Opinnäytetyö (AMK)

Kone- ja tuotantotekniikan koulutus

2019

Aki Heinonen

PUUKUIVAAMON KÄYTTÖLIITTYMÄN SUUNNITTELUN KEHITYS

TURKU AMK
TURKU UNIVERSITY OF APPLIED SCIENCES

OPINNÄYTETYÖ (AMK) | TIIVISTELMÄ

TURUN AMMATTIKORKEAKOULU

Kone- ja tuotantotekniikan koulutus

2019 | 24 sivua

Aki Heinonen

PUUKUIVAAMON KÄYTTÖLIITTYMÄN SUUNNITTELUN KEHITYS

Opinnäytetyön tavoitteena on päivittää käyttöliittymän suunnitteluohjelma uudempaan ohjelmaversioon sekä kehittää suunnittelusta kustannustehokkaampaa. Työ tehdään, koska halutaan minimoida suunnittelun kustannuksia säästämällä työhön kuluvia työtunteja.

Työ etenee tutkimalla ongelma osa-alueet suunnittelussa ja selvittämällä vanhemman sekä uudemman suunnitteluohjelman version eroavaisuudet. Kehityskohteiksi tutkimusten perusteella todettiin ongelmalliseksi kommunikaatioiden tekeminen valvomokoneen sekä logiikan välillä, jotka ratkaistaan uudemman ohjelman mahdollistamalla tagien integroinnilla. Muutoksia tehdään myös ikkunoihin, jotka muokataan sisältämään uudet muuttujanimet. Kielikäännösten helpottamiseksi luodaan taulukko, joka saadaan tuotua uudelle projektille.

Luokkarakenteisella ohjelmoinnilla saavutetut edut yhdessä muuttujien integroinnilla saavutetaan suunnittelusta mielekkäämpää sekä yhtenäisempää. Muuttujien integroinnilla suoraan logiikan ohjelmasta saatiin kommunikaatio toimimaan logiikan sekä valvomokoneen välillä helpottamaan kommunikoinnin tekemistä. Ikkunoiden muutoksilla modulaarisemmiksi ja integroinnilla on mahdollista saavuttaa suunnitteluun käytettävien työtunteja vähemmäksi.

ASIASANAT:

käyttöliittymä

BACHELOR´S / MASTER'S THESIS | ABSTRACT

TURKU UNIVERSITY OF APPLIED SCIENCES

Mechanical and Production Engineering

2019 | 24, number of pages in appendices

Aki Heinonen

WOOD DRYERS USER INTERFACE DESIGN DEVELOPING

The objective of this thesis was to update the current design software to a newer version and to make design more cost-effective. The objective was to minimize the cost of design by saving work hours.

The work progressed by examining the problem areas in the design and identifying the differences between the older and the newer design program versions. Based on the researches, the development of communications between the monitoring machine and the logic, which is solved by the integration of tags by a newer program, was found to be problematic. Changes were also made to windows that are modified to include new variable names. To facilitate language translation, a table that can be imported into a new project was created.

The benefits of classroom programming, together with the integration of variables, are more meaningful and coherent than design. By integrating variables directly from the logic program, communication between the logic and the control machine was made to facilitate communication. With the modifications of the windows and the integration of the windows, it is possible to use fewer working hours for design.

KEYWORDS:

User Computer Interface

SISÄLTÖ

KUVAT

KÄYTETYT LYHENTEET TAI SANASTO

HMI = Käyttöliittymä (Human Machine Interface)

IWS = Suunnitteluohjelma (Indusoft Web Studio)

PLC = Ohjelmoitava logiikka (Programmable Logic Controller)

SCADA = Valvomo (Supervisory Control And Data Acquisition)

TAG = Muuttuja

1 JOHDANTO

Opinnäytetyön tavoitteena on tutkia ongelmaosa-alueet puukuivaamon käyttöliittymän suunnittelemisessa. Kyseessä on toiminnallinen opinnäytetyö. Taustana työlle toimii useat projektien tuomat vaikeudet suunnittelussa sekä kustannustehokkuuden parantaminen. Tavoitteena tutkimustyölle on saada suunnittelemisen ongelmat hidasteineen selvitettyä sekä ongelmat ratkaistua. Pohjana työssä käytetään valmiita vanhoja projekteja, joista otetaan mallikuvat. Toiminnallisessa työ osuudessa luodaan projekti uudemmalla suunnitteluohjelman versiolla, joka olisi helposti muokattavissa uudelle projektille. Lähtökohdallisesti tutkimukseen sisältyvistä ongelmista tiedetään jo melkoisesti. Ongelmat voidaan jakaa muutamaan pääkategoriaan, jotka ovat:

• kommunikointi osoitteiden teko

• haasteelliset ”Graafiset” muokkaukset

• kielikäännösten tekeminen sekä yhtenäistäminen

Työ on tärkeä, koska uusien projektien luomiseen kuluu paljon työtunteja ja kustannustehokkaat ratkaisut ovat välttämättömiä. Lähdemateriaalina työssä käytetään ohjelmistojen käyttöohjeita sekä verkkomateriaalia. Aiheeseen liittyvää kirjallista lähdemateriaalia ei juurikaan ole saatavilla.

Käyttöliittymän suunnittelun kehitys tehdään yhteistyössä Heinolan Sahakoneet Oy:n kanssa, jonka vuoksi opinnäytetyöstä jätetään yksityiskohtaiset tiedot pois. Työssä esitellään käytetyt kehitysmahdollisuudet ja parannukset.

2 KÄYTTÖLIITTYMÄN SUUNNITTELUPROSESSI

Suunnittelu on monipuolinen prosessi, ja jokainen prosessi on erilainen. Suunnittelijat eroavat myös erilaisilla lähestymistavoillaan jokaiseen tehtävän ja ongelmanratkaisuun omalla tavallaan. (Saitamaa-Hakkarinen 2019.) Suunnitteluprosesseja on tutkittu paljon ja suunnitteluprosessien eri malleja on todella paljon. Niitä voidaan hyödyntää myös käyttöliittymän niin kuin muussakin suunnittelussa. Esimerkiksi Markus ja Maverin suunnittelumalli, joka koostuu neljästä toistuvasta kohdasta suunnittelun edetessä: analyysi -> synteesi -> arviointi -> päätös (Seitamaa-Hakkarinen 2019). Ketjua toistetaan, kunnes on saavutettu haluttu lopputulos.

