WWW.JAMK.FI

TIETOKANTAPOHJAINEN AUTOMAATION SOVELLUSSUUNNITTELU

Petri Kauppinen

Opinnäytetyö
Huhtikuu 2011

Automaatioteknologia, ylempi AMK
Tekniikan ja liikenteen ala

JYVÄSKYLÄN AMMATTIKORKEAKOULU
JAMK UNIVERSITY OF APPLIED SCIENCES

JYVÄSKYLÄN AMMATTIKORKEAKOULU
JAMK UNIVERSITY OF APPLIED SCIENCES

OPINNÄYTETYÖN
KUVAILULEHTI

Tekijä(t) KAUPPINEN, Petri	Julkaisun laji Opinnäytetyö	Päivämäärä 30.4.2011
	Sivumäärä 41	Julkaisun kieli SUOMI
	Luottamuksellisuus ( ) saakka	Verkkojulkaisulupa myönnetty (x)
Työn nimi TIETOKANTAPOHJAINEN AUTOMAATION SOVELLUSSUUNNITTELU		
Koulutusohjelma AUTOMAATIOTEKNOLOGIAN KOULUTUSOHJELMA, YLEMPI AMK		
Työn ohjaaja(t) SELOSMAA, Seppo RANTAPUSKA, Seppo		
Toimeksiantaja(t) RAMBOLL FINLAND OY HAKKARAINEN, Aki		

Tiivistelmä

Rambollin Automaatio- ja sähkö –yksikkö on kehittänyt Jalmari-suunnittelutyökalua sähkö-instrumentointi ja automaatiosuunnittelun avuksi. Opinnäytetyön lähtökohtana oli käydä läpi mahdollisuuksia ja tapoja integroida automaation sovellussuunnittelu osaksi Jalmaria. Työssä käytettiin Siemensin Step7- ja WinCC-suunnittelutyökaluja.

Opinnäytetyön tehtävänä oli käydä läpi niitä tapoja, joilla Step7:ään ja WinCC:hen on mahdollista generoida tarvittavia tietoja Jalmarin tietokannasta. Työ jakautui neljään pääkategoriaan: perussuunnittelu, Step7 sovellussuunnittelu, WinCC-valvomosuunnittelu ja Jalmari-tietokanta.

Työssä esiteltiin menetelmiä, joilla pystytään generoimaan Jalmari-suunnittelujärjestelmästä ohjelmakoodia ja muuttujataulukkoja Step7-sovellukseen, sekä generoimaan WinCC:hen muuttujataulukkoja, tekstikirjastoja ja hälytyslistoja. Työssä luotiin myös esimerkki yhden moottoritoimilohkon sisäisestä toimintakuvauksesta ja määriteltiin automaatiosuunnittelun asettamat vaatimukset Jalmarin toiminnalle.

Avainsanat (asiasanat)
Step7, WinCC, Jalmari, generointi, tietokanta, toimilohkokirjasto, piirinäyttö

Muut tiedot

JYVÄSKYLÄN AMMATTIKORKEAKOULU
JAMK UNIVERSITY OF APPLIED SCIENCES

DESCRIPTION

Author(s)
KAUPPINEN, Petri

Type of publication
Master's Thesis

Date
30.4.2011

Pages
41

Language
FINNISH

Confidential

( ) Until

Permission for web publication
(x)

Title
DATABASE BASED AUTOMATION SOFTWARE ENGINEERING

Degree Programme
MASTER'S DEGREE PROGRAMME IN AUTOMATION ENGINEERING

Tutor(s)
SELOSMAA, Seppo
RANTAPUSKA, Seppo

Assigned by
RAMBOLL FINLAND OY
HAKKARAINEN, Aki

Abstract

Ramboll's automation and electrification unit is developing their own Jalmari engineering system to make electrification- instrumentation- and automation engineering more cost-effective and competitive. The objective of the thesis was to show the possibilities and ways to integrate automation software engineering as a part of Jalmari engineering system. The automation software used in the thesis were Siemens Step7 and Siemens WinCC.

The task of the thesis was to show the ways how to generate data from Jalmari engineering system to Step7 and WinCC. The study project was divided into four main categories: Basic design, Step7 software design, WinCC operation station design and Jalmari engineering system.

The thesis presents the methods how to generate Step7 program code and symbol tables and how to generate text libraries, alarm lists and tag lists to WinCC. The thesis includes an example of one motor control block's internal function description. The thesis defines the requirements from Jalmari engineering system for the integration with automation softare engineering.

Keywords
Step7, WinCC, Jalmari, database, Function block library, faceplate

Miscellaneous

SISÄLTÖ

KUVIOT

TAULUKOT

KÄSITTEET

I/O	Input/Output
FB	Function Block
FC	Function
DB	Data Block
LAD	Ladder logic, ohjelmointikieli
FBD	Function block diagram, ohjelmointikieli
STL	Structured text language, ohjelmointikieli

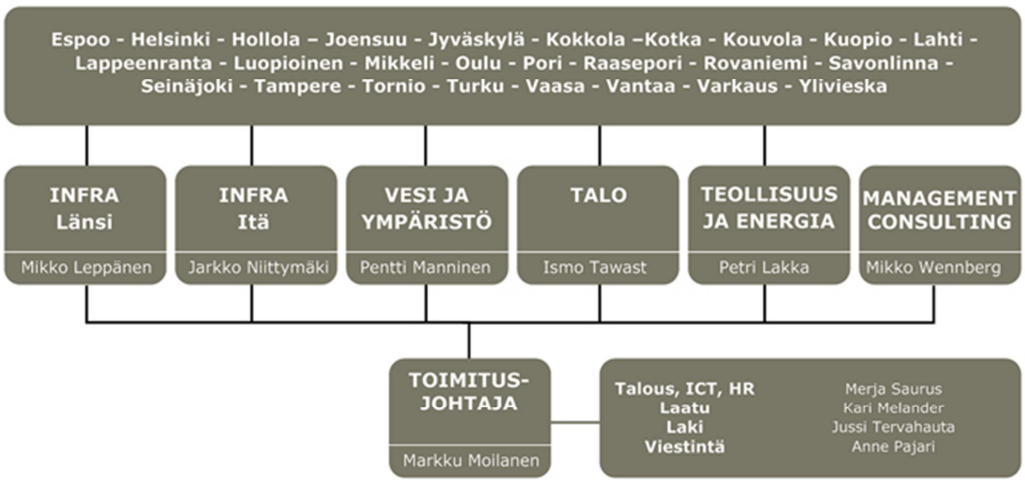
1 OPINNÄYTETYÖN LÄHTÖKOHDAT

Rambollissa tehdään kahdenlaisia projekteja. Pääsääntöisesti projektit ovat asiakasprojekteja, mutta ne voivat olla myös sisäisiä kehittämisprojekteja. Tässä opinnäytetyössä perehdytään sisäiseen kehitysprojektiin, jolla pyritään parantamaan automaatio- ja sähköyksikön kilpailukykyä. Työn tavoitteena on määritellä Jalmari-suunnittelujärjestelmän sovellussuunnitteluosio, jolla pyritään luomaan yhtenäinen sekä kustannustehokas suunnittelutapa toteuttaa logiikka- ja valvomoprojekteja Siemens Simatic-ympäristössä.

1.1 Toimeksiantaja

Ramboll Finland Oy

Ramboll Finland on osa kansainvälistä Ramboll Groupia. Ramboll on johtava kansainvälinen asiantuntijayritys, joka toimii yhteensä lähes 9000 asiantuntijan voimin ympäri maailmaa. Eurooppalaisten kotimarkkinoiden sekä Baltian alueen laajan toiminnan lisäksi Ramboll toimii merkittävästi kansainvälisillä markkinoilla mm. Venäjällä, Aasiassa, Afrikassa ja Etelä-Amerikassa. Ramboll tarjoaa suunnitteluun, rakentamiseen, tuotekehitykseen ja ylläpitoon liittyviä konsultti- ja asiantuntijapalveluita talo-, infra-, teollisuus-, energia- sekä vesi- ja ympäristötoimialoilla ja johdon konsultoinnissa. (Ramboll Finland Oy 2010a)



KUVIO 1. Ramboll Finland Oy:n organsiaatiokaavio

Automaatio ja sähkö

Rambollin laajaa SIA eli sähkö, instrumentointi ja automaatio -asiantuntemusta voidaan hyödyntää kokonais-EPCM -toimeksiannoissa tai se voidaan tarjota omana palvelunaan. Rambollin automaatio- ja sähköasiantuntijoilla on merkittävä kokemus lukuisista energiateollisuuden, metsäteollisuuden ja kemianteollisuuden prosesseista.

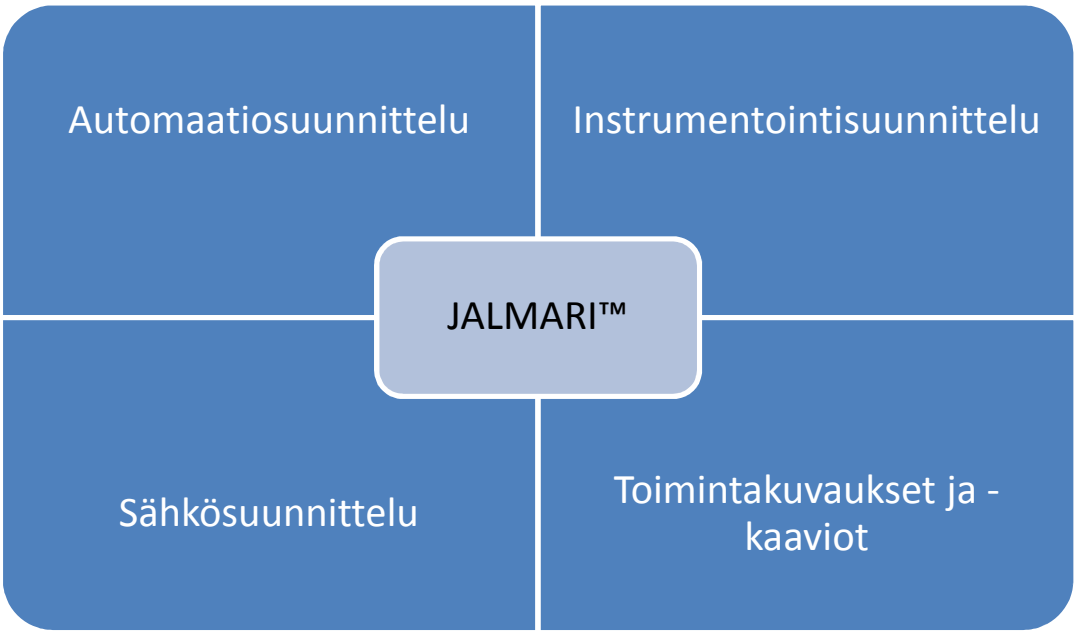
Automaation ja sähkön erityisosaamisalueet:

- konsultointi ja hankesuunnittelu
- esi- ja perussuunnittelu
- hankintasuunnittelu
- toteutussuunnittelu
- toimitus- ja asennusvalvonta
- testaus ja käyttöönotto
- koulutus

1.2 JALMARI Suunnittelujärjestelmä™

Ramboll Finlandin automaatio- ja sähköyksikkö kehittää omaa suunnittelujärjestelmää sähkö- instrumentointi- ja sovellussuunnittelua varten. Suunnittelujärjestelmän tarkoituksena on parantaa suunnittelun laatua ja kustannustehokkuutta. JALMARI-suunnittelujärjestelmä on jo otettu käyttöön instrumentointisuunnittelussa. Tämän työn tavoitteena on määritellä Jalmari-suunnittelujärjestelmän sovellussuunnitteluosio, jolla pyritään luomaan yhtenäinen sekä kustannustehokas suunnittelutapa toteuttaa logiikka- ja valvomoprojekteja Siemens-ympäristössä. Jalmarin sovellussuunnittelu-osiossa pyritään hyödyntämään mahdolisimman paljon instrumentointisuunnittelun jo ennestään Jalmariin syöttämää dataa, johon

lisätään sovellussuunnitteluun liittyvät vaatimukset. Jalmarin perusideana on keskittää suunnitteluprojektin kaikkien osa-alueiden tiedot ja dokumentit yhteen tietokantaan. Jalmari yhdistää sähkö-, automaatio- ja instrumentointisuunnittelun hallinnoimat tiedot yhteiseen tietokantaan, josta jokaisella osa-alueella on mahdollista käyttää tietoja hyväkseen. Tämä parantaa kustannustehokkuutta, koska joka suunnitteluryhmän ei tarvitse hallinnoida projektin dataa erikseen, vaan ne ovat yhteisesti käytössä. Tällaisia tietoja ovat esimerkiksi laitepositiot, piirinimet, piirien kuvaustekstit, laitetyypit, logiikan ohjelmaosoitteet, kaapelitunnukset jne.



KUVIO 2. JALMARI™ suunnittelujärjestelmän osa-alueet

Käytettäessä yhteistä tietokantaa on muutoksien tekeminen yksinkertaisempaa ja vähemmän riskialtista, koska muutokset tehdään vain yhteen paikkaan. Myös ylimääräisen massatyön määrä vähenee huomattavasti, koska useat työvaiheet voidaan automatisoida Jalmarin tehtäväksi. Tietokannan käyttäminen suunnitteluprojektien yhteydessä helpottaa huomattavasti suunnittelun eri vaiheita.

Tällä hetkellä Jalmarista saadaan jo generoitua useita eri automaatiosuunnittelun dokumentteja, joita ovat esimerkiksi

- laiteluettelot
- piirikaaviot
- ristikytkentäluettelot
- toimintakuvaukset
- kaapeliluettelot

2 PERUSSUUNNITTELU

2.1 Määrittely

Jotta suunnittelija pystyy käyttämään suunnittelujärjestelmää tehokkaasti, täytyy ohjelmalohkot olla dokumentoitu hyvin. Perussuunnittelu-osiossa ohjelmalohkoista luodaan toimintakuvaukset, joista selviää lohkojen sisäinen toiminta. Kuvauksessa määritellään lohkon tulot ja lähdöt ja niiden yksikkötyypit sekä lohkon toiminta. Ohjelmalohkoille määritellään yksilöllinen nimi, jonka avulla Jalmarissa valitaan, mitä toimilohkoa projektissa käytetään. Ohjelmalohkon kuvauksessa tulee pyrkiä niin yksityiskohtaiseen selostukseen, että toimilohkon toiminta ilmenee siitä ilman itse ohjelmakoodiin perehtymistä.

Erilaisia toimilohkoja voi olla useampia samalle laitetyypille. Esimerkiksi moottorilla voidaan tehdä useita erilaisia moottorilohkoja, joilla kaikilla on eri ominaisuudet. Toisaalta voidaan myös luoda vain yksi toimilohko toimilaitteelle, joka sisältää kaikki toiminnat ja ominaisuudet, joita kyseisessä toimilaitteessa voidaan tarvita. Jokaiselle ominaisuudelle on parametri, jolla se

voidaan ottaa käyttöön tai käytöstä pois. Tässä työssä on päädytty ensimmäiseen vaihtoehtoon käytännön osoittamista syistä. Tällä tavalla pystytään pitämään ohjelmakoodi mahdollisimman yksinkertaisena ja välttämään ylimääräistä koodia, jota ei mahdollisesti ikinä käytetä ohjelmassa.

2.2 Moottoritoimilohko (1-kela)

Moottoritoimilohkolla ohjataan 1-suuntaista ja 1-nopeuksista moottoria päälle tai pois, ja luetaan takaisinkytkentätietona moottorin käyntitieto. Takaisinkytkentätiedosta luodaan moottorin tilatieto, KÄY ja SEIS, sekä valvotaan ohjauksen toteutumista valvonta-ajan sisällä. Valvonta-aika on aseteltavissa valvomosta käynnistykselle ja pysäytykselle erikseen. Toimilohkossa on tulot turvakytkintiedolle ja moottorikeskusvika-tiedolle.

OHJAUSTILAT

Toimilohko toimii joko automaatti- tai käsiohjaustilassa. Valittu ohjaustila näytetään valvomon moottorisymbolissa sekä moottorin ohjausikkunassa. Käsikäyttötilassa moottoritoimilohkoa voidaan ohjata moottorin ohjausikkunasta. Automaattitilassa sovellusohjelma ohjaa moottoritoimilohkon tilaa. Paikalliskäyttöä varten toimilohko voidaan asettaa paikalliskäytölle, joka on tarkoitettu moottorin käsiajolle paikallisohjauskytkimillä. Paikallisohjaus voidaan asettaa sekä valvomosta että ohjelmalohkon rajapinnasta. Paikallisohjaus ohittaa automaatti- käsikäyttö- ja simulointitilat, sekä moottoritoimilohkoon kytketyt lukitukset. Tila osoitetaan sekä valvomon moottorisymbolissa, että ohjausikkunassa.

PAKKO-OHJAUKSET

Moottoritoimilohkossa on tulot pakko-ohjaus käyntiin ja pakko-ohjaus seis, joilla voidaan pakko-ohjata moottoritoimilohko haluttuun tilaan. Tuloihin voidaan myös liittää erilliset lukituslohkot, joihin yksittäiset lukitsevat tiedot kytketään. Jos molemmat tulot ovat yhtä aikaa päällä, ohjataan toimilohko seis-tilaan.

HÄLYTYKSET

Moottoritoimilohko menee häiriötilaan ja valvomoon annetaan hälytys jos moottorin tilatieto ei vastaa ohjauksen tilaa määritellyn viiveen kuluttua tai turvakytkintieto on aktiivinen tai moottorikeskusvika on aktiivinen. Näissä tilanteissa moottoritoimilohko myös pakko-ohjataan SEIS-tilaan. Moottorin häiriötila voidaan kuitata valvomosta joko moottorin ohjausikkunasta tai yleisestä häiriön kuittauspainikkeesta. Häiriö kuittaantuu vain, mikäli mikään häiriötieto ei ole enää aktiivinen.

SUUNNITTELUJÄRJESTELMÄSTÄ LINKITETTÄVÄT TIEDOT

Jalmarista tuotetaan logiikkaohjelmaan seuraavat tiedot. Moottorin takaisinkytkentä- ja ohjaussignaaleille I/O-osoitteet, positiotunnus, kommentti ja Simaticin toimilohkon (FB) ja datayksikön (DB) numerot.

- I/O-osoitteet
- Piirin positio ja nimi
- Toimilohkon numero
- Datayksikön numero

2.3 Moottoritoimilohkon liitynnät

TULOT

Man= [bool]

Valitsee toimilohkon ohjaustavaksi käsiajon.

Auto= [bool]

Valitsee toimilohkon ohjaustavaksi automaattiajon

Local= [bool]

Valitsee toimilohkon ohjauspaikaksi paikalliohjauksen. Toimilohkon lähtöä ohjataan tuloilla LocalStart ja LocalStop .

Remote= [bool]

Valitsee toimilohkon ohjauspaikaksi järjestelmän. Toimilohkon lähtöä ohjataan tuloilla AutoStart ja AutoStop, kun ohjaustapana on automaattiajo, tai käyttäjän toimesta valvomosta, kun ohjaustapana on manuaaliajo.

AutoStart= [bool]

Automaattiohjaus, moottorin käynnistyskäsky.

AutoStop= [bool]

Automaattiohjaus, moottorin pysäytyskäsky

LocalStart= [bool]

Paikallisohjaus, moottorin käynnistyskäsky

LocalStop= [bool]

Paikallisohjaus, moottorin pysäytyskäsky

MCC_OK= [bool]

Moottorinohjauskeskukselta tuleva ulkoinen tilatieto 1= Ei häiriötä, 0= Häiriö. 0-tila pakko-ohjaa moottorin seis ja kääntää ohjaustavaksi manuaaliohjauksen.

SSW_OK= [bool]

Kentältä tuleva moottorin turvakytkin-tieto. 1= Käynnistysvalmis, 0= Turvatila. 0-tila pakko-ohjaa moottorin seis ja kääntää ohjaustavaksi manuaaliohjauksen.

Run= [bool]

Moottorin käyntitieto kentältä.

Simulate= [bool]

Simuloinnin valinta. Kun simulointi on valittu, muodostetaan moottorin käyntitieto ohjauksen tilasta. Simulointi myös ohittaa MCC_OK ja SSW_OK –tiedot.

