SAVONIA
ammattikorkeakoulu

OPINNÄYTETYÖ - AMMATTIKORKEAKOULUTUTKINTO
TEKNIIKAN JA LIIKENTEEN ALA

PIIRIKAAVIOSUUNNITTELUN AUTOMATISOINTI

TEKIJÄ Mikael Laukkanen

SAVONIA-AMMATTIKORKEAKOULU

OPINNÄYTETYÖ
Tiivistelmä

Koulutusala Tekniikan ja liikenteen ala	
Tutkinto-ohjelma Sähkö- ja automaatiotekniikan tutkinto-ohjelma	
Työn tekijä(t) Mikael Laukkanen	
Työn nimi Piirikaaviosuunnittelun automatisointi	
Päiväys 16.1.2022	Sivumäärä/Liitteet 30
Toimeksiantaja/Yhteistyökumppani(t) Suomen automaatiopalvelu Oy	

Tiivistelmä (Huom. kirjoita teksti alla näkyvään harmaaseen kenttään; huomioi tämä myös kopioitaessa)

Sähkö- ja automaatiosuunnittelu teollisuuden tarpeisiin vaatii paljon dokumentaatiota ja tarkkoja suuntaviivoja. Kuitenkin projektin edetessä tulee usein muutoksia, jolloin myös projektitiedot pitää päivittää. Automatisoinnilla saadaan aikaan parempi kustannustehokkuus, sekä virheiden mahdollisuus pienenee.

Tämä opinnäytetyö oli ohjelmistosuunnittelun kehitysprojekti piirikaaviosuunnittelun avuksi, esisuunnittelusta toteutukseen. Työn tilaajana toimi Suomen automaatiopalvelu Oy. Tavoitteena oli luoda ohjelmisto, jossa analysoitiin asiakkaiden lähettämiä dokumentteja. Näistä tiedostoista kerättiin tiedot myöhempää käyttöä varten. Kuten tarjousten tekemiseen tai piirikaavioiden piirtoon, myös tietojen muuttuminen ja päivitystarve täytyi ottaa huomion.

Työmenetelmät jakautuivat kahteen osioon. Tutkiminen sähköteknisestä näkökulmasta, jossa perehdyttiin erilaisissa projekteissa käytettyihin esitietoihin ja tutkittiin mitkä tiedot olivat piirikaavioiden piirron kannalta tarpeellisia. Tutkiminen rajattiin pääosin IO-luetteloihin, koska kaikki tarvittava tieto ohjelman käyttöön löytyi luetteloista. Toinen osio koostui ohjelmoinnista, joka tapahtui pääosin python ohjelmointikielellä. Ohjelma toteutettiin siten, että se oli muokattavissa ja mahdollista liittää vaivattomasti toisiin koodeihin.

Kehitystyöntuloksena saatiin usean koodin kokonaisuus betatestaukseen. Ohjelmisto sisälsi muun muassa työkalut tarjousten laadintaan, luetteloiden vertailuun sekä lajitteluun. Tästä saatiin puitteet piirikaavioiden piirtoon. Koodi liitettiin toiseen ohjelmaan, joka piirsi instrumentin piirikaavion halutuilla reunaehdoilla. Testaukset vaikuttivat lupaavilta ja ohjelmistolle löytyi useita jatkokehityskohteita.

Avainsanat
Data-analyysi, Python, Instrumentointi, PLC

SAVONIA UNIVERSITY OF APPLIED SCIENCES

THESIS
Abstract

Field of Study Technology, Communication and Transport	
Degree Programme Degree Programme in Electrical and Automation Engineering	
Author(s) Mikael Laukkanen	
Title of Thesis Automation of Electrical Circuit Design	
Date 16.1.2021	Pages/Appendices 30
Client Organisation /Partners Automation Service Finland ltd	

Abstract (NOTE: write/insert all your text in the grey box below, also if you use copy + paste)

The Design of Electrical and automation projects in industrial field requires huge amount of documentation as well as compliance with rules and regulations. Even if these guidelines are followed it is common that several files must updated during the project. Automation of different design stages gives better efficiency and possibility of errors decreases in the project files.

The aim of this thesis was the development of software in order to smoothen the process and reduce the workload of electrical circuit designing. The project was commissioned by the Finnish Automation Service ltd. The purpose of the thesis was to create a program which analyses pregiven documents and data. As an outcome, the filtered data could be utilized at different stages of the project, including making an offer to a customer or draw a circuit diagram. The possibility of changes in the input data files during project duration had to be considered in software development.

The project was divided to two stages. First, the theoretical aspect focused on studying the compulsory information for building a circuit diagram. The Second part consisted of selecting programming language and environment as well as building software.The Program had to be implemented in such a way that it was easy to modify and could be combined with other parts of the code created by other users.

The software created during this project covered various above-mentioned features. Software includes tools to make offers to the customers and a platform for data analysis which enables sorting and filtering specified project information into their own databases. Beta testing results were promising, and the project led to several further development ideas to improve the software in the future.

Keywords
Data-analytic, Python, Instrument design, PLC

SISÄLTÖ

KUVALUETTELO

TERMIT/LYHENTEET

API – Application Programming Interface – Ohjelmointirajapinta

ATEX – Atmosphèes explosibles – Räjähdysvaaralliset tilat

CAD – Computer-aided Design – Tietokone avusteinen suunnitteluohjelmisto

CENELEC – European Committee for Electrotechnical Standardization – Sähköalan eurooppalainen standardointijärjestö

GUI – Graphical user interface – Graafinen käyttöliittymä

IDE – Integrated development environment – Ohjelmointiympäristö

I/O – Input / Output - Sisääntulo / ulostulo

IEC – International Electrotechnical Commission – Kansainvälinen sähköalan standardointijärjestö

PLC – Programmable logic controller– Ohjelmoitava logiikka

SIA – Sähkö, instrumentointi ja automaatio

SFS – Suomen standardisoimisliitto

1 JOHDANTO

Sähkö- ja automaatiosuunnittelu voi olla suuri prosessi, joka kattaa useita osa-alueita, komponenttien valinnasta erilaisten kaavioiden sekä sovellusohjelmistojen tekoon. Projektien laajuudet vaihtelevat yksittäisten komponenttien muutoksista, suuriin satojen jopa tuhansien I/O-pisteiden kokonaisuuksiin. Kaikista sähkölaitteista piirretään piirikaaviot, joista selviää komponentin kytkennät sekä toiminnallisuus.

Opinnäytetyön toimeksiannon antoi Suomen automaatiopalvelu Oy. Tavoitteena kehittää sovelluskokonaisuus, joka vähentää SIA projektien suunnitteluprosessissa suunnittelijoiden sekä myyjien työmäärää työvaiheiden automatisoinnin myötä. Näillä toiminnoilla saadaan aikaan kustannustehokas ratkaisu, joka ottaa huomioon mahdolliset virheet, joita projektimateriaalit voivat sisältää.

Automatisoinnilla saadaan analysoitua suunnitteludokumentteja sekä niiden mahdollisia muutoksia myöhemmässä suunnitteluvaiheessa. Analysoidusta datasta kerätään tietokanta, jota hyödynnetään projektin eri vaiheissa. Näin muuttuneet tiedot voidaan päivittää myös muihin olemassa oleviin materiaaleihin. Tästä syystä työvaiheet, joissa käytännössä toistetaan samaa työnkulkua kopioimalla jää pois. Analyysi tuo esille myös dokumenteissa käytettyjen laitteistojen määrät ja tyypit, joita voidaan hyödyntää niin tarjouslaskennassa kuin mitoituksessa.

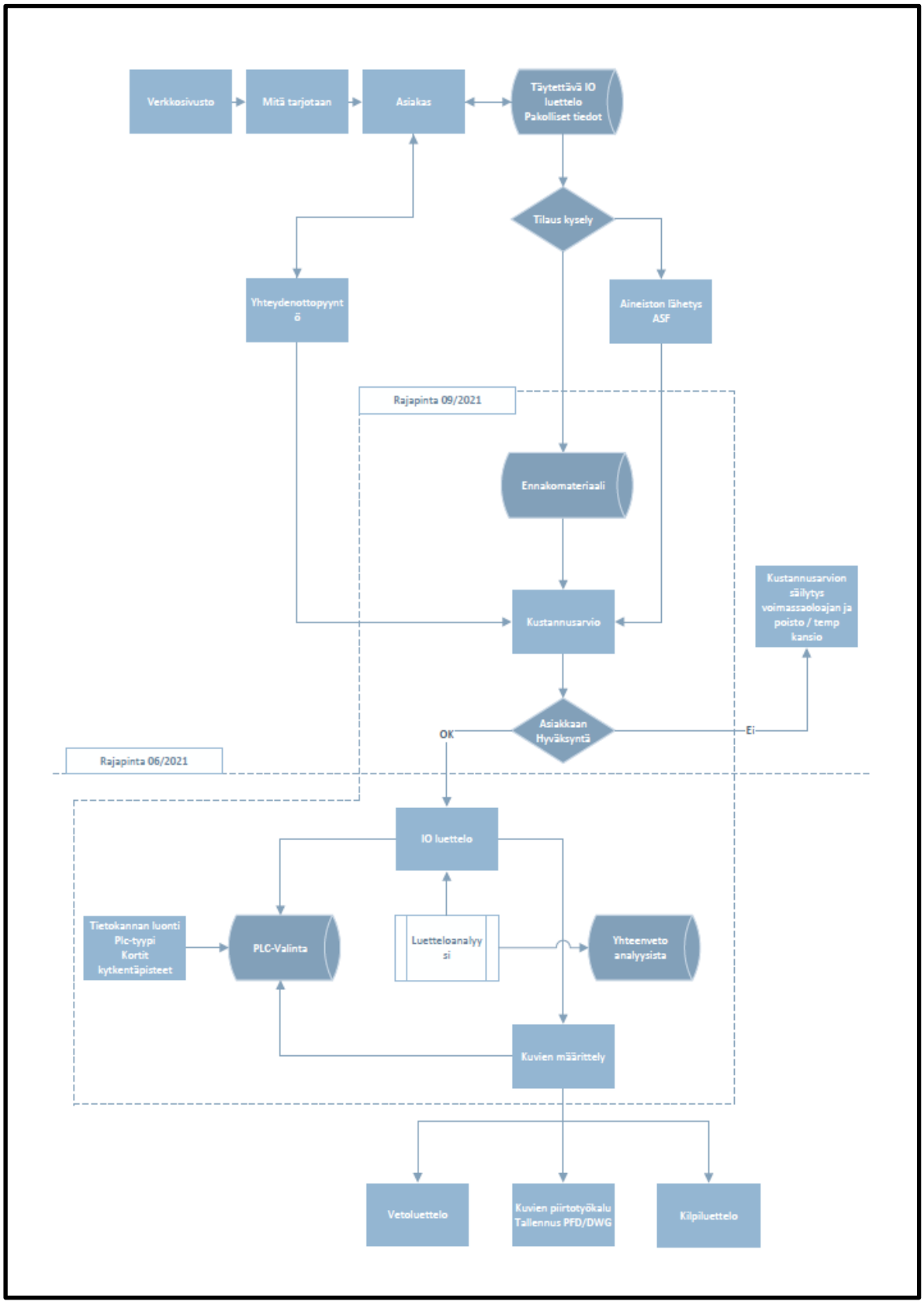
Ohjelmiston luominen vaati tutkimusosion sähkötekniikasta ja ohjelmoinnista. Sähköteknisessä tutkimuksessa pyrittiin löytämään tiedot, joita sovellus tarvitsee analyysin rajapintoihin. Ohjelmoinnin esikartoituksessa etsittiin parhaat toimintatavat ja ohjelmointiympäristöt kyseisen työn käyttöön.

