

# DIGIVALMISTUSHANKKEEN TULOSTUSSOLUN AUTOMAATIOSUUNNITTELU

Lahden ammattikorkeakoulu

LAHDEN AMMATTIKORKEAKOULU  
Insinööri (AMK)  
Kone- ja tuotantotekniikka  
Kevät 2019  
Ville-Pekka Polvi

## Tiivistelmä

Tekijä(t) Polvi, Ville-Pekka	Julkaisun laji Opinnäytetyö, AMK	Valmistumisaika Kevät 2019
	Sivumäärä 35 + 8 liitesivua	
Työn nimi <b>Digivalmistushankkeen tulostussolun automaatio suunnittelu</b> Lahden ammattikorkeakoulu		
Tutkinto Insinööri (AMK)		
<p>Tiivistelmä</p> <p>Opinnäytetyön tavoite oli suunnitella ja toteuttaa Lahden ammattikorkeakoulun Digivalmistushankkeen 3D-tulostussolun automaatiojärjestelmä. Hankkeen tavoitteena on kehittää uusia 3D-tulostusmenetelmiä ja kartoittaa niiden käyttötarkoituksia alueen yrityksissä.</p> <p>Järjestelmä koostuu muovin prosessointijärjestelmästä ja nivelvarsirobotista, jotka yhdessä toimivat 3D-tulostusmateriaalin tuottajana ja tulostimena. Muovin prosessointijärjestelmä tuottaa määräämättömässä muodossa olevasta muovista muovirouhetta, joka sulatetaan ja jaetaan pursottimille. Nivelvarsirobotti pursottaa muovia sen työkalussa olevalla hydraulisylinterillä.</p> <p>Opinnäytetyön aihealue käsitti järjestelmän logiikkaohjelmoinnin, robottiohjelmoinnin sekä käyttöliittymän ohjelmistokehityksen. Lisäksi opinnäytetyössä perehdyttiin automaatioprojektien eri toteutusvaiheiden käytäntöihin ja dokumentointiin.</p> <p>Muovin prosessointijärjestelmän ohjaus toteutettiin Siemens S7-1500 -sarjan ohjelmoitavalla logiikalla. Järjestelmän robotti on ABB:n IRB 6620, joka toimii järjestelmän tulostajana. Käyttöliittymä toteutettiin Windows Forms -applikaationa C#-ohjelmointikielellä. Kommunikoimiseen logiikan ja robottiohjaimen kanssa käytettiin S7.Net- ja ABB:n PC SDK -kirjastoja.</p> <p>Opinnäytetyön tuloksena on toimiva automaatiojärjestelmä järjestelmän valmiusasteeseen nähden työn päätyttyä. Työssä tuotetut ohjelmistot ja dokumentit toimivat pohjana hankkeen jatkokehitykselle.</p>		
Asiasanat automaatio, logiikka, PLC, robotiikka, ohjelmointi, 3D-tulostus, lisäävä valmistus		

## Abstract

Author(s) Polvi, Ville-Pekka	Type of publication Bachelor's thesis	Published Spring 2019
	Number of pages 35 pages + 8 attm. pg.	
Title of publication <b>Automation engineering of a printing cell in a 'Digital Manufacturing' -project</b> Lahti University of Applied Sciences		
Name of Degree Bachelor of Engineering		
<p>Abstract</p> <p>The goal of this thesis was to design and implement the automation system of a 3D-printing cell in a 'Digital Manufacturing' -project at Lahti University of Applied Sciences. The goal of the project is to develop new 3D-printing methods and research their usability in local companies.</p> <p>The system consists of a plastic processing system and an industrial robot, which together act as a producer of 3D-printing material and as a 3D-printer. The plastic processing system grates plastic of undefined form used as raw material. The grated plastic is then melted and distributed to extruder tubes. The industrial robot extrudes the molten plastic using a hydraulic cylinder in the robot's tool.</p> <p>The scope of the thesis consisted of programming the system's programmable logic controller (PLC), robot programming and software development of the user interface. In addition, the thesis explored different practices and documentation types in different phases of an automation project's execution.</p> <p>The controls of the plastic processing system were implemented via a Siemens S7-1500 -series PLC. The system's robot is an ABB IRB 6620, which acts as the printer in the system. The user interface was implemented as a Windows Forms -application, written in the C# programming language. To communicate with the PLC and the robot controller, S7.Net and ABB's PC SDK -libraries were used.</p> <p>The result of this thesis is a working automation system in respect of the project's progress in the time of completing this thesis. The software and documentation produced during this study serve as the basis for future development of the project.</p>		
Keywords automation, PLC, robotics, software development, 3D-printing, additive manufacturing		

## SISÄLLYS

1	JOHDANTO .....	1
2	DIGIVALMISTUSHANKE .....	2
3	3D-TULOSTUSSOLU .....	3
3.1	Lisäävä valmistus .....	3
3.2	Tulostusmateriaalin prosessointi .....	3
3.3	Automaatiojärjestelmä .....	4
4	AUTOMAATIOSUUNNITTELU .....	5
4.1	Suunnitteluprosessi elinkaarimallia tukien .....	5
4.2	Vaatimusten käsittely .....	6
4.3	Toimintakuvaus .....	7
4.4	Laitearkkitehtuuri .....	8
4.5	Ohjelmistoarkkitehtuuri .....	8
4.6	Katsaus IEC 61131 -standardiin .....	9
4.6.1	Ohjelmointikielet .....	9
4.6.2	Ohjelmarakenne ja datatyypit .....	10
5	LOGIIKKAOHJELMOINTI .....	12
5.1	Ohjelmointiympäristö ja -kieli .....	12
5.2	Hardware-konfigurointi .....	12
5.3	Logiikkaohjelman arkkitehtuuri .....	12
5.4	Käyttöliittymän käsittely .....	13
5.5	Käsiaiot .....	14
5.6	Kompaundointi .....	14
5.7	Tulostus .....	15
6	ROBOTTIOHJELMOINTI .....	17
6.1	Ohjelmointiympäristö ja -kieli .....	17
6.2	Robottiohjelman arkkitehtuuri .....	17
6.3	Kalibrointi .....	17
6.4	Tulostus .....	19
6.5	Pursottimen vaihto .....	20
7	KÄYTTÖLIITTYMÄKEHITYS .....	22
7.1	Ohjelmointiympäristö ja -kieli .....	22
7.2	Kirjastot .....	22
7.3	Käyttöliittymänäkymä .....	22

7.4	Yhteyden muodostaminen .....	23
7.5	Muuttujien luku ja kirjoitus .....	23
7.6	G-koodin jäsentely ja tulostusratojen lähetys .....	24
7.7	Automaattiajo .....	26
7.8	Käyttöohje .....	26
8	KÄYTTÖÖNOTTO .....	27
8.1	Taajuusmuuttaja .....	27
8.2	PI-komponentit .....	28
8.3	Koeajo .....	28
9	JATKOKEHITYS .....	29
9.1	Jäljitettävyys .....	29
9.1.1	Ohjelmien kommentointi .....	29
9.1.2	Arkkitehtuuridokumentit ja toimintakuvaus .....	30
9.2	Toteutumattomat järjestelmän osat .....	30
9.3	Kehityskohteita .....	30
9.3.1	Logiikkaohjelma .....	30
9.3.2	Robottiohjelma .....	31
9.3.3	Käyttöliittymä .....	31
10	YHTEENVETO .....	32
	LÄHTEET .....	33
	LIITTEET .....	36

## 1 JOHDANTO

Opinnäytetyön aiheena on Lahden ammattikorkeakoulun Digivalmistus-hankkeen robotisoidun 3D-tulostussolun automaatio suunnittelu. Työ pitää sisällään hankkeen logiikkaohjelmoinnin, robottiohjelmoinnin sekä käyttöliittymän ohjelmistokehityksen. Työn tavoitteena on suunnitella ja toteuttaa automaatiojärjestelmä, joka prosessoi muovin tulostettavaan muotoon ja 3D-tulostaa kappaleita nivelvarsirobottia käyttäen.

Työn toteutushetkellä hanke oli siirtymässä toteutusvaiheeseen. Hankkeen pohjatyötä ja suunnittelua olivat tehneet Lahden ammattikorkeakoulun henkilöstö. Työn aloitushetkellä sulatusjärjestelmän mekaniikan toteutus oli jo pitkällä sekä suurin osa toimilaittevalinnoista tehty. Hankkeeseen liittyen oli samanaikaisesti meneillään opiskelijaprojekteja liittyen mekaniikan viimeistelyyn ja sähkösuunnitteluun.

Hanketta jatkokehitetään työn valmistuttua, joten opinnäytetyön aihealueissa kiinnitetään erityistä huomiota suunnittelun dokumentointiin. Suunnitteluprosessin edetessä käsitellään automaatioprojektien eri toteutusvaiheisiin liittyviä käytäntöjä, standardeja ja erilaisia suunnittelun tukena käytettäviä dokumentteja, joita sovelletaan hankkeen järjestelmän suunnittelussa. Syvennän myös ohjelmointiosaamistani jokaisessa työn osa-alueessa tutustumalla uusiin ohjelmointitekniikoihin ja toimilaitteisiin.

## 2 DIGIVALMISTUSHANKE

Digivalmistushanke on Lahden ammattikorkeakoulun tekniikan alan hanke, jota kuvataan LAMK:n hankkeen esittelysivulla seuraavasti:

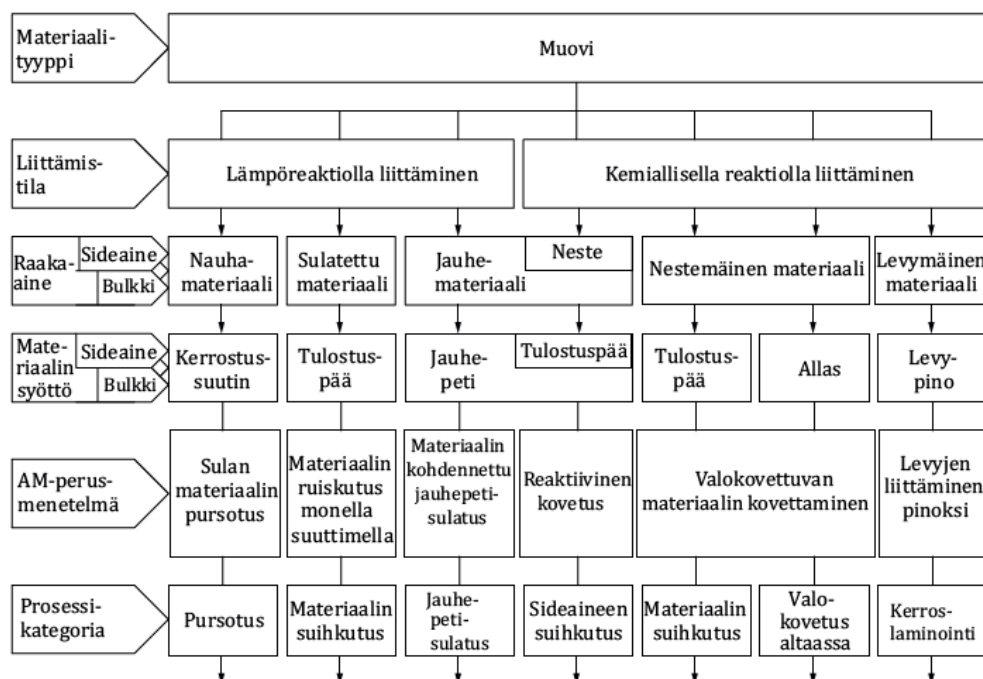
*DigiValmistus-projektissa kehitetään 3D-tulostusteknologioita, hyödynnetään kierrätysmateriaaleja ja kasvatetaan alueellista lisäävän valmistuksen osaamista. Lisäksi parannetaan 3D-tulostusmahdollisuuksia ja niihin liittyviä liiketoimintamahdollisuuksia Päijät-Hämeessä. Konkreettisena toimenpiteenä hankkeessa suunnitellaan ja rakennetaan robottiohjattava pursotin, jolla voidaan 3D-tulostaa monimutkaisia muotoja käyttäen raaka-aineena kierrätysmateriaaleja. (Lahden ammattikorkeakoulu 2019.)*

Robotisoitu 3D-tulostin on LAMK:n henkilöstön ja opiskelijoiden yhteishanke, johon liittyen on useita opiskelijaprojekteja järjestelmän toteutuksen eri osa-alueilta. Järjestelmä sijoitetaan LAMK:n tekniikan osaston robottisolun yhteyteen. Hankkeen toteutusaika on 1.9.2017 – 31.12.2019. Hankkeen on mahdollistanut Euroopan aluekehitysrahasto (EAKR), joka tukee muun muassa innovaatiotoimintaa ja uusien ympäristöteknologioiden kehittämistä. (Lahden ammattikorkeakoulu 2019; Työ- ja elinkeinoministeriö 2019.)

### 3 3D-TULOSTUSSOLU

#### 3.1 Lisäävä valmistus

Lisäävässä valmistuksessa (eng. Additive Manufacturing) muodostetaan valmistettava kappale valmistusprosessin raaka-aineesta lisäämällä materiaa työstöalueelle tyypillisesti kerroksittain. Yleisimmät materiaalit lisäävässä valmistuksessa ovat metallit, muovit, ke-raami ja erilaiset komposiitit. Materiaalin ominaisuuksista ja käytettävän raaka-aineen ti-lasta riippuen materian liitos yhtenäiseksi kappaleeksi voi tapahtua esimerkiksi sulatta-malla, kovettamalla tai sintraamalla. (SFS-EN ISO/ASTM 52900:2017, 16.)



Kuva 1. Muovimateriaaleiden yksivaiheisten AM-prosessien periaatteet (SFS-EN ISO/ASTM 52900:2017, 18)

Hankkeen järjestelmän prosessikategoria on muovin pursotusta (kuva 1). Järjestelmän raaka-aine tuotetaan omalla laitteistolla.

#### 3.2 Tulostusmateriaalin prosessointi

Tyypillisesti lisäävän valmistuksen järjestelmät tarvitsevat prosessin raaka-aineen jossakin määrättyssä muodossa, esimerkiksi jauheena, nauhana tai nesteinä (SFS-EN ISO/ASTM 52900:2017, 17). Hankkeen järjestelmä käyttää raaka-aineenaan muovirouhetta, joka tuotetaan muovin murskauslaitteistolla. Se antaa laajat mahdollisuudet valmistaa itse erilaisia komposiittimateriaaleja, ja laitteistoa ei ole sidottu mihinkään tiettyyn raaka-aineen muotoon.