ForceON= [bool]

Pakko-ohjaus käyntiin. Vaikuttaa muista ohjauksista riippumatta, paitsi MCC_OK, SSW_OK ja ForceOFF. Vaikutus ohjaustapaan (Manual/Auto) riippuu parametrista FC_MA. Jos ForceON ja ForceOFF ovat yhtä aikaa vaikuttuneena, on ForceOFF voimakkaampi.

ForceOFF= [bool]

Pakko-ohjaus seis. Vaikuttaa muista ohjauksista riippumatta. Vaikutus ohjaustapaan (Manual/Auto) riippuu parametrista FC_MA.

FC_MA= [bool]

Valinta pakko-ohjausten vaikutukselle moottorin ohjaustapaan (Manual/Auto). 0= Pakko-ohjaus ei vaikuta ohjaustapaan. 1= Moottori pakotetaan käsiajolle, sillä suorituskierroksella, jolla

pakko-ohjaus on tullut voimaan, jos pakko-ohjaus on vaikuttanut moottorin tilaan.

Kuittaus= [bool]

Häiriötiedon kuittaus.

LÄHDÖT

ON= [bool]

kentälle vietävä moottorin ohjaustieto, joka suoraan ohjaa moottoria tilaan käy tai seis.

MA= [bool]

Auto / Käsi tilatieto. Lähtö on päällä kun moottori on automaattiohjauksella.

FON= [bool]

Pakko-ohjaus päälle on aktiivinen

FOFF= [bool]

Pakko-ohjaus seis on aktiivinen

ALARM= [bool]

Moottori on häiriössä. Moottorin tilatieto ei vastaa ohjauksen tilaa määritellyn viiveen kuluttua tai turvakytkintieto on aktiivinen tai moottorikeskusvika on aktiivinen.

ACK= [bool]

Häiriön kuittauskäsky valvomosta, tai KUITTAUS-tulo on aktiivinen. Päällä yhden ohjelmakierron. Voidaan käyttää esim. lukituslohkon häiriösiepparin nollaukseen.

VALVOMOLIITYNTÄ

VIIVE1= [int]

Moottorin käynnistysviive, jonka jälkeen annetaan häiriöilmoitus ”Moottorin tilatieto ei vastaa ohjauksen tilaa”

VIIVE2= [int]

Moottorin pysäytysviive, jonka jälkeen annetaan häiriöilmoitus ”Moottorin tilatieto ei vastaa ohjauksen tilaa”

OHJAUSSANA [word]

bit0= [bool]

Häiriön kuittaus.

bit1= [bool]

Valitsee toimilohkon ohjaustavaksi automaattiajon

bit2= [bool]

Valitsee toimilohkon ohjaustavaksi käsiajon.

bit3= [bool]

Valitsee toimilohkon ohjaustavaksi paikallisohjauksen.

bit4= [bool]

Moottorin käynnistyskäsky

bit5= [bool]

Moottorin pysäytyskäsky

bit6 – 15= [bool]

Varalla

TILASANA [word]

bit0= [bool]

Moottorin käyntitieto

bit1= [bool]

Moottorinsuojakytkin. . 1= Käynnistysvalmis, 0= Turvatila.

bit2= [bool

Moottorikeskusvika. 1= Ei häiriötä, 0= Häiriö

bit3= [bool]

Pakko-ohjaus päälle aktiivinen

bit4= [bool]

Pakko-ohjaus seis aktiivinen

bit5= [bool]

Auto / Käsi tilatieto. 1= Automaatilla, 0= Käsiohjauksella

bit6= [bool]

Paikallisohjauksella

bit7= [bool]

Simulointi aktiivinen

bit8= [bool]

Moottorin käynnistyskäsky on aktiivinen

bit9 – 15= [bool]

Varalla

3 STEP7 SOVELLUSSUUNNITTELU

Step7 ohjelmistolla voidaan hallita kaikki tärkeimmät automaatioprojektin vaiheet, joihin kuuluu järjestelmän konfigurointi, ohjelmointi sekä testaaminen. Step7:ää ohjelmoidaan käyttämällä standardin IEC 1561131-3 –mukaisia ohjelmointikieliä. Standardi sisältää grafiikkapohjaiset ja helppolukuiset ohjelmointikielet, kuten LAD (Ladder) ja FBD (Function Block Diagram) sekä tekstipohjaisen, mutta hieman laajemmat ohjelmointimahdollisuudet omaavan STL (Structured Text Language). Ohjelmistossa on valmiina binääriset logiikkatoiminnot sekä kattavan kokoelman mm. erilaisia ajastimia, laskureita, vertailu- ja muunnostoimintoja sekä matemaatisia funktioita. Lisäksi Step7:llä voidaan luoda sekvenssiohjauksia S7 Graph -ohjelman avulla. (Siemens AG 2010a.)

3.1 Toimilohkokirjasto

Toimilohkokirjasto sisältää mallipohjat yleisimmin käytettävien toimilaitteiden ohjelmalohkoista, joita kutsutaan Tyyppilohkoiksi. Toimilohko on valmis logiikkasovelluksen osa, joka suorittaa tietyn toiminnon, esimerkiksi moottorilähdön ohjaamisen ja valvonnan tai säätöalgoritmin. Toimilohkoa voidaan kutsua useaan kertaan pääohjelmasta tai muista toimilohkoista. Ohjelmoinnissa esiintyy usein tilanne, että samanlaista toimintoa tarvitaan ohjelman useassa eri kohdassa. Jos sama toiminto kirjoitettaisiin ohjelmaan yhä uudelleen, ohjelmakoodi pitenisi ja tulisi epäselväksi, virheiden mahdollisuus kasvaisi ja korjaukset ohjelmakoodiin pitäisi tehdä useaan paikkaan. Toimilohkokirjaston käyttäminen jokaisen suunnittelijan vapaamuotoisten ohjelmointiratkaisujen sijasta, vähentää yrityksen sisäistä henkilöriippuvuutta ja parantaa osaltaan logiikkasovellusten laatua. Toimilohkojen ollessa helposti uudelleen käytettävissä nopeutuu logiikkasovelluksen ohjelmointi.Tyyppilohko ohjelmoidaan kerran jonka jälkeen sitä voidaan käyttää useita kertoja eri projekteissa. Näin voidaan vähentää ohjelmointiin ja testaukseen kuluvaa aikaa. Toimilohkokirjaston kasvaessa ohjelmointi tehostuu merkittävästi, koska testattua ohjelmakoodia on helposti saatavana kirjastosta. Tilanteessa jossa toimilohkon toimintaa on tarve muuttaa, riittää kun muutoksen tekee toimilohkokirjaston tyyppilohkoon. Kaikki ohjelmassa käytetyt kyseisen tyypin toimilohkot muuttuvat automaattisesti, koska ne ovat tyyppilohkon instansseja.

3.2 Lähdekoodipohjat

Toimilohkot luodaan Simatic Managerissa. Toimilohkon luonti sisältää kolme vaihetta: muuttujien määrittely, ohjelmakoodin luominen ja toimilohkon ominaisuuksien määrittely. Kun peruspiiri on luotu, testataan piirin kaikki toiminnallisuudet. Piirin ollessa toiminnallisuudeltaan haluttu, käännetään toimilohkon ohjalmakoodi lähdekoodimuotoon. Lähdekoodi on täysin tekstimuotoista ja sitä voidaan editoida millä tahansa tekstieditorilla. Liitteessä 2. on esimerkki moottoritoimilohkon lähdekoodipohjasta.

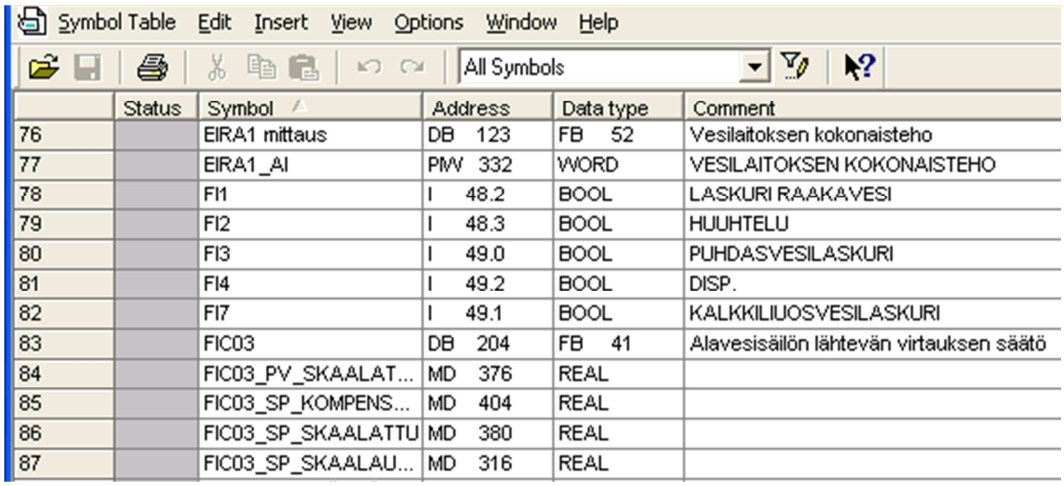
Lähdekoodit tallennetaan Jalmariin lähdekoodipohjiksi. Ennen Jalmariin tallentamista, lähdekoodia muokataan siten, että niihin kohtiin lähdekoodia, joihin halutaan Jalmarin generoivan tietoa, vaihdetaan nykyisen tekstin tilalle syntaksiparametri. Parametri on sellainen merkkijono, jonka Jalmari tunnistaa. Tässä esimerkissä käytetään muotoa ”#TUNNUS”. Oheisessa taulukossa on esitetty moottoritoimilohkon Jalmarista generoitavat muuttujat. Nämä tiedot ovat sellaisia, jotka Jalmarille on annettava, ennekuin se pystyy generoimaan S7-ohjelmalohkon uudelle moottorille. Liitteessä 2. on esimerkki lähdekoodista, johon on lisätty Jalmarin käyttämät syntaksiparametrit.