Käyttöliittymän suunnittelussa on keskiössä loppuasteen käyttäjä. HMI:n eli ihmisen ja koneen välisessä käyttöliittymän suunnittelussa on paljon erilaista huomioitavaa verrattuna esimerkiksi ohjelmarajapintaan, joka kommunikoi eri ohjelmien välillä. Koneen välisessä rajapinnassa tulee myöskin huomioida toimintavarmuus sekä käyttäjäkohtaiset tasot. Tasojen avulla luodaan eri käyttäjien mahdollisuus operoida eri asioita koneessa. Tällä on mahdollista estää asiaankuulumattomien pääsy operoimaan konetta, kuten käynnistämään moottoria. Turvatasojen määrittely käyttöliittymässä on olennainen osa turvallisuutta. Myös ohjelmarajapinnoissa käytetään tasokohtaisia käyttäjiä mutta se harvemmin on niin kriittistä kuin koneiden kanssa. HMI:n suunnittelussa on myös paljon muuta huomioitavaa, kuten värimaailman, kontrollien ja indikaattoreiden määrässä sekä objekteissa, jotka eroavat tavallisista tietokone- ja mobiiliohjelmista melkoisesti. Taustan värisävyinä suositellaan käyttämään harmaata sävyä sekä muissa objekteissa hillittyjä värejä. Tiettyjä värisääntöjä asetellaan värien käyttämisissä koneiden käyttöliittymissä, joita on:

Punainen: Hälytys tai vaara.

Vihreä: Käynnistys tai hyvä tila.

Keltainen: Varoitus.

Sininen: Pakollinen operaatio.

Näitä värejä ei tulisi käyttää muissa toimissa kuin edellä mainituissa. Yleisesti hyvän HMI:n tulisi näyttää tylsältä ennemmin kuin kauniilta tai loistokkaalta hienoilla animaatioilla (HMI design, 2012.)

Tämän projektin käyttöliittymää ei varsinaisesti suunnitella vaan se on jo muodostunut monien vuosien saatossa. Työssä keskitytään suunnittelemisen helpottaviin ja parantaviin toimiin. Lyhykäisyydessään suunnittelu alkaa tavallisesti lähtötietojen hankinnasta ja valmistelusta. Valmistelujen jälkeen otetaan muutokset ja käyttäjän vaatimukset huomioon ja määritellään käyttäjän tarpeet. Usein HMI:n suunnittelussa pyydetään käyttäjän mielipidettä käytettävyyden kannalta merkittäviin asioihin. Prosessin aikana saattaa useasti joutua palaamaan edellä hankittuihin lähtötietojen hankintaan, kuten Markus ja Maverin suunnittelumallissa on esitelty. Tässä työssä lähtötiedot kerätään vanhoista projekteista ja työssä ei ole varsinaista asiakaskäyttäjää.

3 SUUNNITTELUOHJELMA

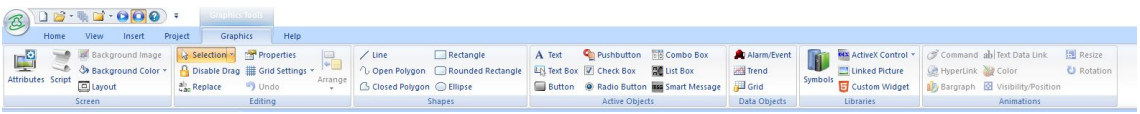
Käyttöliittymä on toteutettu Indusoft web studio (myöhemmin tekstissä IWS) -ohjelmalla, joka on yhdysvaltalainen vuonna 1997 perustetun yrityksen tuottama ohjelma. Nykyään IWS kuuluu myös Schneider Electrici:n ohjelmistoon. IWS-ohjelma on tehokas web-pohjainen HMI SCADA -ohjelma, joka soveltuu moniin eri teollisuuden tarpeisiin. Graafisella muokkausmenetelmällä suunnittelu onnistuu helposti ilman kehittyneitä tietoteknisiä taitoja, mutta mahdollisuudet VBscriptin käyttöön on myös. VBscriptillä saavutetaan monimutkaisempia laajennuksia tehtyä. Ohjelmassa on myös paljon valmiita yleisiä funktioita, joita ohjelma käyttää nimityksellä ”Built-in language”.

3.1 Lisenssi

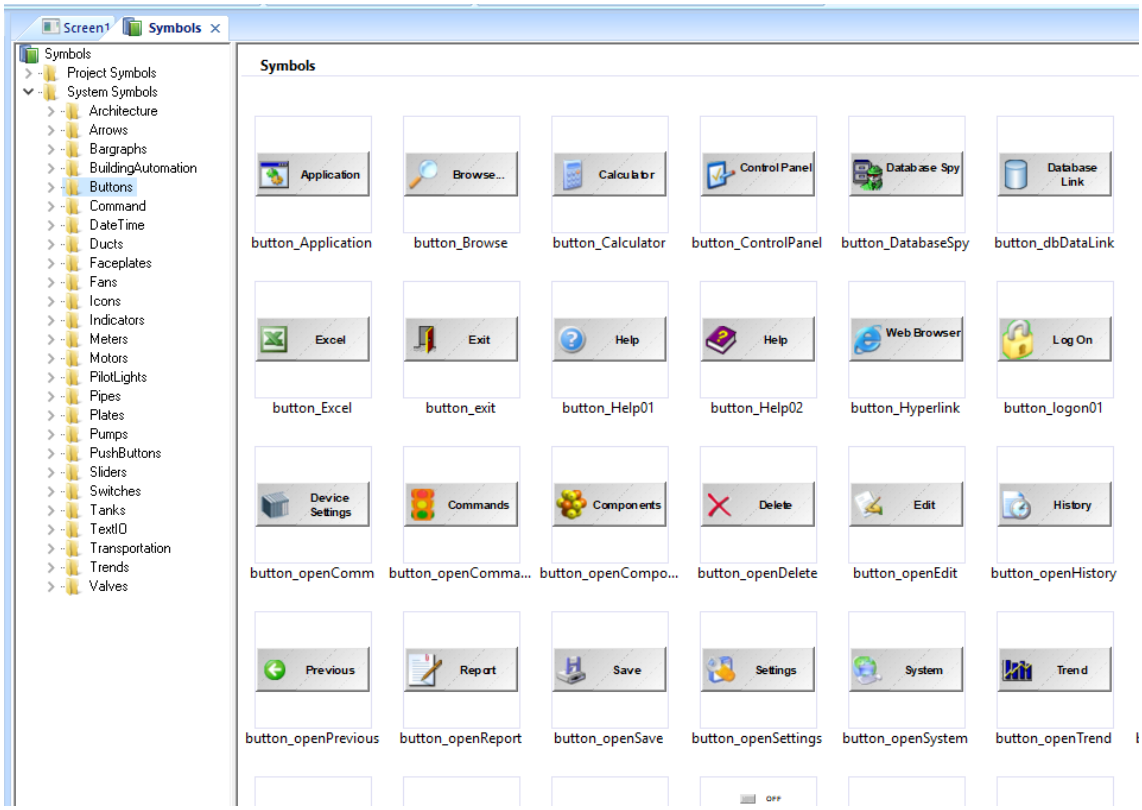
Ohjelman lisensointi jaottuu kahteen pääryhmään kehitysversioon sekä ”runtime”-versioon, joka on ainoastaan valmiin valvomon käyttämiseen tarkoitettu ja lisenssillä ei pysty muokkaamaan valvomoa. Kehitysversion ”Engineering”-lisenssi jaotellaan projektin koon mukaan muuttujien perusteella. Muuttujien määrä siis määrää lisenssin koon.