1.1 Toimeksiantaja

Suomen Automaatiopalvelu Oy (ASF) on suunnittelu/konsultointi yritys, joka on saavuttanut laajan erikoisaamisen teollisuuden alalla. Yritys on perustettu 2017 omistajan Tero Auvisen johdosta ja toimipaikka sijaitsee Siilinjärvellä. Yritys tarjoaa palveluitaan SIA-järjestelmien suunnittelun ja toteutuksen lisäksi, muun muassa prosessien analysoinnin, optimoinnin sekä ohjelmoinnin parissa. (Suomen Automaatiopalvelu OY. 2018)

1.2 Rajapinta

Koska kyseessä oli kohtuullisen suuri projekti, työtä rajattiin ohjelmoinnin osalta analyysityökaluihin sekä tietokantojen luomiseen (kuva1). Ohjelmassa pyrittiin välttämään niin sanottua kovakoodausta, joka tarkoittaa sitä, että tiettyjä ohjelmiston osia ei voida muuttaa kajoamatta lähdekoodiin. Näin ollen jokainen kovakoodattu muuttuja jouduttaisiin muuttamaan erikseen koodin sisällä. Kovakoodatut ohjelmisto-osat kasvattavat työmäärää jatkokehityksessä. (Technopedia. 2021)



Kuva 1 Opinnäytetyön aihekuvaus, muokattu alkuperäisestä työsuunnitelmasta. (Laukkanen. 2021)

2 SÄHKÖSUUNNITTELU

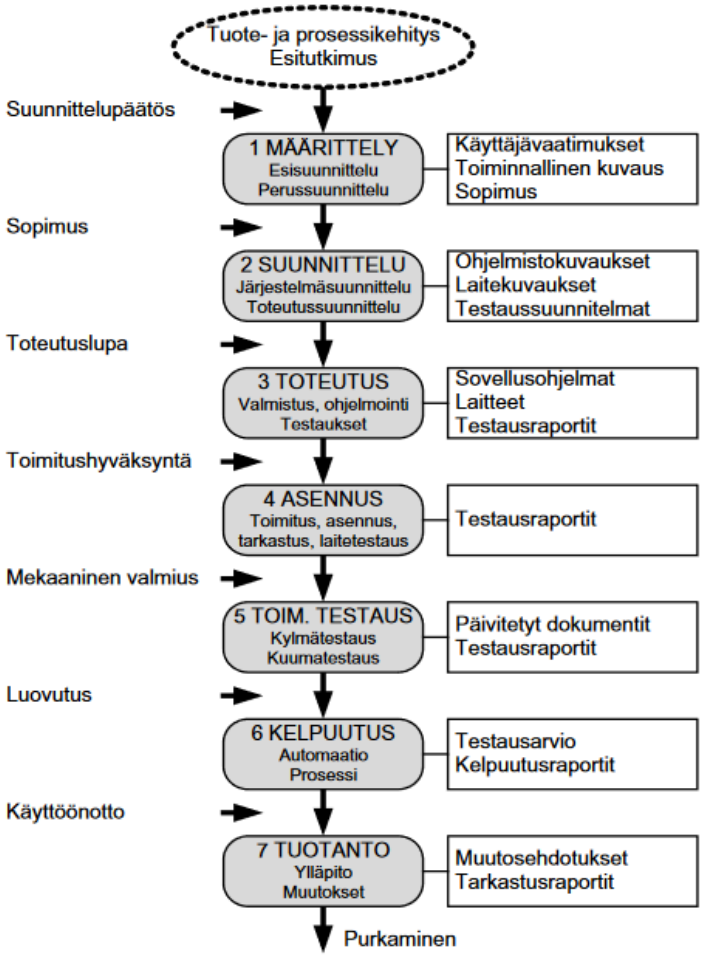
Sähkösuunnittelu piirrosteknisestä näkökannasta kattaa useita sääntöjä ja standardeja. Vaikka jokaisella suunnittelijalla ja yrityksillä on oma tapa tehdä suunnitteludokumentteja, kuitenkin samat ominaisuudet pitää asiakirjoista löytyä. Yleinen tapa on, että aiempien projektien dokumenttipohjia käytetään uusissa projekteissa. Vanhojen pohjien käyttö nopeuttaa suunnittelua, mutta myös altistaa siihen, että vanhentuneita tietoja edellisistä projekteista voi siirtyä uuteen projektiin.

Dokumenttilajit jaetaan omiin kategorioihin, joille kansainväliset standardit määrittävät omat säännökset (SFS-EN 61082-1. Sähkötekniikassa käytettävien dokumenttien laatiminen, Osa 1: Säännöt, 2015.):

- Piirustukset
- Kaaviot
- Taulukot
- Diagrammit

Euroopassa dokumentoinnissa noudetaan IEC, CELECIN, sekä SFS:n standardeja. Säännöt pitää sisällään millaisia tietoja asiakirjoista pitää ilmetä sekä millainen ulkomuoto dokumentilla on. Näin asiakirjasta tulee suoraviivainen, informatiivinen ja tunnistettava, joka estää väärinkäsitykset dokumentin luvussa. (SFS-EN 61082-1. Sähkötekniikassa käytettävien dokumenttien laatiminen, Osa 1: Säännöt, 2015.)

Suunnitteluvaiheessa kommunikaatio muiden suunnittelualojen ja asiakaan jäsenten kanssa on elintärkeää projektin onnistumisen kannalta. Automaatioprojektin elinkaarella tarkoitetaan koko projektin kulkua määrittelystä tuotantovaiheeseen. Elinkaarimallissa on useita vaiheita, jotka nivoutuvat yhteen kaikkien suunnittelualojen kanssa. (Suomen Automaatioseura ry. 2007.) Kuvasta 2 voidaan nähdä, että pelkästään sähkösuunnittelun osuus kuuluu osioihin 1, 2 ja 4. Edellä mainittujen kohtien viivästykset tai puutteet vaikuttavat radikaalisti muun projektin läpivientiin.



Kuva 2 Automaatioprojektin elinkaarimalli (Suomen Automaatioseura ry. 2007)

Yksi suurista ongelmista projekteissa on tiedon ja päivitettyjen dokumenttien ajantasainen tiedonsiirto kaikille osapuolille. Tiedonsiirto on nykyään helpompaa dokumenttien sähköistymisen myötä sekä yhtenäisten projektipankkien ansiosta. Projektipankit sijaitsevat pilvipalvelimilla ja kaikilla asianosaisilla on pääsy suunnitteludokumentteihin. Saneerausprojekteissa tilanne voi olla haastavampi. On mahdollista, että sähkölaitteiden päivityksiä tai muutoksia ei ole dokumentoitu kuin prosessitiloissa sijaitseviin piirikaavioihin. Lähtötietojen hakeminen vaatii laitoksen laajamittaisen kartoittamisen. (Suomen Automaatioseura ry. 2007)

2.1 Luettelointi

Teollisuusprojekteissa on monia luetteloita, joita käytetään projektin aikana, mutta myös työn valmistuttua esimerkiksi kunnossapidon sekä vianetsinnän apuna. Nämä ovat tietokantoja, jotka ovat eritelty osiin käyttötarkoituksen mukaisesti. Kuitenkin nykypäivänä on yleistä, että luettelot yhdistetään suuremmiksi kokonaisuuksiksi.

I/O-luettelo:

- Sisältää kaikki sähköisesti ohjatut kenttälaitteet. Luettelossa kerrotaan kuinka laitteet kommunikoivat PLC:n kanssa ja kuinka muut järjestelmät ovat yhteydessä logiikkaan.

Laiteluettelo:

- Kattaa kaikki käytetyt komponentit osanumeroineen sekä laitteiden tekniset tiedot. Luetteloa voidaan käyttää komponenttien hankintalistana.

Kaapeliluettelo:

- Tarvitaan kaapelointivaiheessa. Kertoo pisteet mistä kaapelointi lähtee ja mihin päättyy. Luettelosta tulee ilmi kaapelitunnukset, jotka kiinnitetään kaapeleihin.

Kilpiluettelo:

- Komponentit ovat yksilöityjä ja niihin asennetaan omat tyyppikilvet. Kilvet kiinnitetään jokaiseen koteloon, komponenttiin ja laitteeseen omalla koodilla.

Kytkentäluettelo:

- Piirikaavioiden avuksi tehty luettelo, josta selviää kaapeloinnin kytkentäpisteet.

2.1.1 I/O-luettelo

Usein I/O-luettelo on yhdistetty instrumenttiluettelon kanssa, jolloin luetteloon on koostettu kaikki tarvittavat tiedot kenttälaitteista ja prosessin osista. Tietojen avulla voidaan luoda muita asiakirjoja, tehdä karkeat mitoitukset sekä piirustukset. Työssä keskityttiin eri projekteissa käytettyihin I/O-luetteloihin, joissa tietosisältö on perinteisiä input/output luetteloita laajempi. Kenttälaitteet ja instrumentit ovat yksilöityjä, listasta selviää komponenttien tyyppi sekä ohjaustapa. Kaavio voi sisältää myös informatiivisia erikoisvaatimuksia laitteelle, jotka on otettava huomioon suunnittelussa. Erikoisvaatimuksia voi olla muun muassa ATEX-luokitukset tai instrumentin turvapiiriin kuuluminen.

Räjähdysvaarallisissa tiloissa käytettävien laitteiden komponentit, kaapelointi ja kytkennät eroavat normaaleista. SFS-EN 60079-14 on piirien suunnittelun ja laitevalintojen kannalta tärkeä standardi. Standardi antaa erikoisvaatimukset ATEX-tilojen suunnittelua varten eri ympäristöolosuhteissa.

Turvapiirikomponentit voidaan suojata erillisellä turva PLC:llä tai muulla turvalaitteella kuten turvareleellä. SFS-EN 60204-1:2018 määrittää tarkemmat vaatimukset turvapiirisuunnittelulle.