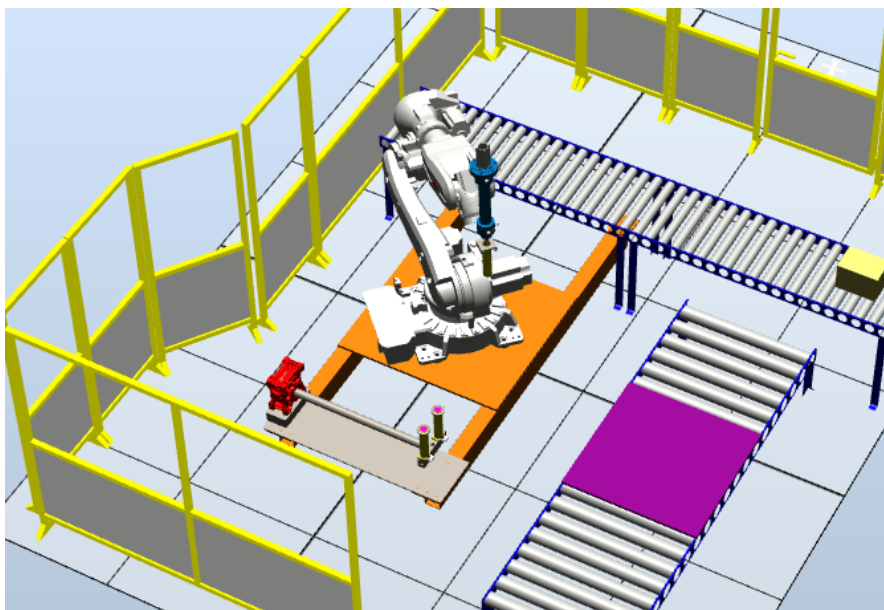


### 3.3 Automaatiojärjestelmä

Tulostussolun automaatiojärjestelmä ohjaa sulatuslaitteistoa ja robottia, sulattaen raaka-aineena käytettävää muovirouhetta ja tulostaa siitä kolmiulotteisia kappaleita. Muovin murskausjärjestelmä koostuu erillisistä laitteista, jotka ovat käyttäjäohjattuja. Sulatusjärjestelmä jakaa sulatetun muovin kahdelle pursottimelle. Pursottimien täytettyä nivelvarsi-robotti noutaa pursottimen työkaluunsa ja liikuttaa pursotinta generoitujen tulostusratojen mukaisesti. Robotin työkalussa on hydraulisylinteri, joka pursottaa sulaa muovia tulostusalustalle.

Järjestelmän pääasiallisena ohjausjärjestelmänä toimii ohjelmoitava logiikka. Logiikka ohjaa sulatusjärjestelmän toimintaa, pursottimen hydraulisylinteriä ja robotin toimintatilaa. Järjestelmän käyttöliittymänä toimii käyttöliittymä-PC:llä toimiva Windows-ohjelma, jolla ohjataan järjestelmän tilaa ja siirretään tulostusradat robotille.

Tulostettavien kappaleitten koko rajoittuu robotin ulottuvuuksien, sekä järjestelmän komponenttien sijoittelun mukaan robottisolussa. Kuvan 2 alustavan layoutin mukaan tulostettavat kappaleet voivat olla maksimissaan noin 1000 mm x 1000 mm x 1000 mm kokoisia.



Kuva 2. Solun layout RobotStudiassa

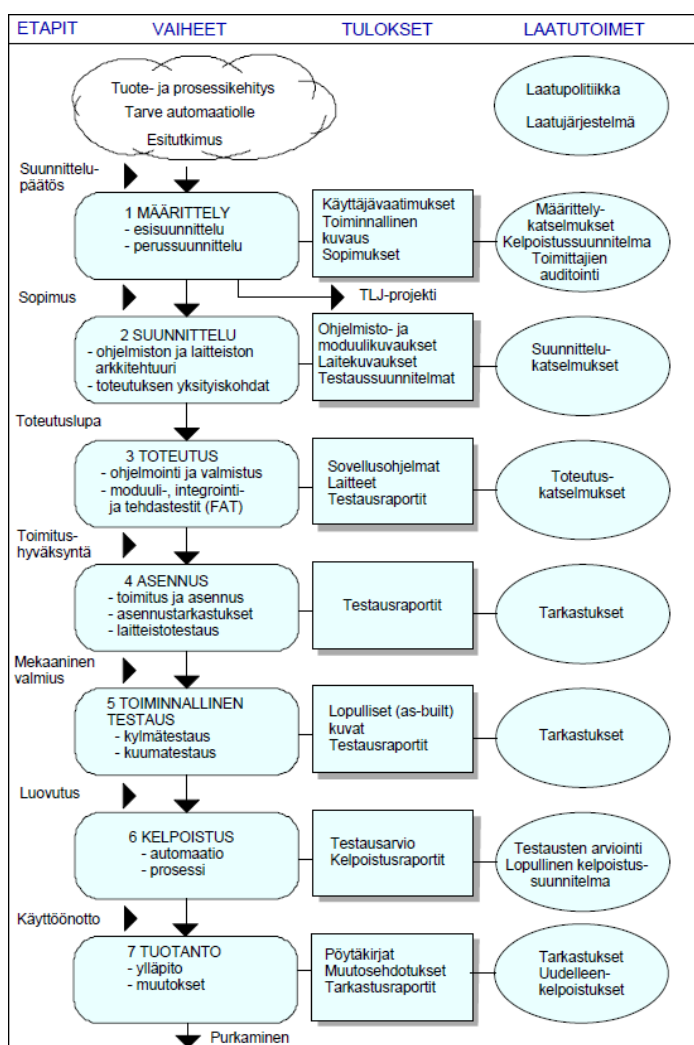
Tulostuskapasiteetti riippuu sulatusjärjestelmän pursottimien täyttökapasiteetista, pursottimen mekaanisista ominaisuuksista ja robotin suurimmista liikenopeuksista. Rajoittavan tekijän löytämiseksi ei ole vielä tehty tutkimusta, mutta todennäköisesti se on sulatusjärjestelmän kapasiteetti. Hankkeen edetessä kunkin osa-alueen toimintaa optimoidaan suurimman mahdollisen kapasiteetin saavuttamiseksi. Alustavaksi tavoitteeksi on valittu 216 000 mm<sup>3</sup>/min.

## 4 AUTOMAATIOSUUNNITTELU

### 4.1 Suunnitteluprosessi elinkaarimallia tukien

Yritysten tehdessä uusia hankintoja organisoidaan ne yleensä hankkeiksi. Hankkeen tavoite on tukea hankkeen liikkeellepanijan liiketoiminnallisia tavoitteita. Hankkeet ovat yleensä suurempi kokonaisuus kuin järjestelmän tekninen toteutus. Hankkeita voidaan kuvata järjestelmän elinkaarimallilla, joka sisältää esitutkimuksen, suunnittelun, käyttöönoton, tuotantokäytön, huollon ja tuotantokäytöstä poistamisen (kuva 3). (Haikala & Mikkonen 2011, 19.)

Hankkeiden toteutus voidaan jakaa osiin projekteilla. Tyypillisiä projekteja hankkeen läpiviemisessä ovat esitutkimus, määrittely ja toteutus. Riippuen hankkeen laajuudesta, varsinkin toteutusvaiheen osa-alueet voidaan jakaa osaprojekteihin. (Haikala & Mikkonen 2011, 20.)



Kuva 3. Automaatiojärjestelmän suositeltava elinkaarimalli laadun kannalta (Suomen Automaatioseura ry 2019, 17)

Esitutkimuksen toteuttaa tyypillisesti hankkeen liikkeellepanija, ja se voi sisältää toteutusvaihtoehdot, kannattavuusanalyysin, investoinnin riskinarvioinnin ja alustavan aikataulun (Haikala & Mikkonen 2011, 20-21). Määrittelyvaiheessa kuvataan tarkemmin, millaiset vaatimukset järjestelmän tulee täyttää ja pääpiirteittäin, millainen järjestelmän tekninen toteutus on. Määrittelyvaiheen tuotokset toimivat rajapintana projektin tilaajan ja toimittajan välillä. Määrittelyvaiheen lopputuotoksina ovat toimintakuvaus, joka kuvaa, miten tilaajan vaatimukset täytetään, sekä kelpoistussuunnitelma, joka kuvaa, millä käytännöillä vaatimusten tähtyminen todetaan. Toimitusvaiheessa toimittaja suunnittelee, valmistaa ja asentaa järjestelmän ohjelmistot ja laitteet hyväksytyn toimintakuvauksen pohjalta. Laitteen teknisen dokumentaation lisäksi tyypillisesti tuotetaan asennus- ja käyttöohjeet, testaussuunnitelmat ja testausraportit. Toimituksen jokaisessa vaiheessa järjestelmää kelpoistetaan kelpoistussuunnitelman mukaisesti. (Suomen Automaatioseura ry 2019, 18-19, 25.)

#### 4.2 Vaatimusten käsittely

Järjestelmän vaatimukset luovat perustan kaikille projekteille. Vaatimusten hyvä määrittely on yksi tärkeimmistä, mutta samalla haastavimmista osa-alueista projektinhallinnassa (Haikala & Mikkonen 2011, 61). Vaatimukset voidaan jakaa kolmeen luokkaan: toiminnalliset vaatimukset, ei-toiminnalliset vaatimukset ja reunaehdot. Toiminnalliset vaatimukset määrittävät järjestelmän toiminnot sekä suurpiirteisesti, minkä tietojoukkojen perusteella laite toimii ja millainen on järjestelmän laitearkkitehtuuri. Ei-toiminnalliset ominaisuudet määrittävät laitteen käyttöön ja ylläpidettävyyteen liittyvät ominaisuudet, kuten suorituskyky, huollettavuus ja laajennettavuus. Reunaehdot ovat vaatimuksia järjestelmän toteutustavasta, esimerkiksi jokin tietty teknologiavalinta. (Suomen Automaatioseura ry 2019, 37.)

Hyvän vaatimuksen ominaisuuksia ovat sen oikeellisuus, ymmärrettävyys, riittävä tarkkuus, testattavuus ja jäljitettävyys sekä asiakkaan tarpeeseen, että lopulliseen ratkaisuun. Jokaisessa projektin teknisessä toteutuksessa pitäisi pystyä osoittamaan, mihin käyttäjävaatimukseen ratkaisu perustuu ja miten vaatimuksen tähtyminen voidaan testata. Vaatimusten dokumentoinnin lisäksi vaatimusten pohjalta luodaan kelpoistussuunnitelma. Kelpoistussuunnitelmassa määritellään, millä menetelmillä automaatiojärjestelmän käyttäjävaatimusten tähtyminen todetaan. (Haikala & Mikkonen 2011, 64; Suomen Automaatioseura ry 2019, 38.)

### 4.3 Toimintakuvaus

Toimintakuvaus (eng. Functional Requirements Specification) on dokumentti, johon dokumentoidaan käyttäjävaatimusten pohjalta, millainen automaatiojärjestelmän täytyy olla täyttääkseen käyttäjävaatimukset. Se toimii rajapintana automaatioprojektin kaikkien osapuolien välillä sekä tilaajan, että toimittajan puolella. (Edwards 2019, 1.)

Toimintakuvaus ei ole pakollinen dokumentti, eikä sen rakennetta tai sisältöä ole standardoitu (Grek 2011, 50). Yleisesti automaatioprojekteissa jonkin tasoinen toimintakuvaus luodaan, koska huolellisesti laadittu toimintakuvaus voi vaikuttaa erittäin paljon projektin läpiviemisen sujuvuuteen. Toimintakuvauksen laajuus ja muoto voi vaihdella automaatioprojektin laajuuden ja laatuvaatimusten perusteella huomattavasti. Laajuudestaan ja rakenteestaan riippumatta toimintakuvauksen tulisi palvella kolmea tarkoitusta: järjestelmän toiminnallinen kuvaus, tiedonvälitys eri osapuolien välillä ja toimittajan vastuualeen määrittely. (Edwards 2019, 1)

Toiminnallinen kuvaus määrittää, millaisilla toiminnallisilla kokonaisuuksilla käyttäjävaatimukset aiotaan täyttää. Toiminnalliset osat pyritään kuvaamaan mahdollisimman yksityiskohtaisesti, mutta sitomatta niitä mihinkään tiettyyn teknologiavalintaan. Lähtökohtaisesti järjestelmän kuvaus tulisi olla toteutusriippumaton. Toiminnallisen kuvauksen tulisi myös käsitellä ei-toiminnalliset käyttäjävaatimukset ja reunaehdot. (Suomen Automaatioseura ry 2019, 44-45.)

Toinen toimintakuvauksen tarkoitus on mahdollisimman yhtenäisen käsityksen muodostaminen järjestelmästä projektin eri osapuolien välillä. Täten toimintakuvauksen tulisi kuvata järjestelmää kaikkien projektiin liittyvien tahojen kannalta mahdollisimman kattavasti. Kriittisimpien tietojen esiintuomiseksi kohderyhmät täytyy kuitenkin priorisoida ja toimintakuvauksen sisältö muotoilla niin, että se palvelee tärkeimpiä projektin osapuolia. (Edwards 2019, 1.)

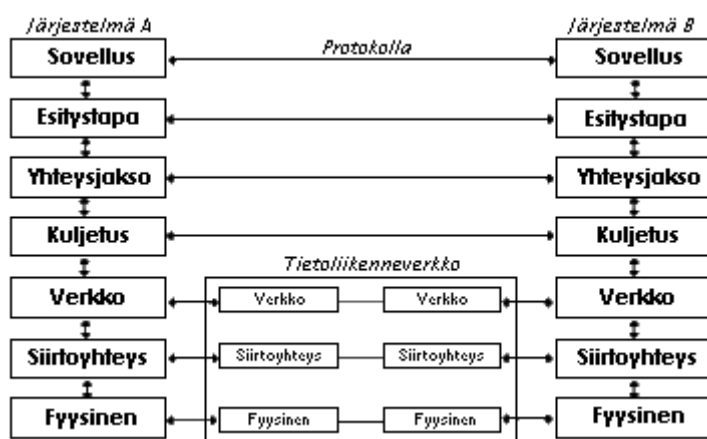
Toimintakuvauksesta tulee käydä ilmi selkeästi kriteerit järjestelmälle, jotka täyttäessään järjestelmä täyttää käyttäjävaatimukset. Toimintakuvausta tulisikin käyttää perustana projektin toteutusvaiheessa tapahtuville kelpoistuksille, kuten FAT:ille (Factory Acceptance Testing). (Edwards 2019, 2.)

Hankkeen järjestelmän toimintakuvauksen tavoitteena on kuvata järjestelmän päätoiminnot, laiteliittymät, reunaehdot ja ei-toiminnalliset ominaisuudet. Päätoimintoihin tarkempan kuvaukseen sisältyy järjestelmän toimintatilat, käsiajot ja turvatoiminnot. Lisäksi ei-toiminnallisista ominaisuuksista kuvataan huollettavuus ja ylläpidettävyyys. Järjestelmän

dokumentoinnin lisäksi toimintakuvausella varmistuttiin hankkeen vetäjien kanssa yhteisestä käsityksestä järjestelmän toiminnasta. Toimintakuvaus on liitteessä 1.

