet ylimääräiset merkinnät ja muistiinpanot.

4. Settings, Serial number: Sarjanumeron syöttö toteutetaan siten, ettei käyttäjä voi vahingossa antaa testattavalle laitteelle sarjanumeroa, joka ei ole Vaisalan sarjanumerointiperiaatteen mukainen. Koska sarjanumerokenttä testerin

käyttöliittymässä on vapaa tekstikenttä, voi operaattori teoriassa antaa laitteen sarjanumeroksi minkä tahansa merkkijonon. Siksi ohjelmakoodiin rakennetaan kritisointi, joka määrittelee sarjanumeron pituuden ja sen onko kukin sarjanumeron merkki numeerinen vai aakkosellinen. Koska Vaisalan sarjanumerot alkavat aina kirjaimella, tulee testerin estää testin aloitus, jos sarjanumero on merkitty alkamaan numerolla. Seuraavien seitsemän merkin on oltava numeerisia, jotta testin voi käynnistää. Sarjanumeron pituuden on myös oltava 8 merkkiä, jotta testeri hyväksyy sen. Kritisoinnin toimivuus validoidaan jokaiselle tuotetyypille erikseen seuraavan taulukon mukaisesti.

Taulukko 2. Testerin sarjanumerokritisoinnin validointitaulukko

BARO-1	Allow	Status	Verified Date
"" (empty)	No	Pass	2014-04-02
X123456	No	Pass	2014-04-02
X1234567	Yes	Pass	2014-04-02
X12345678	No	Pass	2014-04-02
12345678	No	Pass	2014-04-02
X123456X	No	Pass	2014-04-02
BARO-1A	Allow	Status	Verified Date
"" (empty)	No	Pass	2014-04-02
X123456	No	Pass	2014-04-02
X1234567	Yes	Pass	2014-04-02
X12345678	No	Pass	2014-04-02
12345678	No	Pass	2014-04-02
X123456X	No	Pass	2014-04-02
PTB210	Allow	Status	Verified Date
"" (empty)	No	Pass	2014-04-02
X123456	No	Pass	2014-04-02
X1234567	Yes	Pass	2014-04-02
X12345678	No	Pass	2014-04-02
12345678	No	Pass	2014-04-02
X123456X	No	Pass	2014-04-02

Taulukko 2:n ensimmäisessä sarakkeessa näkyy validoinnissa syötetty sarjanumero. Toisessa sarakkeessa ilmoitetaan, saako testeri hyväksyä syötetyn sarjanumeron. Kolmas ja neljäs sarake ilmoittaa, onnistuiko validointi ja milloin validointi on suoritettu.

5. Settings, Sensor batch number: Testerille syötetty anturin eränumero kritisoidaan samalla periaatteella kuin sarjanumerot. Eränumeron kritisoinnissa varmistetaan, että merkkejä on neljä kappaletta: ensimmäinen merkki on kirjain ja kolme seuraavaa numeerisia (ks. taulukko 3).

Taulukko 3. Testerin anturieränumeron validointitaulukko

BARO-1	Allow	Status	Verified Date
"" (empty)	No	Pass	2014-04-02
X12	No	Pass	2014-04-02
X123	Yes	Pass	2014-04-02
X1234	No	Pass	2014-04-02
1234	No	Pass	2014-04-02
X12X	No	Pass	2014-04-02
BARO-1A	Allow	Status	Verified Date
"" (empty)	No	Pass	2014-04-02
X12	No	Pass	2014-04-02
X123	Yes	Pass	2014-04-02
X1234	No	Pass	2014-04-02
1234	No	Pass	2014-04-02
X12X	No	Pass	2014-04-02
PTB210	Allow	Status	Verified Date
"" (empty)	No	Pass	2014-04-02
X12	No	Pass	2014-04-02
X123	Yes	Pass	2014-04-02
X1234	No	Pass	2014-04-02
1234	No	Pass	2014-04-02
X12X	No	Pass	2014-04-02

6. Settings, Pressure range: Painealueen valinta on testerissä tehty monivalintana, joten väärässä muodossa olevaa painealuetta ei pysty syöttämään. Operaattori saattaa kuitenkin tehdä virheen ja valita väärän painealueen. Virheen mahdollisuus on kuitenkin pieni, sillä testeri jättää edellisen testiajon valinnan oletukseksi seuraavaa testiajoa varten. Jos operaattori epähuomioissaan jättää valinnaksi väärän painealueen, tulee virhe esille viimeistään tarrojen tulostuksen yhteydessä visuaalisessa tarkastuksessa.