I/O-luettelo on tärkeä osa projektin alkuvaiheessa, koska siitä selviää käytettävät I/O-pisteet ja ohjaustavat. Luettelon avulla voidaan mitoittaa PLC laitteistoon vaadittavien komponenttien ja I/O korttien määrä. Pisteiden lukumäärän avulla voidaan myös tehdä laskenta piirikaavioiden määrästä mitä projektiin tarvitaan. Laskenta mahdollistaa tarjouksen luonnin piirikaavioiden piirrosta. Luettelo voi myös sisältää tarkemmat tiedot komponenteista kuten osanumerot, tekniset tiedot ja valmistajan (Kuva3).

	A	B	C	D	F	G	H	I	J	K	L	M	O	T	Z	AA	AB	AC	AD	AJ	BG	BJ
1	Rev	PROSESSIALUE	E:I	INSTRUM TYYPPI	POSITIO	PIIRIN NIMI	Nimellisjännite V	Teho kW	Virta A	Cos phi	1/min	TAAJUUSMUUTTAJA	LAITEVALMISTAJA	ATEX	DI	DO	AI	AO	Profinet väylä	Profibus	MITTA-ALUE	Fail Safe
2	A	ALUE1	I	MANIKKUVENTTIILI KALVO	TESTI1V04	magneettiventtiili1	24										1					
3	A	ALUE1	I	NTAKYTKIN VÄRÄHTELEV	TESTI1LH1	Pintakytkin1	24						Endress Hauser		1							
4	A	ALUE1	E	TÄRYMOOTTORI	TESTI1M01	Täry1	400	2	4	0,86	2850				3	1						
5	B	ALUE1	E	KULJETIN	TESTI1KK1	Kuljetin1	400	2,2	4,95	0,79	1420				3	1	1					
6	B	ALUE1	E	SULKUSYÖTIN	TESTI1M01	Sulkusyötin1	400	1,1	2,9	0,81	1400	X							X			
7	C	ALUE1	I	PAINEKYTKIN	TESTI1PS01	säiliön paine1	24															
8	C	ALUE1	I	VIRTAUSKYTKIN	TESTI1FS01	putken virtaus1	24						Endress Hauser	X							0-100	
9																						
10																						
11																						
12																						
13																						
14																						

Kuva 3 Tyypillisiä I/O-luettelossa sijaitsevia tietoja. (Laukkanen.2021)

2.2 Piirikaavio

Piirikaavio on tuo esiin vähintään yhden sähkölaitteen toiminnan yksityiskohtaisesti sekä liitäntäpisteet muihin piireihin. Tavoitteena on havainnollistaa komponentin toiminallisuus ja kytkentä. Tämä toimii kytkentäohjeena asennuksessa, huoltotoimenpiteissä sekä mahdollistaa tarkemman vianetsinnän vikatilanteissa. Kaavio sisältää standardien mukaisia piirrosmerkkejä, jotka liitetään yhteen seuraavaan kohteeseen joko viivoilla tai viitetunnuksilla. (SFS-EN 61082-1. Sähkötekniikassa käytettävien dokumenttien laatiminen, Osa 1: Säännöt, 2015.)

Kaavioiden esitystavat vaihtelevat komponentin kytkennän monimutkaisuuden mukaan. Esitystapoja on muun muassa vapaa, toistettu ja koottu. Esitystavoilla pyritään toteuttamaan selkeä kokonaisuus, jossa vältetään piirien risteämistä.

- Vapaassa esitystavassa komponentti voidaan jakaa kahteen eri osaan piirin selkeyttämisen vuoksi. Esimerkiksi kontaktorin kärkitiedot näytetään päävirtapiirissä ja kela ohjauspiirissä. Sama komponentti on viitemerkattu tunnistamisen vuoksi.

- Toistetussa esitystavassa komponentti voi esiintyä piirroksessa useaan otteeseen joko osina tai kokonaisuudessa. Esitystapaa voidaan hyödyntää, jos komponentin liitäntätiedoista tarvitaan vain tiettyä osaa kerrallaan.

- Kootussa esitystavassa ohjaus sekä syöttö on liitetty samaan kuvaan lähekkäin ja mekaaninen yhteys on yhdistetty katkoviivalla. Tätä esitystapaa suositellaan vain yksikertaisissa piireissä, jotka eivät sisällä monimutkaisia kytkentöjä. Koottua esitystapaa voidaan käyttää esimerkiksi yksinkertaisen moottoripiirin luomiseen, jota ohjataan vain kytkimellä ja kontaktorilla.

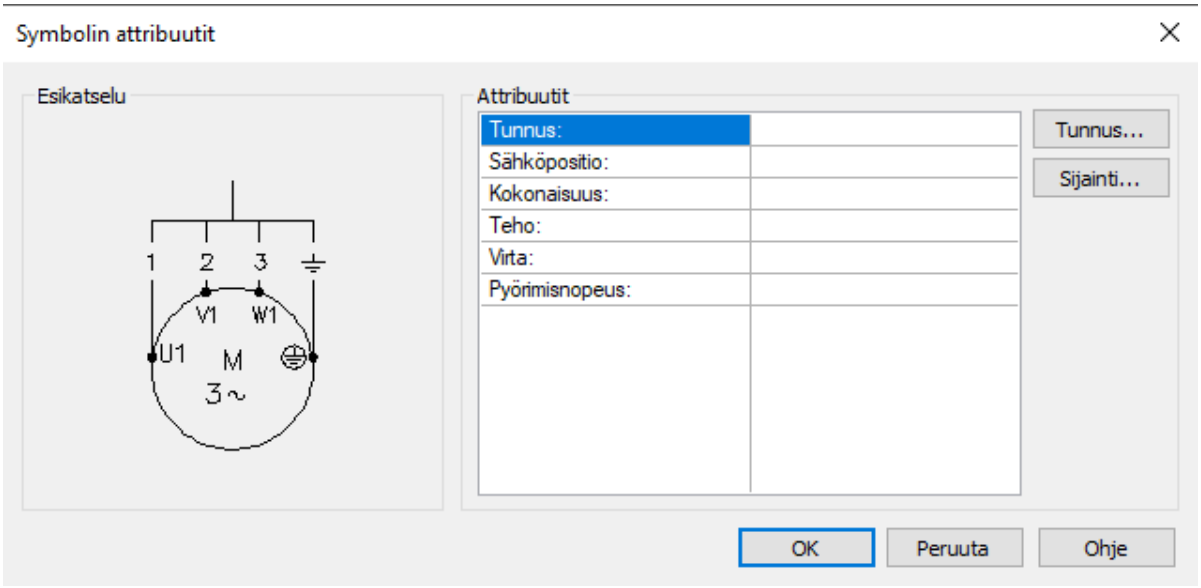
(SFS-EN 61082-1. Sähkötekniikassa käytettävien dokumenttien laatiminen, Osa 1: Säännöt, 2015.)

2.3 Piirustusohjelmat CAD

Piirustusten tekemiseen käytettäviä ohjelmistoja on useita eri valmistajilla. Puhutaan CAD-ohjelmistoista, joka tarkoittaa tietokoneavusteista suunnitteluohjelmistoa. Ohjelmistot kehitettiin graafiseen suunnitteluun tekniikan aloilla 80-luvun alussa. Yksi ensimmäisistä ohjelmista oli Autocad, joka julkaistiin vuonna 1982. (Technopedia. 2021)

Ohjelmistot pohjautuvat vektorigrafiikkaan, joissa piirretään viivoja ja muotoja koordinaatistoihin. Sähköpiirustukset tehdään joko DWG- tai DFX-formaateissa ja ovat piirikaavioissa yleensä kaksiulotteisia. Ohjelmistovalmistajat ovat tehneet eri lisäosia tuotteisiinsa, jotka ovat erikoistuneet tiettyyn tekniikan alaan. Esimerkiksi sähkötekniikan standardoidut piirrosmerkit voivat olla suoraan ohjelmiston kirjastossa symboleina. Piirikaavioiden piirtoa voidaan myös tehdä muilla ohjelmistoilla, jolloin ohjelmistosta pitää löytyä tuotetuki, joka kääntää tiedoston DWG-formaattiin.

Piirustusohjelmien symbolit sisältävät attribuutteja. Attribuutit ovat identifioituja, jolloin samaa symbolia voidaan hyödyntää useissa kuvissa ja muuttaa pelkästään tietosisältö symbolista. Käyttäjä voi myös itse määrittää mitä attribuutti pitää sisällään ja mitä osia on näkyvissä. (Autodesk help. 2020) Esimerkiksi moottorisymbolin tekniset tiedot muutetaan vastaamaan haluttua moottorityyppiä (kuva4). Valmiit tai käyttäjän määrittämät symbolit lyhentävät piirtoon kulunutta aikaa. Kyseistä tapaa hyödynnetään, kun objekteja on useita samanlaisia. Ominaisuutta voidaan käyttää myös ohjelmoinnissa, jolloin ohjelma valitsee halutun symbolin ja täyttää sen määrätyillä tiedoilla. Lopuksi symboli sijoitetaan kuvaan koordinaattien perusteella.

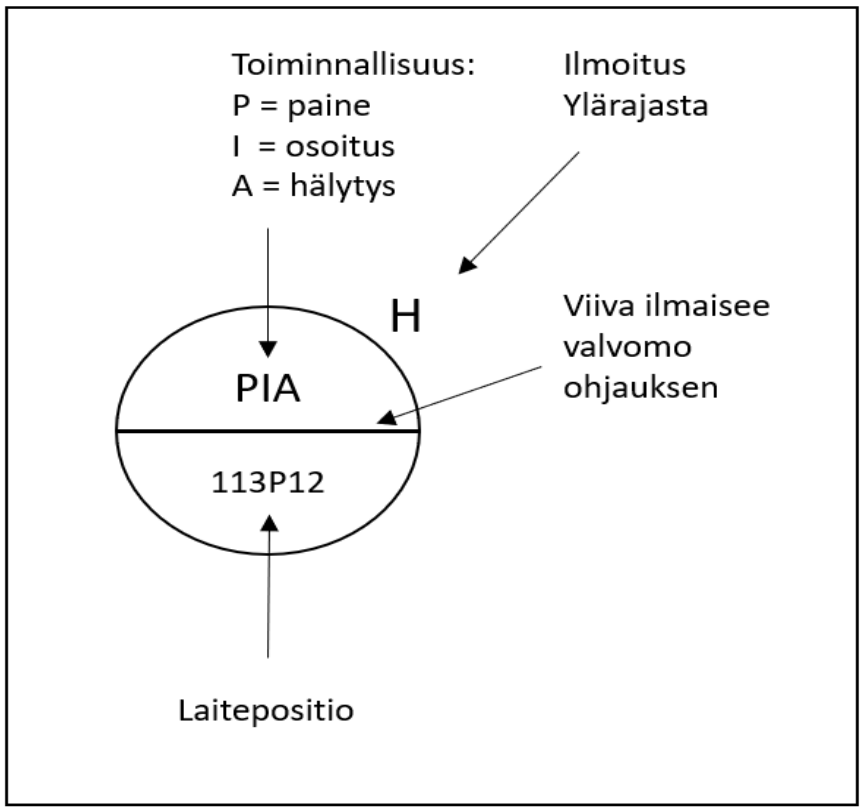


Kuva 4 Attribuutin täyttö. (Laukkanen. 2021)

2.4 Kenttälaitteet

Työn analysoinnissa haettiin eroavaisuuksia eri kenttälaitteiden välillä. Tavoitteena oli spesifioida jokainen komponentti ja sen käyttötarkoitus. Luokittelun avulla saatiin luotua tietokannat, josta pystyi päättelemään piirin toiminnan. Toiminallisuudet kertovat onko kyseessä piiri, joka antaa tietoa eteenpäin, mittaa tai ohjaa jotain prosessin osaa. Tämä mahdollistaa piirikaavioiden luonnin, sekä karkeasti instrumenttien kaapeloinnin valinnan.

