Tampereen ammattikorkeakoulu

Tuotantolinjan graafisen käyttö-liittymän päivittäminen

Jaakko Myllymäki

OPINNÄYTETYÖ
Toukokuu 2020

Konetekniikan tutkinto-ohjelma
Koneautomaation opintosuunta

TIIVISTELMÄ

Tampereen ammattikorkeakoulu
Konetekniikan tutkinto-ohjelma
Koneautomaation opintosuunta

MYLLYMÄKI, JAAKKO:
Tuotantolinjan graafisen käyttöliittymän päivittäminen

Opinnäytetyö 58 sivua, joista liitteitä 1 sivu
Toukokuu 2020

Tässä opinnäytetyöraportissa käsitellään toimeksiantona tehtyä tuotantolinjaston käyttöliittymäpäivitystä sekä työhön ja käyttöliittymiin liittyvää yleistä teoriaa. Opinnäytetyön tarkoituksena oli toteuttaa toimeksiantajan asiakkaan tuotantolinjastolle käyttöliittymäpäivitys. Käyttöliittymäpäivityksen tarkoituksena oli kasvattaa tuotannon tehokkuutta sekä parantaa valmiin tuotteen laatua ja työntekijöiden turvallisuutta.

Opinnäytetyön tavoitteena oli siirtää vanhan käyttöliittymän toiminnallisuudet uuteen käyttöliittymään. Toiminnallisuuden siirtämiseen sisältyi kommunikaatiorajapinnan tekeminen. Tavoitteena oli myös kehittää käyttöliittymän ulkoasua ja käyttäjäkokemusta. Tämän jälkeen uusi käyttöliittymä oli tarkoitus testata, asentaa uudelle tietokoneelle sekä ottaa käyttöön asiakkaan tuotantotiloissa Michiganissa, Yhdysvalloissa. Käyttöliittymän testaus sekä käyttöönotto on rajattu pois raportin piiristä.

Opinnäytetyöprojekti suoritettiin tiukan aikataulun puitteissa, sillä käyttöliittymä täytyi olla toimivana ja testattuna ennen toisella mantereella tapahtuvaa käyttöönottoa. Onnistuneen projektin tuloksena asiakas sai käyttöönsä uudistetun käyttöliittymän päivitetyllä ulkoasulla sekä parannetulla käyttäjäkokemuksella. Käyttöliittymäpäivitys tehostaa ja parantaa asiakkaan tuotantoa sekä vähentää käyttöliittymän ja sen ohjelmiston häiriöistä johtuvia tuotantokatkoksia ja niistä aiheutuvia tulonmenetyksiä.

Opinnäytetyöprojekti onnistui toimeksiantajan haluamalla tavalla. Käyttöliittymä saatiin toimitettua asiakkaalle täysin toimintakuntoisena ja ajallaan. Käyttäjäkokemuksen kehittäminen jäi osittain kesken tiukan aikataulun vuoksi. Kehittämisehdotuksena on saattaa käyttäjäkokemuksen parantaminen loppuun sekä uudistaa käyttöliittymää visuaalisesti.

Asiasanat: graafinen käyttöliittymä, käyttöliittymäpäivitys, UX-suunnittelu

ABSTRACT

Tampereen ammattikorkeakoulu
Tampere University of Applied Sciences
Degree Programme in Mechanical Engineering
Machine Automation

MYLLYMÄKI, JAAKKO:
Update of a Production Line Graphical User Interface

Bachelor's thesis 58 pages, appendices 1 page
May 2020

This thesis covers the update process of a graphical user interface and general theory related to it. The purpose of this thesis was to execute a user interface update for the client's manufacturing line. The purpose of the user interface update was to increase the production efficiency, product quality and employee safety. The goal was to transfer the functionalities from the old interface to the new interface which included the making of a communication interface. Finally, the aim was also to develop and enhance appearance and usability of the user interface.

The theoretical section of this thesis consists of a general review of theory related to user interfaces, graphical user interfaces and user experience of user interfaces. The empirical part consists of development of the user interface and the programmable logic controller.

As a result of this project, the client received an updated interface with improved visuals and user experience. The interface update improves production and decreases the amount of production stoppages, therefore reducing income losses. The usability update was not finished due to the strict schedule of the project. The client was satisfied with the results, so the project was a success. The focus in further development is to finish the user experience update and to visually reshape the graphical user interface.

Key words: graphical user interface, interface update, UX-design

SISÄLLYS

1 JOHDANTO 5
2 KÄYTTÖLIITTYMÄN PERUSTEET 6
2.1 Käyttöliittymä yleisesti 6
2.2 Graafinen käyttöliittymä 8
2.3 GUI-suunnittelun erot teollisuudessa ja kuluttajakäytössä 10
2.4 UX-suunnittelu 12
2.4.1 Luonnollisuus 12
2.4.2 Yhdenmukaisuus 13
2.4.3 Käyttäjäystävällisyys 14
2.4.4 Selkeys 15
3 KÄYTTÖLIITTYMÄPÄIVITYKSEN ESITIEDOT 17
3.1 Työn tausta 17
3.2 Suunnittelutyökalut 22
3.2.1 Trello 22
3.2.2 Vijeo Look 23
3.2.3 iX Developer 24
3.2.4 PL7 Pro 25
3.3 Modbus-rajapinta 26
4 KÄYTTÖLIITTYMÄPÄIVITYS 28
4.1 Projektin hallinta 28
4.2 Modbus-rajapinnan tekeminen 30
4.3 Toiminnallisuuksien siirto PLC:stä käyttöliittymään 35
4.4 Käyttöliittymään kustomointi 41
4.4.1 Objektien lisääminen käyttöliittymään 42
4.4.2 Objektien poistaminen käyttöliittymästä 48
4.4.3 Näkymien visuaalinen parantaminen 50
4.5 Yhteenveto 53
5 POHDINTA 55
LÄHTEET 56
LIITTEET 58
Liite 1. Gantt-kaavio 58

1 JOHDANTO

Nykypäivänä ihmiset ovat tekemisissä käyttöliittymien kanssa useita kertoja päivässä. Graafisen käyttöliittymän löytää ihan arkisistakin laitteista, kuten esimerkiksi tietokoneesta, puhelimesta, tv:stä ja jopa pyykinpesukoneesta. Käyttöliittymän kautta käyttäjä välittää koneelle tai laitteelle haluamiansa ohjeita ja käskyjä. Toisin sanoen koneen toimintaa ohjataan käyttöliittymällä. Tämän vuoksi on erittäin oleellista, että käyttöliittymä on mahdollisimman selkeä ja helppokäyttöinen.

Tämän opinnäytetyön tarkoituksena on toteuttaa käyttöliittymäpäivitys UCSolutionsin asiakasyrityksen tuotantolinjastolle Yhdysvaltoihin. Asiakasyrityksen tuotantolinjastolla on tällä hetkellä käytössä Windows XP -käyttöjärjestelmälle pohjautuva käyttöliittymä, joka on tulossa elinkaarensa päähän ulkoasun, toiminnallisuuksien, sekä tietoturvan osalta. Käyttöliittymä on asiakasyrityksen tuotannon keskeisenä elementtinä toimivan linjaston tärkein osuus. Sen optimoinnilla, selkeyttämisellä ja kehittämisellä voidaan saavuttaa tuotantoon tehokkuutta, laadun parannusta sekä turvallisuutta.

Tavoitteena on siirtää vanhan käyttöliittymän toiminnallisuudet uuteen käyttöliittymään sekä tehdä kommunikaatiorajapinta käyttöliittymän ja ohjelmoitavan logiikan välille. Tavoitteena on myös siirtää osa toiminnallisuuksista ohjelmoitavalta logiikalta suoraan käyttöliittymään sekä parantaa käyttöliittymän ulkoasua ja käyttäjäkokemusta. Tämän jälkeen uusi käyttöliittymä on tarkoitus mennä ottamaan käyttöön paikan päälle asiakasyrityksen tuotantolaitokseen Michiganiin, Yhdysvaltoihin.

Tässä opinnäytetyöraportissa käsitellään käyttöliittymiin liittyvää teoriaa, asiakkaan vanhan käyttöliittymän toiminnallisuuksien siirto uuteen käyttöliittymään, sekä uuden käyttöliittymän elementtien ja ulkoasun räätälöinti asiakkaan tarpeisiin. Käyttöliittymän ja logiikan testaaminen sekä käyttöönotto on rajattu ulos raportin piiristä.

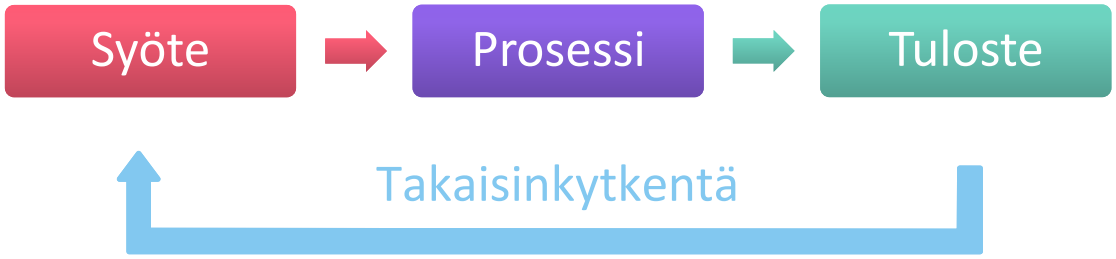
2 KÄYTTÖLIITTYMÄN PERUSTEET

Tässä luvussa selitetään aluksi mikä käyttöliittymä on ja mitä se tarkoittaa. Sen jälkeen paneudutaan syvemmin tämän opinnäytetyön kannalta tärkeimpään käyttöliittymätyyppiin: graafiseen käyttöliittymään. Luku päätetään käyttöliittymän käyttäjäkokemuksen suunnittelusta kertovaan osuuteen. Luku toimii pohjustuksena käyttöliittymän kokonaiskuvan hahmottamiseksi.

2.1 Käyttöliittymä yleisesti

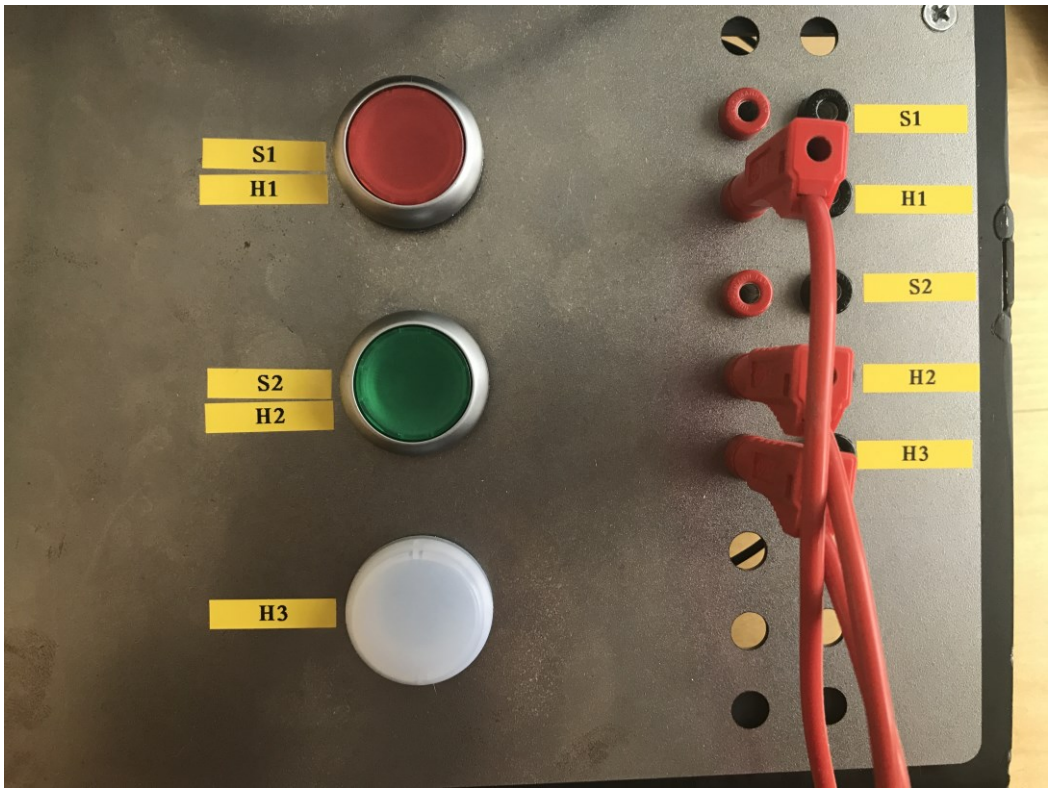
Käyttöliittymä on tietokoneen, laitteen tai muun koneen ohjelmiston osa, jonka käyttäjä voi nähdä tai kuulla. Sen avulla ihminen pystyy kommunikoimaan laitteen kanssa ilman syvällistä tuntemusta tai ymmärrystä laitteen ohjelmointikielestä. Pohjimmiltaan käyttöliittymä sisältää kaksi pääkomponenttia: **syötteen** ja **tulosteen**. Syötteen avulla käyttäjä voi antaa laitteelle käskyjä ja ohjeita esimerkiksi toimintaan liittyen. Syötteen antamiseen voidaan käyttää esim. painonappia, hiirtä, näppäimistöä, sormea (kosketusnäytölle) tai ääntä (mikrofoni ääniohjaukselle). Tulosteella laite kerto käyttäjälle tietoja ja vaatimuksia laitteen toimintaan liittyen. Nykypäivänä näyttöpääte on yleisin tapa tulosteen esittämiselle. (Galitz 2007, 4)

Jokainen järjestelmä sisältää syötteen, prosessin, tulosteen ja takaisinkytkennän. Kuviossa 1 on esitettynä yksinkertainen järjestelmämalli. Tästä voidaan nähdä mikä yhteys näillä eri järjestelmäkomponenteilla on toisiinsa ja erityisesti tulosteen ja syötteen välinen yhteys. Täysin automaattisen järjestelmän kohdalla takaisinkytkentä tapahtuu automaattisesti, jolloin järjestelmä säätää ja ohjaa itse itseään. Manuaalikäyttöisen järjestelmän tapauksessa koneen käyttäjä reagoi käyttöliittymältä saatavaan tulosteeseen ja näiden perusteella päättää itse annettavat syötteet. Tällöin automaattisen takaisinkytkennän korvaa ihminen. Käyttöliittymä toimii siis rajapintana ihmisen ja koneen välillä.



KUVIO 1. Yksinkertainen järjestelmämalli

Käyttöliittymä voi yksinkertaisimmillaan sisältää esimerkiksi yhden painikkeen ja yhden lampun. Tästä hyvä esimerkki on tavallinen kahvinkeitin. Kahvinkeitin käynnistyy nappia painamalla ja keittimen ollessa päällä lampussa palaa valo. Tässä tapauksessa painike toimii syötteenä, jolla konetta ohjataan ja lamppu tulosteena, joka kertoo käyttäjälleen laitteen tilan. Kuvassa 1 on esitettynä yksinkertainen opetuskäyttöön tarkoitettu ohjauspaneeli. Siinä etulevyasenteisten valopainikkeiden (S1/H1 ja S2/H2) sekä merkkivalon (H3) yhdistelmällä on toteutettu yksinkertainen käyttöliittymä.



KUVA 1. Opetuskäyttöön tarkoitettu ohjauspaneeli

Monimutkaisempien laitteiden ja koneiden käyttöliittymät useimmiten sisältävät useampia syötteitä ja tulosteita, vaikkakin niiden lukumäärä ei suoraan korreloi laitteen kehittyneisyyteen tai monimutkaisuuteen. Joissain tapauksissa yhdellä napilla voidaan ohjata monimutkaistakin prosessia, jos se on toteutettu esimerkiksi sekvenssiohjauksella. Sekvenssiohjauksessa laite suorittaa yksittäiset ohjelmavaiheensa ennalta määritellyssä järjestyksessä, jossa siirtyminen vaiheesta seuraavaan on riippuvainen edellisen vaiheen ehtojen täyttymisestä (Center for Chemical Process Safety 2014). Näin ollen peräkkäisten vaiheiden ketjutuksella voidaan toteuttaa monimutkaisia järjestelmiä. Mitä enemmän toimintoja toteutetaan yhtä painallusta kohden, sitä enemmän järjestelmän automatisointia se vaatii.

2.2 Graafinen käyttöliittymä

Graafinen käyttöliittymä (kuva 2) eli **GUI** (Graphical User Interface) on käyttöliittymätyyppi, joka sisältää interaktiivisia komponentteja, joita visualisoidaan näyttöpäätteessä. GUI näyttää käyttäjälle erilaisia toimintoja kuvaavia objekteja, joita käyttäjä voi halutessaan suorittaa. Objektit voivat myös sisältää laitteeseen tai prosessiin liittyvää informaatiota. (Computer Hope 2019)

KUVA 2. Windows-käyttöjärjestelmän graafinen käyttöliittymä sisältää monia kuvaavia objekteja, joiden takaa löytyy erilaisia toimintoja.

Graafisen käyttöliittymän objektien takaa löytyvät toiminnot ovat esiohjelmoituja. Tällöin käyttäjän ei tarvitse tietää mitään tarkkoja komentoja tai tiedostorakenteita – käyttäjän tarvitsee vain painaa painiketta, ja ennalta määrätty toimenpide suoritetaan. Tällä voidaan saavuttaa tietyissä tapauksissa huomattavaa ajansäästöä ja helpottaa käyttökokemusta. Siksi käyttäjä, jolla ei ole minkäänlaista kokemusta tietokoneista voi käyttää graafiseen käyttöliittymään perustuvaa järjestelmää melko vaivattomasti. Toimintojen esiohjelmoinnilla ja sen helppokäyttöisyydellä on myös kääntöpuolensa – kokenut ja perehtynyt käyttäjä on pitkälti pakotettu käyttämään muiden määrittämiä toimintoja, eikä näin ollen pysty muuttamaan järjestelmän perustoiminnallisuutta. (Educba 2019)

Hyvin suunniteltu GUI on visuaalinen ja informatiivinen. Objektit muuttavat väriä ja kokoa käyttäjän syötteen tai järjestelmän tulosteen toimesta – näin ollen käyttäjälle on selvää, mitä laitteessa tapahtuu toimintoja suoritettaessa tai mitä tuloksia laitteelta saadaan. Toimintoja kuvaavien objektien ulkonäkö vaihtelee paljon eri graafisten käyttöliittymien välillä ja jopa käyttöliittymän sisällä riippuen niiden tyylistä ja käyttötarkoituksesta. Graafista käyttöliittymää pidetään paljon käyttäjäystävällisempänä kuin komentoliittymää (command-line interface), joka on myös melko yleinen tietokoneissa. (Computer Hope 2019)

Kuvassa 3 on esimerkin vuoksi tehty kuvituskuva mahdollisesta teollisuuskäytössä olevasta graafisesta käyttöliittymästä. Vasemmalla sijaitsee käyttöliittymän näkymän valintaan tarkoitetut painikkeet. Keskellä näkyy valittu näkymä ja siitä löytyvät toiminnot. Oikealla olevassa kentässä näkyy valitusta näkymästä huolimatta koko ajan esillä olevat tärkeimmät tiedot prosessista. Käyttöliittymän elementeistä ja yleisestä ulkoasusta on yritetty tehdä mahdollisimman selkeä ja havainnollinen. Valittu ikkuna on korostettu sekä kääntämällä aktiivisen painikkeen väri, että lisäämällä sen nimi ikkunan vasempaan yläkulmaan. Ikkunasta löytyvät toiminnot on nimetty, mutta niille on myös sen lisäksi tehty pieni kuvake visualisoimaan toimintoa. Aina esillä olevat kriittiset prosessitiedot on erotettu vieressä olevasta ikkunasta laatikolla selkeyttämään näkymää. Eri prosessitietojen arvojen värit myös muuttuvat sen mukaa, onko arvo normaali (valkoinen), normaalista poikkeava (keltainen) tai kriittinen (punainen).