3.2 Graafinen editori

Graafinen editori tarjoaa paljon valmiita työkaluja graafiseen suunnitteluun ja helpottaa indikaattoreiden sekä ohjauspainikkeiden käyttöä suoraan ”drag and drop” -menetelmällä (Kuva 1). Symbolikirjastossa on valmiita symboleita, joilla voidaan helposti luoda näyttäviä kuvia (Kuva 2). Mahdollista käyttää myös ActiveX-komponentteja.



Kuva 1. Graafiset työkalut.

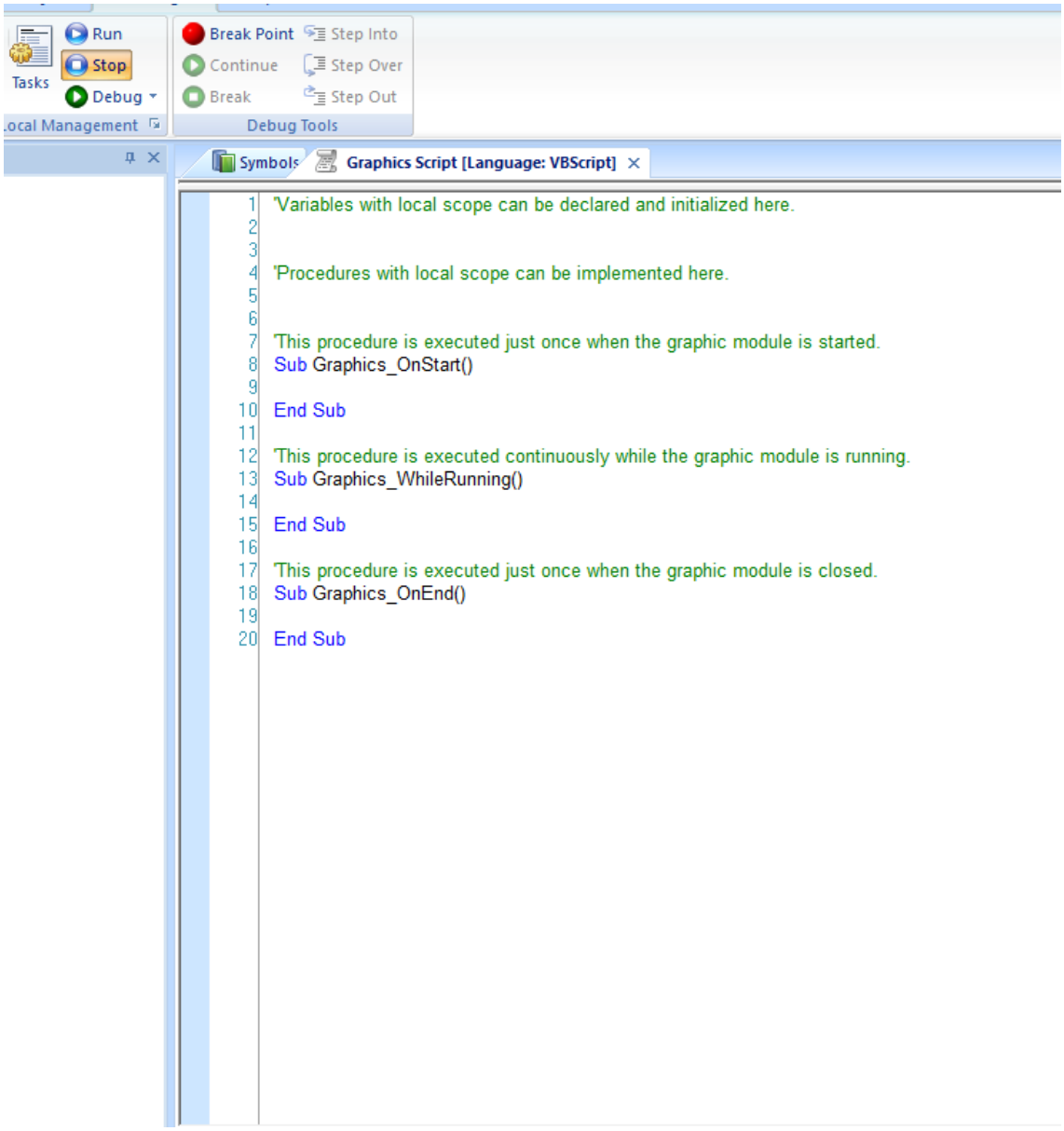


Kuva 2. Valmis symbolikirjasto.

