jamk.fi

Automaation itsetestausohjelma

Fidelix-keskusyksiköt

Julius Koivuvirta

Opinnäytetyö
Maaliskuu 2018
Tekniikan ja liikenteen ala
Insinööri (AMK), Automaatiotekniikan tutkinto-ohjelma

Jyväskylän ammattikorkeakoulu
JAMK University of Applied Sciences

jamk.fi

Kuvailulehti

Tekijä(t)	Julkaisun laji	Päivämäärä
Koivuvirta, Julius	Opinnäytetyö, AMK	13.03.2018
	Sivumäärä 51	Julkaisun kieli Suomi
		Verkkojulkaisulupa myönnetty: x

Työn nimi
Automaation itsetestausohjelma
Fidelix-keskusyksiköt

Tutkinto-ohjelma
Automaatio- ja sähkötekniikka

Työn ohjaaja(t)
Ari Kuisma, Teppo Flyktman

Toimeksiantaja(t)
LVI-Elektro Oy

Tiivistelmä

Opinnäytetyön tavoitteina oli määritellä ja ohjelmoida kiinteistöautomaation itsetestausohjelma Fidelix Fx-20xx -säädinperheelle, sekä Fidelix Spider -automaatiosäätimille. Työn tuloksilla haluttiin saavuttaa aikasäästöä automaatiojärjestelmien testauksessa ja vikatilanteissa, sekä helpottaa säätimiin- ja järjestelmiin liittyvää vianetsintää. Itsetestausohjelmalla voitaisiin esimerkiksi testata kiinteistöautomaatiojärjestelmän toiminta automatisoidusti käyttöönottotestien aikana, tai huoltotoimenpiteen yhteydessä.

Työn alussa määriteltiin itsetestausohjelman toiminnot ja toteutusmahdollisuudet. Työ päädyttiin tekemään mahdollisimman helposti muokattavaksi eri projekteihin, jotta sen tulokset hyödyttäisivät yritystä myös jatkossa. Ohjelman toteutus koostui enimmäkseen kolmesta osiosta: logiikkaohjelmasta, käyttöliittymästä ja ohjelmakoodin luontityökalusta.

Työn tuloksina saatiin toimiva automaation itsetestausohjelma, sen luomiseen käytettävä koodinluontityökalu ja käyttöliittymä. Ohjelma tuo säätimen käyttöliittymään säätimen omia diagnostiikkatietoja ja testauksen tuloksena saatavia virhetietoja. Lisäksi käyttöliittymästä saa avattua yksityiskohtaisemman raporttitekstin, josta nähdään automaatiojärjestelmään liitettyjen pisteiden tilat. Työn tulokset testattiin Jyväskylän Ammattikorkeakoulun opetuskäyttöön suunnitellulla kiinteistöautomaatiolaitteistolla.

Työn tulokset vastasivat toimeksiantajan tarpeisiin ja niihin oltiin tyytyväisiä. Työn tuoma aikasäästön vaikutus kasvaa testattavan järjestelmän koon ja tarvittavien testauskertojen myötä. Automaation itsetestausohjelmaa tullaan käyttämään jatkossa erityisesti automaatiojärjestelmien käyttöönottotesteissä, sekä vikaseurannassa. Ohjelman jatkokehitysehdotuksena siitä voitaisiin tehdä eri versioita, jotta se soveltuisi vielä paremmin erityyppisiin järjestelmiin.

Avainsanat
Kiinteistöautomaatio, automaatiosäädin, logiikka-ohjelmointi, testaus, Fidelix

Muut tiedot

jamk.fi

Description

Author(s)	Type of publication	Date
Koivuvirta, Julius	Bachelor's thesis	13.03.2018
		Language of publication: Finnish
	Number of pages 51	Permission for web publication: x

Title of publication
Self-test program of automation
Fidelix controllers

Degree programme
Automation- and electrical engineering

Supervisor(s)
Ari Kuisma, Teppo Flyktman

Assigned by
LVI-Elektro Oy

Abstract

The goals of the thesis were to define and program a self-test program for automation, to be used with the Fidelix Fx-20xx –product family and Fidelix Spider -automation controllers. The results of the work were hoped to bring timesavings regarding the testing and problem solving of building automation systems and to ease the troubleshooting of said systems. The self-test program could be used e.g. to test the functionality of a building automation system automatically, during the system's commissioning or maintenance.

First, the functions and implementation possibilities of the program were defined. The program was opted to be as easy to modify to different projects as possible, thus its results would also benefit the client in the future. The result program included three major parts: a logic program, a user interface and a code-generation tool.

The end results were a functional self-test program for automation, the code-generation tool used for creating program code and a user interface. The program shows the controller's self-diagnostics and the results of the program's error tests in its user interface. In addition, a more detailed text-report of the automation system's condition can be opened in the user interface. The results were tested using the building automation equipment at JAMK University Of Applied Sciences, which is designed for educational use.

The results corresponded to the client's needs and were gratifying. The timesavings provided by the program were estimated to increase with the target-system's size and the required amount of tests. The self-test program of automation will be used in the future especially for functional testing and fault management of automation systems. As further development suggestions, there could be more versions of the program better suited for different kinds of systems.

Keywords/tags
Building automation, automation controller, programming, testing, reporting, Fidelix

Miscellaneous

Sisältö

Kuviot

Taulukot

1 Johdanto

1.1 Automaatiojärjestelmien itsediagnostiikka

Nykypäivän automaatiotekniikka mahdollistaa prosessien ja tilojen valvonnan suurella tarkkuudella ja varmuudella. Toisaalta jos itse automaatiojärjestelmän laitteita ja sen tekemiä toimenpiteitä ei ole testattu ja todettu toimiviksi, järjestelmään jääneet virheet voivat vaikuttaa merkittävästi koko prosessin toimintaan. Esimerkkinä tästä voi olla tilanne, jossa prosessia valvotaan erilaisilla mittauksilla, mutta järjestelmän vääränlainen toiminta hukkaa energiaa, tai rasittaa laitteistoa.

Automaatiojärjestelmän huolellinen toimivuuden testaus on siis hyödyllistä, mutta sen toteuttaminen voi myös viedä paljon aikaa ja olla työlästä. Tämän vuoksi moniin automaatiojärjestelmiin on kehitelty itsediagnostiikkaa, joka voi tarkistaa järjestelmän kommunikaation, koodisyntaksin, moduulit ym. asioita virheiden varalta. Mahdollisista virheistä voidaan tiedottaa käyttäjää, jolloin vikoja ei tarvitse itse etsiä ja todeta järjestelmästä.

Yleensä sisäänrakennettu itsediagnostiikka pystyy tarkastelemaan järjestelmän tilaa vain hyvin suppeasti. Säädin voi tarkastaa esimerkiksi oman toimintakuntonsa ja laitteiden välisen kommunikaation toimivuuden, mutta ei osaa tehdä enempää tulkintoja järjestelmän tilasta. Tämä voi aiheuttaa ongelmia esim. silloin kun kaikki järjestelmään kytketyt laitteet ja niiden välinen kommunikaatio toimii, mutta jossain toimielimessä on fyysinen vika, jokin laite on kytketty väärin tai jokin automaatiojärjestelmän säätö on viritetty huonosti. Erityisesti säätöihin liittyviä virheitä voi olla vaikea havaita perustarkistuksissa ja nopeasti silmäilemällä. Esimerkiksi jos lämmityssäätö on huonosti viritetty, se voi huojua tarpeettomasti prosessin käynnistystilanteessa.

1.2 Työn lähtökohdat

Opinnäytetyön toimeksiantajana toimi Lvi-Elektro Oy. Lvi-Elektro on Jyväskylässä sijaitseva yritys, joka erikoistuu kiinteistöautomaatioon, -energiansäästöön ja -etähallintaan. Yritys tekee kiinteistöautomaatiojärjestelmien urakointia ja tarjouskäsittelyä,

kiinteistöjen energiakartoituksia, talotekniikan ylläpitoa ja automaation etähallinta-ratkaisuja (Lvi-Elektro Oy, 2017).

Lvi-Elektro käyttää monissa projekteissaan Fidelix Oy:n automaatiojärjestelmiä. Automaatiojärjestelmien käyttöönotossa, testauksessa ja vikatilanteiden ratkaisussa voi kulua paljon aikaa. Tätä kentällä käytettyä aikaa haluttaisiin vähentää, jotta järjestelmän testaus, säätöjen viritys ja vianmääritys sujuisi mahdollisimman kustannustehokkaasti ja vaivattomasti.

Työn aihe on tarkemmin kuvailtuna Fidelix-automaatiosäätimien itsetestauksen määrittely ja ohjelmointi. Työssä määritellään asiat, jotka automaatiosäätimen voisi tarkistaa itsediagnostiikan ja erilaisten testien avulla omasta toiminnastaan ja ohjattavasta järjestelmästä, jotta niiden toimivuus voitaisiin todeta mahdollisimman laajasti. Nämä testaukset ohjelmoidaan ja testataan soveltuvilta osin. Työhön voi sisältyä esim. mahdollisten laitteisto-, kommunikaatio- ja säätövirheiden toteaminen ja niistä ilmoittaminen säätimen käyttäjälle.

Toimeksiantaja ehdotti kyseistä aihetta, koska koki sen olevan sen olevan molemmille työn osapuolille hyödyllisin vaihtoehtoina olleista aiheista. Aihe todettiin yhdessä hyväksi valinnaksi. Aihe katsottiin olevan myös sopivan rajattu ja tulosten arvioitiin hyödyttävän yritystä huomattavasti.

Aiheesta tekee kehittämisen arvoista se, että itsediagnostiikan avulla pystytään välttämään monia säätimen virhetilanteita, varsinkin uuden säätimen asennuksen tai järjestelmän ensikoestuksen yhteydessä, kun viasta tiedotetaan säätimen käyttäjälle. Kun järjestelmässä ilmenee vika, siitä voidaan antaa heti virheilmoitus käyttäjälle tai huoltohenkilölle ja vian etsimiseen kuluva työaika voi vähentyä huomattavasti. Testausohjelma auttaa peruskäyttäjää ymmärtämään mikä ei toimi järjestelmässä ja mistä vikaa kannattaisi lähteä etsimään.

Automaation itsetestauksen määrittelystä on tehty jonkin verran opinnäytetöitä, mutta ne eivät vastaa suoraan toimeksiantajan tarpeisiin. Muutama opinnäytetyö käsittelee Fidelix-järjestelmiä ja muiden säädinten testausta, joten niitä on käytetty apuna tässä työssä.

1.3 Tavoitteet

Opinnäytetyön tavoitteena on saada määriteltyä ja ohjelmoitua automaation itsetestausohjelma, jota voidaan käyttää Fidelix Fx-20xx -sarjan ja Fidelix Spider -säätimillä. Myös pelkkä automaatiosäätimen itsetestauksen määrittely olisi ollut mahdollinen tavoite, mutta se katsottiin melko suppeaksi aiheeksi opinnäytetyölle ja sen arvo toimeksiantajalle olisi ollut vähäinen.

Työ rajattiin siten, että itsetestauksen määrittelyn ja ohjelmoinnin tuloksena saataisiin toimiva säätimen itsetestausohjelma, jonka täytyy testata säätimen- ja sen hallitseman automaatiojärjestelmän toiminta soveltuvilta osin ja antaa käyttäjälle mahdollisimman kattavat tiedot testauksen tilasta ja sen tuloksista. Ohjelman testaamat asiat määritellään osana opinnäytetyöprosessia. Lisäksi ohjelmalta toivottiin säätöjen automaattista viritystoimintoa, joka virittäisi järjestelmän säätöjä järjestelmästä kerätyn tiedon perusteella.

Työn rajaus on tarkoituksellisesti hieman joustava, sillä tarkkaa tietoa järjestelmän itsetestausmahdollisuuksista ei ollut ja niiden selvittäminen oli osana työtä. Ohjelma oli tarkoitus testata myös käytännön projektilla tai -simulaatiolla, jotta työn tuloksia pystytään arvioimaan paremmin konkreettisella tasolla.

Työn tulosten on tarkoitus tuoda lisätietoa automaation itsetestauksen määrittelystä ja toteuttamisesta, helpottaa Fidelix-automaatiosäätimiin ja niiden ohjaamiin järjestelmiin liittyvää vianetsintää, sekä nopeuttaa automaatiojärjestelmien testaamista.

1.4 Raportin rakenne

Tässä raportissa tullaan ensin käymään läpi opinnäyteytön teoriaperustaa, eli kiinteistöautomaation keskeisimmät asiat ja Fidelix-järjestelmien perusteet.

Teoriapohjan jälkeen käydään läpi itse työn suunnitteluun ja toteutukseen liittyvät asiat ja esitellään työn tulokset. Raportin lopussa on työn tuloksiin ja laatuun liittyvää arviointia ja pohdintaa.

2 Aineisto ja menetelmät

2.1 Tiedonhankinta

Opinnäytetyön tiedonhankinta tehtiin pääosin internet-lähteiden kautta, mutta myös kirjallisia lähteitä ja opinnäytetöitä on käytetty työn tietoperustana. Työn suunnittelussa käytettiin myös hieman toimeksiantajan asiantuntijoiden tietämystä apuna.

Työn luonteesta johtuen sen tietoperusta on suhteellisen rajattu. Työn keskeisiä aiheita ovat mm. Fidelix-automaatiosäätimet, kiinteistöautomaatio, säätöjen viritys, itsediagnostiikka, automaation testaus ja PLC-ohjelmointi. Näistä aiheista käsitellään tässä raportissa vain tämän työn kannalta keskeisin teoria.

2.2 Työn luonne

Työn ongelmanratkaisumenetelmänä oli tavoitteiden, työharjoittelukokemuksen ja tiedonkeruun pohjalta tehtävä itsetestausohjelman määrittely ja ohjelmointi. Tutkimustyö jäi pienemmäksi osaksi opinnäytetyötä. Työn luonne vaati vain perusymmärryksen suuremmista aihekokonaisuuksista, mutta paljon ohjelmoinnin- ja Fidelix-järjestelmien tuntemusta. Nämä asiat selkeytyivät paremmin ainoastaan tekemällä ohjelmointityötä ja käyttämällä Fidelix -ohjelmia.

Kyseessä on laadullinen tutkimus- ja kehitystyö. Työn vaatimukset ja aihe eivät juurikaan hyödy määrällisen tutkimuksien tiedoista, eikä niitä voida soveltaa työn aiheeseen luontevasti. Vaikka kiinteistöautomaation tavoitteet ja menetelmät ovat hyvin samankaltaiset eri kohteissa, on melkein jokainen rakennus kuitenkin erilainen toteutukseltaan ja tiloiltaan. Näin ollen myös jokainen automaatiokokonaisuuden testaus vaatii eri määrän aikaa, resursseja ja toimenpiteitä. Työn sisältö oli myös suurimmaksi osaksi ohjelmointityötä, jota on haastava kehittää tilastomateriaalin avulla.

3 Kiinteistöautomaatio

3.1 Periaatteet

Kiinteistö-, eli rakennusautomaatio on automaatiotekniikan merkittävä osa-alue. Se on eritelty prosessiautomaatioon verrattavissa olevaksi omaksi ryhmäkseen, vaikka sen ominaisuudet ja toiminnot voivat olla hyvinkin samankaltaisia kuin prosessiautomaatiossa. (Pertti V. & Jukka-Matti M. 1999, 5)

Nykyaikaisen kiinteistöautomaation tavoite on yleensä rakennuksen monipuolinen automatisoitu hallinta. Tämä kokonaisuus voi pitää sisällään rakennuksen ilmanvaihdon-, lämpötilojen-, valaistuksen-, turvallisuuden- ja kulunvalvonnan hallinnan. Lisäksi kiinteistöautomaatiolla voidaan hoitaa myös muita rakennuskohtaisia tarpeita.

Kiinteistöautomaatiosta on monia hyötyjä sekä rakennuksen käyttäjälle, että kiinteistönomistajalle. Kiinteistöjen viihtyisyys paranee, kun automaatio säätää ilmanvaihdon ja lämmityksen sopivalle tasolle. Murtohälyttimet ja paloilmoittimet lisäävät rakennusten turvallisuutta. Automaation mahdollistamat energiasäästöt vähentävät rakennusten käyttökustannuksia. (Pertti V. & Jukka-Matti M. 1999). Kiinteistöautomaatio rahoitetaankin yleensä energiansäästöillä, automaation mahdollistamilla ennaltaehkäisevillä huoltotoimilla, työntekijöiden työtehon nousemisella ja asiakkaiden viihtyvyydellä. (Phil, Z. 2017)

Kiinteistöautomaation hierarkkisessa rakenteessa on yleensä kolme tasoa:

- Hallintotaso (paikallisvalvomot, kauko-/etävalvomot)
- Automaatiotaso (valvonta-alakeskukset, IO-moduulit)
- Kenttätaso (kenttälaitteet, itsenäiset säätimet)

Hallintotasolta valvotaan keskitetysti automaatiojärjestelmien toimintaa ja tehdään tarvittaessa säännöllisiä tilannekatsauksia. Valvomosta voidaan myös ohjata järjestelmää ja tehdä säätö- ja asetuskorjauksia. Automaatiotasolla toimii valvomon toiminnasta suurimmaksi osaksi riippumaton valvonta-alakeskuslaitteisto, joka koostuu automaatiosäätimestä ja sen tietokytkennöistä. Tämä laitteisto ohjaa kenttälaitteiden

toimintaa ja mittaa niiden tiloja. Alimpana hierarkiassa on kenttälaitteet, jotka mittaavat eri olosuhteita ja toteuttavat automaatiosäätimen ohjaamia tehtäviä erilaisin toimilaittein. (Pentti, H. ym. 2012, 93)

3.2 Mittaukset ja tilatiedot

Kiinteistöautomaatiossa yleisin mitattava kohde on lämpötila. Rakennuksen sisäilman ja patteriston oikea lämpötila lisää käyttäjien viihtyvyyttä ja terveyttä, sekä suojaa laitteita ja rakenteita. Osassa tiloista tarvitaan myös erilaisille koneille tai tuotteille niiden käyttöikää pidentävä ilmanlämpö.

Huoneilman laatuun vaikuttaa olennaisesti myös ilman hiilidioksidipitoisuus ja ilmankosteus. Näitä suureita tarkkaillaan huone- tai kanava-antureilla. Tilan hiilidioksidipitoisuutta voidaan vähentää tehostamalla ilmanvaihtoa. (Pertti V. & Jukka-Matti M. 1999, 42–43). Yleensä tila halutaan hieman alipaineiseksi, sillä ylipaine voi puskea huoneilman kosteutta talon rakenteisiin, lisäten kosteuden aiheuttamia rakenteellisia ja terveydellisiä haittatekijöitä.