TAULUKKO 1. Moottorin lähdekoodipohjan generoitavat muuttujat

Tunnus	Oletus	Kuvaus	Tyyppi
#DB_NUM		Moottorin tiedostoyksikön numero	INT
#DB_TITLE_TXT		Tiedostoyksikön kuvaus	STRING
#AUTHOR		Suunnittelijan nimi	STRING
#FC_NUM		Funktion numero	INT
#TITLE_TXT		Otsikko	STRING
#COMMENT_TXT		Kuvaus	STRING
#MOTOR_TXT		Virtapiirin otsikko	STRING
#RUN		Käyntitiedon osoite	S7-OSOITE
#SSW_OK		Turvakytkintiedon osoite	S7-OSOITE
#MCC_OK		Moottoriohj.keskushäiriön osoite	S7-OSOITE
#OUTPUT		Lähdön osoite	S7-OSOITE

3.3 Symbolitaulukko

Simatic Step7 –ohjelmointiympäristössä voidaan jokaiselle ohjelmassa käytettävälle muuttujalle antaa symbolinen nimi. Ohjelma on helppolukuisempaa ja paremmin ymmärrettävää, kun muuttujilla on kuvaavat nimet ja kommentit. Symbolitaulukko sisältää muuttujan symbolisen nimen, absoluuttisen osoitteen, muuttujan tyypin sekä kommenttikentän. Symbolitaulukko on mahdollista tuoda Step7-projektiin tekstitiedostosta, joka on .dif-formaatissa.



	Status	Symbol	Address	Data type	Comment
76		EIRA1 mittaus	DB 123	FB 52	Vesilaitoksen kokonaisteho
77		EIRA1_AI	PIW 332	WORD	VESILAITOKSEN KOKONAISTEHO
78		FI1	I 48.2	BOOL	LASKURI RAAKAVESI
79		FI2	I 48.3	BOOL	HUUHTELU
80		FI3	I 49.0	BOOL	PUHDASVESILASKURI
81		FI4	I 49.2	BOOL	DISP.
82		FI7	I 49.1	BOOL	KALKKILIUOSVESILASKURI
83		FIC03	DB 204	FB 41	Alavesisäilön lähtevän virtauksen säätö
84		FIC03_PV_SKAALAT...	MD 376	REAL	
85		FIC03_SP_KOMPENS...	MD 404	REAL	
86		FIC03_SP_SKAALATTU	MD 380	REAL	
87		FIC03_SP_SKAALAU...	MD 316	REAL	

KUVIO 3. Symbolitaulukko

4 WINCC-VALVOMO

4.1 Määrittely

Simatic WinCC on tehokas valvomo-ohjelmisto erilaisten automaatioprosessien valvontaan ja hallintaan. SIMATIC WinCC on monipuolisesti laajennettavissa oleva ohjelmisto, johon voidaan liittää tehtaanlaajuinen tietokoneverkosto ja sen antamaa tietoa voidaan tarkkailla myös web – selaimella (Siemens AG, 2010b).

WinCC-lisenssien hinnoittelu perustuu liityntämuuttujien, tagien, määrään. WinCC käyttää tageja kommunikoinnissa automaatiojärjestelmän kanssa. Tagi voi olla kooltaan 1 bitistä 64:een bittiin. Mitä enemmän tietoa tuodaan yhdessä tagissa, sitä pienemmällä tagimäärällä pystytään kommunikointi automaatiojärjestelmän kanssa totetuttamaan. Tästä syystä on järkevää

pakata automaatiojärjestelmässä yhden toimilohkon bittikohtaiset ohjaus- ja tilatiedot esimerkiksi yhteen ohjaussanaan ja yhteen tilasanaan. WinCC:ssä ei täten tarvitse luoda kuin minimissään 2 tagia yhdelle toimilohkolle. Tagien säästämisessä voidaan mennä vielä tästäkin pidemmälle, käyttämällä epäsuoraa osoitusta. Tällöin WinCC:hen luodaan vain yksi ohjaus- ja yksi tilasana kullekkin toimilohkotyypille. Esim. Moottorit, venttiilit, mittaukset, jne. Tällöin logiikan ohjelmassa siirretään esimerkiksi kaikkien moottorilohkojen tiedot valvomoon käyttäen vain yhtä ohjaus- ja yhtä tilasana. Moottorilohkot indeksoidaan ja WinCC:hen kerrotaan indeksinumerolla, minkä moottorilohkon tietoja ollaan seuraavaksi lähettämässä. Käytännön osoittamista syistä tätä tapaa ei kuitenkaan ole syytä käyttää. Kyseinen tapa monimutkaistaa järjestelmän toimintaa liiaksi, ja vaikeuttaa vianetsintää ja muokattavuutta.

WinCC:ssä käytettävät tagit, hälytystekstit, tag logging –listat ja tekstikirjastot voidaan tuoda WinCC:hen excel-muotoisina taulukkoina Siemensin tarjoamalla Configuration Tool –työkalulla.

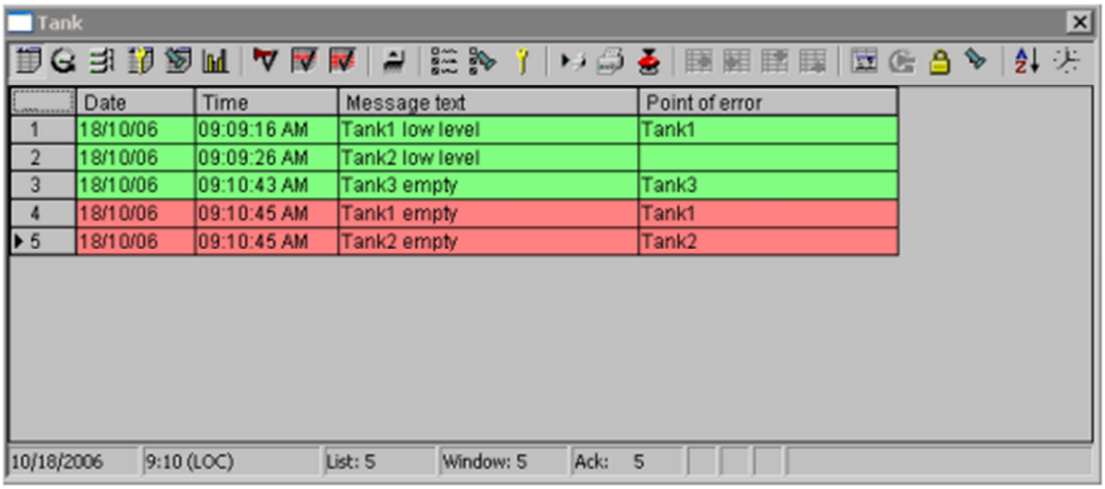
4.2 Hälytyslista

Hälytystekstit ilmoittavat operaattorille prosessin tilasta ja häiriöistä valvomon hälytysnäytöllä. Hälytykset kerätään hälytysnäytöllä yhteen taulukkoon, jossa yksi hälytystapahtuma näytetään yhdellä taulukon rivillä. Hälytysrivi koostuu kolmesta eri tietolohkosta:

- Järjestelmälohko, jossa näytetään esimerkiksi kellonaika, päivämäärä, hälytysnumero ja hälytyksen tila

- Prosessilohko, jossa näytetään kyseiseen hälytykseen liittyvä prosessin arvo, esimerkiksi paine, lämpötila, tms.

- Tekstilohko, jossa näytetään hälytystä kuvaava teksti, esimerkiksi hälytyksen syy ja viallisen laitteen tunnus.



KUVIO 4. Hälytyslista

Hälytysteksteille voidaan tehdä useita eri luokkia esimerkiksi hälytyksen kriittisyyden mukaan. Hälytystekstit generoidaan Jalmarista .CSV –muotoiseen tiedostoon, joka voidaan importoida WinCC:hen. Liitteessä 1. on esiteltynä CSV-tiedoston rakenne.

4.3 Prosessimuuttujien arkistointi

WinCC:ssä on arkistointijärjestelmä, jolla voidaan tallentaa haluttujen prosessimuuttujien arvot tietokantaan myöhempää käyttöä varten. Tällaisia voivat olla esimerkiksi prosessimuuttujien arvojen näyttäminen historiatrendeinä, tai arvojen käyttäminen tuotantoraporttien luonnissa. Prosessimuuttujille voidaan määritellä haluttu tallennustapa. Mahdollisia tapoja ovat:

- Syklinen tallennus: Prosessimuuttujan arvo tallennetaan tietokantaan asetetun tallennussyklin mukaan jatkuvasti.

- Valinnainen-jatkuva tallennus: Prosessimuuttujan arvo tallennetaan tietokantaan jatkuvasti, tiettyjen ehtojen täytyttyä. Esim. tietty aikaväli.

- Asyklinen tallennus: Prosessimuuttujan arvo tallennetaan tietokantaan vain kun tietty ehto täyttyy. Esim. tallennus, kun hälytysraja ylittyy.

- Muutoskohtainen tallennus: Prosessimuuttujan arvo tallennetaan tietokantaan vain kun muuttujan arvo vaihtuu.

Pääsääntöisesti prosessimuuttujista arkistoidaan analogiamittaukset. Tällöin valvomonäytölle voidaan tehdä jokaiselle mittaustiedolle trendi-ikkuna, josta voidaan katsoa mittauksen arvoja myös historiasta. Historiatiedon näkeminen auttaa useasti operaattoreita löytämään häiriöiden syitä ja joskus myös havaitsemaan tulevat häiriöt etukäteen. Arkistoitavat prosessimuuttujat voidaan tuoda WinCC-projektiin .csv-tiedostona. Jalmarissa on oltava mahdollisuus valita, mitkä tagit halutaan arkistoida. Tämän tiedon pohjalta Jalmari luo .csv-tiedoston. Tiedoston rakenteesta ei löydy tietoa Siemensin manuaaleista. Yksi mahdollisuus selvittää .csv-tiedoston rakenne on luoda WinCC:n TagManager-editorilla yksi kutakin arkistoitavaa tagityyppiä ja exportata konfiguraatiosta .csv-tiedosto. Vertaamalla TagManageria ja excelissä avattua exportattua tiedostoa, voidaan selvittää eri sarakkeiden käyttötarkoitus.

