Taha Sehli

Siemens PLC:n OPC UA -palvelin Node-RED-alustalla

Metropolia Ammattikorkeakoulu

Insinööri (AMK)

Sähkö- ja automaatiotekniikka

Insinöörityö

1.1.2023

Tiivistelmä

Tekijä:	Taha Sehli
Otsikko:	Siemens PLC:n OPC UA -palvelin Node-RED-alustalla
Sivumäärä:	35 sivua + 3 liitettä
Aika:	1.1.2023
Tutkinto:	Insinööri (AMK)
Tutkinto-ohjelma:	Sähkö- ja automaatiotekniikka
Ammatillinen pääaine:	Automaatiotekniikka
Ohjaajat:	Lehtori Reijo Leinonen

Tämän opinnäytetyön tarkoituksena oli antaa käytännön selitys siitä, miten OPC UA -kommunikaation protokolla toimii ja oppia pääsemään OPC UA -palvelimeen laitteistossa, kuten Siemensin S7-1200-PLC:ssä käyttämällä tätä tiedonsiirtoprotokollaa.

Työn tavoitteena oli selvittää, kuinka luodaan avoimeen lähdekoodiin (Node-RED) räätälöity OPC UA -palvelin, jota käytetään laajasti teollisuusautomaatiossa erilaisiin prosessitietojen visualisoitiin ja erilaisten ohjelmistojen väliseen kommunikointiin.

Näiden protokollien ja rajapintojen periaatteiden ymmärtämiseksi työssä tarkastellaan yksityiskohtaisesti OPC UA -palvelimen pääkonseptia, Node-REDin virtapohjaista ohjelmointia (Flow Based Programming) ja tapaa viestiä ja vaihtaa visualisoitiin käytettävä dataa.

Tämän työn tuloksena saadaan tietoa Node-RED-alustasta ulkoisesta laitteistosta (S7-1200 Siemens PLC) tulevalle datalle OPC UA -tiedonsiirtoprotokollan kautta. Lopputulosta voidaan käyttää automaatioalan koulutusympäristönä, joka antaa selkeän käsityksen OPC UA -palvelimen ja Node-RED-alustan välisestä liittämisestä ja tiedonhallintasovelluksen tekemisestä.

Avainsanat: OPC UA, Node-RED, FBP, S7-1200 PLC, rajapinta, protokolla, datan vaihto, datan visualisointi

Abstract

Author:	Taha Sehli
Title:	Siemens PLC OPC UA server on Node-RED platform
Number of Pages:	35 pages + 3 appendices
Date:	1 January 2023
Degree:	Bachelor of Engineering
Degree Programme:	Electrical and automation technology
Professional Major:	Automation technology
Supervisors:	Reijo Leinonen, Senior Lecturer

The purpose of thesis work was to give a practical description of how OPC UA communication works and learn how to access OPC UA server in a hardware device, such as S7-1200 Siemens's PLC, using this communication protocol.

The aim of the work was to demonstrate how to create a customization for Siemens S7-1200 PLC's OPC UA server on Node-RED open source that is widely used in industrial automation for making data control applications.

To understand the principles of these two protocols, the work examined in detail the main concept of OPC UA server, Node-RED Flow Based Programming (FBP) visualisation tool, and the way these communicate in order to exchange data that will be used for the visualization.

The outcome of this work is information concerning Node-RED software for data coming from an external hardware device (S7-1200 PLC) via OPC UA communication protocol. The final result can be used as an educational environment for automation field, giving a clear idea about interfacing these two platforms to make a data control application with the aid of FBP tool that called Node-Red.

Keywords: OPC UA, Node-RED, FBP, S7-1200 PLC, automation, interface, protocol, data exchange, data visualisation

Sisällys

Lyhenteet

DCOM: *Distributed Component Object Model*. Windowsin kehittämä mekanismi, jonka avulla ohjelmistokomponentit voivat kommunikoida toistensa kanssa eri tietokoneiden välillä.

DCS: *Distributed Control System*. Hajautettu automaatiojärjestelmä, jonka tarkoitus ohjata isompia prosesseja.

FBP: *Flow-based programming*. Vuo-pohjainen ohjelmointi, joka yksinkertaistaa radikaalisti sovellusten kehitysprosessia.

HMI: *Human-Machine interface*. Käyttöliittymä, joka yhdistää käyttäjän koneeseen, järjestelmään tai laitteeseen.

HTTP: *Hypertext Transfer Protocol*. Hypertekstin siirtoprotokolla, jota käytetään selaimien ja www-palvelimien tiedonsiirtoon.

MES: *Manifacturing Execution Systems*. Tuotannon operatiiviseen ohjaukseen liittyviä toimintoja, jotka keräävät tarvittavaa tietoa tuotannon tapahtumista ja ohjauksesta valvontaa varten.

MQTT: *Message Queue Telemetry Transport* on protokolla, joka on luotu keräämään tietoja monista laitteista ja siirtämään ne IT-infrastruktuurin. Se soveltuu etävalvontaan, erityisesti Machine to Machine-yhteyksissä.

OPC UA: *Open Platform Communication Unified Architecture*. Alustariippumaton kommunikaatioprotokolla perinteisistä Windows-pohjaisista tietokoneista Linux-järjestelmiin ja mobiilialustoihin.

OT/IT: *Operating Technology* (OT) on laitteisto ja ohjelmisto, joka valvoo ja ohjaa laitteita, prosesseja ja infrastruktuuria. Sitä käytetään teollisissa ympäristöissä. *Information Technology* (IT) on tietojen ja sovelluksien hallitsemat järjestelmät.

PID: *Proportional-integral-derivative*-säädin. Laite, jota käytetään teollisissa ohjaussovelluksissa lämpötilan, virtauksen, paineen, nopeuden ja muiden prosessimuuttujien säätelyyn.

PLC: *Programmable Logic Controller*. Pieni teollisuustietokone, joka on suunniteltu suorittamaan logiikkatoimintoja ja ohjaamaan prosesseja.

SOAP: *Simple Object Access Protocol*. Viestintäprotokolla, jolla voidaan toteuttaa Web Service -tyyppisiä palveluja, kuten sovellusohjelmien välinen kommunikointi ja tietojen vaihtaminen laitteiden välillä.

TCP/IP: *Transmission Control Protocol/Internet Protocol*. Internetissä käytetty tiedonsiirtoprotokolla, jonka avulla tietokoneet ja muut laitteet voivat lähettää ja vastaanottaa dataa keskenään.

1 Johdanto

Tietokone- ja ohjelmistopohjaisten automaatiojärjestelmien käyttö teollisuusautomaatiossa lisääntyi nopeasti 1990-luvun alusta lähtien. Erityisesti Windows-pohjaisia tietokoneita käytetään visualisointi- ja ohjaustarkoituksiin. Yksi viime vuosien suurimmista panostuksista standardoitujen automaatio-ohjelmistojen kehittämiseen oli automaatiotietojen saatavuus laitteissa, joissa on käytössä lukematon määrä erilaisia protokollia ja rajapintoja. Tämän takia vuodesta 1996 lähtien OPC Foundationilla oli tehtävä hallita globaalia organisaatiota, jossa käyttäjät ja toimittajat tekevät yhteistyötä luodakseen turvallisia ja luotettavasti yhdessä toimivia tiedonsiirtostandardeja useille toimittajille ja alustoille teollisuusautomaatiossa. Tuloksena syntyi useita ohjelmistorajapintoja standardoimaan tiedonkulkua prosessitasolta tiedonhallintatasolle. [1.]

Tärkeimmät käyttötapaukset ovat teollisuusautomaatiosovellusten, kuten käyttöliittymien liitännät, jotka käyttävät dataa laitteista hallintasovelluksen (MES) luomista varten.

Tämän insinöörityön pääasiallisena tavoitteena oli selvittää käytännössä, miten saadaan toteutettua turvallinen ja luotettava tiedonvaihto Siemensin S7-1200 -PLC-palvelimen ja Node-RED-alustan välillä käyttämällä OPC-Foundationin kehittämää OPC UA -tiedonsiirron protokollaa. OPC UA tulee sanoista Open Platform Communication Unified Architecture ja on kehitetty sitä varten, että eri valmistajien laitteet ja palvelimet pystyvät keskustelemaan keskenään mahdollistaakseen turvallisen tiedonvaihdon analytiikkaa ja visualisointia varten. Lisäksi työn tarkoituksena oli luoda räätälöity OPC UA -palvelin Node-REDin avoimen lähdekoodin alustalle, jonka avulla voidaan ohjelmoida verkkoselaimella hallintasovelluksia.

Opinnäytetyön käytännön osuudessa vaihtoehto, jonka avulla voidaan simuloida eri teollisuuden järjestelmät, kuten PID ja tuotteiden lajittelu kuljettimen ja toimilaitteen avulla, on Factory IO, joka on 3D-tehdassimulointiohjelmisto.

2 Integraation tausta automaatiossa

Viime vuosina yrityksille on tullut yhä selvemmäksi, että kiireellinen prioriteetti tuotantotoiminnalle on digitaalinen muutos, jonka avulla ne voivat pysyä kilpailukykyisinä ja kestävinä tulevaisuudessa. Monet perinteiset tehtaat käyttävät eri toiminnoissaan automatisoituja koneita ja laitteitta, mutta ne eivät ole yhteydessä toisiinsa. Perinteisen tehtaan operaattorit ja tiedonhallintajärjestelmät toimivat erillään toisistaan ja niitä on jatkuvasti koordinoitava ja integroitava manuaalisesti. [2.]

Tämän vuosikymmenen aikana digitalisaatioaallon leviämisen ja viestintätekniikan nopean kehityksen myötä laitteiden yhteen liitettävyydestä ja etäyhteydestä on tullut keskeisiä osia kehityssuunnassa. Tämän seurauksena on syntynyt uusi teollinen vallankumous, jonka nimi on Teollisuus 4.0. [2.]

Neljännen teollisen vallankumouksen alkaessa teollisuussektori on kokenut suuria muutoksia, joiden prosessi jatkuu edelleen. Tämä Teollisuus 4.0:n tärkein pilari teollinen esineiden internet (IIoT) muuttaa automaatiomaailman ennennäkemättömällä vauhdilla. Näin ollen IIoT-konsepti voi ilmaista sekä haasteita että mahdollisuuksia koskien liitettävyyttä, integroitavuutta ja tiedonvaihtoa laitteiden ja järjestelmien välillä.

2.1 Integraation haasteita

Integraation tärkeä haaste on yhteys laitteisiin, pohjimmiltaan tiedonsiirtoprotokollat sekä uusille että vanhoille järjestelmille. Automaation ja teollisuuden maailma on edelleen täynnä vanhoja järjestelmiä, mikä johtuu

pitkäaikaisista tuotteiden ja ohjelmistojen elinkaarista ja yritysten kyvyttömyydestä muuttaa vanhoja järjestelmiä perusteellisesti. [2.]

Lisäksi aiemmin perinteisissä tehtaissa tai tuotannossa tietolähteet ja laitteet olivat pääosin samalta valmistajalta tai kahdelta yhteensopivalta tuotteelta. Toisin sanoen tietoihin pääsi käsiksi vain toimittaneiden yritysten työkaluilla. Siksi näihin yrityksiin on palattava, kun on tarvetta laajentaa projekteja ja toimintaa tai lisätä siihen uusia laitteitta, mikä lisää kustannuksia ja riippuvuutta valmistusyrityksistä.

2.2 Ratkaisut

Ajan myötä protokollien, laitteiden ja valmistajien eroavaisuudet ovat johtaneet siihen, että käytettyjen vanhojen laitteiden ja nykyaikaisten viestintäprotokollien välille on syntynyt aukkoja, jotka ilmaantuvat niiden välisen yhteensopimattomuuden vuoksi. Tämä edellytti yhtenäistä standardia, joka turvaa tiedonsiirron vanhojen laitteiden välillä ilman tarvetta vaihtaa niitä. OPC-teknologia oli tämä standardi, joka mahdollisti turvallisen tiedon siirron mistä tahansa datalähteestä mihin tahansa sitä tarvitsevaan sovellukseen. Näin ollen OPC-teknologiasta on tullut globaaleja standardeja, joiden spesifikaatiot OPC Foundation määrittää ja julkaisee jatkuvasti uusia spesifikaatioita tälle standardille. [6, s. 94–95.]

Lisäksi digitaalisen muutoksen vahvistamiseksi monet yritykset ovat kehittäneet uusia avoimia alustoja, joiden avulla ne voivat saada nopeasti toteutettua ja integroitua erilaisia protokollia. Yksi mielenkiintoisista alustoista on Node-RED, joka on IBM:n kehittämä uusi tapa ohjelmoida visuaalisesti. Node-RED on suunniteltu IoT-ympäristöksi, mutta teollisten käyttöliittymien käyttämisen myötä se siirtyi IIoT-ympäristöön [2]. Node-REDissä on monia sarja- ja Ethernet-pohjaisia perinteisiä sekä uusia protokollia [2].

Tämän työn käytännön osuudessa esitetään tarkemmin visuaalisen ohjelmoinnin soveltamista integroitaessa OPC UA -protokolla, joka mahdollistaa monen eri järjestelmän välisen kommunikoinnin.

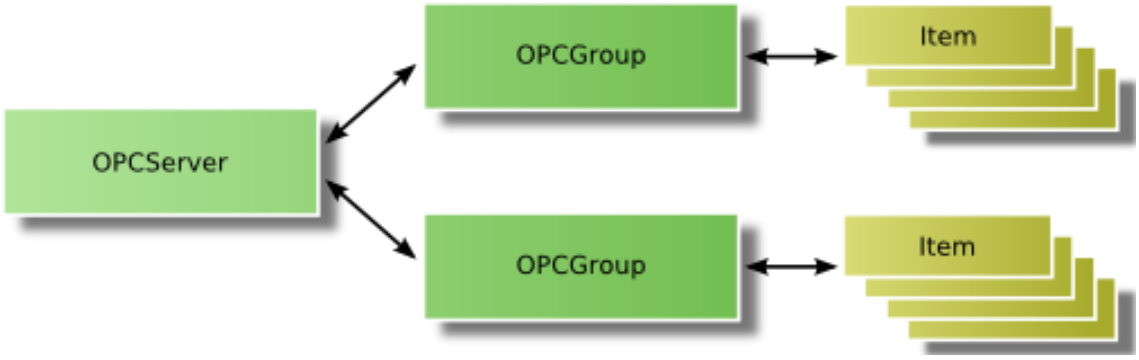
3 Klassinen OPC

Teollisuuden erilaisten tarpeiden täyttämiseksi on kehitetty kolme erillistä OPC-spesifikaatiota. Näissä määrityksissä on erityiset määritelmät prosessitietojen, historiatietojen ja hälytysten käsittelylle. Klassisen OPC:n tekniset tiedot perustuvat Microsoft Windows -teknologiaan. Se käyttää COM/DCOM-tekniikkaa helpottamaan ohjelmistokomponenttien välistä tiedonvaihtoa. [3.]

3.1 OPC Data Access (DA)

OPC DA on OPC-rajapinta, joka on erityisesti suunniteltu helpottamaan tietojen vaihtoa ohjelmistojen ja laitteistojen välillä. Tyypillisesti OPC DA:n ensisijainen tehtävä on siirtää reaaliaikaista dataa prosessiohjauslaitteista, kuten PLC:istä ja DCS:istä, käyttöliittymään tai näyttöasiakkaaseen. Lisäksi se mahdollistaa nykyisen prosessidatan sisältävien muuttujien seurannan, lukemisen ja kirjoittamisen. Sen päätarkoituksena on tarjota standardoitu tiedonsiirtomenetelmä prosessilaitteiden ja Windows-pohjaisten sovellusten välillä. [4, s. 29–30.]

OPC DA-asiakkaat voivat valita tietyt muuttujat (items), joita he haluavat lukea, kirjoittaa tai valvoa palvelimella. Yhdistääkseen palvelimeen OPC-asiakas luo OPCServer-objektin, joka tarjoaa keinoja tai menetelmiä navigoida osoiteavaruuden hierarkiassa. Tämän kautta asiakas voi löytää asiaankuuluva muuttujat ja niihin liittyvät ominaisuudet, kuten käyttöoikeudet ja tietotyypit. Päästäkseen käsiksi prosessitietoihin asiakkaat ryhmittelevät OPC-objekteja, joilla on samat asetukset, kuten päivitysaika, käyttämällä OPCGroup-objektia. [4, s. 29–30.] Kuva 1 näyttää eri objektit, jotka OPC-asiakas luo palvelimelle.



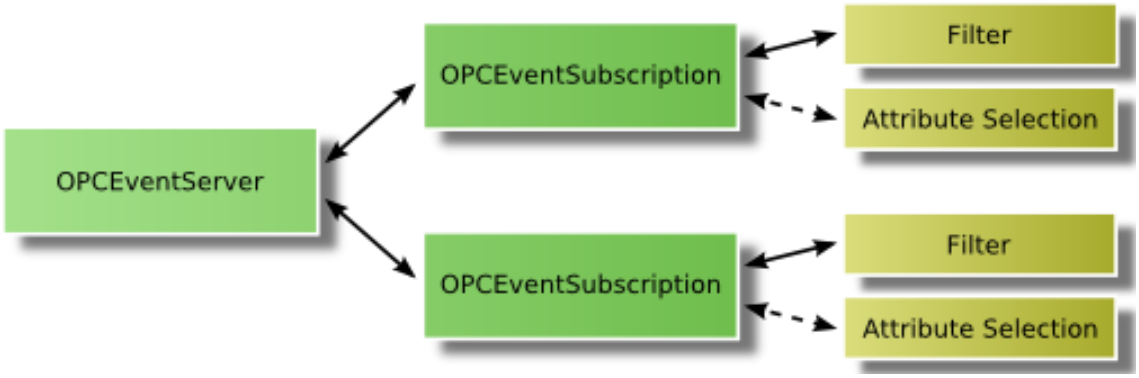
Kuva 1. OPC-asiakkaan luomat objektit OPC-palvelimeen [5].

OPC-asiakas määrittää päivitysvälin muuttujien ryhmässä (OPCGroup), jossa on kiinnostavat muuttujat. Päivitysastetta (update rate) käytetään muuttujien arvojen muutosten sykliseen tarkistamiseen palvelimessa. Jokaisen jakson jälkeen palvelin lähettää vain muuttuneet arvot. [4, s. 29–30.]