#### 4.4 Laitearkkitehtuuri

Tietoliikennejärjestelmien suunnittelussa käytetty OSI-malli (Open Systems Interconnection) jakaa tiedonsiirron 7 kerrokseen, jotka ovat hierarkkisessa järjestyksessä (kuva 4). Mallin alin kerros on fyysinen kerros, joka sisältää järjestelmän komponentit ja tiedonsiirto-kaapelit. (Tampereen teknillinen yliopisto 2002.)



Kuva 4. OSI-mallin kerrokset (Tampereen teknillinen yliopisto 2002)

Suunniteltaessa järjestelmiä jokainen kerros voidaan dokumentoida kaavioin. Hankkeen järjestelmän yleiseen kuvaukseen kuvan 4 kahden alimman tason kaavion malli sopi mielestäni parhaiten. Kaavion rakennetta ei ole standardisoitu, mutta siitä on hyvä tulla ilmi vähintään järjestelmän fyysiset komponentit ja niiden liittymät (Galler 2013). Järjestelmän laitearkkitehtuurikaavio pyrkii kuvaamaan laitteen fyysiset komponentit, niiden väliset kaapelointityypit ja dokumentoida IP-osoitteet. Laitearkkitehtuurikaavio on liitteessä 2.

#### 4.5 Ohjelmistoarkkitehtuuri

Arkkituurityylit ovat työkaluja suunniteltavan ohjelmiston rakenteen hahmottamiseksi. Se toimii myös dokumentaationa järjestelmän rakenteesta abstraktilla tasolla. Arkkitehtuurityylejä on useita, joista yleisin on kerrosarkkitehtuuri. Kerrosarkkitehtuurissa järjestelmän komponentit järjestetään kerroksiin, joilla on jokin kategoria (kuva 5). Kerroksilla on hierarkia, joka usein määräytyy abstraktiotasoin. Ylimmän kerroksen komponentit tuottavat ylemmän abstraktiotason toiminnallisuuksia alempien kerroksien alemman abstraktiotason toiminnallisuuksilla. Rakenne ei kuitenkaan ole sitova ja käytännössä järjestyksestä joudutaan usein poikkeamaan. Poikkeamia on kahdenlaisia: alemman tason palvelu voi kutsua ylemmän tason palvelua (eng. hierarchy breach) ja ylemmän tason palvelu voi ohittaa

kerroksia (eng. bridging). Kerrosarkkitehtuurin vapaamuotoisuudesta johtuen se soveltuu hyvin monenlaisten järjestelmien kuvaamiseen. (Koskimies & Mikkonen 2005, 125-128, 131.)

Käyttöliittymä
Sovelluslogiikka
Sovellusaluelogiikka
Tietokanta

Kuva 5. Esimerkki kerrosarkkitehtuurin käytöstä (mukailtu Koskimies & Mikkonen 2005, 128.)

Järjestelmän ohjelmistoarkkitehtuurikaavio pyrkii kuvaamaan jokaisen laitteen arkkitehtuurin, laitteiden ohjelmistojen komponentit ja eri laitteiden komponenttien väliset riippuvuudet. Kerrosten välisiä rajapintoja ei ole kaaviossa määritelty. Ohjelmistoarkkitehtuurikaavio on liitteessä 3.

#### 4.6 Katsaus IEC 61131 -standardiin

Ohjelmoitavien logiikoiden (PLC) vallitseva standardi on IEC 61131. Standardi on yhdeksänosainen. Standardin osat määrittelevät ohjelmoitavien logiikoiden laitteistoon, ohjelmointiin ja käyttöön liittyviä vaatimuksia ja ohjeistuksia. Ohjelmoinnin kannalta oleellisin osa on IEC 61131-3, joka määrittelee ohjelmoitavien logiikoiden ohjelmointikielet. (PLCopen 2019b.)

##### 4.6.1 Ohjelmointikielet

IEC 61131-3 -standardi mahdollistaa laitetoimittajasta riippumattoman ohjelmoinnin. Ohjelmointikieliä on yhteensä 5.

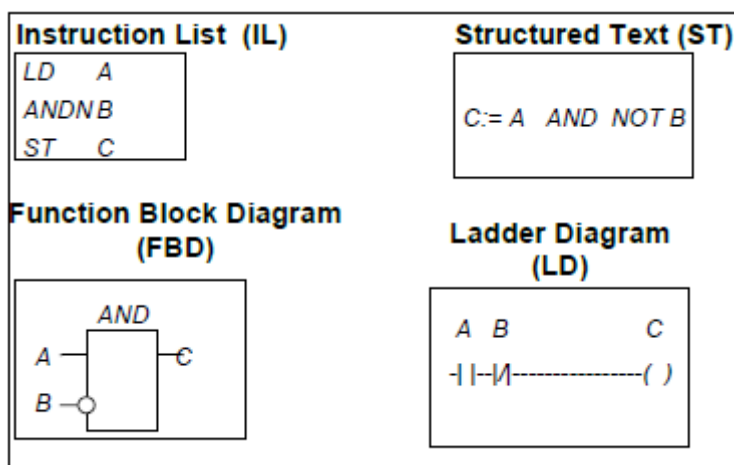
Graafiset kielet:

- Ladder diagram (LAD)
  - On vanhin ohjelmointikielistä. Ulkoasu pohjautuu piirikaavioihin ajalta, jolloin prosessiohjaukset toteutettiin releillä.
- Function block diagram (FBD)
  - Ulkoasu koostuu funktiolohkoista, joita ketjuttamalla luodaan toiminnallisuuksia.

- Sequential function chart (SFC)
  - Ohjelmat on jaettu sekvenssilohkoihin. Sekvenssien välissä on siirtymäehto, jonka täytyttyä siirrytään seuraavaan sekvenssiin. Ohjelmarakenteen suunnittelu muistuttaa ulkoasultaan toimintakaaviota. Sekvenssilohkojen toiminnot voidaan toteuttaa millä tahansa IEC 61131-3:n ohjelmointikielistä.

Tekstimuotoiset kielet:

- Instruction list (IL)
  - Käskylista on alhaisen tason tekstimuotoinen ohjelmointikieli. Rakenteeltaan se muistuttaa assemblyä.
- Structured text (ST)
  - Rakenteellinen teksti on korkean tason tekstimuotoinen ohjelmointikieli, jonka juuret ovat Ada-, Pascal- ja C-ohjelmointikielissä.



Kuva 6. Saman toiminnallisuuden toteutus eri ohjelmointikielillä (PLCopen 2019a, 2)

Kaikilla ohjelmointikielillä on toteutettavissa samat asiat (kuva 6). Ohjelmakielen valinta voi riippua muun muassa ohjelmoijan taustasta, yrityksen standardista, ratkaistavasta ongelmasta, ohjattavasta järjestelmästä tai muista sidosryhmistä, kuten kunnossapito-osaston taidoista käsitellä eri ohjelmointikielillä kirjoitettuja ohjelmia. (PLCopen 2019a, 3.)

#### 4.6.2 Ohjelmarakenne ja datatyypit

Standardissa määritellään ohjelmoitavien logiikoiden ohjelmarakenne. Ohjelmat koostuvat osista, joita ovat ohjelmat, funktiot ja funktiolohkot. Näitä osia kutsutaan POU:iksi (Program Organization Unit). Osilla on hierarkia; ohjelmat kutsuvat funktioita ja funktiolohkoja.

Ohjelman sisäisillä funktioilla ja funktiolohkoilla luodaan ohjelman sisäinen arkkitehtuuri. (PLCopen 2019a, 2.)

Funktioita kutsuttaessa ne suorittavat funktion ohjelmoidun toiminnallisuuden. Funktiolla voi olla funktion syötettäviä parametreja, tuloja, jotka luetaan funktion suorituksen alussa, sekä funktio voi antaa suorituksen jälkeen ulos parametreja, lähtöjä, jotka kirjoitetaan, kun funktion suoritus päättyy. Olemassa on myös näiden yhdistelmä, in-out, joka mahdollistaa funktion syötetyn parametrin arvon muuttamisen. Lisäksi funktioissa voidaan määritellä väliaikaisia muuttujia, joita kirjoitetaan ja luetaan funktion sisällä. (PLCopen 2019a, 2.)

Funktiolla ei ole varattua muistia. Funktion suorituksen loputtua sen lähdöt ja sisäiset muuttujat nollaantuvat. Täten samoilla syötetyillä arvoilla funktio palauttaa aina samat arvot. Funktiolohkon muuttujille on varattuna oma muistinsa, joten käytettäessä funktiolohkoja on mahdollista säilyttää lohkon tila eri suorituskertojen välillä. (Ua Automation 2019.)

Standardi määrittää ohjelmissa käytettävät datatyypit (taulukko 1). Muuttujien datatyyppien määrittäminen mahdollistaa automaattisen muistinvarauksen ja estää suoritusvirheitä yritettäessä suorittaa esimerkiksi laskutoimituksia yhteensopimattomilla datatyypeillä. (PLCopen 2019a, 1.)

Tyyppi ja kuvaus	Koko biteissä	Näyttömuodot	Alin arvo	Ylin arvo
BOOL (Bitti)	1	Teksti	FALSE	TRUE
		Numero	0	1
BYTE (Tavu)	8	Heksadesimaali	B#16#0	B#16#FF
WORD (Sana)	16	Binääriluku	2#0	2#1111_1111_1111_1111
		Heksadesimaali	W#16#0	W#16#FFFF
		BCD	C#0	C#999
		Desimaaliluku, etumerkitön	B#(0,0)	B#(255,255)
DWORD (Kaksoissana)	32	Binääriluku	2#0	2#1111_1111_1111_1111_1111_1111_1111_1111
		Heksadesimaali	W#16#0000_0000	W#16#FFFF_FFFF
		Desimaaliluku, etumerkitön	B#(0,0,0,0)	B#(255,255,255,255)
INT (Kokonaisluku)	16	Desimaaliluku, etumerkillinen	-32768	32767
DINT (Kaksoiskokonaisluku)	32	Desimaaliluku, etumerkillinen	L#-2147483648	L#2147483647
REAL (Liukuluku)	32	IEEE Liukuluku	+/-1.175495e-38	+/-3.402823e+38
S5TIME (SIMATIC aika)	16	S7 aika 10ms askelmin	S5T#0H_0M_0S_0MS	S5T#2H_46M_30S_0MS
TIME (IEC aika)	32	IEC aika 1ms askelmin	T#0D_0H_0M_0S_0MS	T#24D_20H_31M_23S_648MS
DATE (IEC päivämäärä)	16	IEC päivämäärä 1pv askelmin	D#1990-1-1	D#2168-12-31
TIME_OF_DAY (Kellonaika)	32	Kellonaika 1ms askelmin	TOD#0:0:0.0	TOD#23:59:59.999
CHAR (Merkki)	8	ASCII merkit	-	-

Taulukko 1. Step 7 perusdatatyypit (mukailtu PLCdev 2019)

Taulukon 1 perusdatatyypeistä ohjelmoijan on mahdollista luoda komplekseja datatyyppejä, kuten taulukoita, struktuureja ja käyttäjämääriteltyjä datatyyppejä.



## 5 LOGIIKKAOHJELMOINTI

### 5.1 Ohjelmointiympäristö ja -kieli

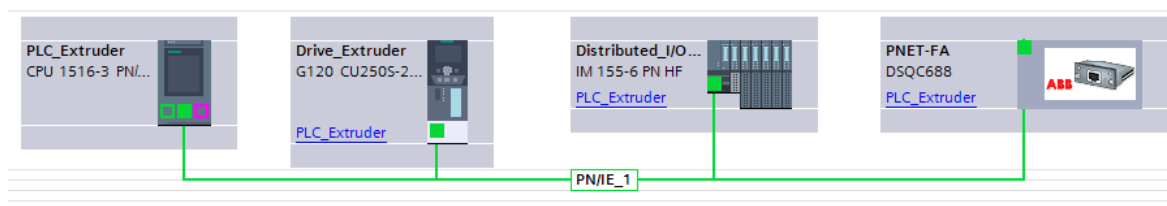
Logiikkaohjelmoinnin kehitysympäristönä käytin Siemensin TIA Portal:a. TIA (Totally Integrated Automation) Portal sisältää Step 7 -ohjelmiston Siemensin logiikoiden ohjelmointiin, WinCC-ohjelmiston käyttäjäpaneelien ja valvomo-ohjelmien ohjelmointiin, sekä Start-drive -ohjelmiston taajuusmuuttajien parametrintointiin. (Siemens AG 2019.)

Koko logiikkaprojekti toteutettiin Ladder-ohjelmointikielellä. Ladder on logiikkaohjelmoinnissa opetuksen pohjalta tutuin kieli, eikä ohjelmassa ole osia, joissa muun ohjelmointikie-  
len käyttö olisi tuonut merkittävää hyötyä.

### 5.2 Hardware-konfigurointi

Aloittaessa uusi projekti tulee määrittää, millaiselle järjestelmälle ohjelmaa tehdään. 'Devices & networks' -osiossa määritellään järjestelmän komponentit, niiden väliset yhteydet sekä laitteiden tulojen ja lähtöjen osoiteavaruudet. Jos logiikkamallia ei ole vielä tiedossa, voidaan sen paikalle asettaa määrittelemätön CPU. Jos laite on jo käytössä, voidaan sen konfiguraatio ladata projektiin laitteiston tunnistusfunktiolla. (Siemens AG 2013, 7-16.)

On tärkeää, että projektissa määritelty laitekonfiguraatio vastaa oikeaa järjestelmää (kuva 7). Jos esimerkiksi jonkin järjestelmän laitteen malli, firmware-versio, IP-osoite tai PROFINET-nimi eroaa todellisesta, ladatessa laitekonfiguraatio logiikalle se voi mennä vika-  
laan, jossa logiikka ei pysty suorittamaan ohjelmaa.