Jotta ilmanvaihdon tasoa voidaan seurata ja ylläpitää tasaisena, täytyy ilmanvaihtokanavien ilmanpainetta, tai ilmamääriä mitata kanavapaine-anturilla. Nesteelle soveltuvia painemittareita käytetään vesikiertoisten verkostojen paineiden seurannassa. Merkittävä paine voi hajottaa venttiileitä ja toimilaitteita, kun taas liian vähäinen paine vähentää järjestelmän tehokkuutta ja voi vaikuttaa laitteiden ja koko järjestelmän toimintaan haitallisesti.

Mittausten lisäksi järjestelemästä kerätään tilatietoja, jotka ovat mittauksista poiketen porrastettuja tietoja, eli niillä on yleensä vain kaksi tilaa. Tällaisia tietoja ovat esimerkiksi hälytystiedot ja kytkimien asennot. Tilatiedot liittyvät yleensä enemmän ohjattaviin koneisiin ja laitteisiin, kuin mitattaviin olosuhteisiin. Olosuhteiden karkeassa mittauksessa voidaan tosin käyttää ajoittain kaksiasentoisia raja-kytkimiä ym. porrastettuja mittatietoja.

3.3 Ohjaukset ja säädöt

Kiinteistöautomaation ohjaamiin asioihin voi kuulua kohteesta riippuen: rakennuksen turvalaitteet, valaistus, lämmitys, jäähdytys, ilmanvaihto ja kulunvalvonta. Näiden lisäksi voi kiinteistöautomaatio ohjata tarvittaessa myös muita laitteita ja mukautua erilaisten rakennusten tarpeisiin. Turvalaitteet ovat yleensä hajautettu omiin ohjauskeskuksiinsa, jolloin ne eivät ole kaikki riippuvaisia yhden järjestelmän toimivuudesta ja niiden toimintavarmuus paranee.

Rakennuksen lämmitystä säädetään lämmönjakoverkoston venttiileillä, joilla hallitaan kaukolämmön- tai kiinteistön oman lämmitysjärjestelmän tuottaman kuuman veden energian jakamista eri lämmityspiireille. Energia siirretään piireille yleensä lämmönsiirtimien kautta.

Myös tiloihin puhallettavaa ilmaa täytyy lämmittää tai viilentää, jotta sisäilma pysyy ympäri vuoden tasaisen lämpöisenä, myös kauempana sisätilojen lämmitys- ja viilennyslähteistä. Ilmanvaihtoa ohjataan ilmanvaihtokoneikoilla tai ts. IV-koneilla. IV-koneet puhaltavat ulkoilmaa sisätiloihin ja palauttavat sisäilmaa ulos. Sisään puhallettavan ilman lämpötilaa säädetään tiloihin ja niiden käyttäjille sopiviksi erilaisilla lämmitys- ja jäähdytysratkaisuilla. Yleensä ilmaa lämmitetään ja jäähdytetään puhaltamalla se patterisäleikön läpi, jonka sisällä virtaa kylmä, tai lämmin neste. Patteri voi olla myös sähkölämmitteinen.

Pelkästään em. keinoin ilman lämpötilahallinta toimii, mutta se tulee kalliiksi, kun kaikki lämmitysteho tulee patteriverkoston vedestä, joka taas on kaukolämmöltä tulevaa, laskutettavaa lämpöenergiaa, tai rakennuksen oman öljypolttimen- ja sähkövastusten tuottamaa energiaa. Tämän välttämiseksi suurimmassa osassa uusia kiinteistöjä käytetään lämmön talteenottoa. Lämmön talteenotto, lyhennettynä LTO, perustuu sisään puhallettavan tuloilman lämmittämiseen ulos puhallettavan poistoilman lämpöenergialla. LTO voi toimia IV-koneessa tulo- ja poistoilmakanavien välillä pyörivällä lämmönsiirrinkennolla, tai omana nestepiirinään, jonka kautta lämpö siirretään tulo- ja poistopuolen patterisäleikköjen välillä. (Pentti, H. ym. 2012, 74–76)

Rakennuksen sisäilman muita ominaisuuksia, kuten hiilidioksidipitoisuutta ja kosteutta voidaan säätää esimerkiksi ilmanvaihtopuhaltimien kierrosnopeuksilla, tai säätöpelleillä. Tunkkainen sisäilma sisältää paljon hiilidioksidia ja kosteutta. Raikasta ulkoilmaa pitää puhaltaa tällöin enemmän, jotta olosuhteet pysyvät hyvinä. Liian voimakas ilman vaihtuvuus voi toisaalta tuntua rakennuksen käyttäjille epämiellyttävänä vedon tunteena. Ilmankosteutta voidaan hallita ilmankuivaimilla ja -kostuttimilla.

3.4 Energiatehokkuus

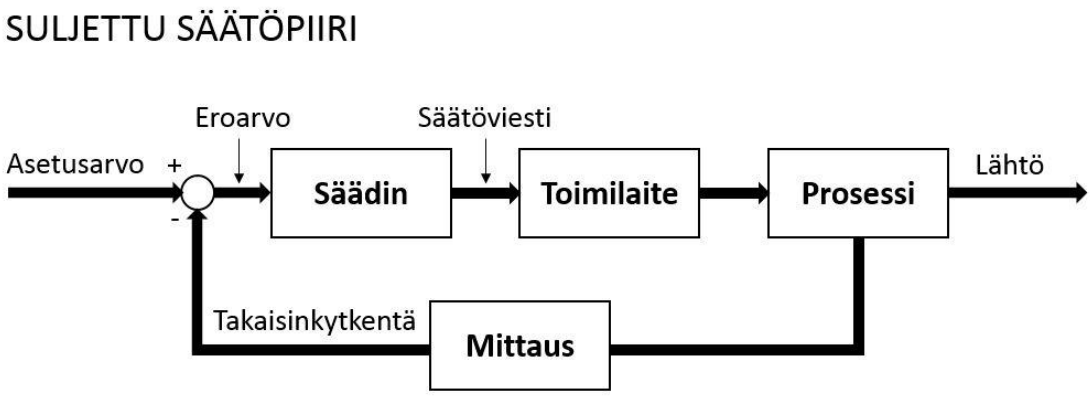
Nykypäivänä energiatehokkuus on noussut suureksi puheenaiheeksi ympäri maailmaa, varsinkin teollisuusmaissa. Joillekin kiinteistönomistajille ja yrityksille energiatehokkuus tarkoittaa vain säästöä sähkö- ja lämmityskuluissa, mutta monet perustelevat energiansäästön tarpeellisuutta myös ”vihreillä arvoilla”. Eli kun energiaa kuluu vähemmän, myös ympäristöä säästetään, sillä energian tuottaminen nykymenetelmillä tuottaa aina jonkinlaisia päästöjä tai ympäristöhaittoja.

Kiinteistöautomaation kannalta tätä asiaa on lähdetty edistämään mm. LTO-järjestelmillä, taajuusmuuntajilla, EC-puhaltimilla (EC=Electronically commutated), aikaohjelmilla ja mukautuvilla säädöillä. LTO-järjestelmä kierrättää talon käyttämää lämpöenergiaa. Taajuusmuuntajilla ohjatut puhaltimet, säädettävät EC-puhaltimet sekä pumput tuottavat vain tilanteen vaatiman virtauksen, kun niiden tehonsäätöä ohjataan portaattomaksi. Aikaohjelmat rajoittavat puhaltimien ja valojen käyttöä, kun rakennusta ei käytetä. Mittausten perusteella tehtävät säätöarvomuutokset säästävät energiaa vaarantamatta rakennuksen viihtyisyyttä.

3.5 Säätöpiirit

Automaatiotekniikassa säätöpiirin signaalit koostuvat yksinkertaisimmillaan kolmesta suureesta: asetusarvosta, mittauksesta ja säätöviestistä. Tätä kutsutaan suljetuksi säätöpiiriksi (ks. kuvio 1). Jos säädöllä ei ole mittausta, eli takaisinkytkentää järjestel-

mästä, on kyseessä avoin säätöpiiri, jota kutsutaan yleensä ohjaukseksi. Ohjaus voidaan toteuttaa erilaisten tilatietojen, ohjelmaehtojen ja aikaohjelmien perusteella, mittauksen sijaan.

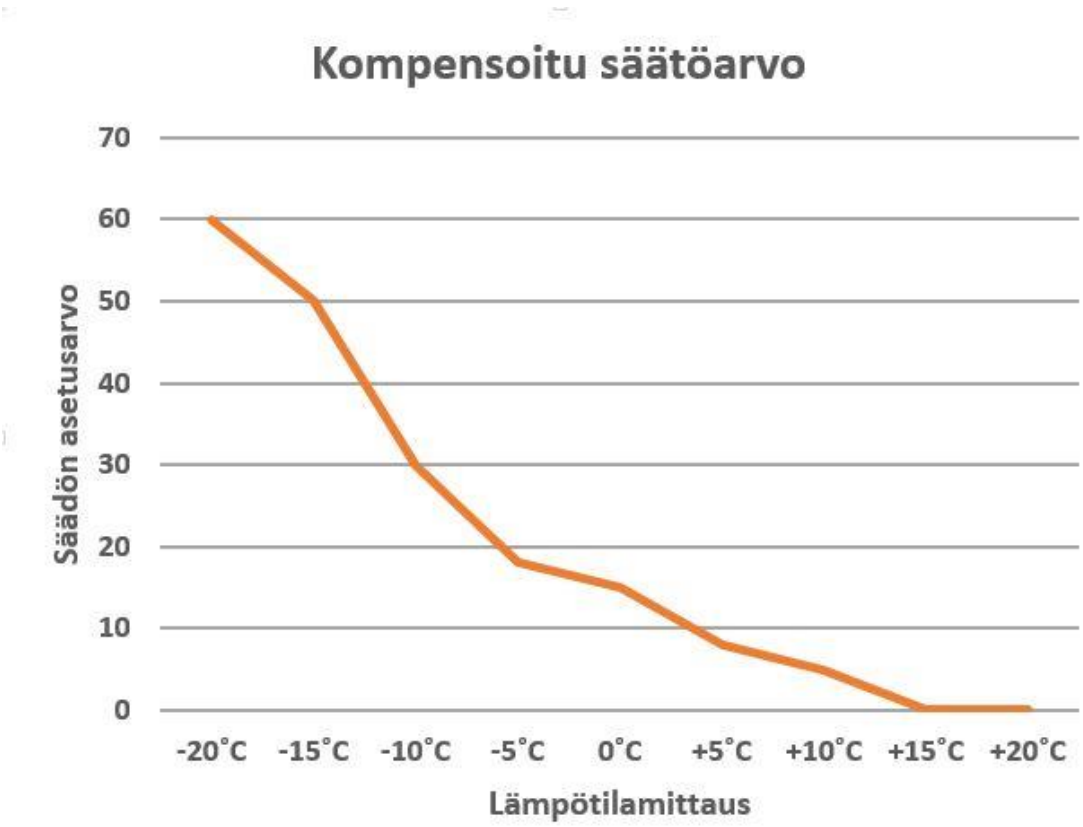


Kuvio 1. Suljetun säätöpiirin periaatekaavio

Asetusarvo määrittelee mihin arvoon mitattava suure halutaan. Asetusarvo voi olla käyttäjän määrittelemä vakioarvo, eli staattinen asetusarvo, tai säätimen laskema muuttuva arvo, eli dynaaminen asetusarvo. Mittaus kuvaa säädettävän suureen nykyistä tilaa, mutta sen luotettavuus riippuu mittauksen toteutuksen laadusta. Eroarvo on asetusarvon ja mittauksen välinen erotus. Eroarvon ja säätimen tekemän laskennan perusteella, säädin antaa säätöviestin toimilaitteelle, jotta mittauksen arvo saataisiin muuttumaan mahdollisimman lähelle säätöarvoa. (Pentti, H. ym. 2012, 55–56). Esimerkkinä tästä on tilanne, kun huonelämmön mitataan olevan +20,3 °C ja huoneeseen halutaan asetusarvon mukaisesti +21,0 °C. Tällöin halutaan lisää lämpöä ja säätimen säätöviesti kasvaa, jonka seurauksena patterin lämmitysventtiilin moottori alkaa avaamaan venttiiliä.

Säätöjen virityksellä tarkoitetaan säätimen ohjausviestin laskennassa käytettävien muuttujien ja kertoimien muuttamista. Tällä tavoin voidaan saada säätöviesti muuttumaan nopeammin tai hitaammin suhteessa mittauksen ja säätöarvon erotukseen. Säätöviestin muodostama säädin voi olla oma kokonaisuutensa, eli yksikkösäädin, tai ohjelmoitavan logiikan suorittama toimintolohko. (E.A. Parr, 2003, 164–169)

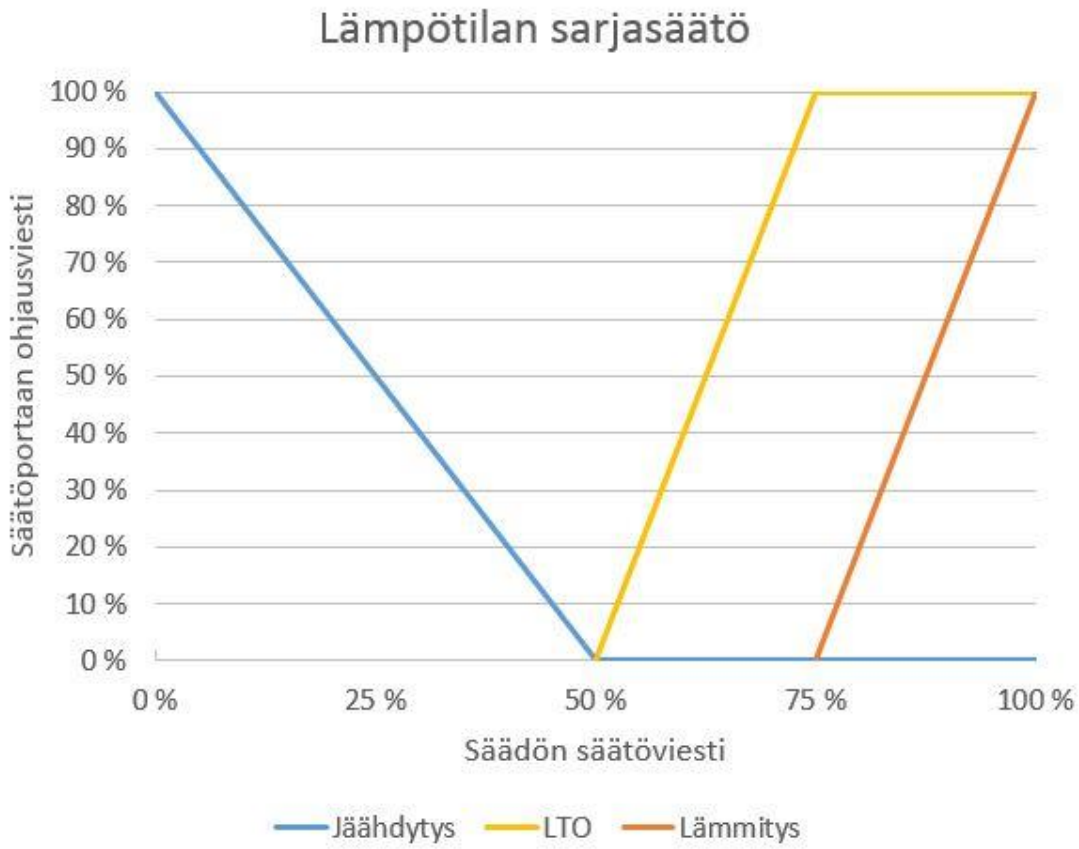
Säätöjen toteutustapoja on erilaisia, ja kiinteistöautomaatiossa käytetään vakioasetuspistesäätöjen lisäksi myös muita säätötapoja. Tärkeimpiä näistä on kompensointi- ja sarjasäätö. Kompensointisäädön toimintaa on kuvattu kuvion 2 kaaviossa. Kaavion vaaka-akselilla on lämpötilamittauksen arvoja ja pysty-akselilla niitä vastaavia säädön asetusarvoja. Kompensointisäädön asetusarvoa muutetaan jonkin mitattavan suureen perusteella. Esimerkiksi patteriverkoston lämpötiloja säädetään yleensä ulkolämpötilan mukaan kompensointisäätönä, jotta rakennuksen pattereiden lämpöenergia riittäisi lämmittämään tilat yhtä hyvin talvipakkasilla kuin kesälläkin.



Kuvio 2. Lämpötilakompensoitu säätöarvokäyrä

Sarjasäätöä on kuvattu kuviossa 3. Sarjasäädön 0-100 % välillä liikkuva säätöviesti on jaettu säätöportaisiin, joilla kaikilla on omat 0-100 % väliset säätöviestialueensa. Näin yhdessä säädössä saadaan ohjattua jokaisella säätöportaalla eri toimilaitetta. Esimerkkinä tästä on IV-koneen tuloilman lämpötilan sarjasäätö. Ensimmäisessä portaassa ohjata jäähdytyspatterin venttiiliä, eli vähennetään jäähdytystä. Kun halutaan lisää lämpöä, lopetetaan jäähdytys ja aloitetaan lämmitys lämmön talteenotolla. Kun

halutaan vielä enemmän lämpöä, mutta LTO ei enää pysty siirtämään sitä enempää, aletaan avaamaan lämmityspatterin venttiiliä. (Pentti, H. ym. 2012, 78–79)



Kuvio 3. Lämpötilan sarjasäätö

3.6 Säätimet ja logiikat

Automaation mittaukset, ohjaukset, indikoinnit, säädöt yms. ovat hyödyllisiä vain, jos niiden sisältämää tietoa voidaan valvoa ja järjestelmää ohjataan hallitusti. Tähän tarkoitukseen käytetään automaatiosäädintä, tai ts. logiikkaa, joka seuraa ja hallitsee siihen liitettyä järjestelmää automatisoidusti.

Logiikka voi sisältää valmiit, ennalta määritellyt toiminnot, jotka se suorittaa toistuvasti. Tämänlaiset logiikat sopivat pieniin sovelluksiin, joiden toteuttamat toimenpiteet eivät juurikaan muutu tilannekohtaisesti. Yleensä kuitenkin järjestelmät ovat

laajoja ja järjestelmää halutaan hallita monipuolisemmin, monimutkaisemmilla toiminnoilla. Tällöin käytetään ns. vapaasti ohjelmoitavaa logiikkaa. Nimensä mukaisesti tällaisen logiikan toiminnot ovat vapaasti ohjelmoitavissa, sen kanssa yhteensopivilla ohjelmistoilla ja ohjelmointikielillä.