3.3 Tekstieditori

Tekstieditori käyttää VBScript (Visual Basic Script) -kieltä, joka on yksi Windowsin monista ohjelmointikielistä. VBScript on alamuoto Microsoftin Visual Basic -ohjelmointikielelle, joten siinä on tietynlaisia rajoituksia. VBScriptillä kaikki muuttujat ovat variant-tyyppisiä eli ohjelma muodostaa itse muuttujan tietotyypin. Käyttö rajoittuu vain Microsoftin selaimiin suoraan. Selaimiin asennettavilla lisäolilla pystytään myös käyttämään muissa selaimissa. Built-in language on IWS-ohjelman sisään rakennettu koodi tai funktiokirjasto, jolla pystytään mm. muotoilemaan päivämääriä, aikoja ja tekemään tiedostojen käsittelyt. Built-in-funktioiden käyttö vaatii ohjelmakoodissa dollarimerkin funktion eteen funktiota kutsuessa. Dollarimerkillä ohjelma osaa erottaa ohjelman omat muuttujat ja funktiot VBScriptin muuttujista ja funktioista ja osaa kutsua niitä oikeasta paikasta. Nykyinen IWS 8 -version tekstieditori on parantunut huomattavasti vanhemmista verioista. Uudemmassa versiossa on tekstin rivit näkyvissä sekä mahdollista testata koodi ennen julkaisemista. Vanhemmissa versioissa varsinainen testauksen on saanut tehtyä vasta kun ohjelman on ottanut käyttöön (Kuva 3). Omia

ohjelmakoodeja pystytään muodostamaan jokaisessa ikkunassa, omana tekstitiedostona sekä globaaleina funktioina. Myös omista koodi tiedostoista on mahdollista tuoda koodeja ohjelmalle.



Kuva 3. Tekstieditori.

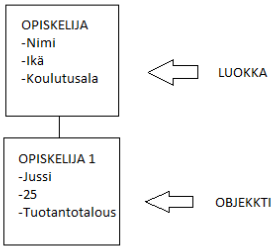
4 KÄYTTÖLIITTYMÄN SUUNNITTELUN KEHITYS

4.1 Lähtökohdat

Projektin suunnittelu alkoi kokonaan alusta luomalla tyhjä projekti, johon tarvittavat tiedot poimittiin vanhoista projekteista. Vanhoista projekteista käytettiin hyväksi valmiita ikkunoita, jotka ovat jo muodostuneet monien vuosien varrella ilmeisen hyviksi eikä niihin tehty muutoksia. Ikkunoiden resoluutiot tuli vaihtaa samankokoiseksi jokaiseen ikkunaan sekä määrittämällä symbolikuvakkeiden lähteet uusiksi. Vanhan version symbolit eivät uuteen suoraan muodostunut, vaan symbolit piti asettaa kuviin uudelleen hakemistosta. Testilogiikkaan ladattiin malliprojekti, josta sai tarvittavat tagit uudelle projektille. Testilogiikalla pystyttiin kokeilemaan kommunikoinnin ja tietokantaominaisuuden toimivuus. Logiikkana toimi Siemensin CPU 1510SP F-1 PN. Logiikkaohjelma myös integroinnissa välttämätön, jotta tyhjään valvomo-ohjelmaan sai tagit tuotua oikeilla nimillä.

4.2 luokkarakenteet

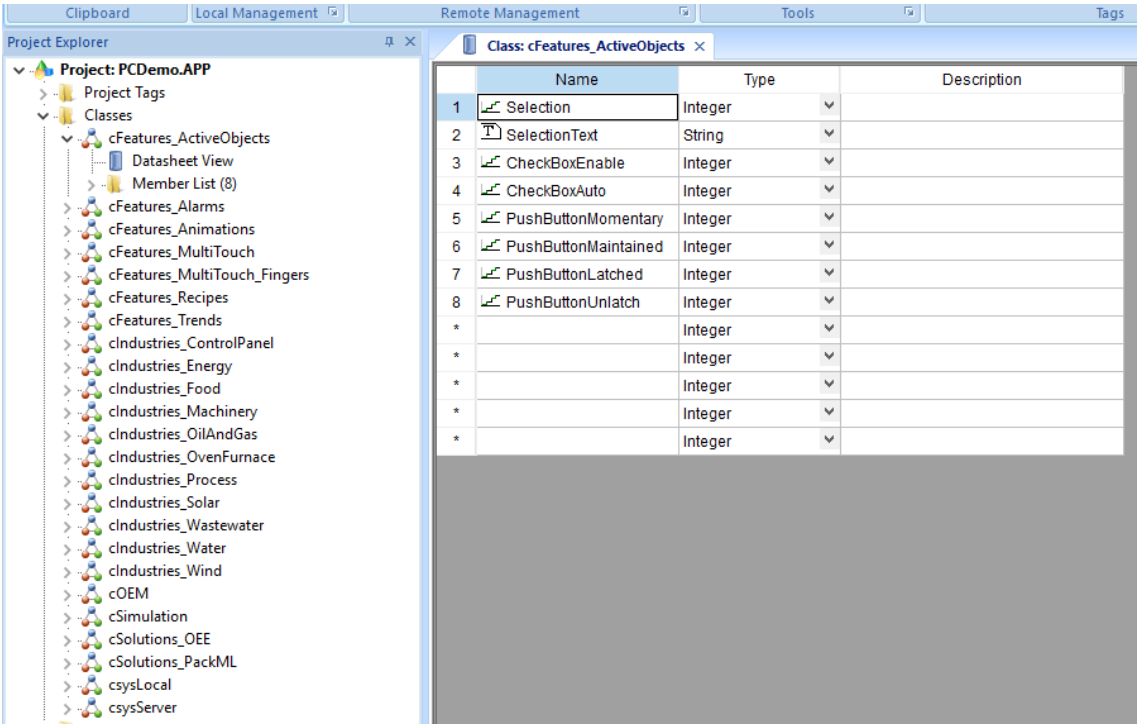
Luokkarakenteita kutsutaan nimellä ”Class”. Luokkarakenteilla saadaan aseteltua muuttujille tietynlainen mallipohja. Mallipohjia voidaan hyödyntää muuttujissa ja sillä saadaan luokkaominaisuudet käyttöön jokaiseen luokkatyyppiin määriteltyyn muuttujaan. Voidaan ajatella muuttujalla olevan tietynlainen rakenne tai tietyt ominaisuudet. Luokallisesta muuttujasta kutsutaan nimellä objekti. Objekti sisältää kaikki luokan ominaisuudet (Kuva 4).



Kuva 4. Esimerkki luokasta.

4.2.1 Hyödyt

Paras hyöty luokkarakenteista saadaan, kun tehdään useampi samanlainen rakenne, ohjaus tai ikkuna, jossa käytössä samat muuttujat. Luokkarakenteiden käyttö onkin siis suotavaa isoissa projekteissa sekä projektien yksinkertaistamisessa (Kuva 5). Luokkarakenteet yhdessä epäsuorien muuttujien kanssa on tavoiteltu rakenne, koska muutokset on helppo tehdä ja kopiointi uuteen kanavaan on nopeaa. Nykyisessä valvomon ohjelmassa on melko hyvin jo luokkarakenteita käytetty, mutta niiden uudelleen rakentaminen on tässä välttämätöntä, jotta saadaan integroidut logiikan muuttujat toimivaan uudessa tyhjässä projektissa.