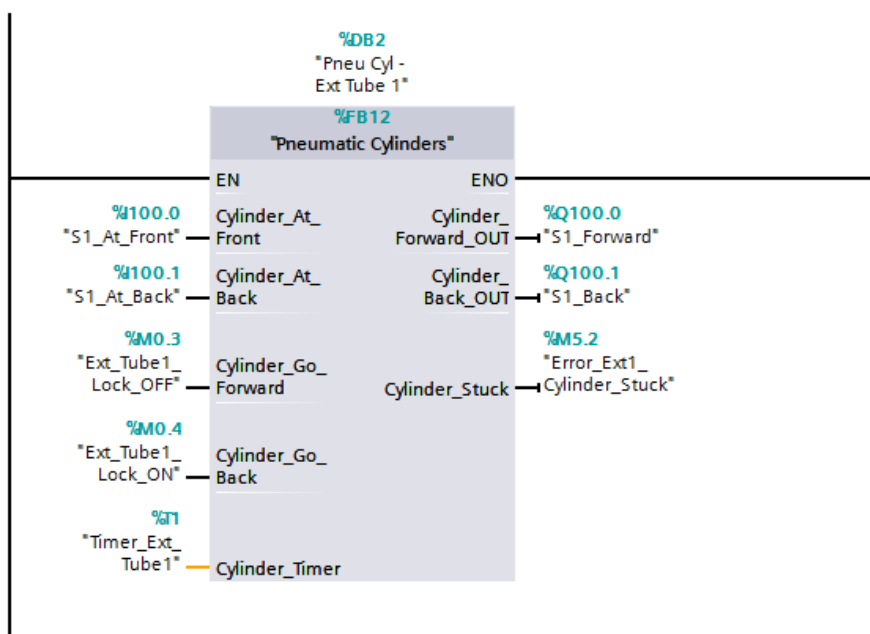


Kuva 7. Järjestelmäkonfiguraation yhteysnäkö

### 5.3 Logiikkaohjelman arkkitehtuuri

Ohjelman hierarkia ja funktioiden määrä perustui järjestelmän toimilaitteiden määrään ja järjestelmän toimintatiloihin. Ohjelmassa on kolme toimintatilaa: käsiajo, kompaundointi ja tulostus, joista jokaiselle on oma funktionsa. Yksittäiselle toimilaitetyypille muodostettiin oma funktio, joita kutsutaan toimintatilojen funktioissa.

Muodostamalla oman funktion toimilaitteen ohjaukseen voidaan toimilaitetta helposti ohjata eri tavalla eri toimintatiloissa, sekä jos samanlaisia toimilaitteita on useampia, voidaan samaa funktiota käyttää useamman kerran eri toimilaitteille muuttamalla funktion parametrejä (kuva 8). Toimintatilojen funktioilla voidaan rajata, mitä toimilaitteita kussakin tilassa voidaan käyttää ja mitkä asiat ohjaavat niiden käyttöä. Tätä kutsutaan modulaariseksi ohjelmoinniksi. Modulaarinen ohjelmointi auttaa suunnittelijaa jäsentelemään käyttäjävaatimukset toiminnallisiksi kokonaisuuksiksi ja samalla yksittäinen moduuli auttaa osoittamaan, mitkä käyttäjävaatimukset toiminnallisuus täyttää. Kuvan 8 funktiolohkolla ohjataan järjestelmän paineilmasylintereitä, joita on yhteensä neljä kappaletta.



Kuva 8. Paineilmasylintereitä ohjaava funktiolohko

#### 5.4 Käyttöliittymän käsittely

Käyttöliittymän ja logiikan rajapintana toimii tietokanta DB1. Käyttöliittymän nappeja painaessa PC kirjoittaa nappia vastaavan kuvan 9 tietokannan boolean-tyyppisen muuttujan päälle. Tietokanta sisältää myös virhekoodin indeksinumeron muuttujan, jonka eri arvot vastaavat järjestelmässä tapahtuvia eri suoritusvirheitä. Arvon ollessa suurempi kuin 0, käyttäjälle ilmestyy käyttöliittymään indeksinumeroa vastaava virheviesti seuraavalla järjestelmän tilan lukusyklillä. Virhe kuitataan käyttöliittymästä sulkemalla virheviestin ikkuna. Lisäksi robotti lähettää prosenttiluvun suorittamistaan liikepisteistä tulostuksessa, jonka käyttöliittymä lukee tietokannasta.

	Name	Data type	Offset
1	▼ Static		
2	Auto_Start_Compoun...	Bool	0.0
3	Auto_Start_Printing	Bool	0.1
4	Auto_STOP	Bool	0.2
5	Manual_Robot_Home	Bool	0.3
6	Manual_Robot_Return	Bool	0.4
7	Manual_Robot_STOP	Bool	0.5
8	Manual_Ext_Tube1_L...	Bool	0.6
9	Manual_Ext_Tube2_L...	Bool	0.7
10	Manual_Valve1_Dir	Bool	1.0
11	Manual_Valve2_Dir	Bool	1.1
12	Manual_HeatCs_ON	Bool	1.2
13	Manual_HeatCs_OFF	Bool	1.3
14	Manual_HeatC1_ON	Bool	1.4
15	Manual_HeatC2_ON	Bool	1.5
16	Manual_HeatC3_ON	Bool	1.6
17	Manual_HeatC4_ON	Bool	1.7
18	Manual_HeatC5_ON	Bool	2.0
19	Manual_HeatC6_ON	Bool	2.1
20	Clear_Extruder_Motor...	Bool	2.2
21	Error_Code	UInt	4.0
22	Print_Percentage	UInt	6.0

Kuva 9. Käyttöliittymän tietokannan rakenne

Funktiossa FC2 käyttöliittymän napista päälle asetettu muuttuja asetetaan pois päältä yhden logiikan syklin jälkeen. Funktio FC3 kirjoittaa järjestelmävirheen tapahtuessa sitä vastaavan indeksinumeron käyttöliittymän tietokantaan.

## 5.5 Käsiäjot

Käsiäjoilla käyttäjä voi ohjata käyttöliittymästä järjestelmän paineilmasylintereitä, lämmityselementtejä, sekä káskeä robotin kotipisteeseen ja palauttamaan työkaluun jäänyt pursotin. Moottoreita ei voida ajaa käsiajolla. Sulatusjärjestelmän käsiajoja varten täytyy sulatusjärjestelmän turvapiiri olla kuitattuna. Robotin käsiajot ovat puoliautomaattiajoja, joita varten täytyy robotin turvapiiri olla kuitattuna ja avainkytkin AUTO-asennossa.

## 5.6 Kompaundointi

Automaattiajon käynnistys on mahdollista vain, jos keskuksen ja robotin turvapiirit ovat kuitattuna, robotin avainkytkin AUTO-asennossa ja virheviestin indeksi on 0. Kompaundointi täytyy aloittaa ennen tulostusta. Käyttäjän painettua kompaundoinnin aloitusnappia kutsutaan kompaundoinnin ohjelmaa FC5. Aloittaessa kompaundointi järjestelmä

tarkastaa, onko molemmat pursottimet paikallaan jakopalkissa. Jos lähtötilanne on OK, aloitetaan järjestelmän lämmitys.

Lämmitystä ohjaa lämmönohjausyksikkö, joka on PID-säädin. Säätimeen asetetaan muun muassa haluttu lämpötila, lämpötilan sallittu vaihteluväli ja ramppiaika. Säätimeen kytetään jokaiselle lämmitysalueelle yksi lämpötila-anturi. Käytetyt anturit ovat J-tyyppin termopareja. Säätimen lähdöt tuodaan logiikan tuloihin ja järjestelmän toimintatilalla ohjataan, kaiutetaanko säätimen lähtöjen tilat logiikan lähtöihin. Lämmityksen alettua säätimen lähdön laskevalla reunalla tiedetään järjestelmän saavuttaneen halutun lämpötilan. Lämmitystä ohjaavat funktiot FC10 & FC11. (Elotech 2019, 22-24.)

Järjestelmän lämmettyä lukitaan pursottimet jakopalkkiin ja käännetään suuntaventtiilit ohisyötölle paineilmasyylintereillä. Paineilmasyylintereitä ohjaa funktiolohko FB12. Funktiolohkon saadessa liikekäskeyn, se kääntää venttiilin suunnan käskeyn suuntaan ajaen sylinterin haluttuun asentoon. Sylinterin männän saapuessa raja-anturille käskey resetoidaan ja venttiili palaa keskiasentoon. Venttiilit ovat 5/3-suuntaventtiilejä, joissa keskiasento on suljettu.

Virtaussuunnan ollessa oikea, käynnistetään sekoittajan ja ekstruuderin moottori. Sekoitajan moottoria ohjataan logiikan lähdöillä, ekstruuderin moottoria ohjaa funktiolohko FB285. Ekstruuderin ruuvi työntää muovimassaa jakopalkille, mistä massavirta ohjataan ohisyötölle. Ohisyöttö tuo massavirran takaisin ekstruuderin ruuvin syöttöpuolelle kierrättäen muovia järjestelmässä. Sula muovi pidetään liikkeessä koko työkierron ajan.

## 5.7 Tulostus

Käyttäjän katsottua muovimassan laadun olevan tarpeeksi tasaista tulostusta varten, käynnistää käyttäjä tulostusohjelman. Tulostusohjelma valvoo, mitkä pursottimet ovat täytettävissä. Massavirta ohjataan vapaille pursottimille, joista pursotin 1 on priorisoitu. Jos vapaita pursottimia ei ole, ohjataan massavirta ohisyötölle, kunnes jompikumpi pursotin vapautuu täytettäväksi.

Robotin ja logiikan välinen kommunikointi on määritelty funktiossa FC7. Logiikka välittää robotille tiedon noudettavista pursottimista, sekä koordinoi robotin liikkeitä pursotinta vaihtaessa erilaisilla lupatiedoilla. Lupatiedot välittävät pursottimen vaihdon sekvenssin tilaa robotin ja logiikan välillä, sallien robotin eri liikkeitä pursottimen vaihdon aikana ja pursottimen lukituksen tilan muuttamisen.

Tulostusohjelma jatkaa pursottimien täyttämistä, kunnes robotti kertoo tulostuksen olevan valmis, tai käyttäjä keskeyttää tulostuksen käyttöliittymästä. Ohjelman pysähtyessä moottorit pysähtyvät ja lämmityselementit lakkaavat lämmittämästä.

## 6 ROBOTTIOHJELMOINTI

### 6.1 Ohjelmointiympäristö ja -kieli

Robottiohjelmointi toteutettiin ABB:n RobotStudio-ohjelmalla, joka on kehitysympäristö ABB:n roboteille. RobotStudiolla robottien ohjelmoimisen ja konfiguroimisen lisäksi voi myös mallintaa ja simuloida virtuaalisia robottisoluja. (ABB Group 2018, 9.)

ABB:n robottien viimeisimmän sukupolven robottiohjain on IRC5. Ohjausyksiköllä on mahdollista suorittaa monta yksittäistä tehtävää, TASK:a rinnakkain. TASK:t voi olla usean robotin tai muun laitteen ohjausta, tai pelkästään taustalla suoritettava ohjelma. (ABB Group 2019a.)

ABB:n robotteja ohjelmoidaan RAPID-ohjelmointikielellä, joka on korkean tason proseduraalinen ohjelmointikieli. RAPID:n käskykanta sisältää robotin liikutuskäskyt, lähtöjen ohjaukset ja matemaattiset funktiot. Muuttujatyypeistä RAPID pitää sisällään yleisimmät numeeriset ja tekstimuuttujat sekä lisäksi omia muuttujatyyppejä muun muassa liikepisteille (robtarget), työkalujen määrittämiselle (tooldata) ja työkohteiden määrittämiselle (workobject). (ABB Group 2017, 5.)

### 6.2 Robottiohjelman arkkitehtuuri

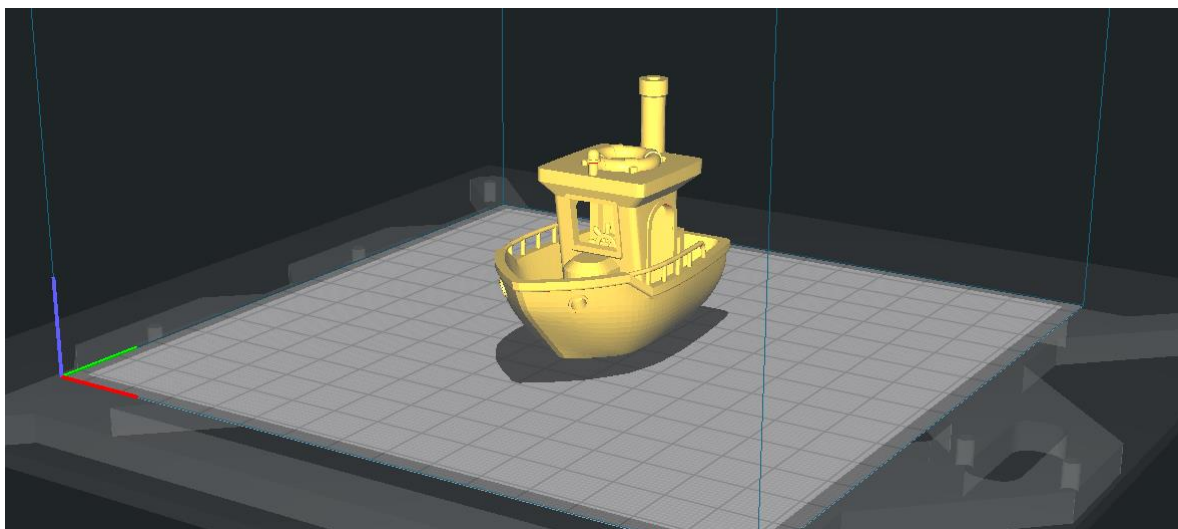
Robotin ohjelma koostuu yhdestä TASK:sta, joka ohjaa robotin liikkeitä. Ohjelmassa on kolme toimintatilaa: käsiajot, tulostus ja pursottimen vaihto. Käsiajoilla voidaan ajaa robotti kotipisteeseen tai palauttaa robotin työkalussa oleva pursotin täyttöpaikalle.

Ohjelmassa on kolme taulukkoa, joihin käyttöliittymä-PC lähettää tulostuksen parametreja. Ensimmäinen taulukko on positiotaulukko, joka sisältää kaikkien liikkeiden X-, Y- ja Z-koordinaatit. Toinen taulukko on liikkeiden nopeustaulukko ja kolmas pursottimen hydraulisylinterin positiotaulukko. Näiden taulukkojen kunkin indeksinumeroa vastaavat parametrit ovat järjestysnumeroltaan indeksinumeroa vastaavan tulostusliikkeen tarvittavat parametrit.

### 6.3 Kalibrointi

Robotin ohjaimelle on mahdollista määrittää työkalukohtaisia koordinaatistoja, workobjecteja. Workobjectit mahdollistavat liikepisteiden määrittämis- ja käyttöliittymä-PC:lle koordinaatiston suhteen. Tulostuksen tapauksessa koordinaatisto määritellään tulostusalustan suhteen. (ABB Group 2017, 1725.)

Käyttäessä koordinaattien lähteenä perinteisten 3D-tulostimien CAM-ohjelmistoja, täytyy tulostusalueen koordinaatisto vastata näissä ohjelmistoissa käytettävää koordinaatistoa. Tyypillisesti portaalmallin 3D-tulostimen koordinaatiston origo on tulostusalueen kulmassa, joten työn sovelluksessa koordinaatisto on määritelty vastaamaan kuvan 10 mallitulostimen koordinaatistoa.