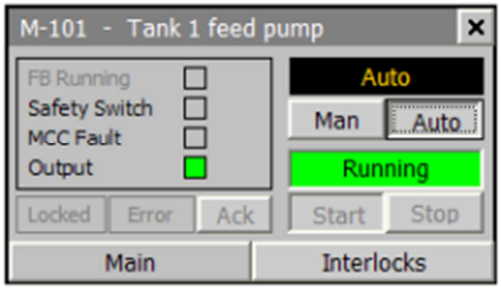
4.4 Symboli- ja piirinäyttökirjasto

WinCC-valvomoon luodaan symboliobjekti ja piirinäyttö jokaiselle S7-toimilohkolle, jotka on luotu Jalmarin käyttöön. Symboleista ja piirinäytöistä luodaan tyyppikirjasto, joka sisällytetään jokaiseen projektiin. Tyyppikirjasto on siis kokoelma mallipohjia, joista kopioidaan projektissa käytettävät symboliobjektit ja piirinäytöt.

Symboliobjektit ovat valvomon ajonäytöillä näytettäviä animoituja kuvia, jotka näyttävät toimilaitteen tilan. Esim. pumpun symboliobjekti on vihreä, kun pumppu käy ja harmaa kun pumppu on pysähdyksissä jne.

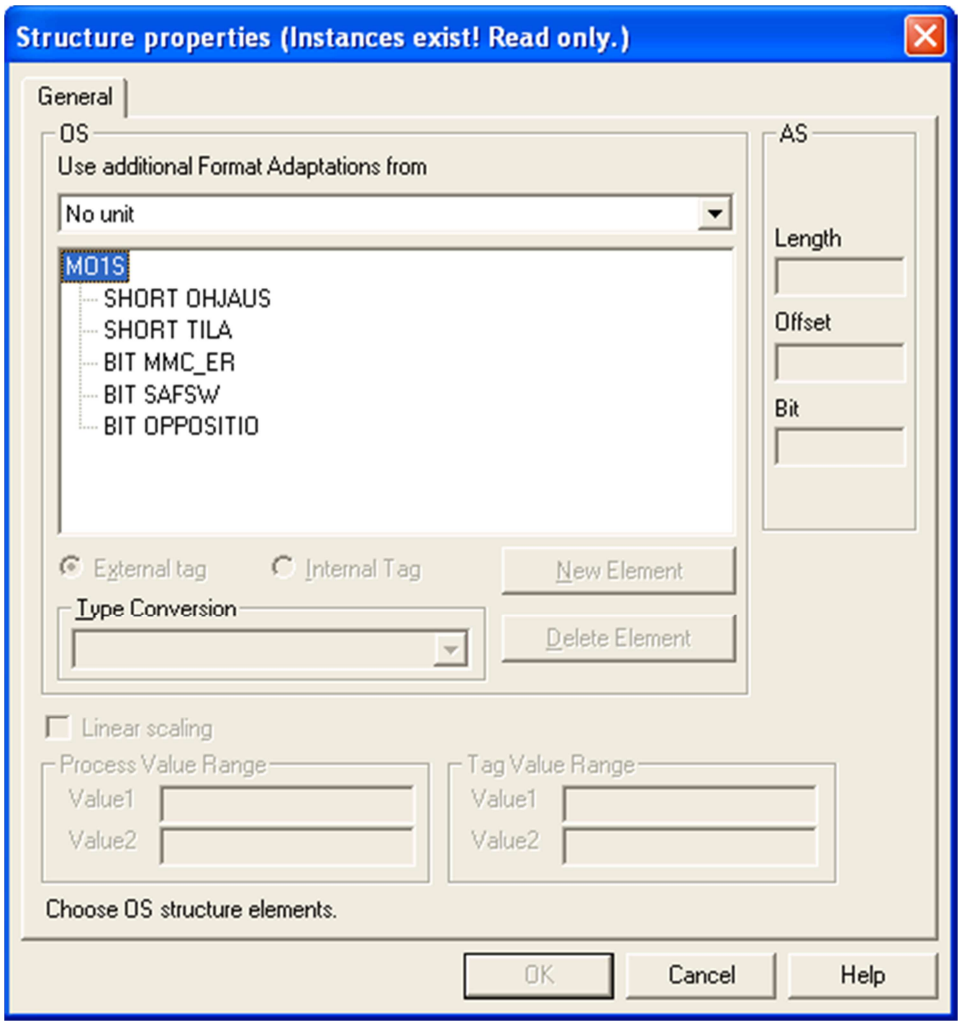
KUVIO 5. Moottorin symboliobjekti

Piirinäyttö on valvomosovelluksen sivu, johon on koottu toimilaitteen valvontaan ja operointiin tarvittavat tiedot ja kontrollit. Piirinäytössä voi olla myös useita sivuja, joissa näkyy esimerkiksi toimilaitteen lukitustiedot, toimilaitteen toimintaan vaikuttavat parametrit jne.



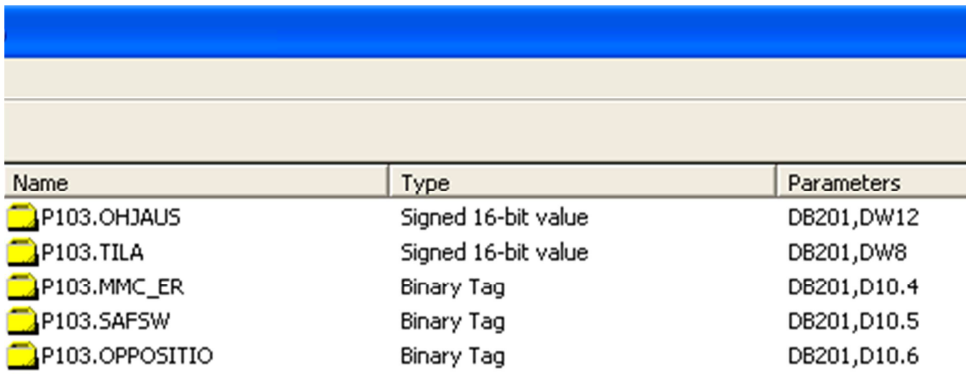
KUVIO 6. Moottorin piirinäyttö

Valvomosovelluksessa käytettävät piirinäytöt ovat piirinäyttötyypin instansseja. Yksittäiselle toimilaitteelle ei siis luoda omaa piirinäyttösivua, vaan näyttötyyppiä käytetään dynaamisesti kaikille toimilaitteille. WinCC versio 7:stä löytyy työkalu, jolla piirinäyttötyyppejä on helppo luoda. Työkalu on faceplate designer, ja se kuuluu WinCC:n perustyökaluihin. Piirinäyttöjen dynaamisuus on helpompi rakentaa, kun tagien luomisessa käytetään struktuuritageja.



KUVIO 7. Esimerkki struktuuritagin rakenteesta

Struktuuritagi on tietue, joka sisältää useita yksittäisiä tageja. Jokaiselle eri toimilaitteelle luodaan oma struktuuritagiryhmä, joka sisältää kaikki toimilohkon tarvitsemat tiedot kommunikoidakseen WinCC:n ja logiikan välillä. Struktuuritagin rajoituksena on, että sen sisältämät yksittäiset tagit pitää olla samana S7:n datablockin sisällä. Tietueryhmään sisällä oleville yksittäisille tageille annetaan kuvaava nimi ja osoite. Luotaessa uusi struktuuritagi, annetaan tagille haluttu etuliite ja S7:n datablockin numero. Luotavan struktuuritagin muoto on ”etuliite.yksittäisen tagin nimi”.



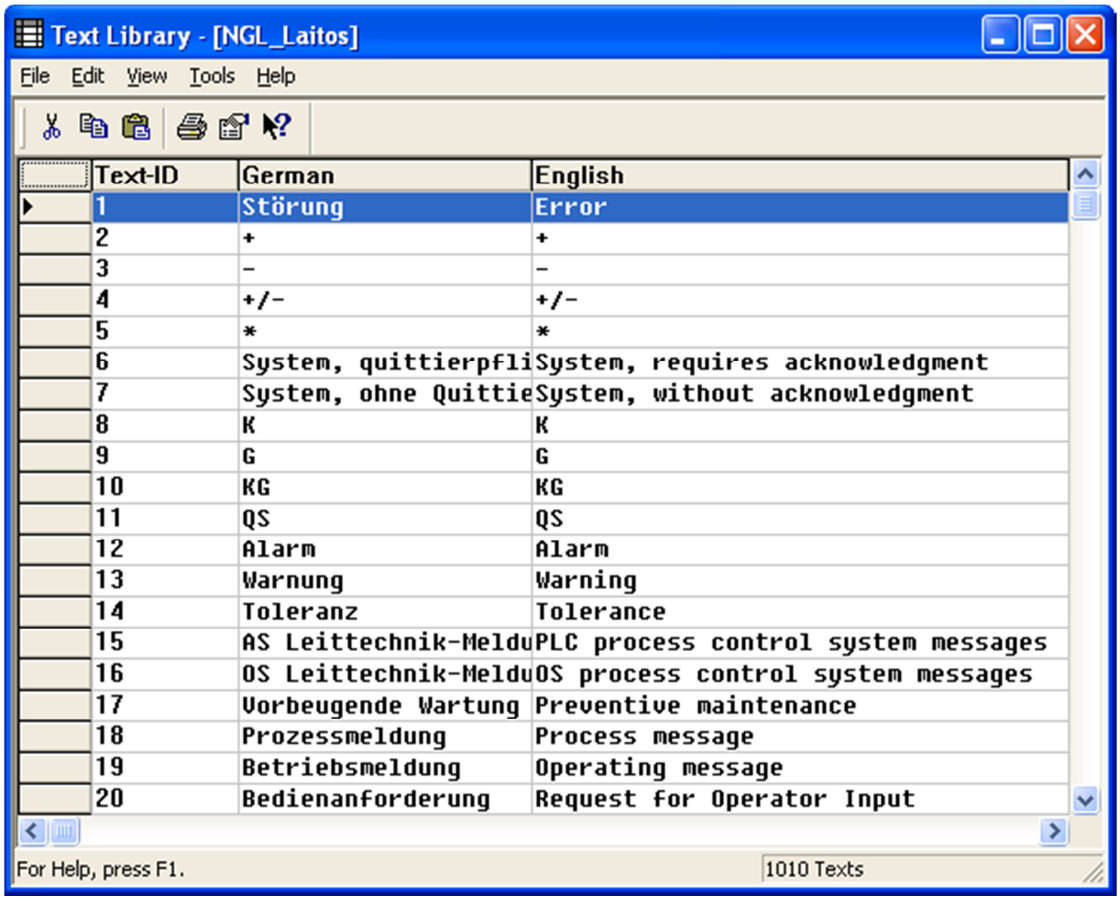
Name	Type	Parameters
P103.OHJAUS	Signed 16-bit value	DB201,DW12
P103.TILA	Signed 16-bit value	DB201,DW8
P103.MMC_ER	Binary Tag	DB201,D10.4
P103.SAFSW	Binary Tag	DB201,D10.5
P103.OPPOSITIO	Binary Tag	DB201,D10.6