Kuva 5. Esimerkki luokkien määrittelystä.

4.2.2 Ryhmittely

Järkevällä ryhmittelyllä eli luokilla saavutetaan suunnittelusta helpompaa. Mitkä ovat hyvät ryhmät? Ryhmiä miettiessä otetaan huomioon oleellisesti, miten halutaan tulevaisuudessa projekteja luoda. Ikkunoittain ryhmittely olisi siis vaihtoehto, mutta mm. prosessi ikkunassa muuttujien määrä todella iso, joten isoissa ikkunoissa ryhmittely toteutetaan pienempinä kokonaisuuksina. Ryhmittely uudessa projektissa tehdään logiikka ohjelmassa, jotta integroinnin tuomat edut saadaan maksimaaliseen hyötyyn.

Useat ponnahdusikkunat toimivat tietylle toimilaitteelle, joten niissä on järkevä käyttää luokkarakenteita. Luomalla ikkunassa esiintyvistä muuttujista luokka on mahdollista asettaa luokka sisältämään tietyt sille kuuluvat muuttujat. Ikkunassa ei kuitenkaan ole pakko käyttää kaikkia luokan muuttujia.

4.3 Siemens integrointi

Uudella IWS-versiolla on mahdollista integroida logiikassa käytettyjen tagien muuttujat suoraan IWS-ohjelmaan. Integrointi luo kommunikointiosoitteet automaattisesti SITIA-ajurille. Tämä on erittäin hyvä ominaisuus, jota ei aikaisemmissa versioissa ollut, joten ainoan vaihtoehtona oli kirjoittaa muuttujat käsin sekä logiikkaan ja valvomo ohjelmaa. Muuttujien ja osoitteiden määrä vaihtelee projekteittain mutta pahimmillaan taulukkoon joutuu kirjoittamaan enemmän kuin 10 000 osoitetta. Tämä aiheuttaa todella helposti virhelyöntien määrää sekä ongelmia kommunikoinnissa PLC:N ja valvomokoneen välillä. Turhat ja ylimääräiset tagit hidastavat ohjelman pyörimisnopeutta huomattavasti sekä niiden sisältäessä virheen saattaa tagin luenta katkaista osittain myös muiden tagien luennan. Virheiden hallintaa saa seurattua ainoastaan logia seuraamalla.

4.3.1 Integrointi ongelmat

Integroidessa muuttujia suoraan logiikasta tuottaa muutamia haittapuolia. Ohjelma vaatii toimivan logiikan, jossa valvomoon tuotavat muuttujat ovat jo määriteltyä. Integrointi luo valvomokoneen sekä PLC:n välille jaetun tietokannan, joka määrittää muuttujien tietotyypit ja yhteydet suoraan. Muuttujien tietotyyppejä ei pystytä IWS-ohjelmassa muuttamaan vaan muutos täytyy tehdä logiikassa. Muuttujat siis täytyy tehdä etukäteen logiikkaan ja tuoda IWS-ohjelmaan, jossa niiden käyttö toimii nimien perusteella. Myös useammasta logiikasta saa tuotua samannimisiä muuttujia, jolloin integroinnissa asetetaan useampi eri laite ja näiden omat IP-osoitteet. IWS erottaa eri logiikoiden muuttujat luodun laitteen mukaan, mikä näkyy käytettävässä muuttujan nimessä. Logiikasta on mahdollista valita tuotavat muuttujat. Logiikkaohjelman datalohkojen nimien on oltava yhteen kirjoitettuja eikä niissä saa olla erikoismerkkejä. Tuonnin jälkeen IWS osaa päivittää muuttujan tietotyypin, mikäli sitä vaihdetaan logiikan ohjelmassa. Muuttujan tai datalohkon nimen muuttamisen jälkeen on IWS-ohjelmassa määriteltävä muuttuja tai datalohko uudelle nimelle.

Luokkarakenteet IWS-integrointi osaa luoda myös automaattisesti, mutta vaatii, että logiikan ohjelmassa nämä ovat tehtyinä. Saman nimiseen luokkaan integrointi ei osaa

tuoda objektia. Luokkien käyttö vaatii logiikassa valmiiksi määritellyt luokat. IWS:ssa luotujen luokkien sekä projektin omien muuttujien yhteydet joudutaan kirjoittamaan käsin SITIA-ajurille.

4.3.2 Integrointi projektiin

Luomalla valmis ohjelma logiikkaan, johon on määriteltyä omiin datalohkoihin IWS ja logiikan väliset muuttujat, saadaan logiikan muuttujia käytettyä suoraan IWS-ohjelmassa. Myöskin näihin on kommunikointi valmiina rakennettu. Integroinnilla pystytään minimoimaan kommunikointiongelmat. Muuttujien tietotyypit ovat myös valmiiksi määriteltynä, eikä valvomo ohjelmaa suunniteltaessa tarvitse puuttua muuttujien tietotyyppeihin. Tämän hetken valvomossa käytettyjen muuttujien nimet on kirjoitettava uudella nimellä, jotta saadaan integroidut muuttujat käyttökelpoisiksi IWS-ohjelmassa. Integrointi luo tuotujen tagien nimet itse eikä niitä pystytä muokkaamaan kuin logiikan ohjelmassa. Muutosten jälkeen IWS:ssa on määriteltävä muuttunut muuttujan nimi uudelle nimelle. Luotu muuttujan nimi koostuu laitteen nimestä, mahdollisesta tietolohkosta sekä itse tagista. Jokaiselle eri logiikalle luotu yhteys määritetään erinimiseksi laitteeksi, jolla mahdollistetaan saman nimisten tagien tuonti eri logiikoilta.