7. Settings, Specification file(s): Tämä vaihe suoritetaan varmistamalla, onko testerin käyttämän testirajamäärittelyn ID-numero sama, kuin Vaisalan testirajatietokannan viimeisimmän määrittelytiedoston ID. Sama tarkistus tehdään erikseen jokaisen laitteen kohdalla.

8. Measurements, Measurement accuracy settings of measurement equipment: Tässä vaiheessa luetaan testerin ulkoiselta virtalähteeltä sen ilmoittama käyttöjännite ja virrankulutus testin aikana. Näyttölukemaa verrataan testerin saamaan jännitteeseen ja luettuun virrankulutukseen (ks. taulukko 4).

Taulukko 4. Testerin virtalähteen näyttölukemien todentaminen

	DC Power	Read from test software	Status	Verified Date
BARO-1				
Voltage	9.997 V	9.997 V	Pass	2014-04-02
Current	2 mA	1.89 mA	Pass	2014-04-02
BARO-1A				
Voltage	9.998	9.998 V	Pass	2014-04-02
Current	2 mA	2.06 mA	Pass	2014-04-02
PTB210				
Voltage	9.999 V	9.999 V	Pass	2014-04-02
Current	7 mA	6.53 mA	Pass	2014-04-02

9. Measurements, Measurement results measured correctly: Testitiedostosta tarkistetaan, että testerin kirjaamat mittayksiköt ovat oikein. Varmistetaan, että jännitteet ovat voltteja, virrat milliampeereja, painelukemat hehtoPascaleja ja lämpötilat Celcius-asteita.

10. Measurements, Instrument calibration check: Varmistetaan testerin asetusmäärittelytiedostosta, että kalibroitavien mittalaitteiden ID-numerot ovat oikeat. Käynnistetään testi ja annetaan kalibrointipäivämäärän tarkistuskomento. Vastauksena tulee kuukausi ja vuosi, jolloin mittalaite on kalibroitava. Jokaiseen Vaisalassa käytössä olevan kalibroitavaan oheislaitteeseen on kiinnitetty tarra, jossa lukee seuraava kalibrointiaika. Varmistetaan, että tarran ja testerin antamat ajankohdat ovat samat.

11. – 17. Result file: Varmistetaan, että seuraavat tiedot vastaavat testitulostiedostossa juuri testattua tuotetta: tuotteen nimi, sarjanumero, anturin eränumero, piirilevyn eränumero, painealue, mittaustulokset ja mittayksiköt.

18. – 20. Labels: Testin päätyttyä testeri tulostaa kaksi laitetarraa. Alumiinisen tuotetarran ja paperisen painealuetarran. Validointivaiheessa varmistetaan, että alumiiniselle tuotetarralle tulostuu oikean tuotteen nimi, sarjanumero ja painealue. Seuraavaksi varmistetaan, että paperiselle tarralle tulostuu oikea painealue.

21. Database, Measurements results: Testeri kirjoittaa testitiedoston paikallisena tiedostona testeritietokoneelle, josta testeri siirtää automaattisesti aika ajoin kaikki testitiedoston sisältämät tiedot ja mittaustulokset jäljitettävyystietokantaan myöhempää tarkastelua varten. Aluksi varmistetaan tietokannasta, että testeri varmasti lähettää testitiedoston tietokantaan. Kun tiedot ovat siirtyneet, verrataan tietokannan tietoja alkuperäiseen testitiedostoon ja varmistetaan, etteivät tulokset ole muuttuneet tiedonsiirtovaiheessa.

22. Settings read from UUTs and end settings of UUT: Kun moduuli on testattu ja sille on syötetty tarvittavat tiedot, kytketään moduuli vielä takaisin testeriin ja otetaan laitteelle kommunikointiyhteys. Laitteelta varmistetaan, että kaikki testauksen aikana syötetyt tiedot löytyvät myös laitteelta. Laitteelta luettavissa olevat tiedot tulee olla sarjanumero, piirilevyn eränumero ja laitteen painealue.