3.2 OPC Alarm & Event

OPC A&E-spesifikaatiolla on kyky vastaanottaa ilmoituksia sekä hälytyksistä että tapahtumista. Hälytykset ovat ilmoituksia, jotka pitävät asiakkaat ajan tasalla prosessin tilan muutoksista. Asiakkaat voivat esimerkiksi saada hälytyksiä, kun säiliön nestepinta ylittää maksimitason tai laskee vähimmäistason alapuolelle. Jotkut hälytykset edellyttävät käyttäjän kuittaamista, mikä voidaan tehdä myös OPC A&E:n kautta. Toisaalta tapahtumat ovat viestejä, jotka tarjoavat asiakkaille tietoa tietyistä tapahtumista. [4, s. 29–30.]

Ilmoitusten saamiseksi OPC A&E muodostaa yhteyden palvelimeen, tilaa saatavilla olevat ilmoitukset ja vastaanottaa tämän jälkeen kaikki palvelimella olevat käynnistetyt ilmoitukset. Tämän yhteyden aloittamiseksi OPC-asiakas luo ensin OPCEventServer-objektin muodostaakseen yhteyden A&E-palvelimeen. Toisessa vaiheessa asiakas luo OPCEventSubscription-objektin tapahtumaviestien vastaanottamista varten. [4, s. 29–30.] Kuvassa 2 on esitetty OPC-asiakkaan luomat objektit A&E-palvelimen yhteyden muodostamiseen.



Kuva 1. OPC-asiakkaan luomat objektit A&E-palvelimeen [5].

Ilmoitusten ja tapahtumien määrän rajoittamiseksi OPC-asiakas voi määrittää tietyt suodatusehdot erikseen kullekin tilaukselle (subscription), esimerkiksi suodattaa tapahtumatyyppien, prioriteetin tai tapahtumalähteen mukaan. [4, s. 29–30.]

3.3 OPC Historical Data Access

OPC HDA -palvelimella on historiallinen aikaleima (timestamp), kuten milloin hälytys tapahtui tai milloin jokin lukema on tulossa ja milloin tätä lukemaa muutettiin.

OPC HDA tarjoaa myös pääsyn tallennettuihin tietoihin yksinkertaisista sarjatietojärjestelmistä monimutkaisiin järjestelmiin. OPC HDA -spesifikaatiossa on kolme erilaista mekanismia tallennettujen tietojen käyttämiseen. Ensimmäinen mekanismi lukee raakadataa arkistosta, jossa asiakkaat määrittelevät yhden tai useamman muuttujan, jonka he haluavat lukea. Toinen mekanismi lukee yhden tai useamman muuttujan arvon tietyllä aikaleimalla. Kolmas lukumekanismi laskee ryhmitellyt arvot, yhdelle tai useimmalle muuttujalle historiallisista tietokantatiedoista tietyltä ajanjaksolta, joka sisältää aina siihen liittyvän aikaleiman ja laadun. Näiden mekanismien lisäksi OPC HDA määrittelee menetelmät tietojen lisäämiseksi, korvaamiseksi ja poistamiseksi historiatietokannasta. [4, s. 31.]

Yksi OPC-standardin puutteista, joka ilmeni sen leviämisen ja uusien käyttöjärjestelmien syntymisen myötä, on se, että standardi on linkitetty vain Windows-käyttöjärjestelmään, koska se on täysin riippuvainen DCOM/COM-tekniikasta palvelin ja asiakaan välillä. Tällä tekniikalla oli alun perin tärkeä rooli OPC:n levittämisessä, mutta markkinoiden tarpeiden lisääntyminen ja teknologian kehittyminen on johtanut useisiin haittoihin tässä tekniikassa, mukaan lukien:

- DCOM:lla on ongelmia, kun käytetään etäviestintää OPC:n kanssa.
- DCOM on vaikea konfiguroida, siinä on pitkät ja ei-konfiguroitavissa olevat aikakatkaisut, eikä sitä voi käyttää Internet-viestintään.
- DCOM/COM-kanavat ovat suljettuja. Niitä voidaan pitää mustana laatikkona, koska ne eivät anna kehittäjien hallita ominaisuuksiaan tai käyttää niitä. Siksi kehittäjät joutuvat sopeutumaan olemassa oleviin puutteisiin ja virheisiin pystymättä käsittelemään niitä.
- Turvallisuustaso on alhainen, sillä OPC on juurtunut Windows-käyttöjärjestelmään. Siksi DCOM:n käyttö tekee siitä erittäin haavoittuvan hyökkäyksille hyödyntämällä kaikkia käyttöjärjestelmän haavoittuvuuksia.

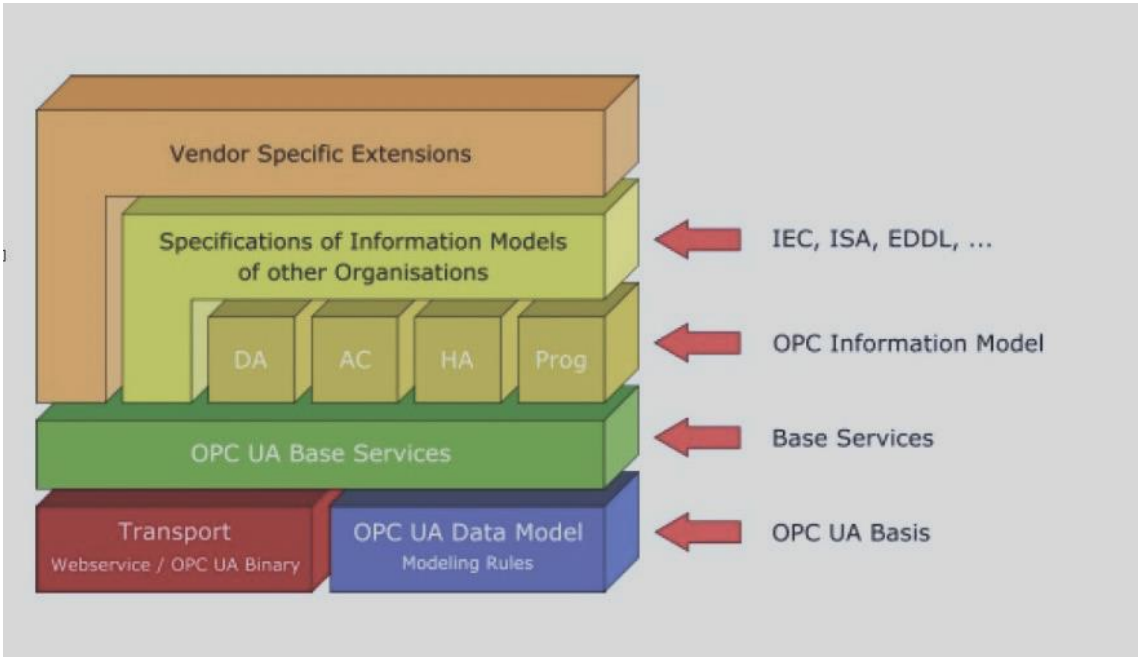
Näitten useiden puutteiden vuoksi OPC-säätiö harkitsi luopumista Windows-käyttöympäristöön. Alkuperäinen OPC-standardimääritys oli OPC XML-DA, ja sen tekijät yrittivät korvata DCOM/COM-tekniikan SOAP/HTTP- ja Web-tekniikoilla säilyttäen samalla OPC DA:n onnistuneet ominaisuudet. Tämän spesifikaation hyödyntäminen havaittiin kuitenkin myöhemmin riittämättömäksi uuden OPC-sukupolven vaatimusten täyttämiseksi sen hitauden ja tehottomuuden vuoksi. Tämän seurauksena OPC-säätiön johtajat luovat jatkuvasti päivitettyjä versioita OPC-standartista täyttääkseen markkinoiden tarpeet ja toiveet sekä korjatakseen aiempien OPC-standardien viat ja virheet. Vuonna 2008 luotiin OPC Unified Architecture (OPC UA), joka ei ole

riippuvainen Microsoft Windowsista vaan on yhteensopiva kaikkien käyttöjärjestelmien kanssa. [6, s. 94–95.]

4 OPC UA:n arkkitehtuuri

OPC UA tuli parantamaan aiempaa versiota. Se yhdistää kaikki klassisen OPC-standardin onnistuneet spesifikaatiot ja toiminnot yhdeksi monipuoliseksi kehykseksi. OPC UA:n kautta eri järjestelmät ja laitteet voivat kommunikoida lähettämällä viestejä asiakkaan (Client) ja palvelimen (Server) välillä erityyppisten verkkojen kautta Web Service-tekniikan viestintäprotokollaa käyttäen (SOAP). Tämä auttaa varmistamaan erilaisten laitteiden ja järjestelmien yhteensopivuuden sillä, että ne voivat vaihtaa tietoja turvallisesti ja olla vuorovaikutuksessa toistensa kanssa valmistajasta tai mallista riippumatta. [8.]

OPC UA:n tavoitteiden saavuttamiseksi sen arkkitehtuuri perustuu eri loogisille kerroksille, jotka on esitetty kuvassa 3.



Kuva 3. OPC UA:n kerroksellinen arkkitehtuuri [4, s. 36].

Kuvassa 3 nähdään, että uusi OPC UA:n yhtenäinen arkkitehtuuri yhdistää ja laajentaa kaikkia aiempia OPC-spesifikaatioita, erityisesti laitetietoja (prosessiarviot, mitatut tiedot, parametrit jne.).

4.1 Kuljetus

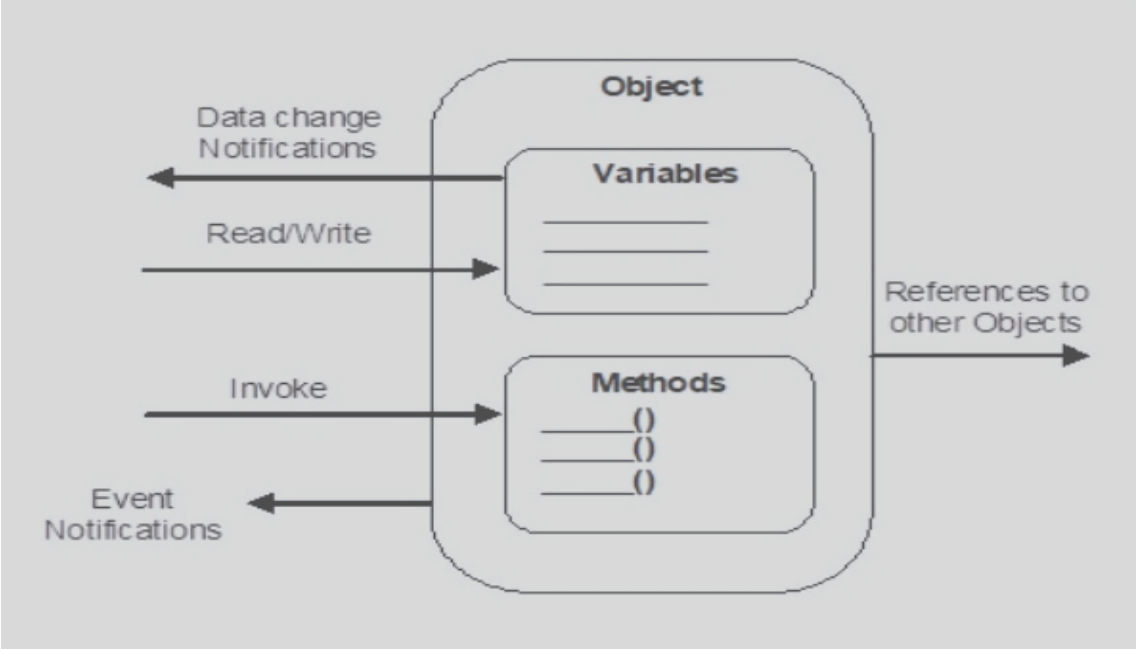
Kuljetuskerroksessa on erilaisia yhteysmekanismeja eri käyttötapauksiin. OPC UA:n ensimmäinen versio määrittelee binaarisen TCP-protokollan tehokkaaseen Internet-viestintään ja toinen versio määrittelee Web Servicen, XML:n ja HTTP:n palomuuriystävälliseen Internet-viestintään. Lisäksi on olemassa binaarisen TCP-protokollan ja Web Servicen yhdistelmä, niin kutsuttu hybridiprotokolla, jonka kautta kommunikointiviestejä lähetetään käyttämällä salattua kanavaa (HTTP). Lisäprotokollat ovat mahdollisia ja niitä voidaan lisätä tarpeen mukaan. [4, s. 35–58.]

4.2 Metamalli

Ohjelmistollisesti objekti on kokoelma attribuutteja (lämpötila, paine), menetelmiä (On, Off) ja tapahtumia (lämpötila liian korkea, paine liian matala). Objektit järjestetään hierarkioihin, jotta objektilla voi olla ominaisuuksina yksinkertaisempia ja pienempiä objekteja. Objekti kantaa sisältään valtavan määrä tietoa. Objektit ovat muuttujien, tapahtumien ja menetelmien (Attribute, Events, Methods) paikkamerkkejä, ja ne linkitetään toisiinsa viittauksilla (References). OPC UA:n kautta tiedonvaihto perustuu oliopohjaiseen tietojenkäsittelykonseptiin, jossa käytetään kentistä, tapahtumista ja menetelmistä koostuvia tietorakenteita (Objects) ja niiden vuorovaikutusta sovellusten suunnittelussa. Objektit edustavat todellisia entiteettejä (entities), kuten anturia, ohjausyksikköä. Nämä objektit sisältävät attribuutteja yksittäisillä arvoilla sekä parametreja sisältäviä menetelmiä, joita asiakassovellus käyttää laitteen ohjaamiseen ja datan keräilyyn. [9.]

OPC UA tarjoaa puitteet (framework), joita voidaan käyttää taustalla olevan järjestelmän tietojen ja toimintojen esittämisessä objekteina osoiteavaruudessa

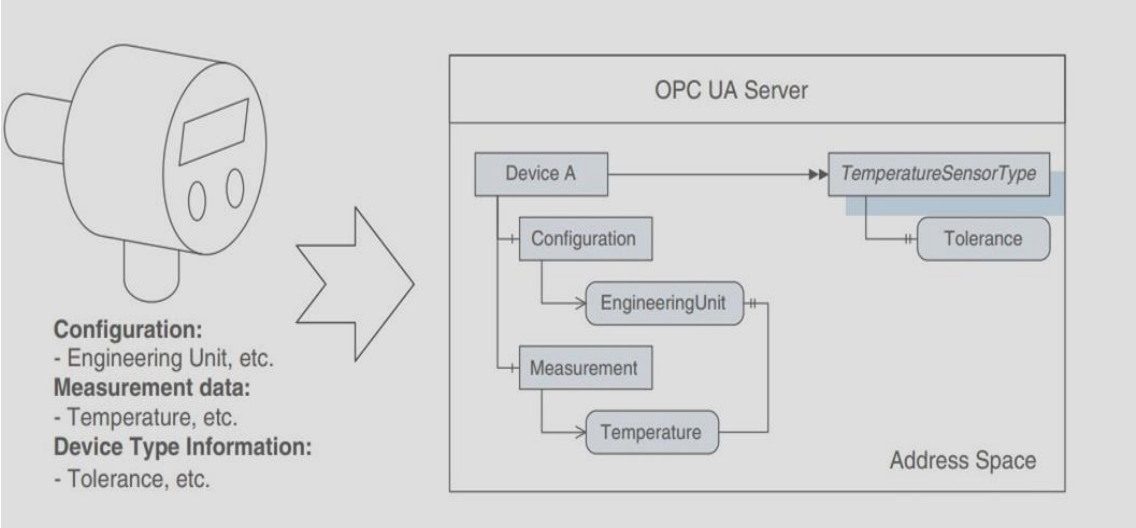
(AddressSpace). Osoiteavaruus on tietojen kokoelma, jonka OPC UA -palvelin paljastaa asiakkailleen objektimallina, joka sisältää siihen liittyvän tyyppijärjestelmän sekä määritellyt säännöt, jotka kuvaavat, kuinka kukin fyysinen järjestelmä muunnetaan OPC UA -yhteensopivaksi malliksi edustamaan sitä OPC UA -palvelimessa. Tällä metamallilla voidaan kuvata kaikenlaisia laitteita, toimintoja ja järjestelmätietoja standardoidulla tavalla niin, että samalla peruslaitemallilla esitettynä asiakkaat pääsevät käsiksi eri toimittajakohtaisten OPC UA -palvelimien tarjoamiin laitetietoihin. [9.] Kuvassa 4 on esitetty OPC UA -perustietomalli tai objektimalli.



Kuva 4. OPC UA -objektimalli [9].

Osoiteavaruus on objektien ja niihin liittyvien tietojen joukko, jonka OPC UA:n palvelin asettaa asiakkaiden saataville. Osoiteavaruus esittää sisältönsä joukkona solmuja (Nodes) yhdistettynä viittauksilla (References). Jokainen solmu on määritelty solmuluokkaan (NodeClass), ja jokainen solmuluokka edustaa objektimallin eri elementtiä. Tämän mallin avulla tiedot, hälytykset ja tapahtumat sekä niiden historia voidaan integroida yhdeksi OPC UA -palvelimeksi. Esimerkiksi OPC UA -palvelimet voivat esittää lämpötila-anturin objektina osoiteavaruudessaan, joka koostuu lämpötila-arvosta,

joukosta konfigurointiparametreja ja laitteen tiedoista. Anturin toimittamat tiedot OPC UA -palvelimelle käyttävät peruslaitemallia. Peruslaitemalli mahdollistaa olemassa olevien OPC-toimintojen siirtämisen OPC UA:han. [4, s. 45–46.] Kuvassa 5 on esitetty anturin objekti.



Kuva 5. Lämpötila-anturin toimitetut tiedot OPC UA -palvelimelle [4, s. 45].