Prosessitekniikassa käytetään standardoituja SFS-ISO 14617-6 mukaisia kirjaintunnuksia määrittämään kenttälaitteen tyyppi. Kirjaintunnukset löytyvät prosessi- ja instrumentointikaaviosta graafisessa muodossa, jotka ovat tietyn muodon sisällä. Muotorakenne kertoo tarkemmin, onko kyseessä paikallis- vai valvomo-ohjaus. Muodon ulkopuolella voi olla lisäselitteitä komponentille kuten onko kyseessä ylä- vai alarajatieto (kuva5) (TL121105 AUTOMAATIOTEKNIIKKA 1. 2009.).



Kuva 5 Esimerkki PI-kaavion mukaisesta symbolista selityksineen. (Laukkanen. 2021)

PI-kaavion graafisen kuvan lisäksi kirjaintunnus on myös sisällytetty kenttälaitteen positioon. Positiosta voidaan nähdään millainen suure on kyseessä ja mikä on sen käyttötarkoitus.

Taulukko 1. SFS 14617-6 mukaiset kirjaintunnusten selitykset

Kirjaintunnus	Mittasuure	Lisämäärite	Toiminta
A			Hälytys
B			Eri tilojen näyttö
C			Ohjaus
D	Tiheys	Ero	
E	Sähkösuureet		Anturitoiminta
F	Virtaama	Suhde, murtoluku	
G	Suhde, Asento, Pituus		Tarkastelu
H	Käsiohjaus		
I			Osoitus
J	Voima	Pyyhkäisy	
K	Aika	Muutosnopeus	
L	Pinnan korkeus		
M	Kosteus	Hetkellisesti	
N	Käyttäjän valittavissa		
O	Käyttäjän valittavissa		
P	Paine		Testauskohdan yhteys
Q	Laatu	Yhtenäinen, kokonainen	Yhdistäminen Summaaminen
R	Säteily		Tallennus
S	Nopeus		Kytkentä
T	Lämpötila		Lähetin
U	Monimuuttuja		Monitoiminta
V	Käyttäjän määrittämä arvo		Vaikuttaminen
W	Paino	Kertominen	
X	-		-
Y	Käyttäjän määrittämä arvo		Laskenta
Z	Lukumäärä		Hätätoiminto

Kirjaintunnukset koostuvat usean kirjaimen yhdistelmästä. Ensimmäinen kirjain viittaa mittaussuureeseen. Seuraavat kirjaimen kertovat laitteen toimintatavasta. Komponentin hälytysominaisuus tulee aina kirjanyhdistelmän viimeiseksi. Esimerkiksi positiotunnus: 115LIA021 kertoo, että kyseessä on pinnankorkeuden osoitus hälytyksellä. Numerot voivat viitata esimerkiksi prosessikohteeseen tai monesko komponentti on kyseessä, riippuen siitä missä kohtaa numerot sijaitsevat. Standardin SFS 14617-6 mukaiset kirjainten selitykset löytyvät taulukosta 1.

Positiossa voi olla myös vain yksi kirjain, joka yleensä viittaa toimilaitteeseen kuten M moottoriin tai V venttiiliin. I/O-Luetteloista löytyy lisätietoa, joita yhdistelemällä saadaan tietää laitteen tarkempi toiminta.

Kenttälaitteen toiminnan rajauksen jälkeen selvitetään kytkentätapa ja millä tavoin instrumentti kommunikoi. Selvitetään, onko kyseessä aktiivinen vai passiivinen kenttälaite. Aktiivinen kenttälaite ei tarvitse erillistä virtalähdettä toimiakseen.

PLC:n kanavien kytkentään vaikuttaa millainen instrumentti on kyseessä ja millaiset laitteen kytkentäohjeet ovat. Instrumenteissa käytetään 2, 3 ja 4 johdon kytkentöjä. Pääsiassa useamman johdon kytkennät koskevat analogiaviestejä. Usean kaapelin kytkennöissä mittaustarkkuus paranee mutta kytkennät ovat monimutkaisempia. Työssä luotiin kytkentätietokannat Siemensin s7-1500 logiikkasarjan DI/DO/AI/AO korteille kahden ja neljän johdon kytkennöille.

Kenttälaitteiden tiedonsiirtotapoja on erilaisia. I/O-pisteiden kautta tuleva signaalisiirto, väylätekniikka tai langattomat vaihtoehdot. Väylätekniikka on yleistynyt teollisuuden tiedonsiirtotapana. Komponenttien ohjausväylät voivat koostua ethernet pohjaisista ratkaisuista, kuten Siemensin Profi-NET tai Beckhoffin TwinCAT. Instrumenteilla on usein käytössä kenttäväylä, esimerkiksi IO-link tai ASI-väylä. Väylätekniikan käytöllä saadaan vähennettyä kaapelointia ja siirretty data voi olla kattavampaa perinteiseen signaalisiirtoon verrattuna. Kuitenkin I/O-pisteiden kautta tuleva kommunikointi pitää paikkaansa yhtenä käytetyimpänä tiedonsiirtomenetelmänä yksinkertaisuutensa vuoksi. Langaton tiedonsiirto kenttälaitteissa on yleistymässä mutta on toistaiseksi vähemmän käytetty teollisuudessa. Prosessien kannalta kriittisesti tärkeissä komponenteissa halutaan varmistaa toimintavarmuus kaapeloidulla ratkaisulla.

3 OHJELMOINTI

Ohjelmointi osiossa valittiin käytettävä ohjelmointikieli, joka soveltuu parhaiten analyysiin sekä keskustelemaan muiden ohjelmien kanssa. Ohjelmointi suoritettiin pääosin Python ohjelmointikielellä. Työn kannalta oli myös tärkeää, että käytettävä kieli oli osin tuttu tekijälle, näin työhön laskettu aika saatiin mahdollisimman tehokkaasti sovelluskehityksen käyttöön.

Osin piirikaavioihin käytettävissä ohjelmistoissa on mahdollisuus myös automatisoida suunnittelua, mutta tällöin muunneltavuus olisi vaikeaa ja yritys olisi sidoksissa tiettyyn ohjelmistoon. Myös työssä määritettyjen toiminnallisuuksien yhdistäminen olisi vaatinut useamman ohjelman käyttöä. Tästä syystä toimeksiannossa kehitettiin oma sovelluskokonaisuus, joka soveltuu parhaiten yrityksen käyttöön.

3.1 API

API on ohjelmistorajapinta, joka mahdollistaa kahdensuuntaisen kommunikaation kolmansien osapuolien ohjelmistojen tai verkkopalvelimien kesken. Yritykset tekevät ohjelmiinsa ominaisuuden käyttää rajapintaa integraation sekä brändin kehityksen takia. Aluksi ilmainen rajapinta voidaan muuttaa maksulliseksi myöhemmin tunnettavuuden kasvettua. (IBM Cloud Education. 2020)

API tarjoaa turvallisen kommunikaatio vaihtoehdon, johon on mahdollista lisätä turvaominaisuuksia eri tasojen ja tunnistautumismenetelmien kautta. (IBM Cloud Education. 2020)

3.2 Python

Python on ohjelmointikieli, jonka Guido Von Rossum kehitti vuonna 1991 jatkona ABC-ohjelmointikielelle. Kieli sisältää ominaisuuksia, jotka tekevät siitä helposti ymmärrettävän sekä nopeasti opittavan. Python ei sinänsä ole alusta riippuvainen, vaan python tarvitsee toimiakseen tulkin. Tulkki kääntää komentorivit ohjelmassa, jolloin koodi voidaan toteuttaa. Näin ollen koodi ei muutu ja pythonia voidaan käyttää kaikissa käyttöliittymissä kuten Windows, Linux sekä Applen Mac käyttöjärjestelmillä (Nelli, 2018)

Python ohjelmointikieltä käytetään laajalti data-analyysissa mutta se sisältää myös useita kirjastoja muihin ohjelmistokehityksen tarpeisiin. Kieli perustuu myös avoimeen lähdekoodiin ja on tehty helpoksi käyttää. (Nelli, 2018)

Laajalti käytössä oleva kieli on kasvattanut suosiotaan vuosien varrella ja on kerännyt kokoon ammattilais- ja harrastelijayhteisöjä. Yhteisöt testaavat, päivittävät ja kehittävät koodia alituiseen. Myös avoimet foorumit kuten Stack Overflow on tehty tiedonhakuun ohjelmoinnin tueksi. Foorumilta löytää kysymyksiä ja vastauksia ohjelmoinnissa vastaan tulleisiin ongelmiin harrastajilta sekä ammattilaisilta. Stack Overflowsta löytyy tietoa eniten käytetyille ohjelmistokielille. Ongelmakohtiin, joita koodin kirjoittamisessa voi syntyä, mahdollisesti on jo löydetty ratkaisu.

3.2.1 Kirjastot

Python sisältää useita integroituja toimintoja, jotka sisältyvät suoraan python asennuspakettiin. Lisäksi on tarjolla lisäosia ja kirjastoja, jotka voidaan ladata ohjelmaan. Kirjastojen lähdekoodi on rakennettu siten, että käyttäjällä on käytössään komentoja, joilla päästään syventymään juuri tiettyyn toiminallisuuteen. Kuten matemaattiset laskutavat, visuaaliset taulukoinnit tai graafiset käyttäjänäkymät. Kirjastojen tarkoitus on, että monimutkaisetkin toiminnot voidaan toteuttaa muutamalla koodirivillä satojen rivien pohjakoodin sijaan. Kirjastojen lähdekoodi voi olla koodattu myös eri koodikielillä, kuten C++.