23. Initialization settings and end settings of station: Varmistetaan, että testin alkuvaiheessa testin etenemistä osoittava aikaindikaattori näyttää 0 % lukemaa, eikä edistymispalkki ole liikkunut 0-tilasta. Testiautomatiikka suoritetaan askel askeleelta ja varmistetaan, että aikaindikaattori ja edistymispalkki nousee tasaisesti kymmenen prosentin välein kohti sataa prosenttia testin loppuun asti. Testin loputtua tulee aikaindikaattorin näyttää 100 % ja edistymispalkin olla täysin edennyt loppuun saakka. Kun testerikäyttöliittymästä painetaan ”New”-painiketta, nollautuvat molemmat indikaattorit ja alkavat taas alusta uuden testin käynnistyessä.

24. Special cases, error handling: Testivaiheessa pakotetaan testattava laite hylkääntymään vuorollaan jokaisessa testivaiheessa. Kun testeri toteaa, että testivaihe on hylätty, antaa testeri siitä ilmoituksen käyttäjälle ja suorittaa hallitun testin lopettamisen. Testin hallittu sammutus tapahtuu erillisellä alisekvenssillä ”onError” (ks.

taulukko 5). Kun testeri menee ”onError”-tilaan, näyttää testeri raportin, josta ilmenee hylkäyksen syy. Hylkääntyneet laitteet toimitetaan eteenpäin tutkittavaksi hylkääntymisraportin kanssa, jolloin selvitetään, romutetaanko vai korjataanko vikaantunut laite.

Taulukko 5. Virheenkäsittelyn validointitaulukko

BARO-1	Step	Status	Verified Date
Power fail	Goto onPowerFail	Pass	2014-04-02
CB comm. fail	Goto onError	Pass	2014-04-02
Sensor solder fail	Goto onError	Pass	2014-04-02
Temperature meas. fail	Goto onError	Pass	2014-04-02
Pressure range fail	Goto onError	Pass	2014-04-09
BARO-1A			
Power fail	Goto onPowerFail	Pass	2014-04-02
CB comm. fail	Goto onError	Pass	2014-04-02
Sensor solder fail	Goto onError	Pass	2014-04-02
Temperature meas. fail	Goto onError	Pass	2014-04-02
Pressure range fail	Goto onError	Pass	2014-04-09
PTB210			
Power fail	Goto onPowerFail	Pass	2014-04-02
CB comm. fail	Goto onError	Pass	2014-04-02
Sensor solder fail	Goto onError	Pass	2014-04-02
Pressure range fail	Goto onError	Pass	2014-04-09

4.9 Testiohjelmiston liittäminen testerimekaniikkaan

Testipenkkien oli määrä valmistua toukokuussa 2014 heti testisovelluksen valmistumisen jälkeen. Aikataulu ei kuitenkaan pitänyt mekaniikkasuunnittelijan osalta. Aikaa testipenkin suunnitteluun ja testipenkkien valmistuttamiseen kului huomattavasti alkuperäistä aikataulusuunnitelmaa enemmän. Kaikki tarvittavat testipenkin osat saatiin vasta loppuvuodesta 2014.

Testipenkkien materiaaliksi valittiin ESD-suojattu polyasetaali. ESD-suojaukseen on suhtauduttava testerin kannalta erityisellä vaativuudella Vaisalan tiukkojen ESD-standardien vuoksi.

Testerimekaniikan liittämisessä ja käyttöönotossa avustajana toimi vanhempi testaussuunnittelija (senior test engineer) Mika Mannonen. Testipenkin ja testeritietokoneen välinen kommunikointi tehtiin tammikuussa 2015 ja testerit kytkettiin lopulliseen sijoituspaikkaansa painemoduulituotannon kokoonpanosoluun.

Kytkentä tehtiin suunnittelemani testauksen vaatimusmäärittelyn mukaisesti (ks. luku 4.3). Käyttöönotto testimekaniikan kanssa onnistui ongelmitta ja projekti oli valmiina seuraavaan työvaiheeseen, ensimmäisen tuotantosarjan testiajoon. Testiajo, jota kutsutaan 0-sarjaksi, on testerin toiminnan testaamista osana tuotantoa. 0-sarjaa varten valmistettiin 20 kappaleen erä jokaista testattavaa painemoduulia, joiden kulkua seurataan koko tuotantoputken läpi alusta loppuun asti. Jos kaikki tuotantovaiheet ovat menneet tuotantoputken läpi odotetulla tavalla ja hyväksytysti, annetaan hyväksyntä testerin käyttöönottoon osaksi tuotantoa.