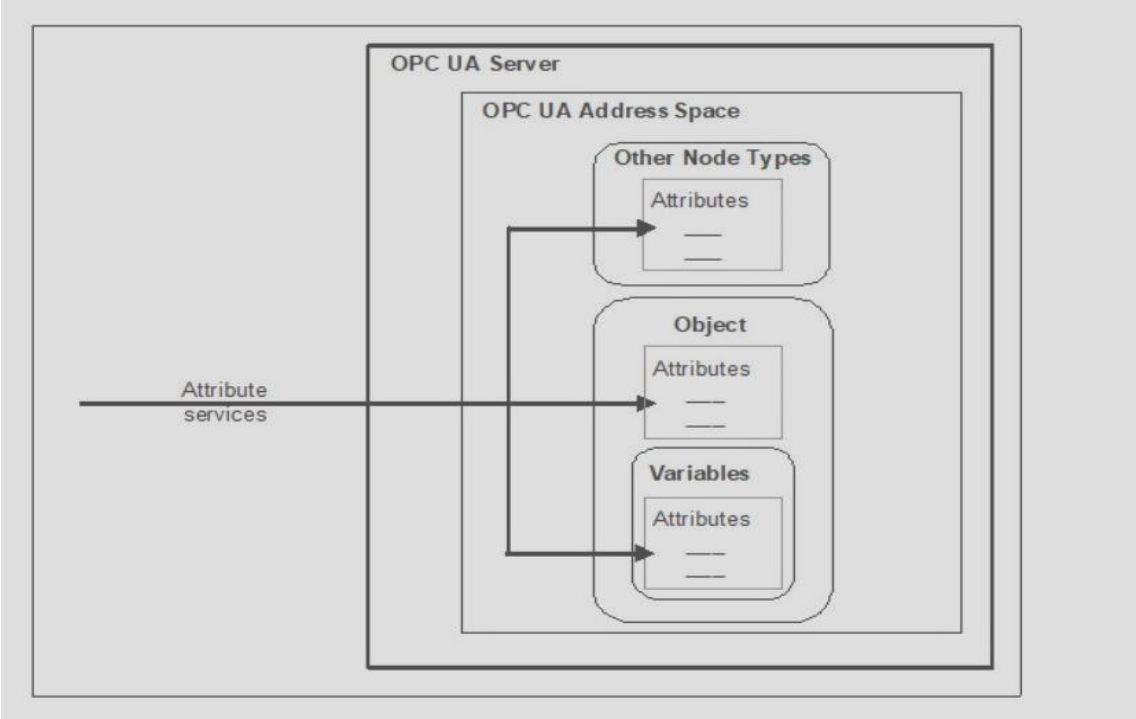
Mikä tahansa OPC UA -voi tarjota sopivan rajapinnan, joka paljastaa laitteen mitatut arvot ja konfiguroi laitteen palvelimen toimittaman laitemallin avulla.

4.3 OPC UA -peruspalvelut (Base Services)

OPC UA määrittelee kiinteän joukon palveluita (Services Sets), joilla on hyvin määritellyt parametrit ja käyttäytyminen. Nämä palvelut ovat yleisiä, esimerkiksi tietojen ja ominaisuuksien lukeminen on vain yksi ”Lue”-palvelu ja OPC UA -palvelimen osoiteavaruuden selaamiseen on vain yksi ”selaa”-palvelu DA-selauksen, AE-selauksen ja HDA-selauksen sijaan, jossa asiakas päättää, mitä viittauksia (solmujen välillä) seurataan. Nämä palvelut tarjoavat asiakkaille kaksi ominaisuutta, joista ensimmäisen avulla asiakkaat voivat lähettää pyyntöjä palvelimille ja vastaanottaa vastauksia niiltä. Toisen ominaisuuden avulla asiakkaat voivat tilata (Subscribe) palvelinilmoituksia. Palvelin käyttää

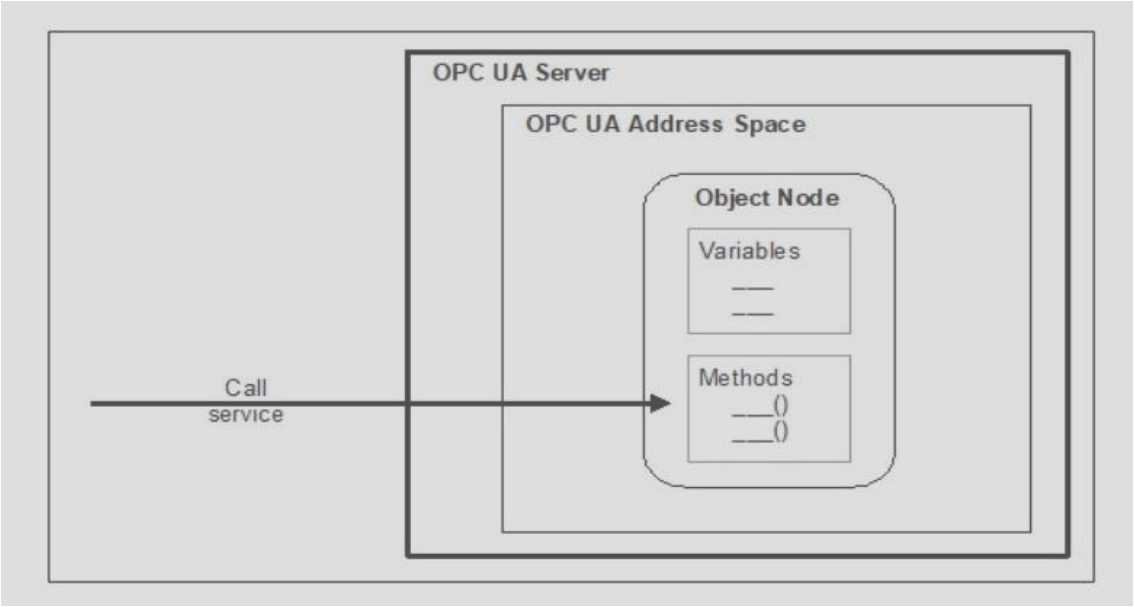
ilmoituksia raportoimaan tapahtumista, kuten hälytyksistä, data-arvon muutoksista, tapahtumista ja ohjelman suoritustuloksista. [10.]

Attribuuttipalveluiden joukko määrittelee palvelut, joiden avulla asiakkaat voivat lukea ja kirjoittaa solmujen attribuutteja (read/write), mukaan lukien niiden historialliset arvot (kuva 6).



Kuva 6. Attribuuttipalvelusarja (Attribute Service Set) [10].

Menetelmäpalvelusarja (Method Service Set) mahdollistaa asiakassovelluksille menetelmien kutsumista. Sen avulla asiakassovellukset voivat lähettää pyyntöjä palvelimelle, joka suorittaa pyydetyn toiminnon ja palauttaa tuloksia, jotka liittyvät tiettyihin menetelmiin (kuva 7).



Kuva 7. Menetelmäpalvelusarja [10].

OPC UA-palvelut edustavat UA-asiakas- ja UA-palvelinsovellusten välistä vuorovaikutusta. Asiakas käyttää palveluita löytääkseen ja käyttääkseen palvelimen antamia tietoja.

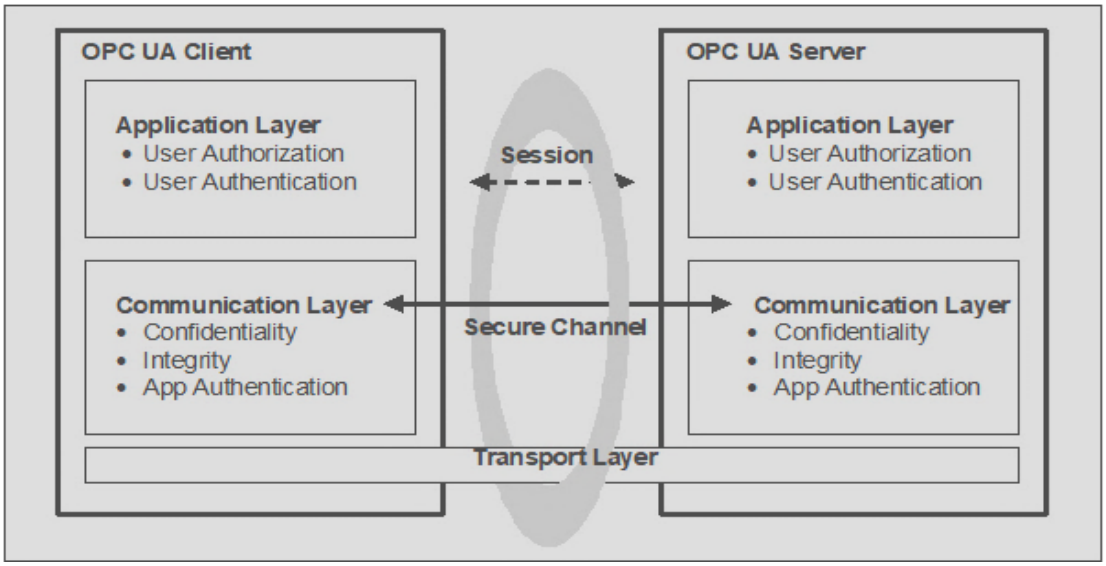
4.4 OPC UA -tietomalli (Information Model)

OPC UA -palvelimen ensisijainen tavoite on paljastaa tietoa, jonka avulla asiakkaat voivat hallita taustalla olevaa reaaliaikaista prosessia ja koko yritystä kokonaisuutena. Suurin haaste on integroida järjestelmät ja johtamisresurssit yhdeksi ympäristöksi. Tiedot kuvaavat prosessin tilaa ja käyttäytymistä, ja palvelimen tulee pystyä siirtämään ne molempiin suuntiin. OPC UA -tietomallin rooli on tukea tätä siirtoa ainutlaatuisella ja läpinäkyvällä tavalla huolimatta prosessin monimutkaisuudesta ja asiakkaiden rooleista yrityksen johtamishierarkiassa. Tietomalli standardoi tapaa, jolla tieto tulee esitettynä tietokonekeskeisenä datana, jotta osoiteavaruuden paljastama tieto olisi eri järjestelmissä molemminpuolisesti ymmärrettävää. [11.]

Tietomalli sisältää myös standardispesifikaatioita, jotka ovat osa yleistä OPC UA -standardisarjaa ja määrittelevät eri pääsytyyppeihin, kuten Data Access (DA), Alarm & Condition (AC), Programs (Prog) ja Historical Access (HA) liittyvän tietomallin. Tietomalli kuvaa palvelimen osoiteavaruuden standardisolmut, jotka ovat standardityyppejä, joita käytetään palvelinkohtaisten solmujen sisääntulopisteinä. Kaksi tärkeintä spesifikaatiota tiedon mallinnuksen ja käytön ymmärtämiseksi ovat osoiteavaruusmalli (Address Space Model) ja palvelu (Services). Nämä kaksi eritelmää ovat tärkeitä OPC UA -sovellusten suunnittelussa ja kehittämisessä. [11.]

4.5 OPC UA -turvallisuus (Security)

Tietoturva on olennainen osa tietokonejärjestelmissä, erityisesti prosessien hallintaan tarkoitetuissa järjestelmissä. Turvallisuusinfrastruktuurin tulee olla tehokas ja riittävän joustava tukemaan eri organisaatioiden vaatimia erilaisia turvallisuuspolitiikkoja. Näitten vaatimusten täyttämiseksi OPC UA -tietoturva-arkkitehtuuri on määritelty ratkaisuksi, joka mahdollistaa tarvittavien suojausominaisuuksien toteuttamisen eri paikoissa sovellusarkkitehtuurissa. OPC UA -tietoturva on monikerroksinen konsepti, joka koostuu toisistaan erotetuista suojauskerroksista: sovelluskerroksesta, viestintäkerroksesta ja kuljetuskerroksesta. Se sisältää käyttäjän todennuksen (authentication) ja valtuutuksen (authorization) sekä salauksen ja tietojen eheyden allekirjoittamalla (data integrity by signing) (kuva 8). [4, s. 234–238.]



Kuva 8. OPC UA -suojausarkkitehtuuri [12].

OPC UA määrittelee yleisen tietoturva-arkkitehtuurin eri tasoilla, joilla kullakin on omat turvallisuusvastuunsa. Jokainen kerros voidaan siten toteuttaa käyttämällä erilaisia tekniikoita, jotka on määritelty OPC UA -spesifikaatiossa (Servise Mappings). Viestintä asiakkaiden ja palvelimien välillä on turvattu luomalla istuntoja (Sessions) suojattujen kanavien yli. Yhteyden muodostus perustuu julkisen avaimen salaukseen sertifikaateilla. Sertifikaatit tunnistavat ihmiset, tietokoneet tai sovellukset, ja niitä käytetään luomaan kahden entiteetin välinen luottamus. Varmenteita hallinnoi PKI (Public Key Infrastructure), joka edustaa infrastruktuuria varmenteiden pyytämiseen, luomiseen, jakeluun, validointiin ja kumoamiseen. [4, s. 234–238.]

Sovelluskerroksen suojaus

Sovelluskerrosta käytetään istunnon aikana siirtämään laitostietoja, asetuksia, ohjeita ja reaaliaikaista dataa laitteista asiakkaan ja palvelimen välillä. Istuntoa käytetään asiakkaan kanssa työskentelevien käyttäjien todentamiseen ja valtuutukseen sekä tiettyjen tuotteiden todentamiseen ja valtuutukseen. Tässä suojauskerroksessa on useita suojausmekanismeja. Sovelluksen käyttäjä voidaan tunnistaa käyttäjä/salasana-yhdistelmästä tai käyttäjävarmenteesta. Tietojen käyttöoikeuksia voidaan säätää kullekin yksittäiselle solmulle:

esimerkiksi käyttäjä saa vain lukea arvoja, kun taas ylläpitäjällä voi olla myös kirjoitusoikeus, mutta vieras ei välttämättä pysty edes selaamaan solmua (Nodes). Lisäksi tarkastusmekanismit on määritelty niin, että palvelin voi kirjata, kuka henkilö on muuttanut mitä arvoa milloin tahansa (audit event, audit log). [4, s. 234–238.]

Viestintäkerroksen suojaus

Viestintäkerros luo suojatun kanavan asiakkaan ja palvelimen välille käyttämällä salausta, allekirjoituksia ja digitaalisia varmenteita. Viestintäkerroksessa OPC UA -asiakas muodostaa yhteyden palvelimeen ja vaihtaa sertifikaatteja yhteyksien todentamiseksi sekä lähetettyjen viestien salaamiseksi ja allekirjoittamiseksi. Viestintäkerroksessa luodun suojatun kanavan kautta sovelluskerroksen istunto luottaa siihen suojattuun viestintään. Kaikki istuntotiedot välitetään viestintäkerrokseen jatkokäsittelyä varten. [4, s. 234–238.]

Kuljetuskerroksen suojaus

Kuljetuskerros on vastuussa suojattujen tietojen lähettämisestä ja vastaanottamisesta Socket-yhteyden kautta. OPC UA määrittelee viestisuojauksen binaarille sekä verkkopalveluprotokollalle (TCP, Web-Service). Hybridiversio käyttää kuljetuskerroksen suojausta (Transport Layer Security) kuljetusreitin turvaamiseen. Kuljetuskerros on OPC UA:n ensimmäinen puolustuslinja, joka keskittyy ensisijaisesti koneen tai laitteen IP-osoitteeseen ja asiaankuuluviin portteihin. Lisäksi siinä on suojauksia, kuten käyttäjien käyttöoikeusluettelot tai palomuurit yhteyksien hallitsemiseksi. [4, s. 234–238.]

OPC UA PubSub

Teollisuusmaailman siirtyessä Teollisuus 3.0:sta Teollisuus 4.0:aan, jossa kaikki komponentit on kytketty tuotantoverkkoon, kasvaa niiden tuotannon komponenttien määrä, jotka haluavat tuottaa ja kuluttaa dataa jaetussa verkossa. Tämä aiheuttaa yhteysongelmia.

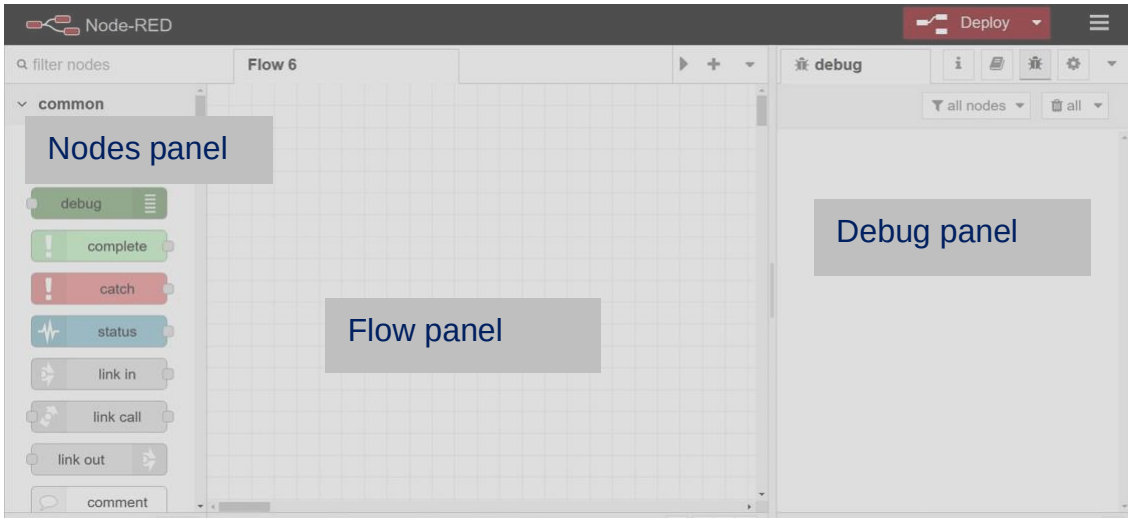
OPC UA:n olemassa olevan asiakas-palvelinviestinnän pyyntö-vastausmekanismin lisäksi, joka vaatii pysyvän yhteyden asiakkaan ja palvelimen välillä, OPC UA:ta tehostetaan lisäviestinnällä Publish-Subscribe-mekanismilla. Publish-Subscribe-mekanismissa palvelin lähettää tietonsa yhteydettömästi verkkoon ja asiakkaat voivat käyttää näitä tietoja ilman resursseja kuluttavaa yhteyttä. [13.]

PubSub-mallissa Publisher-komponentti määrittää ensisijaisesti tietojoukkoja (DataSets), jotka sisältävät muuttujia tai tapahtumien ilmoituksia (EventNotifiers), ja sitten julkaisee verkkoon näitä tietoja tietojoukkoviesteinä (DataSetMessages), jotka sisältävät datamuutoksia tai tapahtumia. Siirrättävät tiedot ovat samanlaisia kuin asiakas-palvelintilaukset, ne vain on järjestetty eri tavalla. Verkossa julkaistujen tietojoukkoviestien tilaajat (Subscribers) voivat kuunnella ja suodattaa tietoja. PubSub-mallissa julkaisijoiden (Publishers) ja tilaajien määrää ei ole rajoitettu. Ne ovat kaikki yhteydessä samaan verkkoon, mutta eivät toisiinsa. [13.]