KUVA 3. Kuvitteellinen teollisuuslaitoksen graafinen käyttöliittymä

2.3 GUI-suunnittelun erot teollisuudessa ja kuluttajakäytössä

Teollisuuden tuotantolinjastojen ja kuluttajatuotteiden käyttöliittymissä ja niiden suunnittelussa on paljon yhtäläisyyksiä, mutta myös huomattavia eroja. Käyttöliittymän tärkein yksittäinen aspekti on käyttäjäkokemus. Näissä tapauksissa loppukäyttäjien ero on varsin merkittävä. Teollisuudessa laitteen käyttäjiä on todennäköisesti vain yksi yksittäinen joukko, yleensä yhden tehtaan juuri sitä kyseistä laitetta tai tuotantolinjaa operoiva koulutettu työntekijäryhmä. Kuluttajatuotteessa taas samaa tuotetta ja käyttöliittymää saattaa käyttää miljoonat ihmiset kulttuurista, kielestä tai koulutuksesta riippumatta. (Sirin Software 2019)

Teollisuuden käyttöliittymät

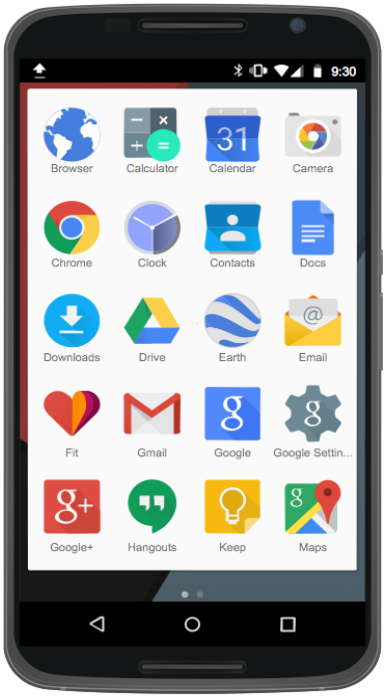
Teollisuudessa olevien tuotantolinjastojen käyttöliittymien graafiset ulkoasut ovat usein yksinkertaisia ja pelkistettyjä. Tämän perimmäinen syy on usein ylimääräisten kustannusten leikkaamisessa – käyttöliittymän näkymien ja objektien graafiseen ulkoasuun ja sen näyttävyyteen ei ole tarpeen tuhlata ylimääräisiä resursseja, sillä tuotantolinjaston käyttöliittymän suunnittelun päätavoite ei ole kehittää järjestelmää, joka olisi visuaalisesti näyttävä ja houkutteleva kuluttajamarkkinoilla, vaan joka olisi mahdollisimman selkeä, informatiivinen ja joka täyttää kyseisen käyttökohteen vaatimukset. Näin on mahdollista säästää sekä suunnittelukustannuksissa, että tuotantoa ohjaavan järjestelmän hardwaressa eli tietokonekomponenteissa.

Teollisuuden käyttöliittymiä ei usein ole suunniteltu yhtä holhoaviksi kuin kuluttajapuolella. Tämä tarkoittaa sitä, että käyttöliittymä odottaa käyttäjänsä osaavan laitteen käytön ja ymmärtävänsä toimintojen ja komentojen seuraukset. Jokaista eri toimintoa ei ole välttämättä erikseen selitetty tai kommentoitu. Tällöin käyttöliittymä ei ole yhtä käyttäjäystävällinen ja anteeksiantava kuin kuluttajatuotteiden vastaavat. Siksi on ensiarvoisen tärkeää, että näitä käyttöliittymiä operoivat käyttäjät koulutetaan perinpohjaisesti.

Kuluttajatuotteiden käyttöliittymät

Kuluttajatuotteiden ja elektroniikkalaitteiden graafinen käyttöliittymä suunnitellaan aivan toisenlaiset tavoitteet mielessä pitäen kuin teollisuudessa. Kuluttajatuotteiden käyttöliittymän on oltava geneerinen ja suurille massoille sopiva suurien valmistusmäärien takia. Toisin kuin teollisuuden käyttöjärjestelmät, jotka usein tilataan ja suunnitellaan yksilöllisesti jokaiselle tuotantolaitokselle, kuluttajatuotteiden käyttöjärjestelmä toimitetaan samanlaisena miljooniin saman tuotteen kopioihin. Tämän takia käyttöliittymää suunniteltaessa on huomioitava esimerkiksi eri loppukäyttäjien kulttuurierot, tekniikan tuntemus ja kieli.

Kuluttajatuotteiden käyttöjärjestelmissä painotetaan usein myös sitä, että se olisi visuaalisesti näyttävä, mutta myös sitä, että se olisi myös mahdollisimman helppokäyttöinen ja selkeä tekniikkaan perehtymättömällekin henkilölle. Tällöin käyttöliittymän objekteista yritetään tehdä mahdollisimman selkeitä ja kuvaavia. Tästä hyvä esimerkki on kuvassa 4 esiintyvä matkapuhelimen Android-käyttöjärjestelmä. Jokainen näytöllä esiintyvä objekti sisältää kuvakkeen ja lyhyen kuvaavan tekstin kuvaamaan sen toiminnallisuutta. Tällä voidaan luoda mielleyhtymä reaalimaailman esineen tai kuvan ja toiminnon välille, mikä parantaa käytettävyyttä. Tästä kerrotaan lisää luvussa 2.4.1.



KUVA 4. Esimerkkikuva Android-käyttöjärjestelmän graafisesta käyttöliittymästä (ConceptDraw 2014)

2.4 UX-suunnittelu

Laadukkaan käyttöliittymän suunnittelussa yksi kaikkein tärkeimmistä huomioon otettavista asioista on käyttäjäkokemus eli **UX** (User experience). Jotta käyttöliittymän käyttäjäkokemus olisi tarpeeksi hyvä, täytyy käyttäjän tuntea hallitsevansa ohjelmistoa, ei toisinpäin. Tällaisen käyttäjäkokemuksen saavuttamiseksi on huomioitava tärkeimmät UX-suunnittelun aspektit. Tässä luvussa on listattuna muutama tärkeä asia UX-suunnittelun kannalta.

2.4.1 Luonnollisuus

Ditte Mortensen (2020) kertoo artikkelissaan, että käyttäjäkokemus on silloin luonnollisimmillaan, kun käyttöliittymä ei pakota käyttäjää muuttamaan merkittävästi heidän tavallista tapaansa lähestyä ja ratkaista ongelmia. Se tarkoittaa sitä, että järjestelmän tai ohjelmiston lähettämät viestit, sekä antamat tulokset

eivät pitäisi tarvita ylimääräisiä selityksiä, vaan niiden pitäisi olla mahdollisimman selkeitä. Tällöin ohjelmiston merkintäjärjestelmän ja terminologian pitäisi olla selkeä ja yhdenmukainen koko ajan. (Mortensen 2020)

Käyttäjälle tutuilla käsitteillä ja kuvilla pystytään tekemään intuitiivinen ja helposti lähestyttävä käyttöliittymä (Mortensen 2020). Tätä konseptia hyödyntämällä luodaan eräänlainen linkki oikean maailman ja toiminnon välille. Käyttäjät muistavat tuttuun esineeseen tai kuvaan assosioidun toiminnon paljon helpommin (kuvio 2), kuin esimerkiksi komennon nimeen assosioidun toiminnon, kuten jo luvussa 2.2 mainittiin.

KUVIO 2. Esineen tai asian kuvaan assosioitu toiminto kuvitteellisessa käyttöliittymässä.

Myös kosketusnäytöllisten käyttöliittymien eleohjauksessa on huomioitava luonnollisuus – on tärkeää, että kosketusnäyttöä pystyy käyttämään yhdellä sormella, joka on ihmisille luonnollisempaa, kuin vaikkapa kolmella sormella. Ja jos esimerkiksi käyttöliittymässä olevaa luetteloa selaa pyyhkäisemällä alhaalta ylös, on erittäin luonnollista olettaa, että tällöin kuva näytöllä siirtyy luettelossa alemmas.

2.4.2 Yhdenmukaisuus

Yhdenmukaisuus on yksi tärkeimmistä ominaisuuksista käytettävyyden ja oppivuuden parantamiseen. Yhdenmukainen käyttöliittymä antaa käyttäjälle mahdollisuuden adaptoitua ja hyödyntää aikaisempaa tietoaan sekä osaamistaan uudessa tehtävässä. Tällöin käyttäjä pystyy keskittymään uusien ominaisuuksien

hallitsemiseen ja keskittymään ongelmanratkaisuun mieluummin, kuin käyttämään aikaansa ymmärtääkseen tiettyjen komentojen ja ohjaimien eroavaisuudet. (Babich, 2019)

Maria De La Riva (2018) on artikkelissaan kertonut, että yhdenmukaisuus on välttämätöntä käyttöliittymän jokaisella osa-alueella mukaan lukien komentojen nimissä, tiedon visuaalisessa esittämisessä ja interaktiivisien elementtien käyttäytymisessä. Tällä pystytään parantamaan käytettävyyttä, vähentämään epäselvyyttä ja näin ollen tuottamaan positiivinen tunnetila miellyttävästä käyttökokemuksesta. (De La Riva 2018)

2.4.3 Käyttäjäystävällisyys

Useimmiten käyttäjät oppivat käyttämään uutta järjestelmää virheiden kautta oppimalla. Hyvän käyttöliittymän pitäisi ottaa tämä huomioon. Sen pitäisi aina tarjota oikeat ja turvalliset toiminnot, sekä varoittaa käyttäjää tilanteista, jossa on mahdollisuus aiheuttaa vahinkoa järjestelmälle, datalle, laitteelle, ihmisille jne. Vielä parempi käyttöliittymä tarjoaa käyttäjälle mahdollisuuden perua tai korjata virheelliset toimet. Tällöin järjestelmä on anteeksiantavainen. (UX Planet 2017)

Vaikka käyttöjärjestelmä olisi suunniteltu kuinka hyvin tahansa, käyttäjä voi siltikin tehdä inhimillisiä virheitä. Nämä virheet voivat johtua käyttäjän tahattomasti väärin antamasta syötteestä, komennosta (esim. käyttäjä painaa vahingossa väärää painiketta) tai käyttäjän virheellisestä loogisesta päätöksestä (esim. käyttäjä tekee väärän päätöksen valitessaan syötettä tai komentoa). Professori Ben Shneiderman (2016) kirjastaan kertovassa artikkelissa linjaa, että käyttäjäystävällisen käyttöliittymän tulisi välttää tilanteita, jotka todennäköisesti johtavat virheisiin. Virhetilanteiden sattuessa käyttöjärjestelmän pitäisi kertoa käyttäjälle selkeästi virheistä, jotta käyttäjä osaa ottaa oikeat toimet asian korjaamiseksi. (Shneiderman 2016)

Käyttäjäystävällinen käyttöliittymä tarjoaa käyttäjälle hyödyllistä informaatiota, jotta hän osaa syöttää tarvittavat tiedot tai toimia oikein kaikissa tilanteissa. Käyt-

täjän on hyvä myös saada reaaliaikaista palautetta, että järjestelmä on vastaanottanut annetut syötteet tai komennot, jotta käyttäjä voi olla varma annettujen komentojen toimittamisesta. Kuviossa 3 on esitettynä painikeobjekti, joka muuttuu aktivoimisen jälkeen. Vasemmalla painike ennen painamista ja oikealla painamisen jälkeen. Ennen aktivoimista painike on valkoinen ja siinä lukee ”Käynnistä”. Aktivoimisen jälkeen painike vaihtaa väriään siniseksi, ja siinä lukee ”Sammuta”. Näin ollen käyttäjä saa välittömän palautteen komennon perille menosta, sekä voi jälkeenpäin tarkistaa painikkeen tilan näytöltä.

KUVIO 3. Kuvitteellisen järjestelmän ohjainpainike, joka muuttuu aktivoimisen jälkeen väreiltään käänteiseksi.

2.4.4 Selkeys

Selkeys on yksi käyttöliittymän tärkeimmistä ominaisuuksista. Jotta käyttöliittymä voi olla tehokas ja käyttäjäystävällinen, Shneidermanin mukaan sen pitää olla tunnistettava, ennalta-arvattava ja sillä pitää olla selvä tarkoitus. Käyttäjän pitää pystyä näkemään käyttöliittymästä mitä hän on tekemässä ja minkä kanssa hän on vuorovaikutuksessa, ilman että asiassa on minkäänlaista epäselvyyttä. Selkeys antaa käyttäjälle halua jatkaa käyttöliittymän käyttämistä ilman epävarmuutta. (Shneiderman 2016)

Käyttöliittymän selkeyttäminen vähentää käyttäjän kognitiivista kuormaa, joka tarkoittaa psyykkistä prosessointia vaativien tehtävien vähentämistä tai keventämistä. Käyttäjän kognitiivista kuormaa voidaan pienentää suunnittelemalla käyttöliittymä niin, ettei se pakota käyttäjää muistamaan suuria määriä asioita, vaan että tarvittava tieto olisi pikemminkin nähtävissä aina. Kuitenkaan tietoa ei saisi

olla kerralla liikaa esillä, jotta käyttäjä pystyy prosessoimaan sitä tehokkaasti. Samaa tietoa voidaan myös esittää helpommin luettavassa muodossa käyttämällä selkeää jakoa, sekä helppolukuisia ja selkeitä objekteja ja fontteja (kuvio 4). Tehtävien suorittamiseen vaadittavien toimintojen ja vaiheiden lukumäärän pitämisellä mahdollisimman pienenä on selvästi positiivinen vaikutus kognitiiviseen kuorman pienentämiseen, sekä käyttöliittymän käytettävyyden parantamiseen. (Babich, 2019)

KUVIO 4. Sama numerosarja esitettynä kahdella eri tavalla. Alempi on huomattavasti selkeämpi, helpompi sisäistää ja tyyliltään asiallisempi.

3 KÄYTTÖLIITTYMÄPÄIVITYKSEN ESITIEDOT

Tässä luvussa kerrotaan opinnäytetyön taustoista. Lukuun sisältyy mm. taustatietoa yrityksistä, laitteistoista ja niissä käytetyistä ohjelmistoista. Lukuun on tiivistetysti kerrottu opinnäytetyön kannalta keskeisimmistä työkaluista ja ohjelmistoista, sekä hieman avattu PLC:n ja käyttöliittymän välistä Modbus-rajapintaa.

3.1 Työn tausta

Opinnäytetyön tilaajana toimi UCSolutions Oy. UCSolutions on pieni tamperelainen automaatiopalveluja tarjoava yritys. He tarjoavat automaation kokonaisratkaisuja suunnittelusta aina käyttöönottoon asti. UCSolutionsin toimialaa kotimaassa on kaikki automaatio sekä ohjelmointiprojektit. Ulkomailla he toimivat pääasiassa lasikonealalla, jossa heillä on monia asiakkaita. Yksi näistä asiakkaista on GLASSource.

GLASSource on yhdysvaltalainen lasituotteita valmistava teollisuusyritys. Heidän tuotantotilansa sijaitsevat Grand Havenissa, Michiganin osavaltiossa. GLASSource on erikoistunut monenlaisiin teollisuuden lasi- ja peilituotteistiin. Yhtenä heidän erikoisosaamisestansa ja tarjoamistaan palveluista on lasin lämpökarkaisu.

Lasin lämpökarkaisulla eli termisellä karkaisulla tarkoitetaan lasin lämpökäsittelemistä niin, että sen ominaisuudet muuttuvat. Ensin lasi kuumennetaan uunissa haluttuun lämpötilaan ja sitten jäähdytetään nopeasti puhaltamalla ilmaa lasin pinnalle. Termisesti karkaistun lasin vetolujuus on moninkertainen verrattuna tavalliseen lasiin. Lasin terminen karkaiseminen muuttaa myös tapaa, jolla lasi särkyy – termisesti karkaistu lasi särkyy pieniksi muruiksi, jotka ovat ihmiselle suhteellisen vaarattomia. (Glasstech, n.d.)

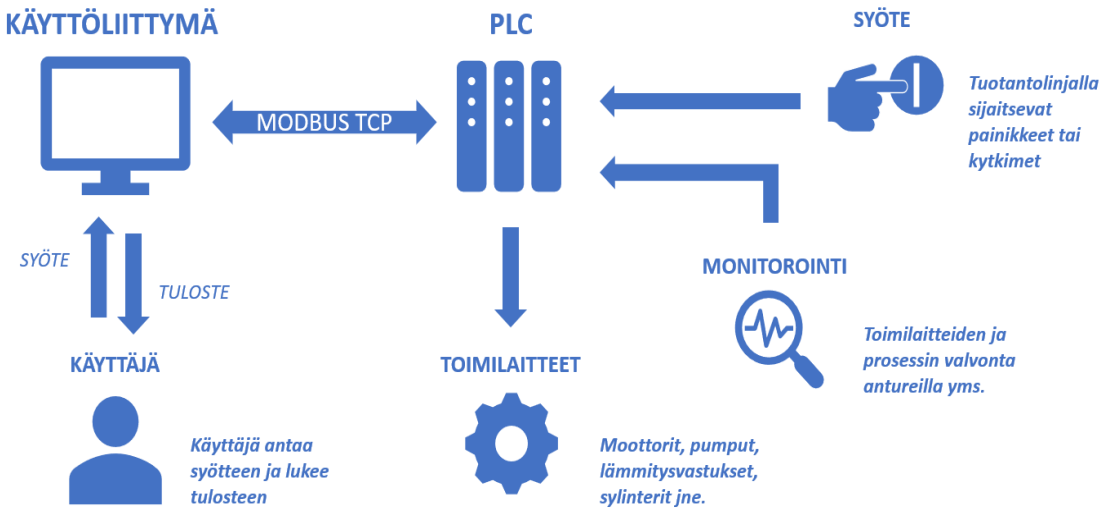
GLASSource karkaisee lasinsa Uniglassin lasikarkaisulinjalla (kuva 5). Karkaisulinjaa ohjataan graafisella käyttöliittymällä, joka on suorassa yhteydessä PLC:hen (programmable logic controller). PLC on suoritinpohjainen ohjelmoitava logiikkayksikkö, joka valvoo järjestelmän lähtöjä ja tuloja, sekä tekee loogisia päätöksiä automatisoiduille prosesseille ja laitteille (Carlos Gonzalez, 2015).

KUVA 5. Uniglassin valmistama lasinkarkaisulinja (Glas In Beeld, 2014)

Kuvassa 6 on nähtävissä Modiconin TSX Micro -PLC, jollainen GLASSourcella on käytössä karkaisulinjan ohjauksessa. Kaikki karkaisulinjan toimilaitteet, anturit ja painikkeet ovat yhteydessä PLC:hen, joita se lukee ja ohjaa ohjelmansa ja käyttöliittymältä saatujen ohjauksien mukaisesti.

KUVA 6. Modicon TSX Micro -PLC (Schneider electric, n.d.)