Automaatiojärjestelmän laitteet ja anturit liitetään johdinkaapeleilla logiikkaan erillisten moduulien tai säätimeen integroituitujen moduuliliittimien kautta. Oleellisin syy tähän on, että ohjelmoitavat logiikat toimivat yleensä pienillä jännitteillä tietokoneiden tapaan, joten niiden piirit eivät kestä kenttälaitteiden käyttämiä suurempia jännitteitä. Lisäksi kun logiikka eristetään jollain tavalla kenttälaitteiden jännitteistä, eivät kentällä tapahtuvat häiriöt vaikuta logiikan toimintaan (E.A. Parr, 2003, 21). Nykyään myös langattomat yhteydet ovat yleistymässä, mutta niiden toimintaan liittyy yleensä enemmän häiriöherkkyyttä kuin langallisiin yhteyksiin.

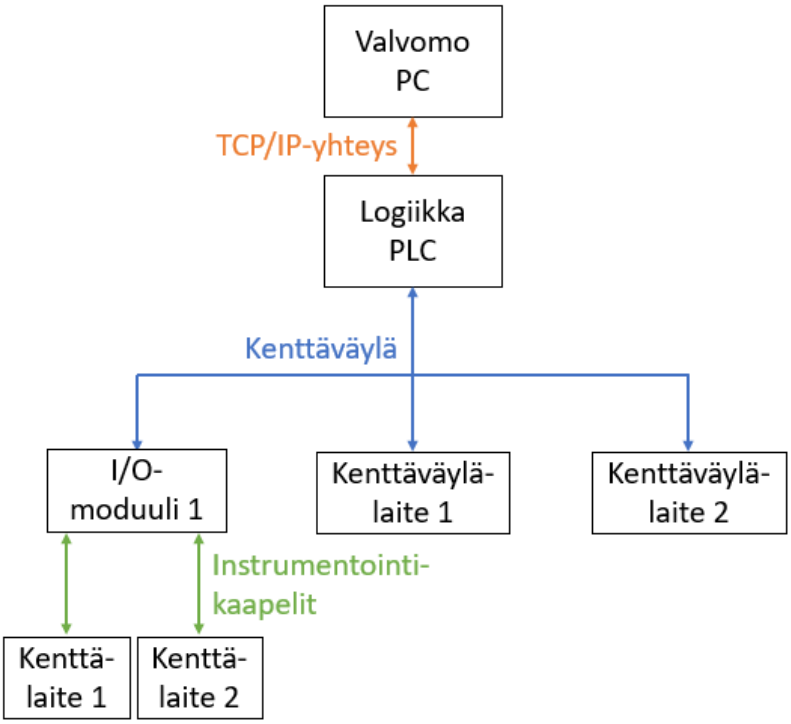
Moduulit muuttavat järjestelmän laitteista tulevat jännite-, tai virtasignaalit digitaaliseen muotoon. Tällä tavoin logiikka voi tulkita arvoja ohjelmakoodina. Sama tapahtuu myös toisinpäin, eli säätimen digitaaliset signaalit muutetaan moduuleilla kenttälaitteiden käyttämiksi jännite- ja virtaviesteiksi. Esim. jänniteviesti jonka minimiarvo on 0 V ja maksimiarvo 10 V, voi tulla lämpötilalähettimeltä, jonka pienin mahdollinen lukema on 0 ˚C ja suurin lukema 50 ˚C. Tässä tilanteessa 0 V vastaisi mitattua lämpötilaa 0 ˚C, 5 V vastaisi lämpötilaa 25 ˚C ja 10 V lämpötilaa 50 ˚C. Kun tämä jännite-viesti muutetaan digitaaliseksi, voi sen arvo olla säätimen muistissa vaikkapa kokonaisluku väliltä 0-100 000. Tämä arvo muutetaan säätimen käyttöjärjestelmää varten mittaukselle valitun muunnostaulukon avulla takaisin lämpötila-arvoksi.

Monesti näistä tietokytkennöistä käytetään nimitystä ”piste”. Fyysisten pisteiden yleisimpiä tyyppejä ovat: Analogiatulo (Analog Input), Analogialähtö (Analog Output), digitaalitulo (Digital Input) ja digitaalilähtö (Digital Output). Analogiatuloja käytetään portaattomien mittaustietojen lukemiseen ja analogialähtöjä toimilaitteiden portaattomaan ohjaamiseen. Digitaalituloilla luetaan yksinkertaisia tilatietoja (esim. käyntitilat, hälytykset, pulssitiedot) ja digitaalilähtöjen avulla voidaan ohjata yksinkertaisia toimilaitteita, valoja ja koneita (esim. ON/OFF-kytkin). Näitä pisteitä voi olla yhdessä

kiinteistöautomaatioprojektissa jopa tuhansia, riippuen rakennuksen- ja sen toimintojen laajuudesta. (Pentti, H. ym. 2012, 104–107)

Logiikassa ei monesti ole integroituja I/O-pisteitä, tai järjestelmässä voi olla liikaa pisteitä logiikan omiin liittimiin kytkettäväksi. Tällöin täytyy käyttää ulkoisia IO-moduuleja, jotka liitetään logiikkaan kenttäväylän avulla. Hajautetusta IO-järjestelmästä on monia hyötyjä, joista huomattavin on kaapelikustannusten väheneminen. (E.A. Parr, 2003, 29–30)

Kenttäväyliä on monenlaisia, mutta niiden periaate on samantapainen: muutetaan monelta eri laitteelta lähtevät- ja niille tulevat signaalit sarjaliikennemuotoon ja siirretään kaikkien laitteiden tiedot samaa johdinparia pitkin laitteelta toiselle (ks. kuvio 4). Tällä tavoin logiikkaan voi tulla yhtä väyläkaapelia pitkin jopa satoja, tai jopa tuhansia I/O tietoja. (Pentti, H. ym. 2012, 143–144)



Kuvio 4. Esimerkki kenttäväyläratkaisusta

Fyysisesti kytkettyjen pisteiden järjestelmän käytössä on yleensä myös ohjelmallisia pisteitä. Ohjelmalliset pisteet saavat arvonsa logiikkaohjelman, ja säätimellä tehtävien laskutoimitusten tai ehtolauseiden tuloksena. Nämä pisteet eivät pysty yksin suoraan vaikuttamaan laitteiden toimintaan, mutta niiden tilojen perusteella voidaan välillisesti ohjata muita fyysisiä pisteitä. (Pentti, H. ym. 2012, 104)

4 Fidelix-järjestelmät

4.1 Yritys

Fidelix Oy on Suomessa v. 2002 perustettu kiinteistöautomaatio- ja turvajärjestelmiä toimittava yritys. Yrityksen liikevaihto vuonna 2016 oli 20 M€ ja se työllistää päämarkkina alueellaan yli 150 henkilöä. (Fidelix Oy, 2017)

4.2 Automaatiosäätimet

Fidelix on kehittänyt ja tuotteistanut mm. erilaisia säätimiä, keskusyksiköitä, I/O-moduuleita ja turvamoduuleita. Työn kannalta merkittävimmät ovat keskusyksiköt tai ts. automaatiosäätimet, joihin testausohjelma kohdistuu. Näitä ovat Fidelixin vapaasti ohjelmoitavat logiikat, eli FX2020 – perheen yksiköt, sekä Fidelix Spider (ks. kuvio 5). Suurin osa Fidelixin keskusyksiköistä sisältää kosketusnäytön, joka toimii automaation käyttöliittymänä. (Fidelix Oy, 2017)

Kuvio 5. Fidelix Spider -keskusyksikkö

4.3 Ohjelmistot

Fidelix Oy:n kehittämiä automaatioprojektien kannalta tärkeimpiä ohjelmistoja ovat Fidelix HTML-Editor, Fidelix PointGen, Fidelix Editor ja Fidelix Connection. Näistä uudempia ohjelmia ovat Fidelix Editor ja Fidelix Connection, mutta vanhemmat ohjelmat Fidelix HTML-Editor ja Fidelix PointGen sisältävät kuitenkin suunnilleen samat perustyökalut.

Fidelix-Editor pyrkii yhdistämään kaikki Fidelixin laitteilla tehtävän projektin vaatimat työkalut ja ohjelmat. Se sisältää pistelistan luomistyökalun, jolla voidaan määritellä projektiin tulevat indikointi-, hälytys-, ohjaus-, mittaus- ja säätöpisteet, sekä aikaohjelmat. Ohjelmalla voidaan ladata pisteet ja konfiguraatiot säätimeen, tehdä muutoksia säätimen pisteisiin ja tarvittaessa monitoroida niiden tiloja. Ohjelmisto sisältää myös grafiikkaeditorin, jolla pystytään luomaan säätimen näytöllä näkyvät

kuvat ja valikot. Ohjelmistossa on myös paljon muita ominaisuuksia, joita kaikkia ei käydä tässä läpi. (Fidelix Ohjelmointimanuaali, 2017)

Fidelixin ohjelmoinnissa käytetään infoteam OpenPCS-ohjelmaa. Ohjelma mahdollistaa IEC 61131-3 standardin mukaisen ohjelmoinnin, jossa käytettäviä ohjelmointikieliä ovat FBD, STL, SFC, CFC, ST, IL ja LAD. (Korhonen V. 2013) OpenPCS ohjelman kautta voidaan ladata ohjelmat säätimeen ja seurata pisteiden tiloja, sekä koodin suoritusta reaaliaikaisesti. (infoteam software AG, 2017)

5 Työn toteutus

5.1 Suunnittelu

Suunnittelu aloitettiin ohjelman toimintojen määrittelyllä ja työn tietoperustaan pohjautuvalla pohdinnalla. Toimeksiantajan kanssa todettiin, että ohjelmasta saadaan eniten hyötyä, kun se tehdään mahdollisimman joustavaksi ja helposti muunneltavaksi eri projektien tarpeisiin.

Tärkeimpiä kysymyksiä työn suunnittelun kannalta olivat: mitä säätimen itsediagnostiikan täytyy- ja mitä se voi tutkia säätimen toiminnasta, jotta mahdollisimman paljon virheitä voidaan tunnistaa? Mistä virheistä säätimen tulisi ilmoittaa ja millä tavalla? Miten Itsediagnostiikasta saadaan mahdollisimman kevyt toimenpide, joka on helposti ohjelmoitavissa? Mitä muita käyttömahdollisuuksia ohjelmalla voisi olla virhetestauksen lisäksi?

Testausohjelman tärkeimmiksi toiminnoiksi määriteltiin suunnitteluvaiheessa:

- Fyysisten pisteiden virhetilojen tunnistaminen
- Säätövirheiden tunnistaminen / säätöjen virittäminen
- Säätimen itsediagnostiikkatietojen tuominen käyttäjän nähtäville
- Kerätyn tiedon koostaminen mahdollisimman havainnolliseen muotoon

Fidelixin luomasta ohjelmakirjastosta löytyi jo suunnitteluvaiheessa sopivia ohjelmalohkoja haluttujen toimintojen suorittamiseen, mutta suoraa säätöjen viritysmahdol-

lisuutta ei niissä ollut. Vaihtoehtoiseksi ratkaisuksi suunniteltiin säätötestausten perusteella tehtävää laskentaa, joka ehdottaa käyttäjälle sopivia säätöarvoja säätöpistekohtaisesti.

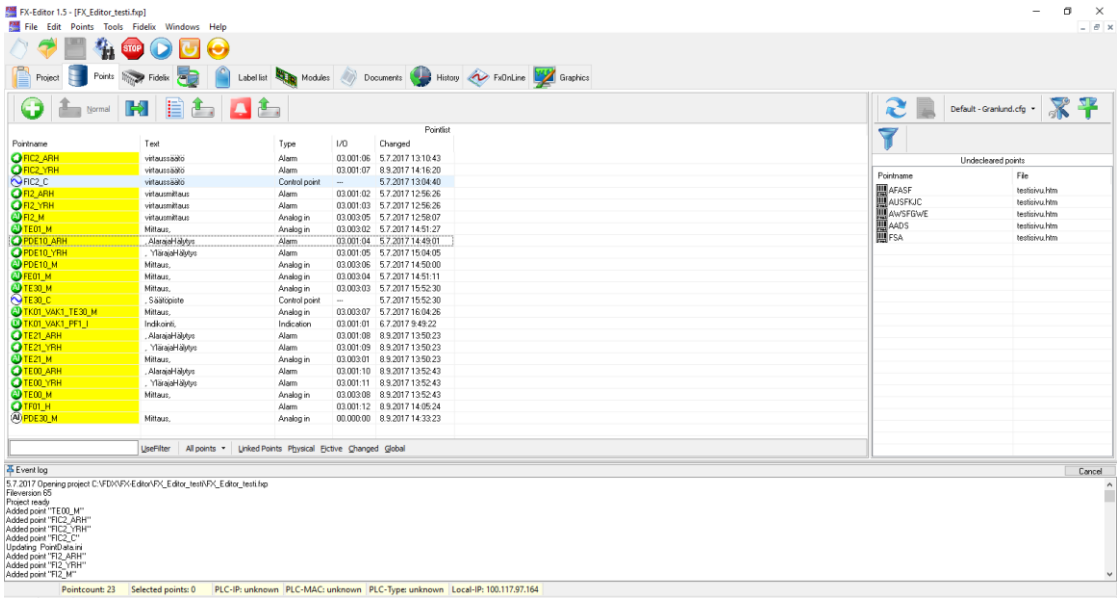
Itsetestausohjelmasta tuli saada mahdollisimman helppokäyttöinen ja helposti muunneltava eri projektien tarpeisiin. Lisäksi ohjelman muokkaus uuteen projektiin sopivaksi ei saisi viedä liikaa aikaa, sillä yksi työn tavoitteista on työtuntien säästäminen. Testausohjelmasta pitäisi saada myös mahdollisimman hyödyllinen ja havainnollistava. Näiden asioiden yhteensovittaminen oli suunnittelun tärkein tavoite, sillä samaan aikaan mahdollisimman kattavan ja helposti muunneltavan testausohjelman toteuttaminen luo haasteita.

Näiden tavoitteiden ja haasteiden perusteella syntyi ajatus Excel-pohjaisesta ratkaisusta, jossa ohjelmakoodi voitaisiin muodostaa tekstinä, projektissa käytettävien pisteiden perusteella. Excel on Microsoftin taulukkotyökalu, jolla voi tehdä laskentataulukkoja, kaavakkeita, kuvaajia ja monia muita sovelluksia.

Fidelix-projektien pisteet määritellään joko Pointgen- taulukkotyökaluun (ks. kuvio 6), tai Fidelix Editor -pistelistaan (ks. kuvio 7), joka voidaan myös muuntaa Excel-taulukoksi suoraan Fidelix Editor -ohjelmasta. Pistetunnukset saadaan taulukosta samassa muodossa, kuin missä ne ladataan säätimeen. Säätimeltä voidaan myös myöhemmin ladata pisteet tekstimuodossa ja liittää ne Excel-taulukkoon. Pistelistasta on mahdollista Excel-laskukaavojen ja -ehtolausekkeiden avulla muodostaa projektiin puhdasta ja virheetöntä koodia automatisoidusti. Tämä koodi voitaisiin kopioida Excel-taulukosta suoraan OpenPCS-ohjelmistoon ja liittää osaksi testattavan projektin ohjelmakoodia.

1	Version	Delete points	Count points			Add Modules			Check Module and Point number of Points	
2	123									
3	SaveAndExit	Delete modules	Zoom all						Add PointTable points to PointList	
4										
68	VAK1_TK01_SC02_H	Mallihuone LTO-Kiekko	3	1	?	1	5	0	HALYTYS	5
69	**IND**									
70	**Name**	**Text**	**Port**	**Module**	**Point**	**OnDelay**	**OffDelay**	**Open Contact 0/1**	**StateText**	**TriState 0/1**
71	**Write DefaultValues into this line**		**3**	**?**	**?**	**0**	**5**	**0**	**SEIS_KAY**	**0**
72	VAK1_LL03_P01_I	IV-Verkosto Pumppu	3	?	?	0	5	0	SEIS_KÄY	0
73	VAK1_IV02_P01_I	IV-Verkosto Pumppu	3	?	?	0	5	0	SEIS_KÄY	0
74	VAK1_PV01_P01_I	Patteriverkosto Pumppu	3	?	?	0	5	0	SEIS_KÄY	0
75	VAK1_KV01_P01_I	Käyttövesiverk. pumppu	3	?	?	0	5	0	SEIS_KÄY	0
76	VAK1_TK01_PF02_I	Mallihuone PF01 Taaj.muuttaja	3	?	?	0	5	0	SEIS_KÄY	0
77	VAK1_TK01_SC21_I	Mallihuone PF01 Taaj.muuttaja	3	?	?	0	5	0	SEIS_KÄY	0
78	VAK1_TK01_FG19_I	Mallihuone Poistoilma	3	?	?	0	5	0	SEIS_KÄY	0
79	VAK1_TK01_HS16_I	Mallihuone Lisäaikakytkin	3	?	?	0	5	0	SEIS_KÄY	0
80	VAK1_TK01_SC08_I	Mallihuone TF01 Taaj.muuttaja	3	?	?	0	5	0	SEIS_KÄY	0
81	VAK1_TK01_P04_I	Mallihuone LP-Pumppu	3	?	?	0	5	0	SEIS_KÄY	0
82	VAK1_TK01_FG01_I	Mallihuone Raitisilmapelti	3	?	?	0	5	0	SEIS_KÄY	0
83	**DO**									
84	**Name**	**Text**	**Port**	**Module**	**Point**	**OnDelay**	**OffDelay**	**Open Contact 0/1**	**StateText**	**TriState 0/1**
85	**Write DefaultValues into this line**		**3**	**?**	**?**	**0**	**0**	**0**	**SEIS_KAY**	**0**
86	VAK1_LL03_P01_O	IV-Verkosto Pumppu	3	?	?	0	0	0	SEIS_KÄY	0
87	VAK1_IV02_P01_O	IV-Verkosto Pumppu	3	?	?	0	0	0	SEIS_KÄY	0
88	VAK1_PV01_P01_O	Patteriverkosto Pumppu	3	?	?	0	0	0	SEIS_KÄY	0
89	VAK1_TK01_PF02_O	Mallihuone PF01 Taaj.muuttaja	3	?	?	0	0	0	SEIS_KÄY	0
90	VAK1_TK01_SC21_O	Mallihuone PF01 Taaj.muuttaja	3	?	?	0	0	0	SEIS_KÄY	0
91	VAK1_TK01_FG19_O	Mallihuone Poistoilma	3	?	?	0	0	0	SEIS_KÄY	0
92	VAK1_TK01_HS16_O	Mallihuone Lisäaikakytkin	3	?	?	0	0	0	SEIS_KÄY	0
93	VAK1_TK01_SC08_O	Mallihuone TF01 Taaj.muuttaja	3	?	?	0	0	0	SEIS_KÄY	0
94	VAK1_TK01_FG01_O	Mallihuone Raitisilmapelti	3	?	?	0	0	0	SEIS_KÄY	0
95	**AI**									
96	**Name**	**Text**	**Port**	**Module**	**Point**	**Analog=1**	**Scaling**	**Min Pulse**	**Offset**	**Lookup Table**
97	**Write DefaultValues into this line**		**3**	**?**	**?**	**1**	**1**	**10**	**0**	
98	VAK1_KV01_FQ03_M	Käyttövesiverk. Kylmävesimäärä	3	0	0	1	1	10	0	
99	VAK1_TK01_FE21_M	Mallihuone PF01 ilmamäärä	3	0	0	1	1	10	0	
100	VAK1_TK01_FE08_M	Mallihuone TF01 ilmamäärä	3	0	0	1	1	10	0	
101	VAK1_TK01_SC01_M	Mallihuone Tuloilmakone	3	0	0	1	1	10	0	
102	VAK1_LL03_TE42_M	IV-Verkosto Paluuvesi	3	?	?	1	1	10	0	
103	VAK1_LL03_TE41_M	IV-Verkosto Menovesi	3	?	?	1	1	10	0	
104	VAK1_LL03_PE42_M	IV-Verkosto Paine	3	?	?	1	1	10	0	
105	VAK1_LL03_PDE01_M	IV-Verkosto Paine-ero	3	?	?	1	1	10	0	
106	VAK1_IV02_TE42_M	IV-Verkosto Paluuvesi	3	?	?	1	1	10	0	
107	VAK1_IV02_TE41_M	IV-Verkosto Menovesi	3	?	?	1	1	10	0	
108	VAK1_IV02_PE42_M	IV-Verkosto Paine	3	?	?	1	1	10	0	
109	VAK1_IV02_PDE01_M	IV-Verkosto Paine-ero	3	?	?	1	1	10	0	