- Numpy
  Numpy kirjasto on erikoistunut tieteellisten laskutoimitusten tekemiseen. Kirjastoa voidaan käyttää data-analyysiin, mutta myöhemmin kehitetty Pandas syventyy aiheeseen enemmän. Numpy tarjoaa toimintoja peruslaskutoimituksista kehittyneempiin matemaattisiin ongelmanratkaisu metodeihin kuten: vektori- ja matriisilaskentaan. (Nelli, 2018)

- Pandas
  Pandas on data-analyysi työkalu, joka on kehitetty Numpy-kirjaston pohjalle. Kirjasto pystyy prosessoimaan dataa omissa datarakenteissa: ”series & dataframe”. Tiedon prosessointi ei rajoitu pelkästään numeerisen dataan. (Nelli, 2018)

  Pandas kirjaston tietorakenteiden ”series” & ”dataframe” käyttö riippuu halutusta käyttötarkoituksesta. Molemmissa rakenteista löytyy samankaltaisia ominaisuuksia, mutta sarjat ovat yksiulotteisia taulukoita, joita käytetään yksinkertaisemman datan kanssa. Dataframe soveltuu monimutkaisten ja laajempien datan määrien analyysiin moniulotteisen rakenteensa ansiosta. (Nelli, 2018)

  Analyysitoiminnoista löytyy erilaisia ominaisuuksia mitä opinnäytetyössä tarvittiin, kuten:
    - Aineiston vertailu
    - Materiaalin manipuloiminen
    - Datan siirto toisiin tietokantoihin.
    - Permutointi eli datan järjestely

  Panda pystyy kirjoittamaan ja lukemaan useita eri tiedostoformaatteja kuten XML, JSON, SQL. Tämä tekee työkalusta täydellisen välineen data-analyysiin eri ohjelmaformaattien kanssa. (Wes McKinney, Pandas Development Team. 2021)

- Tkinker
  Tkinker kuuluu nykyään integroituna pythonin paketteihin. Tkinkerillä voidaan luoda graafinen käyttöliittymä (GUI), jolloin ohjelma pyörii taustalla ja käyttäjällä oleva visuaalinen näkymä on helpompi omaksua. Ohjelmalla voidaan myös hakea kansioita ja tiedostoja, jotka saadaan talletettua suoraan muuttujiin. (Python software foundation. 2021.)

3.2.2 Virtuaaliympäristö

Pythonin versiot ja ohjelmakirjastot sisältävät erilaisia ominaisuuksia ja eroavaisuuksia versioiden välillä. Virtuaaliympäristöillä saadaan luotua oma eristetty ympäristö, jonka asetukset määritetään projektikohtaisesti. Jos käyttäjä päivittää käyttöjärjestelmää tai ohjelmia, niin yhteensopivuus ongelmia ei synny. Virtuaaliympäristöön luodussa projektissa käytetään ohjelmisto- ja kirjastoversioita, jotka määritetään projektin luonnin yhteydessä. Versiot ladataan virtuaaliympäristöön ja ne eivät muutu normaalien päivitysten mukana. (Python software Foundation. 2011)

3.2.3 Ohjelmointiympäristö

Python koodia voidaan käsitellä joko terminaaleissa ”Shell”, jossa komentorivit kirjoitetaan tekstikenttään, jolloin tulkki kääntää koodin ja suorittaa ohjelman. Toinen lähestymistapa python ohjelmointiin on IDE-ympäristöt. Seuraavassa kohdassa esitetään ohjelmia, joihin sisältyy python ohjelmistotuki tai ovat pelkästään pythonille kehitetty.

- Anacondas

  Laaja ilmainen kokonaisuus, johon on integroitu laajalti käytettyjä ohjelmistoja kuten Jupyter, QTconsole ja spyder IDE. Tukee Windows, Linux, MACOS X käyttöjärjestelmiä. Anaconda navigaattorin sisällä voidaan luoda virtuaaliympäristöjä tietyille python versiolle, johon saadaan konfiguroitua halutut ohjelmistopaketit.

- Jupyter

  Jupyter notebook verkkopohjainen käyttöliittymä. Käyttöliittymässä voidaan kirjoittaa ja yhdistää komentoja, kuvia ja kaavoja.

- Eclipse

  Javalla kirjoitettu IDE, joka tukee useita ohjelmointikieliä. Vaatii pyDev lisäosan python ohjelmointiin.

- SciPy

  Käytössä laajalti tiedepiireissä. Sisältää työkaluja matemaattisiin ja analyyttisiin ongelmiin, kuten matplotlib, NumPy ja Pandas.

- Ninja IDE

  Erikoistunut pythonkieliin. Hyvä laajennettavuus lisäosilla.

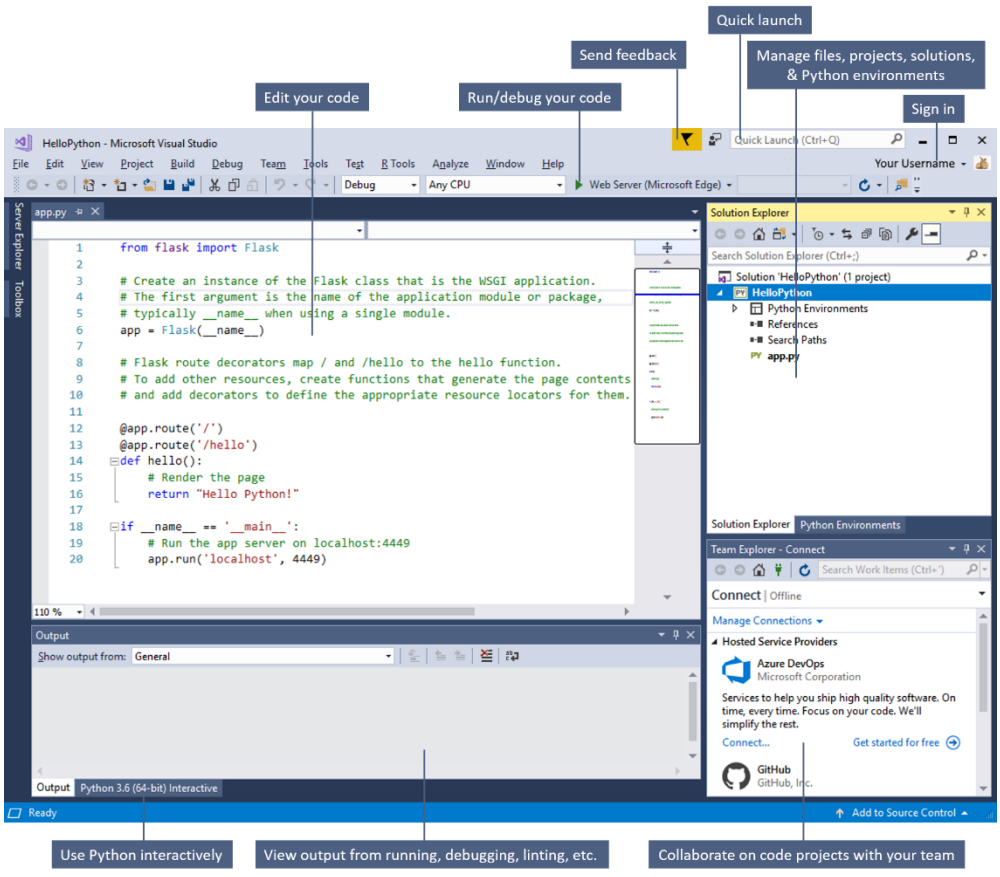
- Komodo IDE

  Maksullinen ympäristö, joka on kirjoitettu C++ ohjelmointikielellä. Tukee useita ohjelmistokieliä

- Visual studio

Työssä ohjelmointialustaksi valittiin Visual Studio 2019, koska VS sisältää myös laajan muiden ohjelmointikielien tuen sekä lisäosat. Tämä helpottaa ohjelmointiosuutta, koska tiedossa oli, että jossain määrin työssä käytetään myös muita ohjelmointikieliä. Kuitenkaan ohjelman käyttö ei poista mahdollisuutta käyttää muita ohjelmia koodin lukemiseen, koska python koodi ei muutu ohjelmien välillä. Visual Studio on Microsoftin tekemä ohjelma, jonka juuret kantavat yli 20 vuoden taakse. Visual Studiolla voidaan ohjelmoida käytetyimmillä ohjelmointikielillä.

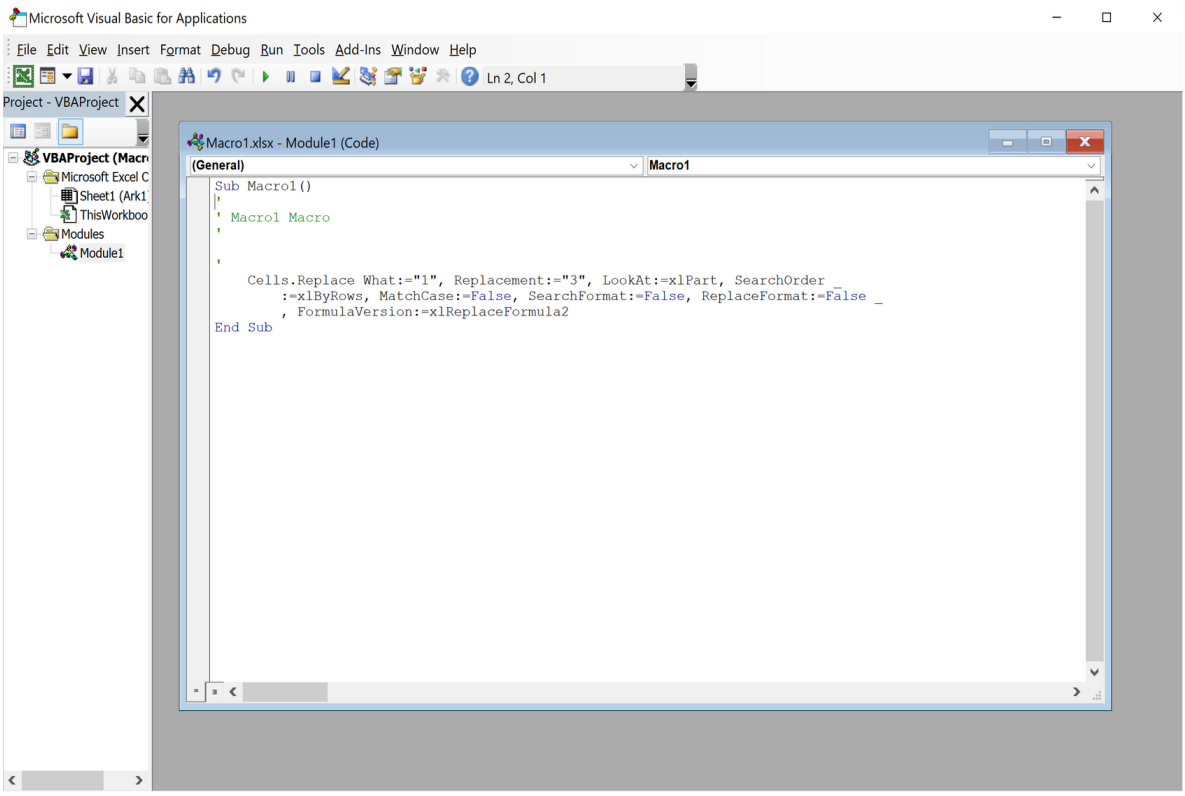
Visual studion python käyttöliittymän tuotetuet, muokattavuus ja monimuotoisuus tekee ohjelmasta hyvän ohjelmistoympäristön työlle (kuva6). Ohjelma tukee useita saman aikaista tulkkaustoimintoa, sisältää oman virheen ilmaisimen koodille ja laajat testausmahdollisuudet. (Microsoft. 2021)