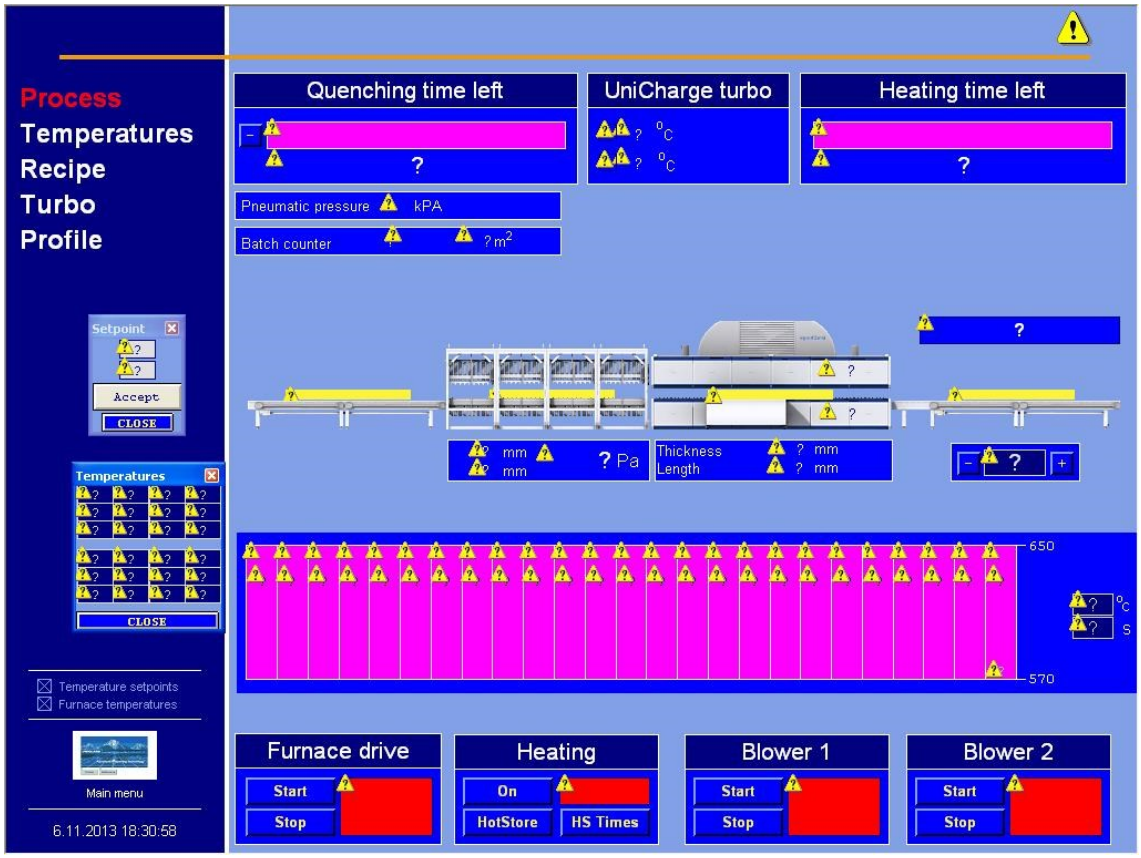
Käyttöliittymä on tärkeä osa tuotantolinjaa. Käyttöliittymän kautta koneen käyttäjä ohjaa tuotantoa, sekä määrittää tuotannossa olevan tuotteen parametrit. Käyttöliittymä toimii linkkinä PLC:lle ja näin ollen koko järjestelmälle. Käyttöliittymä on yhteydessä PLC:hen Modbus TCP -väylällä. Modbus on Modiconin (nykyään Schneider Electric) vuonna 1979 julkaisema sarjaliikenneprotokolla, joka oli aluksi tarkoitettu käytettäväksi pelkästään Modiconin omien PLC:iden kanssa ja siitä on sittemmin muodostunut yleinen standardi teollisuudessa, jossa se on yleisesti käytössä erilaisten laitteiden välisessä kommunikoinnissa (Mobdus, n.d.). PLC vastaanottaa käyttöliittymään annetun syötteen ja toimittaa käyttöliittymälle tulosteen. PLC ohjaa toimilaitteita logiikkaohjelmansa mukaan, johon vaikuttaa syöte sekä antureilta saatu data. Kuviossa 5 on nähtävissä tuotantolinjan eri osien väliset kommunikaatiorajapinnat. Nuolien suunnat kuvaavat mihin suuntaan laitteiden välinen kommunikaatio tapahtuu.



KUVIO 5. Lasinkarkaisulinjan eri osien suhde toisiinsa sekä kommunikointi

GLASSourcen lasinkarkaisulinjaston kommunikaatio käyttöliittymän ja ohjelmoitavan logiikan välillä perustui vanhaan teknologiaan. Maahantuoja ei enää tue laitteeseen liittyviä osia. Tämä on valtava riski ja se voi johtaa tuotannon seisahtamiseen viikoiksi. Tämä on yksi iso syy minkä takia GLASSource päätti tilata käyttöliittymäpäivityksen – modernilla käyttöliittymällä ja uudella robustilla teollisuus-PC:llä voidaan alentaa riskejä huomattavasti ja tuoda tuotantoon paljon uusia ominaisuuksia.

GLASSourcella oli karkaisulinjalla käytössään vanha Windows XP -käyttöjärjestelmälle pohjautuva käyttöliittymä. Käyttöliittymä on nykypäivän standardeilla vanhanaikainen sekä visuaalisuudeltaan, että toiminnallisuudeltaan. Myös tietokoneen sekä sen käyttöjärjestelmän elinkaari alkoi olemaan lopuillaan. Tietokone oli seissyt teollisuuslaitoksen tuotantotiloissa monta vuotta ilman minkäänlaista pölysuojausta. Windows on lopettanut XP-käyttöjärjestelmän päivittämisen jo vuosia sitten, joten toimintavarmuus on alentunut ja tietoturvariskit ovat valtavat. Kuvassa 7 on näkyvissä vanha käyttöjärjestelmä.



KUVA 7. Prosessinäkymä vanhasta käyttöjärjestelmästä, jossa nähtävissä selvästi vanhentunut tyyli ja värimaailma.

Työn tarkoituksena oli siis toteuttaa käyttöliittymäpäivitys sekä parannella ja optimoida uutta käyttöliittymää. Vanhasta käyttöliittymästä olisi siirrettävä toiminnallisuudet ja rekisteriosoitteet uuteen käyttöliittymään. Sen lisäksi PLC:lle pitäisi tehdä muutamia muutoksia. Uuden käyttöliittymän ulkoasua tultaisiin kehittämään ja osa PLC:n koodista siirrettäisiin sinne. Lopuksi uusi käyttöliittymä asennettaisiin uudelle modernille teollisuus-PC:lle.

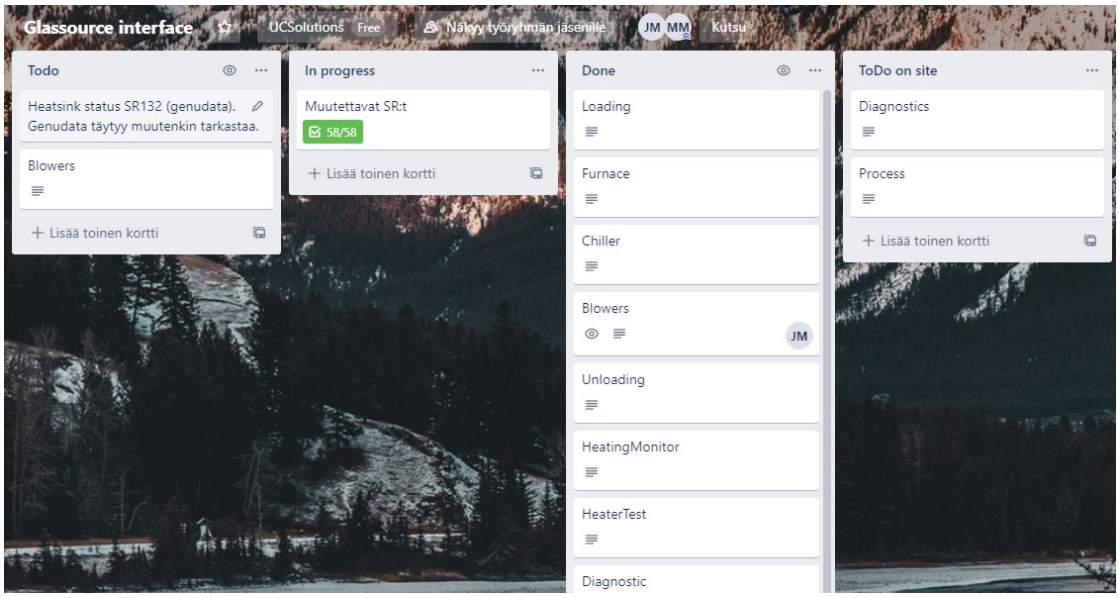
UCSolutions oli sopinut GLASSourcen kanssa käyttöliittymäpäivityksen ajankohdasta. Tämän vuoksi oli ehdottoman tärkeää, että käyttöliittymäpäivityksen ohjelmistopuolen muutokset olisivat valmiina, kun sovittu aika koittaisi. Paikan päällä ollessa ei olisi aikaa tehdä muuta kuin uuden tietokoneen asennus, kytkennät, sekä lopulliset testaukset ja korjaukset. Asiakkaan sijainnin takia käyttöliittymän päivityksen oli onnistuttava kerralla.

3.2 Suunnittelutyökalut

Tässä luvussa käydään läpi opinnäytetyössä käytettävät suunnittelutyökalut. Luvussa ensimmäisenä käsitellään projektin aikataulutuksessa ja ylläpidossa käytettyä projektinhallintatyökalua Trelloa. Tämän jälkeen luvussa käydään läpi käyttöliittymän päivittämisessä käytetyt ohjelmistot: Vijeo Look, iX Developer sekä PL7 Pro.

3.2.1 Trello

Trello on ilmainen projektinhallintaan suunniteltu työkalu (kuva 8). Sen toimintaperiaate on hyvin yksinkertainen: Se pohjautuu Toyotan kehittämään Kanbaniin. Kanbanissa keskeisessä asemassa on Kanban-taulu. Se koostuu pystysuorista sarakkeista, joista jokainen kuvaa yhtä projektin vaihetta. Perinteisessä Kanban-taulussa on kolme eri vaihetta: Tehtävät työt, työn alla olevat työt ja tehdyt työt. Näihin sarakkeisiin sijoitetaan kortteja, joista jokainen sisältää yhden työtehtävän. Kortteja liikutellaan taululla sarakkeiden välillä projektin etenemisen mukaan. Tämä helpottaa projektin hallintaa suunnattomasti ja antaa selkeän kokonaiskuvan projektin edistymisestä, sekä toimii muistilistana projektin aikana. (Karri Hiekkanen, 2016)



KUVA 8. Työssä käytetty Trello-taulu, jossa nähtävissä työvaiheet ja niihin sisältyvät kortit.

Trellon etuna on sen todella helppokäyttöinen ja selkeä käyttöliittymä. Kevyillä ja yksinkertaisilla työkaluilla on kuitenkin mahdollista hallita laajojakin projekteja. Tehtäväkortteja on yksinkertaista luoda, niihin voidaan kirjoittaa lisäkuvauksia, määrittää tarkistuslistoja, määrittää ketä käyttäjiä kortti koskee sekä asettaa määräaikoja ja liitetiedostoja jne. Trelloa käyttääkin monet ja menestyvät yrityksen, kuten esimerkiksi Google, Adobe, Pixar ja National Geographic (Karri Hietanen, 2016).

Tässä työssä Trello osoittautui tärkeäksi työkaluksi. Sitä käytettiin päivittäin työn tilaajan kanssa kommunikoimiseen, sekä projektin statuksen ylläpitämiseen. Jokainen työvaihe voitiin erikseen kirjata ylös taululle, ja merkata tehdyksi sitä mukaa, kun projekti eteni. Näin sekä työn tekijä että työn tilaaja olivat perillä projektin etenemisestä.

3.2.2 Vijeo Look

Vijeo Look oli Schneider Electricin kehittämä pieniin ja keskikokoisiin sovelluksiin tarkoitettu SCADA-ohjelmisto (Supervisory Control And Data Acquisition) (Schneider Electric, n.d.). SCADA on teollisuuskäyttöön tarkoitettu valvontajärjestelmä, jolla on mahdollista:

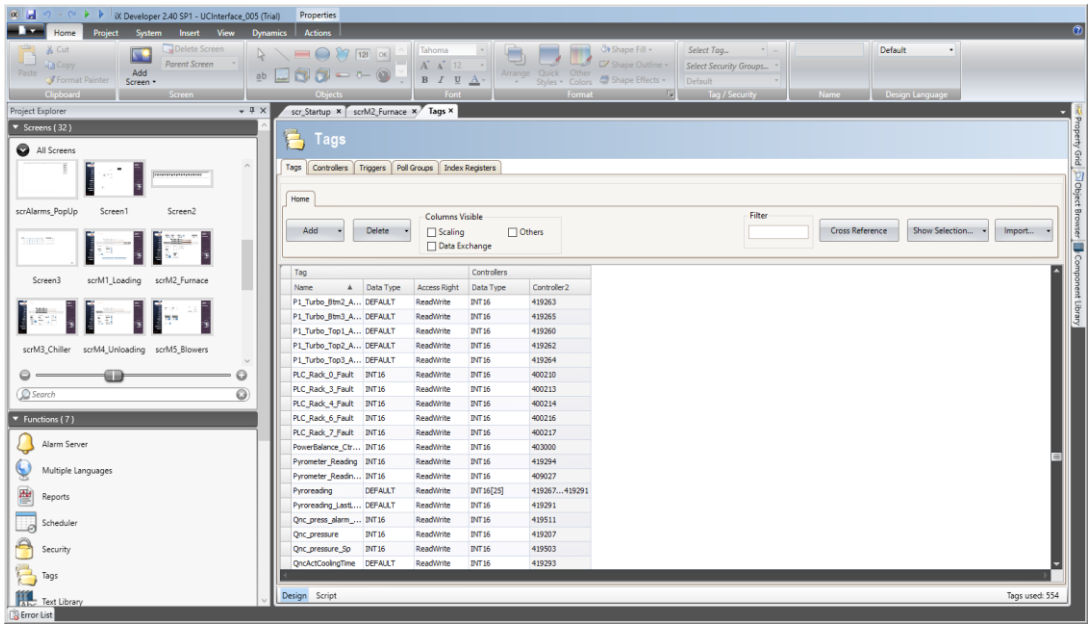
- olla graafisen käyttöliittymän kautta suorassa vuorovaikutuksessa erilaisten antureiden ja toimilaitteiden kuten venttiilien, moottorien ja pumppujen kanssa.
- monitoroida, kerätä ja prosessoida dataa reaaliaikaisesti.
- ohjata prosesseja paikallisesti tai etänä.
- tallentaa tapahtumia ja tietoa lokitiedostoon (Inductive Automation, 2018).

Vijeo Lookin avulla on mahdollista helposti luoda graafinen käyttöliittymä, joka on yhteydessä järjestelmää ohjaavaan PLC:hen. Vijeo Look oli ideaali suunnittelu- ja käyttöohjelmisto pieniin sovelluksiin yksinkertaisuutensa ja helppokäyttöisyytensä vuoksi. Se on ohjelmistona kuitenkin niin vanha ja ajastaan jäljessä, ettei sitä käytetä enää.

Vijeo Lookia käytettiin tässä työssä vanhan käyttöliittymän opiskeluun ja tutkimiseen. Oli ensiarvoisen tärkeää, että uudessa käyttöliittymässä oli käytössä kaikki vanhan käyttöliittymän ominaisuudet ja että uuden käyttöliittymän lähdöt ja tulot olisivat samat kuin vanhassa. Jokaisen syöte- ja tulosteobjektin rekisteriosoite piti erikseen selvittää, jotta ne voitaisiin toteuttaa uudessa käyttöliittymässä ja sen kommunikaatio PLC:n kanssa toimisi. Näistä käyttökohteista on kerrottu lisää luvussa 4.2, jossa on kerrottu tarkemmin kommunikaatiorajapinnan tekemisestä.

3.2.3 iX Developer

iX Developer on Beijer Electronicsin kehittämä ohjelmisto, jolla voidaan toteuttaa iX-paneelissa tai PC:llä toimivia käyttöliittymiä (kuva 9). Se sisältää kaikki perustoiminnallisuudet, joita prosessisovelluksessa tarvitaan. iX Developerilla on helppoa luoda loogisia, joustavia ja tehokkaita käyttöliittymiä, jotka tarjoavat käyttäjälleen oikeaa informaatiota oikeaan aikaan. Se sisältää myös edistyneempiä toiminnallisuuksia, kuten C# -scriptauksen. (Beijer Electronics, n.d)



KUVA 9. iX Developerin käyttöliittymä ja sen tag-lista

iX Developeria käytetään käyttöliittymän ja järjestelmän ohjelmiston suunnittelussa ja toteuttamisessa. Kun ohjelmisto on valmis, siitä luodaan ohjelmistopaketti. Tämän ohjelmistopaketin suorittamiseksi täytyy järjestelmää ohjaavalle tietokoneelle asentaa iX Runtime. iX Runtime on pakollinen, mikäli halutaan ajaa ohjelmistopakettia eli käyttää käyttöliittymää. iX Runtimen pystyy asentamaan millekä tahansa tietokoneelle, jonka halutaan ohjaavan järjestelmää. Vaatimuksena on, että tietokone on yhdistettynä järjestelmään väylällä ja että käytettävässä tietokoneessa on iX Runtimen vaatima lisenssi. Lisenssit ovat tietokonekohtaisia, joten jokainen käyttöliittymää suorittava tietokone vaatii erillisen lisenssin.

iX Developer oli tämän opinnäytetyön keskeisin ohjelmisto. Kaikki vanhat toiminnallisuudet oli toteutettava uudessa käyttöliittymässä, ja myös osa ennen PLC:ssä toteutetusta koodista oli siirrettävä nyt käyttöliittymän sisälle, josta kerrotaan tarkemmin luvussa 4.3. Monimutkaisemmat toiminnallisuudet oli pakko toteuttaa käyttämällä iX Developerin Script -työkalua, jossa voitiin toteuttaa halutut toiminnot C#:lla koodaamalla.

3.2.4 PL7 Pro

PL7 Pro on Telemecaniquen kehittämä ja nykyisin Schneider Electricin omistama ohjelmointiohjelmisto. Sillä on mahdollista ohjelmoida esim. Schneiderin Modicon PLC:tä. PL7 Pro:ssa on saatavilla seuraavia ominaisuuksia:

- Sovelluskohtaiset moduulit, joita ei tarvitse erikseen ohjelmoida.
- Mahdollisuus ohjelmoida usealla eri IEC 61131 standardin ohjelmointikielellä esim. FDB (Function block diagram), LD (Ladder diagram) tai ST (Structured text).
- Esimääritetyt toiminnot, kuten erilaiset ajastimet ja laskurit.
- Mahdollisuus tehdä muutoksia PLC-ohjelmaan ajon aikana (On-line muutos).
- Animoitu koodi, joka havainnollistaa selkeästi mitkä toiminnot tai tulot/lähdöt ovat aktiivisena.

- Mahdollisuus lukea rekistereitä ja muisteja ohjelman verifioimiseksi tai vikojen etsimiseksi. (Schneider Electric, n.d.)

PL7 Pron etuna on sen helppous ja tehokas virheiden etsintä (debugging). Myös käytön ja diagnostiikan helppous tekevät siitä yksinkertaisen ja suorituskykyisen ratkaisun. Moduulipohjaisuus tuo joustavuutta ja selkeyttä ohjelmoimiseen. (Schneider Electric, n.d.)

PL7 Pro toimi tässä työssä keskeisessä asemassa, sillä PLC on linkki käyttöliittymän ja tuotantolinjan toimilaitteiden ja antureiden välillä. PL7 Prota käytettiin varmistamaan, että PLC:n koodi lukee ja kirjoittaa oikeita rekisteriosoitteita ja näin ollen väylä PLC:n ja käyttöliittymän välillä toimii. PL7 Prota käytettiin myös pienten muutosten tekemiseen logiikkaohjelmaan.

3.3 Modbus-rajapinta

Kuten aikaisemmin mainittiin, käyttöliittymän ja PLC:n välinen kommunikointi tapahtuu Modbus TCP -väylän avulla. Modbus-väylä on yksi opinnäytetyön keskeisimpiä asioita, ja rajapinnan tekemiseen tullaankin paneutumaan tarkemmin kappaleessa 4.2. Modbus-väylällä on rekisteriosoitteiksi kutsuttuja osoitteita, joita käytetään varastoimaan lukuarvoja. Rekisterit on jaettu neljään ryhmään:

- Lähtökela (Coil), rekisteriosoite 0xxxxx.
- Tulo (Discrete Input), rekisteriosoite 1xxxxx.
- Tulorekisteri (Input register), rekisteriosoite 3xxxxx.
- Pitorekisteri (Holding register), rekisteriosoite 4xxxxx.

Eri rekisteriryhmät on eroteltu toisistaan etumerkillä. Etumerkki kuvaa sitä, mihinkä rekisteriryhmään kyseinen rekisteri kuuluu (esim. tulorekisterin etumerkki on 3). Rekisterien merkintä saattaa vaihdella laitteen iän mukaan, sillä rekisteriryhmien koko riippuu laitteen iästä – vanhemmat laitteet tukevat 9999 rekisteriä per rekisteriryhmä, uudemmat jopa 65535 rekisteriä. Tämän takia vanhemmissa

laitteissa tulorekisterin osoite 112 merkitään muotoon 30112, kun taas uudemmissa laitteissa se on muodossa 300112. Kuitenkin nämä kaksi viittaavat samaan rekisteriin. (Kepware, 2018)

Lähtökelojen ja tulojen yksittäisen rekisterin koko on yksi bitti. Tämä tarkoittaa sitä, että näissä rekistereissä on mahdollista säilyttää vain tilatietoa, joka on joko 0 tai 1. Tulorekisteiden ja pitorekistereiden koko on 16 bittiä, jolloin niissä voidaan säilyttää esimerkiksi kuuttatoista eri tilatietoa (16 kappaletta yksittäisiä bittejä, jotka voivat olla joko 0 tai 1) tai vaikkapa 16 bittistä lukuarvoa, jolle voidaan antaa arvo 0 ja 65 535 väliltä. (National Instruments, 2019)

Käyttöliittymän jokaisella dataa sisältävällä elementillä on rekisteriosoite. Myös PLC:n eri toiminnoilla on olemassa rekisteriosoitteita. Tämän rekisteriosoitteen avulla käyttöliittymässä tai PLC:ssä oleva komento, tulosteobjekti, syöte tai vaikkapa laskuri voidaan sitoa tiettyyn osoitteeseen, jota voidaan sekä lukea, että kirjoittaa PLC:llä ja käyttöliittymässä (jos rekisteri tai sovellus sen sallivat). Kuvitellaan tilanne, jossa käyttöliittymässä on mahdollista tarkastella prosessissa olevan moottorin kierrosnopeutta. Moottorissa oleva takometri mittaa moottorin pyörimisnopeutta ja antaa signaalin PLC:lle, joka kirjoittaa takometriltä saatavaa dataa rekisteriosoitteeseen 400010. Kun käyttöliittymä on Modbus-väylällä kiinni PLC:ssä, voidaan käyttöliittymään tehty tulosteobjekti sitoa samaan rekisteriosoitteeseen kuin PLC:llä. Näin ollen käyttöliittymä lukee samaa rekisteriä, jota PLC kirjoittaa ja tällöin haluttu tieto saadaan näkymään näytöllä. Signaali on tietenkin jossain vaiheessa skaalattava ja muunneltava, jotta se saadaan käyttäjälle helposti luettavaan muotoon (esimerkiksi takometriltä saatavat hertsit muutetaan kierrosta/minuutti muotoon).