Kuvio 6. Fidelix PointGen -pistelista



Kuvio 7. Fidelix Editor -pistelista

Testausohjelma tarvitsee myös tavan ilmaista käyttäjälle testien tulokset ja mahdolliset virheet. Tähän tarkoitukseen voidaan luoda käyttöliittymä Fidelix HTML-Editor –ohjelmalla, jolla luotu HTML-sivu näkyy säätimen kosketusnäytöllä. Kaikkien järjestelmän pistekohtaisten tietojen näyttäminen käyttöliittymässä ei olisi kuitenkaan mahdollista, koska grafiikkakuvaan mahtuu kerrallaan vain rajattu määrä numerokenttiä

ja tekstiä. Lisäksi jos kuvan koko paisuu suureksi, täytyy käyttäjän selata sivua edestakaisin, joka sekavoittaa sen käyttöä. Tämän vuoksi tutkittiin vaihtoehtoisia tapoja välittää testaustiedot käyttäjälle. Ratkaisuksi suunniteltiin Fidelixin ohjelmalohkoa, jonka avulla pystytään kirjoittamaan raporttitekstitiedostoja säätimen muistikortille.

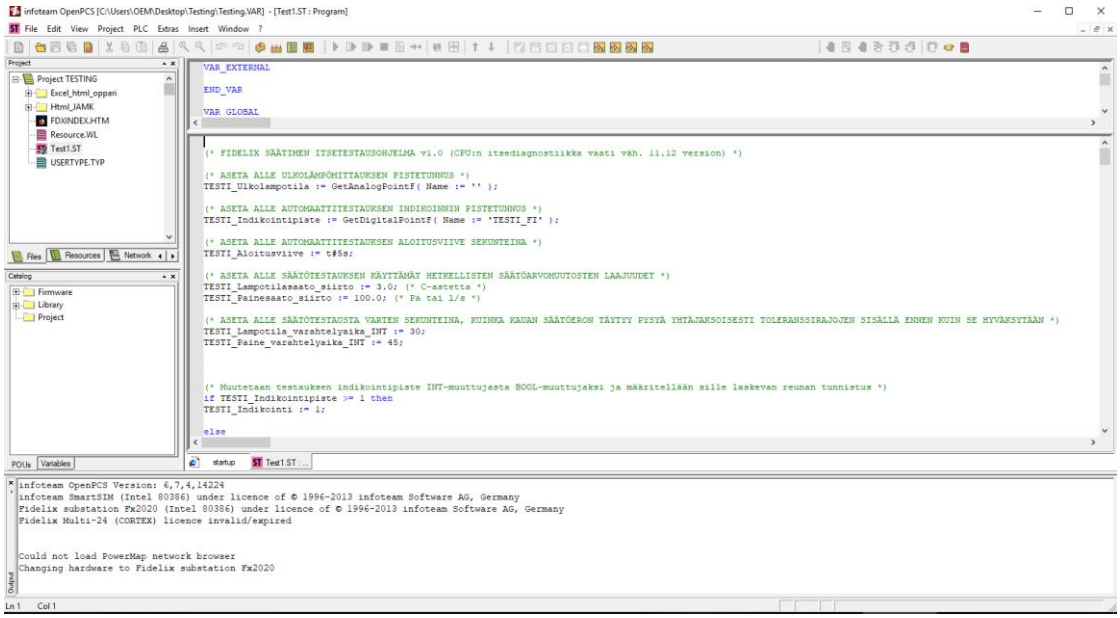
Käyttöliittymä pyrittiin suunnittelemaan mahdollisimman helppokäyttöiseksi ja havainnolliseksi. Käyttöliittymän laadulla voi olla suuri vaikutus käyttömukavuuteen ja järjestelmän tehokkaaseen hyödyntämiseen. Käyttöliittymä päätettiin pitää yksinkertaisena, mutta paljon tietoa sisältävänä.

Jotta työn tulokset eivät jäisi vain teorian varaan, päätettiin itsetestausohjelmaa testata työhön soveltuvalla simulaatio-ohjelmalla, tai oikeassa järjestelmä-ympäristössä. Simulaatio-ohjelman käyttö vaatisi kuitenkin työhön soveltuvan ohjelman löytämisen ja siihen tutustumisen. Lisäksi fyysisessä järjestelmässä testattu työ lisäisi tulosten uskottavuutta.

Parhaaksi testauskäytännöksi arvioitiin työelämän projektissa käytettävän järjestelmän testaus, mutta aikarajoitteista johtuen ohjelman testaukset päätettiin tehdä Jyväskylän Ammattikorkeakoulun rakennusautomaatiolaboratoriossa sijaitsevalla opetuskäyttöön tehdyllä laitteistolla. Samaa laitteistoa käytettiin tämän työn kehitysvaiheessa, välitulosten testaamiseen.

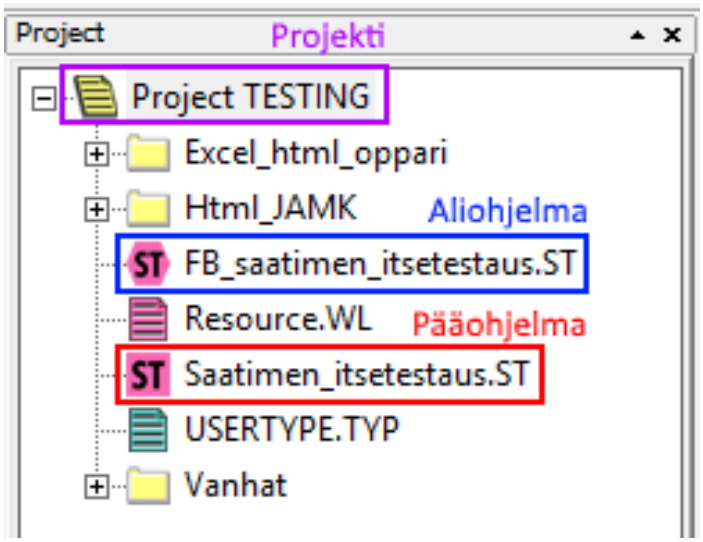
5.2 Ohjelmointi

Työn ohjelmointi toteutettiin OpenPCS – ohjelmistolla (ks. kuvio 8), jota käytetään Fidelixin säätimien ohjelmoinnissa. Työssä käytettiin ohjelmointikielenä ST-kieltä, joka muistuttaa rakenteeltaan hieman C-kieltä (Korhonen, V. 2013). ST-kieli sopi testausohjelman ohjelmoimiseen hyvin, sillä siinä voi helposti tehdä laskutoimituksia ja ehtoja muutamalla rivillä koodia, sekä kommentoida koodia riveittäin. Tärkein syy ST-kielen käytölle on kuitenkin testausohjelman muuttujien määrittelyssä, eli parametroinnissa käytettävä tekstipohjainen koodin luominen. Tämän lisäksi Fidelixin ohjelmointimanuaali ohjeistaa ohjelmointia vain ST-kielellä.



Kuvio 8. OpenPCS-ohjelmointinäkymä

Ensimmäisenä ohjelmaa varten täytyi luoda uusi OpenPCS-projekti, johon määriteltiin logiikan hardware module, eli logiikan fyysinen konfiguraatio. Konfiguraatio kertoo ohjelmalle käytettävän logiikan ominaisuudet. Projektiin luotiin ohjelman function block, eli aliohjelma ja program, eli pääohjelma (ks. kuvio 9). Tehtyjä ohjelmia suoritetaan jatkuvasti sykleinä. Tämä tarkoittaa sitä, että ohjelmakoodi prosessoidaan ohjelmointinäkymässä ylhäältä alaspäin ja kun koodin loppu saavutetaan, otetaan käyttöön muuttuneet arvot, jonka jälkeen aloitetaan alusta uusi ohjelmakierros, eli sykli. Ohjelman suoritusväliä voidaan muuttaa tarvittaessa.



Kuvio 9. OpenPCS-projektin ohjelmaosat

Koodin yläpuolella olevassa ikkunassa määritellään ohjelman käyttämät muuttujat. Muuttujat ovat ikään kuin ohjelman omia muistipaikkoja, joihin voi tallentaa erilaisia arvoja ja joista näitä arvoja voi myöhemmin lukea, viittaamalla muuttujan nimeen koodissa.

Tässä ohjelmassa käytetyt muuttujatyypit ovat integer (lyh. INT), real, boolean (lyh. BOOL) ja string. Integer-muuttuja voi sisältää vain kokonaisluvun ja real-muuttuja vain desimaaliluvun. Boolean-muuttuja voi saada vain kaksi eri arvoa, true ja false, jotka voi määritellä sille myös numeroina 1 ja 0. String-muuttuja voi sisältää merkkijonon, eli tekstiä ja muita kirjoitusmerkkejä. Muuttujille voi antaa arvot ohjelmassa, esim. ehtolauseiden perusteella, tai niiden arvot voi antaa muuttujamäärittelyssä. Tätä kutsutaan muuttujien alustamiseksi (ks. kuvio 10).

```
VAR

Totuusarvo := TRUE :BOOL;
Kokonaisluku := 25 :INT;
Reaaliluku := 12.35 :REAL;
Merkkijono := 'tekstipatka' :STRING(20);

END_VAR
```

Kuvio 10. Erityyppisten muuttujien alustus

Kun muuttujat on määritelty, voidaan aloittaa toimenpiteiden ohjelmointi. Tämän ohjelman kannalta keskeisin ohjelmakielikomento on if-else -lause (ks. kuvio 11). If-lauseessa määritellään yksinkertaisimmillaan aluksi ehto, sitten toimenpide joka suoritetaan jos em. ehto täyttyy. Yleensä lauseessa on kuitenkin lisäksi toimenpide, joka suoritetaan, jos ehto ei täyty. Väliin voidaan lisätä myös enemmän ehtoja ja toimenpiteitä elsif-komennolla. Sisäkkäisillä ehtolauseilla saadaan suoritettua jo verrattain monimutkaisia toimenpiteitä.

```
if Ehto = TRUE then      Jos tämä ehtolause toteutuu...
    Toimenpide := TRUE;  Suoritetaan tämä komento
else                     Muussa tapauksessa...
    Toimenpide := FALSE; Suoritetaan tämä komento
end if;                  if-else -lause loppuu
```

Kuvio 11. If-else –lauseen rakenne

Pisteiden muuttujat ja ohjelman toiminnot ovat kaikki ohjelmalohkossa jota kutsutaan aliohjelmana pääohjelmasta. Tällä tavoin pääohjelmassa täytyy vain antaa muuttujien arvoja aliohjelmalle, joka hoitaa ehtolauseiden tarkastelun ja suorittaa halutut toiminnot (ks. kuvio 12). Aliohjelmatasoja voi olla useita, eli aliohjelma voi sisältää kutsun toiseen aliohjelmaan. Tässä työssä aliohjelmatasoja on kaksi, joista syvemmällä tasolla suoritettiin Fidelixin luomat, säädinpisteiden käsittelyyn käytettävät funktiot.

```
VAR
ITSETESTAUS :FB_saatimen_itsetestaus;   Määritellään muuttuja nimeltä ITSETESTAUS, jonka tyyppi on FB_saatimen_itsetestaus -aliohjelma
END_VAR

(* FIDELIX SÄÄTIMEN ITSETESTAUSOHJELMA v1.0 *)

(* Huom. CPU:n itsediagnostiikka vaati väh. 11.12 version, jännitetieto vaati FX2030 säätimen *)

ITSETESTAUS                             Kutsutaan muuttujaa/aliohjelmaa suorittamaan siihen määritellyt toimenpiteet
(

(***************************** YLEISASETUKSET *****************************)

(* ASETA ALLE ULKOLÄMPÖMITTAUKSEN PISTETUNNUS (OPTIO, NÄYTTÄÄ ULKOLÄMPÖTILAN RAPORTIN ALUSSA)*)
TESTI_Ulkolampotila_tunnus := '',       Annetaan aliohjelman muuttujille arvoja, joiden avulla se suorittaa halutut toimenpiteet

(* ASETA ALLE AUTOMAATTITESTAUKSEN INDIKOINNIN PISTETUNNUS (OPTIO, TESTAUS VOIDAAN MYÖS MANUAALIS
TESTI_Indikointipiste_tunnus := 'TESTI_FI',

(* ASETA ALLE AUTOMAATTITESTAUKSEN ALOITUSVIIVE TASASEKUNTEINA (OPTIO, IV-KONEEN KÄYNNISTYMISVAIH
TESTI_Aloitusviive_ohjelma := 5,
```

Kuvio 12. Aliohjelman kutsu pääohjelmassa

Pääohjelman alkuun tehtiin perustoiminnot ja yleiset muuttujat, joiden lisäksi siihen kopioitaisiin Excelillä luotu järjestelmäkohtainen ohjelmakoodi. Tällä tavoin ohjelman käyttäjän ei tarvitse ymmärtää aliohjelmassa tapahtuvista toiminnoista mitään ja luodun ohjelmakoodin liittäminen helpottuu.

Pisteiden testauksessa käydään läpi fyysisten pisteiden helposti todettavat virhetilat. Fidelix-säädin osaa itse luokitella tietynlaisia virhetiloja, antamalla pisteelle havaittua virhetilaa vastaavan arvon. Pisteen arvo -9999 tarkoittaa, että kyseisen nimistä pistettä ei löydy säätimeltä. Arvo -9998 tarkoittaa, että pisteen arvo ei ole sen muunnostaulukon alueella. Kyseessä voi tällöin olla fyysinen anturivika, tai pisteen muunnostaulukko on vääränlainen. Pisteen muunnostaulukko muuntaa säätimen pisteelle antaman 0-100 000 digitaaliarvon mittasuureen yksiköiksi, esim. lämpötilaksi välillä 0-50 °C. (Fidelix Manual, 2017.)

Muita virhetiloja ovat pisteen arvot -9997 ja -9996. Arvo -9997 tarkoittaa globaalipisteen tiedonsiirtovirhettä. Globaalipiste on piste, jonka säädin jakaa toisen säätimen kanssa. Pisteen arvon muuttuessa toisella säätimellä, muuttuu se myös toisessa. Tiedot synkronoidaan erilaisten TCP/IP-yhteyksien kautta. Tästä virhetiedosta voidaan todeta säädinten välisen yhteyden tila. Virhetila -9996 tarkoittaa moduulivikaa. Tällä virhetiedolla kerrotaan, että pisteen IO-moduulissa on fyysinen vika, tai säädin ei saa yhteyttä moduuliin, esim. väylävian vuoksi. (Fidelix Manual, 2017.)

Fyysiset pisteet testataan vain em. virheiden ja käsikäytön varalta. Kun piste on käsin-tilassa, sen arvoa pakko-ohjataan, jolloin säädin ja logiikkaohjelma eivät yleensä pysty vaikuttamaan pisteen arvoon. Tämä tila ei ole yleensä haluttu, muulloin kuin väliaikaisten huoltotoimenpiteiden ja järjestelmän väliaikaisen tilamuutoksen yhteydessä. Yleensä halutaan, että järjestelmä pystyy itse säätämään ja mittaamaan pisteiden arvoja. Tällöin pisteet ovat automaatti-tilassa.

Pääohjelman rakenne on seuraava:

1. Kutsutaan aliohjelmalohkoa tekemään toimenpiteet
2. Annetaan aliohjelmalohkolle käyttäjän määrittelemät muuttujien arvot
3. Annetaan aliohjelmalohkolle Excelillä luodut muuttujien arvot

Aliohjelmalohkon rakenne on yksinkertaistettuna seuraava:

- Alustusvaiheet:

1. Luetaan testin perusmuuttujat ja Excel-generoidut muuttujat
2. Tallennetaan diagnostiikkatietoja muuttujiin ja pisteisiin

3. Alustetaan testin laskurit, ajastimet, sekä aktiivisuus- ja vaihe-ehdot

- Toistettavat vaiheet

4. Kun uusi testaus alkaa, aloitetaan myös Excel-generoidun koodin käsittely ja uuden raportin kirjoitus
5. Testataan Excel-työkalusta tuotujen pisteiden tilat ja luetaan niiden arvot
6. Fyysisten pisteiden jälkeen testataan säätöpisteiden toiminta säätötestauksilla (jos säätötestaukset ovat käytössä)
7. Kun kaikki pisteet on testattu, lopetetaan testaus ja raportin kirjoitus
8. Kun testaus aloitetaan uudestaan, nollataan laskurit ja toistetaan vaiheet 4-8

Toteutusvaiheessa huomattiin että ohjelmasta on kehittymässä hieman raskas kokonaisuus, joten sopivien säätöarvojen laskennasta luovuttiin. Toiminnon toteuttaminen olisi lisännyt pistekohtaisen koodin määrää huomattavasti ja vaatinut monia lisämuuttujia. Toiminnon tilalle tehtiin säätötestaus, joka testaa vain valmiiksi viritettyjen säätöjen toimivuuden.