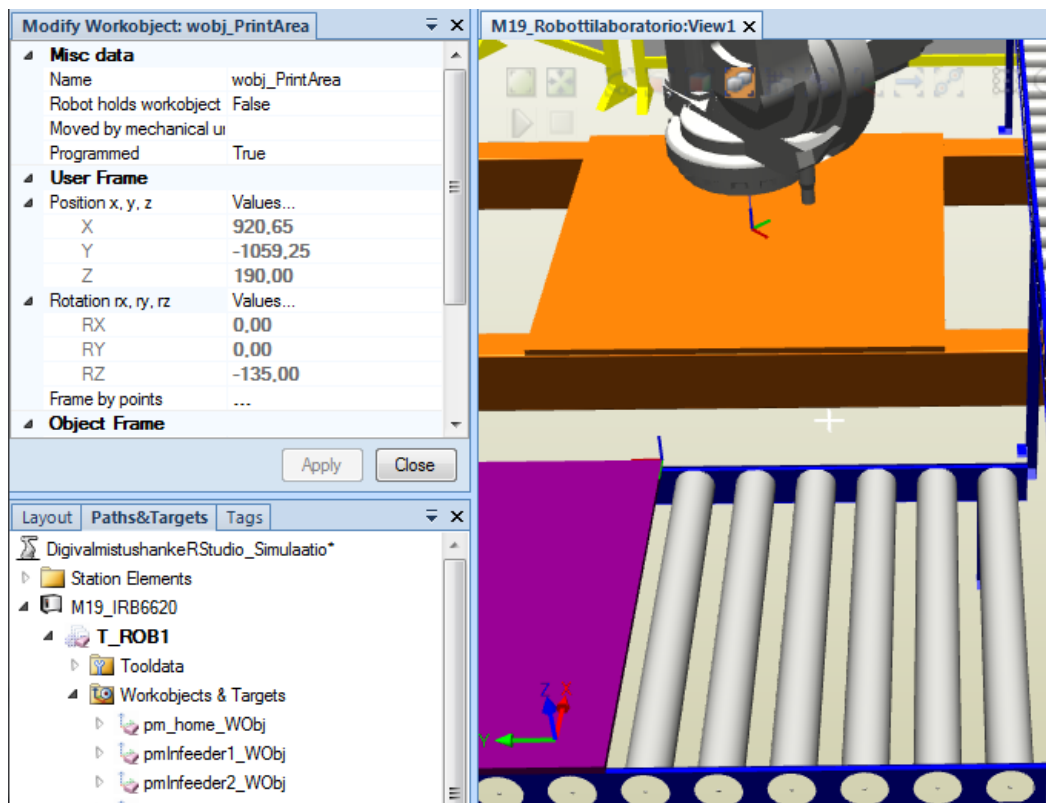


Kuva 10. Esimerkki 3D-tulostimien CAM-ohjelman koordinaatistosta

Koordinaatiston määrittäminen voidaan tehdä sekä virtuaaliympäristössä, että oikeassa ympäristössä. Virtuaaliympäristössä voidaan koordinaatisto luoda virtuaaliympäristön geometriasta. Oikeassa ympäristössä koordinaatisto voidaan määrittää ajamalla robotin työkalupiste kolmeen pisteeseen, joista robotin ohjain laskee X- ja Y-akselien muodostaman tason. Tämän tason normaalista ja XY-akselien leikkauspisteestä muodostetaan Z-akseli. Koordinaatisto voidaan myös määrittää parametrein molemmissa ympäristöissä. (ABB Group 2017, 1725-1728.)

Koordinaatiston määrittäminen on toteutettu määrittämällä robotin jalustan koordinaatiston suhteen piste, joka on tulostusalueen origossa (kuva 11). Tämän pisteen X-, Y- ja Z-koordinaattiarvot syötetään tulostusalueen koordinaatiston määrittämisparametreihin, jossa määritellään koordinaatiston origon sijainti. Koordinaatiston orientaatio on määritelty

ohjelmallisesti XY -tasoltaan yhdensuuntaiseksi jalustan koordinaatiston XY -tason kanssa ja akselien suunnat 135 astetta käännettynä myötäpäivään Z-akselin suhteen.



Kuva 11. Tulostusalueen koordinaatiston määrittäminen virtuaaliympäristössä

## 6.4 Tulostus

Kun käyttäjä katsoo kompaundoinnin olevan valmis ja aloittaa tulostusohjelman, robotin suoritusosoitin siirtyy alkuun ja tulostusohjelma alkaa. Robotti ajaa odottamaan ensimmäisen pursottimen täyttymistä ja noutaa pursottimen sen täytyttyä. Jos käyttöliittymä-PC ei ole vielä lähettänyt puskuriksi määritellyn määrän tulostusliikkeitä robotin ohjaimelle, robotti odottaa puskurin täyttymistä, jonka jälkeen aloittaa tulostuksen.

Kuvan 12 tulostusohjelmassa robotin ohjain lukee seuraavan liikepisteen indeksinumeroa vastaavat tulostusparametrit taulukosta ja muodostaa seuraavan liikepisteen. Liikepisteen muodostamiseen robotti käyttää taulukon positio- ja nopeusparametrejä, sekä kalibrointi-pisteen orientaatio- ja konfiguraatioparametrejä. Robotti ilmoittaa tulostuksen etenemisestä käyttöliittymälle lähetettävällä prosenttiluvulla suoritetuista liikkeistä.

Liikkeen nopeus muodostetaan neljästä parametrasta: työkalun nopeus (mm/s), kiertymisen nopeus (astetta/s), ulkopuolisten askelien nopeus (mm/s), sekä ulkopuolisten akselien kiertymisen nopeus (astetta/s). Taulukko pitää sisällään kunkin liikkeen työkalun nopeuden ja muut parametrit pidetään vakioina. Yksi parametreista on rajoittava tekijä robotin



interpoloidessa akselien liikkeitä, joka on tulostusohjelman tapauksessa aina työkalun nopeus. (ABB Group 2017, 1673.)

```
!Tulostusohjelma, käy liikepistetaulukon läpi
!Lue ensin seuraavan liikepisteen ja nopeuden
PROC print()
  firstPosition := [posArray{1},calibrationPoint.rot,calibrationPoint.robconf,calibrationPoint.extax];
  MoveL Offs(firstPosition, 0, 0, 50),v500,z1,Extruder\WObj:=wobj_PrintArea;

  WaitUntil posBufferOK = TRUE;  !Odota, että pistepuskrin mukainen määrä pisteitä on siirretty

  FOR nextPosIndex FROM 1 TO totalMoves DO
    !Lue seuraava paikka ja nopeus
    nextPosition := [posArray{nextPosIndex},calibrationPoint.rot,calibrationPoint.robconf,calibrationPoint.extax];
    nextSpeed := [speedArray{nextPosIndex}, 500, 5000, 1000];

    !Laske tämänhetkinen tulostuksen edistyminen prosenttimääränä suoritetuista liikkeistä
    printingProgressPercent := Round((nextPosIndex / totalMoves) * 100);
    SetGO simPrintPercentageComplete, printingProgressPercent;

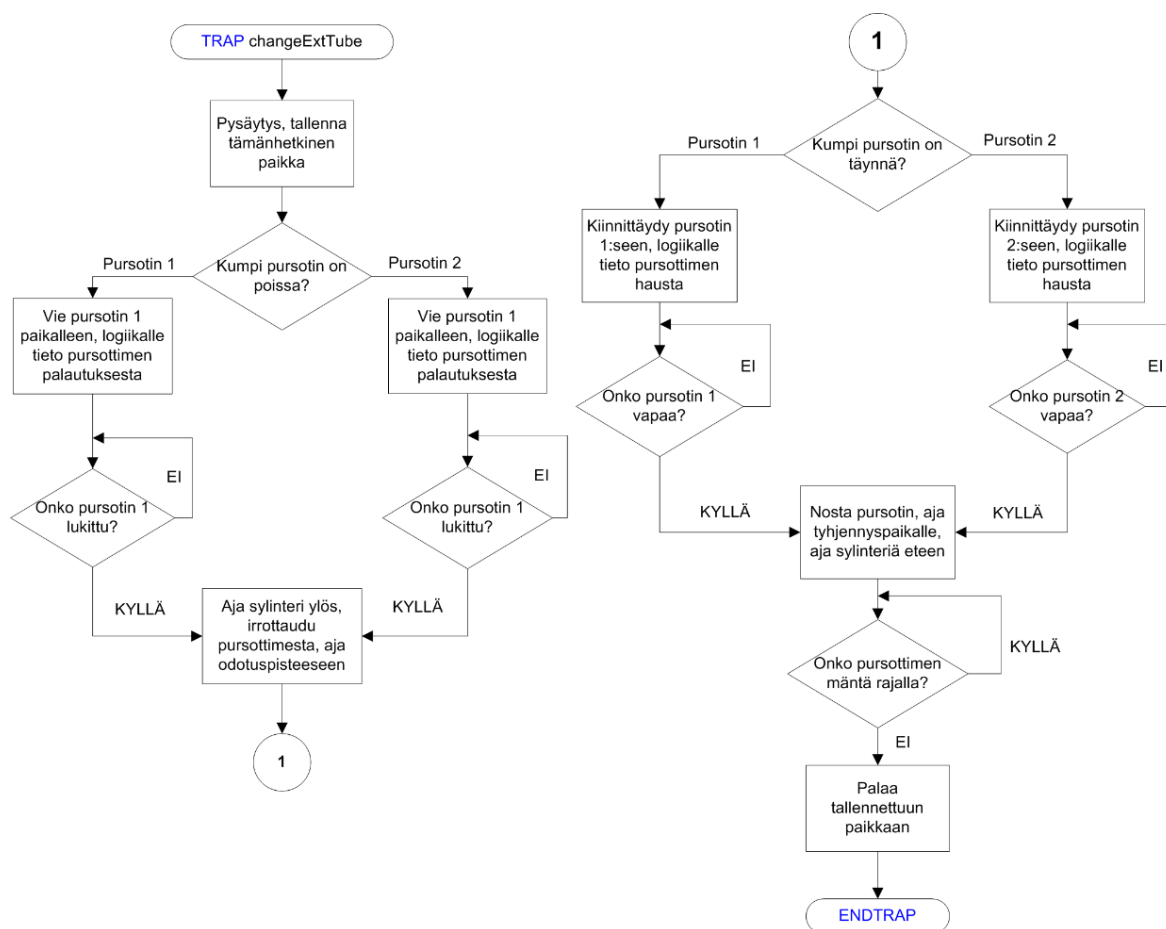
    MoveL nextPosition,nextSpeed,z0,Extruder\WObj:=wobj_PrintArea;
  ENDFOR
ENDPROC
```

## Kuva 12. Tulostusohjelma

Tulostuksessa robotti liikuttaa pursotinta generoituihin liikepisteisiin indeksi kerrallaan, kunnes taulukon koon osoittava määrä liikkeitä on suoritettu ja tulostus saatu päätökseen. Tulostuksen päättyessä robotti tyhjentää pursottimiin jääneen muovin takaisin sulatusjärjestelmän ohisyötön kanavaan, palauttaa pursottimet ja ajaa kotipisteeseen.

### 6.5 Pursottimen vaihto

Kun pursotin tyhjenee, lähettää logiikka siitä tiedon robotille. Tämä signaali kutsuu keskeytusrutiinin, jossa robotin tämänhetkinen paikka tallennetaan muuttujaan ja vaihdetaan pursotin, jonka jälkeen jatketaan tulostusta (kuva 13). Pursottimen vaihto koordinoidaan erilaisilla vaihtosekvenssin tilatiedoilla, joita logiikka ja robotti välittävät toisilleen.



Kuva 13. Keskeytysrutiinin toimintakaavio

## 7 KÄYTTÖLIITTYMÄKEHITYS

### 7.1 Ohjelmointiympäristö ja -kieli

Käyttöliittymä toteutettiin Windows Forms -applikaationa. Windows Forms on Microsoftin .NET Framework -ohjelmistokomponenttikirjaston graafinen luokkakirjasto Windows-applikaatioiden luomiseen (Watson, Nagel, Pedersen, Reid & Skinner 2010, 9). Kehitysympäristönä oli Microsoftin Visual Studio ja ohjelmointikielenä C#.

Visual Studio on pääasiallinen kehitysympäristö .NET Framework:a käyttäville ohjelmointikielille. Se sisältää muun muassa ohjelmointikielien kääntäjän, syntaksiavustajan ja graafisen suunnittelu ympäristön Forms-applikaatioille. Forms-applikaatioiden luonti suunnittelu ympäristössä on helppoa useiden valmiiden käyttöliittymäkomponenttien myötä. (Watson ym. 2010, 447-448).

### 7.2 Kirjastot

Kommunikointiin logiikan kanssa käytin S7.Net-kirjastoa, joka mahdollistaa kirjaston metodeiden avulla logiikan kaikkien tietotyyppien lukemisen ja kirjoittamisen. Toimiakseen S7.Net vaatii Siemensin logiikan ja yhteyden logiikkaan Ethernetin kautta. (GitHub 2019.)

Liikepisteiden siirtämiseen robotin ohjaimelle käytin ABB:n PC SDK:a (Software Development Kit). PC SDK mahdollistaa robotin kanssa kommunikoivien ohjelmien luomisen .NET Frameworkin ohjelmointikielillä. PC SDK:n avulla on mahdollista lukea ja kirjoittaa kaikkia muuttujatyppejä, sekä hallita ohjelmasuorituksen tilaa. Yhteys robottiin muodostetaan Ethernetin kautta. (ABB Group 2019b.)

### 7.3 Käyttöliittymänäkymä

Käynnistäessä käyttöliittymäohjelma avautuu pääikkuna, jossa on taustalla kuva järjestelmästä (kuva 14). Kuvaan on liitetty järjestelmän tilaa osoittavia elementtejä, sekä sivulla listana järjestelmän tilatietoja. Pääikkunassa on 4 valittavaa välilehteä eri toiminnoille: automaattiajo, käsiajo, slicing ja yhteysasetukset. Välilehteä vaihtamalla ilmestyy välilehteen liittyvät painikkeet. Lisäksi välilehtien valintapainikkeiden vieressä on käyttöohjepainike, joka avaa uudessa ikkunassa järjestelmän käyttöohjeen.



Kuva 14. Käyttöliittymä

## 7.4 Yhteyden muodostaminen

Aloitettaessa järjestelmän käyttö täytyy ensiksi muodostaa yhteys sekä logiikkaan, että robottiin. 'Yhteysasetukset' -välilehdestä löytyy yhteyden muodostamiseen tarvittavat ohjaukset. Yhteyden muodostamiseen logiikkaan käyttäjä syöttää logiikan IP-osoitteen, sekä Rack:n ja Slot:n numerot. Painaessa 'Yhdistä logiikkaan' kutsutaan S7.Net:n metodi, joka yrittää muodostaa yhteyden logiikkaan. Jos yhteyttä ei onnistuta luomaan, se ilmoitetaan virheviestillä.