KUVIO 8. Esimerkki struktuuritagin muodosta

Etuliitteeksi kannattaa valita kuvaava nimi, kuten esimerkiksi laitteen positio. Tällöin esimerkiksi moottorin ”P103” käyntitiedon tagi voisi olla ”M103.Running”, jossa P103 on struktuuritagin etuliite ja Running on struktuurin yksittäisen tagin nimi. Piirinäytön mallipohjaa luotaessa määritellään piirinäytön sisällä käytettävien tagien osoitteeksi struktuuritagin loppuosa. Esimerkiksi piirinäytöllä näkyvän moottorisymbolin animoinnissa käytettävän käyntitiedon tagiksi määritellään ainoastaan ”Running”. Struktuuritagin etuosa on muuttuva tieto, joka riippuu kyseisen toimilaitteen positiosta. Sitä kutsutaan WinCC:ssä sanalla ”prefix”. Prefix on yksi parametri piirinäytön ominaisuuksissa. Määrittelemällä tähän parametriin struktuuritagin alkuosa, esimerkiksi ”M103.”, tulee kaikkien piirinäytössä käytettävien tagien eteen kyseinen alkuosa. Esimerkiksi ”M103.Running”. Struktuuritageja käyttämällä piirinäytön luominen piirinäyttökirjastosta uudelle toimilaitteelle on helppoa. Ei tarvitse kuin struktuuritagin prefix-parametrin määrittelyn, jonka jälkeen kaikki piirinäytön tagikytkennät ovat valmiit.

4.5 Tekstikirjasto

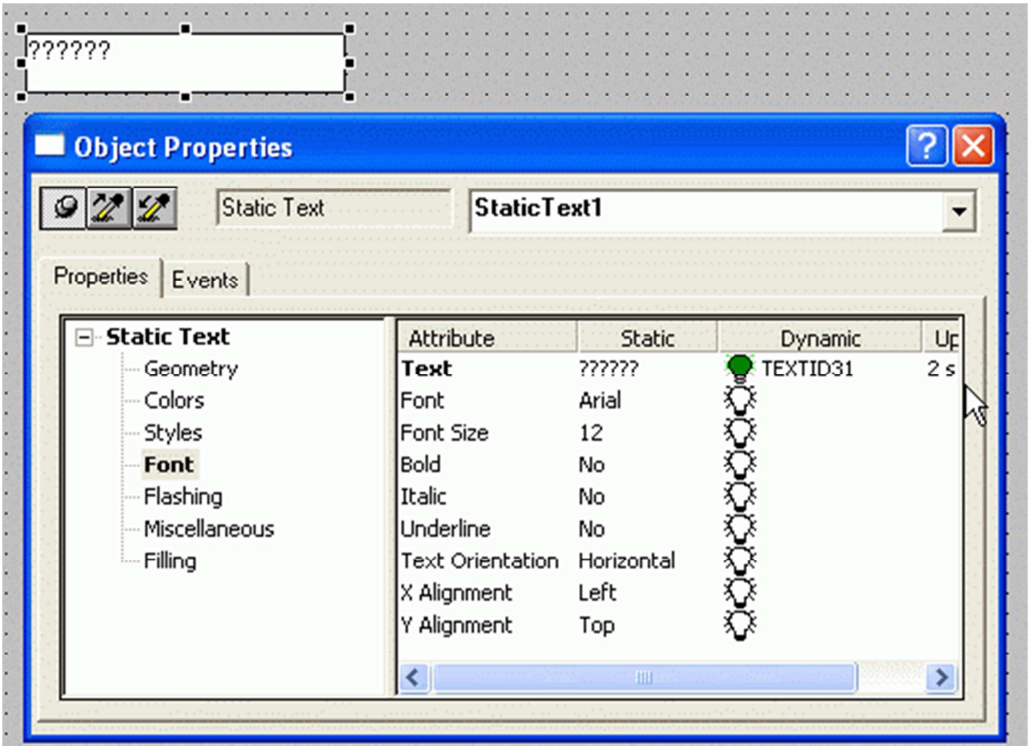
WinCC:n yksi ominaisuus on tekstikirjasto. WinCC tallentaa tekstikirjastoon kaikki projektin tekstit, lukuunottamatta operointisivuilla olevia staattisia tekstejä. Kirjastoon voidaan myös itse lisätä rivejä ja käyttää riveille kirjoitettuja tekstejä valvomosovelluksessa. Tekstikirjaston jokainen rivi saa oman teksti-ID:n. Teksti-ID:tä ei voi itse määritellä, vaan WinCC määrittelee ID:n juoksevana numerona. Kirjaston teksteihin voidaan viitata projektissa teksti-ID:llä. Tekstikirjaston pääasiallinen tarkoitus on monikielisyys. Kirjastoon voidaan määritellä useita kieliä, jolloin kirjastoon luodaan yksi sarake kullekin kielelle. WinCC:ssä on valmis toimintonappi, jolla voidaan vaihtaa kieltä vaikka valvomosovellus olisi ajossa.



Text-ID	German	English
1	Störung	Error
2	+	+
3	-	-
4	+/-	+/-
5	*	*
6	System, quittierpfli	System, requires acknowledgment
7	System, ohne Quittie	System, without acknowledgment
8	K	K
9	G	G
10	KG	KG
11	QS	QS
12	Alarm	Alarm
13	Warnung	Warning
14	Toleranz	Tolerance
15	AS Leittechnik-Meldu	PLC process control system messages
16	OS Leittechnik-Meldu	OS process control system messages
17	Vorbeugende Wartung	Preventive maintenance
18	Prozessmeldung	Process message
19	Betriebsmeldung	Operating message
20	Bedienanforderung	Request for Operator Input

KUVIO 9. Tekstikirjasto

Monikielisyyden lisäksi tekstikirjastoa voidaan käyttää hyväksi operointisivuilla näytettävissä teksteissä. Keräämällä kaikki operointinäyttöjen tekstit tekstikirjastoon on niiden hallinnoiminen helpompaa. Esimerkiksi moottorin faceplatessa näkyvät moottorin lukitustekstit on järkevää toteuttaa tekstikirjaston avulla. Tekstikirjaston sisältöä voidaan käyttää WinCC-projektissa sisäisen tekstireferenssi-tagin avulla. Referenssitagille annetaan kiinteänä arvona teksti-ID:n numero, johon halutaan viitata. Käyttämällä kyseistä sisäistä muuttujaa operointinäytöissä, voidaan näytöllä esittää tekstireferenssi-tagin viittaman tekstikirjaston rivin teksti. Kuviossa 3. on esimerkki, kuinka ”TEXTID31”-tagia käytetään operointinäytöllä.



KUVIO 10. Tekstikirjaston tekstin käyttäminen WinCC-näytöllä

WinCC:n tapa määritellä teksti-ID juoksevana numerona luo ongelmaksi sen, että emme voi itse määrätä tiettyä teksti-ID:tä tietylle tekstille. Ongelma pystytään kiertämään varaamalla tekstikirjastosta tarpeeksi iso alue, johon WinCC ei tule automaattisesti koskaan kirjoittamaan. Alueen varaamisessa käytetään hyväksi WinCC:n tapaa valita seuraava vapaa teksti-ID numero. WinCC katsoo pelkästään tekstikirjaston suurimman teksti-ID numeron ja luo

uudelle tekstille seuraavan vapaan numeron. WinCC ei siis tarkista onko ID-numeroiden välissä tyhjää. Rajoitus sille, että emme itse pysty määrittelemään teksti-ID numeroa, on voimassa vain WinCC:n tekstikirjasto-editorissa. Tekstikirjasto voidaan importata .csv-tiedostosta, jolloin tekstit on myös mahdollista kirjoittaa toisessa sovelluksessa ja tuoda sitten WinCC:hen. Muokkaamalla tekstikirjastoa WinCC:n ulkopuolella, voimme määritellä teksti-ID:t haluamaksemme. Tekstikirjastosta voidaan varata tietty alue omaan käyttöön tekemällä kirjastoon yksi rivi, jolle annetaan halutun suuri arvo. Esimerkiksi jos halutaan varata 10000 riviä omaan käyttöön, tehdään uusi rivi, jonka teksti-ID:ksi annetaan 10000 suurempi, kuin kirjaston viimeisen teksti-ID:n arvo. Näin meille jää 10000 riviä tyhjää tilaa, johon WinCC ei tule automaattisesti lisäämään rivejä. Tätä aluetta voimme käyttää vapaasti haluamallamme tavalla.

Moottoreiden, venttilileiden ja muiden laitteiden lukitustekstit on tarkoitus tuoda Jalmarista. Jos Jalmariin on tehty prosessin toimintakuvaus, voidaan lukitustekstit poimia toimintakuvauksesta suoraan ja generoida niistä WinCC:n tekstikirjaston tekstit. Näitä tekstejä voidaan käyttää suoraan lukitusteksteinä toimilohkon operointinäytöissä. Lukitustietojen määrä on järkevää ryhmittää 16 kappaleen sarjoihin. Tällöin yhden lukitusryhmän lukitustiedot pystytään siirtämään logiikasta valvomoon yhdessä datasanassa. Jokaiselle lukitustiedolle täytyy myös varata yksi rivi WinCC:n tekstikirjastosta, eli yhdelle 16:n lukitustiedon ryhmälle 16 tekstiriviä. Määrittelemällä näille 16:n lukitustiedon ryhmille indeksinumerot, pystymme yhdistämään oikeat tekstikirjaston rivit oikealle laitteelle. Määritellään tekstikirjastosta lukitustekstien alueeksi rivit 0..10000. Tälle alueelle mahtuu 245:n lukitusryhmän lukitustekstit. Määritellään lukitusryhmien indeksinumerot alkamaan numerosta 0. Näin indeksinumero 0 vastaa tekstikirjaston rivejä 0..15. Indeksinumero 1 vastaa rivejä 16..31. Generoitaessa Jalmarissa toimilaitteille lukitustoimilohkoja S7-ohjelmaan, määritellään samalla jokaiselle lukituslohkolle indeksinumero. Viemällä tämä indeksinumero logiikasta WinCC:hen yhtenä tietona lukitustietojen lisäksi, voimme yhdistää kyseiseen lukitustoimilohkoon oikeat lukitustekstit. Indeksinumeron avulla pystymme

yhdistämään oikeat lukitustekstit oikealle toimilohkolle kaavalla: Kyseisen toimilohkon ensimmäinen rivi tekstikirjastossa = (lukituslohkon indeksinumero * 16).

