Welcome to the presentation

“Fieldware: So wird das IoT programmiert!”

Übersicht

1) Was ist ‘Fieldware’ ?

2) Welche Anwendungen dafür gibt es ?

3) Wie programmiert man so etwas ?

1. Was ist “Fieldware”: So wird das IoT programmiert!

Mit IoT wird es 10 mal so viele Geräte geben wie heute, aber nicht 10 mal so viele Programmierer.

Diese IoT Geräte werden von Handwerkern und Facharbeitern installiert, nicht von Informatikern.

Die IoT Geräte müssen mit anderen Systemen vernetzt werden und interagieren.
Nur “Plug And Play” wird nicht reichen.

→ Die SW muss einfacher und automatischer werden
→ Die SW muss die Sprache der Handwerker sprechen
→ Die SW muss flexibel genug sein, um Systeme verschiedener Hersteller sowie diverse Gewerke verbinden zu können.

1. Die Dritte Programmierebene: FIELDWARE

Neben der FIRMWARE braucht es die FIELDWARE

Browserbasierte Programmierung von Funktionsplan (FUPLA) und Web-HMI's.
Für die Umsetzung von IoT Devices wichtig, weil bisher die 3. Ebene fehlt:

- Die 1. Ebene umfasst die Grundfunktionen des Gerätes:

OS, Netzwerk, Display, Treiber und Middleware. Dieser Teil der Firmware ist fast immer in C/C++ implementiert.

- Die 2. Ebene ist die Applikationsebene.

Hier werden Regel- und Steuerfunktionen der industriellen Applikation programmiert. Die Umsetzung erfolgt entweder auch in C/C++ oder häufig in IEC61131.

- Die 3. Ebene ist die Feldebene.

I4.0 und IoT verlangen vermehrt nach einer Möglichkeit zur einfachen Programmierung bei der Inbetriebnahme.

2. Welche Anwendungen dafür gibt es ?

1) Einfachst-SPS

2)Appliances: Programmierbarkeit als Erweiterung

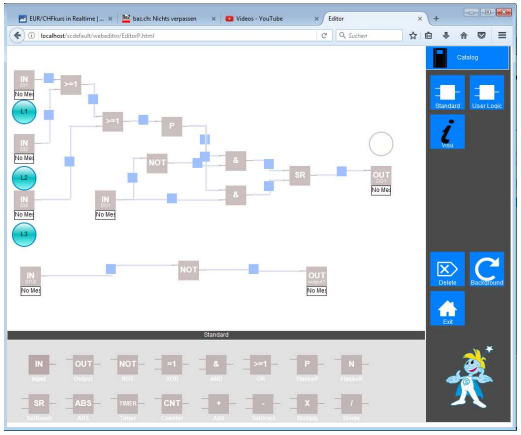
3)Edge- und Fog-Computing: Vom Daten sammeln (SCADA) zum Funktionen sammeln

2.1 Anwendungen: Kleinsteuerung mit LCD

SmartPLC für einfache Anwendungen Programmierung nur per Browser HMI

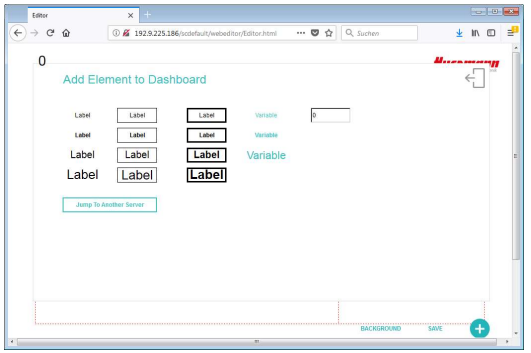
Funktionsplan

-> Massive Reduktion der Komplexität

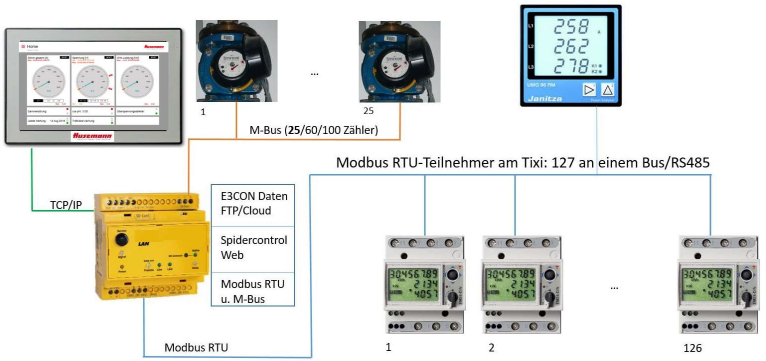


2.2 Anwendungen Appliances: Energy Management

Die “Firmware”:
Energiemanagement System mit Browser-Bedienung, findet Energiezähler (Modbus) automatisch und kann diese Visualisieren:
Anzeige der Werte, Trends, Alarme (werden konfiguriert)



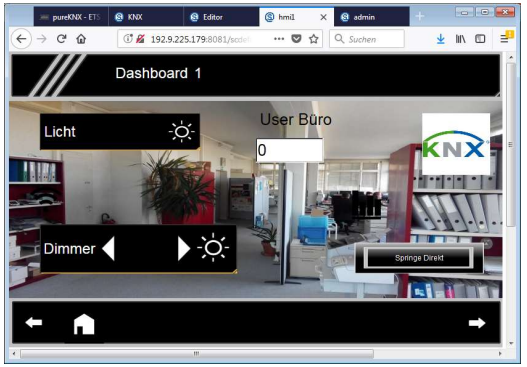
Die “Fieldware”: Programmiert mit SpiderPLC
Kunden benötigen Dashboards mit wichtigen Werten (HMI), Werte müssen teilweise errechnet werden (FUPLA), programmierbare I/Os der Messgeräte sollen Alarmkontakte erfassen, verknüpfen und an einen Sammelalarm weiterleiten
Auch Modbus I/Os können angeschlossen werden
Weitermeldung an Cloud, SCADA und SPS über Kommunikations-FUP