Robottiin yhteyden muodostamista varten PC SDK:ssa on metodi saatavilla olevien robotiohjainten hakuun. Painaessa 'Päivitä ohjainlista' metodi etsii verkosta kaikki robotiohjaimet ja lisää ne pudotusvalikkoon. Haluttu ohjain valitaan valikosta ja painaessa 'Yhdistä robottiin' PC SDK:n metodi yrittää yhdistää ohjaimeen.

## 7.5 Muuttujien luku ja kirjoitus

Muodostettuaan yhteyden logiikkaan käyttöliittymä lukee järjestelmän tilatietoja logiikalta sekunnin välein taustaprosessina ja päivittää käyttöliittymän elementtejä, jos jokin tilatieto muuttuu (kuva 15). Jos luku epäonnistuu, esimerkiksi yhteyden katkettua, ilmoitetaan siitä käyttäjälle virheilmoituksella. Robotilta luetaan tulostuksen suoritettu prosenttimäärä.

```
public byte[] ReadBytes(DataType dataType, int db, int startByteAdr, int count)
```

Kuva 15. S7.Net tavujen luku -metodi (GitHub 2019)

Kaikki järjestelmän ohjauspainikkeet kutsuvat S7.Net:n metodia, joka kirjoittaa logiikan käyttöliittymälle osoitettuun tietokantaan painiketta vastaavaan osoitteeseen bitin päälle (kuva 16). Jos tilatietojen luku on painikkeen painamisen hetkellä käynnissä, odotetaan luvun päättymistä ennen kirjoitusta. Kirjoituksen epäonnistuttua siitä ilmoitetaan käyttäjälle virheviestillä.

```
public ErrorCode Write(string variable, object value)
```

Kuva 16. S7.Net yksittäisen muuttujan kirjoitus (GitHub 2019)

Käyttäjän ohjatessa järjestelmää käyttöliittymästä logiikka tekee erilaisia tilatarkistuksia käskyjen kelpoisuudesta. Jos järjestelmä on tilassa, jossa pyydettyä toimintoa ei voida toteuttaa, logiikka ei toteuta käskyä ja kirjoittaa virheviesti-indeksiin tietokantaan. Seuraavalla tilatietojen luvun syklillä käyttöliittymä antaa käyttäjälle virheilmoituksen, joka vastaa logiikan kirjoittamaa indeksiä. Indeksini nollataan sulkemalla virheviesti-ikkuna.

## 7.6 G-koodin jäsentely ja tulostusratojen lähetys

Käyttöliittymä pitää sisällään 3D-tulostimissa käytettävistä G-koodi -tiedostojen muunnon robotin ohjaimen käytettäviksi parametreiksi. Käyttäjän painettua 'Valitse tiedosto...' -painiketta avautuu tiedostonvalintaikkuna, joka suodattaa valittavassa olevat tiedostotyytit. Valittuaan tiedoston ohjelma käy läpi tiedoston jokaisen rivin ja tallentaa ne muuttujataulukkoon. Käyttäjän painettua 'Luo tulostusradat' -painiketta ohjelma etsii G-koodi -käskyistä koordinaattiparametrit, nopeusparametrit sekä ekstruuderin position parametrit.

G-koodi -tiedostot toimivat samanlailla, kun Windowsin tekstitiedostot, joten niiden lukemiseen ja käsittelymiseen voidaan käyttää .Net-kirjastoja. G-koodissa yhdellä rivillä on yksi konekäsky, jota seuraa käskyn parametrit (kuva 17) (RepRap 2019). Tallentamalla tiedoston jokainen rivi muuttujataulukkoon erotellaan mahdolliset käskyrivit erilleen toisistaan.

```
9 G92 E0
10 ;LAYER_COUNT:84
11 ;LAYER:0
12 M107
13 G0 F2400 X384.091 Y500.295 Z3
14 ;TYPE:WALL-INNER
15 G1 F3000 X384.045 Y499.991 E0.00282
16 G1 X384.09 Y499.69 E0.00561
17 G1 X384.091 Y496.664 E0.03335
18 G1 X384.008 Y495.86 E0.04076
19 G1 X384 Y490.304 E0.09169
20 G1 X384.006 Y488.61 E0.10722
```

Kuva 17. G-koodi -esimerkki

G-koodi voi sisältää useita erilaisia käskyjä liittyen koneen parametointiin ja työkalumäärittäisiin, sekä ohjelmakoodi saattaa sisältää kommentteja. Löytääkseen oleelliset käskyt, ohjelman tulee etsiä rivit, jotka alkavat käskytunnuksella 'G0' tai 'G1'. Näillä tunnuksilla olevat käskyt ovat liikekäskyjä. Liikekäskyn parametrit voivat koostua liikenopeudesta, X-Y- ja Z-koordinaateista, sekä ekstruuderin positioista. G-koodissa jokaista parametria ei tarvitse määrittää jokaisen liikekäskyn kohdalla, parametrin puuttuessa aikaisemmassa käskyssä asetettu parametri jää voimaan. (RepRap 2019.)

G-koodin jäsentely on toteutettu kuvan 18 säännöllisillä lausekkeilla. Säännölliset lausekkeet ovat ohjelmoinnin työkaluja, joilla voidaan etsiä ja suodattaa tekstiä (Computer Hope 2019). Jäsentelyn funktiossa on kuusi kaavaa, joilla yhdellä etsitään liikekäsky ja viidellä muulla käskyyn liittyvät parametrit. Jos luettu rivi on liikekäsky, seuraavat kaavat etsivät, onko käskyssä määritelty muita parametrejä.

```
Regex findCommand = new Regex(@"G0|G1");
Regex findF = new Regex(@"F(\d+\W\d+|\d+)");
Regex findX = new Regex(@"X(\d+\W\d+|\d+)");
Regex findY = new Regex(@"Y(\d+\W\d+|\d+)");
Regex findZ = new Regex(@"Z(\d+\W\d+|\d+)");
Regex findE = new Regex(@"E(\d+\W\d+|\d+)");
```

Kuva 18. G-koodin jäsentelyn säännölliset lausekkeet

Liikekäskyn lauseke etsii, löytyykö riviltä teksti 'G0' tai 'G1'. Käyttäessä 3D-tulostimien CAM -ohjelmista saatua G-koodia koodin rakenne on kontrolloitu, joten näitä tekstejä ei pitäisi esiintyä muussa yhteydessä. Jos tekstiä ei löydy, siirrytään seuraavaan riviin.

Jos rivi on liikekäskyriivi, loput lausekkeet etsivät kunkin parametrin tunnuskirjainta, sekä kirjainta seuraavat numerot. G-koodissa desimaalejen erotustunnus on piste, joten lausekkeeseen on muodostettu ehtolauseke, joka poimii pisteen jälkeiset numerot. Parametrin löydyttyä päivitetään parametrin arvo. Jos parametria ei ole käskyssä, säilytetään edellisen käskyn arvo. Parametrien etsinnän jälkeen saaduista arvoista muodostetaan kolme muuttujaa: robotin positio, nopeus ja pursottimen positio. Muuttujat lisätään taulukkoon, jonka jälkeen luetaan seuraava rivi.

G-koodin muunnoksen jälkeen käyttöliittymä ilmoittaa löydettyjen liikekäskyjen määrän ja pyytää käyttäjää vahvistamaan liikepisteiden lähetyksen aloittamisen robotin ohjaimelle. Vahvistuksen jälkeen käynnistyy taustaprosessi, joka lähettää liikepisteiden parametrit robotin ohjaimen parametritaulukkoon. Lähetettyään käyttäjämääritellyn puskurin verran liikepisteitä, annetaan robotille lupa aloittaa tulostus. Lähetyksen valmistumisesta ilmoitetaan käyttöliittymän 'Info' -osiossa.

## 7.7 Automaattiajo

Automaattiajon ollessa päällä evätään pääsy muihin välilehtiin. Tällä varmistetaan, ettei muita toimintoja, kuten tulostustiedoston vaihtamista kesken tulostuksen tai käsiajolla robotin pysäyttäminen ole mahdollista. Jos ohjelma ei vastaa, varokeinona järjestelmä voidaan pysäyttää myös fyysisestä stop-napista ohjauskeskuksen kannessa.

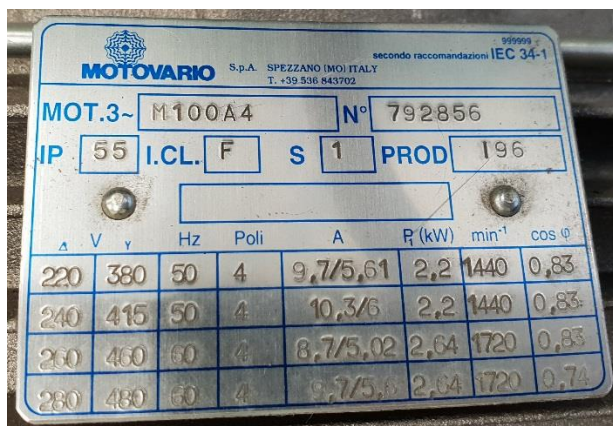
## 7.8 Käyttöohje

'Käyttöohje' -painiketta painaessa avautuu käyttöohjeikkuna. Käyttöohjeen 'Laitteen käyttö' -osiossa kerrotaan laitteen toimintaperiaatteesta, tarvittavista valmisteluista laitteen käyttöä varten, ohjeita robotin kalibrointiin sekä mitä tietoja ja toiminnollisuuksia käyttöliitymä pitää sisällään. 'Tietoja' -osiossa kerrotaan digivalmistushankkeesta, hankkeen osallistujista, sekä mainitaan ilmaisten ikonien tekijät Flaticon -sivuston käyttöehtojen mukaan (Freepik Company S.L 2019).

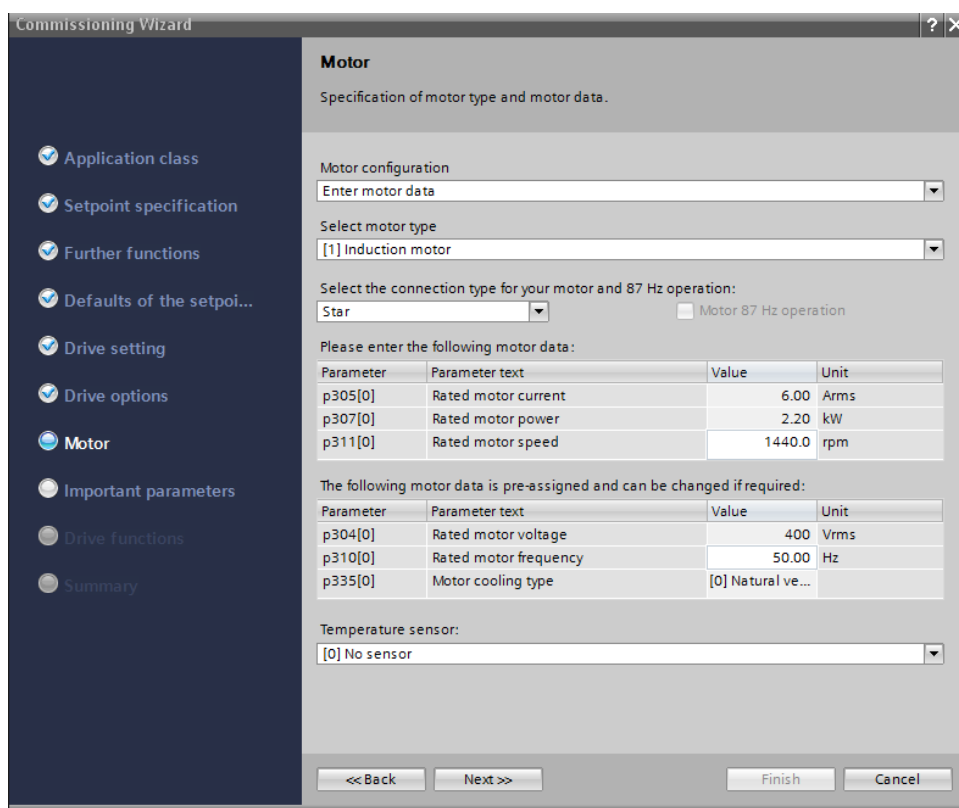
## 8 KÄYTTÖÖNOTTO

### 8.1 Taajuusmuuttaja

Taajuusmuuttajan käyttöönottoon ja parametointiin käytin TIA Portal:n Startdrive -ohjelmiston Commissioning Wizard:a (kuva 20). Käyttöönottoavustajassa määritellään vaiheittain käyttökohde, kommunikaatio logiikan välillä, moottorin tiedot ja taajuusmuuttajan parametrit. Kuvassa 20 on tehty parametrimäärittely kuvan 19 moottorin arvokilven tiedoista.



Kuva 19. Moottorin arvokilpi



Kuva 20. Käyttöönottoavustaja



Parametroinnin jälkeen moottoria pystyy koeajamaan Startdrive:n Control Panel -ikkunasta. Käyttökohteen yksinkertaisuudesta ja avustajan helppokäyttöisyydestä johtuen käyttöönotto oli helppoa.

## 8.2 PI-komponentit

Paineilmakomponenttien käyttöönotto sisälsi raja-antureiden säädöt ja toiminnan testausten. Käsiajoilla varmistettiin sylintereiden liikekäskyjen toiminta. Käyttöönottohetkellä sylinterit eivät olleet kiinnitettynä suuntaventtiilien ja pursottimien lukitusten mekanismeihin.

## 8.3 Koeajo

Järjestelmän valmiuden tasosta johtuen koeajossa ei päästy testaamaan kaikkia toiminnallisuuksia. Koeajo piti sisällään paineilmakomponenttien ja ekstruuderin moottorin toiminnan toteamisen, liikepisteiden siirron robotin ohjaimelle, robotin toimintojen testausten, sekä robotin ja logiikan yhteistoiminnan testausten automaattiajolla. Tulostusohjelman aikana robotti liikutti työkalua ilmassa tulostusratojen mukaisesti. Manipuloiden logiikan tuloja käyttöliittymästä simuloitiin lämmityselementtien toimintaa, sekä pursottimen tyhjentymistä ja täyttymistä. Järjestelmä todettiin vastaavan toimintakuvausta koeajossa testattujen toiminnallisuuksien osalta.



Kommentteja tulisi käyttää jokaisen ohjelman alussa, kuvaten yleisellä tasolla ohjelman toiminnan. Jos ohjelma sisältää useita funktioita, tulisi funktion alussa olla funktion toimintaa kuvaava kommentti. Lisäksi käskyissä, joiden toiminnallisuus ei tule selväksi koodista itsestään, on hyvä lisätä kommentti. (University of Utah 2019.)