5 JALMARI TIETOKANTA

Jalmari yhdistää sähkö-, automaatio- ja instrumentointisuunnittelun hallinnoimat tiedot yhteiseen tietokantaan, josta jokaisella osa-alueella on mahdollista käyttää tietokannan tietoja hyväkseen. Tietokannan käyttäminen suunnitteluprojektien yhteydessä helpottaa huomattavasti suunnittelun eri vaiheita. Erityisesti isot projektit sisältävät paljon tietoa, jota pystytään tietokannan avulla hallitsemaan tehokkaasti ja järkevästi.

5.1 Jalmari sovellussuunnittelun apuna

5.2 Step7

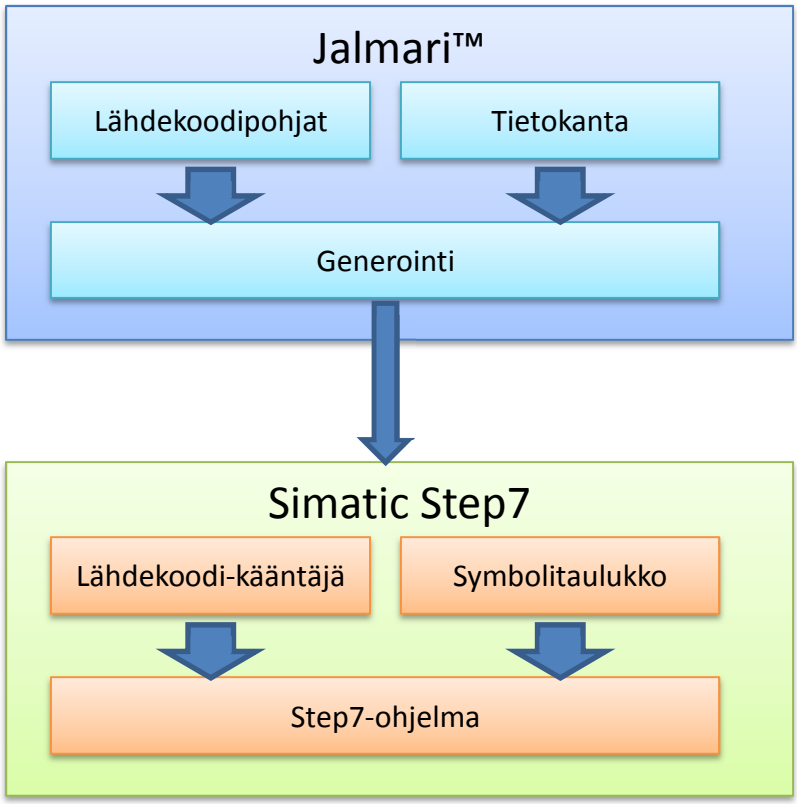
Jalmarista tulee löytyä Step7:ssä käytettäville peruspiireille kappaleessa 3.2 esitetyn mukaiset lähdekoodipohjat. Lähdekoodipohjat sisältävät syntaksiparametreja, joihin Jalmari lisää tarvittavat tiedot. Seuraavassa listassa on esitetty ne tiedot, joita kaikki Step7-lähdekoodipohjat tarvitsevat:

- Ohjelman käyttämät IO-osoitteet
- IO-osoitteiden symboliset nimet ja niiden kuvaustekstit
- Toimilohkojen käyttämien funktioiden numerot ja symboliset nimet
- S7-funktioiden tiedostoyksiköiden numerot ja symboliset nimet

- Toimilaitteiden piirinimet ja niiden kuvaustekstit

Jokaisella peruspiirillä voi näiden lisäksi olla omia eritystarpeita, jotka on otettava huomioon tehtäessä kyseiselle piirille lähdekoodipohjaa Jalmariin.

Monet listassa esitetyistä tiedoista ovat sellaisia, jotka on yleensä jo syötetty Jalmariin joko instrumentointisuunnittelussa, tai toimintakuvauksia tehdessä. Tällöin tehtäväksi jää tarvittavien tietojen kerääminen tietokannasta ja niiden linkittäminen lähdekoodipohjiin oikeille paikoille. Kun lähdekoodipohjiin on syötetty sen tarvitsemat tiedot. Jalmari luo jokaisesta lähdekoodipohjasta .awt-tekstitiedoston. Tämän jälkeen kyseisestä tiedostosta voidaan kääntää Simatic Managerissa Step7-ohjelmafunktio.



KUVIO 11. S7-lähdekoodipohjien generointi Jalmarista

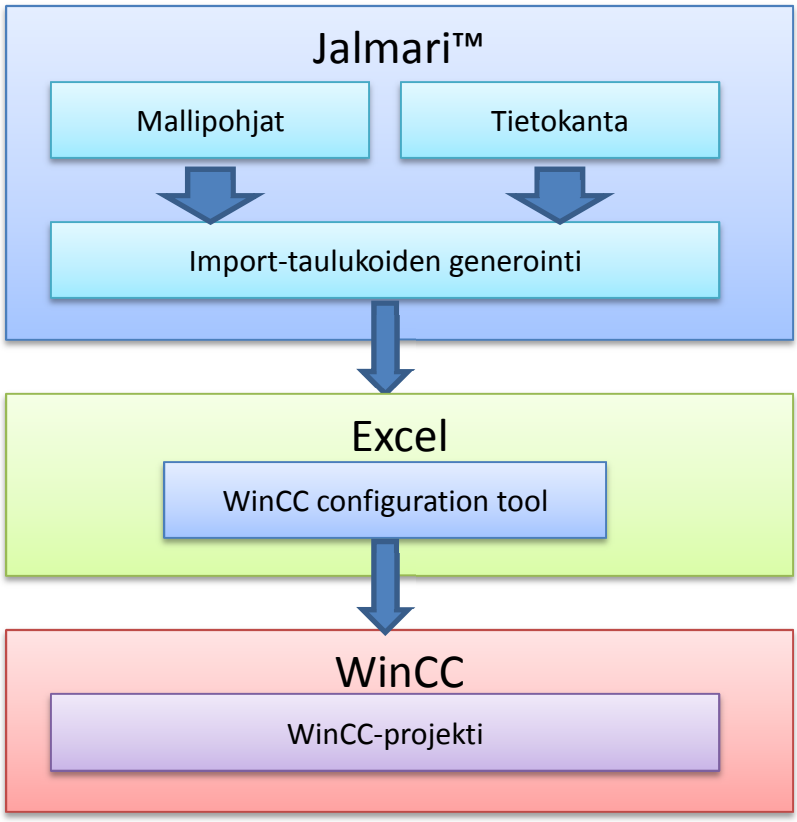
5.3 WinCC

Kaikki tiedot, joita WinCC:hen on mahdollista tuoda Jalmarista ovat Excel-tyyppisissä import-tiedostoissa. Seuraavassa listassa on esitetty ne tiedot, joita WinCC:hen on mahdollista Jalmarista generoida:

- Hälytystekstit excel-taulukkona
- Tagilista excel-taulukkona
    - Toimilaitteiden positiot
    - S7-toimilohkojen tiedostoyksikköjen numerot
- Arkistoitavien prosessimuuttujien lista excel-taulukkona
- Tekstikirjasto excel-taulukkona
    - Toimilaitteiden lukitustietojen lukitustekstit
    - Toimilaitteiden piirinimet ja niiden kuvaustekstit

Import-tiedostoja varten Jalmarissa täytyy olla tallessa taulukon rakenne. Import-tiedosto on Excel-taulukko, jossa yksi taulukon rivi vastaa yhtä tietuetta. Tietueen sisältämät arvot on jaoteltu omiin sarakkeisiin. Liitteessä 1. on esitelty hälytyslistan import-taulukon eri sarakkeiden sisältö.

Generoitujen tietojen tuonti WinCC:hen ei onnistu suoraan Jalmarista, vaan välissä joudutaan käyttämään Microsoft Exceliä. WinCC:n asennuspaketilla on apusovellus nimeltään WinCC Configuration Tool, joka asentuu MS Exceliin lisäominaisuudeksi. Työkalulla pystytään siirtämään import-tiedostot WinCC-projektiin.



KUVIO 12. Generointi Jalmarista WinCC:hen

6 JATKOKEHITTÄMINEN

Teorian vieminen käytäntöön on seuraava luonnollinen vaihe kehitysprojektissa. Toimilohkokirjaston luominen kaikille yleisimmille toimilaitteille, ja niiden käyttäminen projekteissa, luo suunnittelutoimiston sisälle yhtenäisen tavan tehdä logiikkaohjelmia. S7-toimilohkokirjaston luomisen jälkeen on järkevää tutkia saman toimintatavan käyttämistä myös muiden logiikkavalmistajien ohjelmointiympäristöissä. Tämä edelleen lisäisi logiikkaohjelmien rakenteen yhteneväisyyttä suunnittelutoimiston sisällä. Ohjelmakoodin generointia monimutkaisempiin toimintoihin, kuten esimerkiksi sekvenssiohjauksille, on myös järkevää tutkia. Pelkän tietokannan avulla monimutkaisemman ohjelmakoodin luonti ei ole järkevää, vaan apuna on

syytä käyttää Visual Basic –ohjelmointikieltä, tai vastaavaa. Ohjelman ryhmittelyllä pieniin osakokonaisuuksiin, joita voidaan yhdistellä halutulla tavalla, pystytään rakentamaan monimutkaisempiakin ohjelmakokonaisuuksia Simatic Managerin ulkopuolella.

