Joustava tapa integroida järjestelmiä node-red:llä visuaalisesti - Internet of Things & Industrial Internet

Mika Karaila
Metso Automation, Technology & Application Research
PL 237, 33101 TAMPERE
Tel. (040) 761 2563, telefax (02 483171)
{mika.karaila}@metso.com, http://www.metso.com/

AVAINSANAT Internet of Things, Industrial Internet, Node-RED, Visuaalinen ohjelmointi

TIIVISTELMÄ

IBM on kehittänyt uuden tavan ohjelmoida visuaalisesti. Ohjelmointi on helppoa ja käytettävät komponentit ovat uudelleenkäytettäviä. Ohjelmointi muistuttaa hiukan visuaalista toimilohkoilla tehtävää automaatiojärjestelmille tyypillistä ohjelmointia, mutta on enemmän IoT:n mukaista ohjelmointia. Funktiot perustuvat JavaScript:iin [1] ja templetointi on Mustache:n [2] avulla toteutettu. Tässä esitetään visuaalisen ohjelmoinnin soveltamista automaatio- / IoT-ympäristössä integroitaessa OPC UA [3] ja MODBUS TCP [4] protokollia sekä OPC UA:lle yksinkertaista HTML5 käyttöliittymää.

1 JOHDANTO

Node-red:in [5] avulla saa nopeasti toteutettua ja integroitua erilaisia protokollia. Yksi mielenkiintoisista standardi protokollista on OPC-UA, joka mahdollistaa nykyään jo monen eri järjestelmän välisen kommunikoinnin. Node-red:iin upotettuna JavaScript pohjainen node-opcua kytkee DCS-järjestelmän ja esimerkiksi kunnonvalvonnan yhteen. Kunnonvalvonta saa laitteiden tilatietoja sekä muita diagnostiikkaan liittyvää tietoa suhteellisen yksinkertaiselta näyttävän ohjelman kautta. Ohjelma voi hakea laitteet ja poimia niistä vain tietyt muuttujat ja tarkistaa niiden arvot. Kaikki suoritus on tapahtumapohjaista ja kommunikointi voi tehdä yksittäisen luku- tai kirjoitusoperaation. Yhteen ohjelmaan voi palastella omia välisivuja ja niillä voi olla omat valmiit kirjastoidut toiminnot. Näitä sopivasti yhdistelemällä voidaan tehdä tietojen hakuja eri protokollilla ja eri järjestelmistä.

Artikkelin sisältö on seuraava. Luvussa 2 käydään läpi node-red ohjelmointia OPC UA:n integroinnin näkökulmasta. Luku 3 kertoo muita käyttötapoja esimerkkien avulla. Lopuksi, luvussa 4 on yhteenveto demonstraatiosta.

2 VISUAALINEN OHJELMOINTI NODE-RED & OPC UA

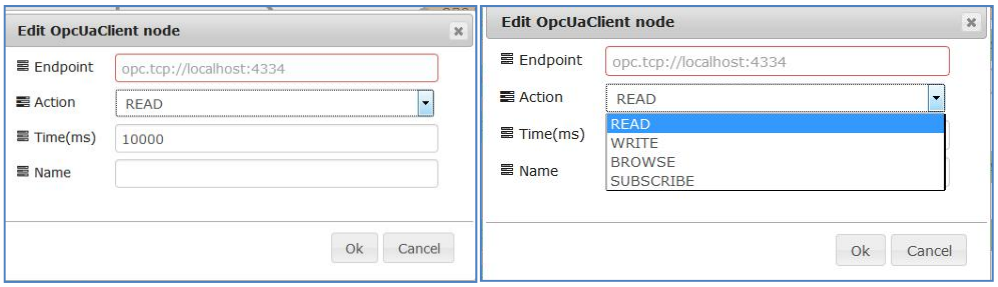
Node-RED on kehitetty “IBM Emerging Technology Services” ryhmän Nick O'Leary ja Dave Conway-Jones toimesta. Ohjelma perustuu solmuihin (node) ja niiden väliseen tapahtumapohjaisiin signaaleihin. Solmuja on suhteellisen yksinkertaista toteuttaa itse ja näin laajentaa ohjelmointia. Toinen oleellinen komponentti on node-opcua [6], joka on Etienne Rossignon toteuttama JavaScript-pohjainen OPC UA SDK -toteutus. Molemmat ohjelmat toimivat nodejs:n [7] päällä, jossa ytimenä on Google V8 JavaScript engine.