PubSub-viestintämekanismi on määritelty vaihtoehdoksi OPC UA:n asiakkaan ja palvelimen väliselle perusviestinnälle, mutta useimmissa tapauksissa ne on yhdistetty toisiinsa. PubSub mahdollistaa viestinnän OPC UA -sovellusten välillä. Se käyttää välittäjäpohjaisia viestintäprotokollia, kuten Message Queue Telemetry Transport (MQTT), joka on koneiden välinen viestintäprotokolla (machine-to-machine), josta on nopeasti tulossa teollisen esineiden internetin johtava viestintäprotokolla. Merkityksellisen teollisen viestinnän saavuttamiseksi UA:n PubSub tarjoaa saumattoman OT-IT-integraation pilvipohjaisiin liiketoimintaratkaisuihin tilaamalla (Subcribe) MQTT-välittäjän tai muita olemassa olevia ratkaisuja, jotka mahdollistavat OT-tietojden siirtoa pilviympäristöön ilman toimittajan protokollana toimivien OT-laitteiden konfigurointia tai purkamista. Kenttälaitteiden tietojen yhdistäminen pilvisovellusten kanssa auttaa yrityksiä tekemään nopeita ja tietoisia päätöksiä. OPC UA:n PubSub-mallin tarkoitus on tuoda skaalautuvuutta, parannettua suorituskykyä ja langatonta tiedonvaihtoa reunasta pilveen (edge-to-cloud). [14.]

5 Node-RED

Node-RED on IBM:n kehittämä virtauspohjainen visuaalinen ohjelmointityökalu (flow-based), jolla voidaan helposti luoda web-sovelluksia kaikille ohjelmistosovelluksille, kuten tavallisiin verkkosovelluksiin, verkkosovellusliittymiin ja IoT-tietojen käsittelyyn. Ohjelma perustuu solmuihin (nodes) ja niiden väliseen tapahtumapohjaisiin viesteihin. Solmut ovat yksittäisiä lohkoja, jotka tarjoavat tuloja ja lähtöjä eri solmujen yhdistämiseen. Tarkemmin sanottuna se on prosessipohjainen virtuaalinen ohjelmistoympäristö, joka luo tietovirtoja (data flows) kenttälaitteista pilveen yhdistämällä laitteistoja ja ohjelmistoja. Se soveltuu dataprosessien kirjoittamiseen, mikä helpottaa tietojen käsittelyä. Sen avulla voidaan helposti siirtää tiedot ylemmän tason järjestelmiin, kuten hallintajärjestelmä, keskitetty tiedonkeruu ja pilvipohjainen palvelu. Node-RED koostuu kolmesta peruselementistä: solmupaneelista (node panel), virtauspaneelista (flow panel) ja tieto- ja virheenkorjauspaneelista (Debug panel) (kuva 9). [15, s. 25–35.]



Kuva 9. Node-REDin perusnäkymä [17].

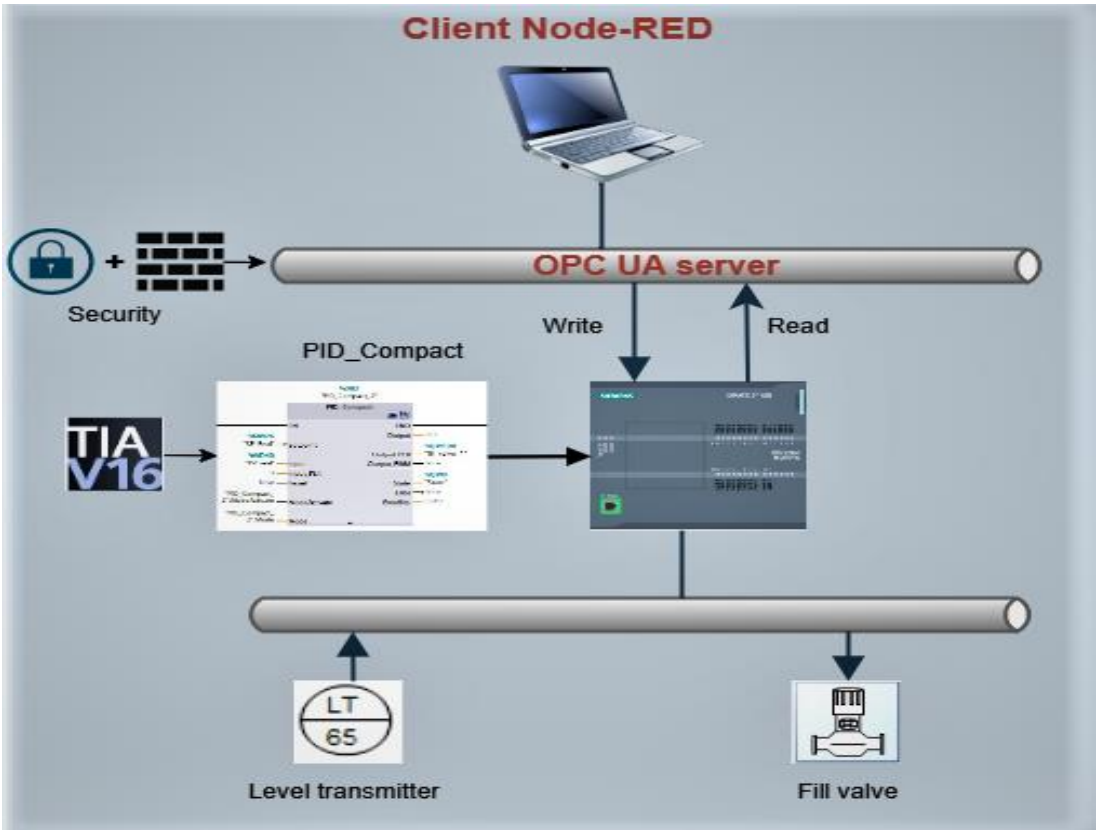
Node-RED-ohjelma pystyy omien solmuina saatavien automaatioprotokollien, kuten OPC UA, MQTT, TCP tai HTTP, avulla kytkemään ohjausjärjestelmiä ja

esimerkiksi valvontasovelluksia yhteen. Valvontasovellus saa tilatietoja laitteistosta sekä muita diagnostiikkaan liittyviä tietoja yksinkertaisella ohjelmoinnin tavalla. Ohjelma voi etsiä laitteita ja kerätä niistä vain tiettyjä muuttujia tiedonhallintasovelluksen tekemiseksi. Node-RED on yksi virtauspohjaisista ohjelmointityökaluista (Flow Based Programming), joka kuvaa sovellusten toimintaa solmuverkkona (node). Prosessointi määritellään jokaisessa solmussa, sille annetaan dataa, prosessointi suoritetaan tätä dataa käyttäen ja data välitetään seuraavaan solmuun. Jokainen sovelluksen solmu suorittaa tietyn tehtävän, kuten tietojen keräämisen, käsittelyn tai lähettämisen ja tietojen yhdistämisen eri koneista (koneen välinen tiedonsiirto tai M2M). [16.]

Node-RED tuli avoimeksi lähdekoodiksi vuonna 2013, ja sitä kehitetään edelleen ilmaisena avoimena lähdekoodina. Siitä tuli yksi JS-säätiön perustamisprojekteista 2016, ja se on sittemmin sulautunut Node.js-säätiön kanssa OpenJS-säätiön luomiseksi vuonna 2019. [15, s. 25–35.] Liite 1 sisältää selityksen ohjelman eri käyttöjärjestelmien lataamisesta.

6 Node-REDin ja OPC UA:n soveltaminen

Opinnäytetyön käytännön osassa esitellään teollisuusalustan digitalisointi, johon on valittu PID-säädin säätämään säiliön nestetasoa. Ohjelma ei saisi olla kuitenkaan liian massiivinen työn toteuttamisen yksinkertaistamiseksi. Siemens tarjoaa valmiina olevia PID-toimintalohkoa STEP 7:ssä, kuten PID_Compact, jonka ohjelmointi on helppoa ja yksinkertaista. Ohjelman koodi on esitetty liitteessä 3. Ajatuksena on hakea dataa PLC:ltä Siemens S7-1200, jonka tehtävänä on ohjata tätä prosessinohjausta ja joka pystyy sisärakennetun OPC UA:n avulla paljastamaan PID:n prosessidatan osoiteavaruudessaan ja asettamaan sen OPC UA -asiakassovelluksen (Node-RED) saataville. Alla olevassa kuvassa 10 on esitetty järjestelmän arkkitehtuuri, joka auttaa ymmärtämään, mitä ollaan suunnittelemassa ja toteuttamassa.

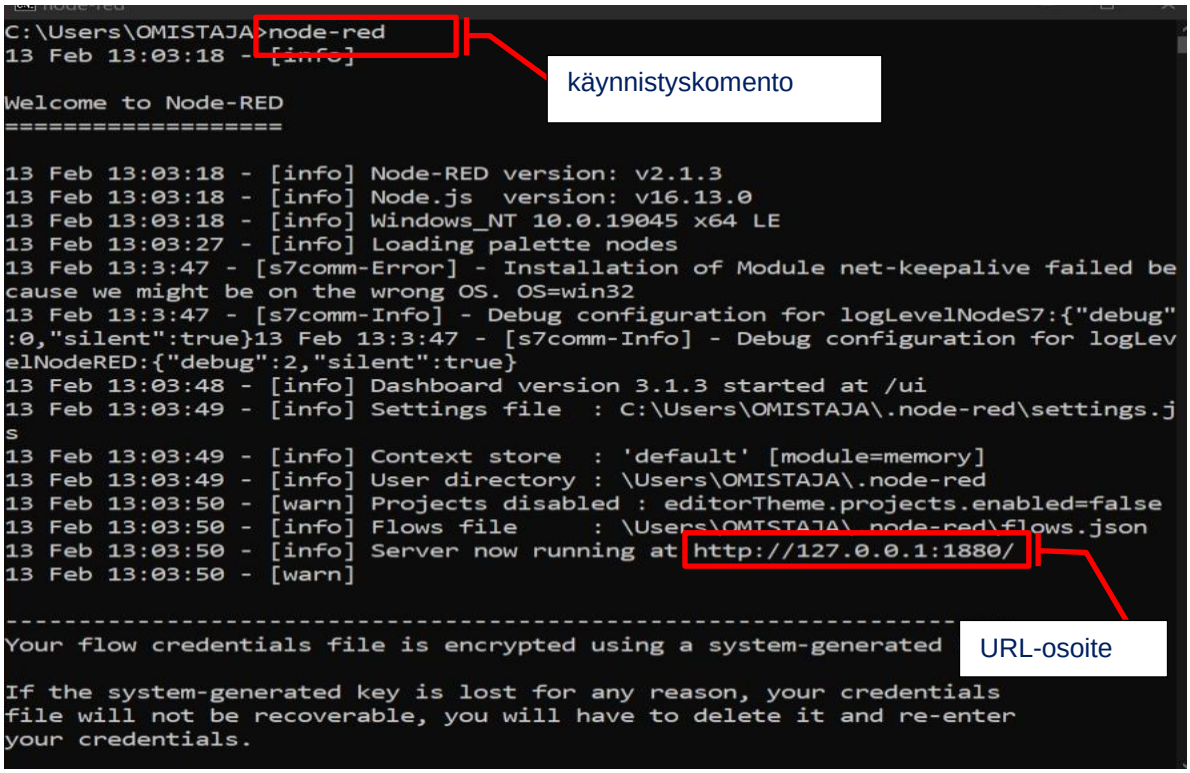


Kuva 10. Node-REDin ja OPC UA -palvelimen integrointi.

OPC UA:n ja Node-REDin integroinnin avulla voidaan helposti luoda vuo (Flow) tietojen lukemista varten PLC:stä ja näyttää tiedot kojelaudalla (dashboard) tai ohjata PLC:tä muiden solmujen syötteen perusteella.

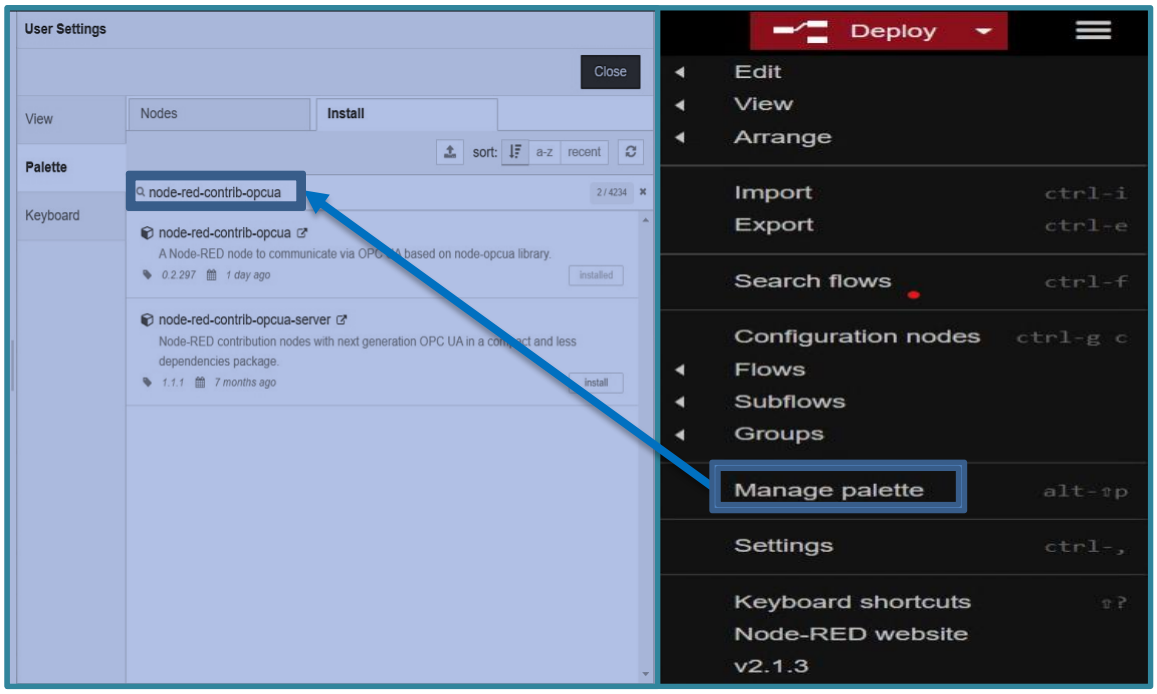
6.1 Tarvittavien solmujen lisääminen Node-REDiin

Kun Node-RED on valmiiksi asennettuna tietokoneelle, se pystytään ottamaan käyttöön taustapalveluksi, joka käynnistyy terminaaliin kirjoitetun komennon (node-red) avulla. Tämä komento komentokehotteessa paljastaa Node-RED-virtauseditorissa käytetyn URL-osoitteen, joka kirjoitetaan selaimen osoiteriville (kuva 11). Näin on mahdollista avata tietokoneen selaimella Node-RED-käyttöliittymä esille.



Kuva 11. Node-REDin käynnistäminen taustapalveluksi [17].

Seuraavaksi asennetaan pakolliset lisäsolmut, kuten kommunikointi PLC-laitteille (”node-red-contrib-opcua”) ja web-liittymä toteutukselle (”node-reddashboard”), jotka eivät sisälly Node-RED-peruspakettiin. Asennus tapahtuu Node-REDin paketinhallintatyökalun (Manage palette) kautta, jonka avulla voidaan etsiä eri tarkoituksiin haluttuja solmuja kirjoittamalla niiden nimet hakukenttään kuvan 12 mukaisesti.



Kuva 12. Lisäsolmujen asennus [17].

Dashboard ja kaikki muut solmut asentuivat yksinkertaisella tavalla painamalla Install-painiketta. Dashboard on web-pohjainen käyttöliittymä (User Interface tai UI), jota käytetään tietojen visualisointiin Node-REDissä. Sen avulla voidaan suunnitella ja luoda käyttöliittymiä, joissa on kaavioita (charts), mittareita (gauges), painikkeita (buttons) ja paljon muita solmuja, jotka mahdollistavat tietojen hallinnan ja visualisoinnin yhdistetyistä laitteista ja palveluista (kuva 13).

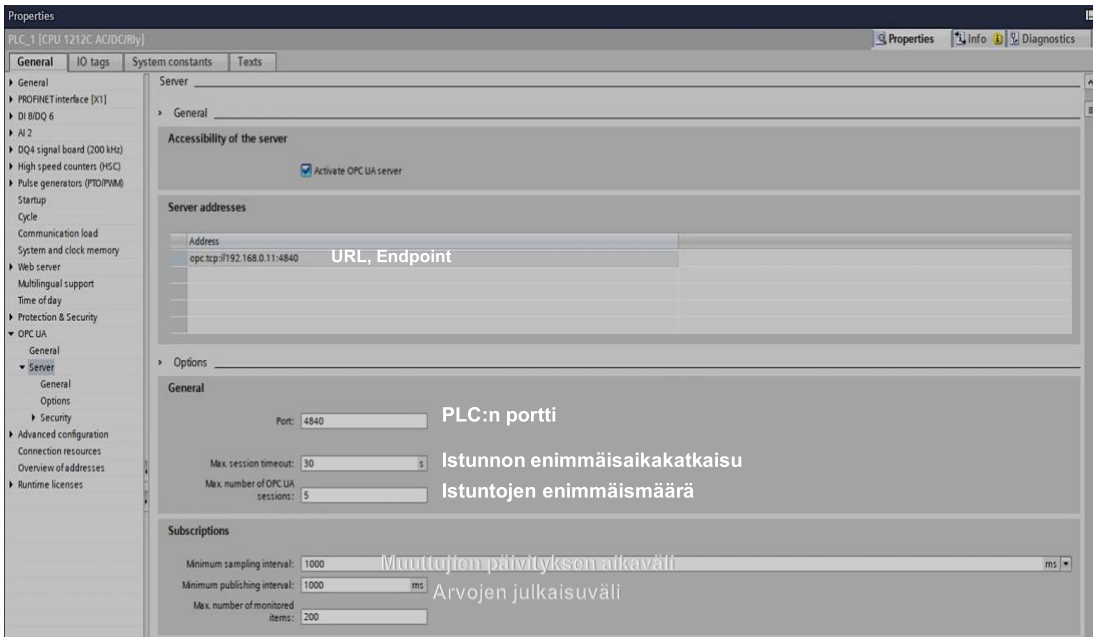
Kuva 13. Dashboardin erilaiset graafiset elementit [17].

Node-RED-dashboardin käyttöliittymää voidaan käyttää minkä tahansa laitteen verkkoselaimella osoitteesta ”http://*koneen IP, missä Node-RED on asennettu*:1880/ui” ja paikallisessa koneessa osoitteesta ”http://localhost:1880/ui”.