Koska kiinteistöautomaation säädöt sisältävät vakiosäätöjen lisäksi paljon kompensointi- tai sarjasäätöjä, todettiin helpoimmaksi testaustavaksi yksinkertainen säätöarvon muutos. Tällöin säädön tyypillä ei ole väliä ja testauksen ohjelmointi helpottuu huomattavasti. Säätöpisteen toiminnan kannalta tärkeintä on että se pystyy saavuttamaan mittauksen uuden asetusarvon halutulla aikavälillä ja pitämään sen tasaisesti uudessa arvossa.

Säätöpisteiden testaukseen oli joka tapauksessa luotava monia muuttujia ja ohjelmaehtoja, joten niiden suoritus rajoitettiin tässä testausohjelman versiossa lämpötila-, paine- ja ilmamäärätesteihin. Nämä ovat yleisimmät säätökohteet, joten niiden testaamisella saataisiin jo suurin osa järjestelmän säädöistä testattua. Testattavia säätösuureita rajoitettiin, sillä jokaista testattavaa suuretta kohti täytyi ohjelmaan määritellä kyseisen suureen ominaisuuksia ja peruskäyttöä vastaavat tarkastelut ja säädinpisteet.

Lämpötilan kanssa samoilla muutosalueilla voisi testata ilmankosteuden säätöjä. Paine- ja ilmamääräsäädöt yhdistettiin samaksi kokonaisuudeksi, sillä niiden suurealueet ovat karkeasti samaa suuruusluokkaa. Myös hiilidioksidisäätöjä voitaisiin mahdollisesti testata ilmamääräsäätöjen kanssa samoilla muutosalueilla.

Säädön asetusarvoa muutetaan testauksen ajaksi ohjelmallisesti käyttäjän säätötyypille määrittelemän erotuksen verran. Esimerkiksi säätöarvo on 200 Pa painearvo ja testausohjelman käyttäjä määritellyt painesäätöjen testausmuutokseksi +20 Pa. Tällöin muutetaan säätötestausten ajaksi säädön asetusarvo 220 Pascaliin ja seurataan, pystyykö säätöpiste saavuttamaan uuden asetusarvon ja pitämään sen tasaisena. Säädön tulee saavuttaa uusi asetusarvo käyttäjän määrittelemän ajan kuluessa ja sen jälkeen pitää säädön eroarvo käyttäjän määrittelemien toleranssirajojen sisällä. Värähtelyvahti-ominaisuus testaa käyttäjän määrittelemän ajan verran pysyykö säätö asetusarvossaan jokaisella ohjelmasyklillä. Ohjelman sykliaika on 1s, joten tarkasteluja tehdään esimerkiksi 30 s värähtelyseurannan aikana n. 30 kpl.

Kuvaus säätötestauksen vaiheista:

1. Tutkitaan, onko kyseessä lämpötila- vai paine- ja ilmamääräsäätö
2. Säädön luonteesta riippuen annetaan sille sopivat säätötestausparametrit
3. Muutetaan pakko-ohjauksella säädön asetusarvoa hieman nykyisestä
4. Odotetaan säädöltä odotettavan asettumisajan verran
5. Asettumisajan päätyttyä seurataan käyttäjän määrittelemän ajan verran, pysyykö säätö toleranssirajojen sisällä uudesta asetusarvosta
6. Kirjoitetaan säätöpisteen tiedot ja säätötestin tulos raporttiin
7. Pakotetaan säädön asetusarvo alkuperäiseen arvoonsa
8. Vapautetaan säätöarvo takaisin automaattiasentoon

Itsetestaus käynnistetään käyttöliittymään tehdyllä napilla, tai automaattisesti aina käyttäjän määrittelemän pisteen vaihtaessa tilaansa. Automaattitestauksen suoritusta pystyy hallitsemaan käyttöliittymästä, tai kiinteästi ohjelmasta aseteltavalla muuttujan arvolla. Ominaisuus saadaan näin haluttaessa myös kokonaan pois käytöstä.

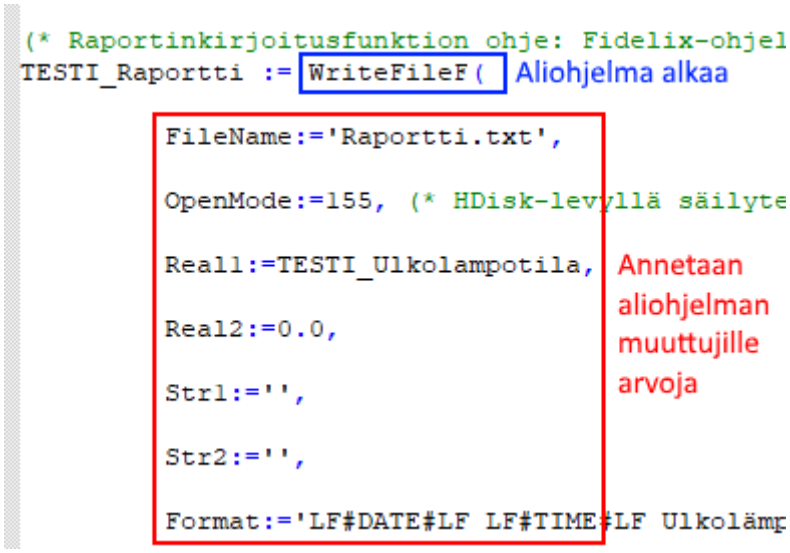
Automaattitestausta varten käyttäjä määrittelee ohjelmaan haluamansa pistetunnuksen automaattitestauksen indikointitunnukseksi. Kun tämä tunnus vaihtaa tilansa arvoon, joka on suurempi kuin 0, odotetaan käyttäjän määrittelemän alkuviiveen verran ennen testin aloittamista. Tämän jälkeen itsetestaus suoritetaan automaattisesti. Testi suoritetaan vain kerran aina indikointitunnuksen tilan nousevan reunan jälkeen.

Automaattinen itsetestaus tehtiin, jotta järjestelmä voi itse tarkkailla säännöllisesti tilaansa, esimerkiksi aina kun IV-koneen tulopuhallin käynnistyy, järjestelmä tekee testin ja kirjoittaa tilanteesta raporttitiedoston. Aloitusviive antaa järjestelmälle aikaa palata tasaiseen käyntitilanteeseen ennen testiä, mahdollisen käynnistymisen jälkeen. Automaattisella itsetestauksella voidaan myös raportoida vikatilanteita, jotka vaativat tarkempaa perehtymistä koko järjestelmän hetkelliseen tilaan. Tämä voidaan toteuttaa esim. luomalla testin indikointitunnukseksi pistetunnus, joka vaihtaa tilaansa kun jokin mittaus saavuttaa vikatilanteelle ominaisen arvon.

Itsetestauksen ohjelmoinnissa pyrittiin ottamaan huomioon kaikki ohjelman suunnittelussa hyväksi todetut käytänteet ja ohjelmointiperiaatteet. Ohjelmakoodia kommentoitiin jatkuvasti, jotta muutostyöt ja vianmääritys helpottuisivat. Ohjelmakoodi pyrittiin muodostamaan toimivaksi mahdollisimman kevyillä toimenpiteillä ja lyhyillä ehtolauseilla. Ohjelmasta muodostui tästä huolimatta hyvin laaja ja monimutkainen kokonaisuus, joten sitä ei käydä tässä kokonaisuudessaan läpi.

5.3 Raportin toteutus

Raporttitiedoston kirjoitus tehtiin Fidelix-raportinkirjoitusfunktiolla (ks. kuvio 13). Tämä ohjelmalohko toimii samalla periaatteella kuin työn toteutuksessa käytetty aliohjelmarakenne. Sitä kutsutaan toisesta ohjelmasta suorittamaan siihen määritellyt toimenpiteet. Valmis raportti ilmestyy tekstitiedostona säätimelle, samaan kansioon jossa säilytetään käyttöliittymän grafiikkatiedostoja. Sieltä raportti voidaan siirtää FTP-yhteydellä tietokoneelle, raporttia varten luotuun Word-dokumenttipohjaan.



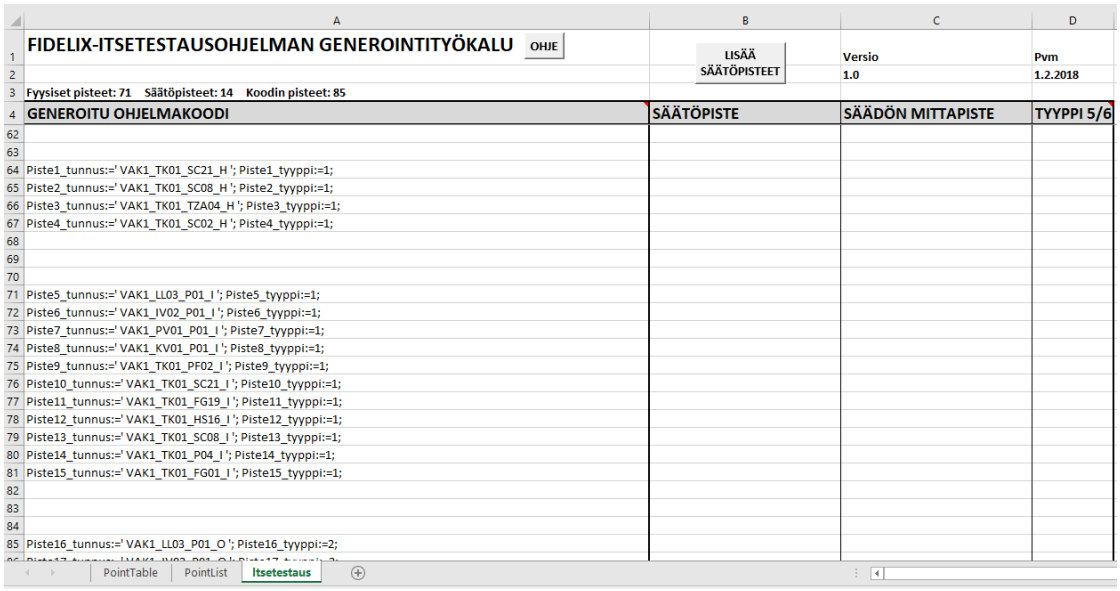
Kuvio 13. Writefile-aliohjelman määrittely

Jotta säätimen levytila ei täyttyisi raporttitekstitiedostoista, rajoitettiin raporttien samanaikainen määrä säätimellä viiteen. Kun kuudes raportti kirjoitetaan, vanhin raportti poistetaan säätimeltä automaattisesti. Raportteja voidaan säilyttää säätimen muistissa enintään 10 kerrallaan.

Raportin alkuun tulostetaan testauksen päivämäärä, kellonaika ja senhetkinen ulkolämpötila. Raporttiin kirjoitetaan pistekohtaisesti pistetunnus, pisteen arvo testaushetkellä, sekä sen mahdollinen virhetila, tai käsiasento. Säätöpisteiden kohdalla raportoidaan niiden säätöarvot ja mittaukset ennen säätötestausta, sekä säätötestin tulos.

5.4 Koodin generointi

Excel-pohjainen koodin luontityökalu tehtiin uutena Fidelix Pointgen-työkaluun lisättävänä taulukkona (ks. kuvio 14), joka kerää PointGen-taulukkoon määritellyt projektin pistetiedot ja muodostaa niiden perusteella automatisoidusti testausohjelmaan liitettävän projektikohtaisen koodin.



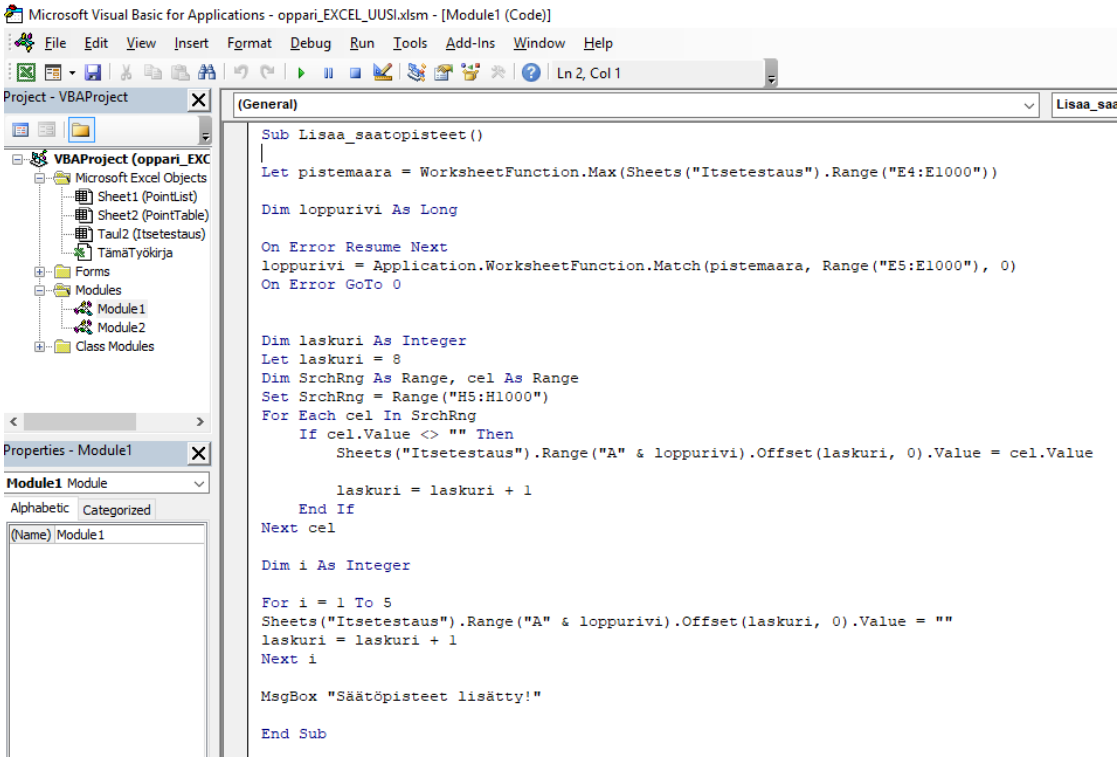
	A	B	C	D
1	FIDELIX-ITSETESTAUSOHJELMAN GENEROINTITYÖKALU OHJE	LISÄÄ SÄÄTÖPISTEET	Versio	Pvm
2			1.0	1.2.2018
3	Fyysiset pisteet: 71 Säätöpisteet: 14 Koodin pisteet: 85			
4	GENEROITU OHJELMAKOODI	SÄÄTÖPISTE	SÄÄDÖN MITTAPISTE	TYYPPI 5/6
62				
63				
64	Piste1_tunnus:=' VAK1_TK01_SC21_H '; Piste1_tyyppi:=1;			
65	Piste2_tunnus:=' VAK1_TK01_SC08_H '; Piste2_tyyppi:=1;			
66	Piste3_tunnus:=' VAK1_TK01_TZA04_H '; Piste3_tyyppi:=1;			
67	Piste4_tunnus:=' VAK1_TK01_SC02_H '; Piste4_tyyppi:=1;			
68				
69				
70				
71	Piste5_tunnus:=' VAK1_LL03_P01_I '; Piste5_tyyppi:=1;			
72	Piste6_tunnus:=' VAK1_IV02_P01_I '; Piste6_tyyppi:=1;			
73	Piste7_tunnus:=' VAK1_PV01_P01_I '; Piste7_tyyppi:=1;			
74	Piste8_tunnus:=' VAK1_KV01_P01_I '; Piste8_tyyppi:=1;			
75	Piste9_tunnus:=' VAK1_TK01_PF02_I '; Piste9_tyyppi:=1;			
76	Piste10_tunnus:=' VAK1_TK01_SC21_I '; Piste10_tyyppi:=1;			
77	Piste11_tunnus:=' VAK1_TK01_FG19_I '; Piste11_tyyppi:=1;			
78	Piste12_tunnus:=' VAK1_TK01_HS16_I '; Piste12_tyyppi:=1;			
79	Piste13_tunnus:=' VAK1_TK01_SC08_I '; Piste13_tyyppi:=1;			
80	Piste14_tunnus:=' VAK1_TK01_P04_I '; Piste14_tyyppi:=1;			
81	Piste15_tunnus:=' VAK1_TK01_FG01_I '; Piste15_tyyppi:=1;			
82				
83				
84				
85	Piste16_tunnus:=' VAK1_LL03_P01_O '; Piste16_tyyppi:=2;			

PointTable | PointList | Itsetestaus

Kuvio 14. Itsetestausohjelman luontityökalu

Säätöpisteiden nimiä ei ole yleensä projektin toteutussuunnitelmissa mukana, joten niiden nimeämistavat vaihtelevat ohjelmoijasta riippuen. Tätä varten työkaluun tehtiin erillinen säätöpisteosio, jossa ohjelman automatisoitua arviota säätöpisteen tyypistä ja sen käyttämästä mittauspisteestä voidaan vielä käsin muokata, ennen koodiin sisällyttämistä. Säätöpisteitä voidaan myös lisätä manuaalisesti, jos säätöpisteet luodaan vasta säätimen käyttöliittymästä, eikä niitä ole tehty pistelistaan.

Itsetestauskoodin luontitaulukkoon tehtiin nappi, jolla luodaan muokatuista ja lisätyistä säätöpisteistä ohjelmakoodia ja lisätään se muun ohjelmakoodin loppuun. Napin toiminnot toteutettiin Excel-makrona, eli VBA-koodina (ks. kuvio 15). VBA (Visual Basic for Applications) on Microsoftin kehittämä koodikieli.