2.1 OPC UA laajennus

Näiden avulla on toteutettu yksinkertaiset solmut, joiden avulla on helppoa lukea ja kirjoittaa OPC UA muuttujia.

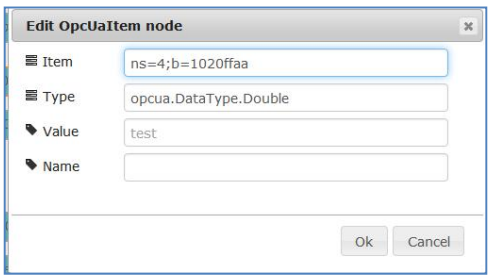
2.1.1 OPC UA client node

Alla olevassa kuvassa on solmun parametrit.



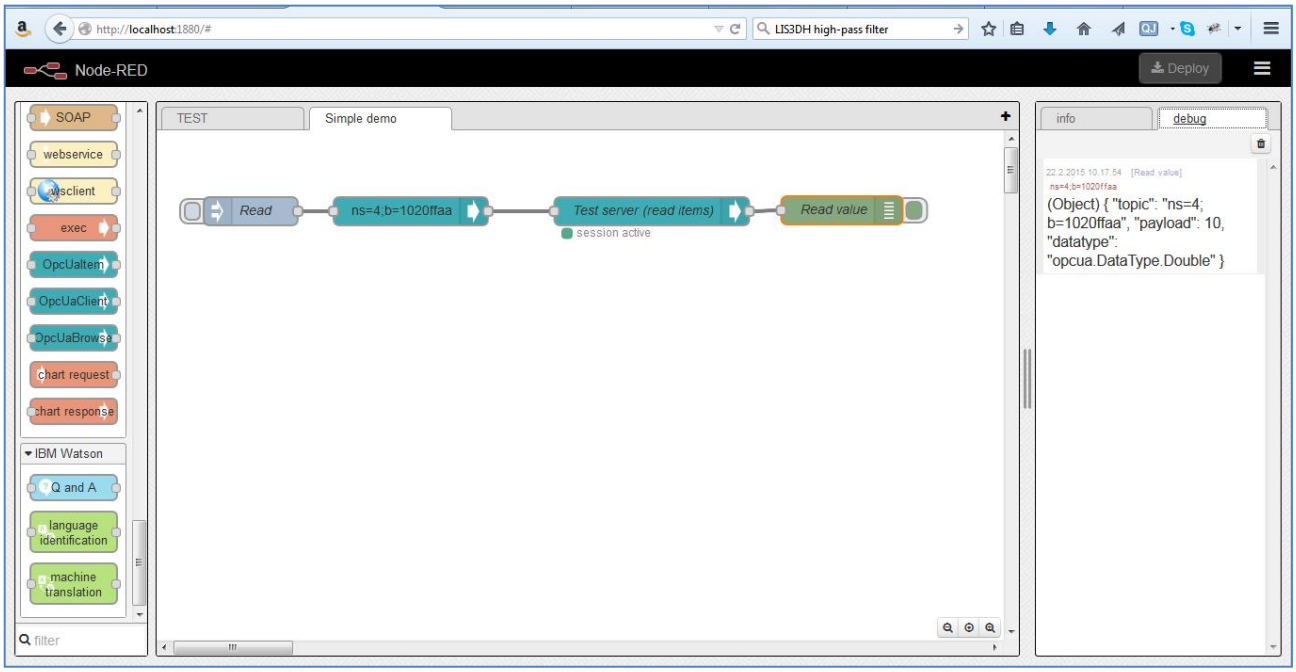
Alasvetovalikosta voi valita halutun toiminnon. Tämä solmu siis muodostaa varsinaisen yhteyden OPC UA serveriin.

2.1.2 OPC UA item node



Tässä on määriteltynä varsinaisen OPC UA muuttujan nimi ja tietotyyppi.

Alla on yksinkertainen vuo (flow), jonka perusteella luetaan yksi muuttuja. Oikealla näkyy luettu arvo. OPC UA client node näyttää aktiivisen session vihreällä pallolla (solmun tila-tieto). Ohjelman todentamisessa on hyvä käyttää toista OPC UA client ohjelmaa ja tässä tapauksessa hyvä ohjelma on UaExpert [8], jonka avulla voi tarkastella osoiteavaruutta sekä eri muuttujien attribuutteja sekä arvoja.