6.2 Siemens PLC:n OPC UA -konfigurointi

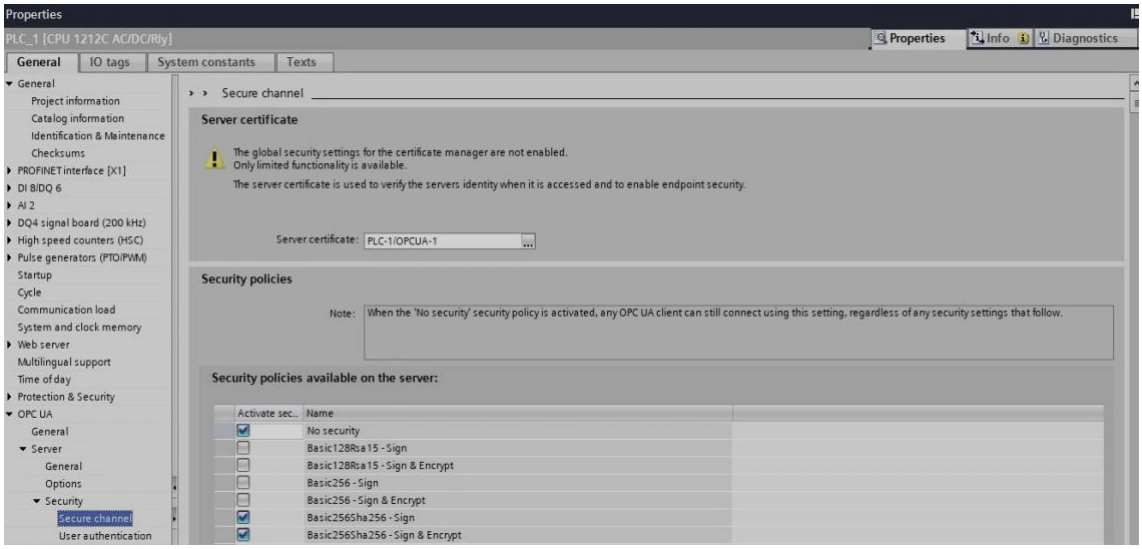
Useimmissa Siemensin PLC:issä, joissa on laiteohjelmisto 4.4 tai uudempi, on sisäänrakennettu OPC UA. Tämän työn tapauksena käytetään S7-1200 -PLC:tä uudella laiteohjelmistolla, johon on valmiiksi asennettu OPC UA. Tässä vaiheessa määritellään OPC UA -palvelimen päätepiste (endpoint URL), suojauskäytännöt (security policies) ja varmenteet (certificates) Tia Portal v16 -ohjelmointiohjelman avulla. Tia Portalia käytetään Siemensin PLC:iden ja automaatiojärjestelmien ohjelmoitiin ja konfigurointiin (liite 2).

OPC UA -palvelin aktivoidaan kohdassa ”Device Configuration” ja ”CPU_1212C”. (→ CPU_1212C → Device Configuration → OPC UA → Activate OPC UA server → OK) (kuva 14).



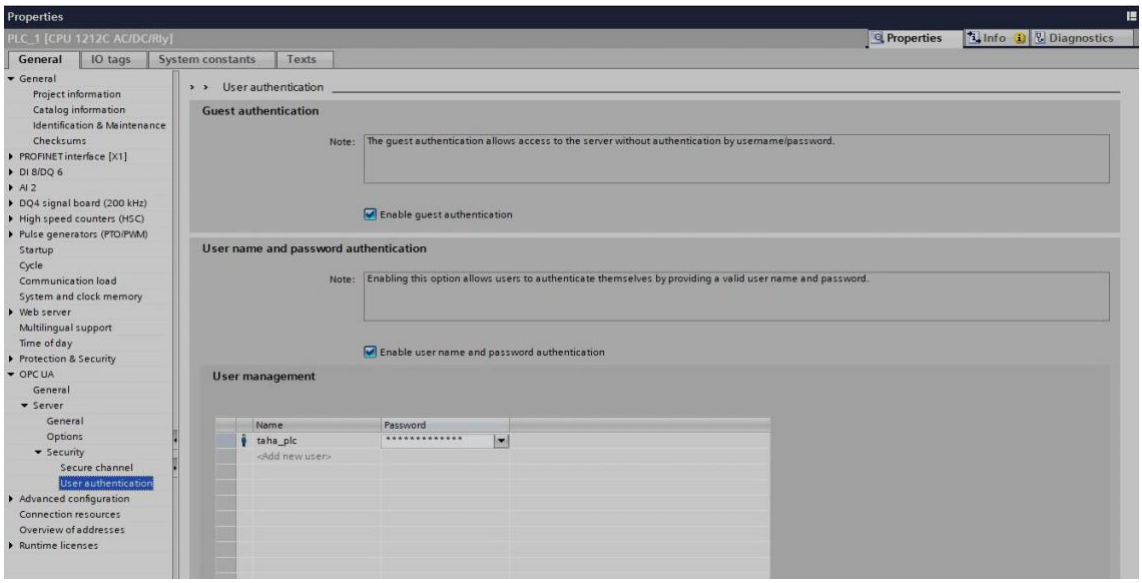
Kuva 14. OPC UA -konfigurointi [18].

Seuraavaksi asetetaan OPC UA -palvelinkäytäntö ja -sertifikaatti palvelimen todentamista varten sekä salaus asiakkaan ja palvelimen välistä viestintää varten (kuva 15) (→ OPC UA → Server → Security → Secure Channel).



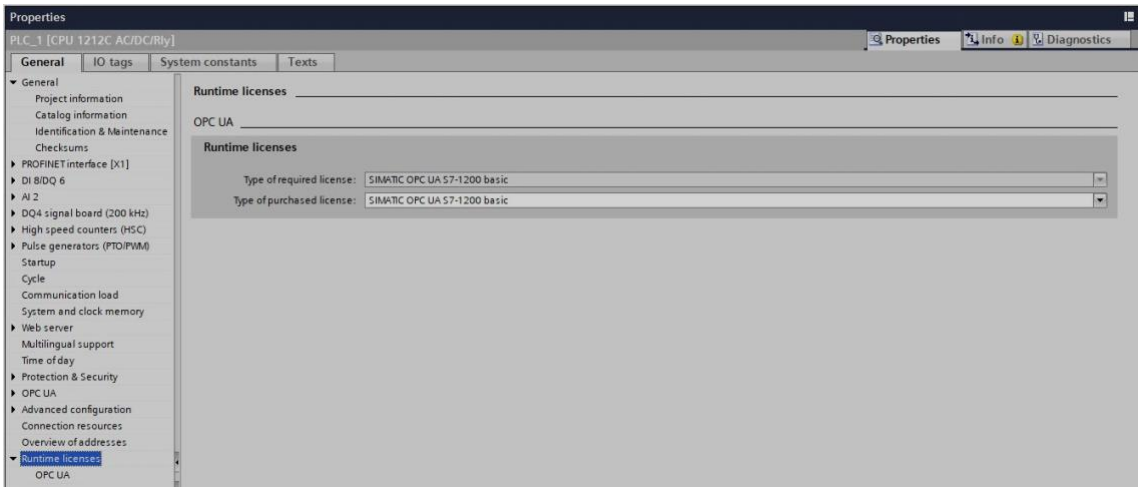
Kuva 15. OPC UA Security [18].

OPC UA -palvelimen käytön yksinkertaistamiseksi jätetään ”Vierastunnistus”- sekä ”Todennus käyttäjänimellä ja salasanalla” -vaihtoehdot päälle. Tämä tarkoittaa, että asiakas-palvelinyhteyden muodostamisessa ei kysytä käyttäjän salasanaa, mutta kun otetaan ”Vierastunnistus”-vaihtoehto pois päältä, silloin salasana, joka on määritelty ”User managerissa”, kysytään (kuva 16) (→ OPC UA → Server → Security → User authentication).



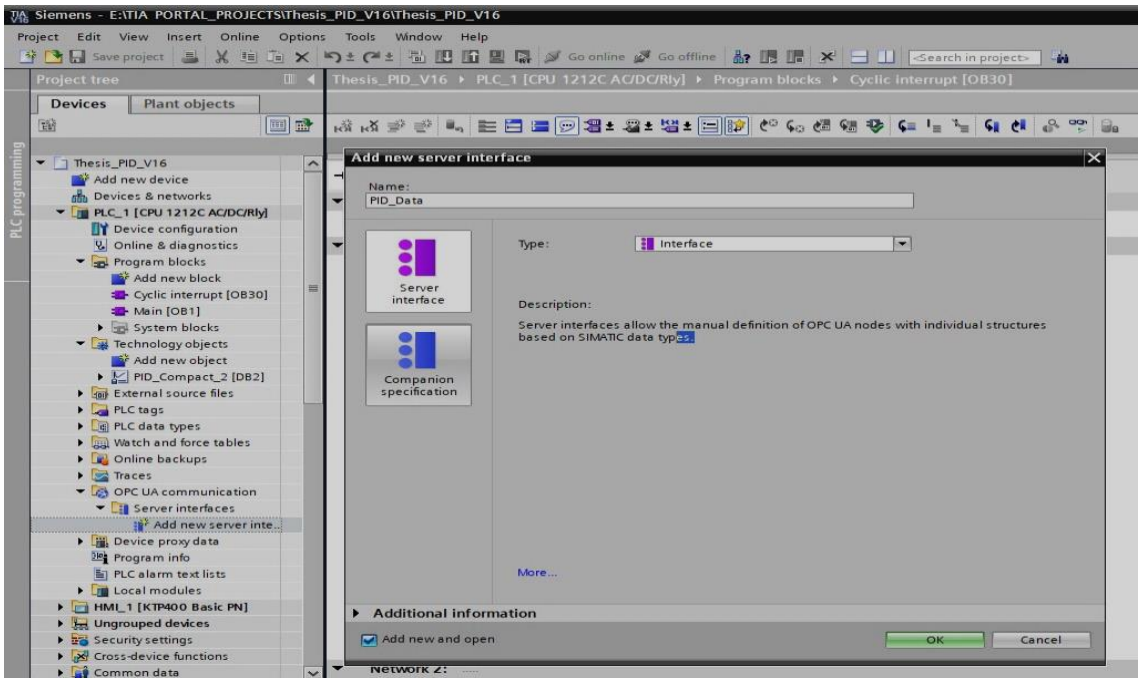
Kuva 16. Käyttäjän todennus [18].

Nyt valitaan ”Ajonaikaiset lisenssit” (→ Licenses Runtime → OPC UA → Type of purchased license → SIMATIC OPC UA S7-1200 basic) (kuva 17).



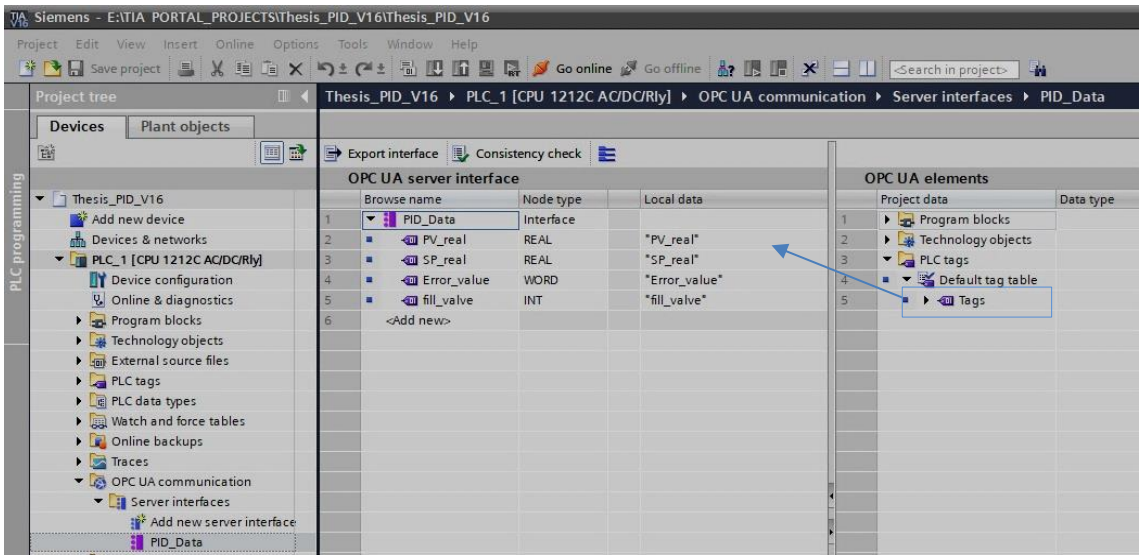
Kuva 17. Ajonaikaiset lisenssit [18].

Tia Portal -ohjelmointiohjelmassa luodaan uusi OPC UA -palvelinrajapinta, joka käytetään liitäntöjen määrittämiseen. Tämä voidaan tehdä avaamalla projektipuussa ”OPC UA -viestintäosio” ja sitten ”Palvelinrajapinnat-kansio”. Napsauttamalla ”Lisää uusi palvelinliittymä” alla oleva ikkuna avautuu. Tämän jälkeen nimetään palvelinrajapinta ja sitten klikataan ok (→ OPC UA communication → Server interfaces → Add new server interface → Name ”PID_Data” → Ok) (kuva 18).



Kuva 18. Palvelinliittymän luominen [18].

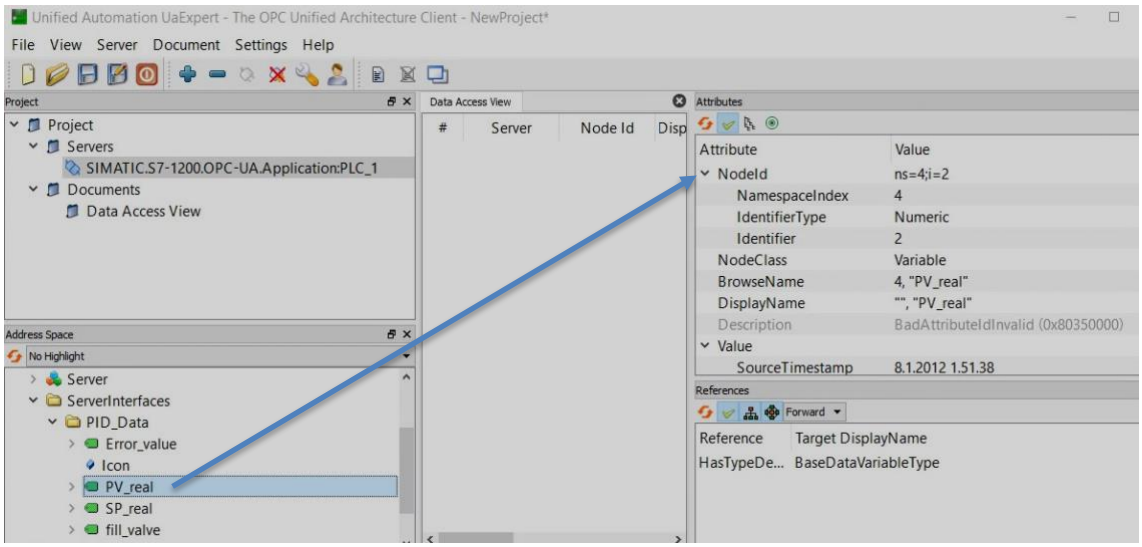
Ensimmäisellä avauskerralla sen jälkeen, kun käyttöliittymä on luotu, siinä ei ole vielä dataa. Tämän takia täytyy vetää (drag and drop) halutut muuttujat (tags) projektin tietolohkoista palvelinliittymään alla olevan kuvan 19 mukaisesti.



Kuva 19. Datan lisääminen palvelinliittymään [18].

Viimeisenä tallennetaan projekti ja sitten ladataan konfiguraation asetukset PLC:hen.

Kuvassa 19 näkyvät visualisointiin valittuja tagit, asetusarvo (SP_real), oloarvo (PV_real), lähtöviesti (fill_valve) ja virhearvo (Error_value). Nämä tagit ovat OPC UA -muuttujat, jotka ovat edustettuina solmuina OPC UA -palvelimen osoiteavaruudessa. Jokaiselle OPC UA -osoiteavaruuden solmulle (node) on määritelty yksilöllisesti solmutunniste (Node ID). Solmun tunniste (ID) koostuu nimiavaruudesta (ns) eri järjestelmien tunnisteiden erottamiseksi ja tunnisteesta, joka voi sisältää numeerisen arvon (i) tai merkkijonon (s). Solmutunnuksen rakenne on esimerkiksi muodossa ”ns = 3; i = 5”. Seuraava kuva näyttää muuttujien attribuuttien rakenteen UaExpert-ohjelmaa käyttäen.

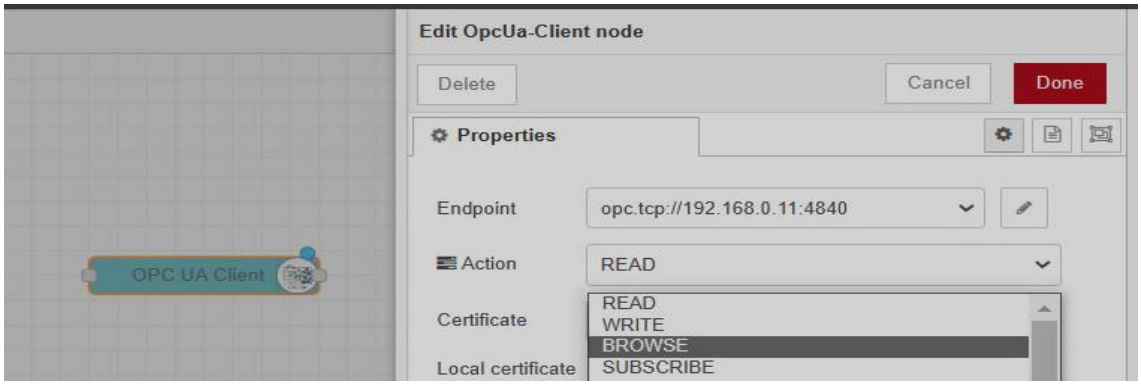


Kuva 20. Muuttujien attribuuttien tarkastaminen [20].

Solmujen tunnisteiden sekä palvelimen osoiteavaruuden tarkastelemiseen on tämän työn tapauksessa käytetty apuna ilmaista saatavilla olevaa UaExpert-ohjelmaa (OPC UA -testClient), joka tukee OPC UA -ominaisuuksia [19].