Kuva 6 Visual studio käyttöliittymä (Microsoft. 2021)

3.3 VBA

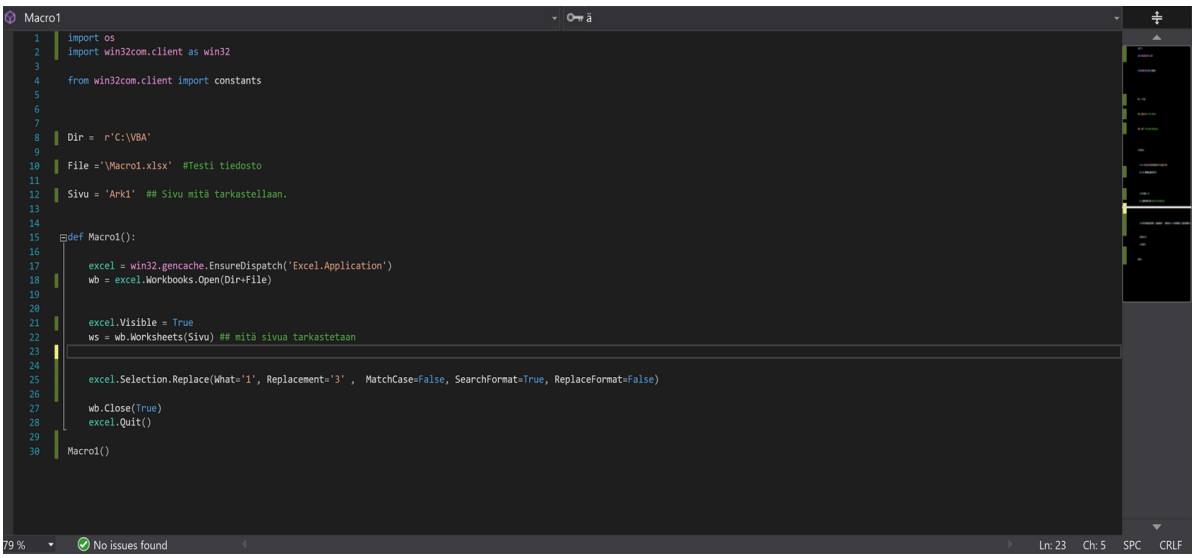
Visual Basic Applications on Microsoftin luoma ohjelmistokieli Office työkaluille. VBA:lle ei tarvita erillistä ohjelmointiympäristöä vaan se toimii Office-sovellusten sisällä. Hyötyjä kielen käytöstä on monia, perinteisten Excel-kaavojen lisäksi ohjelmoinnilla voidaan saavuttaa laajempia kustomoituja sovelluskokonaisuuksia. Microsoft-sovelluksista VBA on eniten käytössä Excel-taulukkolaskentaohjelmassa. VBA-ohjelmat luodaan ohjelman sisäisessä kehitysympäristössä, johon voidaan kirjoittaa suoraan komentorivejä tai tallentaa toimintoja makrojen avulla. (Tutorialspoint. 2021)



Kuva 7 Excel VBA-kehitysympäristö esimerkki. (Laukkanen. 2021)

Suora käännös makroista pythonkoodiin ei ole mahdollista (kuva7). Pythonilla voi kirjoitta VBA-makrojen kaltaisia toimintoja mutta komentorivit ovat hieman erilaisia (kuva8). On myös mahdollista avata ennalta tallennettu makrotiedosto ja suorittaa se pythonilla, mutta tähän työosuuteen se ei soveltunut. Lisäosilla kuten pyxll tai pywin32 voidaan automatisoida Windows Office sovelluksia.

Esimerkki kuvissa ei suoriteta tallennettua makroa pythonkoodilla, vaan ajetaan vastaavanlainen toiminta käännettynä pythonkieleen. Ensin esimerkissä määritetään käytettävä tiedosto. Sen jälkeen koodi avaa Excel-ohjelman ja suorittaa halutun toiminnon. Makrossa etsitään taulukosta kaikki numerot yksi ja muutetaan ne numeroiksi kolme. Tämän jälkeen koodi sulkee ohjelman. Koodi voidaan myös suorittaa taustalla, jolloin käyttäjälle ei tule näkyviin Excel-ohjelmistonäkymää. Tällä tavoin saadaan ohjelmiston käyttämää kuormitusta pienemmäksi.



Kuva 8 VBA-makro muunnettuna python komennoiksi. (Laukkanen. 2021)

Pywin32 on kehitetty 2000-luvun alussa Windows sovellusten automatisointiin ohjelmistokomponenttimallin ”COM component object model” avulla. Kirjaston avulla pythonkoodi voi ohjata Windows-ohjelmia. Käytännössä koodi avaa halutun ohjelman ja suorittaa toimenpiteet ohjelman sisällä. (Practical Business Python. 2020)

Huonoja puolia lisäosassa on se, että kahden päällekkäisen ohjelman käyttäminen vaatii prosessilta enemmän suoritustehoja. Myös koodin luominen on hieman kankeaa, joka lisää virhealttiutta koodille. Lisäosaa käytettäessä täytyy ottaa huomioon, jos työpöydällä on useita samoja ohjelmistosivuja auki. Tällöin ohjelman avaus tai sulkeminen voi aiheuttaa virhetoiminnon. Toisin sanoen kirjaston käyttö vaatii tiettyjä suuntaviivoja tai ehtoja koodiin. Ehdoilla saadaan ohjelmisto pysymään vakaana ja minimoimaan mahdolliset virheet.

4 TOTEUTUS

Ohjelmiston luonti aloitettiin keräämällä projekteissa käytettyjä luetteloita sekä piirustuksia pohjaksi ohjelmalle. Tällä tavoin saatiin luotua raamit, josta koodi lukee dataa ja ottaa huomioon pohjan rakenteelliset ominaisuudet. Dokumenteissa tutkittiin samankaltaisuuksia ja työssä valittiin yksi I/O-luettelo, jota alettiin työstämään pohjaksi koodille. Useamman pohjan mukaan ottaminen ohjelman luontivaiheessa sekoittaa ja hankaloittaa ohjelman tekoa, koska muuttujia pitäisi ottaa enemmän huomioon.

Heti projektin alussa huomattiin, että ohjelmasta kasvaa satojen jopa tuhansien rivien kokonaisuus, joten koodit jaettiin käyttötarkoituksen mukaan aliprosesseihin. Hyöty koodien jakamisesta näkyi välittömästi, koska halutut muuttujat saatiin siirrettyä pääohjelmaan tarvittaessa mutta kaikki ohjelmat toimivat myös itsenäisesti. Tämä helpotti ohjelman virheenkorjausta eli debuggausta. Aliohjelmissa tietojentarve vaihteli, koska kaikkia muuttujia ei tarvinnut kierrättää jokaisen ohjelmisto-osan kautta. Aliprosessit toimivat python funktioiden ja luokkien tavoin, jotka käynnistyvät vain kutsuttaessa. Näin käyttäjän määritelmät sekä ehtolauseet koodin sisällä, lajittelivat mitä koodin osaa käytettiin. Tämä mahdollisti sen, että ohjelmiston kuormitus joko serverillä tai käyttöpäätteellä saatiin minimoitua.

Aineistoja tutkittiin ohjelman avulla eri näkökannoista. Pandas kirjastolla saatiin luotua tietokantoja, joihin pystyttiin erottelemaan tiedot alkuperäisen aineiston pohjalta. Myös suodatettuja tietokantoja jatkojalostettiin erilaisiin käyttöihin. Testipohjasta huomattiin, että Excel-rakenteissa on eroja. Joissain tilanteissa koodin suorittaminen kesti huomattavasti kauemmin, vaikka datan määrä oli kutakuinkin sama. Tämä johtuu siitä, miten Excel-taulukot on luotu. Käyttäjänäkymä voi olla samanlainen, mutta Pandas lukee myös taulukkoihin määritetyt tyhjät arvot. Tällöin jos ohjelmassa esimerkiksi haetaan tiettyjä sanaa tai numeroa, koodi käy läpi jokaisen sarakkeen. Näissä tilanteissa kannattaa suodattaa ensin käyttämättömät solut ja sarakkeet pois dropNa tai isNa toiminolla.

4.1 Tarjoukset asiakkaille

Tarjouslaskentaan haluttiin yhdenmukainen pohja, joka täytetään asiakkaan tiedoilla. Pohjaan tulostetaan tarjous, komponenttityyppien ja IO-pisteiden mukaan. Epäselvissä tapauksissa käyttäjän tulee tarkistaa ja muuttaa tietoja tarvittaessa.

Laskennassa komponenttien tarkempaa analyysia ei tarvitse. Laskennassa selvitetään millainen komponentti ja liitäntätapa on kyseessä. Näin saadaan muodostettua piirrettävien piirien määrä. Ohjelmaan tehtiin suppeampi tietokanta, joka tutki onko aineistossa virheitä kuten duplikaatteja tai puuttuuko jotain tärkeitä tietoja. Tämän jälkeen ohjelma lajittelee ja laskee komponentit tyypeittäin ja tallentaa tarjouspohjan asiakkaan tiedoilla.

Tarjouspohjan täytyi sisältää vähintään seuraavat tiedot:

- Toimittajan tiedot
- Päivämäärät
- Tilaajan tiedot
- Piirienmäärä
- Laskentaperusteet
- Hinnat
- Sopimusehdot
- Toimitusaika/tapa
- Lisäehdot

Ohjelmoinnissa pystyttiin hyödyntämään docx-kirjastoa, joka muodosti Microsoft Word .docx-tiedostoformaatteihin kirjoitus ja jäsennykset suoraan ohjelman kautta. Datetime paketin avulla saatiin tarjouksen päivämäärät sekä voimassaoloajat ajan tasalle. Tämän jälkeen täytetty tarjous tuli käyttäjän tarkastettavaksi ja tallennettiin pdf-formaattiin.