4.3.3 Kommunikointi

Kommunikointi integroinnissa tapahtuu HMI:n sekä PLC:n välillä SITIA SIEMENS S7 -1200/S7-1500 -ajurilla. SITIA-ajuri käyttää Siemens S7 -protokollaa, joka on Siemensin logiikoille tarkoitettu protokolla. Ajurilla on alla listatut laitevaatimukset:

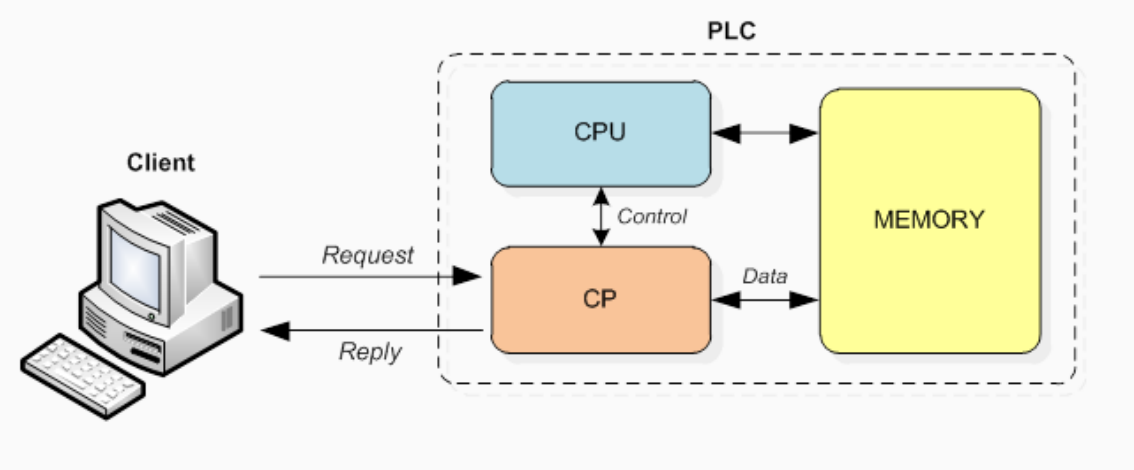
- Valmistaja: Siemens
- Soveltuvat laitteet: S7-1500 PLC:t, S7-1200 PLC:t laiteohjelmisto (firmware) v4.0 tai korkeampi
- Ohjelmointi sovellus: Siemens TIA portal v13 tai korkeampi

Verkkovaatimukset:

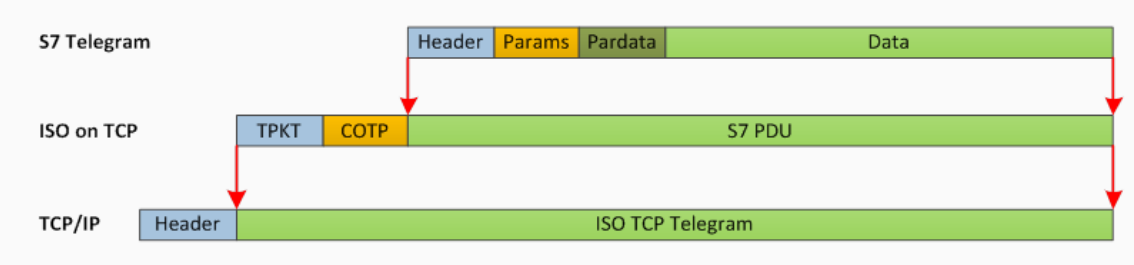
- Laite kommunikointi portti: 102

- Fyysinen protokolla: Ethernet
- Looginen protokolla: Siemens S7 protokolla
- Johdotustyyppi: Tavallinen ethernet kaapeli

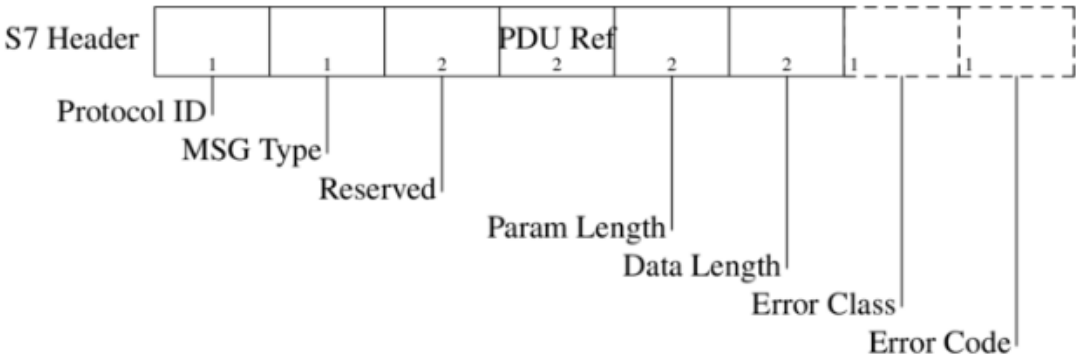
Siemens S7 -protokolla on yleisesti käytetty HMI:n sekä PLC:n välisessä kommunikoinnissa. Datapaketti on kapseloitu TCP/IP-paketin sisään (Siemens S7, 2019). S7-protokolla on funktio / komentokeskeinen, mikä tarkoittaa, että lähetys koostuu pyynnöstä ja sopivasta vastauksesta (Kuva 6). S7-datapaketti koostuu otsikosta, parametreista ja datasta (Kuva 7). Otsikko koostuu 10-12 tavun kokoisesta informaatiotaulusta, mitä tietoja datalohko pitää sisällään (Kuva 8).



Kuva 6. Esimerkki pyyntö ja vastaus kommunikoinnista (S7 Kommunikaatio, 2019).



Kuva 7. Kommunikointipaketti (S7 Kommunikaatio, 2019).



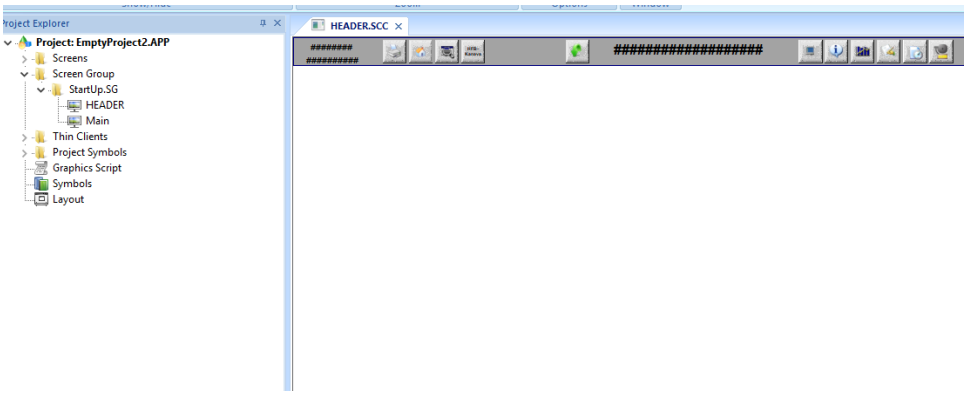
Kuva 8. S7 protokollan otsikkopaketti (Siemens S7, 2019).