6.3 Tietojen visualisointi Node-REDillä OPC UA:n kautta

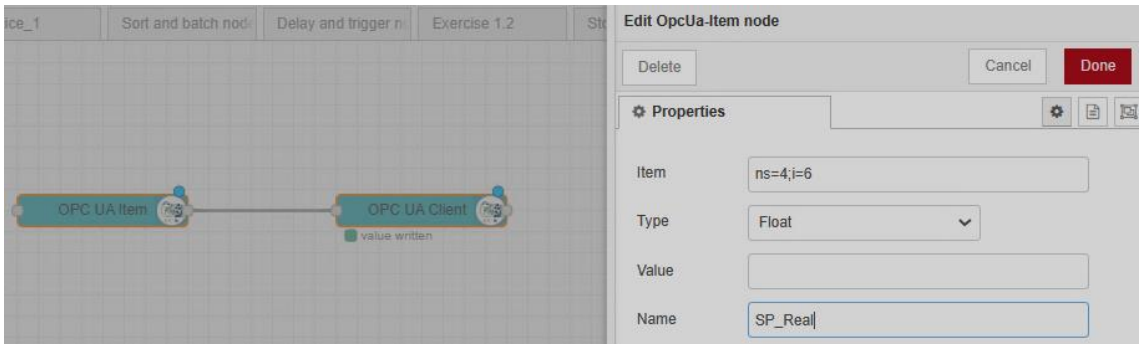
Jotta Node-RED voi kommunikoida Siemens S7-1200 -PLC:n kanssa OPC UA:n kautta, Node-REDissä luodaan vuo, joka yhdistää S7-1200:aan. Tämä tehdään käyttämällä OPC UA-asiakas-solmua, jota käytetään yhteyden muodostamiseen PLC:n OPC UA -palvelimeen. Tämän jälkeen OPC UA-asiakas-solmu konfiguroidaan OPC UA -palvelimen URL-osoitteella (kuva 14, luku 6.2), joka on yleensä PLC:n IP-osoite. Alla olevassa kuvassa 21 ovat solmun parametrit.



Kuva 21. OPC UA-asiakas-solmun konfigurointi [17].

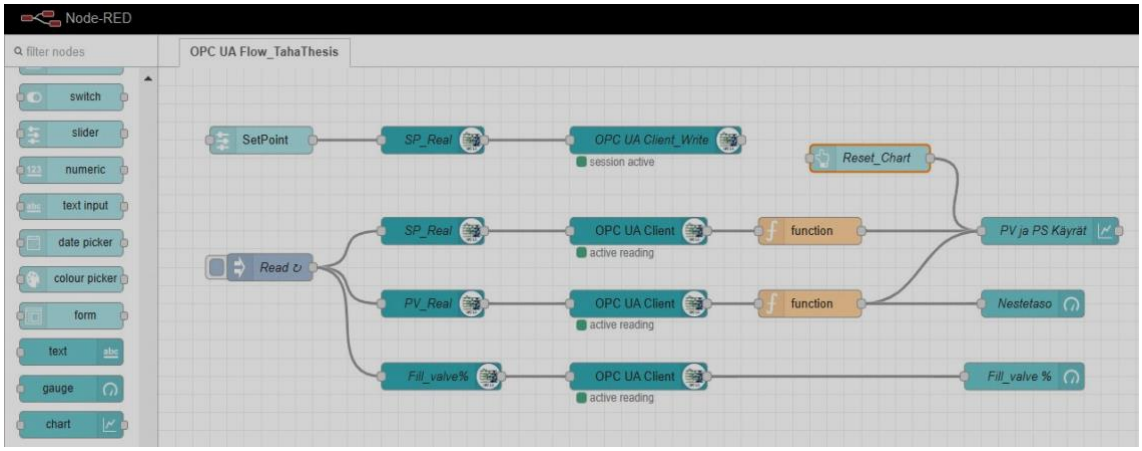
Kuvassa 21 olevan toiminnon (Action) alasvetovalikosta voidaan valita haluttu toiminto, jonka avulla on helppoa lukea, kirjoittaa tai tilata OPC UA -muuttujia.

Muuttujien nimet ja tietotyypit määritellään OPC UA-Item -solmun avulla solmutunnuksella (Node ID) ja solmun tietotyypillä, jotka on selvitetty aiemmin UaExpert ohjelmalla (kuva 22).



Kuva 22. Muuttujien nimien ja tietotyyppien asettaminen [17].

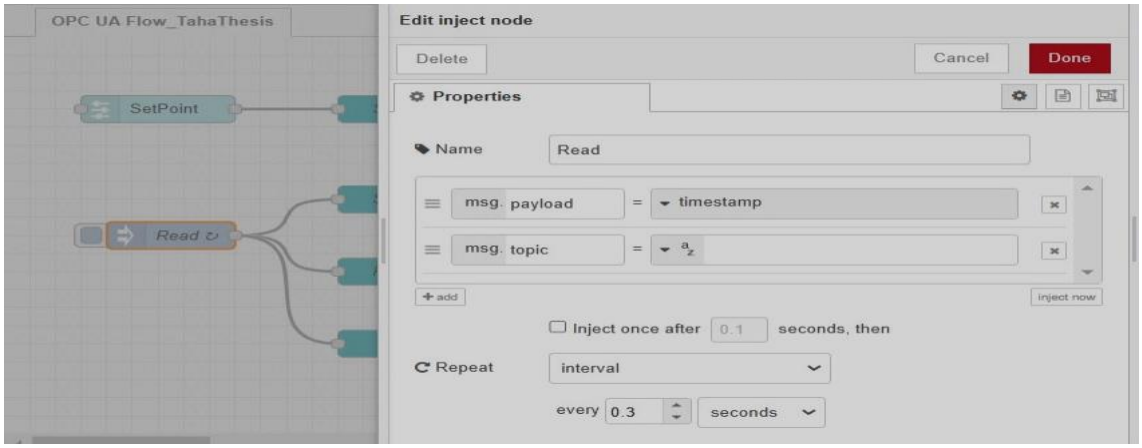
Seuraavaksi luodaan yksinkertainen vuo, jonka perusteella luetaan asetusarvo (SP_Real) PLC:sta sekä kirjoitetaan staattinen asetusarvo PLC:hen käyttämällä liukusäädintä (slider). Tässä esimerkissä vuo lähettää asetusarvon ja prosessiarvotiedot viestinä yksinkertaiseen kaavioon (chart). Tätä varten tarvitaan injektiosolmu, joka käynnistyy toistuvasti ennalta määrätyn ajan välein, ja funktiosolmu, joka muuttaa muuttujien nimet solmutunnuksestaan valituiksi nimiksi, jotka näkyvät PID-trendissä muodossa SP ja PV. Käyttöliittymään lisätään myös kaksi mittaria, joista ensimmäinen näyttää säiliön nestetason ja toinen näyttää täyttöventtiilin avautumisen prosentteina. Viimeisenä tulee painike, joka nollaa PID-trendin käyrät (kuva 23).



Kuva 23. Vuo, jonka kautta yhteys PLC:hen tehty [17].

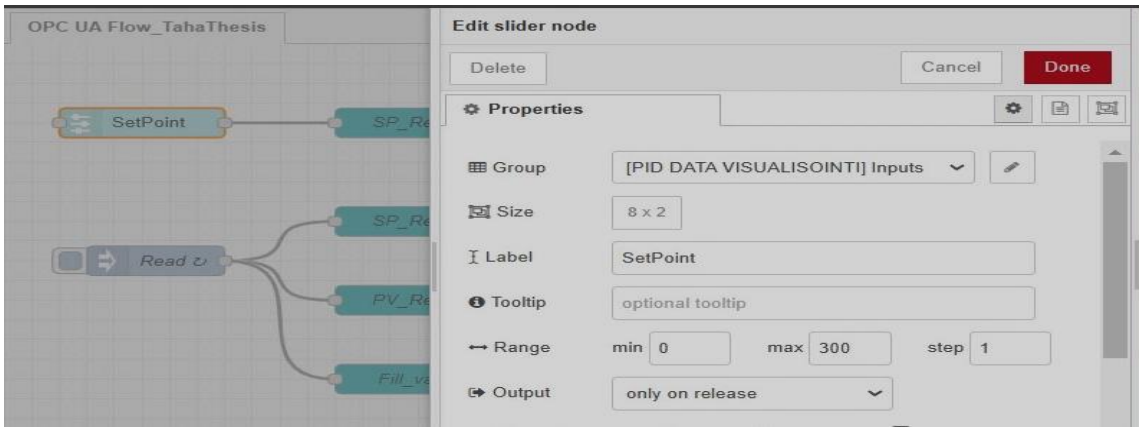
Kuvassa 23 näkyy, että OPC UA-asiakas-solmujen alapuolella on vihreä neliö, joka muuttui punaisesta vihreäksi ja sen eteen tulostui teksti ”active”. Tämä tarkoittaa, että Node-RED kommunikoi onnistuneesti S7-1200-PLC:n kanssa verkon yli, kun luotu vuo oli käynnissä Node-RED-alustalla. Lisäksi kuvan vasemmalla palkissa on Dashboard-valikko, josta löytyy saatavilla olevat solmut, kuten mittari, kaavio, funktio, nappi jne.

Solmujen parametrien määrittäminen tapahtuu tuplaklikkaamalla hiirellä tarkoitetun solmun päällä, minkä jälkeen avautuu ruudulle ikkuna, jossa nämä parametrit määritellään (kuva 24).



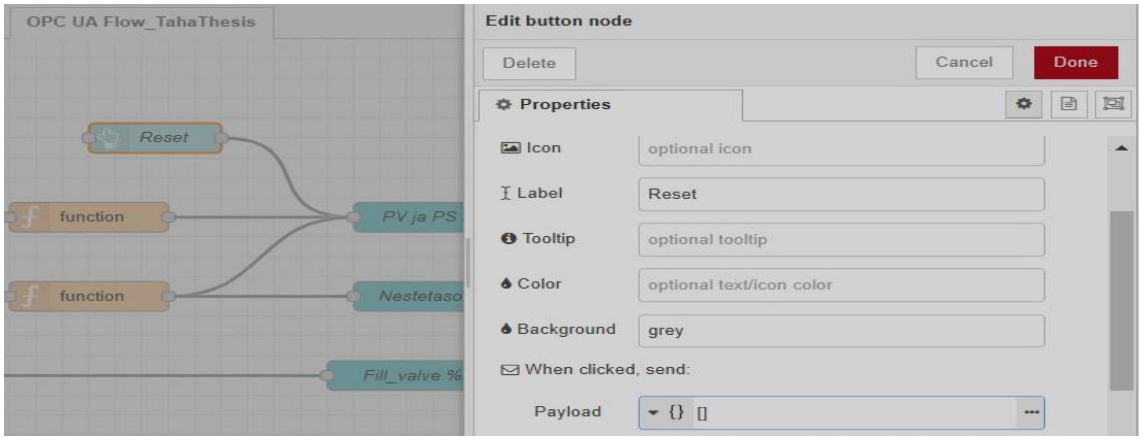
Kuva 24. Injektiosolmun nimi ja toistovälin määrittäminen [17].

Ryhmävälilehti-vaihtoehdon (Group) avulla määritellään, millä nimellä käyttöliittymäsivu nähdään. Oletusnimi on ”Home”, joka tässä tapauksessa muutetaan ”PID DATA VISUALISOINTI” (kuva 25).



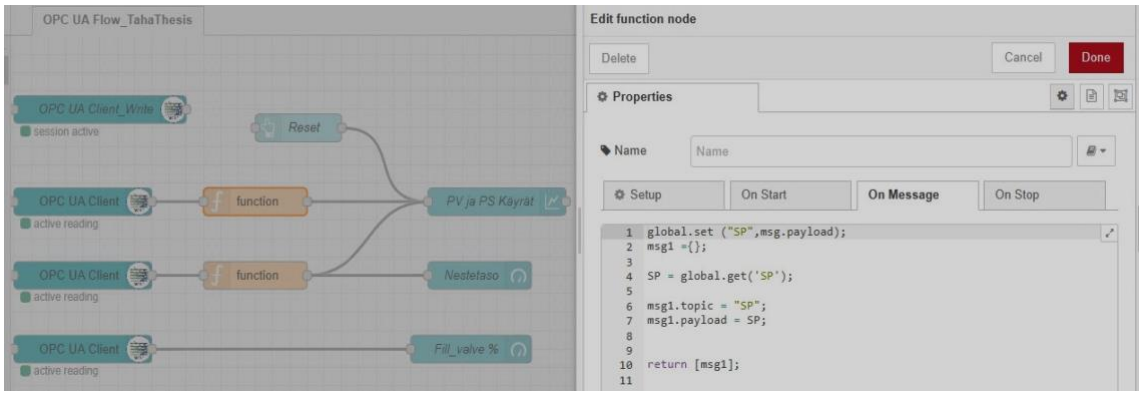
Kuva 25. Nimen (label) ja alueen (range) asettaminen liukusäätimelle [17].

Kuvassa 26 näkyy, kun Reset-nappia painetaan, se lähettää tyhjän JSON-tiedoston ([]) seuraavaan linkitettyyn solmuun, joka tässä tapauksessa on kaaviosolmu, joka resetoi kaavion oletusarvoon.



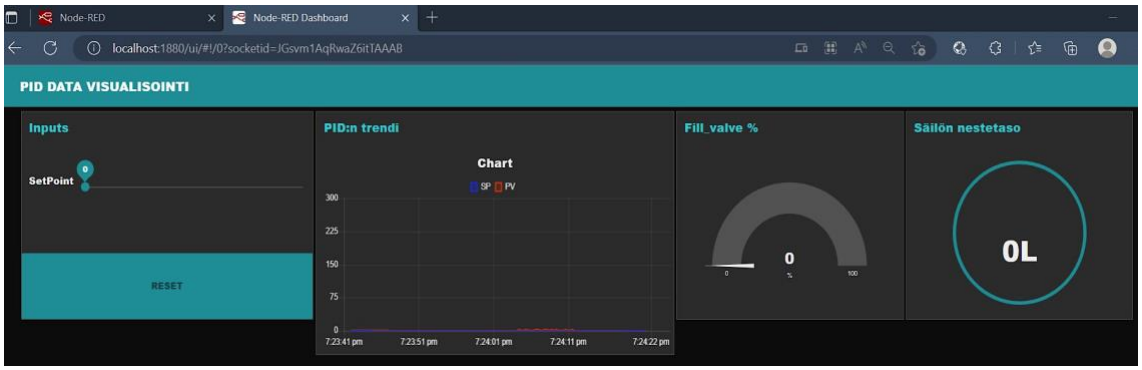
Kuva 26. Toiminallisuuden asettaminen Reset-napille [17].

Kuvassa 27 nähdään kuinka OPC UA-asiakas-solmun lähettämä viestin muodon, joka oli alun perin muuttujan SP_Realin solmutunnuksen muodossa, muutetaan funktiosolmulla asettamalla ensin viesti (msg.payload) globaaliksi muuttujaksi, jonka annettu uusi nimi on ”SP”, ja sitten Return-komennolla viesti palautetaan vuolle uudella nimellä. Sama tehdään muuttujan PV_Realin solmutunnuksen kanssa.



Kuva 27. Funktiosolmun määrittäminen [17].

Näitten solmujen parametrien asettamisen jälkeen painetaan ”Done” ja sitten ”Deploy”-nappia, jolloin vuo tallentuu. Mikäli vuon ohjelmassa ei ole mitään virheitä, ohjelma käynnistyy automaattisesti tallennuksen jälkeen. Tämän jälkeen voidaan käyttää Web-käyttöliittymää laitteen verkkoselaimella osoitteesta localhost:1880/ui (kuvat 28–29).



Kuva 28. Luotu web-käyttöliittymä [17].

Kuva 29. Säätö-prosessin tilatieto ajan kuluessa [17].

Kuvassa 29 nähdään, että on mahdollista luoda käyttöliittymälle erilaisia visualisointielementtejä osoittamaan prosessin tilatiedot. Heti kun asetusarvo muutetaan arvoon 100, säätöprosessi alkaa muuttumaan saavuttaakseen halutun arvon. Alussa nähdään, että täyttöventtiili aukeaa 100 %. Sitten kun prosessiarvo lähestyy asetusarvoa, venttiili alkaa sulkeutumaan, kunnes prosessiarvo on yhtä suuri kuin asetusarvo. Silloin venttiili on mennyt 100 % kiinni.

7 Yhteenveto

Opinnäytetyön tuloksena saatiin onnistuneesti liitettyä Node-RED Siemens S7-1200 -PLC:hen OPC UA:n kautta. Luodun Node-RED-käyttöliittymän avulla pystyttiin manipuloimaan PLC-tuloja antamalla haluttu asetusarvo PID-säätöjärjestelmälle, joka puolestaan palauttaa prosessin etenemisen tilan.

Työn tulos osoitti myös, että Node-RED on tehokas visuaalinen ohjelmointityökalu, jonka avulla voidaan helposti luoda ja ottaa käyttöön verkkoselaimella tiedonhallintasovelluksia, joita perinteiset ohjelmoinnin rajoitukset eivät rajoita. Lisäksi Siemens PLC:n vankan tiedonsiirtoprotokollan OPC UA:n ja Node-REDin integrointi osoitti potentiaalinsa tehostaa tiedonsiirto-, käsittely- ja visualisointiprosessia, mitkä mahdollistavat teollisuusautomaatiomaailman muuttumista rakentamalla skaalautuvia automaatioratkaisuja muuttuvien ympäristöjen vaatimuksesta riippumatta.

Työn edistyessä totesin, että aikarajoitusten ja aiheen pääasiallisten näkökohtien painopisteen vuoksi tietyt opinnäytetyön osa-alueet eivät ole yhtä merkittäviä kuin toiset. Sen takia oli minun mielestäni parempi kiinnittää enemmän huomiota ja keskittymistä tärkeimpiin näkökohtiin ja varmistaa, että on tarpeeksi aikaa tutkia näitä keskeisempiä kohtia perusteellisesti. Tämä koskee Factory IO -osaa, joka ei ollut niin tärkeä kuin Node-REDin ja OPC UA:n integrointi, koska se on vaan tehdassimulointiohjelmisto, jota käytetään simuloimaan virtuaalisesti rakennettuja automaation ohjausjärjestelmiä.

Lähteet

1 What is OPC. Verkkoaineisto. OPC Foundation. < https://opcfoundation.org/about/what-is-opc/>. Luettu 26.12.2022.

2 Supervisory Control and Data Acquisition Approach in Node-RED: Application and Discussions. 2020. Verkkoaineisto. Department of Automation and Applied Informatics. <https://webcache.googleusercontent.com/search?q=cache:lL5IDFQnHbMJ:https://mdpi-res.com/d_attachment/IoT/IoT-01-00005/article_deploy/IoT-01-00005-v2.pdf%3Fversion%3D1597109733&cd=2&hl=fi&ct=clnk&gl=fi>. 10.8.2020. Luettu 19.1.2023.

3 Classic. Verkkoaineisto. OPC Foundation. <https://opcfoundation.org/about/opc-technologies/opc-classic/>. Luettu 27.12.2022.