4.2 Tietoturva

Asiakkaan lähettämät esiaineistot tulevat joko pakattuina tai yksittäisinä tiedostoina käyttäjälle. Uhka viruksista ja haittaohjelmista on nykypäivänä kasvava, jolloin tuntemattomien materiaalien tarkastus suositeltavaa. Mahdolliset tarkastustavat vaihtelee käyttötarpeen mukaan. Ohjelmistoja löytyy muun muassa Online eli internetissä tapahtuvista tarkastuksista maksullisiin virustorjuntaohjelmiin.

Vaikka verkossa tapahtuvat virustarkastukset ovat suurien ja tunnettujen yritysten pitämiä, niin mahdollisuus asiakastietojen leviämiseen on olemassa. Tästä syystä ohjelmaan lisättiin virustarkastus ominaisuus, joka kutsuu toista käyttöpäätteelle asennettua virusturvaohjelmaa suorittamaan kohdetiedoston tarkastuksen.

Työssä tutkittiin myös mahdollisuutta luoda virtuaalikäyttöjärjestelmä ohjelmalle, jota käytetään ohjelmiston käyttöalustana. Käyttöjärjestelmässä materiaalien avaus ja lukeminen ei tapahtuisi fyysisellä käyttöpäätteellä, jolloin mahdollisesti saastuneet tiedostot eivät pääsisi sisään fyysiseen käyttöjärjestelmään. Kuitenkin virtualisointi vie aikaa, joten se ei riittänyt opinnäytetyön rajapintaan. Toki myös oman virusturvaohjelmiston kirjoittaminen on myös mahdollista, mutta suurin ongelma näissä ohjelmissa on testausvaiheessa, missä ohjelmaa pitäisi testata varsinaisilla saastuneilla tiedostoilla. Testauksessa pitäisi päästä käsiksi oikeisiin haitallisiin tiedostoihin hallitusti. Ohjelma vaatisi myös jatkuvaa päivitystä ja kehitystä, joka ei olisi käytön kannalta kustannustehokasta.

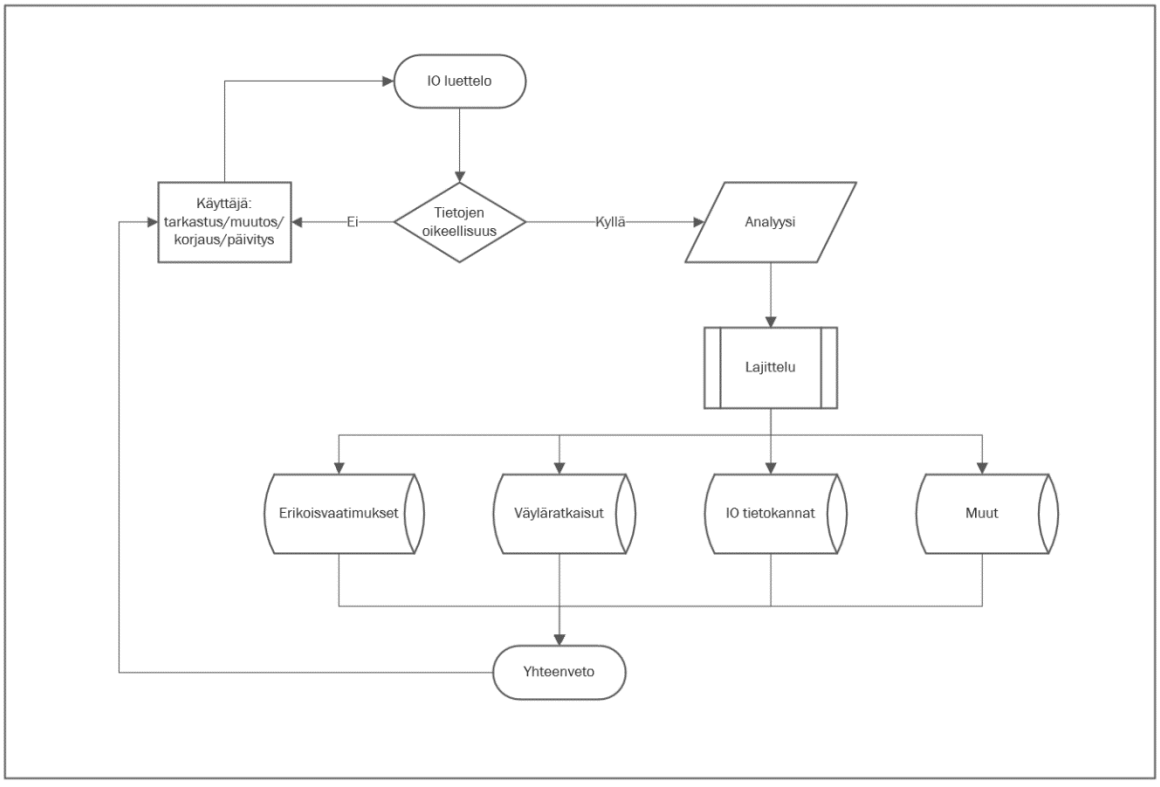
4.3 Analyysitoiminnot

Suurin kokonaisuus ohjelmasta keskittyi analyysitoimintoihin. Toiminnoilla pyrittiin etsimään virheitä, puutteita sekä lajittelemaan aineistoja pohjatiedoista. Usein alustavista I/O-luetteloista puuttuu kriittisesti tärkeitä tietoja, joita tarvitaan kaavioiden piirtoa varten. Tämä täytyi ottaa huomioon tietokantojen luonnissa, koska vääriä tietoja sisältävä kuva on käyttökelvoton.

Ohjelman hakutoiminnoissa täytyi ottaa huomioon case sensitive eli merkkikoko riippuvuus. Joissain tapauksissa ohjelma etsii otsikoista tai aineistosta tarkkoja string-merkkijonoja eli tekstejä. Tällöin isojen- ja pientenkirjainten tai välimerkkien määrä vaikuttaa hakutoimintoihin.

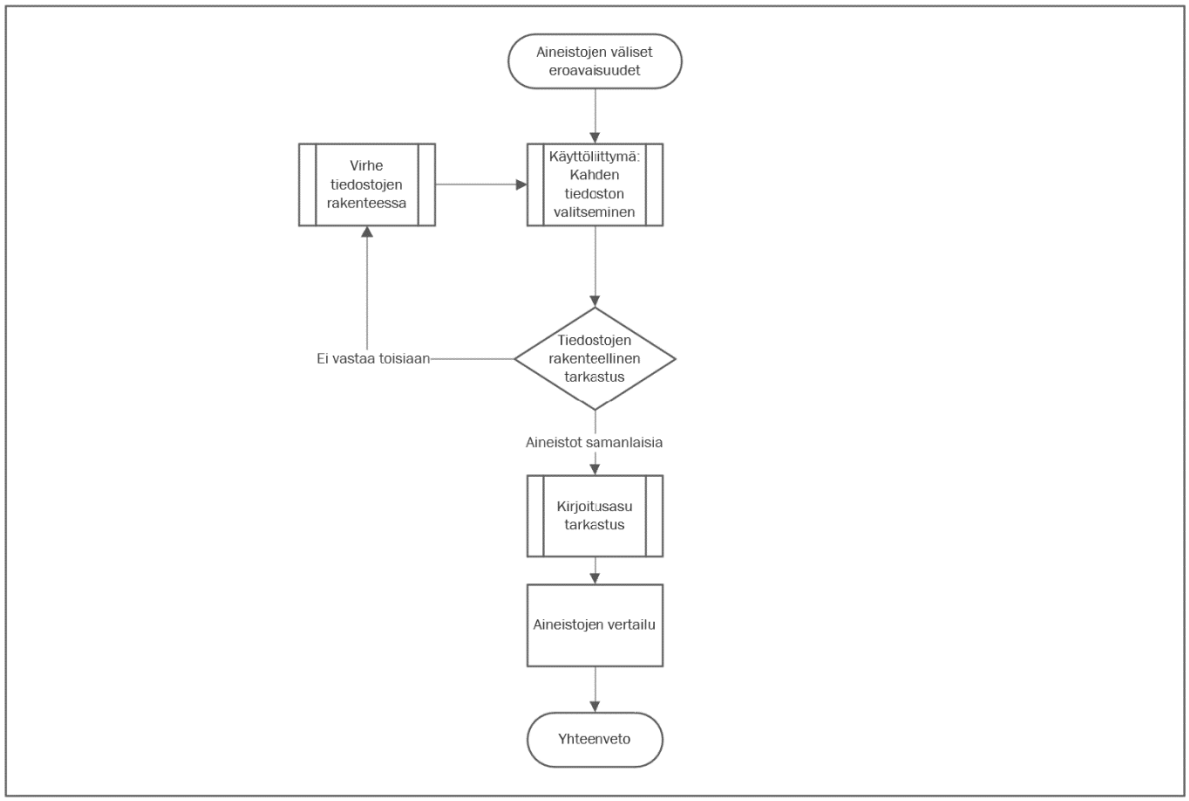
Virheen tarkistukseen kuului muun muassa duplikaattien etsiminen, jossa tarkasteltiin, onko useita kenttälaitteita identifioitu samalla positiolla. Lisäksi PLC-kortti ja kanava paikoissa ei saa olla päällekkäisyyksiä. Analyysista talletettiin erillinen raportti, jonka avulla käyttäjä pystyi lisäämään tai muuttamaan ennakkotietojen dataa. (kuva 9)

Virheiden lisäksi pyrittiin kartoittamaan epäselvät komponentit, kuten onko samassa laitteessa käytetty sekaisin IO- ja väyläkytkentöjä. Tai sisältyykö laitteeseen erikoisvaatimuksia kuten ATEX-luokituksia.