2.1.3 OPC UA browse node

Käyttää erillistä Address.txt tiedostoa, josta saa valmiit jo luetut osoitteet. Tosin helpompi keino on tehdä suoraan valmiita OPC UA item määrittelyjä, joita voi käyttää sivuvalikosta. Tämä idea tuli, kun kysyttiin kuinka saisi Drag&Drop toiminnallisuuden node-red käyttöliittymään. Oikealla oleva sivupaneeli toimii ”Debug”-output:ina ja siinä olevaa tekstiä voi valita. Jos kyseessä oleva teksti on JSON-muotoinen ja sisältää valmiiksi määritellyn oikean rakenteen voi sen avulla muodostaa uuden solmun. Tätä varten käytetään template solmua, jonka avulla voi korvata avainsanoja tekstin joukosta. Tässä tapauksessa teksti on yhden ”OPC UA item” objektin määrittely. Alla olevassa kuvassa on tehty ylimmän vuon ajo ja sen tuloksena on tulostettu kaikki OPC UA muuttujat oikeassa reunassa olevaan debug paneeliin. Siitä on alin valittu ja raahattu keskelle, jolloin saadaan uusi muuttuja lisättyä ohjelmaan.

OPC-UA integroi yhteen OPC-UA serverin ja flow itsessään sisältää OPC-UA client:in, joka voi tehdä OPC-UA osoiteavaruuden selaus (browse) toiminnon. Muuttujia voi lukea (read), kirjoittaa (write) sekä tilata niiltä halutuksi ajaksi päivityksiä (subscribe). Kaikissa protokollissa ei ole osoiteavaruuden browse / scan mahdollisuutta vaan käytetään valmista kiinteää osoiteavaruutta.

Tämä esimerkki ja siinä toteutetut solmut ovat saatavilla huhtikuussa [9].

3 MUUT LAAJENNUKSET: MODBUS JA HTML5 KÄYTTÖLIITTYMÄ

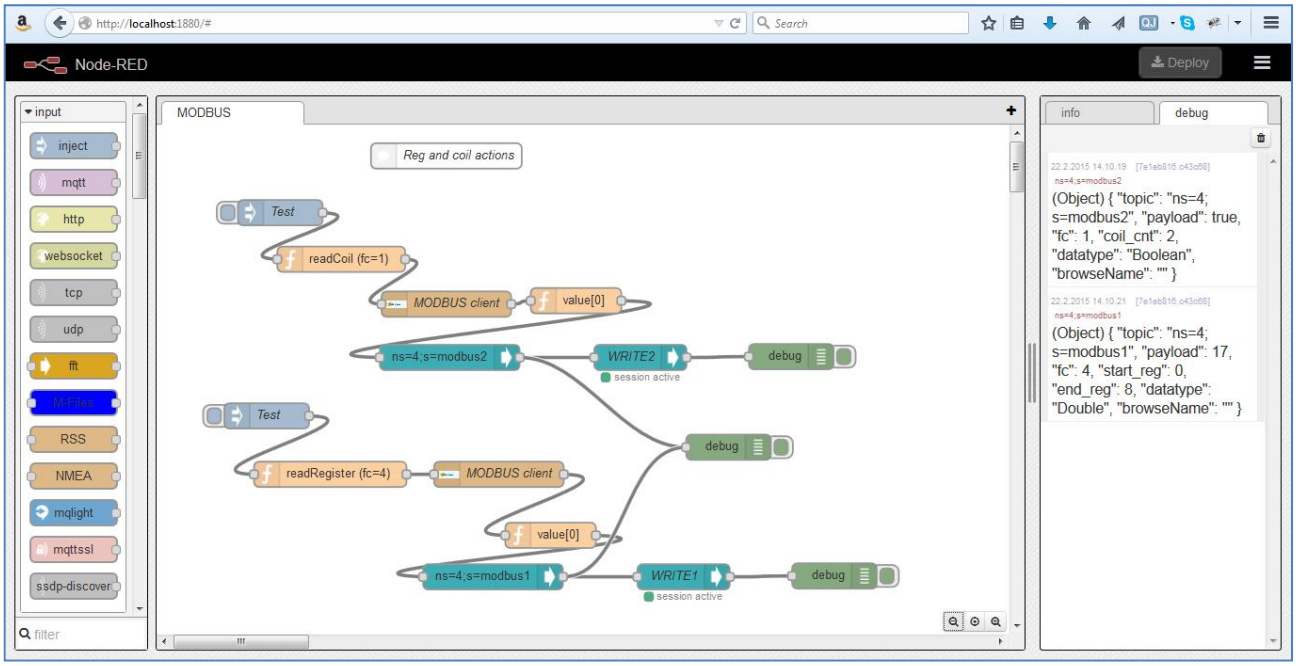
Toinen esimerkki on MODBUS protokollapino, jonka saa valmiina kirjastona ja niiden varsinainen soveltaminen ei vaadi yleensä syvällistä protokollan tuntemista. Käyttäjä voi lisätä näitä valmiita solmuja ja vain parametroida niitä: protokollan mukaisten osoitteiden syöttäminen riittää. Jos kyseessä on useiden satojen signaalien käsittely niin konfiguraatio voi olla talletettuna csv tiedostoon, jota luetaan ja käytetään ajoaikaisesti.