Kuvio 15. Excel VBA-projektinäkymä

Nykyisin PointGen-työkalun käyttö on hieman vähentynyt, sillä Fidelix Editor on uutena ohjelmana korvaamassa sen toimintoja. Työn toimeksiantaja käyttää kuitenkin vielä myös PointGenia. Excel-pohjainen koodin luontiratkaisu vaatii Excel-taulukon, josta se hakee tiedot. Fidelix Editorin pistelista ei ole Excel-taulukko, mutta ohjelmalla pystyy viemään pistelistan Excel-työkirjaan. Tämän jälkeen täytyy vain muuttaa generoidun työkirjan taulukon nimi samaksi, kuin PointGen- pistelistan sisältävän taulukon ja tarkistaa pistelistan sisältävä sarake myös samaksi. Ratkaisu toimii tällöin molempien ohjelmien kanssa ja se ei jää käyttökelvottomaksi, jos tulevaisuudessa projektien pistelista tehdäänkin pelkästään Fidelix Editorin avulla.

Excel-soluviittausten ja ehtolausekkeiden avulla selvitettiin pistelistan pisteistä niiden tyypit. Ehtolausekkeiden avulla varmistutaan myös siitä, että ohjelmaan tuodaan vain fyysisten, eli kaapeleilla kytkettyjen pisteiden testaus, säätöpisteiden lisäksi. Pistelista sisältää myös hyvin paljon ohjelmallisia pisteitä, joiden tiloista ei tehdä virhearviointia. Ohjelmallisten pisteiden virheitä on miltei mahdoton tunnistaa automatisoidusti, sillä niiden toiminnot ja nimeämistavat vaihtelevat paljon.

Pisteet luokiteltiin ohjelmassa seuraaviin tyyppeihin numeroiden avulla:

1 = Digitaalinen tulo (DI), indikoinnit ja hälytykset

2 = Digitaalinen lähtö (DO), ohjaukset

3 = Analoginen tulo (AI), mittaukset

4 = Analoginen lähtö (AO), asetuspisteet

5 = Lämpötilasäätö

6 = Paine-/ilmamääräsäätö

Työkalun yläosaan tehtiin näkyville laskurit, joista käyttäjä näkee koodiin sisällytettyjen fyysisten pisteiden- ja säätöpisteiden lukumäärät. Näin voidaan halutessa vielä tarkistaa että työkalu osaa ottaa luomaansa koodin mukaan oikean määrän pisteitä, vertaamalla lukuja pistetaulukkoon, tai PointGen-työkalun sisältämään pistelaskuriin (ks. kuvio 16).

	Physical	Fictive	Total
AL	4	57	61
IND	11	0	11
DO	9	0	9
AI	33	4	37
AO	10	0	10
CTRL	0	13	13
TT	0	5	5
Total	67	79	146

Count points

Kuvio 16. PointGen-pistelaskuri

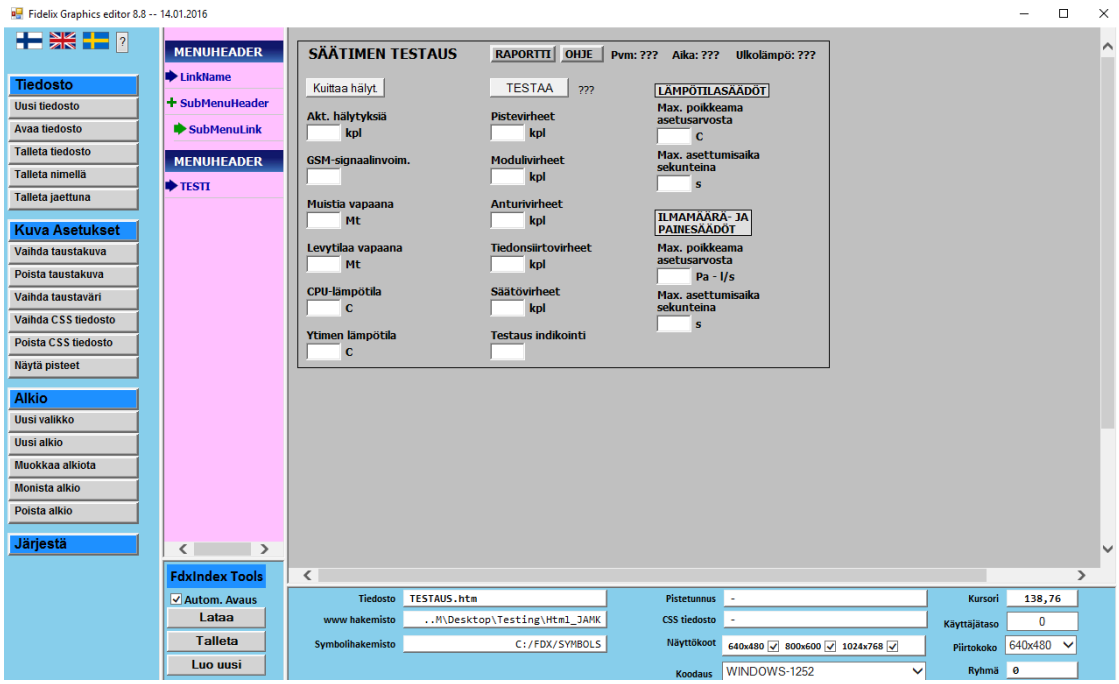
Luotu ohjelmakoodi testattiin lopuksi OpenPCS-ohjelmointityökalulla, mahdollisten kääntövirheiden varalta. Taulukkoon luotiin ohje-nappi, joka avaa käyttäjälle työkalun käyttöä helpottavan ohjeikkunan. Työkirjaan tehtiin myös testauspistetaulukko, josta voidaan kopioida säätimelle ladattavaan pistelistaan itsetestauksen toimintojen vaatimat pisteet.

5.5 Käyttöliittymän toteutus

Testausohjelmalle tehtiin käyttöliittymä, jotta testauksen tila, mahdolliset virheilmoitukset ja diagnostiikkatiedot saadaan helposti ja havainnollisesti käyttäjän nähtäville. Testauksen arvoja ja asetuksia voi myös muokata käyttöliittymän kautta. Graafista käyttöliittymää käytetään automaatiosäätimen kosketusnäytöltä, tai säätimeen yhteydessä olevan tietokoneen selaimella.

Käyttöliittymä tehtiin Fidelix HTML Editor-työkalulla (ks. kuvio 17). Käytössä on Fidelixin oma symbolikirjasto, josta löytyy kaikki kiinteistöautomaation kannalta oleellisimmat kuviot, eli IV-kanavat, LTO-kennot, venttiilit, hälytysvalot yms. Kaikki nämä ovat tallennettuina ohjelman kuvakirjastoon, josta ne voidaan liittää grafiikkaan. Halutessaan kirjastoon voi lisätä myös itsetehtyjä, tai verkosta ladattuja symboleja.

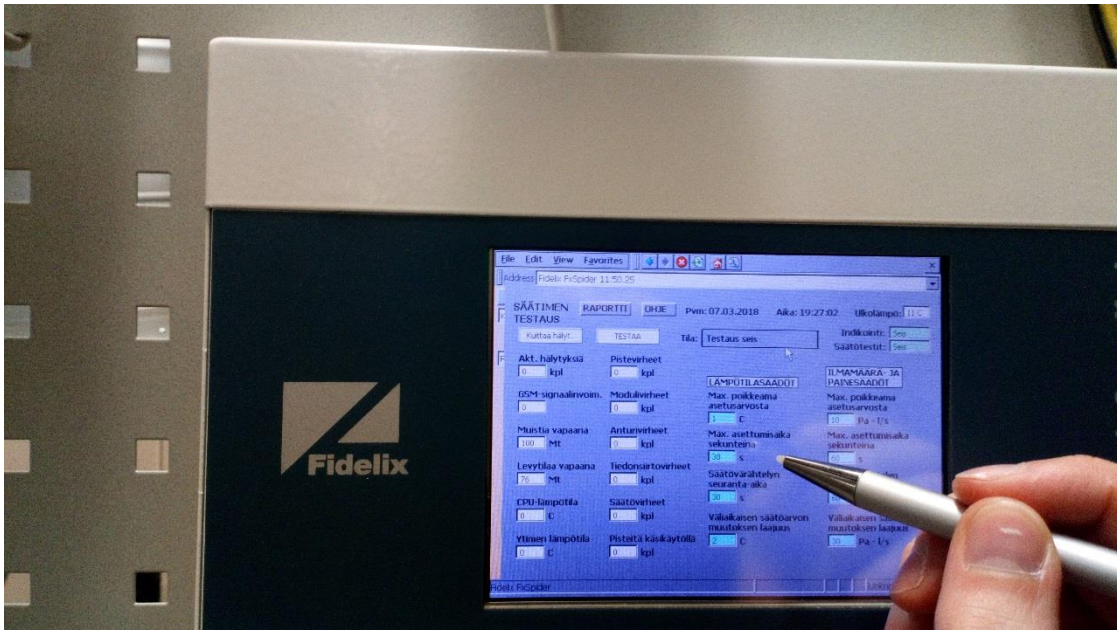
Symboleja voi käyttää passiivisina taustaelementteinä, eli järjestelmää havainnollistavina kuvioina, tai niihin voi liittää säätimelle määritellyn pistetunnuksen, jonka avulla kuvio saadaan näyttämään erilaiselta riippuen pisteen tilasta. Lisäksi ohjelmassa voi määritellä painonappeja, numero- ja teksti-kenttiä ja pylväsnäyttöjä. Grafiikkakuvan lisäksi ohjelmassa tehdään valikkokuva, jonka kautta siirrytään eri grafiikkakuvien välillä käyttöliittymässä.



Kuvio 17. Fidelix HTML-Editor kuvan luonti

Käyttöliittymä haluttiin saada mahdollisimman helppokäyttöiseksi, havainnolliseksi ja aikaa säästäväksi, joten se on tehty yhteen grafiikkakuvaan. Kuvasta tuli hieman tiiviin oloinen, mutta näin mahdollisimman paljon hyödyllistä tietoa saadaan yhteen näkymään, ilman tarvetta liikkua kuvien välillä. Käyttönäkymän kokoa rajoittavana tekijänä on säätimien kosketusnäytön koko. Vaikka kuvissa voi liikkua selainpalkkien avulla, näkymän selailu vähentää kokonaisuuden havainnollisuutta.

Käyttöliittymä (ks. kuvio 18) pidettiin visuaalisesti yksinkertaisena, jotta sen tulkitseminen olisi mahdollisimman helppoa. Samalla se pitää kuitenkin sisällään kaikki diagnostiikkatiedot ja asetukset, joita pidettiin tärkeimpinä säätimen toiminnan ja testauksen kannalta. Grafiikkakuvassa ei käytetty symbolikirjastoa, koska kuvassa käytetyt symbolit pitäisi tällöin aina muistaa ladata säätimen muistiin, jotta ne näkyisivät myös säätimen näytöllä.

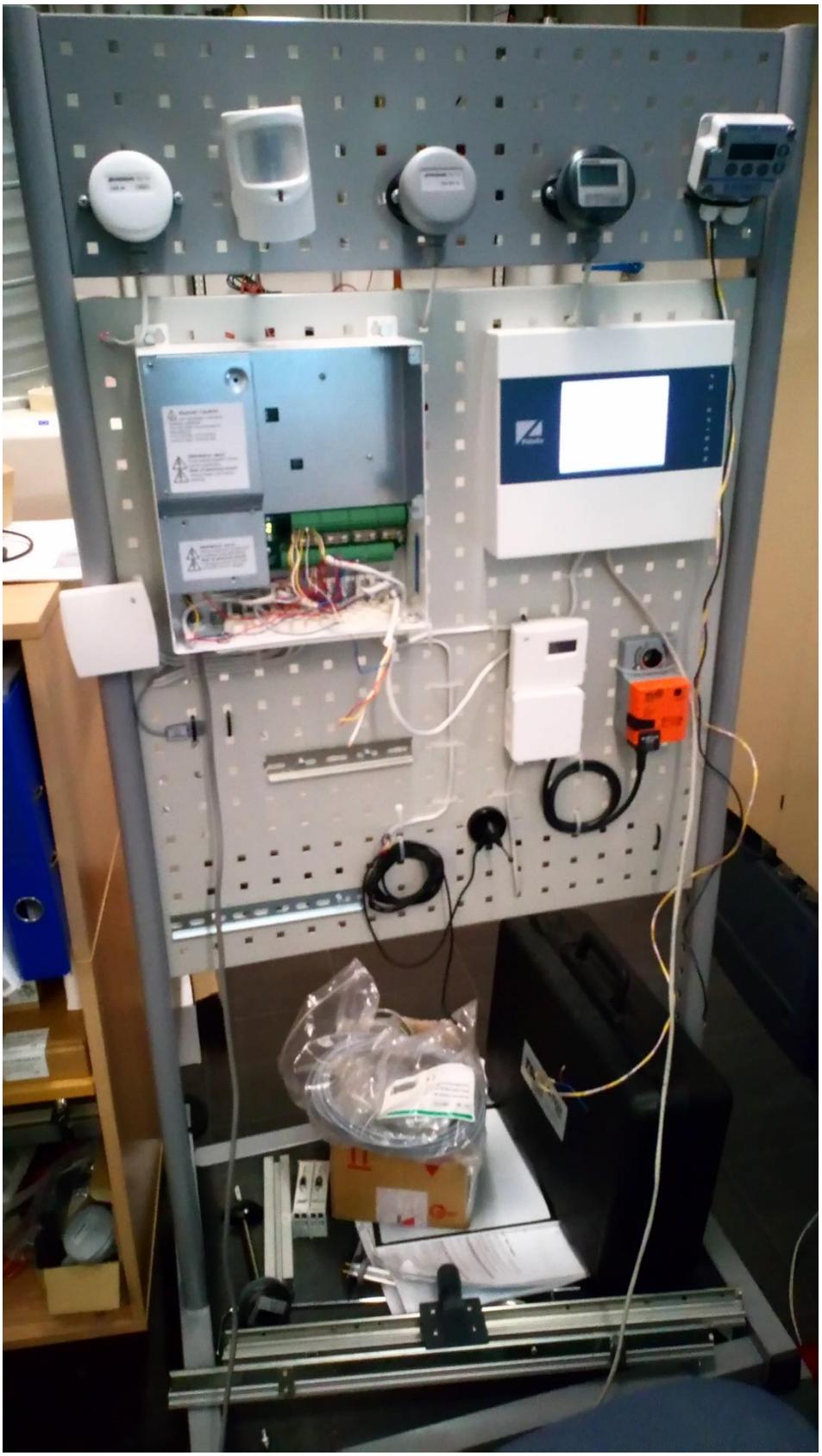


Kuvio 18. Käyttöliittymä säätimen näytöllä

Käyttöliittymä olisi voitu jakaa eri kuviin, joiden välillä siirryttäisiin kuviin tehtävistä napeista. Tällöin kuvia pitäisi ladata säätimelle monta ja kokonaisuutta ei voitaisi tarkastella kerralla. Toisaalta jos jatkossa halutaan vielä enemmän tietoja päänäkymään, voidaan luoda myös toinen sivu, jossa muokataan vain säätötestauksiin liittyviä asetuksia.

5.6 Ohjelman testaus

Ohjelma testattiin lopuksi Jyväskylän Ammattikorkeakoulun opetuskäytössä olevalla automaatiojärjestelmällä (ks. kuvio 19). Laitteiston on toimittanut työn toimeksiantaja Lvi-Elektro Oy ja siihen kuului Fx-Spider -säädin, kanavalämpöanturi, huonelämpöanturi, hiilidioksidilähetin, kosteuslähetin, ulkovaloisuusanturi, paine-erolähetin, peltimoottori ja liiketunnistin. Järjestelmään oli kuulunut GSM-modeemi, mutta se oli poistettu kokoonpanosta, antennia lukuun ottamatta.



Kuvio 19. Fidelix-Spider -säädin ja testauslaitteisto

Laitteistolla voitiin testata kaikki pisteiden arvoihin ja perustoimintaan liittyvät testit, lukuun ottamatta tiedonsiirtovirheitä, koska laitteistoon ei ollut yhdistetty toista säädintä. Laitteiston ainoa toimilaite oli peltimoottori, jolla ei voinut säätää mitään järjestelmällä mitattavaa suuretta, joten säätöjen testaus täytyi tehdä simuloimalla säätötilannetta. Mittauksien arvoja ajettiin siis käsi-tilassa eri arvoihin säätötestauksien aikana.

Lopputestauksia tehtiin yhteensä 10 kappaletta, joissa käytiin läpi kaikki ohjelman erottelemaa kuusi eri pistetyyppiä eri tiloissa. Kaikkia pistetyyppejä ei järjestelmässä ollut saatavilla, mutta ne luotiin testausta varten ohjelmallisina pisteinä
Lopuksi ohjelman toimintaa testattiin ja arvioitiin vielä Lvi-Elektron ohjelmointiasiantuntijan kanssa. Saadun palautteen perusteella työhön tehtiin vielä ohjelmallinen hälytyspiste. Yhdistelmähälytys annetaan, jos jonkin testin virhearvolaskuri aktivoituu (arvo on suurempi kuin nolla), säätimen lämpötila kohoaa liian suureksi, tai vapaan muistin määrä on liian vähäinen. Lisäksi pohdittiin työn mahdollista jatkokehitystä.

6 Työn tulokset

6.1 Testausohjelman edut

Koska ohjelman tarkoitus on säästää aikaa, mitattiin aika joka kuluu testausohjelmaa projektiin liitettäessä. Kokeneelta käyttäjältä testausohjelman kaikkien osien projektiin liittämiseen ja alustamiseen tarvittavaksi ajaksi mitattiin sekuntikellolla noin kolme minuuttia. Automaattitestauksen suorituksen kesto riippuu säätötesteille käyttäjän määrittelemistä ajoista. Ilman säätötestejä ja testauksen aloitusviivettä esimerkiksi 10 pisteen testaus kestää ohjelmalla vain muutamia sekunteja.

Työn tuomaa aikasäästöä arvioitiin vertaamalla itsetestausaikaa ja muutaman referenssikohteen manuaaliseen testaukseen vaadittavaa aikaa. Referenssikohteita oli kaksi. Toisessa kohdejärjestelmässä oli 275 fyysistä pistettä ja 25 säätöpistettä, toisessa oli 135 fyysistä pistettä ja 8 säätöpistettä. Näiden kohteiden manuaaliseen testaamiseen menneistä ajoista otettiin keskiarvot, joita käytettiin manuaalisen testausajan arvioimiseen.