BARO-1 ja BARO-1A moduulien 0-sarja käytiin läpi kokonaisuudessaan helmikuussa ja PTB210SUB 0-sarja maaliskuussa 2015. Molemmat 0-sarjat hyväksyttiin ja testeri liitettiin osaksi tuotantoa.

4.10 Työssä laaditut tekniset dokumentit

Koko projektin dokumentointiin kuuluivat tässä työssä esitellyn vaatimusmäärittelyn ja validointidokumentin lisäksi tuotannon käyttöohje, huolto-ohje testausylläpidolle, testerin osaluettelo tilausnumeroineen ja testipenkkien kaapelointikuvat.

Dokumentit ovat vain Vaisalan sisäiseen käyttöön eivätkä siksi ole julkaistavissa.

4.11 Tuotannon aloittaminen ja testerin toiminta tuotantolinjalla

Testeri on ollut tätä kirjoittaessa BARO-1- ja BARO-1A-moduulien osalta käytössä tuotannossa 10 viikkoa. Testerissä on testattu yhteensä 2126 kappaletta BARO-1- ja 862 kappaletta BARO-1A-laitteita. Näiden testattujen moduulien joukosta on löytynyt yhteensä 98 kappaletta testerin hylkäämiä laitteita.

PTB210SUB-laitteita on testattu tuotannossa 6 viikkoa. Tänä aikana testattuja laitteita on ollut 669 kappaletta, joista hylättyjä on ollut 57 laitetta.

Yleisin hylkääntymisen syy kaikilla laitteilla on toimimaton paineanturi.

Tähän mennessä testerin saanto BARO-1- ja BARO-1A-laitteiden osalta on 96,72 % ja PTB210SUB:n osalta 91,48 %.

Huhtikuussa 2015 teetettiin pienimuotoinen sanallinen haastattelu testerin käyttäjille, jolloin kysyttiin, mitä mielipiteitä uuden testivaiheen liittäminen tuotantoon on herättänyt tai kokevatko he testerin käytön hankalaksi tai muuta työtä haittaavaksi kokoonpanovaiheeksi. Yleinen vastaanotto uudelle testerille on yllättänyt positiivisesti eikä kenelläkään ollut kielteistä sanottavaa testerin käytöstä. Erityisesti tuotetarrojen tulostaminen testauksen yhteydessä on koettu positiivisena asiana, sillä operaattorin ei enää tarvitse erikseen käydä tehtaan yleisellä tarratulostimella tekemässä itse tarroja. Inhimillisten virheiden mahdollisuuden kokoonpanossa koetaan vähentyneen, eikä kukaan maininnut uuden testausvaiheen pidentävän yksittäisten moduulien tuotantoaikaa. Moni kokee myös työnsä merkityksekkäämmäksi, kun turhalta työtä vältytään viallisten tuotteiden kohdalla tuotantoputken seuraavissa vaiheissa.

5 Pohdinta

Insinöörityön tavoitteena oli luoda kokonaan uusi testivaihe Vaisalan paineenmittauslaitteiden valmistusprosessiin. Testivaiheen tarkoituksena oli parantaa paineenmittauslaitteiden tuotannon saantoa ja vähentää hukkatyön määrää. Lisäksi testerin oli tarkoitus helpottaa tuotantoprosessia ja parantaa Vaisalan painelaitteiden sarjanumeroseurantaa ja jäljitettävyyttä.

Vaikka minulla ei ollut entuudestaan lainkaan kokemusta testaussuunnittelusta, suoriuduin mielestäni minulle annetusta tehtävästä kiitettävästi. Testausohjelmiston suunnittelu ja toteutus eteni ilman suurempia ongelmia, ja lopputuloksena saatiin käytännöllinen testeri, joka on saanut Vaisalan tuotantolinjalta ja testauskehitykseltä kiitettävää palautetta. Vaikka sainkin oman osani työstä tehtyä Vaisalalle projektin alkuvaiheessa määritellyn aikataulun mukaisesti, jouduttiin alkuperäisestä aikataulusta poikkeamaan reilusti testerin mekaanisen toteutuksen vuoksi.

Testeri on käyttöönoton jälkeen täyttänyt kaikki vaatimuksensa ja toimii tänä päivänä merkittävänä osana kaikkien Vaisalan valmistamien painelaitteiden laadunvalvontaa.

Työ oli minulle hyvin positiivinen ja opettavainen kokemus ja opin kuluneen vuoden aikana paljon automaattisesta testaamisesta, testaussuunnittelusta, suuren mittakaavan tuotannosta ja työskentelystä osana suurta organisaatiota ja projektityötä.