4 KÄYTTÖLIITTYMÄPÄIVITYS

UCSolutionsin käyttöliittymä on itsessään valmis toimiva käyttöliittymä, joka tavallisesti vaatii ainoastaan konfigurointia asiakkaan laitteiston ja tarpeiden mukaan. Monet toiminnallisuudet ovat siinä valmiiksi toteutettuja ja toimintakuntoisia ja vaativat ainoastaan rajapinnan tekemisen. Joitain toiminnallisuuksia voi joutua tekemään tyhjästä asiakkaan tarpeiden täyttämiseksi. Tässä luvussa kerrotaan projektin hallinnasta sekä uuden käyttöliittymän muokkaamisesta toimintakuntoiseksi. Lukuun sisältyy mm. uuden käyttöliittymän Modbus-rajapinnan toteuttaminen, PLC-koodin muutokset, sekä käyttöliittymän tehdyt muutokset ja lisäykset. Lukuun on lyhyesti kerätty esimerkit opinnäytetyön kannalta keskeisimmistä työtehtävistä.

4.1 Projektin hallinta

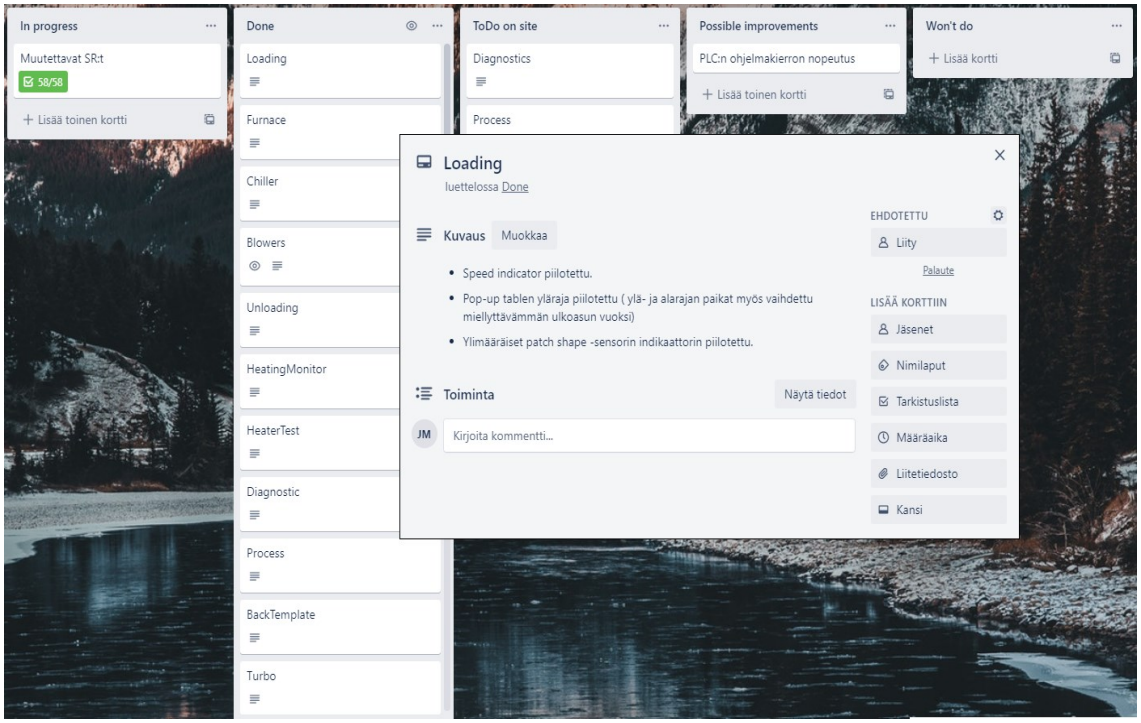
Ennen varsinaista opinnäytetyötehtävän aloittamista oli projekti suunniteltava ja aikataulutettava, jotta käyttöliittymäpäivitys olisi valmiina käyttöönottopäivään mennessä. Tämä oli ehdottoman tärkeää, sillä asiakkaan tuotantotilat sijaitsivat Yhdysvalloissa, minkä vuoksi aikataulussa ei ollut mahdollisuutta joustaa – käyttöliittymän oli oltava toimintakuntoinen viimeistään matkaan lähdettäessä.

Projektin aikataulutus ja ylläpito tapahtui Trellon sekä Microsoft Exceliin luodun Gantt-kaavion (liite 1) avulla. Gantt.com -sivuston (2020) mukaan Gantt-kaavio on projektinhallinnassa suosittu ja yleisesti käytetty työkalu. Siitä on helposti nähtävissä esimerkiksi mitä tehtäviä projekti sisältää ja koska mikäkin tehtävä alkaa ja loppuu. Ensimmäisen Gantt-kaavion loi puolalainen insinööri Karol Adamiecki 1890-luvun puolivälissä. Noin 15 vuotta myöhemmin yhdysvaltalainen insinööri ja projektinhallintakonsultti Henry Gantt suunnitteli oman versionsa kaaviosta ja siitä tuli laajalti tunnettu länsimaissa. (Gantt.com, 2020).

Opinnäytetyöprojektin Gantt-kaavioon luotiin pääotsikkotasolla lista työtehtävistä, joita projektiin sisältyi, sekä aikataulut jokaiselle työtehtävälle. Aikataulutus

kaavioon tehtiin viikkotasolla. Näin projekti pystyttiin helposti jakamaan aikataulun suhteen suurpirteisiin osiin. Jokaiselle projektin tehtävällä annettiin suunniteltu aloitusaika ja suunniteltu kesto. Nämä kaksi lukua antoivat ohjearvot projektin kestolle. Työtehtävän tullessa päätökseen kaavioon täytettiin tehtävän todellinen aloitusaika, sekä todellinen kesto. Tällä tavalla aikataulua pystyttiin seuraamaan projektin aikana, sekä suorittamaan korjaavia toimenpiteitä tarpeeksi ajoissa, jos oli tarpeen.

Trellossa työtehtävät pilkottiin pienempiin osiin – työkortteihin – jolloin projektista tuli helpommin lähestyttävä. Jokainen työkortti edusti yhtä käyttöliittymän osaa eli näkymää. Kun työ sen näkymän osalta oli valmis, kortti voitiin siirtää keskeneräisten töiden sarakkeesta valmiiden töiden sarakkeeseen. Näin pystyttiin pitämään lukua siitä mikä osa käyttöliittymästä oli kesken ja mikä valmiina. Trellon työkortteihin oli myös mahdollista kirjoittaa työtehtäviin tai näkymiin liittyviä muistiinpanoja ja huomioita projektin edetessä. Tällä pystyttiin seuraamaan, mitä muutoksia kyseiseen näkymään oli tehty. Kuvassa 10 on nähtävissä projektissa käytetty Trello-taulu. Kuvassa on myös nähtävissä Loading-työkortti avattuna. Kuten havaitaan, työkortista on helppo tarkistaa, mitä muutoksia kyseiseen näkymään on tehty.



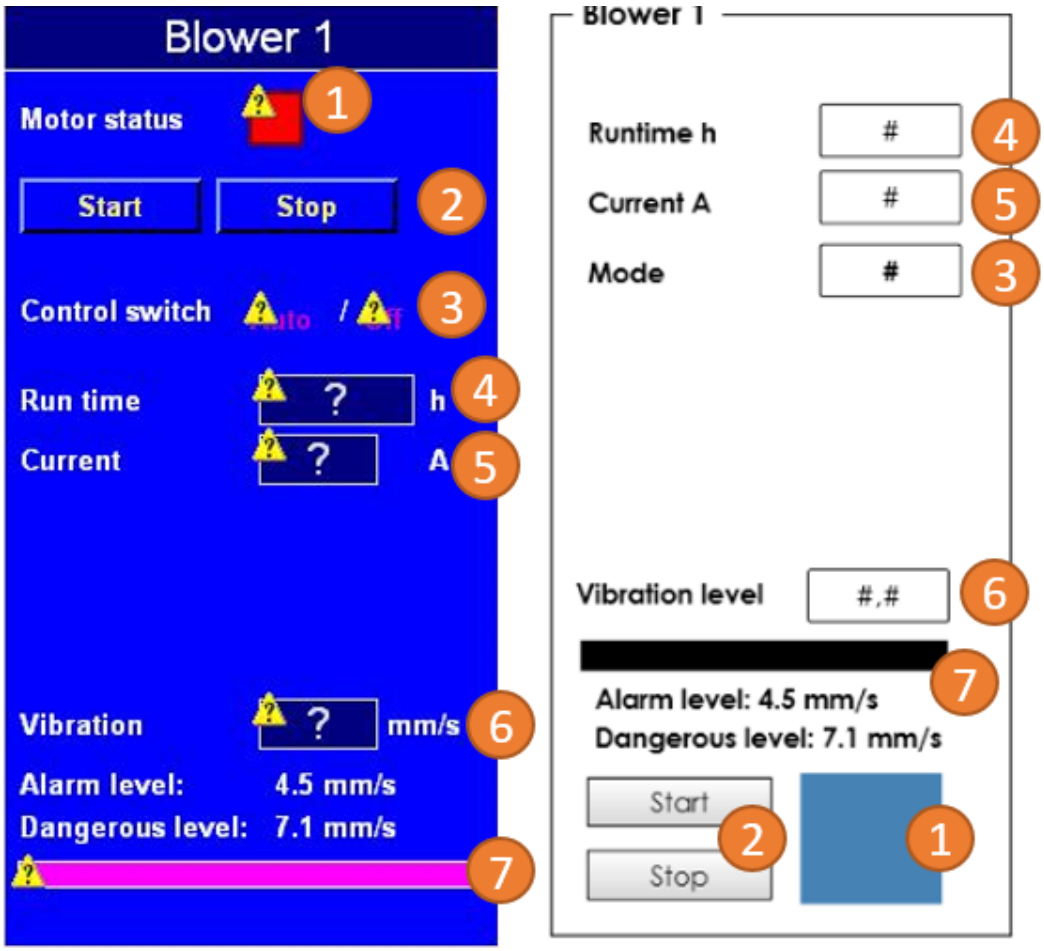
KUVA 10. Opinnäytetyön Trello-taulun näkymä, jossa Loading-työkortti avattuna.

Näiden kahden projektihallintatyökalun lisäksi projektin eri työtehtävistä tehtiin muistiinpanoja ruutupaperille töiden aikana. Ruutupaperille tehdyt muistiinpanot toimivat lyhyen aikavälin muistina – kaikki muu pitempiaikaista tallennusta vaativa informaatio tallennettiin Trelloon työkorteille. Tämä toimi hyvänä ja nopeana tapana säilyttää informaatiota lyhyitä aikoja sen hetkistä työtä tehdessä. Näillä tavoilla projektia ja sen aikataulua pystyttiin hallitsemaan niin, että työ pysyi aikataulussa.

4.2 Modbus-rajapinnan tekeminen

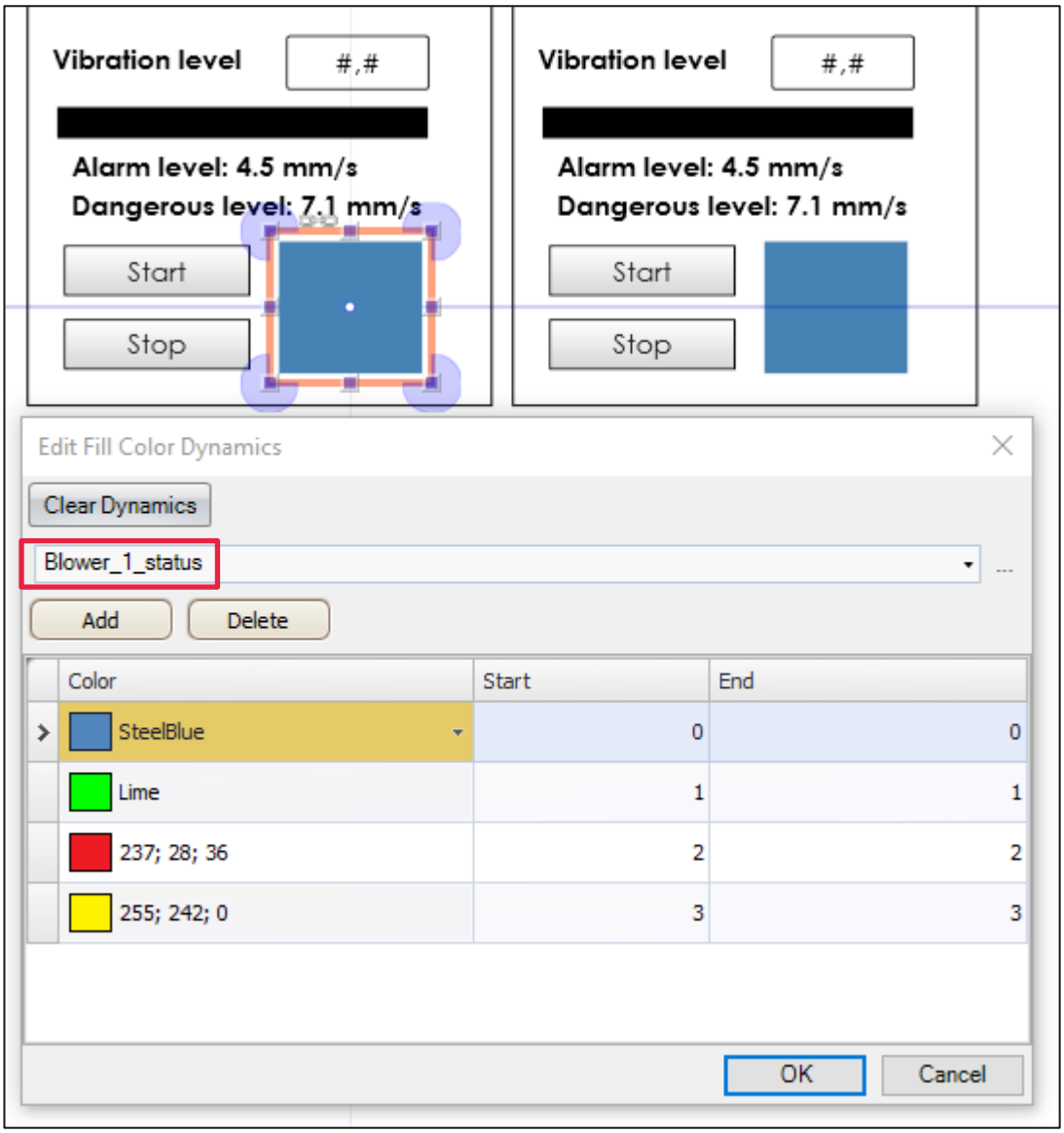
Ensimmäisenä opinnäytetyössä oli tarkoitus siirtää kaikki toiminnallisuudet vanhasta käyttöliittymästä uuteen. Uusi käyttöliittymä oli valmiiksi objekteiltaan ja toiminnallisuuksiltaan melkein samanlainen kuin edeltäjänsä. Jos uudessa käyttöliittymässä ei ollut jotain vanhan käyttöliittymän objektia, se oli luotava tyhjästä. Jokainen vanhassa käyttöliittymässä ollut toiminnallisuus ja objekti oli saatava toimimaan uudessa käyttöliittymässä. Tämä tapahtui Modbus-rajapinnan tekemisellä, eli rekisteriosoitteita asettamalla.

Kuvassa 11 on nähtävissä uuden ja vanhan käyttöliittymän sama näkymä, jossa on ykköspuhaltimen kannalta tärkeimmät syötteet ja tulosteet. Moottorin tila on esitettynä väriä vaihtavalla laatikolla (1). Vanhassa käyttöliittymässä moottoritilan indikaattori vaihtaa tilan mukaan väriään punaisesta vihreään. Uudessa käyttöliittymässä ei-aktiivisten tilojen indikaattorina toimii muulta käyttöliittymästä tuttu teräksensininen ja aktiivisten tilojen indikaattorina vihreä. Tässä on haettu käyttöliittymän tyylin yhtenäistämistä (luku 2.4.2). Puhaltimen moodin indikaattorinäkymästä on luotu muiden osiossa olevien tulosteiden kanssa yhtenäinen (3). Puhaltimen tärinätasosta ilmoittava palkki on muutettu väriltään paremmin tyyliin sopivaksi (7). Muuten näkymä on melkein identtinen vanhan kanssa. Jokainen kuvassa 11 oleva vanhan käyttöliittymän objekti on linkitettynä tiettyyn rekisteriosoitteeseen. Jotta uuden käyttöliittymän objektit toimisivat samanlailla kuin vanhan käyttöliittymän vastaavat, oli ne linkitettävä samoihin ositteisiin.



KUVA 11. Käyttöliittymien ykköspuhaltimien näkymät vanhasta ja uudesta käyttöliittymästä.

Valitaan esimerkkiobjektiksi puhaltimen tilaindikaattori. Jokaisen objektin rekisteriosoitteen asettaminen aloitettiin selvittämällä Vijeo Lookilla vanhassa käyttöliittymässä kyseiseen objektiin linkattu rekisteriosoite. Tässä tapauksessa objektille oli linkattu rekisteriosoite 9021. Tätä numeroa tullaan myöhemmin tarvitsemaan varmistettaessa, että PLC:ssä käsiteltävä muuttuja on oikea. Seuraavaksi selvitettiin uuden käyttöliittymän vastaavaan objektiin linkattu tagi. Tilaindikaattorien toiminta perustuu eri väreihin ja nämä eri värit viestivät käyttäjälle missä tilassa laite kulloinkin on. Tällöin tagi on linkattava objektin väritäyttöön, joka tagiin linkatun rekisteriosoitteen arvon mukaan muuttaa objektin väriä. Kuvassa 12 on esitettynä puhaltimen tilaindikaattorin objektin väritäytön valinnat. Kuten väritäytön valintaikkunassa on nähtävissä, objektilla on olemassa neljä eri tilaa, jotka muuttuvat arvojen 0–3 välillä. Jos esimerkiksi rekisteriosoitteen arvo on 2, silloin käyttöliittymässä oleva tilaindikaattori on väriltään punainen, ja viestii käyttäjälle laitteessa olevasta viasta. Objektiin linkattu tagi ”Blower_1_status” on korostettu kuvassa punaisella laatikolla.



KUVA 12. Puhaltimen tilaindikaattorin väritäytön valintaikkuna

Seuraavaksi on selvitettävä mihinkä rekisteriosoitteeseen objektin tagi on sidottu. Sen pystyy selvittämään iX Developerin sisäisestä Tag-listasta (kuva 13). Tag-listasta pystytään helposti rajaamaan tagit nimen perusteella, jolloin saadaan listaan näkymään vain halutut. Kuvassa 13 on rajattu lista niin, että siinä on nähtävissä haluttu Blower_1_status -tagi. Kuten kuvasta voidaan nähdä, puhallin 1 tilan tagi on sidottu rekisteriosoitteeseen 419254 (korostettu kuvassa punaisella).