4.4 Graafiset muutokset ja kielikäännökset

Graafisilla muutoksilla tarkoitetaan ikkunoiden ominaisuuksien sekä asettelujen parantamista. Graafisesti voidaan helpottaa uusien kanavien käyttöön ottamista sekä lisätä olemassa olevaan projektiin. Mikäli huomataan saman ikkunan ominaisuuden toistuvan useasti, on olennaista pyrkiä saamaan tuon lisääminen mahdollisimman helpoksi. Yleisimpiä toistokohteita ovat ponnahdusikkunat sekä navigaatiopalkit.

4.4.1 Navigaatio palkki

Aikaisemmissa projekteissa navigaatiopalkki oli ikkunakohtaisesti jokaisessa ikkunassa. Se tarkoittaa, että jokainen navigaatiomuutos tuli tehdä jokaiseen ikkunaan erikseen. Tämän helpottamiseksi asetetaan yksi erillinen ikkuna, johon navigaatiopalkki tehdään ja luodaan siitä ”startup”-ryhmä, johon kuuluu myös ”main”-ikkuna. Muut ikkunat muokataan kooltaan sopivan navigaatiopalkin alle ja ikkunan käynnistäessä ikkuna asettuu navigaatiopalkin alle. Tehdessä muutokset ei tarvitse muokata kuin navigaatiopalkin tietoja ja sama navigaatioryhmä näkyy käyttäjälle koko ajan (Kuva 9).

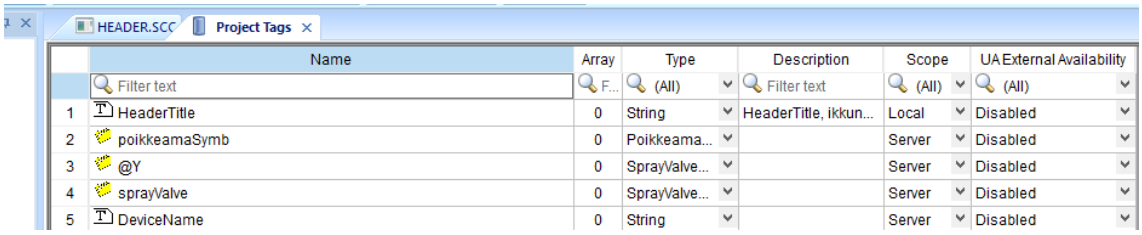


Kuva 9. Navigaatiopalkki.

4.4.2 Ponnahdus ikkuna

Ponnahdusikkunat eli ”PopUp”-ikkunat ovat hyvin käyttökelpoisia toimilaitteiden ohjaamiseen sekä asetusten asettamiseen. Ponnahdusikkunoiden paras hyöty tulee, kun niitä ei tarvitse tehdä kuin yksi projektia kohden. Muussa tapauksessa joudutaan tekemään kullekin kuivaamon kanavalle omat ponnahdusikkunat. Ikkunoiden muuttujien käyttö onnistuu parhaiten epäsuorilla muuttujilla. Epäsuorat muuttujat mahdollistavat kirjoittamisen sekä lukemisen varsinaiseen muuttujaan ns. epäsuorasti. Kyseisellä epäsuoralla muuttujalla on oltava samat ominaisuudet varsinaisen muuttujan kanssa, eli muuttujan on oltava samaa tyyppiä. Epäsuora muuttuja asetetaan IWS-ohjelmassa ”@”-merkillä muuttujan eteen (Kuva 10). Kun halutaan epäsuoran muuttujan käyttävän varsinaisen yhteyttä, asetetaan varsinainen muuttuja ””-merkkien sisäpuolelle. Pon-

nahdusikkunoissa muuttujat on järkevää asettaa epäsuoriksi muuttujiksi, että saadaan käytettyä samaa ikkunaa useammassa kuivaamon kanavassa.



	Name	Array	Type	Description	Scope	UA External Availability
	Filter text	F...	(All)	Filter text	(All)	(All)
1	HeaderTitle	0	String	HeaderTitle, ikkun...	Local	Disabled
2	poikkeamaSymb	0	Poikkeama...		Server	Disabled
3	@Y	0	SprayValve...		Server	Disabled
4	sprayValve	0	SprayValve...		Server	Disabled
5	DeviceName	0	String		Server	Disabled

Kuva 10. Epäsuoramuuttuja.

4.4.3 Linkitetyt symbolit

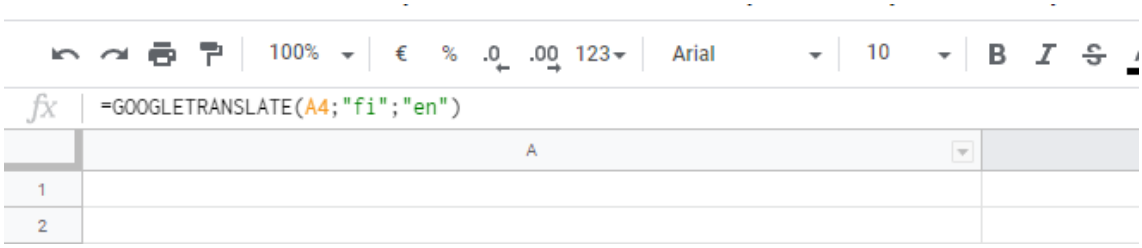
Linkitetyillä symboleilla tarkoitetaan symboleja, jotka ovat käytettävissä missä tahansa ikkunassa sekä symboli pystyy ottamaan vastaan niille määritellyt arvot. Symboleja pystyy luomaan sekä asettamaan erilaisia ominaisuuksia tai syötettäviä tietoja symbolille. Kun symboliin tehdään muutoksia, muutokset näkyvät jokaisessa symbolin esiintymässä. Niiden käytössä kuitenkin huomioitavaa, että monimutkaisten laskutoimitusten tekeminen saattaa olla hieman hankalahkoa. Sekä symboliin syötettävät arvot nollaantuvat suhteellisen helposti. Symboleilla kuitenkin helpotetaan ikkunoiden muokkaamista jossa paljon eri mittauksien indikaattoreita. Myös uuden muokatun symbolin lisääminen onnistuu kopioimalla vanha symboli ja tehtäessä muutokset uuteen.