Järjestelmäraportin muodostaminen on manuaalisesti suhteellisen helppoa, jos kopioidaan pikanäppäinyhdistelmillä pistelistat tekstinä säätimeltä ja liitetään ne Excel-taulukkoon. Tällä tavoin myöskään järjestelmän koolla ei ole juurikaan merkitystä, koska kopioinnin nopeus ei muutu. Huomionarvoista kuitenkin on, että manuaalinen testaus työllistää koko testausajan käyttäjää. Automaattisen testauksen ajan käyttäjä voi tarvittaessa tehdä muita töitä tai järjestelmän silmämääräistä tarkastusta. Lisäksi manuaalisen testauksen nopeus vaihtelee käyttäjän tehokkuuden ja työmenetelmien mukaan.

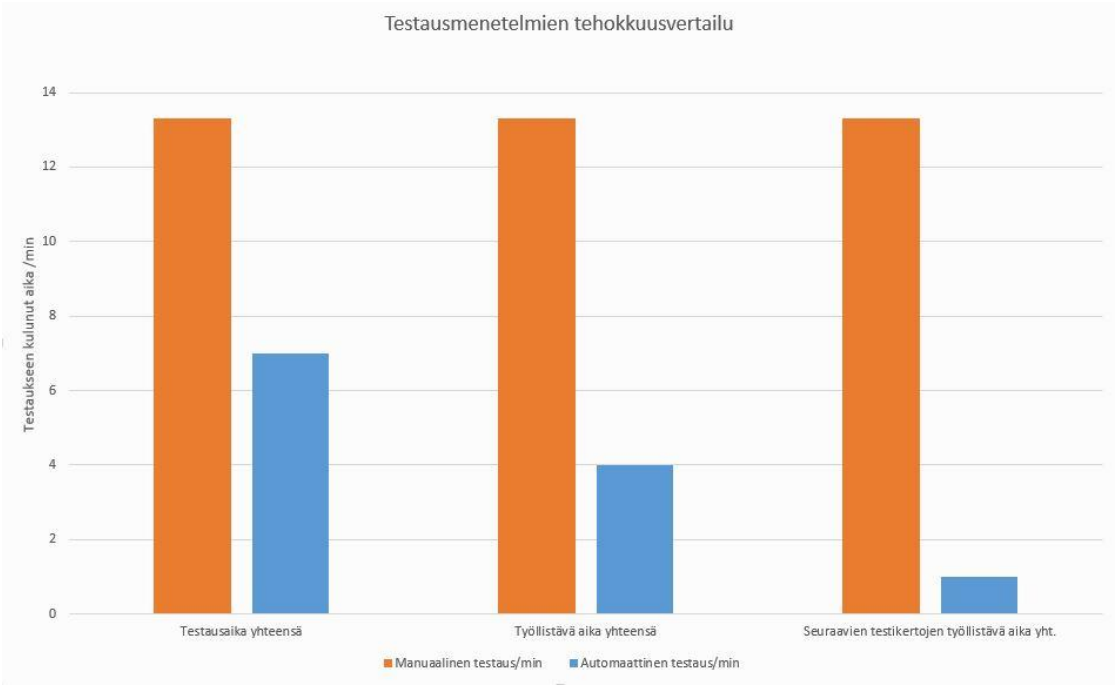
Säätöjen tilat voidaan myös tarkastaa manuaalisesti päällisin puolin, katsomalla että säädettävät arvot ovat lähellä asetusarvoja. Jos kuitenkin halutaan tarkastaa säätöjen asettumisajat ja värähtely, aikaa kuluu huomattavasti enemmän, koska asetusarvot täytyy muuttaa ja palauttaa säätöpistekohtaisesti. Lisäksi testaajan täytyy odottaa säätöjen asettumista ja yrittää seurata silmämääräisesti ehkä jopa kymmeniä säätöjä värähtelyn varalta. Alla olevassa kuviossa 20 kuvataan säätöpistetestauksiin kuluvaa aikaa. Tämä arvio perustuu kokeneen käyttäjä työtehoon. Säätöarvojen edestakaiseen muutokseen menee pistettä kohden noin 20 sekuntia ja säätöjen värähtelyn seurantaan on varattu kaikilla pistemäärillä vain minuutti. Todellisuudessa aikaa voisi kulua vielä enemmän pistemäärän kasvaessa koska säätimen näytöltä pystyy seuraamaan vain n. 10 säätöpistettä kerrallaan. Kaikki tähän käytetty aika on säästettävissä itsetestausohjelmalla.

Kuvio 20. Säätöjen manuaaliseen testaukseen kuluva aika

Taulukosta 1 voidaan huomata, että aikasäästöä on saatavilla jo ensimmäisellä testikerralla, vaikkei säätötestejä tehtäisi ja käyttäjä olisi toimettomana koko automaattitestauksen ajan. Suurempiin aikasäästöihin (jopa yli 90 %) päästään, jos ohjelma jätetään säätimelle ja sitä ei tarvitse seuraavilla testikerroilla alustaa. Aikasäästö muodostuu monesta muuttujasta ja sen laajuus riippuu raportilta vaadittavista ominaisuuksista, testausohjelman käyttötarkoituksesta, testauskerroista ja käyttäjän työtehosta. Taulukon 1 tiedoista koostettu pylväskaavio (ks. kuvio 21) havainnollistaa molempien testaustapojen ajallista tehokkuutta.

Taulukko 1. Testausaikavertailu

Toimenpide	Manuaalinen testaus/min	Automaattinen testaus/min
Pisteet raportoitu ja itsediagnostiikka tarkastettu	4	0,5
Säätötestien seuranta-aika, riippuu käyttäjästä	3	3
Säätötestien asetusarvomuutokset 10 säätöpisteen kanssa	3,3	0
Raportin siistiminen	3	0,5
Ohjelman käyttöönotto	0	3
Testausaika yhteensä	**13,3**	**7**
Työllistävä aika yhteensä	**13,3**	**4**
Seuraavien testikertojen työllistävä aika yht.	**13,3**	**1**



Kuvio 21. Testausmenetelmien ajallinen tehokkuusvertailu

Jos oletetaan että testausohjelmaa tultaisiin käyttämään 30 järjestelmässä ja aikasäästö on ensimmäisellä testikerralla 70 %, voitaisiin ohjelmalla säästää vähintään 4,5 tuntia työaikaa. Jos kaikki näistä järjestelmistä haluttaisiin testata säännöllisesti, yhteensä 10 kertaa, olisi säästö jo vähintään 50 tuntia. Testausohjelmasta on siis hyvin huomattavaa etua erityisesti suurien järjestelmien säännöllisessä tarkkailussa.

6.2 Logiikkaohjelma

Työn keskeisimpiä tuloksia olivat Function-Block, eli ohjelmalohko, johon luetaan Excel-taulukosta ohjelmaan kopioitu koodi, sekä itse pääohjelma (ks. liitteet 1-2). Itsetestausohjelma suoritti siihen ohjelmoidut toiminnot sujuvasti lopputesteissä. Määritellyt pisteet testattiin onnistuneesti ja havaituista virheistä ilmoitettiin käyttäjälle laskureilla ja hälytyksellä. Säätöpisteiden testaus onnistui säätötilannetta simuloivissa testeissä. Kaikki erityyppiset virhelaskurit antoivat oikean määrän virheitä ja kaikki virheet löytyivät kirjattuna raportista. Raporttitekstitiedostojen alkuun tulostettiin testaushetken päivämäärä, kellonaika ja ulkolämpötila.

Testauksen suorituksen hallinta toimi testauksissa hyvin. Ohjelman aktivointinappi toimi muuttamalla sen tila manuaalisesti PÄÄLLE-arvoon. Testin lopuksi testaus pakotti sen takaisin POIS-asentoon ja palautti napin automaattitilaan. Ohjelman automaattinen testaustoiminto todettiin myös toimivaksi, jolloin itsetestauksen suoritus voidaan automatisoida täysin. Testauksen manuaalinen keskeytys toimii myös, mutta sitä ei suositella säätötestien aikana.

Fyysisten pistetestien lisäksi käyttäjä saa valita ohjelmasta, tai myöhemmin käyttöliittymästä, testataanko myös säätöpisteet. Säätöpistetestauksen pois jättäminen voi säästää säätimen prosessointiresursseja myöhemmin, kun säädöt on jo viritetty ja todettu toimiviksi. Tällöin ei kuitenkaan saada testauksen kautta tietoa esim. laiterikosta, tai olosuhdemuutoksesta, jonka vuoksi jokin säätö lakkaa toimimasta.

6.3 Koodinluontityökalu

Koodin luontityökalu oli tärkeä saavutus työn jatkohyödyn kannalta. Excel-työkalu kopioi pistelistasta kaikki fyysiset pisteet ja tunnisti niiden tyypit onnistuneesti lopputestien aikana. Lopputestejä työkalulle tehtiin yhteensä viisi ja niissä käytiin läpi tavallisimpia pistetyyppejä ja niiden eri nimeämistapoja. Työkalu erotti myös ohjelmalliset pisteet, jotka se jätti onnistuneesti pois testauskoodista. Testauspohjana käytettiin Fidelix-esimerkkipistelistaa, joka sisälsi 67 fyysistä- ja 79 ohjelmallista pistettä (ks. kuvio 22). Kaikkien säätöpisteiden tyyppejä ei voida varmasti tunnistaa, mutta tätä varten käyttäjä pystyy muokkaamaan säätöpistetietoja työkalussa.

1	Version	Delete points	Count points		Add Modules	
2	123					
3	SaveAndExit	Delete modules	Zoom all			
4						
68	VAK1_TK01_SC02_H	Mallihuone LTO-Kiekko	3	0	0	1
69	IND					
70	Name	Text	Port	Module	Point	OnDela
71	Write DefaultValues into this line		3	0	0	0
72	VAK1_LL03_P01_I	IV-Verkosto Pumppu	3	0	0	0
73	VAK1_IV02_P01_I	IV-Verkosto Pumppu	3	0	0	0
74	VAK1_PV01_P01_I	Patteriverkosto Pumppu	3	0	0	0
75	VAK1_KV01_P01_I	Käyttövesiverk. pumppu	3	0	0	0
76	VAK1_TK01_PF02_I	Mallihuone PF01 Taaj.muuttaja	3	0	0	0
77	VAK1_TK01_SC21_I	Mallihuone PF01 Taaj.muuttaja	3	0	0	0
78	VAK1_TK01_FG19_I	Mallihuone Poistoilma	3	0	0	0
79	VAK1_TK01_HS16_I	Mallihuone Lisäaikakytkin	3	0	0	0
80	VAK1_TK01_SC08_I	Mallihuone TF01 Taaj.muuttaja	3	0	0	0
81	VAK1_TK01_P04_I	Mallihuone LP-Pumppu	3	0	0	0
82	VAK1_TK01_FG01_I	Mallihuone Raitisilmapelti	3	0	0	0
83	DO					
84	Name	Text	Port	Module	Point	OnDela
85	Write DefaultValues into this line		3	0	0	0
86	VAK1_LL03_P01_O	IV-Verkosto Pumppu	3	0	0	0
87	VAK1_IV02_P01_O	IV-Verkosto Pumppu	3	0	0	0
88	VAK1_PV01_P01_O	Patteriverkosto Pumppu	3	0	0	0
89	VAK1_TK01_PF02_O	Mallihuone PF01 Taaj.muuttaja	3	0	0	0
90	VAK1_TK01_SC21_O	Mallihuone PF01 Taaj.muuttaja	3	0	0	0
91	VAK1_TK01_FG19_O	Mallihuone Poistoilma	3	0	0	0
92	VAK1_TK01_HS16_O	Mallihuone Lisäaikakytkin	3	0	0	0
93	VAK1_TK01_SC08_O	Mallihuone TF01 Taaj.muuttaja	3	0	0	0
94	VAK1_TK01_FG01_O	Mallihuone Raitisilmapelti	3	0	0	0
95	AI					
96	Name	Text	Port	Module	Point	Analog
97	Write DefaultValues into this line		3	0	0	1
98	VAK1_KV01_FQ03_M	Käyttövesiverk. Kylmävesimäärä	3	0	0	1
99	VAK1_TK01_FE21_M	Mallihuone PF01 ilmamäärä	3	0	0	1
100	VAK1_TK01_FE08_M	Mallihuone TF01 ilmamäärä	3	0	0	1
101	VAK1_TK01_SC01_M	Mallihuone Tuloilmakone	3	0	0	1

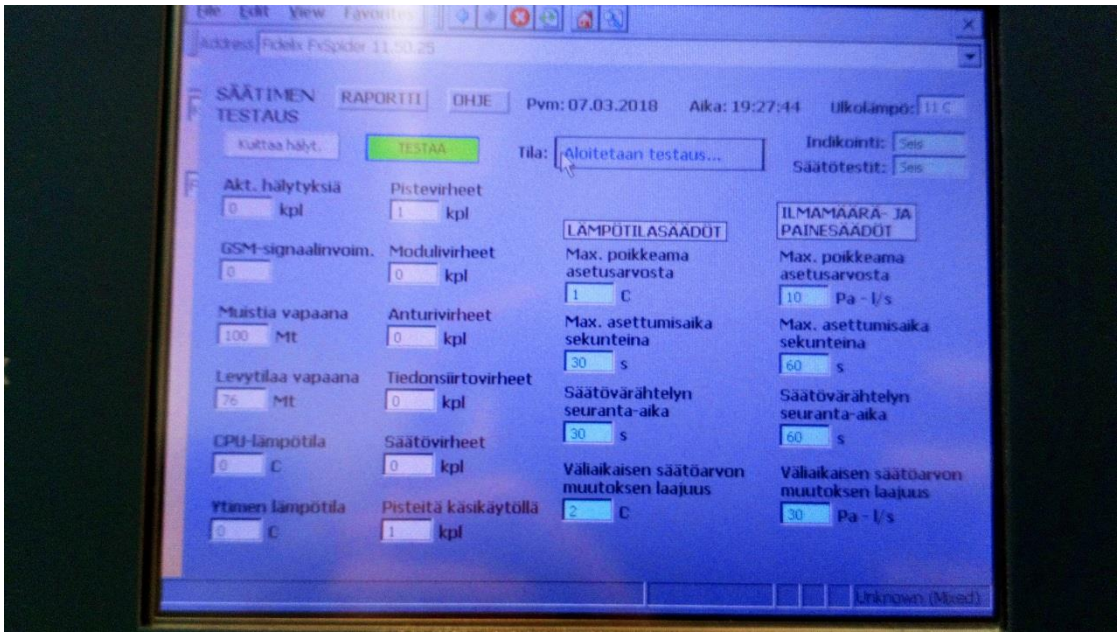
Kuvio 22. Osa Fidelix-esimerkkiprojektin pistelistasta

Työkalun toiminta testattiin myös Fidelix Editorista tuodun pistelistan kanssa. Pisteiden tunnistusvarmuus arvioitiin olevan erityisen hyvä. Tunnistus perustuu enimmäkseen Fidelix-ohjelmien käyttämiin pistetunnuksien loppupäätteisiin, jotka ovat Fidelix-ohjelmissa pyritty vakioimaan. Muutamat mahdollisesti testiin mukaan tulevat ohjelmalliset pisteet eivät haittaisi testausohjelman toimintaa, mutta ne tuottaisivat mahdollisesti käyttöliittymään ja raporttiin aiheettomia virhetietoja.

6.4 Käyttöliittymä

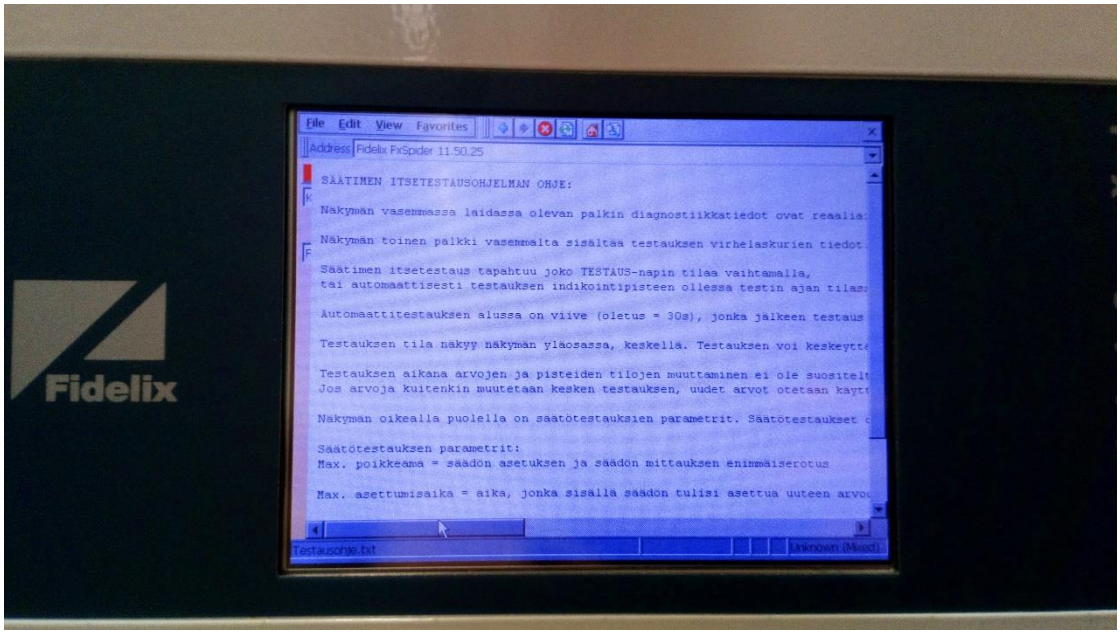
Käyttöliittymä (ks. kuvio 23) mahdollisti testauksen helpon suorituksen, sekä seurannan ja siinä näytettävät tiedot olivat testausten jälkeen paikkaansa pitäviä. Käyttöliittymässä näytetään reaaliaikaisesti säätimen diagnostiikkatiedot, mutta pistevirhei-

den lukemat ovat aina viimeisimmän testauksen tuloksia. Piste- ja säätövirheiden reaaliaikainen tarkkailu ei olisi kovinkaan hyödyllistä säätimen normaalikäytön kannalta ja se voisi rasittaa säädintä turhaan, sekä aiheuttaa turhia hälytyksiä hetkellisten vikatilojen seurauksena.