Tag			Controllers	
Name	Data Type	Access Right	Data Type	Controller2
Blower_2_Power	DEFAULT	ReadWrite	INT16	419510
Blower_1_Power	DEFAULT	ReadWrite	INT16	419509
Blower_2_Vibra	FLOAT	ReadWrite	INT16	419508
Blower_1_Vibra	FLOAT	ReadWrite	INT16	419507
Blower_2_Current_A	DEFAULT	ReadWrite	INT16	419505
Blower_2_Runtime_h	DEFAULT	ReadWrite	INT16	419504
Blower_1_Current_A	INT16	ReadWrite	INT16	419501
Blower_1_Runtime_h	INT16	ReadWrite	INT16	419500
Blower_2_Status	DEFAULT	ReadWrite	INT16	419255
Blower_1_status	DEFAULT	ReadWrite	INT16	419254
Blower2_ctrl	DEFAULT	ReadWrite	INT16	400041
Blower1_ctrl	DEFAULT	ReadWrite	INT16	400040
Blower_2_Ctrl_Swi...	DEFAULT	ReadWrite	BIT	01009
Blower_1_Ctrl_Swi...	BOOL	ReadWrite	BIT	01006

KUVA 13. Puhaltimien tagit listattuna.

Puhallin 1 tilan tagin rekisteriosoitteen selvittämisen jälkeen sitä verrataan PLC:llä määritetyn muuttujan osoitteeseen. PLC-ohjelmassa on määritetty melkein jokaiselle käyttöliittymästä löytyvälle objektille ja toiminnolle vastaavat tagit tai muuttujat. Kuvassa 14 on nähtävissä profiilisivulla käytettyjen tagien muuttujat. Kuten siitä voidaan havaita, jokaiselle objektille on olemassa kaksi eri muuttujaa. Pienempi luku on PLC:n sisäinen muuttuja, jota käytetään ohjelman suorittamisen aikana sekä vanhan käyttöliittymän kanssa kommunikoitaessa. 19000-alueella sijaitseva luku on väylämuuttuja, joka on lisätty PLC-ohjelmaan jälkeenpäin. Väylämuuttujia käytetään tässä työssä samassa näkymässä sijaitsevien tagien rekisteriosoitteiden alueiden yhtenäistämiseen. Kun väylä päivittyy, se tapahtuu järjestyksessä pienimmästä rekisteristä isoimpaan rekisteriin. On hyvä, että samassa näkymässä näkyvät objektit päivittyisivät yhtäaikaisesti. Siksi näkymässä olevat objektit pyritään sitomaan toisiaan lähellä sijaitseviin rekisteriosoitteisiin. Muuttujassa oleva kirjain M tarkoittaa, että se on sisäinen muuttuja (internal variable). Kirjain W taas kertoo, että muuttujan formaatti on sana (word) joten sen koko on kaksi tavua (byte) eli 16 bittiä. Kuten kuvasta 14 huomataan, puhaltimelle on määritetty muuttujat %MW9021 sekä %MW19254. Muuttuja %MW9021 vastaa vanhassa käyttöliittymässä olevaa 9021 rekisteriosoitetta. Tästä saadaan varmistus, että käsiteltävä muuttuja on oikea. Toinen muuttuja %MW19254 on nelosta lukuun ottamatta sama kuin uudessa käyttöliittymässä

oleva rekisteriosoite 419254. Tässä tapauksessa etumerkkinä toimivaa nelosta ei tarvita, sillä PLC ottaa asian huomioon väylälle kirjoittaessaan, ja muuttaa muuttujat osoiterekisterimuotoon. Nyt kun rekisteriosoitteet vastaavat toisiaan, voidaan olettaa, että kommunikaatiorajapinta käyttöliittymän ja PLC:n välillä on kunnossa.

```
(* Profile page *)
%MW19250:=%MW9012;(* Heating Left *)
%MW19251:=%MW9013;(* Cooling Left *)
%MW19252:=%MW9018;(* Drivestatus *)
%MW19253:=%MW9020;(* Heatingstatus *)
%MW19254:=%MW9021;(* Blower_1_status *)
%MW19255:=%MW9022;(* Blower_2_Status *)
%MW19256:=%MW9031;(* GlassInLd *)
%MW19257:=%MW9033;(* GlassInFce *)
%MW19258:=%MW9035;(* GlassInChl *)
%MW19259:=%MW9037;(* GlassInUnLD *)
%MW19260:=%MW9048;(* Turbo_Top1_AirTemp *)
```

KUVA 14. Näkymä PLC:stä, jossa on määritelty Profiilisivulla käytettävien objektien väylämuuttujat.

Kuten aiemmassa kappaleessa mainittiin, käyttöliittymän ja PLC:n välinen kommunikaatiorajapinta puhallin 1 tilalle on kunnossa. Kun rekisteriosoite ja muuttujan tunnus eivät vastaa toisiaan, on joko PLC-ohjelmaan tai käyttöliittymään tehtävä muutoksia. Vaihtoehtona on joko muuttaa PLC:n muuttujien osoitteita, niin että ne vastaavat käyttöliittymän rekisteriosoitteita tai toisinpäin. Tässä on kuitenkin huomioitava, että samaisia osoitteita esiintyy ja on käytössä useammassa eri paikassa niin PLC:n kuin käyttöliittymän sisällä. Tämän takia muutokset on huomattava tehdä jokaiseen paikkaan, jotta ohjelma toimii kuten aiemminkin. Yleensä helpointa on muuttaa käyttöliittymässä olevan objektin rekisteriosoitteet vastaamaan PLC:ssä olevia osoitteita, sillä PLC:ssä on huomattavasti suurempi määrä viittauksia yksittäisiin rekisteriosoitteisiin kuin käyttöliittymässä ja näin ollen voidaan pienentää virheen mahdollisuutta. Kuitenkin jokaista tapausta oli tutkittava erikseen ja tehtävä päätökset tilannekohtaisesti.

Tämä samainen muutosprosessi oli tehtävä jokaiselle käyttöliittymässä olevalle objektille. Kaikki objektit eivät tietenkään olleet samanlaisia, sillä jotkut objekteista olivat lukumuotoisia tulosteita tai toimivat syötteenä, eli ohjasivat tuotantolinjaa. Silloin objektin ja tagin välinen linkki oli erilainen. Kuitenkin perusperiaatteeltaan Modbus-rajapinnan tekeminen tapahtui kaikille objekteille aiemmin kuvailulla tavalla.

4.3 Toiminnallisuuksien siirto PLC:stä käyttöliittymään

Käyttöliittymäpäivityksen mukana asiakkaalle toimitettiin uusi robusti ja tehokas teollisuus-PC. Uuden tietokoneen myötä käyttöliittymällä on käytössään enemmän resursseja ja laskentatehoa. Tästä syystä päätettiin, että osa PLC:ssä sijaitsevista toiminnallisuuksista voidaan siirtää suoraan käyttöliittymään turhan väyläliikenteen karsimiseksi sekä kuorman siirtämiseksi PLC:ltä käyttöliittymään.

PLC:stä siirrettiin monia toiminnallisuuksia käyttöliittymälle. Toiminnallisuuksia ei kuitenkaan voinut suoraan kopioida, sillä PLC:n ohjelmointiohjelmassa PL7:ssä käytettään IEC 61131-3 standardin Structured text -ohjelmointikieltä, kun taas käyttöliittymän suunnittelutyökalussa iX Developerissa käytetään C# -ohjelmointikieltä. Jokaisen PLC:ssä olevan toiminnallisuuden toimintaperiaate oli erikseen selvitettävä, jonka jälkeen se piti koodata vastaavanlaiseksi käyttöliittymälle. Kuvassa 15 esimerkit molemmista ohjelmointikielistä.

```
186
187 %L100:          (* unld drive status *)
188
189     IF NOT %I600.48 THEN
190
191         %MW7901:=0;         (* stp *)
192
193     ELSE IF %Q601.20 THEN
194
195         %MW7901:=1;         (* run *)
196
197     ELSE
198
199         %MW7901:=3;         (* rdy *)
200
201     END_IF;
```

```
219
220 //loop files
221 foreach ( string s in files )
222 {
223     System.IO.FileInfo fi = null;// Create null FileInfo
224     try
225     {
226         fi = new System.IO.FileInfo(s); // current fileinfo
227     }
228     catch ( System.IO.FileNotFoundException e )//not found
229     {
230                     Console.WriteLine(e.Message);
231         continue;
232     }
233     //file exists
234     if( fi.Exists )
235     {
236             string fileName = fi.Name;
```

KUVA 15. IEC 61131-3 standardin Structured text-ohjelmointikieli sekä C# -ohjelmointikieli

Esimerkiksi tähän on valittu esiteltäväksi lasinkarkaisun karkaisuarvojen laskennan siirto. Lasin lämpökarkaisussa lasin paksuudella on suuri vaikutus siihen, miten lasia pitää karkaista. Lasin paksuus vaikuttaa esimerkiksi karkaisulämpötilaan, karkaisuaikaan, jäähdytyspaineeseen sekä jäähdytysaikaan. Siksi eri lasivahvuuksille on hyvä olla valmiit reseptit, jotka antava näille muuttujille sopivat alkuarvot. Käyttäjä voi halutessaan vielä muuttaa tiettyä osaa näistä arvoista tarpeen vaatiessa tai tallentaa omat arvonsa reseptiin, jos hän on huomannut jonkin tietyn arvon vaikuttavan positiivisesti lopputulokseen.

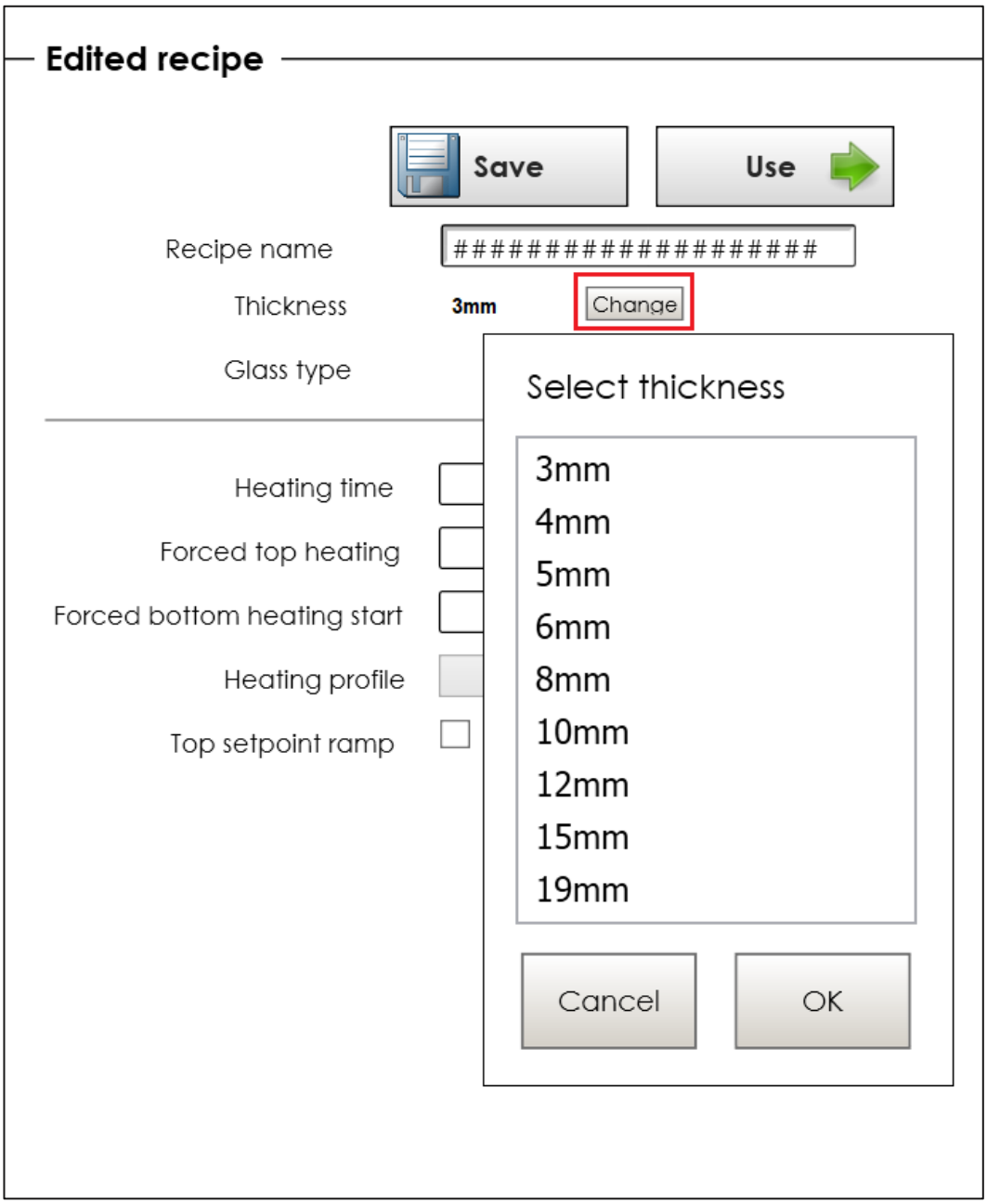
PLC-ohjelmassa oli olemassa koodi, jossa se lukee käyttäjän syöttämän lasivahvuuden ja sen mukaan antaa oikeat alkuarvot karkaisuajalle %MW16045, jäähdytysajalle %MW16046 sekä jäähdytyspaineen nousulle %MW16047. Kuvassa 16 on koodi esitettynä. Muuttuja %MW16022 on varattu lasin paksuudelle. Koodissa yksinkertaisesti verrataan täyttääkö muuttujaan tallennettu lasin paksuuden arvo lausekkeen vaatiman ehdon. Esimerkiksi kuvassa 16 ylimmäisenä näkyvä lauseke ”IF %MW16022<4 THEN” tarkistaa onko lasin paksuuden muuttujaan %MW16022 varastoitu arvo pienempi kuin 4. JOS (IF) lasin paksuus %MW16022 on pienempi kuin 4, SITTEN (THEN) kirjoitetaan reseptien parametreille varattuihin muuttujiin %MW16045, %MW16046 sekä %MW16047 vakiomuisteihin %KW81, %KW91 sekä %KW101 tallennetut arvot. Tämän jälkeen ohjelma hyppää koodissa eteenpäin (JUMP %L120;) ja ohittaa muut vertailut. Jos IF-lausekkeen ehto ei toteudu, hypätään ohjelmassa seuraavaan IF-lauseeseen, jolloin lasin paksuuden muuttujaa %MW16022 verrataan seuraavaan arvoon. Tätä jatketaan niin kauan, kunnes lasinpaksuuden muuttujan arvo toteuttaa yhtälön. Tällä tavoin saadaan jokaiselle lasin vahvuudelle oikeat karkaisuarvot.

```
IF %MW16022<4 THEN          (* s<4mm *)

    %MW16045:=%KW81;
    %MW16046:=%KW91;
    %MW16047:=%KW101;

    JUMP %L120;
    END_IF;


IF %MW16022=4 THEN          (* s=4 *)

    %MW16045:=%KW82;
    %MW16046:=%KW92;
    %MW16047:=%KW102;

    JUMP %L120;
    END_IF;


IF %MW16022=5 THEN          (* s=5mm *)

    %MW16045:=%KW83;
    %MW16046:=%KW93;
    %MW16047:=%KW103;

    JUMP %L120;
    END_IF;


IF %MW16022<8 THEN          (* s=6..7mm *)

    %MW16045:=%KW84;
    %MW16046:=%KW94;
    %MW16047:=%KW104;

    JUMP %L120;
    END_IF;
```

KUVA 16. PLC-koodi oikeiden alkuarvojen asettamiseksi lasin paksuuden mukaan.

Vakioarvot on erikseen määritelty PLC:ssä, ja jokaiselle lasivahvuudelle on olemassa omat varastoidut vakioarvot. Esimerkiksi jos lasin paksuus on 5mm, luettavat vakiomuistit ovat %KW83, %KW93 ja %KW103. Kirjain K tarkoittaa, että kyseessä on sisäinen vakio (internal constant), jota ei pysty ylikirjoittamaan PLC-ohjelmassa suorituksen aikana vaan ainoastaan lukemaan. Näitä vakioarvoja pystyy muuttamaan PL7:n kautta PLC-ohjelman ollessa pois päältä. Tämä karkaisuarvojen asettamisen toiminnallisuus oli siirrettävä PLC:stä käyttöliittymään.

Käyttöliittymässä on reseptinäkymässä painike, josta avautuu ikkuna lasin paksuuden valinnalle (kuva 17). Painike on kuvassa korostettuna punaisella laatikolla. Kuten kuvasta nähdään, lasin paksuudelle on olemassa yhdeksän eri vaihtoehtoa. Jokaisella näistä vaihtoehdoista on olemassa omat arvonsa karkaisuajalle, jäähdytysajalle ja jäähdytyspaineen nousulle. Tästä näkymästä valitaan lasin paksuus ja hyväksytään se painamalla OK-painiketta.



KUVA 17. Käyttöliittymän lasin paksuuden valintaikkuna

Jotta lasin paksuus saadaan vaikuttamaan reseptin arvoihin ja näkymään resepti-ikkunassa, oli valintaikkunaan lisättävä toiminto, joka aktivoi haluttujen reseptiarvojen tarkastuksen. Kuvassa 18 on nähtävissä valmis valintaikkunan koodi. Siitä nähdään mitä tapahtuu, kun lasin paksuus on valittu ja painetaan OK-painiketta. Kuvassa 18 punaisella alleviivattu ”btnConfirmThickness_Click” viittaa OK-painikkeen painamiseen. OK-painikkeen painaminen saa aikaan alempana sinisellä laatikoidun osuuden suorittamisen:

- Ensimmäisenä käyttöliittymän sisäinen muuttuja ”_RIE_Thickness” ylikirjoitetaan valitulla lasin paksuuden arvolla.
- Toisena kutsutaan ScriptModuleRecipe-moduulista löytyvää CheckQuenchValues-metodia.
- Lopuksi valintaikkuna suljetaan.

Toisena oleva CheckQuenchValues-metodin kutsu oli uusi lisäys valintaikkunan koodiin. Tämä mahdollistaa ScriptModuleRecipe-moduulissa olevan koodin suorittamisen ennen valintaikkunan sulkemista.

```
18
19      public partial class popup_SelectThickness
20      {
21
22          void btnConfirmThickness_Click(System.Object sender, System.EventArgs e)
23          {
24              Globals.Tags._RIE_Thickness.SetAnalog(Globals.Tags._RIE_Thickness_Selection.Value);
25              ScriptModuleRecipe.CheckQuenchValues();
26              this.Close();
27          }
28      }
29  }
30
```

KUVA 18. Käyttöliittymän lasin paksuuden valintaikkunan koodi

Seuraavaksi oli luotava ScriptModuleRecipe-moduuliin karkaisuarvojen tarkistamista varten CheckQuenchValues-metodi, jota lasin paksuuden valintaikkunan OK-painikkeen painaminen kutsuu. Koodin oli tehtävä samat asiat, kuin mitä PLC:ssä oleva koodi oli tehnyt. Kuvassa 19 on nähtävissä valmis koodi CheckQuenchValues-metodille. Ensimmäisenä koodissa tarkastetaan, että onko lasin tyyppi erisuuri kuin 1. Se tapahtuu IF-lausekkeella, joka on kuvassa alleviivattu punaisella. Tässä tapauksessa lasin tyyppi 1 tarkoittaa lämpölujitettua lasia, jolle on olemassa omat karkaisuarvonsa. Jos lasin tyyppi on jotain

muuta kuin lämpölujitettua lasia, ohjelma jatkaa eteenpäin switch-lausekkeeseen, joka on alleviivattu orassilla kuvassa 19. Switch-lauseke lukee lasin valintaikkunassa kirjoitutetun ”_RIE_Thickness” -muuttujan arvon ja sen perusteella valitsee sen casen, joka vastaa lasin paksuutta. Sen jälkeen lasin karkaisuarvojen muuttujat ”_RIE_QuenchTime” (karkaisuaika), ”_RIE_CoolingTime” (jäähdytysaika) ja ”_RIE_CoolingPressureRise” (jäähdytyspaineen nousu) kirjoitetaan vastaamaan casen ja lasivahvuuden vastaavia arvoja. Jos esimerkiksi käyttäjä on valinnut lasinvahvuudeksi 5mm, switch-lauseke valitsee käytettäväksi case 5, joka on kuvassa laatikoitu sinisellä. Näin ollen ohjelma saa karkaisuprosessiin karkaisuajalle arvon 25, jäähdytysajalle arvon 140 ja jäähdytyspaineen nousulle arvon 10.

```
704
705         public static void CheckQuenchValues()
706         {
707             if( Globals.Tags._RIE_GlassTypeSend.Value != 1 )
708             {
709                 switch( (int)Globals.Tags._RIE_Thickness.Value )
710                 {
711                     case 3:
712                         Globals.Tags._RIE_QuenchTime.Value = 1;
713                         Globals.Tags._RIE_CoolingTime.Value = 105;
714                         Globals.Tags._RIE_CoolingPressureRise.Value = 10;
715                         break;
716
717                     case 4:
718                         Globals.Tags._RIE_QuenchTime.Value = 1;
719                         Globals.Tags._RIE_CoolingTime.Value = 100;
720                         Globals.Tags._RIE_CoolingPressureRise.Value = 10;
721                         break;
722
723                     case 5:
724                         Globals.Tags._RIE_QuenchTime.Value = 25;
725                         Globals.Tags._RIE_CoolingTime.Value = 140;
726                         Globals.Tags._RIE_CoolingPressureRise.Value = 10;
727                         break;
728
729                     case 6:
730                         Globals.Tags._RIE_QuenchTime.Value = 30;
731                         Globals.Tags._RIE_CoolingTime.Value = 140;
732                         Globals.Tags._RIE_CoolingPressureRise.Value = 10;
733                         break;
734
735                     case 8:
736                         Globals.Tags._RIE_QuenchTime.Value = 45;
737                         Globals.Tags._RIE_CoolingTime.Value = 280;
738                         Globals.Tags._RIE_CoolingPressureRise.Value = 10;
739                         break;
740
741                     case 10:
```

KUVA 19. Käyttöliittymän lasin karkaisuarvojen tarkistamista varten luotu koodi.