4.5 Kielikäännökset

Hyvät kielikäännökset ovat olennainen osa hyvää käyttöliittymää. Heinolan projekteista moni toimitetaan ulkomaille, joten on erittäin tärkeää saada kielikäännösten tekeminen mahdollisimman helpoksi. IWS tarjoaa valmiin työkalun käännösten tekemiseen projektissa, mutta sen suurin kankeus on mm. käännösten kirjoittaminen taulukkoon. Uusien käännösten tekeminen seuraavassa projektissa on myös melkoisen vaivalloista, ilman hyvää pohjaa. Aikaisemmissa IWS-versioissa oli mahdollista käyttää Google translate lisäosaa ohjelmassa suoraan, joka osasi tehdä käännökset suoraan, mutta ohjelmaan liitettävä ominaisuus muuttui Googlella maksulliseksi ja hinnan riippuen käytöstä, niin IWS oli siitä päättänyt luopua.

IWS sisältää valmiin ominaisuuden, jolla helpotetaan kielikäännösten tekemistä. Ohjelma kerää taulukkoon kaikki ohjelman sisältä löytyvät tekstit ja tekee tästä erillisen käännöstiedoston. Tästä syystä taulukko sisältää todella paljon turhia tekstejä, joita ei tarvitse kääntää.

Käännösten tekemisen helpottamiseksi voidaan käyttää apuna Googlen Sheets ohjelmaa, joka toimii suoraan internet selaimella. Käyttö on ilmaista, mutta vaatii rekisteröinnin. Google Sheets:n avulla pystytään luomaan Googlen käännös funktio, joka kääntää taulukossa olevat arvot automaattisesti funktion sisälle määriteltyjen parametrien mukaisesti (Kuva 11). Käännökset pystytään tekemään useimmille käytettäville kielille (Google 2019). Kuitenkin on huomioitavaa, että käännökset on joka tapauksessa tarkistettava, koska osa käännöksistä erittäin huonoja. Taulukon saa tallennettua moneen eri tiedostomuotoon, ja vietyä IWS-ohjelmalle. Onkin siis järkevintä tehdä yksi korjattu taulukko, jossa yleisimmät käännökset ja viedä tämä aina uudelle projektille.



Kuva 11. Käännösfunktio, Google sheets ohjelmassa.

Kaikki ohjelmassa esiintyvät tekstit eivät käänny kääntöohjelmalla. Niitä ovat mm. VBscript:llä luodut viesti-ikkunat. Näissä tarvitsee manuaalisesti asettaa kieli. Viesti-ikkunat ovat käytännöllisisä ponnahdusikkunoita, joiden avulla voidaan pyytää operaattorilta lupaa johonkin toiminnon suorittamiseen sekä ilmoittaa toiminnon aloitus tai lopetus. Joten niiden käyttö melko yleistä ja niiden kieli on päätettävä projekti kohtaisesti.

Hälytys sekä tapahtuma käännöksissä käytetään monesti apuna muuttujaa, joka asetetaan sisältämään tietty tapahtuma tekstinä operaattorin toimesta. Kun operaattori pai-

naa tapahtuman käyntiin luo IWS siitä tapahtuman, joita pystytään seuraamaan jälkeenpäin tapahtumalistasta.

4.6 Tulokset

Tyhjän projektin luonti osoittautui yllättävän työlääksi johtuen monista tagien osoitteiden hakemisesta sekä uudelleennimeämisestä. Ensin oli ikkunakohtaisesti etsittävät vanhat muuttujat ja niiden perusteella vanhoista kommunikointi taulukoista sekä logiikkaohjelmista haettava näille vastikkeet, jonka jälkeen nimettävä uudelleen. Myös logiikkaohjelmaan joutui luomaan monille tageille luokkaryhmät, jotta integrointi toimii tulevaisuudessa mahdollisimman jouhevasti. Graafisten muutosten osuus uuden version myötä sujui melko hyvin ilman ongelmia. Kuvaikkunat oli osittain mahdollista kopioida vanhoista projekteista, mutta ikkunoiden attribuuteista kuvakoot sekä ikkunoiden aukeamispaikka oli aseteltava uudestaan.

5 LOPPUSANAT

Opinnäytetyön tavoitteena oli päivittää suunnittelu ohjelmisto uudempaan versioon ja saada suunnittelua kustannustehokkaammaksi. Työ aloitettiin tutkimalla suunnittelun haastavimmat ja eniten aikaa kuluttavimmat ongelmat. Tutkinnan jälkeen oli selvitettävä uuden version päivittämisen mahdollisuus sekä päivitetyn version mahdollistamat edut suunnitteluun. Päivitysversion ominaisuuksien testailun jälkeen integroitiin vanhan projektien tietoja ja graafiset kuvat uudelle projektille.

Opinnäytetyön tavoitteisiin eli suunnittelun kehittämiseen saatiin paljon käyttöön parempia ominaisuuksia uuden version päivityksen myötä. Myös tavoitteena oli soveltaa ohjelman kehitys kustannustehokkaammaksi suunnitteluksi vähemmillä suunnitteluun kuluvilla työtunneilla. Tyhjän projektin luomisessa oli melkoisesti työtä, joka helpottaa tulevaisuudessa projektien tekemistä ja vähentää virheiden määrää. Työläimpiä osuuksia oli ikkunoiden muuttujien sovittaminen vanhasta projektista uusiin integroituihin tageihin.

LÄHTEET

Google. 2019. Google support. Viitattu 20.4.2019:

https://support.google.com/docs/answer/3093331?hl=en

Siemens S7. The Siemens S7 Communication. Viitattu 20.4.2019:

http://gmiru.com/article/s7comm/

S7 Kommunikaatio. Siemens communications overview. Viitattu 20.4.2019:

http://snap7.sourceforge.net/siemens_comm.html

Seitamaa-Hakkarainen Pirita. Suunnitteluprosessien teoriaa. Viitattu 29.4.2019:

http://www.mlab.uiah.fi/polut/Design/teoria_suunnitteluprosessit.html

HMI Design. AC 2012-3605: HMI DESIGN: AN ANALYSIS OF A GOOD DISPLAY FOR SEAMLESS INTEGRATION BETWEEN USER UNDERSTANDING AND AUTOMATIC CONTROLS. Viitattu 29.4.2019:

https://www.academia.edu/35238248/AC_2012-3605_HMI_DESIGN_AN_ANALYSIS_OF_A_GOOD_DISPLAY_FOR_SEAMLESS_INTEGRATI-ON_BETWEEN_USER_UNDERSTANDING_AND_AUTOMATIC_CONTROLS