4 Mahnke, Wolfgang; Leitner, Stefan-Helmut & Damm, Mattias. 2009. OPC UA Unified Architecture. Germany: Springer.

5 Introduction to Classic OPC. Verkkoaineisto. Unified Automation. <https://documentation.unified-automation.com/uasdkcpp/1.3.2/L2ClassicOpc.html>. Luettu 30.12.2022.

6 Knapp, Eric D.; Langill, Joel Thomas. 2011. Industrial Network Security. USA: Elsevier Inc.

7 History. Verkkoaineisto. OPC Foundation. <https://opcfoundation.org/about/opc-foundation/history/>. Luettu 16.1.2023.

8 UA Part 1: Overview and Concepts. 2022. Verkkoaineisto. OPC Foundation. <https://reference.opcfoundation.org/Core/Part1/v105/docs/>. 11.1.2022. Luettu 26.1.2023.

9 UA Part 3: Address Space Model. 2022. Verkkoaineisto. OPC Foundation. <https://reference.opcfoundation.org/Core/Part3/v105/docs/>. 11.1.2022. Luettu 29.1.2023.

10 UA Part 4: Service Set Model. Verkkoaineisto. OPC Foundation. <https://reference.opcfoundation.org/Core/Part4/v105/docs/4.1>. Luettu 1.2.2023.

11 UA Information Model – Concept. Verkkoaineisto. GitBook. <https://commsvr.gitbook.io/ooi/semantic-data-processing/informationmodelconcept>. Luettu 1.2.2023.

12 UA Part 2: OPC UA security architecture. Verkkoaineisto. OPC Foundation. <https://reference.opcfoundation.org/Core/Part2/v105/docs/4.5>. Luettu 3.2.2023.

13 UA Part 14: PubSub. 2022. Verkkoaineisto. OPC Foundation. <https://reference.opcfoundation.org/Core/Part14/v105/docs>. 11.1.2022. Luettu 05.02.2023.

14 Important Use Case of OPC UA PubSub. 2022. Verkkoaineisto. OPC CONNECT. <https://opcconnect.opcfoundation.org/2022/09/important-use-cases-of-opc-ua-pubsub/>. 09.2022. Luettu 6.2.2023.

15 Taiji, Hagino. 2021. Practical Node-RED Programming. Bermingham: Pakt Publishing Ltd.

16 Node-RED: Low-code programming for event-driven applications. Verkkoaineisto. Node-RED. <https//nodered.org>. Luettu 8.2.2023.

17 Kuvakaappaus Node-RED-ohjelmasta. Versio 2.1.3. Node-RED.

18 Kuvakaappaus Tia Portal -ohjelmasta. Versio 16. Siemens.

19 UaExpert. Verkkoaineisto. Unified Automation. <https://documentation.unified-automation.com/uaexpert/1.6.0/html/gui.html>. Luettu 10.3.2023.

20 Kuvakaappaus UaExpert-ohjelmasta. Versio 1.6.3. Unified Automation.

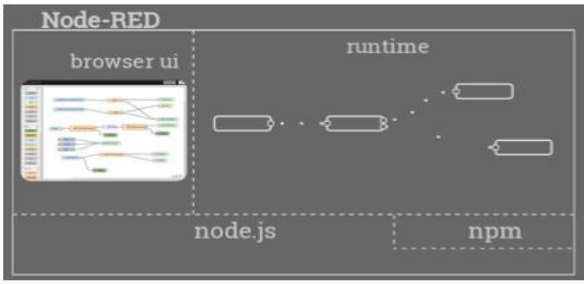
STIIMA. Node-RED Tutorial for linked-data.

Node-RED Tutorial for linked-data

Walter Terkaj, STIIMA-CNR

Node-RED is a powerful tool for building Internet of Things (IoT) applications with a focus on simplifying the 'wiring together' of code blocks to carry out tasks. It uses a visual programming approach that allows developers to connect predefined code blocks, known as 'nodes', together to perform a task. The connected nodes, usually a combination of input nodes, processing nodes and output nodes, when wired together, make up 'flows'. (Link)

Node-RED provides a web browser- based flow editor, which can be used to create JavaScript functions. Elements of applications can be saved or shared for re-use. The runtime is built on Node.js. The flows created in Node-RED are stored using JSON which can be easily imported and exported for sharing with others. By understanding Node-Red, IoT development can be accelerated without unnecessary coding.



References

- Part of the contents in this tutorial can be found on the Node-Red website: https://nodered.org/
- Installation instructions can be found at: https://nodered.org/docs/gettingstarted/installation
- Node RED Programming Guide: http://noderedguide.com/
- Introduction to Node-RED: http://www.steves-internet-guide.com/node-red-overview/

Installation of Node-RED

In order to able to work with Node-RED, you should first install node.js. It is recommended the use of Node.js **LTS 8.x or 10.x**. Node-RED no longer supports Node.js 6.x or earlier. You can download the latest version of node.js from its website choosing which operating system you are using: https://nodejs.org/en/download/

Linux / OsX:

Once installed node.js, open the *terminal* window and run the following commands.

To check your version of Node.js

```
node –v
```

The easiest way to install Node-RED is to use the node package manager, npm, that comes with Node.js.
Installing as a global module adds the command node-red to your system path:

```
sudo npm install -g --unsafe-perm node-red
```

Windows:

Run the downloaded MSI installer of Node.js. Local administrator rights are needed. Accept the defaults settings when installing. After installation completes, close any open command prompts and re-open to ensure new environment variables are picked up.

Once installed, open a command prompt and run the following command to ensure Node.js and npm are installed correctly.

```
node --version && npm --version
```

You should receive back output that looks similar to:

v8. 9. 0

5. 5. 1

Installing Node-RED as a global module adds the command node-red to your system path. Execute the following at the command prompt:

```
npm install -g --unsafe-perm node-red
```

If you have installed Node-RED as a global npm package, you can launch node-red in the command prompt (Windows):

```
c:\>node-red
```

or in the terminal (Linux):

```
$ node-red
```

This will output the Node-RED log to the terminal. You must keep the terminal or command prompt open in order to keep Node-RED running. Note that running Node-RED will create a new folder in your %HOMEPATH% folder called node-red. This is your userDir folder, think of it as the home folder for Node-RED configuration for the current user.

You can then open the Node-RED graphical editor by pointing your browser at

http://localhost:1880

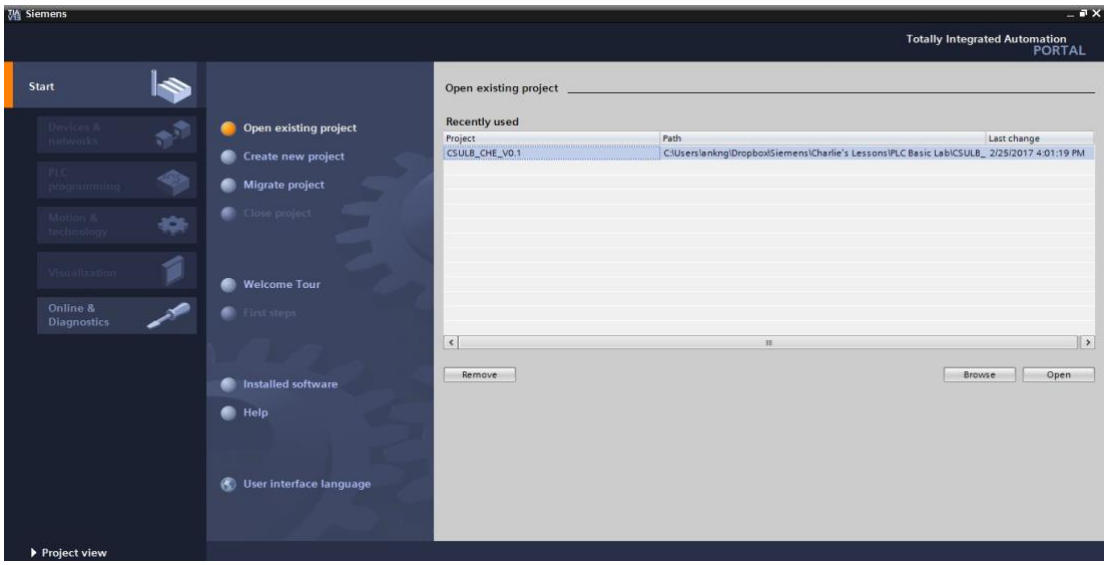
Gothightech. Getting Started with Tia Portal.

Getting Started with TIA Portal Introduction with TIA Portal

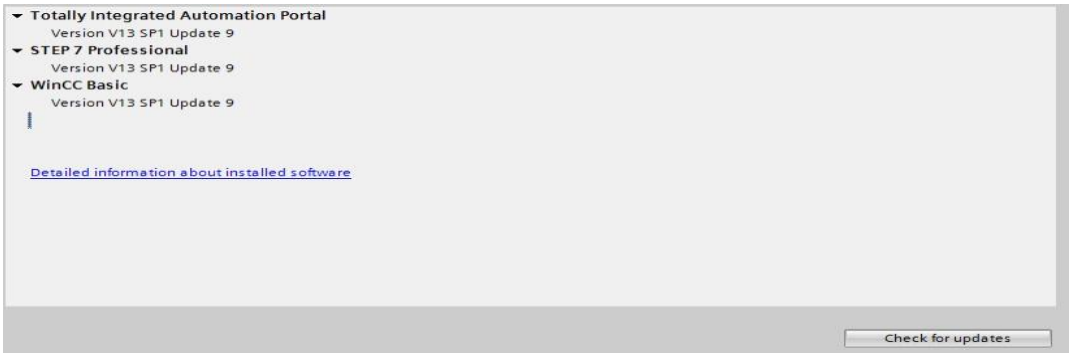
This lab will introduce you to the basics of SIMATIC STEP 7 (TIA Portal). The program allows students to design electrical networks with various elements to control certain processes.

Create a New Project

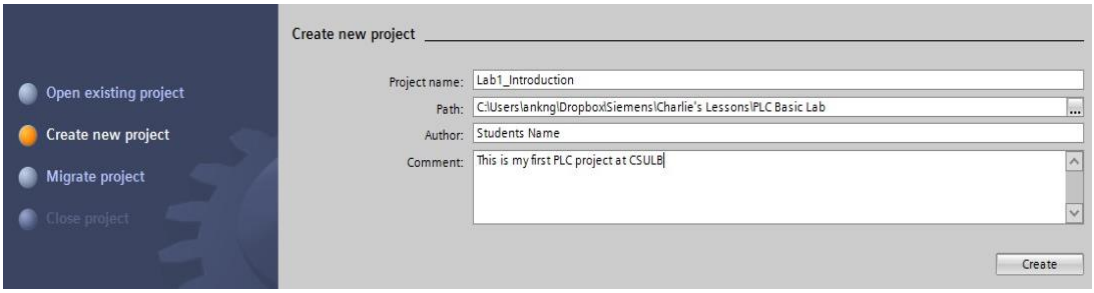
To create a project in TIA Portal, first launch the program on your desktop, if you don't see it, search TIA Portal in the windows search bar at the bottom left of your screen. After the portal successfully loads, you should be able to see a display like below.



As you can see from the menu bar, we have several options to choose from: **Open existing project** and **create new project** are self-explanatory. **Migrate project** takes mature PLC programs and migrate it to new PLC programs. **Welcome Tour** shows you the features of TIA portal, if you are interested and have time to spare, you can proceed with that. **Installed Software** shows you the currently software that is installed, which is important to check to see if all your software is up to date. Below is an example of the screen, showing which software is installed and their versions.



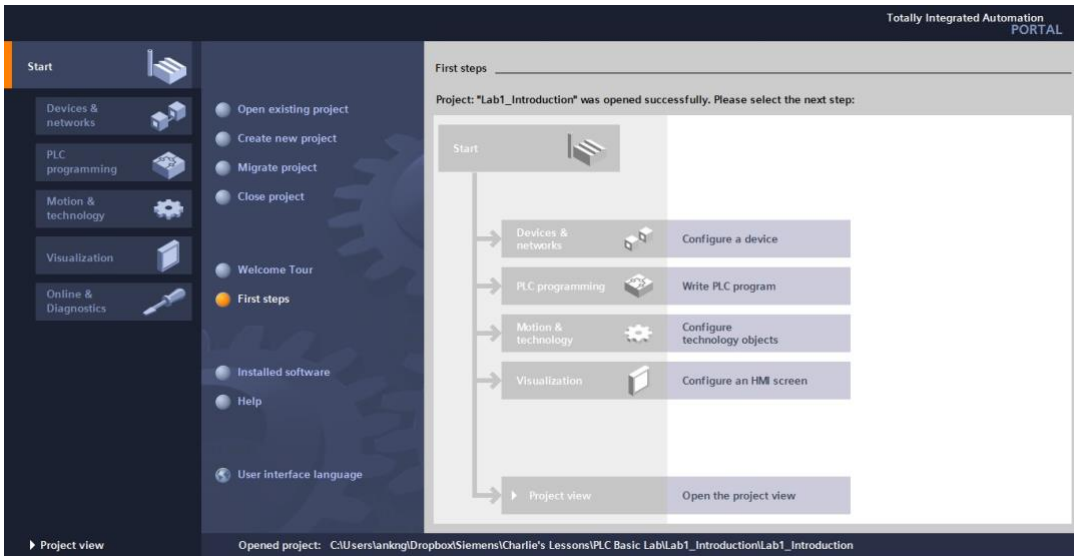
Let's go ahead and proceed with creating a new project, next enter the required information as shown below, take note of the path where you save the project so opening it next time will be much easier.



Click create

Configure Device & Network

Now you should the new addition option on the menu, **First Steps**, here you will see some other options such as:
Configure a device – Allows you to link or set up your PLC (Programmable Logic Controller)
Write PLC program – Allows you to create network
Configure technology objects – Allow students to work in motion modules and drives
Configure an HMI screen – Allows you to create a HMI screen (Human Machine Interface) **Open the project view** – Allows you to access the project view without any of the above options, useful for returning to your older projects.

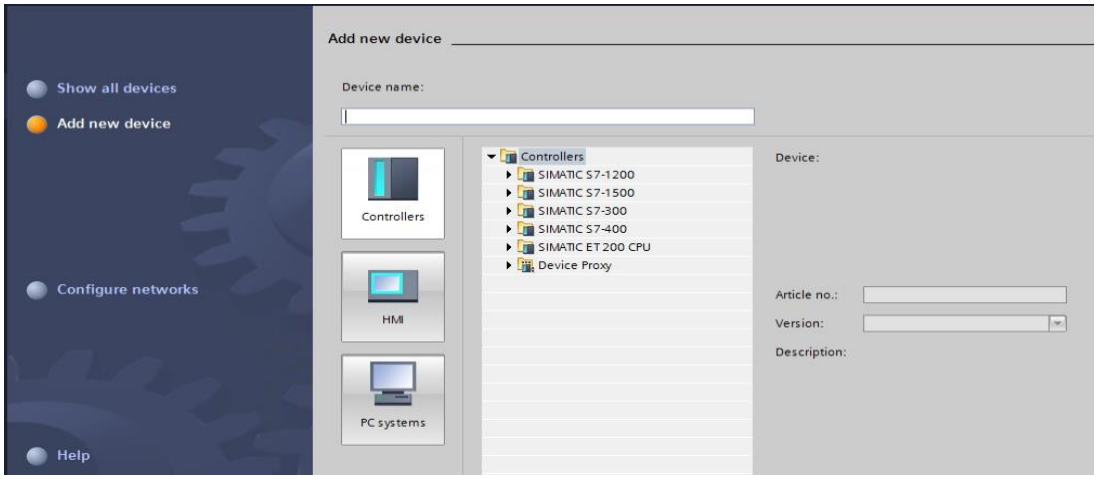


Since it is our first program, go ahead and click on **configure a device**. You can see two options:

Show all devices- Show all the devices you have on your project

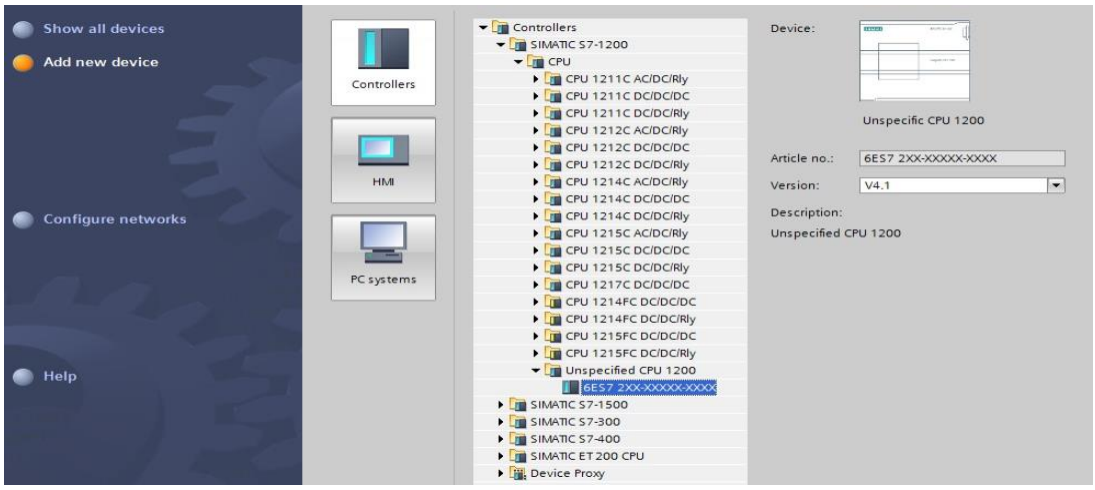
Add new device-Configure a new device

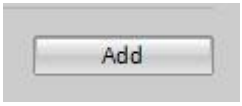
Click on **Add new device**. You can see a similar display as the bottom figure. Here you can configure your Controllers (which are PLCs), HMI screens, and PC systems. Here you can name your PLC however you like. Such as Ryan's PLC 0.1



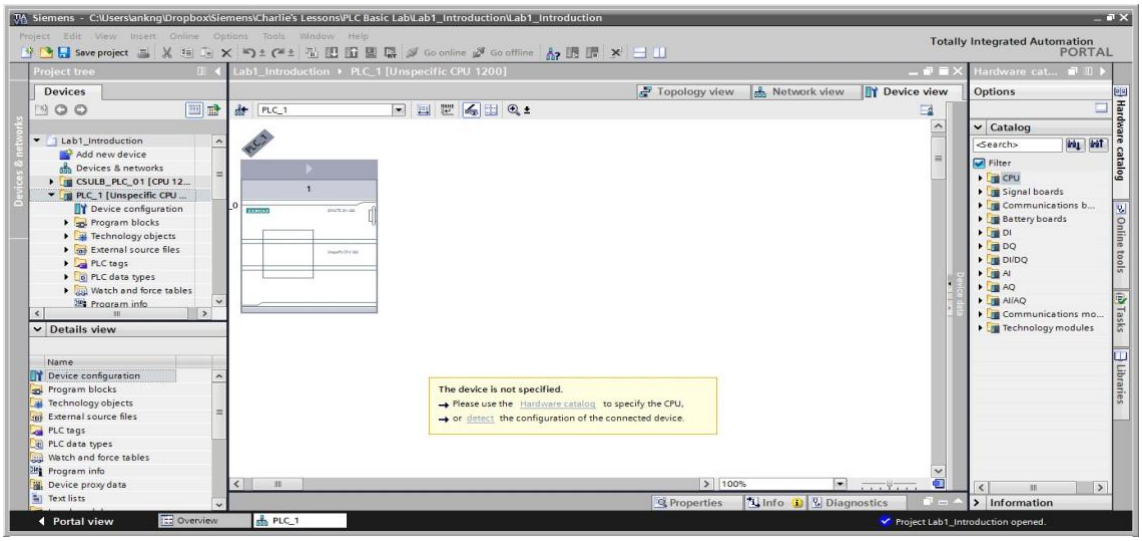
Click on **Controllers**, you will see a drop-down menu that has all sorts of different types of PLC. The one we will be using is one of the SIMATIC S7-1200, so click on the small black arrow next to SIMATIC S7-1200 to get another drop menu one. Continue until you see

6ES7 2XX-XXXXX-XXXX. At this point, if it has not been connected yet, connect the Ethernet cable to your computer.