Perussuunnittelun osana on usein automaation toimintakaavioiden piirto. Toimintakaavioissa esitetään ohjelman toiminta lohkokaavioina, jotka eivät ole riippuvaisia automaatiojärjestelmän mallista. Mahdollisuus generoida toimintakaavioista suoraan Step7-ohjelmakoodia nopeuttaisi sovellussuunnittelua huomattavasti. Tämä on mahdollista, mutta vaatisi huomattavaa panostusta Jalmarin kehitykseen.

7 POHDINTA

Tässä työssä on kuvattu pääosin teoriapohjalta tapoja, joilla tietokantapohjainen sovellussuunnittelu on mahdollista Siemens Simatic –ympäristössä. Alkuperäisestä tilanteesta poiketen työssä ei ollut mahdollista käyttää WinCC:n valvomosovellusta, vaan määrittelyt oli tehtävä täysin teoriapohjalta. Tämä vaikeutti työtä huomattavasti sekä osittain supisti myös työn sisältöä. Jalmari-tietokannan kehittämiseen ei ollut mahdollista saada resursseja, joten se osa-alue on tässä työssä jätetty määritellytasolle.

Siemens julkisti huhtikuussa 2011 uuden ohjelmointiympäristön Siemensin logiikka- ja valvomotuotteille. Siemens Tia Portal yhdistää logiikan, valvomon ja operointipäätteet yhden työkalun alle. Samalla perinteinen WinCC-valvomo-ohjelmisto poistuu käytöstä. Siemensin aikaisemmin ainoastaan operointipaneeleille käynyt suunnitteluohjelma WinCC Flexible korvaa WinCC:n. Uudistuksen myötä operointipaneelit sekä isommat valvomojärjestelmät suunnitellaan tulevaisuudessa samalla työkalulla. Siemens Tia Portal mahdollistaa myös logiikassa määritettyjen muuttujien käyttämisen suoraan valvomo-projektissa, jolloin valvomoon ei tarvitse luoda

erillisiä tageja ollenkaan. Myös hälytystekstit on mahdollista luoda suoraan Simatic Step7:ssä keskitetysti, jolloin kaikki logiikkaan kytketyt valvomot osaavat näyttää hälytykset ilman, että niitä tarvitsee konfiguroida erikseen valvomossa.

Siemens Tia Portal tuo paljon uudistuksia logiikan ja valvomon sovellussuunnitteluun ja ne vaikuttavat suoraan tässä työssä esitettyihin toiminta-malleihin. Tästä syystä onkin tärkeää tutkia, mitkä nykyisistä suunnitelmista toimivat uudessa suunnittelutyökalussa, ja mitkä osa-alueet on kehitettävä uudestaan.

LÄHTEET

Ramboll Finland Oy 2010a. Ramboll Finland Oy:n kotisivut. Viitattu 20.9.2010 http://www.ramboll.fi

Siemens AG 2010a. Simatic Manager STEP7 S7/M7/C7,V5.5, Revision level K5.5.0.0. 2010. Sovellussuunnittelun työkaluohjelmisto.

Siemens AG. 2010b. Industry automation and drive technologies overview. Viitattu 17.12.2010. http://www.automation.siemens.com.

Siemens Simatic HMI, WinCC V6.2.2.0. Valvomo-sovelluksen suunnittelutyökaluohjelmisto.

Siemens Totally Integrated Automation Portal. Viitattu 30.4.2011. http://www.industry.siemens.com/topics/global/en/tia-portal

LIITTEET

LIITE 1. Hälytystekstien import-tiedoston rakenne

Column	Description
A	Message number
B	Parameter bit-coded. For exact coding details, refer to "ODK Open Developers Kit" documentation. The documentation is only available when WinCC option "ODK" is installed.
C	Message classes 1-16 and default message classes 17+18
D	Message type (1-260, depending on class; 256 user-defined message types, 4 system-internal message types) Class 1: ;1-16 Class 2: ;17-32, Class 3: ;33-48, Class n: ;(n-1)*16+1 to n*16
E	Index of 1st user text block in the text library
F	Index of 2nd user text block in the text library
G	Index of 3rd user text block in the text library
H	Index of 4th user text block in the text library
I	Index of 5th user text block in the text library
J	Index of 6th user text block in the text library
K	Index of 7th user text block in the text library
L	Index of 8th user text block in the text library
M	Index of 9th user text block in the text library
N	Index of 10th user text block in the text library
O	Text of the 1st user text block
P	Text of the 2nd user text block
Q	Text of the 3rd user text block
R	Text of the 4th user text block
S	Text of the 5th user text block
T	Text of the 6th user text block
U	Text of the 7th user text block
V	Text of the 8th user text block
W	Text of the 9th user text block
X	Text of the 10th user text block
Y	Tag of the 1st process value bock for the message
Z	Tag of the 2nd process value bock for the message
AA	Tag of the 3rd process value bock for the message
AB	Tag of the 4th process value block for the message
AC	Tag of the 5th process value bock for the message
QD	Tag of the 6th process value bock for the message
AE	Tag of the 7th process value bock for the message
AF	Tag of the 8th process value bock for the message
AG	Tag of the 9th process value bock for the message
AH	Tag of the 10th process value bock for the message

AI	Message tag
AJ	Message bit
AK	Acknowledgement tag
AL	Acknowledgement bit
AM	Status tag
AN	Status bit
AO	Automation system number
AP	PLC subnumber (CPU number)
AQ	Info text
AR	Action type (LoopInAlarm)
AS	Function name
AT	Function parameter (Picture Name)
AU	Name of the Format DLL
AV	Group identification
QW	Group name
AX	Hide mask
AY	Creator identification
AZ	Priority

LIITE 2. Moottoritoimilohkon lähdekoodipohja

```
DATA_BLOCK DB #DB_NUM
TITLE =#DB_TITLE_TXT
AUTHOR : #AUTHOR
VERSION : 0.0

 FB 64
BEGIN
  Run := FALSE;
  SSW_OK := FALSE;
  MCC_OK := FALSE;
  Auto := FALSE;
  Man := FALSE;
  AutoStart := FALSE;
  AutoStop := FALSE;
  Local := FALSE;
  Remote := FALSE;
  LocalStart := FALSE;
  LocalStop := FALSE;
  Simulate := FALSE;
  ForceON := FALSE;
  ForceOFF := FALSE;
  FC_MA := FALSE;
  Kuittaus := FALSE;
  ON := FALSE;
  MA := FALSE;
  FON := FALSE;
  FOFF := FALSE;
  ALARM := FALSE;
  ACK := FALSE;
  VIIVE1 := 5;
  VIIVE2 := 5;
END_DATA_BLOCK

FUNCTION FC #FC_NUM : VOID
TITLE =#TITLE_TXT
//#COMMENT_TXT
VERSION : 0.1

BEGIN
NETWORK
TITLE =#MOTOR_TXT
//Motor
     A    #Run;
     =    L     0.0;
     BLD  103;
     A    #SSW_OK;
```

```
    =    L     0.1;
    BLD  103;
    A    #MCC_OK;
    =    L     0.2;
    BLD  103;
    CALL FB 64 , DB #DB_NUM (
       Run              := L     0.0,
       SSW_OK           := L     0.1,
       MCC_OK           := L     0.2,
       ON               := #OUTPUT);

    NOP  0;
END_FUNCTION
```

LIITE 3. Esimerkki symbolitaulukosta excelissä

ZERO	M 1.0	BOOL	Always 0
01P Kiertovesipumppu	DB 43	FB 51	
01P_HAIRIO	I 66.0	BOOL	KIERTOPUMPPU HÄLYTYS 0=HÄIRIÖ
01P_KAY	I 40.1	BOOL	KIERTOPUMPPU KÄY
01PF1 Poistoilmapuhallin	DB 37	FB 51	
01PF1_HAIRIO	I 66.1	BOOL	POISTOILMAPUHALLIN HÄIRIÖ 0=HÄIRIÖ
01PF1_KAUKO	I 41.0	BOOL	POISTOILMAPUHALLIN OHJASUTAPA KAUKO
01PF1_KAY_1	I 40.3	BOOL	POISTOILMAPUHALLIN KÄY 1/1
01PF1_KAY_2	I 40.2	BOOL	POISTOILMAPUHALLIN KÄY 1/2
01PF2 Poistoilmapuhallin	DB 38	FB 51	
01PF2_HAIRIO	I 66.2	BOOL	POISTOILMAPUHALLIN HÄIRIÖ 0=HÄIRIÖ
01PF2_KAY	I 41.1	BOOL	POISTOILMAPUHALLIN KÄY
01TF_HAIRIO	I 65.1	BOOL	TULOILMAPUHALLIN HÄIRIÖ 1=HÄIRIÖ
01TF_HAIRIO1	I 65.2	BOOL	TULOILMAPUHALLIN JÄÄTYMISSUOJAHÄLYTYS 1=HÄIRIÖ
01TF_HAIRIO2	I 65.3	BOOL	TULOILMAPUHALLIN PALOVAARAHÄLYTYS 1=HÄIRIÖ
01TF_KAUKO	I 40.0	BOOL	TULOILMAPUHALLIN OHJAUSTAPA KAUKO
01TF_KAY	I 39.3	BOOL	TULOILMAPUHALLIN KÄY
01TF1 Tuloilmapuhallin	DB 36	FB 51	
02PF Huippuimuri	DB 39	FB 51	
02PF_HAIRIO	I 66.3	BOOL	HUIPPUIMURI HÄIRIÖ 0=HÄIRIÖ
02PF_KAY	I 41.2	BOOL	HUIPPUIMURI KÄY
05PF Poistoilmapuhallin	DB 40	FB 51	
05PF_HAIRIO	I 67.0	BOOL	POISTOILMAPUHALLIN HÄIRIÖ 0=HÄIRIÖ
05PF_KAUKO	I 42.1	BOOL	POISTOILMAPUHALLIN OHJAUSTAPA KAUKO
05PF_KAY_1	I 42.0	BOOL	POISTOILMAPUHALLIN KÄY 1/1
05PF_KAY_2	I 41.3	BOOL	POISTOILMAPUHALLIN KÄY 1/2