Kuvio 23. Lopullinen käyttöliittymä säätimen näytöllä

Käyttöliittymän yläosassa on otsikon jälkeen raportti-nappi, josta painamalla avautuu itsetestauksen viimeisin raportti. Tämän napin vieressä on ohje-nappi, joka avaa ohjetiedoston (ks. kuvio 24) muistikortilta. Ohjetiedosto tehtiin helpottamaan grafiikan käyttöä. Näiden nappien alapuolella on testi-nappi, josta saadaan käynnistettyä ja pysäytettyä testi. Tämän napin vieressä oikealla tehtiin testauksen tilakenttä, jossa ilmoitetaan testauksen etenemisestä ja sen tilasta. Automaattitestauksen indikoinnin arvo tuotiin grafiikkaan vain ohjelman testaamista varten ja se poistetaan projekteissa käytettävästä versiosta.



Kuvio 24. Käyttöliittymän ohje säätimen näytöllä

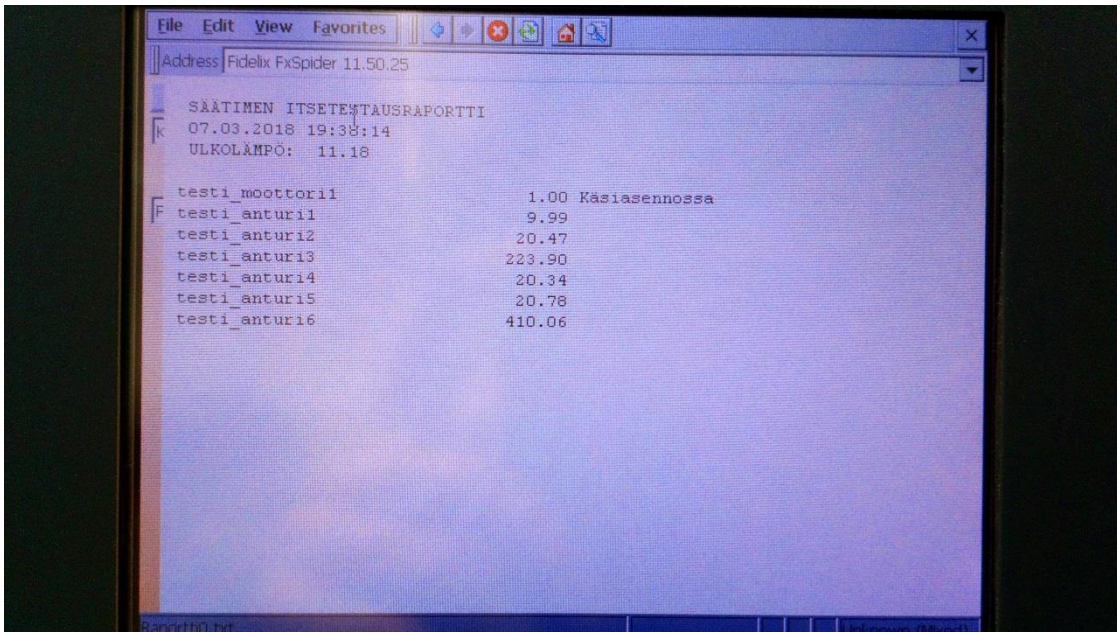
Nappien jälkeen ylhäällä nähdään säätimen päivämäärä ja kellonaika, sekä ulkolämpötila, joka luetaan ulkoanturilta. Nämä tiedot helpottavat tilanteen dokumentointia ja ulkolämpötila auttaa ymmärtämään ulkolämpötilasta riippuvien säätöjen toimintaa.

Näytön vasemmassa laidassa valikkopalkin vieressä oleva palkki on täynnä säätimen diagnostiikkatietoja. Nämä tiedot todettiin tärkeiksi säätimen ja järjestelmän toimivuuden kannalta. Näiden tietojen vieressä oleva seuraava tietopalkki sisältää testauksen fyysisten pisteiden testitulokset. Nämä kentät näyttävät viime testauskerran tilan. Numerokentät nollataan aina uuden testin alussa.

Oikealla puolella kuvaa on säätöjen testaukseen liittyvien muuttujien määrittelyyn tarkoitetut kentät. Näiden yläpuolella on säätötestauksen aktivointi-nappi, josta saadaan säätötestaukset pois tai mukaan testin suoritukseen. Käyttäjä määrittelee kenttiin haluamansa säätötestauksen käyttämät arvot. Numerokenttien eri värit kertovat niiden sisällön tyypin. Toimeksiantaja on käyttänyt Fidelix-projekteissa mm. seuraavia periaatteita: valkoinen = mittaus, vaaleansininen = asetusarvo, vaaleanvihreä = ohjaus, vaaleankeltainen = säätöarvo.

6.5 Raporttitiedosto

Raporttitiedosto (ks. kuvio 25) sisälsi testaustiedon tarkempaan analysointiin vaadittavat ominaisuudet. Raporttitiedostosta käy ilmi pistekohtaisesti kaikki piste- ja säätövirheet, sekä käsikäytöllä olevat pisteet. Raporttiin kirjattiin myös kaikki normaalitilassa olevat pisteet, jotta järjestelmän toimintaa voidaan arvioida myös lievempien virheiden varalta.



Kuvio 25. Raporttitiedosto säätimen näytöllä

Raportista nähdään kaikkien testattujen pisteiden arvot ja -tunnukset riveittäin. Tästä saatiin idea raportin ”siistimiseen” esim. järjestelmän käyttöönottotarkastusta varten. Raportti voidaan kopioida tekstitiedostosta sille luotuun Excel -pohjaan, joka siistii tekstin kirjoitusasun ja muotoilun. Tällä tavalla saadaan helposti arkistoitavaksi soveltuva järjestelmän tilakatsaus.

7 Pohdinta

7.1 Työn arviointi

Työn tulokset vastasivat toimeksiantajan tarpeisiin ja niihin oltiin yhdessä tyytyväisiä. Testausohjelman arvioidaan tuovan tulevaisuudessa säästöä käyttöönoton testausta

nopeuttavana tekijänä. Lisäksi ohjelma voidaan mahdollisesti jättää säätimeen ja sen avulla voidaan automatisoidusti monitoroida järjestelmän tilaa. Testausohjelman havaitessa virheen, käyttäjää tiedotetaan yhdistetyllä hälytyksellä.

Vian ilmaantuessa, myös kyseiseen kohteeseen perehtymättömälle henkilölle pitäisi olla suhteellisen helppoa paikantaa vikoja ohjelman avulla ja tarkentaa näin vaadittavia toimenpiteitä. Työn tulokset siis odotettavasti madaltavat käyttäjän osaamistarvetta automaatiojärjestelmän käyttöönottoon ja vianetsintään liittyen.

Yksi ohjelman tavoitteista oli myös yrittää itsenäisesti virittää säätöpisteitä, mutta Fidelix ei ole luonut ohjelmalohkoa, jonka kautta toiminto olisi ollut mahdollista toteuttaa. Tämä oli pieni takaisku testausohjelman aikaa- ja vaivaa säästävän tarkoituksen kannalta. Tästä huolimatta säätövirheiden tunnistus ja raporttitiedoston tuoma dokumentoitava tilannetieto todettiin selviksi ansioiksi.

Työn tuoman aikasäästön arvioitiin kasvavan järjestelmän koon ja testikertojen myötä. Aikasäästö alkaa kun projektiin liittämiseen menevä aika, eli n. kolme minuuttia, on säästetty ohjelman tuomilla eduilla. Tämä aika saavutetaan pienilläkin järjestelmillä jo ensimmäisellä testikerralla, jos testauksesta halutaan raportti, tai järjestelmälle tehdään perusteelliset säätötestit. Jos vain nopea silmämääräinen tutkailu on tarpeen, ei suuria aikasäästöä saavuteta pienemmillä, alle 100 pisteen järjestelmillä. Suuren aikasäästön saavuttaminen voi olla vaikeaa myös, jos käyttäjä on kokematon ohjelman käytössä.

Toisaalta ohjelma tarjoaa paljon muutakin kuin suoraa testausaikasäästöä. Tiivistetty järjestelmän tilannenäkymä helpottaa virhediagnosointia ja järjestelmän hallintaa. Säätötestejä voidaan käyttää säätöjen virityksessä apuna. Automaattitestauksen kanssa ohjelmaa voidaan käyttää järjestelmän tilan valvontaan ja arkistoidut raportit antavat hyvän tilannekuvan järjestelmän aikaisemmasta tilasta.

Työn lopullinen testaus ei ollut kovin laaja, koska sitä rajoitti testausjärjestelmän pieni koko. Tästä huolimatta lopputestauksessa saatiin todettua testausohjelman perusominaisuuksien ja säätöpistetestausten toiminta. Valitettavasti osa säätimen

omista diagnostiikkatiedoista on luettavissa vain FX2030-säätimiltä, joten esimerkiksi testauslaitteiston FX-Spider -säätimen lämpötilatietoja ei saatu näkyviin. Testauslaitteisto ei sisältänyt myöskään GSM-moduulia, jonka signaalinvahvuus näkyisi grafiikassa. Tiedonsiirtovirheitä ei myöskään päästy testaamaan, koska säädin ei kommunikoi toisten säätimien kanssa verkon ylitse.

Kaikkia ohjelmaan tehtyjä toimintoja ei siis saatu testattua, mutta koska nämä toiminnot perustuvat samoihin aliohjelmiin ja periaatteisiin kuin muutkin ohjelman toiminnot, voidaan myös niiden olettaa toimivan. Lisäksi testaamattomien toimintojen osuudet ohjelmasta ovat pieniä ja helposti hallittavia, joten mahdollisten virheiden korjaaminen onnistuisi uuden järjestelmän testauksen yhteydessä.

Työ oli toteutukseltaan hyvin itsenäinen ja se tehtiin harjoittelun ohella noin kahdeksan kuukauden aikana. Osaaminen kehittyi monipuolisesti ja nopeasti, kun yrityksen toteutuksessa olevissa projekteissa pääsi tekemään ohjelmointi- ja kenttätöitä. Toimeksiantajan hyvä määrittelytyö auttoi koko työn ajan ja työstä saatiin hyvät tulokset.

7.2 Kehitysehdotukset

Työn edetessä sen toteutukseen sisältyi monia omia-, sekä toimeksiantajan oivalluksia, jotka vaikuttivat lopputulokseen positiivisesti. Kaiken hienosäätäminen ei ollut kuitenkaan ajallisista syistä mahdollista ja ohjelmaan voidaan vielä jatkossa tehdä muutamia parannuksia.

Lopputestaukset eivät kattaneet aivan kaikkia testausohjelman sisältämiä toimintoja. Tulevaisuudessa myös testaamatta jääneet toiminnot tarkastetaan niihin soveltuvan järjestelmän kanssa ja ohjelmaan jääneet mahdolliset virheet korjataan.

Työn ohjelmakoodista muodostui melko pitkä ja iso kokonaisuus, joka saattaa rasittaa säädintä, varsinkin suuria järjestelmiä testatessa. Ohjelmaa ei kuitenkaan testattu isoilla pistekuormilla, testijärjestelmän pienen koon ja aikarajoitteiden vuoksi. Jatkossa ohjelmaa tullaan kokeilemaan asteittain suurempien järjestelmien kanssa.

Testausohjelmasta tehtiin kevyen testiversion lisäksi vain yksi tulosversio, joka pystyy testaamaan ja käsittelemään 100 pistettä. Tulevaisuudessa ohjelmasta on tarkoitus tehdä myös suurempia versioita, jotka voisivat käsitellä 500-, tai jopa 1000 pistettä. Näistä versioista voisi valita sopivimman käyttökohteen laajuuden mukaisesti. Fx-2030 -säätimen maksimi-pistemäärä on 2000, mutta näistä jopa yli puolet voi olla projektista riippuen ohjelmallisia pisteitä, joita ei voida testata tällä testausohjelmalla.

Ohjelmasta voitaisiin tehdä myös erilaisilla ominaisuuksilla varustettuja versioita eri käyttökohteisiin paremmin soveltuviksi. Ohjelmaa on mahdollista keventää ja ominaisuuksia voidaan karsia isoja järjestelmiä varten. Käyttöliittymästä hallittavia asioita voidaan vähentää huomattavasti, jos asetukset säädetään sopiviksi jo ohjelmakoodissa. Jättämällä hallittavuutta käyttöliittymästä pois ja lisäämällä ohjelmaan hyviksi todettuja vakioarvoja, voitaisiin myös erilaisia säätötyyppejä lisätä helpommin. Tällöin säätötestien tarkkuus parantuisi. Toisaalta myös ominaisuuksien lisääminen ja pistetyyppien tarkempi erittely voisi lisätä raportin tietoarvoa. Testausohjelmaan jätettiin jatkokehitystä helpottavia tietoja ja kehitysvaroja, joita ei vielä otettu käyttöön tässä ohjelmaversiossa.

Lähteet

E. A. Parr. 2003. Programmable Controllers: An Engineer's Guide. 3. p. Oxford: Elsevier

Fidelix Oy. 2017. Fidelix-ohjelmointimanuaali. Viitattu 12.8.2017

Fidelix Oy. 2017. Tietoa yrityksestä. Fidelix Oy:n internet-sivut. Viitattu 17.7.2017. http://www.fidelix.fi

infoteam Software AG. Tietoa OpenPCS –ohjelmistosta. Infoteam Software AG:n internet sivut. Viitattu 24.9.2017. http://www.infoteam.de

Korhonen, Vili. 2013. Fidelix-ohjelmointiopas. Opinnäytetyö, AMK. Jyväskylän Ammattikorkeakoulu, Tekniikan- ja liikenteen ala, automaatiotekniikan koulutusohjelma. Viitattu 12.10.2017

LVI-Elektro Oy. 2017. Tietoa yrityksestä. LVI-Elektro Oy:n internet-sivut. Viitattu 17.7.2017. http://lvielektro.com

Pentti, H., Juhana, M., Veijo, P., Antti, S., Toivo, S., Börje, S., Arto, S., Tapani, S., Jukka, S. 2012. Rakennusautomaatiojärjestelmät. 3. p. Espoo: Sähköinfo

Phil, Z. 2017. The Ultimate Guide to Building Automation Systems. Artikkeli Building Automation Monthlyn internet-sivuilla. Viitattu 12.10.2017. http://buildingautomationmonthly.com

Värjä, P. & Mikkola J. 1999. Uusi kiinteistöautomaatio. 10. p. Koria: Cadnet.

Liitteet

Liite 1. Itsetestausohjelman määrittelyosio

```
(* FIDELIX SÄÄTIMEN ITSETESTAUSOHJELMA v1.0 *)

(* Huom. CPU:n itsediagnostiikka vaati väh. 11.12 säädinversion, jännitetieto vaati FX2030 säätimen *)

ITSETESTAUS
(

(***************************** YLEISASETUKSET *****************************)

(* ASETA ALLE ULKOLÄMPÖMITTAUKSEN PISTETUNNUS (OPTIO, NÄYTTÄÄ ULKOLÄMPÖTILAN RAPORTIN ALUSSA)*)
TESTI_Ulkolampotila_tunnus := '',

(* ASETA ALLE AUTOMAATTITESTAUKSEN INDIKOINNIN PISTETUNNUS (OPTIO, TESTAUS VOIDAAN MYÖS MANUAALISESTI KÄYNNISTÄÄ AKTIVOINTIPISTEESTÄ)*)
TESTI_Indikointipiste_tunnus := 'TESTI_FI',

(* ASETA ALLE AUTOMAATTITESTAUKSEN ALOITUSVIIVE TASASEKUNTEINA (OPTIO, IV-KONEEN KÄYNNISTYMISVAIHETTA VARTEN)*)
TESTI_Aloitusviive_ohjelma := 5,


(* ASETUKSET JOITA MUUTETAAN JOS MYÖS SÄÄTÖPISTEET TESTATAAN *)

(* ASETA ALLE SÄÄTÖTESTAUSTEN HALUTTU SUORITUSTILA: 0=EI IKINÄ SUORITETA, 1=SUORITETAAN AINA, 2=HALLITAAN KÄYTTÖLIITTYMÄN PISTEESTÄ (PAKOLLINEN) *)
TESTI_saatotestaus_aktivointi_ohjelma := 0,

(* ASETA ALLE SÄÄTÖTESTAUKSEN KÄYTTÄMÄT HETKELLISTEN ASETUSARVOMUUTOSTEN LAAJUUDET (PAKOLLINEN, JOS EI HALLITA KÄYTTÖLIITTYMÄSTÄ) *)
TESTI_Lampotila_siirto_ohjelma := 3.0, (* C-astetta *)
TESTI_Paine_siirto_ohjelma := 100.0, (* Pa tai l/s *)
```

Liite 2. Itsetestausohjelman Excel-luodun osion alku

```
(* ASETA ALLE SÄÄTÖTESTAUSTA VARTEN TASASEKUNTEINA, MISSÄ AJASSA SÄÄDÖN TULISI TASAANTUA UUTEEN ASETUSARVOON (PAKOLLINEN, JOS EI HALLITA KÄYTTÖLIITTYMÄSTÄ) *)
TESTI_Lampotila_asettumisaika_ohjelma := 30,
TESTI_Paine_asettumisaika_ohjelma := 45,

(* ASETA ALLE SÄÄTÖTESTAUSTA VARTEN TASASEKUNTEINA, KUINKA KAUAN SÄÄTÖERON TÄYTYY PYSYÄ YHTÄJAKSOISESTI TOLERANSSIRAJOJEN SISÄLLÄ ENNEN KUIN SE HYVÄKSYTÄÄN
(PAKOLLINEN, JOS EI HALLITA KÄYTTÖLIITTYMÄSTÄ) *)
TESTI_Lampotila_varahtelyaika_ohjelma := 30,
TESTI_Paine_varahtelyaika_ohjelma := 45,

(*************************************************************************)

(* LIITÄ EXCEL-GENEROITU KOODI SEURAAVALLE RIVILLE, TÄMÄN KOMMENTIN JA LOPPUSULKEEN VÄLIIN (PAKOLLINEN)*)

TESTI_Pisteiden_lkm_ohjelma := 2,
TESTI_Saatojen_lkm_ohjelma := 1,

Piste1_tunnus := 'VAK3_SPIDER_TE11_M', Piste1_tyyppi := 3,
Piste2_tunnus := 'TK01_TE10_S', Piste2_tyyppi := 5, Piste2_saatotunnus := 'VAK3_SPIDER_TE11_M',
Piste3_tunnus := 'VAK3_SPIDER_TE11_M', Piste3_tyyppi := 1,
Piste4_tunnus := 'VAK3_SPIDER_TE11_M', Piste4_tyyppi := 2,
Piste5_tunnus := 'VAK3_SPIDER_TE11_M', Piste5_tyyppi := 4,
Piste6_tunnus := 'VAK3_SPIDER_TE11_M', Piste6_tyyppi := 6

);
```