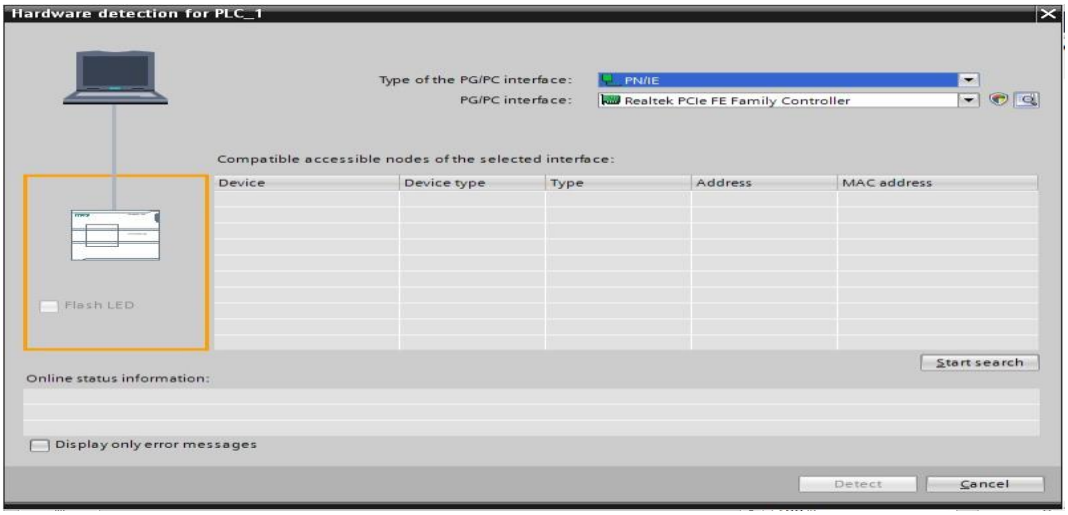


Scroll down a bit to see the add button and click it. 

After successfully adding the PLC, the Portal View will be displayed, this is the main screen you will work on. Currently your PLC is not configured yet, click detect to configure your device. Make sure you have connection between your PLC and your Computer via Ethernet.

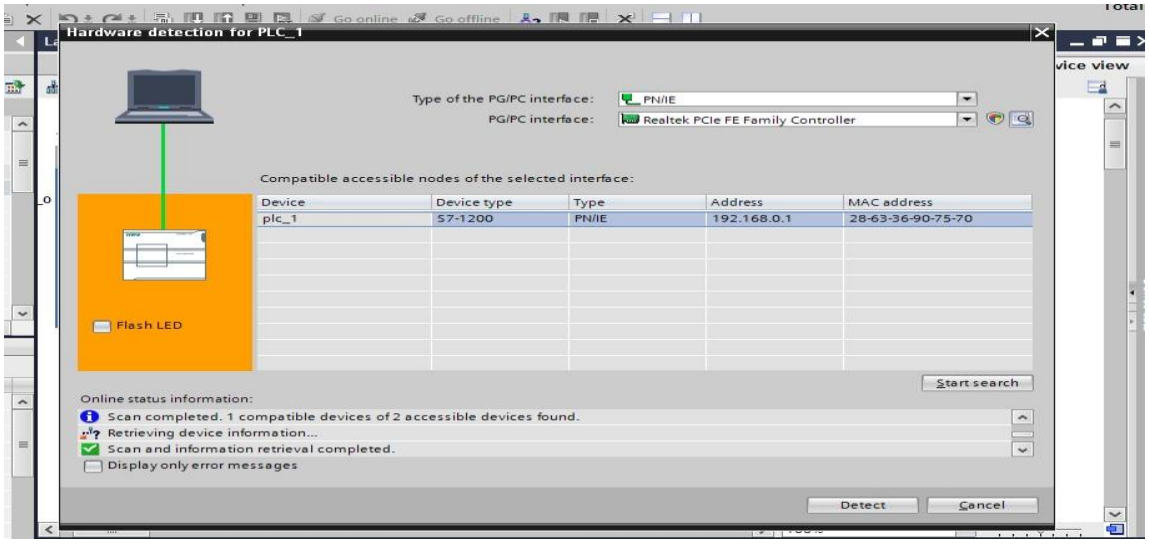


Another window should show up as shown below

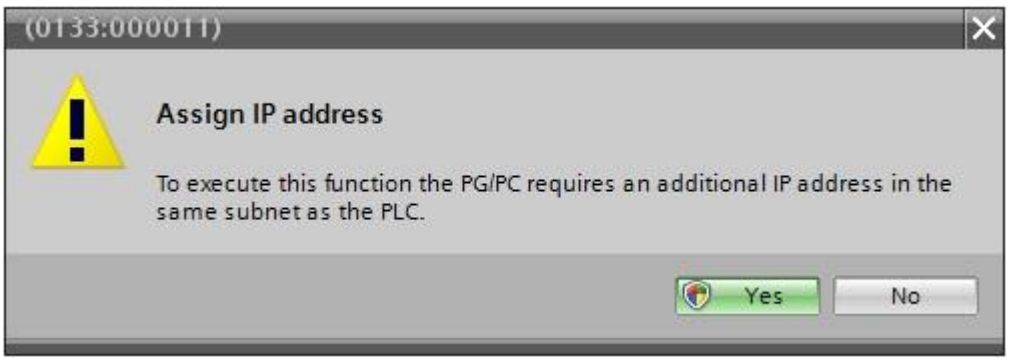


There are several drop down options but we won't bother with that for now, click **Start Search**.

Your computer will try to find compatible devices. After successfully searching, you will see this. If you can't get the compatible PLC to show up, ask your lab instructor for help.

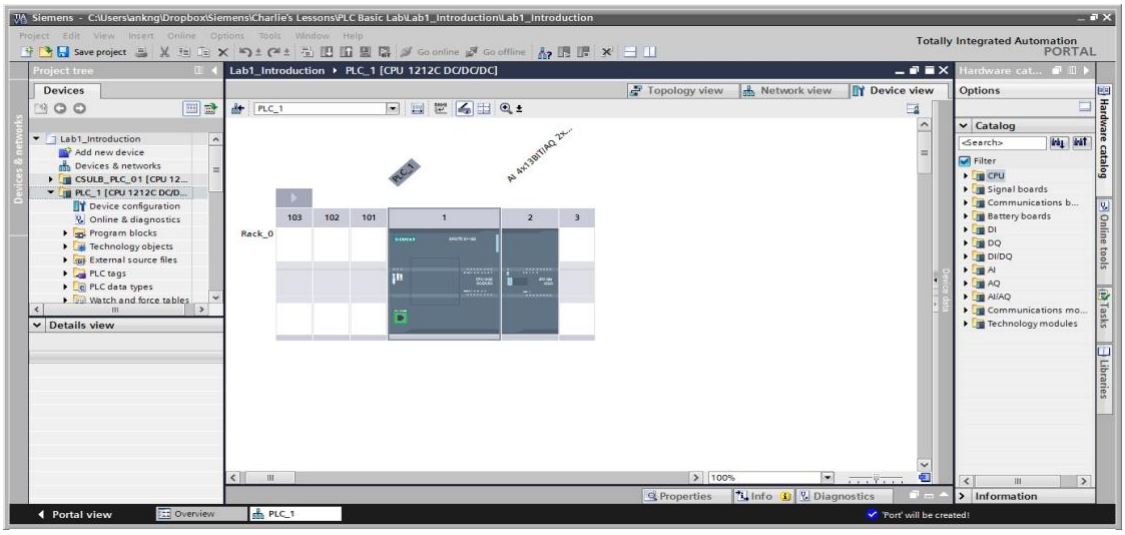


The green connection shows that you are connected with the PLC. To check, you can click **Flash LED** and it should blink on the PLC.
Highlight the plc_1 and click detect.



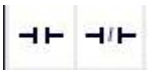
If this pops up, click yes

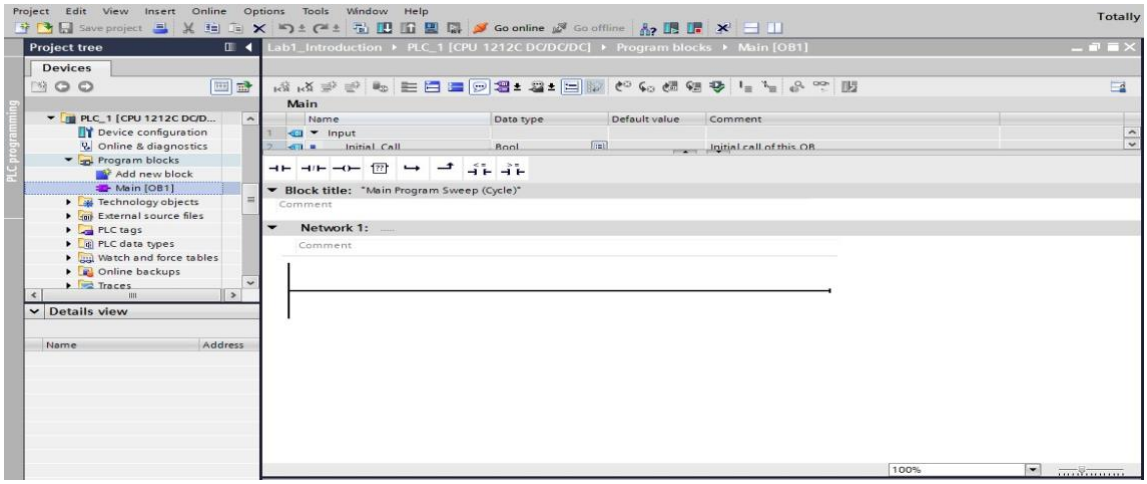
When the structure of the PLC is greyed out as below, you have successfully configured the device.



Programming blocks

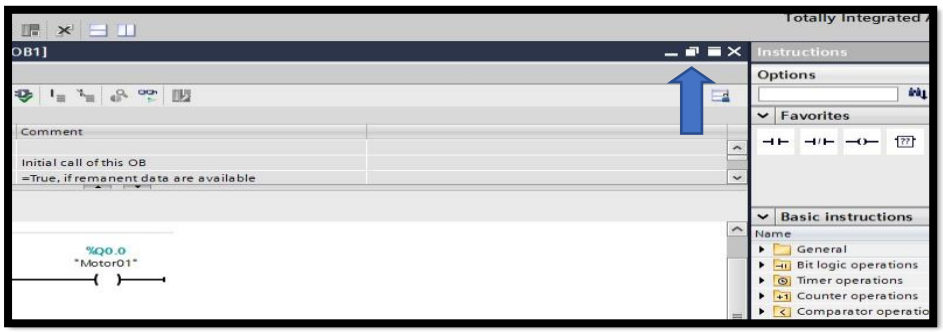
This portion of the lab shows you how to create your first program using network. A network is similar to an electrical wire, where you can place components such as contacts and assignments.
To access the networks, drop down the menus from your PLC in the Project Tree, until you see Program Blocks. Drop down from that and double click Main [OB1].
Another window should pop up in the middle.
As you can see, Network 1 is already made for you, also you can see some of the favorite blocks on the top of the work space.

These are contacts  and this is a coil  . The coil is necessary to complete the line and assign the output.



Network 1: The following steps will introduce you to the beginnings of writing a PLC.

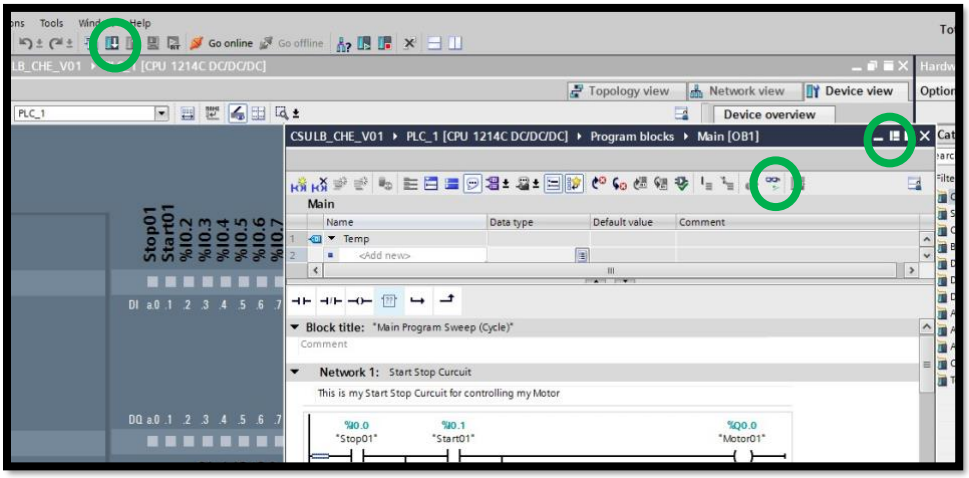
1.) Begin with grabbing the normally open contact (N/O) from the favorite's bar at the top. Drag it on to the network line and name it "Stop01".
2.) Now go to the Bit Logic menu on the right and grab another N/O contact, placing it into the network. Name this one "Start01".
3.) From either options, grab the coil to complete the circuit, naming it "Motor01". When complete, it should look like image below.
4.) When everything is set up, float the screen by selecting the float option on the upper right hand side. You should now be able to see the Device configuration window and also the Program block window over it.



5.) Zoom in onto the CPU on the Device configuration screen. From here, select the “Stop01” contact on the program block window and drag “%I0.0” on the CPU.

6.) Drag “Start01” contact to “%I0.1” and drag “Motor01” to “%Q0.0”.

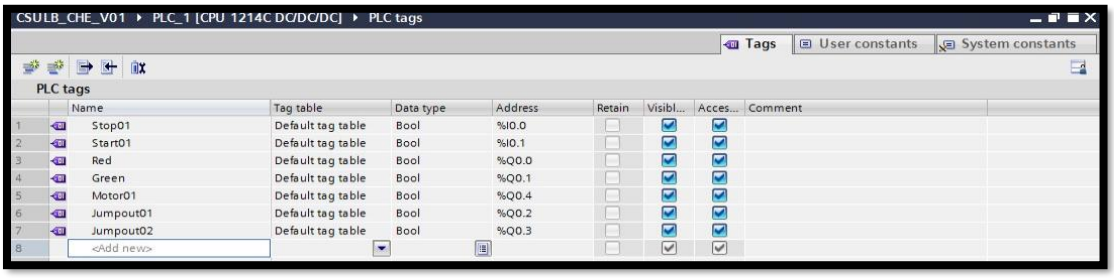
7.) Embed the Program block window and download the program to the device by clicking the download icon at the top. Also, turn on the monitoring option, which icon is a set of glasses.



8.) Notice that the red light turned on. This is not the action we were looking for. We wanted the white light to flash. So we need to fix this. To fix this issue, enter the tag library on the left column.

9.) Select Show all tags. From here, we will name all of the tags being used. Start by readdressing the Motor01 tag to %Q0.4 under the address column.

10.) Now we will add the rest of the tags as below:

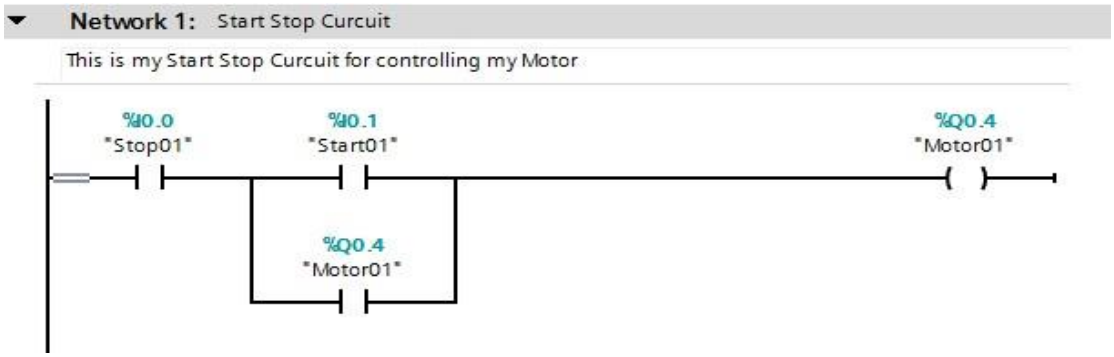


11.) Once done, download to device again. If done correctly, the white light should emit when pressing the green button.

Creating a Latch: The latch is a continuous loop action that activates when an initial action is done.

12.) Grab the open branch option from the favorites bar and drag it to the network inbetween the Stop01 contact and the Start01 contact.
13.) At the end of the open branch, place another N/O contact and addressing it to Motor01.
14.) Complete the branch by dragging the end of it back to the network, placing it after the Start01 and before the Motor01 coil.
15.) Download to the device again and open monitoring.
16.) If done correctly, pressing the green button would activate the white light and keep it on.

When complete, title the network “Start Stop circuit” and comment “This my start stop circuit for controlling my motor”



STEP 7 PID_Compact ohjelma, versio 6.