Nyt PLC:ssä aikaisemmin toteutettu lasin karkaisuarvojen tarkastaminen oli siirretty käyttöliittymään. Tästä lähtien, kun käyttäjä valitsee karkaistavan lasin pak-

suuden, ScriptModuleRecipe-moduulissa sijaitseva CheckQuenchValues-metodi hoitaa lasin karkaisuarvojen tarkistamisen ja kirjoittamisen käyttöliittymän sisällä. Näin ollen PLC:n tarvitse osallistua arvojen tarkastamiseen, vaan se saa valmiiksi oikeat arvot reseptin ajamisen yhteydessä.

Muiden toiminnallisuuksien siirto PLC:stä käyttöliittymään tapahtui perusperiaatteeltaan samalla tavalla kuin esitelty reseptin karkaisuarvojen laskennan siirtäminen. PLC:stä selvitettiin siirrettävän toiminnallisuuden toimintaperiaate, jonka jälkeen se toteutettiin uudestaan käyttöliittymässä C#-koodikielellä. Tämän jälkeen jokainen toiminnallisuus testattiin iX Developerin simulointitilassa koodin toimivuuden varmistamiseksi.

4.4 Käyttöliittymään kustomointi

Uusi käyttöliittymä oli monelta osin valmis kokonaisuus, jossa monet toiminnallisuudet ovat valmiiksi toteutettuja, toimintakuntoisia ja vaativat ainoastaan Modbus-rajapinnan tekemisen. Kaikki lasinkarkaisulinjat eivät kuitenkaan ole samanlaisia ja asiakkaalla voi olla oma selkeä näkemyksensä käyttöliittymän ulkoasun suhteen. Tällöin käyttöliittymään on lisättävä elementtejä ja ominaisuuksia tai muutettava käyttöliittymän ulkoasua asiakkaan toiveiden mukaisesti. Tätä muutostyötä kutsutaan käyttöliittymän kustomoinniksi tai räätälöinniksi.

Käyttöliittymän kustomointi on melkein poikkeuksetta irrotettava käyttöliittymän pääkehityshaarasta, jossa tuotetta kehitetään yleisestä näkökulmasta. Kustomoinnilla saatavat tulokset ovat usein asiakaskohtaisia, ja soveltuvat harvoin tuotteen yleiseen kehityslinjaan. Poikkeuksiakin on kuitenkin olemassa, jolloin asiakkaan toiveet tai tarpeet on huomattu yleishyödylliseksi tuotteen kehityksen ja käytettävyyden kannalta. Tällöin saadut tulokset voidaan implementoida pääkehityshaaran mukaiseen käyttöliittymään.

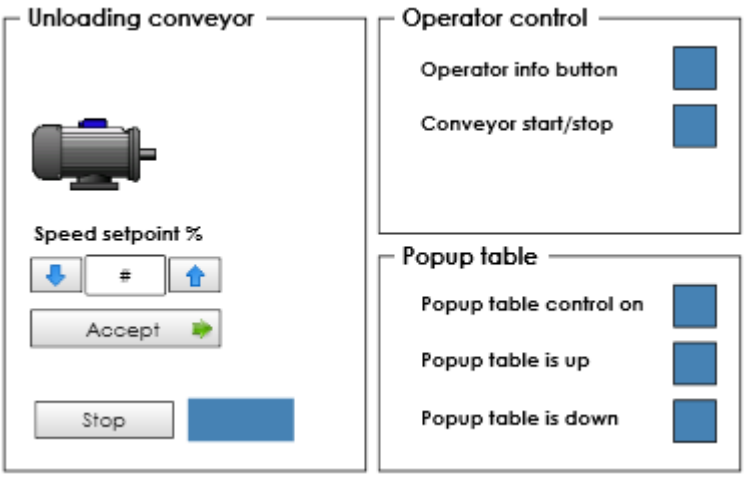
Käyttöliittymän kustomointi alkaa uuden käyttöliittymän puutteiden ja asiakkaan toiveiden kartoittamisella. Tällä saadaan selville mitä pitää toteuttaa, jonka jäl-

keen voidaan tehdä aika-arvio toteutuksen kestosta. Tällöin projekti pysyy aikataulussa. Kun puutteet ja toiveet on kartoitettu sekä aikatauluarvio tehty, voidaan varsinainen muutostyö aloittaa.

4.4.1 Objektien lisääminen käyttöliittymään

Lasinkarkaisulinjat poikkeavat usein toisistaan erilaisten anturointien tai toimilaitteiden suhteen. Joissain tapauksissa asiakkaat ovat itse jälkikäteen asentaneet erilaisia painikkeita tai antureita prosessin ohjausta ja valvontaa varten. Tällöin käyttöliittymään on lisättävä lisätyn laitteen tilasta kertova objekti. Objektien lisääminen pystyttiin helposti toteuttamaan iX Developerilla. Esimerkin vuoksi tähän on valittu tilaindikaattorin lisääminen.

Asiakkaalla on prosessin loppupäähän purkukuljettimelle olemassa jalkakytkin, jolla kuljetinta pystyttiin ohjaamaan manuaalisesti prosessista poiketen. Yleensä karkaisulinjalla työskentelee kaksi työntekijää: Ensimmäinen käyttää prosessia käyttöliittymän kautta, sekä lastaa karkaisuun menossa olevat lasit lastauskuljettimelle. Toinen toimii purkukuljettimen päässä poistaen valmiit lasit kuljettimelta jatkokäsittelyä tai varastointia varten. Tätä varten oli tehtävä käyttöliittymään näkymä, joka kertoo koneen käyttäjälle, milloin hänen näköalueensa ulkopuolella olevaa purkukuljetinta ohjataan manuaalisesti jonkun toisen työntekijän toimesta. Kuvassa 20 on nähtävissä Unloading-näkymä ennen tilaindikaattorin lisäämistä.

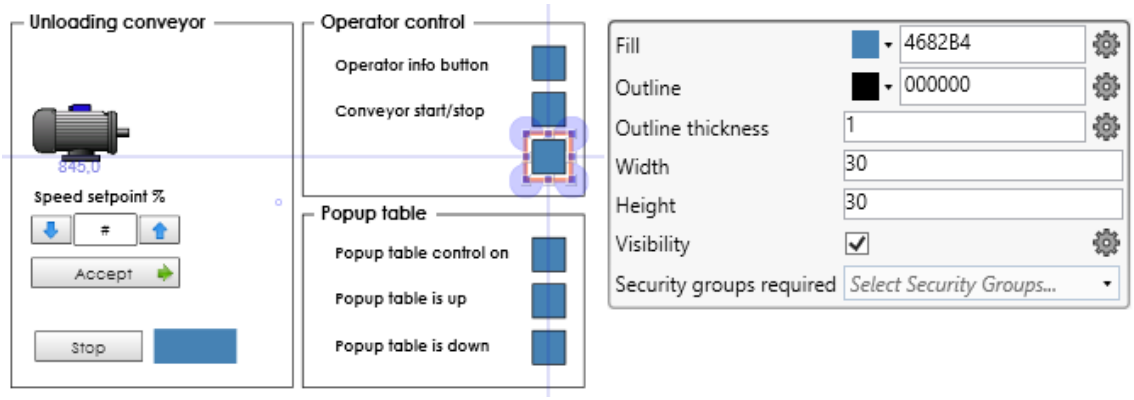


KUVA 20. Unloading-näkymä ennen tilaindikaattorin lisäämistä.

Tilaindikaattorin luominen aloitettiin luomalla sille objekti. Objektin attribuutit, kuten muoto, koko, sekä väritys muutettiin vastaamaan käyttöliittymässä valmiiksi olevia indikaattoreita – näin ollen käyttöliittymän yhdenmukaisuus (luku 2.4.2) ja käyttäjäystävällisyys (luku 2.4.3) säilytetään entisellään. Tämä on tärkeää hyvän käyttökokemuksen kannalta. Muutetut attribuutit olivat:

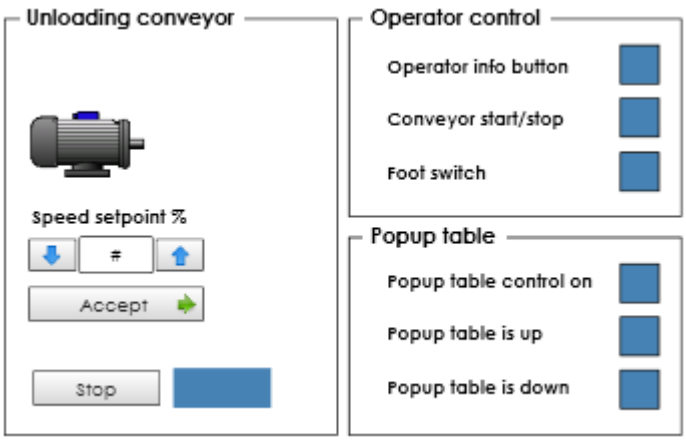
- Fill, jolla määritettiin objektin oletuksena oleva täyttöväri.
- Outline, jolla määritettiin objektin ääriviivan väri.
- Outline thickness, jolla määritettiin objektin ääriviivan paksuus.
- Width ja Height, jolla määritettiin indikaattorin leveys ja korkeus.
- Visibility, joka määrittää onko indikaattori näkyvissä ohjelman aikana.

Attribuutit ovat nähtävissä objektin valitsemisen vuoksi ilmestyvässä ominaisuusikkunassa. Attribuuteille, joiden perässä on nähtävissä harmaa hammasratas, pystyttiin tekemään lisämäärityksiä. Esimerkiksi objektin väristä tai näkyvyydestä on mahdollista tehdä dynaamisia, eli ne voitiin määrittää muuttuviksi eri tapahtumien vaikutuksesta. Tämä on olennaista esimerkiksi tulosteobjektien, kuten tilaindikaattorin kannalta. Kuvassa 21 nähtävissä lisätty tilaindikaattoriobjekti, sekä objektin ominaisuusikkuna, jossa objektin ulkonäköön vaikuttavat attribuutit ovat.



KUVA 21. Lisätty tilaindikaattori sekä sen ominaisuusikkuna.

Seuraavaksi käyttöliittymään oli luotava indikaattoria kuvaava teksti. Jalkakytkimen tilaindikaattorin tekstiksi valikoitui hyvin kuvaava ja selkeä englanninkielinen teksti ”Foot switch”. Yhdenmukaisuuden vuoksi tekstin fontille, fontin koolle, sekä värille valittiin samat arvot kuin muilla indikaattoreita kuvaavilla teksteillä oli. Kuvassa 22 näkyy lisätty teksti.



KUVA 22. Tilaindikaattorille lisätty teksti ”Foot switch”

Objektin luomisen jälkeen se oli konfiguroitava vastaamaan jalkakytkimen painamista. Tätä varten oli PLC:stä selvitettävä mille muuttujalle jalkakytkin oli asetettu. Kuvasta 23 on nähtävissä, että jalkakytkin on määritetty sanamuotoisen muuttujan %MW7951 bitille numero 12. Kyseinen muuttuja on jaettu useampaan osaan useiden tilatietojen varastointia ja siirtämistä varten. Koska sanamuotoisessa muuttujassa on 16 bittiä, se voidaan jakaa 16 eri osaan, jossa jokaisen bitin tila voi olla 0 tai 1 (OFF tai ON). Muuttujan %MW7951 bittiä 12 ohjaa PLC:n fyysinen sisääntulo %I600.52, johon jalkakytkin on kytketty. Eli jalkakytkimen painamisesta aiheutuva signaali siirtyy jalkakytkimeltä johdinta pitkin PLC:n sisääntuloon %I600.52 josta se siirtyy muuttujan %MW7951 bittiin 12.

```
79
80      %MW7951:X9:=%I600.57;       (* tbl ctrl *)
81      %MW7951:X10:=%I600.58;      (* tbl dwn *)
82      %MW7951:X11:=%I600.59;      (* tbl up *)
83      %MW7951:X12:=%I600.52;      (* Foot switch *)
84
```

KUVA 23. Jalkakytkimen muuttuja %MW7951:X12

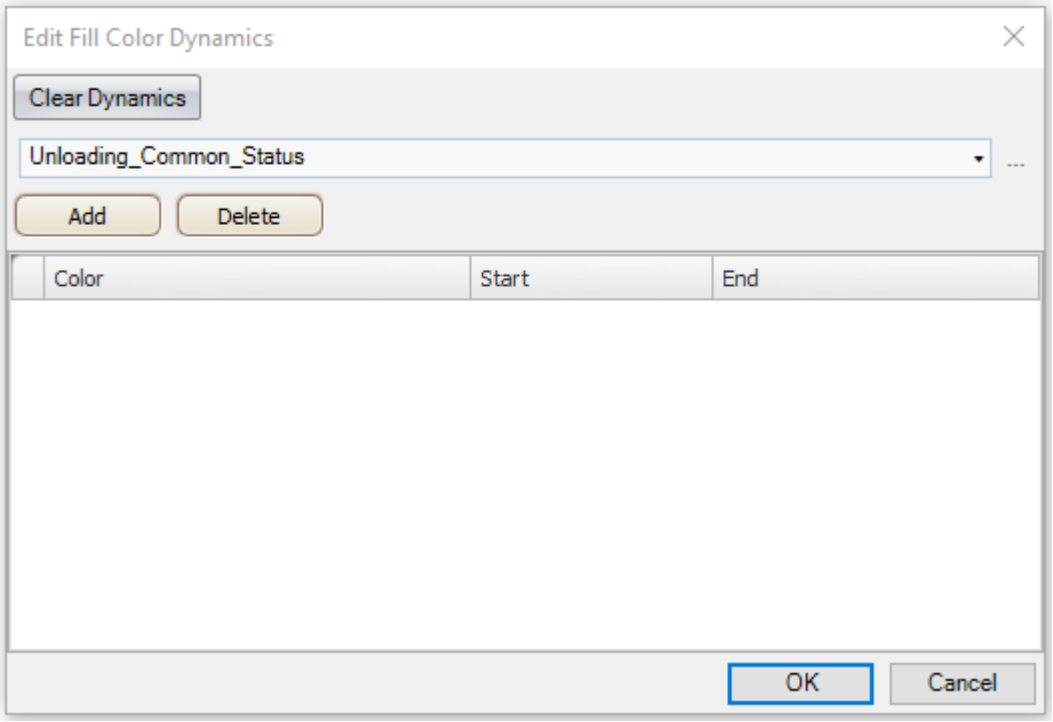
Nyt kun on tiedossa missä muuttujassa jalkakytkimen tilatieto sijaitsee, voidaan vastaava rekisteriosoite ottaa käyttöön käyttöliittymän puolelta. Koska kyseinen rekisteriosoite on ollut jo valmiiksi käytössä muuta tiedonsiirtoa varten, sitä ei tarvitse erikseen luoda. Rekisteriosoitteen luominen on kuitenkin itsessään todella helppo ja suoraviivainen operaatio: iX Developerin Tag-listaan lisätään uusi tagi, jolle syötetään sopiva nimi, tietotyyppi, sekä haluttu osoite. Nyt kuitenkin riitti, että

kyseisen rekisteriosoitteen bitti linkataan aiemmin luotuun tilaindikaattoriin. Tag-listasta selvitettiin, että rekisteriosoite on varattuna ”Unloading_Common_Status” -tagille. Kuvassa 24 nähtävissä kyseinen tagi sekä rekisteriosoite 407951.

Tag			Controllers	
Name	Data Type	Access Right	Data Type	Controller2
Unloading_Common_Status	INT16	ReadWrite	INT16	407951
Unloading_Conv_Speed_Sp	INT16	ReadWrite	INT16	407983
Unloading_Conv_Status	DEFAULT	ReadWrite	INT16	407901
TurboOffsetTopUnloading	DEFAULT	ReadWrite	INT16	409072
TurboOffsetBtmUnLoading	DEFAULT	ReadWrite	INT16	409073

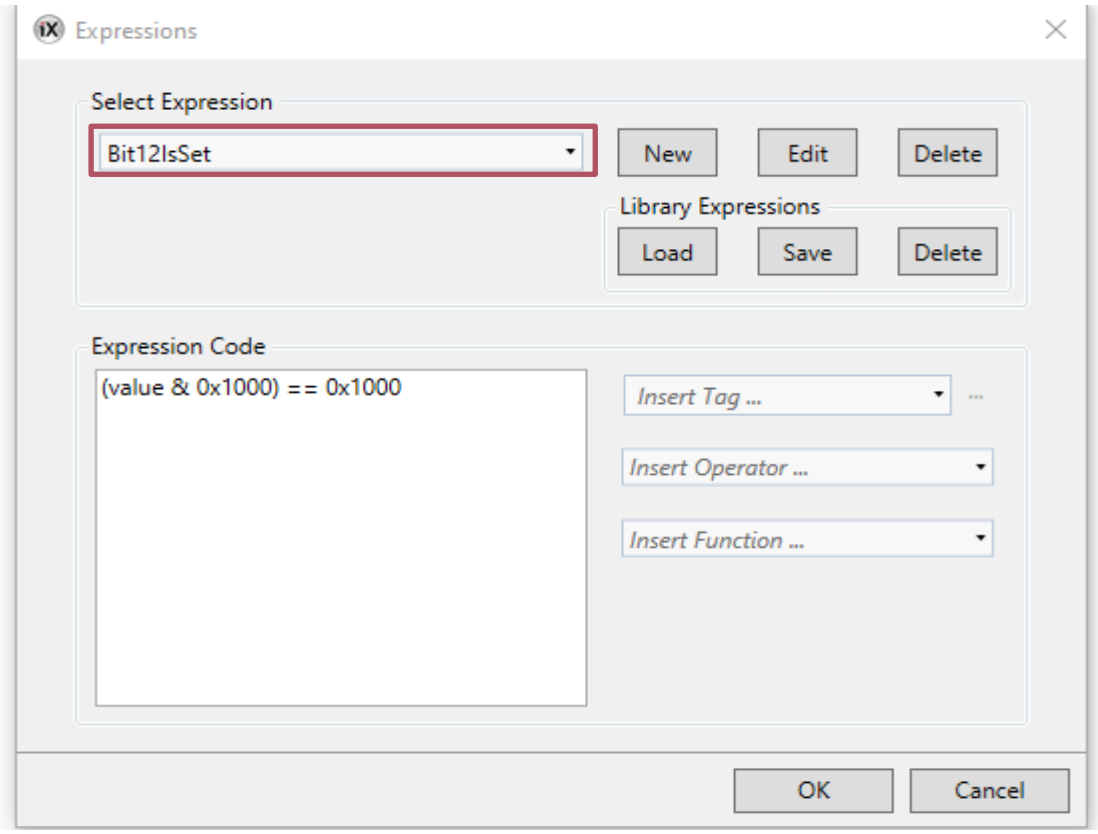
KUVA 24. Tag-lista, josta nähtävissä jalkakytkimen tilatiedon tagi ”Unloading_Common_Status”

Tilaindikaattorin toiminta perustuu objektin värin muuttumiseen. Tämän vuoksi rekisteriosoite oli linkattava objektin väritäyttöön. Se tapahtui objektin väritäytön määrityksistä. Kuten aikaisemmin mainittiin, joillain objektin attribuuteilla oli olemassa ominaisuusikkunassa hammasratas lisämäärityksiä varten. Hammasrattaan takaa löytyy objektin väritäytön dynamiikan määrittäminen (kuva 25). Sieltä piti valita oikea tagi, johon jalkakytkimen rekisteriosoite on sidottu. Tässä tapauksessa tuo tagi oli ”Unloading_Common_Status”.