Kolmas esimerkki on kuinka node-red:in avulla voidaan muodostaa HTML5 käyttöliittymä.

3.1 MODBUS laajennus

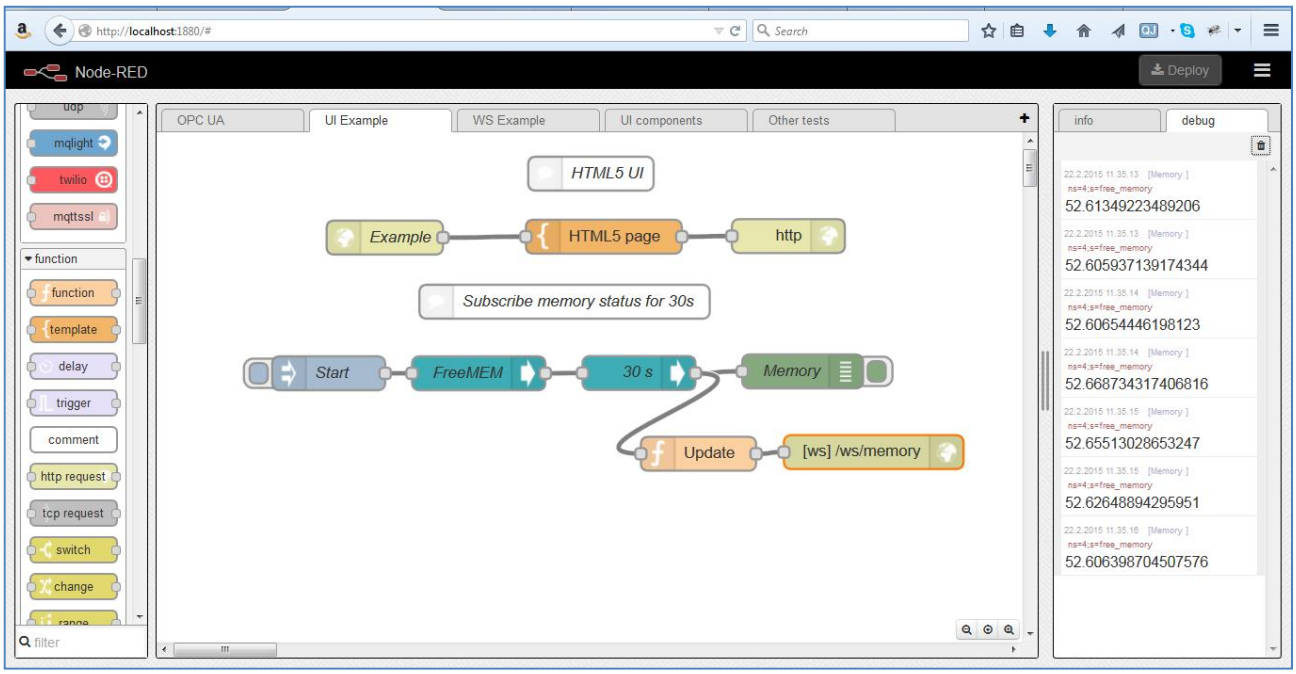
MODBUS-solmu osaa lukea testipalvelimelta coil ja register muuttujia. Nämä arvot kirjoitetaan edelleen OPC UA -palvelimelle. MODBUS-osoite määritetään erikseen function solmussa antamalla fc = Function Code ja osoite-avaruus alkuosoite sekä pituus. Kun MODBUS client -solmu on lukenut

arvon niin se menee edelleen OPC UA item -solmuun ja siinä määritellään OPC UA -osoite, johon arvo kirjoitetaan.