Lähteet

1 Yleistä Leanista. verkkodokumentti. Sig Sigma <http://www.sixsigma.fi/fi/lean/yleinen/> Luettu 15.3.2015.

2 Myers, Glenford. 2012. *The Art of Software Testing, 3rd Edition*. John Wiley & Sons, Inc.

3 Kaner, Cem, Falk, Jack and Nguyen, Hung Quoc. 1999. Testing Computer Software, 2nd Ed., John Wiley & Sons, Inc.

4 Sivistyssanakirja ja synonyymisanakirja. verkkodokumentti <http://www.suomisanakirja.fi/> Luettu 6.4.2015.

5 Metsämuuronen, Jari. 2002. Mittarin rakentaminen ja testiteorian perusteet 2. painos. Helsinki: International Methelp Ky.

6 Gage Repeatability and Reproducibility (Gage R&R) in an Excel Spreadsheet, verkkodokumentti. Muelaner Jody. <http://www.muelaner.com/quality-assurance/gage-r-and-r-excel/>. Luettu 26.4.2015.

7 Kasurinen, Jussi Pekka. 2013. Ohjelmistotestauksen käsikirja. Jyväskylä: Docendo.

8 Vesiputousmalli. verkkodokumentti. <http://fi.wikipedia.org/wiki/Tiedosto:Vesiputousmalli.jpg>. Luettu 26.4.2015.

9 The Software Experts, V-model. verkkodokumentti. <http://www.the-software-experts.com/e_dta-sw-process-model-V.php>. Luettu 26.4.2015.

10 Testaussuunnitelma. verkkodokumentti. <http://www.soberit.hut.fi/T-76.115/03-04/palautukset/groups/PPT/i1/testing/testplan.htm>. Luettu 27.4.2015.

11 Kroll, Per, Kruchten, Philippe. 2003. The Rational Unified Process Made Easy: A Practitioner's Guide to the RUP. Boston. Addison-Wesley.

12 Iteratiivisen mallin vaiheet. verkkodokumentti <http://fi.wikipedia.org/wiki/Tiedosto:IteratiivisenMallinVaiheet.JPG>. Luettu 26.4.2015.

13 Vaisala BAROCAP® -anturi paineen mittaamiseen. 2012. verkkodokumentti. <http://www.vaisala.fi/Vaisala%20Documents/Technology%20Descriptions/CEN-TIA-BAROCAP-Technology-description-B210845FI-B.pdf>. Luettu 10.11.2014.

14 Vaisala Products. Vaisala internal photo bank.

15 BARO-1/BARO-1A/PTB210 PreTest Production Test Specification. 2014. Eskola. Vaisala internal documentation.

16 What is TestStand. verkkodokumentti <http://www.ni.com/teststand/whatis/>. Luettu 10.11.2014.

17 Ilmanpaine. verkkodokumentti <http://ilmatieteenlaitos.fi/ilmanpaine>. luettu 12.4.2015.

Validoidun testerin yhteenveto

Test name	Status	Date	Performer
Make of executable	PASS	2014-04-09	SESK
Latest available source codes from valid location	PASS	2014-04-09	SESK
NI TestStand Analyze	PASS	2014-04-09	SESK
Settings (user selections), User Interface (input form etc.)			
Serial Number	PASS	2014-04-02	SESK
Sensor batch number	PASS	2014-04-02	SESK
Pressure range	PASS	2014-04-09	SESK
Specification file(s)	PASS	2014-04-02	SESK
Measurements and calculations			
Measurement accuracy settings of measurement equipment	PASS	2014-04-02	SESK
Measurement results measured correctly	PASS	2014-04-02	SESK
Instrument calibration check	PASS	2014-04-02	SESK
Result File			
Product type	PASS	2014-04-09	SESK
Sensor batch	PASS	2014-04-09	SESK
CB number	PASS	2014-04-09	SESK
Pressure range	PASS	2014-04-09	SESK
Measurements results	PASS	2014-04-09	SESK
Measurements units	PASS	2014-04-09	SESK
Label printing			
Correct label data	PASS	2015-01-22	SESK
Test results database			
Measurement results	PASS	2014-04-10	SESK
Settings read from UUTs and end settings of UUT	PASS	2014-04-10	SESK
Initialization settings and end settings of "station"	PASS	2014-04-02	SESK
Special cases, error handling	PASS	2014-04-09	SESK