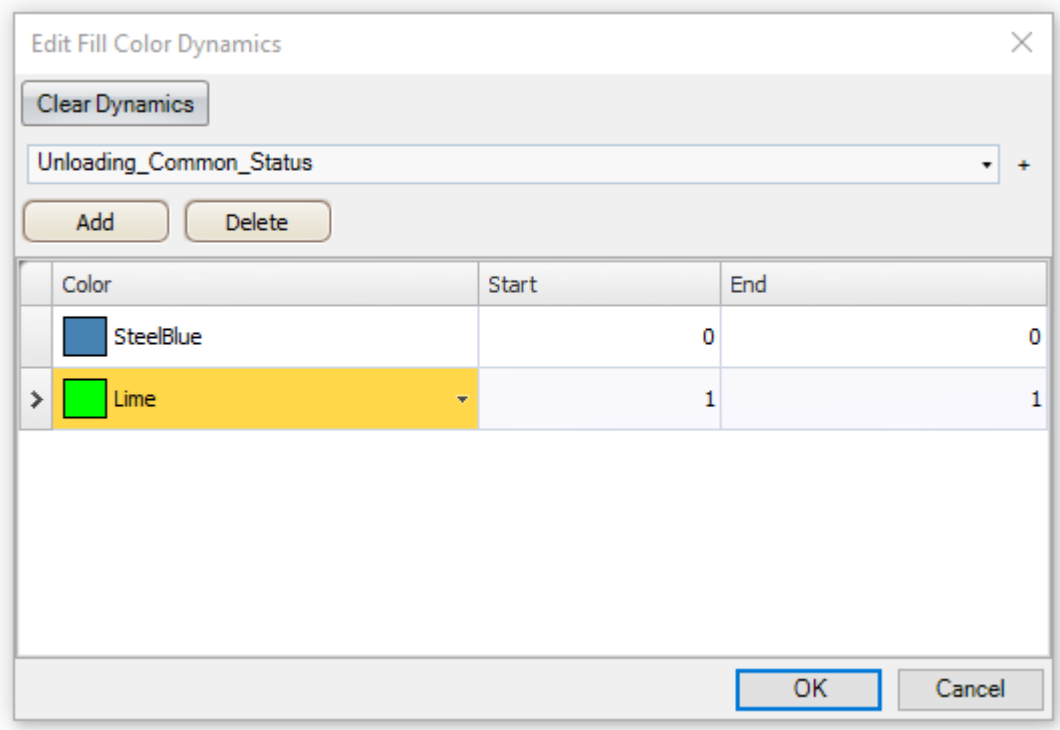
KUVA 25. Objektin väritäytön dynamiikan määritysikkuna

Seuraavaksi valittiin mitä bittiä valitusta tagin rekisteriosoitteesta halutaan käyttää. Se tapahtui samasta ikkunasta tagin perässä olevaa kolmea pistettä painamalla. Tästä avautui uusi ikkuna (kuva 26), josta pystyttiin tekemään lisämääritysiä tagin vaikutuksesta väritäytön dynamiikkaan. Ikkunassa olevasta vetovalikosta valittiin bitti 12 määrittäväksi tekijäksi, jonka jälkeen painettiin OK-painiketta ja palattiin takaisin väritäytön dynamiikan määrittämisikkunaan. Kuvassa 26 punaisella laatikoituna vetovalikko, sekä valittu määrittävä tekijä bitti 12.



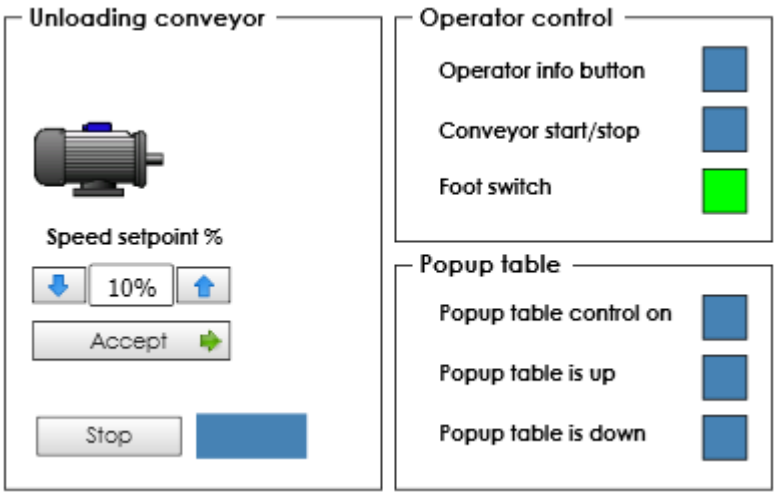
KUVA 26. Objektin väritäyttöön vaikuttavan tagin lisämääritysikkuna

Lopuksi objektin väritäytön dynamiikkaan oli määritettävä molemmille bitin arvoille omat värinsä. Tämä tapahtui helposti dynamiikan määrittämisikkunassa. Painikkeella ”Add” saatiin lisättyä uusi väri. Ensiksi lisättiin passiivisen tilan väri, joka on koko käyttöliittymässä yhtenäinen passiivisille objekteille: teräksensininen. Värin valinnan jälkeen määritettiin mistä luvusta värin aktivointi alkaa ja loppuu. Näihin molempiin valittiin sama arvo – nolla, sillä jalkakytkimen ollessa passiivisena bitin arvo on 0. Seuraavaksi lisättiin aktiivisen tilan väri, limetti, joka on koko käyttöliittymässä sama. Jälleen määritettiin mistä luvusta värin aktivointi alkaa ja loppuu – tällä kertaa arvoksi tuli yksi, koska jalkakytkintä painettaessa bitin arvo on 1. Kuvassa 27 näkyvissä määritetyt värit sekä niiden arvot.



KUVA 27. Objektin väritäytön dynamiikkaan määritetyt värit ja niiden arvot.

Nyt tilaindikaattoriobjektin lisääminen oli valmis. Objektin toimivuus varmistettiin testaamalla se. Testaus tapahtui asettamalla käyttöliittymä testitilaan ja pakottamalla PLC:stä muuttujan %MW7951 bitti 12 aktiiviseksi. Indikaattori muuttui näkymässä vihreäksi, joten näin ollen objektin lisääminen onnistui. Kuvassa 28 nähtävissä testitilanteessa toimivasta jalkakytkimen indikaattorista.

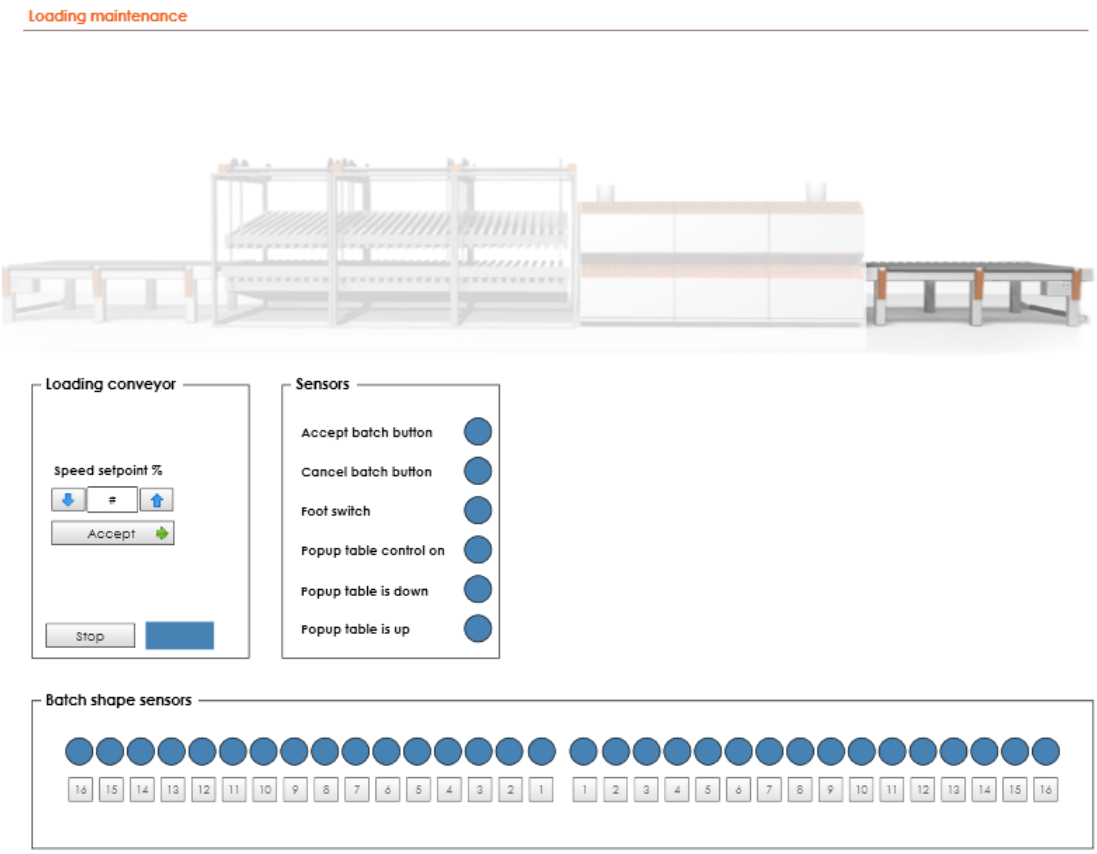


KUVA 28. Tilaindikaattori toimivana testitilanteessa

4.4.2 Objektien poistaminen käyttöliittymästä

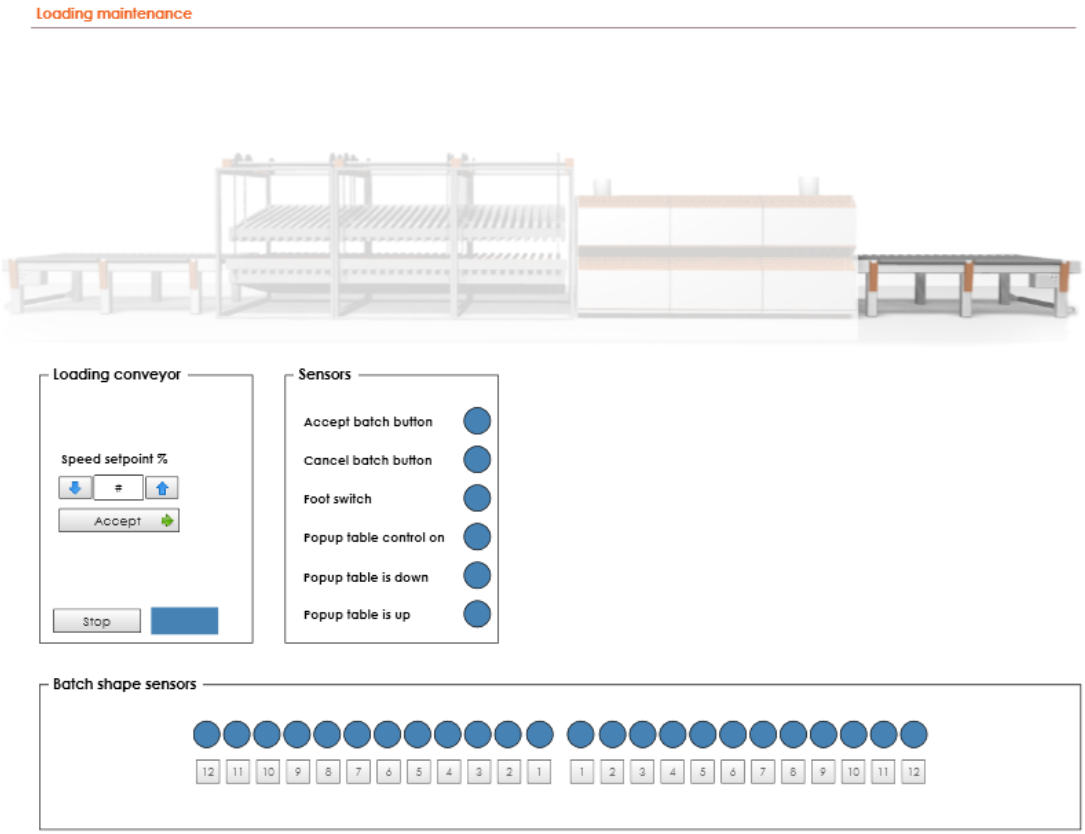
Joissain tapauksissa voi olla niin, että käyttöliittymässä on olemassa ylimääräisiä objekteja prosessista puuttuvien toimilaitteiden takia. Tai sitten kyseessä on muuten ylimääräistä tai turhaa informaatiota käyttäjälle, joka huonontaa näkymän selkeyttä (luku 2.4.4). Tällöin kyseiset objektit voidaan joko poistaa, tai piilottaa käyttäjältä. Piilottamisen etuna on se, että jos puuttuvat toimilaitteet tullaan joskus lisäämään prosessiin, piilotetut objektit on helppo ottaa käyttöön.

Asiakkaan tuotantolinja erosi monissa kohtaa uuden käyttöliittymän objektien oletuskokoonpanosta. Käyttöliittymästä oli piilotettava ja poistettava useita objekteja, jotta näkymät saatiin vastaamaan asiakkaan tuotantolinjaa. Esimerkkinä tähän on otettu lastauskuljettimen tunnistusanturin näkymä. Lastauskuljettimen tunnistusantureita käytetään tunnistamaan karkaisuun menevän lasin leveys. Kuvassa 29 lastauskuljettimen näkymä, jossa alareunassa nähtävissä tunnistusanturien indikaattorit.



KUVA 29. Alkuperäinen lastauskuljettimen näkymä

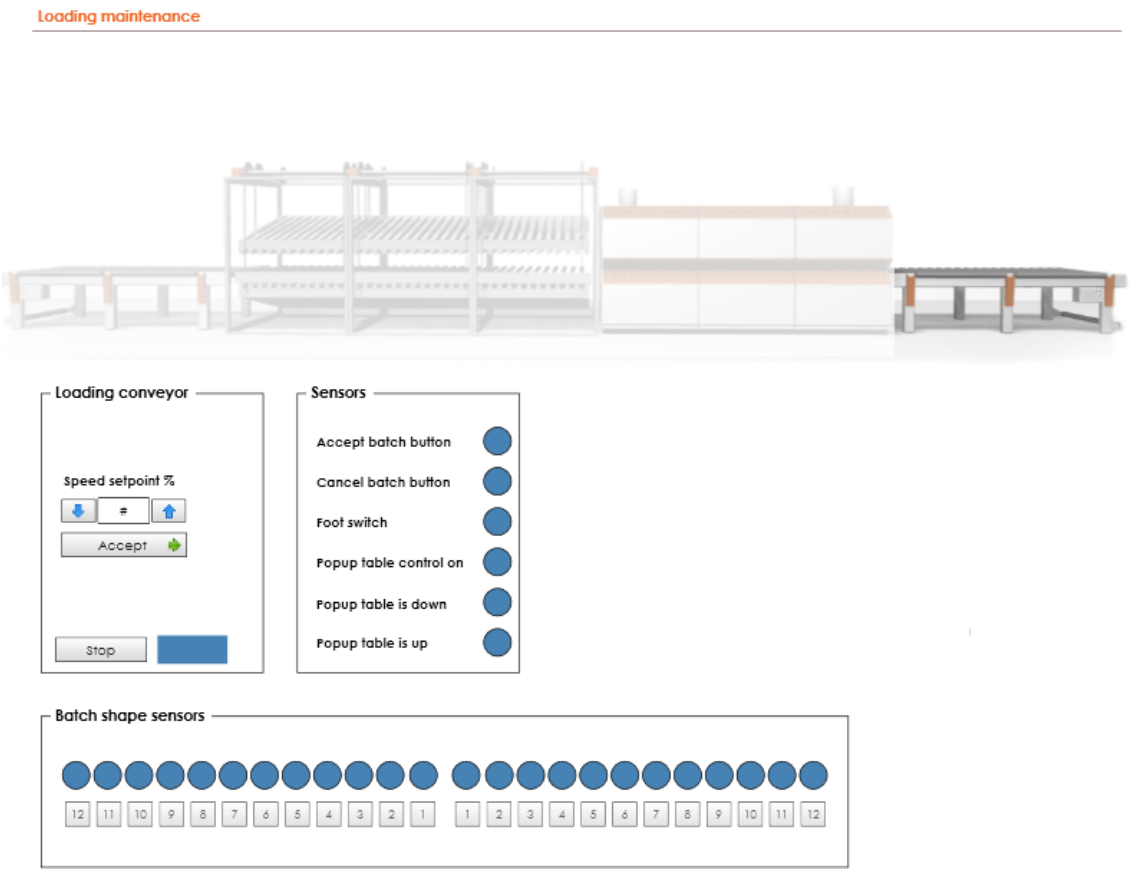
Uudessa käyttöliittymässä on oletuksena näkyvissä 32 anturia; 16 vasemmalla ja 16 oikealla puolella. Asiakkaalla oli kuitenkin käytössään ainoastaan 12 + 12 anturia. Tämän vuoksi käyttöliittymästä oli poistettava ylimääräiset. Tässä tapauksessa anturien indikaattorit piilotettiin, sillä jos asiakas joskus haluaa lisätä antureita lastauskuljettimelle, ne olisi helppo tuoda takaisin näkyviin. Piilottamiseen oli olemassa monta tapaa. Tässä käytettiin tapaa, jossa ylimääräiset objektit valitaan aktiiviseksi, jonka jälkeen ne piilotetaan poistamalla valinta ”Visibility” -sarakkeesta objektin ominaisuusikkunasta. Piilottamista ei kuitenkaan suositella käytettäväksi tapauksissa, jossa piilotettavat objektit ovat syötteitä, kuten esimerkiksi painikkeita tai kytkimiä. Tästä voi joissain tapauksissa seurata syötteen tahaton aktivoituminen. Kuvassa 30 nähtävissä lastauskuljettimen näkymä anturien piilottamisen jälkeen.



KUVA 30. Lastauskuljettimen näkymä anturien piilottamisen jälkeen.

Lastausnäkymässä olevia antureita rajaava laatikko oli niiden piilottamisen takia jäänyt tarpeettoman isoksi. Tämä ei ole visuaalisesti paras ratkaisu, joten laatikon kokoa, sekä anturien sijainta oli muutettava. Käyttöliittymän visuaalinen ulkoasu on todella tärkeää käyttömukavuuden ja helppokäyttöisyyden kannalta, kuten

teoriaosuudessa luvussa 2.3 on kerrottu. Uudelleenjärjestelyn jälkeen, näkymä oli valmis tunnistusantureiden piilottamisen osalta. Kuvassa 31 näkyvissä lastauskuljettimen näkymä valmiina.