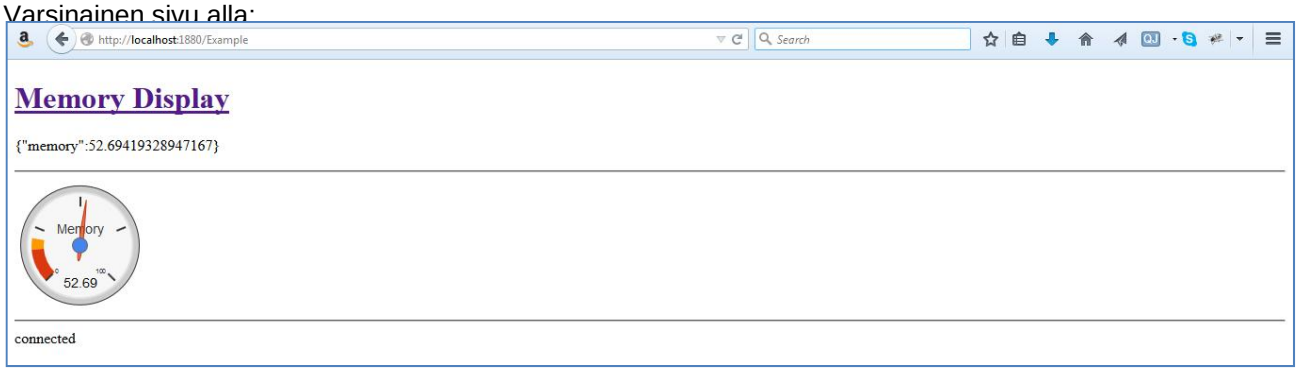


3.2 HTML5 käyttöliittymä

Tämä esimerkki näyttää kuinka WebSocket:in [10] kautta voidaan rakentaa yksinkertainen HTML5 käyttöliittymä. OPC UA:n kautta luetaan palvelimen vapaan muistin määrä ja nod-red vuo:ssa saatu tieto välitetään WebSocket:in kautta varsinaiseen käyttöliittymään HTML5 sivulle.



Varsinainen sivu alla:



4 YHTEENVETO

Node-RED:lle löytyy jo useita valmiita toteutuksia, joissa on toteutettuna jo mm. Coap, MQTT, MQTT-SSL sekä monia muita protokollia. Kuten aikaisemmin esitettiin niin mm. MODBUS protokollan laajennus oli tehtävissä. Näin siis uusia ja vanhoja protokollia saadaan kehittäjäyhteisön kautta koko ajan lisälaajennuksina.

Nämä esimerkit osoittavat että node-red:llä voidaan tehdä ensin uudelleenkäytettäviä solmuja, joita voi edelleen soveltaa ja kytkeä eri tavoin. Näin saadaan aikaan helposti uusia sovelluksia ja integraatioita erilaisten järjestelmien välille. Solmun toiminnallisuus kannattaa pitää yksinkertaisena, jolloin sen uudelleenkäyttö on helppoa. Nämä ovat node-red:iin tehtyjä laajennuksia sekä esimerkki vuo-kuvauksia, jotka tulevat huhtikuussa jakeluun GitHubin kautta [9].

VIITTEET

[1] JavaScript, http://www.w3schools.com/js/

[2] Mustache, https://github.com/mustache/mustache

[3] OPC UA, https://opcfoundation.org/about/opc-technologies/opc-ua/

[4] MODBUS TCP, http://www.modbus.org/

[5] node-red, http://nodered.org/

[6] node-opcua, https://www.npmjs.com/package/node-opcua

[7] nodejs, http://nodejs.org/

[8] UaExpert, http://www.unified-automation.com/downloads/opc-ua-clients.html

[9] node-red-contrib-opcua, https://github.com/mikakaraila/node-red-contrib-opcua (saatavilla huhtikuussa 2015)

[10] WebSocket, https://www.websocket.org/