### 9.1.2 Arkkitehtuuridokumentit ja toimintakuvaus

Järjestelmän arkkitehtuurin dokumentointi on keskeisessä roolissa järjestelmän ohjelmallisen toimintaperiaatteen kuvaamisessa. Se on järjestelmän ohjelmallisen toteutuksen selkäranka, joka ei muutu ilman erittäin suuria toimintakuvauksen muutoksia. Se toimii samalla perehdytysmateriaalina, miten toimintakuvauksessa kuvatut toiminnallisuudet on toteutettu. (Haikala & Mikkonen 2011, 194.)

Toimintakuvauksen tärkein tehtävä on osoittaa, mitä järjestelmä tekee täyttääkseen käyttäjävaatimukset. Toimintakuvauksen ei tulisi ottaa kantaa järjestelmän tekniseen toteutukseen, ellei reunaehdoissa niitä ole esitetty. (Edwards 2019, 2.)

## 9.2 Toteutumattomat järjestelmän osat

Toimintakuvauksessa mainittuja toteuttamattomia toiminnallisia ominaisuuksia ovat purssotimen hydraulisylinterin ohjaus ja turvatoiminnot turvalogiikalla toteutettuna. Työn aikana hydraulikomponenttien valinta viivästyi ja ohjelmointihetkellä sylinterin ohjaustapa ei ollut tiedossa. Opinnäytetyö toteutettiin tavallisella S7-1500 -sarjan logiikalla, eikä työn toteutushetkellä hankkeen turvalogiikka ollut saapunut tavarantoimittajalta. Kelpoistusvaiheesta jäivät toteutumatta työn ulkopuolelle jääneiden toiminnallisten ominaisuuksien testaus, järjestelmän lopullinen asennus, koeajo sekä riskinarviointi.

## 9.3 Kehityskohteita

Järjestelmän kunkin osa-alueen myöhäisissä kehitysvaiheissa tuli ilmi ominaisuuksia, joita olisi voinut lisätä, tai jotka olisi voitu toteuttaa paremmin. Muutokset olivat laajuudeltaan niin suuria, ettei muutoksia keritty toteuttamaan työhön käytettävän ajan puitteissa.

### 9.3.1 Logiikkaohjelma

Käyttöliittymän yhteyden tilaa ei valvota logiikkaohjelman puolella. Tilan valvonta voitaisiin toteuttaa yhdistämisen jälkeen käynnistettävällä tilantarkistusrutiinilla, missä käyttöliittymä ja logiikka varmistaisivat tietyin aikavälein yhteyden olevan kunnossa. Tämä tilatarkistus voisi olla käyttöliittymän syklisen prosessin tilan tarkistuksen yhteydessä.

Käyttöliittymän ja logiikan välistä rajapintaa ei ole määritetty logiikkaohjelmassa, paitsi käyttöliittymän nappien, vikakoodi-indeksin ja tulostuksen edistymistiedon osalta. Välitettävät tiedot voitaisiin koota joko yhteen tietokantaan, muodostaa eri tiedoista käyttäjämääritetty tietotyyppi, tai serialisoida yhteen muuttujaan.

### 9.3.2 Robottiohjelma

Tulostuksessa seuraava robotin liikepiste lasketaan robotin liikkuessa senhetkiseen liikepisteeseen. Jos liikepisteet ovat erittäin lähellä toisiaan, on mahdollista, että robotin ohjain ei kerkeä laskemaan seuraavaa liikepistettä ennen kuin robotti on suorittanut liikkeen. Jos näin käy, pysähtyy robotti liikepisteen kohdalle, luoden nykivää liikettä. Varmistaakseen liikkeiden sulavuuden voidaan luoda toinen TASK, joka laskisi liikepisteitä rinnakkaisena prosessina.

### 9.3.3 Käyttöliittymä

Käyttöliittymän lukiessa prosessin tilatietoja logiikalta käyttöliittymän elementit päivitetään joka lukukerralla ehtolausekkein. Välttääkseen päivittämistä, kun tilatieto ei ole muuttunut, voidaan tilatietomuuttujien arvon muuttuminen sitoa tapahtumiin.

Prosessin tilatietojen luku koostuu seitsemästä luvusta, joissa luetaan eri tietotyyppejä logiikalta. Karsiakseen lukukertoja ja vähentääkseen aikaa, jolloin logiikka on käyttämättömissä painonappien kirjoitusta varten, voidaan lukea yksi tietotyyppi, jonka elementit puretaan käyttöliittymän puolella.

## 10 YHTEENVETO

Opinnäytetyön tavoite oli tuottaa toimiva automaatiojärjestelmä Digivalmistushankkeen tulostussoluun. Siltä osin, mikä tulostussolun valmiusaste oli työn päätyttyä, tavoitteisiin päästiin suurimmalta osin. Järjestelmän kunkin osa-alueen suunnittelu ja toteutus saatiin päätökseen tärkeimmiltä osa-alueiltansa ja järjestelmä todettua toimivaksi. Tuotetut ohjelmistot toimivat pohjana järjestelmän jatkokehitykselle ja käyttöönotolle.

Työ antoi paljon uusia haasteita ja oppimismahdollisuuksia kustakin osa-alueesta. Logiikkaohjelmoinnissa tutustuin uuteen kehitysympäristöön ja toimilaitteisiin, robottiohjelmoinnissa uusiin prosessidatan hallintatapoihin ja hankkeen käyttöliittymän kehitys oli minulle ensimmäinen suurempi ohjelmistokehitysprojekti. Automaatiojärjestelmän osa-alueiden toteutuksen lisäksi perehdyin automaatioprojektien läpiviemiseen kokonaisuuden kannalta ja pyrin soveltamaan aiheen kirjallisuudessa esitettyjä käytäntöjä omaan työhöni siltä osin, kun se oli työn laajuuden merkeissä järkevää.

Kaikista toteutuksen osa-alueesta oli paljon resursseja saatavilla, joten uusien asioiden opettelu oli tiedonhankinnan kannalta jouhevaa. Välitestauksia ja protoilua ohjelmistokehityksen eri vaiheissa pystyi tekemään virtuaaliympäristössä, joka säästi aikaa järjestelmän lopullisessa testausvaiheessa ja antoi vapauksia työn toteutusajalle ja -paikalle.

Työn laajuus osoittautui työn edetessä erittäin suureksi. Varsinkin ohjelmistokehityksen osuus opinnäytetyön laajuudessa yllätti minut. Opinnäytetyön aihealueiden lisäksi ylimääräisiä tehtäviä toi komponenttien valinnat ja testiohjauskeskuksen rakentaminen. Ylimääräistä työkuormaa olisi helpottanut työn aloitus myöhemmässä vaiheessa hanketta, mutta haasteista huolimatta hanketta saatiin vietyä jo pitkälle. Uskon, että tämän työn ja muiden hankkeeseen liittyvien projektien myötä järjestelmä saadaan käyttöön lähitulevaisuudessa.

## LÄHTEET

ABB Group 2017. RAPID Technical reference manual. Käyttöohje [viitattu 08.04.2019]. Saatavissa: [https://library.e.abb.com/public/b227fcd260204c4dbeb8a58f8002fe64/Rapid\\_instructions.pdf?x-sign=f79v/883X1nHGc8fqH+WAJ2F30y/M6TZfYUuPuQpP+jeM-BygouyGg+WSj8A9Otry](https://library.e.abb.com/public/b227fcd260204c4dbeb8a58f8002fe64/Rapid_instructions.pdf?x-sign=f79v/883X1nHGc8fqH+WAJ2F30y/M6TZfYUuPuQpP+jeM-BygouyGg+WSj8A9Otry)

ABB Group. 2018. Product specification - RobotStudio. Tuote-esite [viitattu 19.03.2019]. Saatavissa: ABB Group. 2019a. IRC5 [viitattu 19.03.2019]. Saatavissa: <https://new.abb.com/products/robotics/controllers/irc5>

ABB Group. 2019b. PC SDK [viitattu 20.03.2019]. Saatavissa: <http://developercenter.robotstudio.com/pcsdk/>

Computer Hope. 2019. What is a Regex [viitattu 08.04.2019]. Saatavissa: <https://www.computerhope.com/jargon/r/regex.htm>

Edwards, D. 2019. Functional Requirements Specification - "The Misunderstood Document". G.F.S. Computing, Inc. Julkaisu [viitattu 12.03.2019]. Saatavissa: [https://www.automation.com/pdf\\_articles/FRS\\_White\\_Paper.pdf](https://www.automation.com/pdf_articles/FRS_White_Paper.pdf)

Elotech GmbH. 2019. R2500 S with service interface - Description and operating manual. Käyttöohje [viitattu 17.03.2019]. Saatavissa: [https://www.elotech.de/fileadmin/user\\_upload/Downloads/Bedienungsanleitungen/englisch/R2500-XXX-S-X-000-X-X\\_EN.pdf](https://www.elotech.de/fileadmin/user_upload/Downloads/Bedienungsanleitungen/englisch/R2500-XXX-S-X-000-X-X_EN.pdf)

Freepik Company S.L. 2019. Flaticon - Terms of Use [viitattu 20.03.2019]. Saatavissa: <https://www.flaticon.com/terms-of-use>

Galler, C. 2019. Network Documentation Best Practices: What's Important & How To Track It. Packet Pushers [viitattu 15.03.2019]. Saatavissa: <https://packetpushers.net/network-documentation-best-practices-whats-important-how-to-track-it/>

GitHub Inc. 2019. S7.Net documentation [viitattu 19.03.2019]. Saatavissa: <https://github.com/S7NetPlus/s7netplus/wiki>

Greik, P. 2011. Aloittavan koneensähkösuunnittelijan työohje [viitattu 12.03.2019]. Saatavissa Lahden ammattikorkeakoulun Intranetissa: [https://reppu.lamk.fi/pluginfo.php/764500/mod\\_folder/content/0/Tyo-ohje\\_versio\\_15042011.pdf?forcedownload=1](https://reppu.lamk.fi/pluginfo.php/764500/mod_folder/content/0/Tyo-ohje_versio_15042011.pdf?forcedownload=1)

Haikala, I. & Mikkonen, T. 2011. Ohjelmistotuotannon käytännöt. Helsinki: Talentum.

Koskimies, K. & Mikkonen, T. 2005. Ohjelmistoarkkitehtuurit. Helsinki: Talentum.

Lahden ammattikorkeakoulu. 2019. digiValmistus - Tulevaisuuden valmistusteknologioiden mahdollisuudet liiketoiminnan kehittämisessä [viitattu 08.03.2019]. Saatavissa: <https://www.lamk.fi/fi/hanke/digivalmistus-tulevaisuuden-valmistusteknologioiden-mahdollisuudet-liiketoiminnan>

SFS-EN ISO/ASTM 52900:2017. Materiaalia lisäävä valmistus. Yleiset periaatteet. Terminologia. Helsinki: Suomen Standardisoimisliitto.

PLCdev. 2019. Step 7 Elementary Data Types [viitattu 02.04.2019]. Saatavissa: [http://www.plcdev.com/step\\_7\\_elementary\\_data\\_types](http://www.plcdev.com/step_7_elementary_data_types)

PLCopen. 2019a. IEC 61131-3: a standard programming resource. Artikkelit [viitattu 01.04.2019]. Saatavissa: [https://www.plcopen.org/sites/default/files/downloads/intro\\_iec\\_oct2016.pdf](https://www.plcopen.org/sites/default/files/downloads/intro_iec_oct2016.pdf)

PLCopen. 2019b. Logic [viitattu 01.04.2019]. Saatavissa: <https://www.plcopen.org/technical-activities/logic>

RepRap. 2019. G-code [viitattu 08.04.2019]. Saatavissa: <https://reprap.org/wiki/G-code>

Siemens AG. 2013. TIA Portal - Creating the project and hardware. Käyttöohje [viitattu 28.03.2019]. Saatavissa: [https://cache.industry.siemens.com/dl/files/451/78027451/att\\_14643/v1/config\\_en.pdf](https://cache.industry.siemens.com/dl/files/451/78027451/att_14643/v1/config_en.pdf)

Siemens AG. 2019. Totally Integrated Automation Portal [viitattu 28.03.2019]. Saatavissa: [http://www.siemens.fi/fi/industry/teollisuuden\\_tuotteet\\_ja\\_ratkaisut/tuotesivut/tia\\_portal.php](http://www.siemens.fi/fi/industry/teollisuuden_tuotteet_ja_ratkaisut/tuotesivut/tia_portal.php)

Suomen Automaatioseura ry. 2019. Laatu automaatiassa – Parhaat käytännöt. Artikkelit [viitattu 12.03.2019]. Saatavissa: <https://www.automaatioseura.fi/site/assets/files/1367/laatuautomaatiassa.pdf>

Tampereen teknillinen yliopisto. 2002. Tietotekniikan peruskurssi - 19.2 OSI-malli [viitattu 15.03.2019]. Saatavissa: <http://www.cs.tut.fi/etaopetus/titepk/luku19/OSI.html>

Työ- ja elinkeinoministeriö. 2019. Mitä rakennerahastot ovat? [viitattu 02.04.2019]. Saatavissa: <https://www.rakennerahastot.fi/mita-rakennerahastot-ovat>

Ua Automation. 2019. IEC 61131-3 Functions and Function Blocks: What is the difference? [viitattu 01.04.2019]. Saatavissa: <http://ua.automation.com/resources-tools/application-stories/programmable-logic-controller-plc/iec-61131-3-functions-and-function-blocks-what-is-the-difference>

University of Utah. 2019. Programming – Commenting [viitattu 04.02.2019]. Saatavissa: <https://www.cs.utah.edu/~germain/PPS/Topics/commenting.html>

Watson, K. Nagel, C. Pedersen, J. H. Reid, J. D. & Skinner, M. 2010. Beginning Visual C# 2010. John Wiley & Sons, Incorporated.



## LIITTEET

### Liite 1. Järjestelmän toimintakuvaus

#### DIGIVALMISTUSHANKE – 3D-TULOSTUSSOLU

Versio 2 – 09.04.2019

## SISÄLLYS

1	JÄRJESTELMÄN YLEISKUVAUS .....	37
1.1	Päätoiminnot.....	37
1.2	Laiteliittymät.....	37
1.3	Oletukset .....	38
2	TOIMINTA .....	39
2.1	Toimintatilat .....	39
2.1.1	Muovin sulatus.....	39
2.1.2	Pursottimien täyttö ja tulostus .....	39
2.2	Käsiajot.....	40
2.3	Varmuustoiminnot.....	40
2.4	Turvatoiminnot.....	40
2.4.1	Hätäpysäytys .....	41
2.4.2	Hätä-seis .....	41
2.4.3	Ovirajakytkin .....	41
2.4.4	Huomiovalo .....	41
3	EI-TOIMINNALLISET OMINAISUUDET .....	41
3.1	Huollettavuus.....	41
3.2	Ylläpidettävyys.....	41
4	LIITTEET .....	41

## 1 JÄRJESTELMÄN YLEISKUVAUS

### 1.1 Päätoiminnot

Solun tehtävä on tuottaa raakamuovista sulaa muovia ja tulostaa siitä kolmiulotteisia kapaleita. Solu koostuu muovin murskauslaitteistosta, muovin sulatus- ja jakelulaitteesta, sekä nivelvarsirobotista. Raakamuovi murskataan sulatettaviksi partikkeleiksi, sulatetaan ja jaetaan pursottimille, jonka jälkeen robotti käyttää pursotinta tulostukseen.