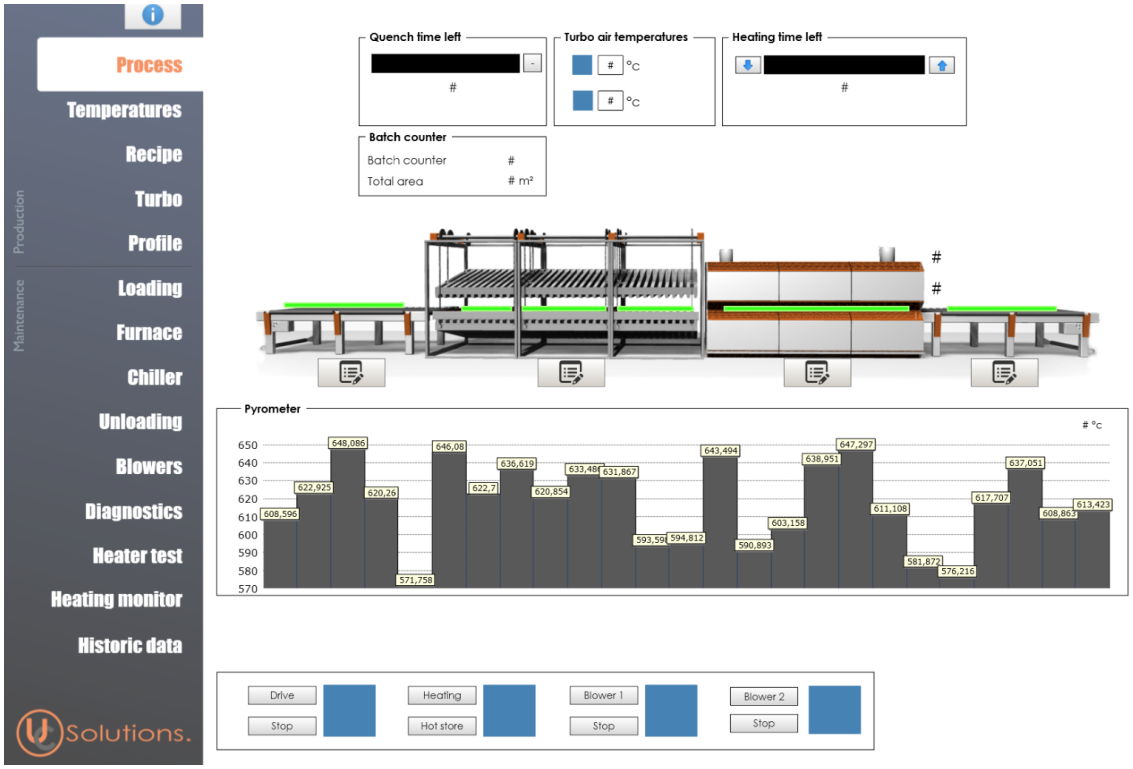


KUVA 31. Lastauskuljettimen näkymä anturien ja niitä rajaavan laatikon uudelleenjärjestelyn jälkeen.

4.4.3 Näkymien visuaalinen parantaminen

Käyttöliittymäsuunnittelussa sekä niiden toteuttamisessa tärkeää on näkymien ulkonäkö, kuten jo aikaisemmin on mainittu. Tämän takia käyttöliittymän visuaalisuutta on pyrittävä parantamaan. Se tuo selkeyttä (luku 2.4.4) näkymiin, mikä parantaa käytettävyyttä sekä helpottaa asioiden hahmottamista. Tässäkin projektissa pyrittiin käyttöliittymän ulkoasua parantamaa mahdollisimman paljon, vaikka se ei ollutkaan projektin ensisijainen tavoite.

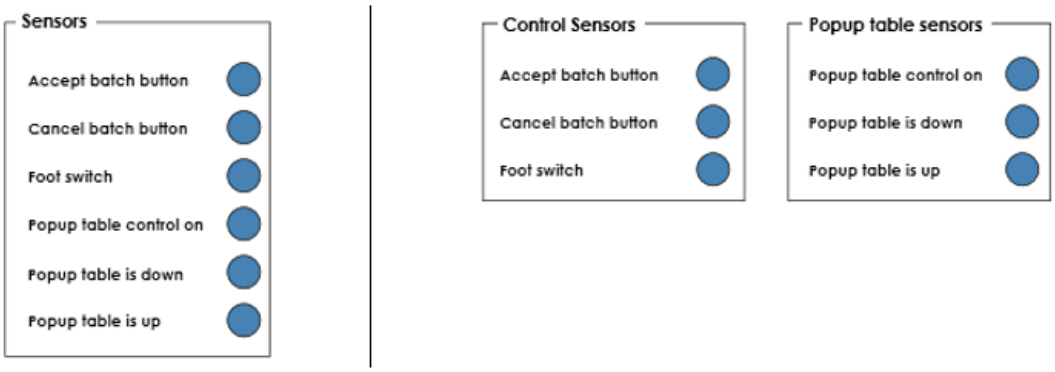
Käyttöliittymän visuaalisuutta voidaan parantaa monin tavoin, kuten luvussa 2 olevassa UX-suunnittelusta kertovassa osiossa käytiin läpi. Objekteja voidaan siirtää eri näkymiin, jos se parantaa asioiden hahmottamista tai se lisää käyttömukavuutta. Paras tapa on saada yhteen näkymään esille kaikki sellainen data, joka on tarpeen nähdä samanaikaisesti, tai jotka liittyvät tiettyyn järjestelmän osaan. Esimerkiksi Process-näkymään on hyvä saada kaikki prosessin käyttämisen kannalta tärkeä data. Käyttöliittymän selkeyttäminen vähentää käyttäjän kognitiivista kuormaa kuten luvussa 2.4.4 on mainittu. Tietoa ei kuitenkaan saa olla kerralla liikaa esillä, jotta käyttäjän on mahdollista prosessoida sitä tehokkaasti. Kuvassa 32 nähtävissä käyttöliittymän prosessinäkymä.



KUVA 32. Käyttöliittymän Process-näkymässä on esitettynä kaikki tarpeellinen data prosessiin liittyen. Tämä näkymä vastaa kuvan 7 vanhan käyttöliittymän prosessinäkymää.

Objekteja voidaan myös lajitella näkymien sisällä tiettyihin segmentteihin, jotka rajataan laatikoilla, jolloin laitteen eri osiin liittyvä data olisi helpommin erotettavissa toisistaan. Tässäkin tapauksessa kyseessä on UX-suunnittelun selkeys. Tästä syystä käyttöliittymän Loading-näkymässä olevien antureiden indikaattoriobjektit on rajattu laatikolla. Tällöin anturi-indikaattorit ovat helposti ja nopeasti erotettavissa muista näkymässä olevista objekteista. Kuvassa 33 on nähtävissä

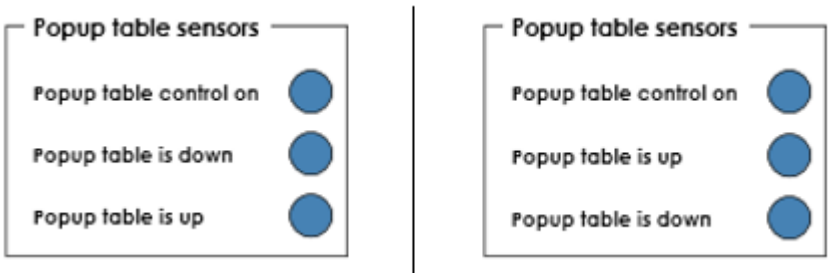
esimerkki, miten objektien rajaamista ja jaottelua voidaan viedä vieläkin pitemmälle. Kuvassa vasemmalla on näkyvissä alkuperäinen toteutus, missä kaikki lastauskuljettimen anturit ovat yhden rajauksen sisällä. Kuvassa oikealla on näkyvissä toteutus, jossa anturit on jaettu kahteen eri osaan; ensimmäisessä ovat purkukuljettimen ohjaukseen liittyvät anturi ja toisessa nostopöydän anturit. Tällä voidaan hakea selkeää linjausta, mihinkä osaan laitetta anturit liittyvät, mikä selkeyttää näkymää ja helpottaa anturien erottamista. Objektien rajaamisessa ja jaottelussa on kuitenkin muistettava olla maltillinen – objektien liiallinen pilkkominen ja jaottelu voi saada aikaan epäselkeän näkymän, joka johtaa sekavaan käyttökokemukseen. Tämä on luvussa 2.3 mainittuja UX-suunnittelun periaatteita vastaan.



KUVA 33. Alkuperäinen, sekä vaihtoehtoinen tapa Loading-näkymän antureiden esittämiselle.

Objektien järjestystä segmenttien sisällä voidaan muuttaa järkevämpään ja luonnollisempaan järjestykseen. Jopa objektien ulkonäköä voidaan tarvittaessa muuttaa, tai niiden kokoa voidaan kasvattaa, jos se helpottaa hahmottamista. Tässäkin tapauksessa on kuitenkin hyvä muistaa olla maltillinen, jotta näkymän yhdenmukaisuus säilytetään kuten luvussa 2.4.2 on kerrottu. Kuvassa 34 on esimerkki, miten näkymää voidaan muuttaa järkevämmäksi ja luonnollisemmaksi. Kuvassa vasemmalla on näkyvissä nostopöydän antureiden objektit sekalaisessa järjestyksessä. Kuvassa oikealla on näkyvissä nostopöydän anturit luonnollisessa järjestyksessä. Nostopöydän ylärajan anturi on siirretty alarajan anturin yläpuolelle, koska luonnollisesti yläraja on ylempänä kuin alarajan. Tällä luodaan eräänlainen yhteys oikean maailman ja käyttöliittymän objektin välille kuten luvussa 2.4.1 on

kerrottu. Tästä syystä käyttäjän on helpompi hahmottaa nostopöydän paikkaa oikeassa maailmassa. Tällaisilla pienillä asioilla on kokonaisuuden kannalta todella iso merkitys, ja ne parantavat huomattavasti käyttöliittymän käytettävyyttä.



KUVA 34. Käyttöliittymän objektit järjestettynä kahdella eri tavalla – vasemmalla objektit ovat mielivaltaisessa järjestyksessä, kun taas oikealla ne ovat luonnollisessa järjestyksessä.

4.5 Yhteenveto

Käyttöliittymäpäivityksen myötä asiakas sai uuden kestävän teollisuustietokoneen uudella modernilla käyttöliittymällä. Opinnäytetyön ensimmäisenä tavoitteena oli vanhan käyttöliittymän toiminnallisuuksien siirtäminen uuteen käyttöliittymään, johon sisältyi kommunikaatiorajapinnan tekeminen. Kuten luvussa 4.2 kerrottiin, Modbus-rajapinnan tekemisen myötä uusi käyttöliittymä sisälsi kaikki samat toiminnallisuudet ja ominaisuudet kuin mitä vanhakin käyttöliittymä. Tämän uudistuksen myötä asiakas sai käyttöönsä uuden käyttöliittymän, joka on visuaalisempi, selkeämpi ja käyttäjäystävällisempi kuin edeltäjänsä. Pelkästään tällä uudistuksella onnistuttiin parantamaan työntekijän käyttäjäkokemusta huomattavasti. Käyttäjäkokemus on käyttöliittymäsuunnittelussa todella tärkeää, kuten luvussa 2.3 on kerrottu.

Toisena tavoitteena oli siirtää osa toiminnallisuuksista ohjelmoitavalta logiikalta suoraan käyttöliittymään, josta on kerrottu luvussa 4.3. Toiminnallisuuksien siirtäminen PLC:ltä käyttöliittymälle oli mahdollista uuden tehokkaamman teollisuus-PC:n asentamisen myötä. Toiminnallisuuksien siirtämisen etuna oli turhan väyläliikenteen karsiminen sekä kuorman siirtäminen PLC:ltä käyttöliittymään. Tällä toimenpiteellä pystyttiin parantamaan järjestelmän suorituskykyä.

Viimeisenä tavoitteena oli tehdä päivityksiä ja parannuksia uuteen käyttöliittymään ulkoasun ja käyttäjäkokemuksen osalta. Käyttöliittymän visuaalinen ulkoasu muuttui päivityksen myötä – uusi käyttöliittymä on yhdenmukainen, käyttäjäystävällinen ja selkeä, kuten sen luvussa 2.3 käydyn UX-suunnittelun näkökulmasta pitääkin olla. Näitä UX-suunnittelun keskeisimpiä asioita pyrittiin vielä entisestäänkin korostamaan käyttöliittymän objektien ja näkymien muokkaamisella – objektien lisäämisellä ja poistamisella sekä näkymien visuaalisella selkeyttämisellä oli huomattava vaikutus käyttöliittymän ulkoasuun ja sitä kautta koko käyttöliittymän käyttökokemukseen. Käyttöliittymän kustomoinnista kerrottiin yksityiskohtaisemmin luvussa 4.4.

5 POHDINTA

Opinnäytetyöprojektin tarkoituksena oli toteuttaa käyttöliittymäpäivitys opinnäytetyön tilaajan asiakasyritykselle. Tavoitteena oli siirtää vanhan käyttöliittymän toiminnallisuudet uuteen käyttöliittymään, sekä kehittää ja kustomoida uuden käyttöliittymän ulkoasua asiakkaan tarpeiden mukaan. Projektin keskeisimpänä kriteerinä oli käyttöliittymän työstäminen toimintakuntoiseksi aikataulussa pysyen – käyttöliittymäpäivityksen oli oltava valmiina sovittuun käyttöönottotapahtumaan mennessä. Asiakas sai käyttöliittymän toimintakuntoisena sovittuun päivämäärään mennessä, joten projekti oli onnistunut. Asiakas oli tyytyväinen tuloksiin, kuten oli myös opinnäytetyön tilaaja.

Käyttöliittymän päivitys onnistui toiminnallisuuksien siirron sekä rajapinnan tekemisen osalta erinomaisesti – käyttöliittymä saatettiin toimivaksi kaikilta osa-alueilta. Toiminnallisuuksien siirto oli määritelty projektin alkaessa ensisijaiseksi tehtäväksi, sillä se oli välttämätöntä käyttöliittymän toimivuuden kannalta. Opinnäytetyön teoreettista lähtökohtaa ja projektin kannalta toissijaista tehtävää eli käyttöliittymän kehittämistä ja kustomointia olisi voinut vielä viedä projektissa pitemmälle. Käyttöliittymää onnistuttiin parantamaan UI- ja UX-suunnittelun perusperiaatteiden mukaisesti. Käyttöliittymän tarkoitus on toimia mahdollisimman yksinkertaisena, nopeana ja selkeänä rajapintana koneen ja ihmisen välillä ja mielestäni tähän päästiin.

Jatkokehitysideana on kehittää käyttäjäkokemusta käyttöliittymässä paremmaksi. Käyttäjäkokemuksen parantaminen on koko tuotteen elinkaaren mittainen projekti, sillä käyttäjäkokemusta ei voi ikinä tehdä täydelliseksi, vaan siitä löytyy aina parannettavaa. Käyttäjäkokemus myös perustuu osaltaan yksittäisen ihmisen mielipiteeseen ja preferenssiin, jolloin se on jokaisella asiakkaalla yksilöllinen. Paremmilla resursseilla käyttöliittymän voisi jopa suunnitella kokonaan uudestaan, jolloin näkymiä olisi mahdollista yhdistää ja näkymien kokonaislukua supistaa. Tämä kuitenkin vaatisi syvällistä ymmärrystä käyttöliittymästä sekä prosessista, johon sitä käytetään.

LÄHTEET

Babich, N. 2019. Adobe. The 4 Golden Rules of UI Design. Luettu 10.12.2019. https://xd.adobe.com/ideas/process/ui-design/4-golden-rules-ui-design/

Beijer Electronics, n.d. iX HMI Software. Luettu 22.2.2020. https://www.beijerelectronics.com/en/Products/software/ix-hmi-software

Center for Chemical Process Safety. 2014. Sequential Control System. Luettu 3.12.2019. https://www.aiche.org/ccps/resources/glossary/process-safety-glossary/sequential-control-system

Computer Hope. 2019. What is a GUI (Graphical User Interface)? Luettu 6.12.2019. https://www.computerhope.com/jargon/g/gui.htm

ConceptDraw. 2015. Android User Interface: Sample 3: Android GUI – Android 5.0 App Drawer. Katsottu 12.5.2020. https://www.conceptdraw.com/samples/software-android-user-interface

De La Riva, M. 2018. Career Foundry. Why Consistency Is So Incredibly Important In UI Design. Luettu 10.12.2019. https://careerfoundry.com/en/blog/ui-design/the-importance-of-consistency-in-ui-design/

Educba. 2019. What is GUI? Luettu 18.12.2019. https://www.educba.com/what-is-gui/

Galitz, W.O., 2007. The Essential Guide to User Interface Design: An Introduction to GUI Design Principles and Techniques. 3. painos. Yhdysvallat: John Wiley & Sons.

Gantt.com. 2020. What is a Gantt Chart? Luettu 22.3.202. https://www.gantt.com/

Glasstech. n.d. Mikä on karkaistu lasi? Luettu 3.1.2020. https://www.glasstech.ee/mika-on-karkaistu-lasi

Glas In Beeld. 2014. Gehard glas vijf keer sterker dan floatglas: goed of fout? Katsottu 12.5.2020. https://www.glasinbeeld.nl/wp-content/uploads/2014/06/Spantex-2-800x599.jpg

Gonzalez, C. 2015. Machine Design. Engineering Essentials: What Is a Programmable Logic Controller?. Luettu 17.2.2020. https://www.machinedesign.com/learning-resources/engineering-essentials/article/21834250/engineering-essentials-what-is-a-programmable-logic-controller

Hiekkanen, K. 2016. Yritä.fi. Trello: Mahtava ja ilmainen organisointityökalu. Luettu 22.1.2020. https://www.yritä.fi/blogi/trello-mahtava-ja-ilmainen-organisointityokalu

Inductive Automation. 2018. What is SCADA? Luettu 21.1.2020. https://inductiveautomation.com/resources/article/what-is-scada

Kepware. 2018. Modbus 5 digit addressing versus 6 digit addressing. Luettu 27.2.2020 https://www.kepware.com/en-us/support/knowledge-base/2008/modbus-5-digit-addressing-versus-6-digit-addressin/

Modbus. n.d. Modbus FAQ. Luettu 27.2.2020. http://www.modbus.org/faq.php

Mortensen, D. 2020. Interaction Design Foundation. Natural User Interfaces – What are they and how do you design user interfaces that feel natural? Luettu 20.1.2020. https://www.interaction-design.org/literature/article/natural-user-interfaces-what-are-they-and-how-do-you-design-user-interfaces-that-feel-natural?ep=ux-planet

National Instruments. 2019. The Modbus Protocol In-Depth. Luettu 29.2.2020. https://www.ni.com/fi-fi/innovations/white-papers/14/the-modbus-protocol-in-depth.html

Schneider Electric. n.d. Modicon TSX Micro. Katsottu 12.5.2020 https://download.schneider-electric.com/files?p_Doc_Ref=Micro_540_RPFJR18002&p_File_Type=rendition_369_jpg

Schneider Electric. n.d. Vijeo Look – PC based HMI software. Luettu 22.1.2020. https://www.se.com/fi/fi/product-range-presentation/657-vijeo-look/#tabs-top

Shneiderman, B. 2016. University of Maryland. The Eight Golder Rules of Interface Design. Luettu 12.12.2019. https://www.cs.umd.edu/users/ben/goldenrules.html

Sirin Software. 2019. HMI and GUI: Are They Substitutes for One Another? Luettu 20.1.2020. https://sirinsoftware.com/blog/hmi-and-gui-are-they-substitutes-for-one-another/

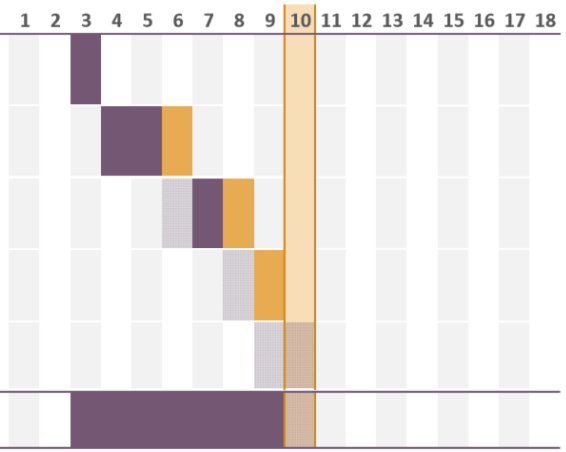
UX Planet. 2017. Design principle: Error & Forgiveness. Luettu 15.5.2020. https://uxplanet.org/design-principle-error-forgiveness-1495f7471113

Liite 1. Gantt-kaavio

Projektin suunnittelu

Viikon korostus: 10

TEHTÄVÄ	SUUNNITELTU ALOITUS	SUUNNITELTU KESTO	TODELLINEN ALOITUS	TODELLINEN KESTO	PROSENTTIA VALMIINA
Kartoitus ja tutustuminen	3	1	3	1	100 %
Modbus-rajapinta	4	2	4	3	100 %
Toiminnallisuuksien siirto	6	2	7	2	100 %
Puutteiden ja toiveiden lisäys	8	1	9	1	75 %
Testaus ja korjaus	9	1	10	1	0 %
Kokonaiskesto	3	7	3	8	88 %



Selitykset

- Suunniteltu kesto
- Todellinen aloitus
- % valmiina
- Todellinen (yli suunnitellun)
- % valmiina (yli suunnitellun)