Kuva 9 Virheiden tarkastuksen prosessikulku. (Laukkanen. 2021)

Projektien edetessä luetteloista tehdään useita revisioita, joista tietoja on poistettu. Useinta vanhat tiedot jätetään luetteloihin ylivedettynä. Lisäyksissä ja muutoksissa käytetään väri-indikointia, jolloin laitteen muuttuneet tiedot ovat sarakkeessa eri väriset. Pandas on data-analyysissa hyvä työkalu, mutta se ei suoraan tunnista fonttimuunnoksia, värejä tai ylivetoja. Ongelma voidaan ratkaista joko toisella python lisäosalla, käyttää ensin Microsoftin VBA-ohjelmointikieltä tai Microsoft-ohjelmien omia makroja. Edellä mainitut toiminnot pitää sisällyttää luotuun ohjelmaan. Esilajittelulla saadaan poistettua tiedot, jotka eivät kuulu uusimpaan versioon, jonka jälkeen analyysityökalu määrittää millaisia muutoksia kahden tiedoston välillä on. (kuva 10)



Kuva 10 Aineistojen vertailuprosessi. (Laukkanen. 2021)

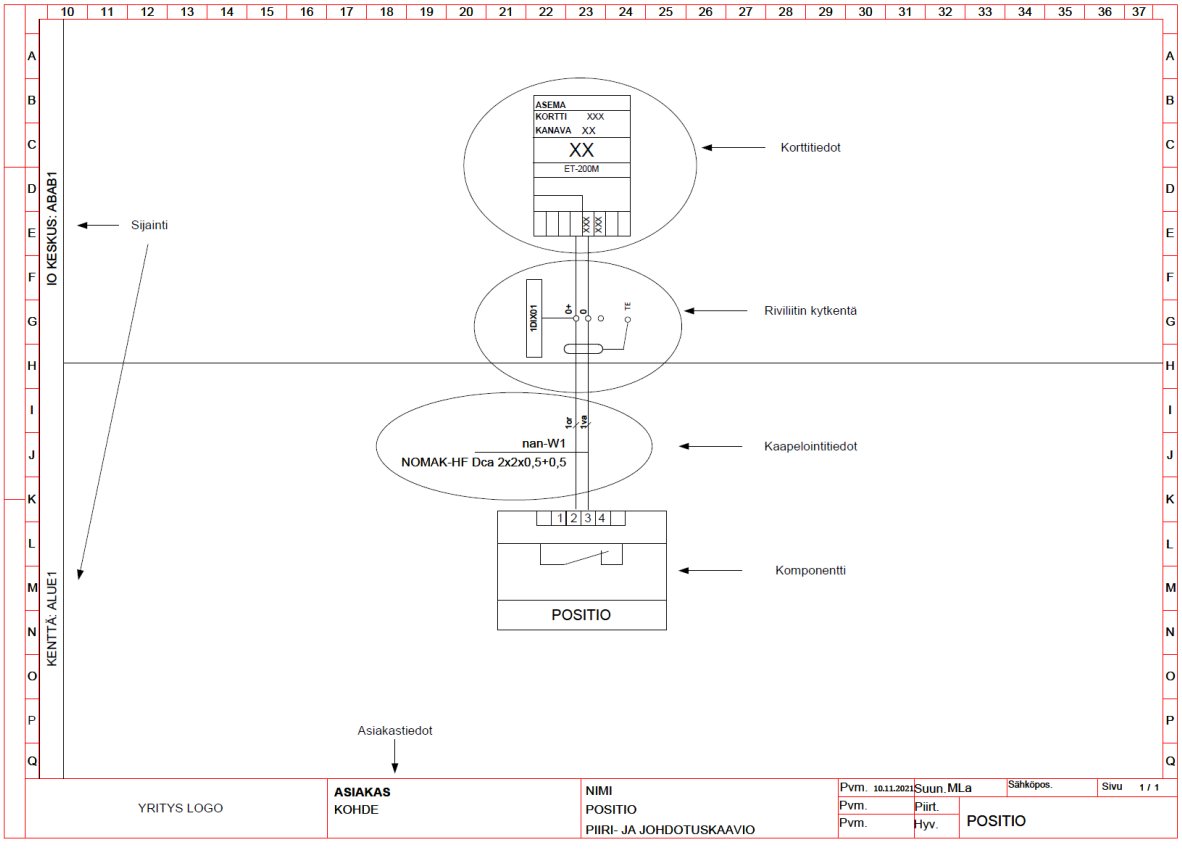
Koodin jaottelun ansiosta analyysityökaluja pystyttiin erittelemään ja muuntamaan ne .exe tiedostoiksi, jolloin käyttäjä voi tehdä materiaalien tarkastuksen ilman ohjelmistoympäristöä. Exe lyhenne tulee sanasta: ”executable file” eli suoritettava tiedosto. Käyttöliittymäksi tehtiin yksinkertainen graafinen ohjelmistoikkuna, jossa käyttäjä valitsee tarkasteltavat tiedostot ja ohjelma tallentaa raportin lopputuloksista.

4.4 Piirikaaviotietokanta

Piirikaavioita varten luetteloista piti hakea tietoja riippuen kenttälaitteen tyypistä. Työssä tehtiin kytkentäluettelot usein käytetyille PLC-korteille. Kytkentäluetteloiden tiedot siirrettiin suodatettuun komponenttitietokantaan. Tietokantaan oli kerätty kaikki saman tyyppiset instrumentit. Ohjelma tutki ja vertaili pohjatietoja siten, että jos ennakkomateriaaliin ei ollut täytetty logiikkakorttien tyyppejä, kanavia ja liitäntäpisteitä, tietokantaan lisättiin korteille seuraavat vapaat paikat. Tietokanta sisälsi muun muassa jokaisen instrumentin seuraavat tiedot:

- Asiakastiedot
- Sijainti
- Kaapelointityyppi
- Korttityyppi
- Kortin kanava
- Kortin liitintiedot
- Instrumentin nimi
- Instrumentin kytkentä tapa: aktiivinen/passiivinen
- Mahdolliset riviliitin tiedot
- Lisätietoja, kuten kenttälaitteen skaalaus arvot

Täydennetyn luettelon pohjalta voitiin luoda joko piirikaavio tai kilpi/kaapeliluettelo instrumenteille.



Kuva 11 Piirikaavio esimerkki instrumentin tietosisällöstä. (Laukkanen. 2021)

5 POHDINTA JA YHTEENVETO

Opinnäytetyön tavoitteena oli luoda ohjelmisto, piirikaaviosuunnittelun sekä myynnin avuksi. Työ analysoi ja automatisoi työvaiheita, jolloin samojen asioiden toistaminen saatiin minimoitua. Aihe oli todella mielenkiintoinen mutta myös haastava. Työ käynnistyi kesällä 2021 ja aloituspalavereissa saatiin luotua rajapinnat mitä työ pitää sisällään. Alun perin työn kokonaisuus olisi ollut hieman laajempi mutta jo aikaisessa vaiheessa pystyi huomaamaan kehitysprojektin työmäärän olevan suuri, joten työn rajapintoihin tehtiin muutoksia.

Ohjelmisto kehitys on aikaa vievää ja lisäämällä siihen sähköteknisen teorian tutkimisen opinnäytetyön laajuus alkoi kasvamaan todella laajaksi. Tästä syystä työssä keskityttiin pienempiin osiin, joista suunniteltiin sellaisia, joita voidaan käyttää ohjelmiston jatkokehityksessä. Ohjelma saatiin siihen pisteeseen, että ensimmäinen versio työstettiin valmiiksi ja virhetestaukset aloitettiin. Ilman minkäänlaisia pohjatietoa sähkösuunnittelusta tai ohjelmoinnista työ olisi jäänyt vajavaiseksi.

Analyysityökalujen suunnittelu kehitti erilaisia näkökantoja mitkä asiat vaikuttavat projektien läpivientiin. Pienikin virhe dokumentoinnissa voi viivästyttää projektin kulkua tai jopa pysäyttää sen. Varsinkin näin korona aikana komponenttien toimitusajat ovat olleet todella pitkät. Väärät kytkennät voivat aiheuttaa komponentin rikkoutumisen ja uuden osan toimitusajat voivat olla useita kuukausia. Ohjelmanluominen ja kartoitusosiot kehittivät laajasti omaa taitoa tutkia erilaisia materiaaleja ja löytää niistä mahdollisia ongelmakohtia.

Lopullisessa testausvaiheessa ohjelma liitettiin toiseen koodin, jolla saatiin piirrettyä instrumenttikuvia luotujen tietokantojen pohjalta. Pienten muutosten jälkeen ohjelma toimi moitteettomasti piirikaavioiden piirrossa. Analyysityökalujen osalta testauksia on jatkettu oikeilla projektimateriaaleilla projektien ohessa. Testauksien ansiosta mahdolliset muutostarpeet, lisäykset ja virheet tulevat esiin.

6 LÄHTEET

Technopedia. 2021. Technology dictionary. Verkkojulkaisu. https://www.techopedia.com/definition/. Viitattu 11.10.2021

Microsoft. 2021. Welcome to the Visual Studio IDE | Python. Verkkojulkaisu. https://docs.microsoft.com/en-us/visualstudio/python/visual-studio-ide?view=vs-2022. Viitattu 1.11.2021

Autodesk help. 2020. About Defining and Attaching Block Attributes. Verkkojulkaisu. https://knowledge.autodesk.com/support/autocad/learn-explore/caas/CloudHelp/cloudhelp/2019/ENU/AutoCAD-Core/files/GUID-67A2DDAD-2217-412F-8AEF-D4495192F45B-htm.html. Viitattu 15.11.2021

Suomen Automaatiopalvelu Oy. 2018. palvelumme. Verkkojulkaisu. https://www.suomenautomaatiopalvelu.fi/palvelumme/. Viitattu 5.10.2021

SFS-EN 61082-1. Sähkötekniikassa käytettävien dokumenttien laatiminen, Osa 1: Säännöt, 2015. Helsinki: Suomen standardisoimisliitto SFS ry. Viitattu 25.10.2021

TL121105 AUTOMAATIOTEKNIIKKA 1. 2009. Oulun seudun ammattikorkeakoulu. Verkkojulkaisu. http://www.tekniikka.oamk.fi/~terohi/auto1_s2009u.htm. Viitattu 15.1.2022

IBM Cloud Education. 2020. Application Programming Interface (API). Verkkojulkaisu. https://www.ibm.com/cloud/learn/api. Viitattu 15.11.2021

Nelli, Fabio. 2018. Python Data Analytics. Italy. Viitattu 1.11.2021

Wes McKinney, Pandas Development Team. 2021. pandas: powerful Python data analysis toolkit. Verkkojulkaisu. https://pandas.pydata.org/docs/pandas.pdf. Viitattu 2.11.2021

Python software foundation. 2011. PEP-405 – Virtual Environments. Verkkojulkaisu. https://docs.python.org/3/whatsnew/3.3.html#pep-405-virtual-environments. Viitattu 5.11.2021

Python software foundation. 2021. Graphical User interface with Tk. Verkkojulkaisu. https://docs.python.org/3/library/tkinter.html. Viitattu 5.11.2021

Suomen Automaatioseura ry. 2007. Automaatiosuunnittelun prosessimalli. Verkkojulkaisu-. https://www.automaatioseura.fi/site/assets/files/1426/automaatiosuunnittelun_prosessimalli.pdf. Viitattu 6.11.2021

Tutorialspoint. 2021. VBA tutorial Verkkojulkaisu. https://www.tutorialspoint.com/vba/index.html. Viitattu 5.11.2021

Practical Business Python. 2020. Automating Windows Applications Using COM. Verkkojulkaisu https://pbpython.com/windows-com.html. Viitattu 5.11.2021