Murskauslaitteisto on täysin käyttäjäohjattu. Sen toimintaa ei ohjata tai valvota muussa järjestelmässä. Toimintakuvaus käsittelee automatisoidun sulatusjärjestelmän ja tulostajana toimivan robotin toimintaa.

Sulatusjärjestelmä koostuu syöttökaukalosta, ekstruuderista, jakopalkista, sekä kahdesta pursottimesta. Syöttökaukaloon syötettyä muovirouhetta sekoitetaan, mistä rouhe tippuu ekstruuderin syöttökanavaan. Ekstruuderin työntää muovirouhetta ruuvikuljettimella lämmityspiirien läpi sulattaen muovin. Ekstruuderin päässä on jakopalkki, jossa on kaksi suunta-venttiiliä. Suuntaventtiileillä massavirta ohjataan joko ohisyötölle, tai pursottimille.

Pursottimen ollessa täynnä robotti käy hakemassa pursottimen jakopalkilta. Robotti tulostaa pursottimen tyhjäksi, minkä aikana toinen pursotin täyttyy. Robotti vaihtelee pursottimia tulostustyön ajan.

### 1.2 Laiteliittymät

Järjestelmään kuuluu PC, jossa toimii käyttöliittymä, sulatusjärjestelmää ohjaava ohjelmoitava logiikka, lämmönsäädin, nivelvarsirobotti ohjaimineen, sekä hydraulikoneikko.

PC on yhteyksissä sulatusjärjestelmän ohjausyksikköön ja robottiin käyttöliittymän osalta. Prosessin tilaa ohjaa sulatusjärjestelmän ohjelmoitava logiikka, joka välittää tilatietoja robotille ja PC:lle. Robotin ohjain välittää liikkeidensä tilatietoja sulatusjärjestelmän ohjausyksikölle.

PC:n ja muun laitteiston välinen tiedonsiirtoprotokolla on TCP/IP, logiikan ja robotin välinen PROFINET.

### 1.3 Oletukset

Tarkemmat komponenttispesifikaatiot ovat liitteenä olevassa komponenttilistassa.

Ohjauskeskus tulee toiseen ohjauskeskukseen kiinni, ohjauskeskus oltava 700mm korkea.

Sulatusjärjestelmää ja turvatoimintoja ohjaa turva-PLC.

Lämmitystä ohjaa erillinen lämmönsäädin. Lämmönsäätimen säätöalueet on jaettu lämmitysalueisiin, jotka koostuvat yhdestä lämpötila-anturista ja 2-4kpl lämmitysvastuksista.

Ekstruuderin kolmessa lämmityspiirissä ja molemman pursottimen lämmityspiireissä on 2kpl pantavastuksia/piiri. Jakopalkin lämmityspiirissä on 4kpl patruunavastuksia.

Jakoventtiilien ja pursottimien lukitukset jakopalkkiin toteutetaan pneumaattisilla sylintereillä. Sylintereitä ohjataan 5/3 venttiileillä.

Venttiiliryhmä tulee sijoitettavaksi erilliseen ohjauskeskukseen sulatusjärjestelmän läheisyyteen. Venttiiliryhmää ja sen toimilaitteiden anturointeja ohjaa I/O-hajautusyksikkö.

Pursotin lukitaan robotin työkaluun mekaanisesti kiinnityslaipalla ja hydraulisylinterillä, joka painaa pursottimen mäntää alaspäin. Robotin työkaluun ei tule sähköisiä kytkentöjä, joten pursottimen sähköisten toimilaitteiden kaapeloinnit on tuettava asianmukaisella tuennatavalla ja kaapelityypit on oltava sovellukseen sopivia.

Käyttöjärjestelmä on PC:llä toimiva sovellus. Sovelluksen kaatumisen varalta solu täytyy olla pysäytettävissä fyysisestä painonapista muun tavoin, kun hätäpysäytyksenä.

## 2 TOIMINTA

### 2.1 Toimintatilat

Tulostusprosessissa on kaksi toimintatilaa. Käyttäjä määrittää, milloin ensimmäinen toimintatila on tullut päätökseen. Toimintatilojen vaihto tapahtuu käyttöliittymästä.

#### 2.1.1 Muovin sulatus

Järjestelmää käynnistäessä tehdään lähtötilanteen tarkistus. Molempien pursottimien täytyy olla paikoillaan, joka tarkistetaan anturoinnilla. Lämmitysohjausyksikölle annetaan lupa alkaa lämmittämään jokaista lämmityspiiriä ohjausyksikköön asetettuun lämpötilaan. Lämmönohjausyksikkö pitää lämmityspiirit asetetussa lämpötilassa koko työkierron ajan.

Järjestelmän lämmettyä pursottimet lukitaan jakopalkkiin, ohjataan jakopalkin suuntaventtiilit ohisyötölle, sekä käynnistetään syöttökaukalon sekoittajan moottori ja ekstruuderin ruuvikuljettimen moottori. Massavirtaa kierrätetään järjestelmässä ohisyötöllä, kunnes käyttäjä katsoo silmämääräisesti massan olevan tarpeeksi tasalaatuista.

#### 2.1.2 Pursottimien täyttö ja tulostus

Käyttäjän valittua tulostusohjelman jakopalkin ensimmäinen suuntaventtiili ohjaa massavirran täytölle ja toinen suuntaventtiili pursottimelle 1. Pursottimen täytyessä sen mäntä nousee, ja täynnä ollessaan mäntä osuu rajakytkimeen. Suuntaventtiili ohjaa massavirran täyttämään pursotin 2:sta

Ohjausyksikkö antaa robotille luvan tulla noutamaan pursottimen 1. Robotti lukitsee pursottimen työkalun laippaan, ja pursottimen jakopalkin lukitus vapautuu. Robotti tulostaa pursottimen tyhjäksi, palauttaa sen paikalleen ja noutaa seuraavan. Kiertokulku jatkuu, kunnes tulostettava tuote on valmis.

Kun tulostus on päättynyt, robotti tuo käyttämänsä pursottimen jakopalkille ja ajaa kotipisteeseen. Ohjausyksikkö sammuttaa sekoittajan ja ekstruuderin moottorit, sekä estää lämmönohjausyksikön lämmittämästä lämmityspiirejä. Pursottimet jätetään lukituiksi jakopalkkiin.

## 2.2 Käsiäjot

Osa järjestelmästä voidaan ohjata käsin käyttöliittymästä. Ohjauksiin liittyy lukituksia. Käsiäjolla ei voi ajaa moottoreita.

- Jos pursotin on jäänyt kiinnitetyksi robotin työkaluun, voidaan robotti ohjata tuomaan pursotin takaisin jakopalkille
- Robotti voidaan ajaa kotipisteeseen
- Lämmitysohjausyksikkö voidaan asettaa aktiiviseksi
- Pursottimien lukitukset voidaan avata pursottimen irrotusta varten
- Jakoventtiilien suuntaa voidaan ohjata

## 2.3 Varmuustoiminnot

Järjestelmä valvoo toimintaansa virheiden varalta anturoinnilla. Virheen sattuessa solun toiminta pysäytetään ja käyttäjälle ilmoitetaan virheestä käyttöliittymään.

Käynnistyksen yhteydessä tehdään erilaisia tilatarkastuksia liittyen toiminnan aloittamiseen. Jos ehdot eivät täyty, käyttöliittymässä tulee puutteen osoittava virheviesti käyttäjälle.

Suuntaventtiilien ja pursottimien lukitusmekanismien toimintaa valvotaan sylinterien rajaantureilla. Jos sylinteri ei saavuta haluttua asentoa tietyssä aikamääreessä, voidaan päätellä, että mekanismi on jumittunut.

## 2.4 Turvatoiminnot

Turvatoimintojen tarkoitus on ehkäistä vaaratilanteiden syntyminen konetta käytettäessä. Vaaratilanteita voi syntyä esimerkiksi käyttäjän mennessä solun sisään, koneen väärinkäytöstä tai ohjelmallisesta virheestä.

Turvatoimintoja ohjaa turva-PLC. Turva-PLC:n toimintojen suunnittelu ja ohjelmointi toteutetaan koneturvallisuusstandardin SFS-EN ISO 13849 mukaan.

Mahdollisten vaaratilanteiden, turvatoimintojen ja komponenttien määrät ja yksityiskohdat voi muuttua tai tarkentua toteutusvaiheessa tehtävän riskinarvioinnin myötä.

### 2.4.1 Hätäpysäytys

Hätäpysäytys pysäyttää robotin liikkeen, katkaisee moottorien sekä lämmityksien energiat ja säilyttää pursottimien lukituksen tilan. Robotin pysäytys tulisi tehdä mahdollisimman hallitusti. Vaadittu pysäytysaika tarkentuu riskinarvioinnissa. Hätäpysäytys tulee kuitata fyysisestä painonapista solun ulkopuolelta.

### 2.4.2 Hätä-seis

Hätä-seis –painikkeita on kaksi, yksi käyttäjän PC:n välittömässä läheisyydessä ja yksi keskuksen kannessa. Hätä-seis –painikkeen painaminen käynnistää hätäpysäytyksen.

### 2.4.3 Ovirajakytkin

Solussa on kaksi ovirajakytkintä. Oven avaus käynnistää hätäpysäytyksen.

### 2.4.4 Huomiovalo

Jos ekstruuderin osat ovat liian kuumia kosketeltaviksi käsin, tulisi siitä ilmoittaa huomiovalolla.

## 3 EI-TOIMINNALLISET OMINAISUUDET

### 3.1 Huollettavuus

Ekstruuderin huoltoa varten ekstruuderin ruuvi on purettavissa vetämällä ruuvi ja vaihde-moottori ulos ekstruuderiputkesta.

### 3.2 Ylläpidettävyys

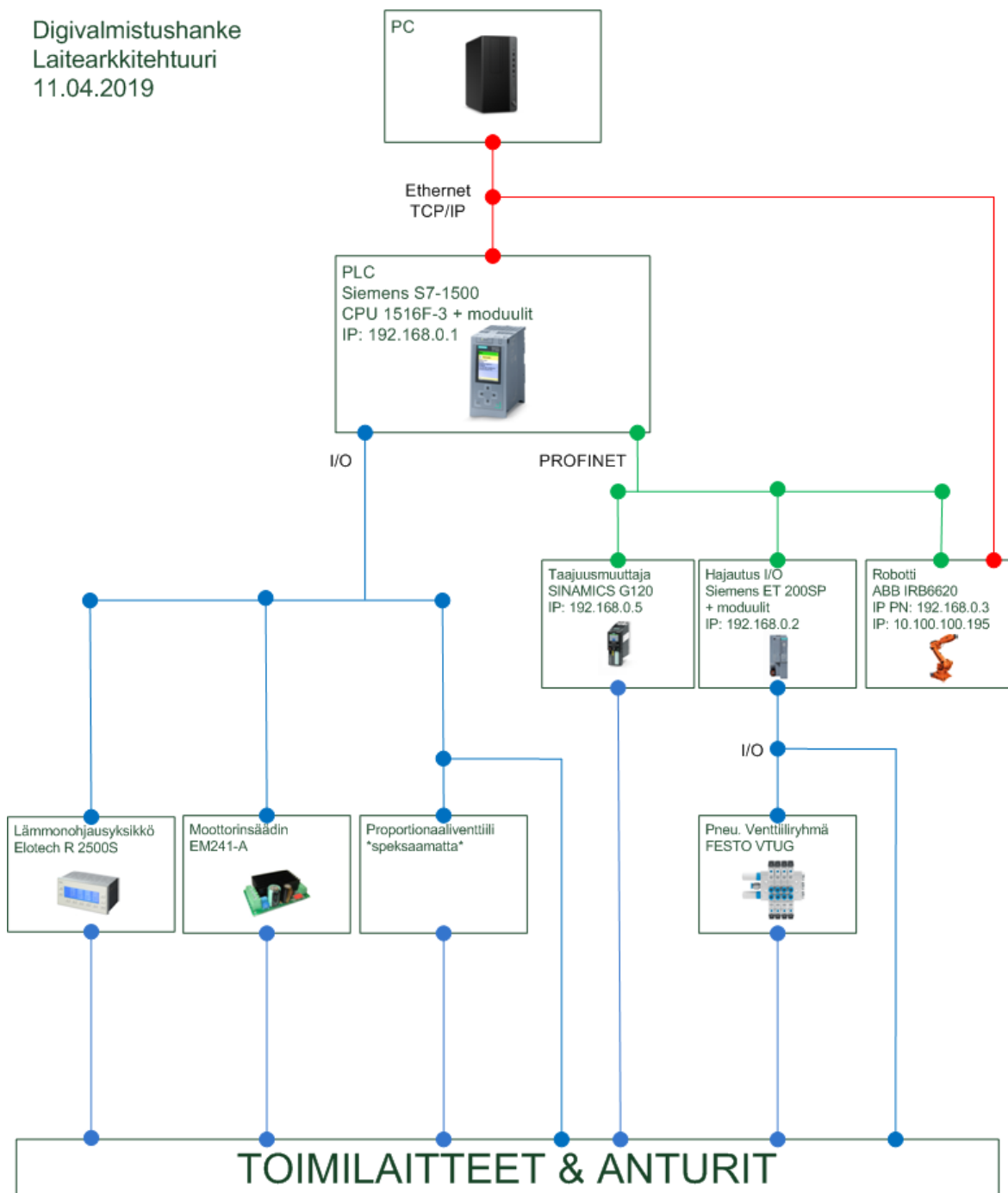
Hanketta tulee kehittämään todennäköisesti hyvin moni taho, joten hyvä dokumentointi on tärkeässä roolissa. Jokaisen toteutuksen osa-alueen dokumentit on oltava ajan tasalla ja asianmukaisesti tuotettu.

## 4 LIITTEET

Komponenttilista

## Liite 2. Laitearkkitehtuurikaavio

Digivalmistushanke  
Laitearkkitehtuuri  
11.04.2019



### Liite 3. Ohjelmistoarkkitehtuurikaavio

Digivalmistushanke  
Ohjelmistoarkkitehtuuri  
11.04.2019