2.2 Anwendungen Appliances: ETS KNX Touch Panel

Die “Firmware”:

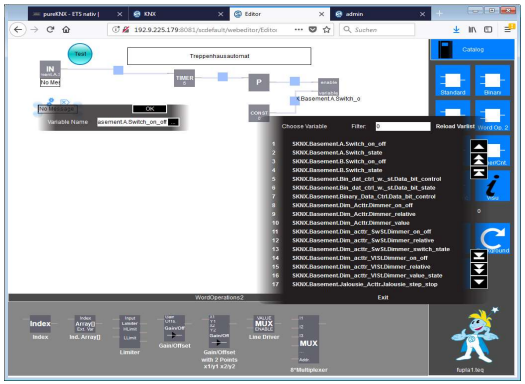
KNX Touchpanel (Building Automation) wird per ETS programmiert, HMI wird automatisch daraus erzeugt.



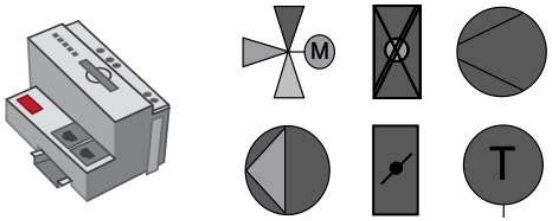
Die “Fieldware”: Programmiert mit SpiderPLC

HMI kann angepasst werden, Hinterlegen von Bildern, kopieren von automatisch erzeugten Controls

Zugeordnete Tasten können per FUPLA gesteuert werden: Timer (jeden Tag ab 18.00 schliessen, Treppenhausautomat,...), Sollwerte (Nachtabsenkung), Anbidung an SPS und SCADA.

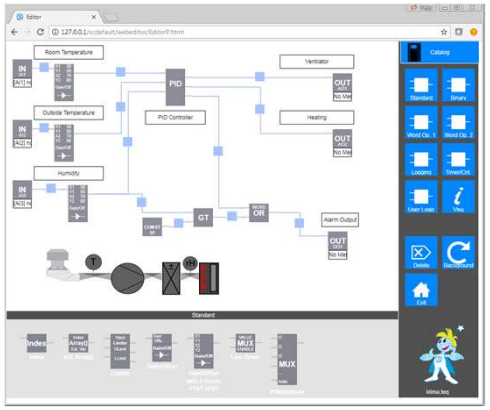


2.2 Anwendungen Appliances:
Industrial Controller Building Automation



Die “Firmware”:
Industrieregler (Building Automation) auf SPS Basis.
Reglerfunktionen mit CODESYS programmiert.

Die “Fieldware”: Programmiert mit SpiderPLC
Monteure programmieren Anlage mit FUPLA
Komplexe Algorithmen der Firmware (z.B. PID Regler) werden als FB's gekapselt, Umfang der FB's ist nach Bedarf angepasst.
Für jedes HW-Element des Herstellers (Ventil, Pumpe, etc.) existiert ein FB und ein HMI Objekt.
Der Monteur programmiert die Logik und das HMI auf der Komplexitätsebene der Mechanik und Elektrik, die er soeben zusammengebaut hat.



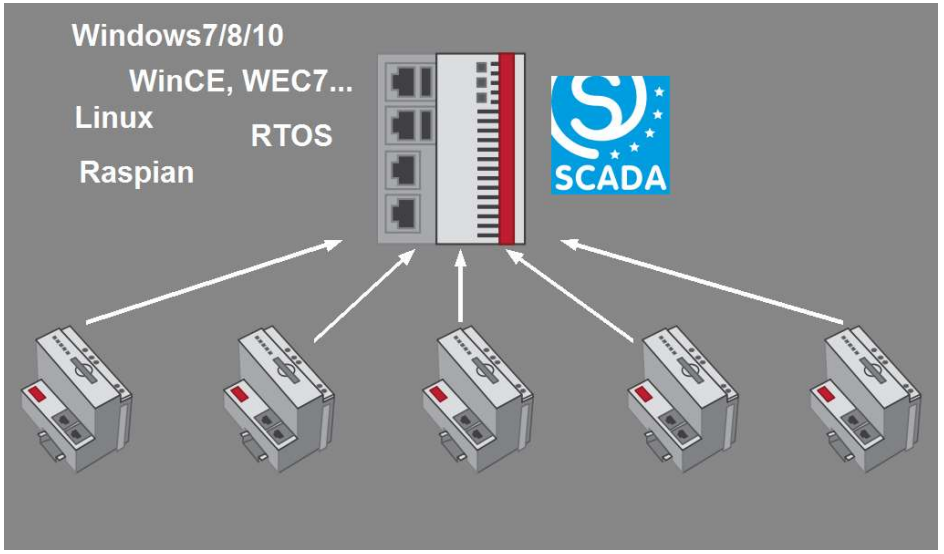
2.3 Anwendungen: Fog- und Edge-Computing

SCADA: Vom Daten sammeln zum Funktionen sammeln

-Embedded SCADA hat Schnittstellen zu verschiedenen SPS und Automationskomponenten

-SCADA loggt Daten

-**Es braucht die Verbindung zur Welt der IT**



2.3 Anwendungen:
Fog- und Edge-Computing

Edge Computing durch Funktionsplan:

-Einfache Logik wird in SpiderRT ausgeführt

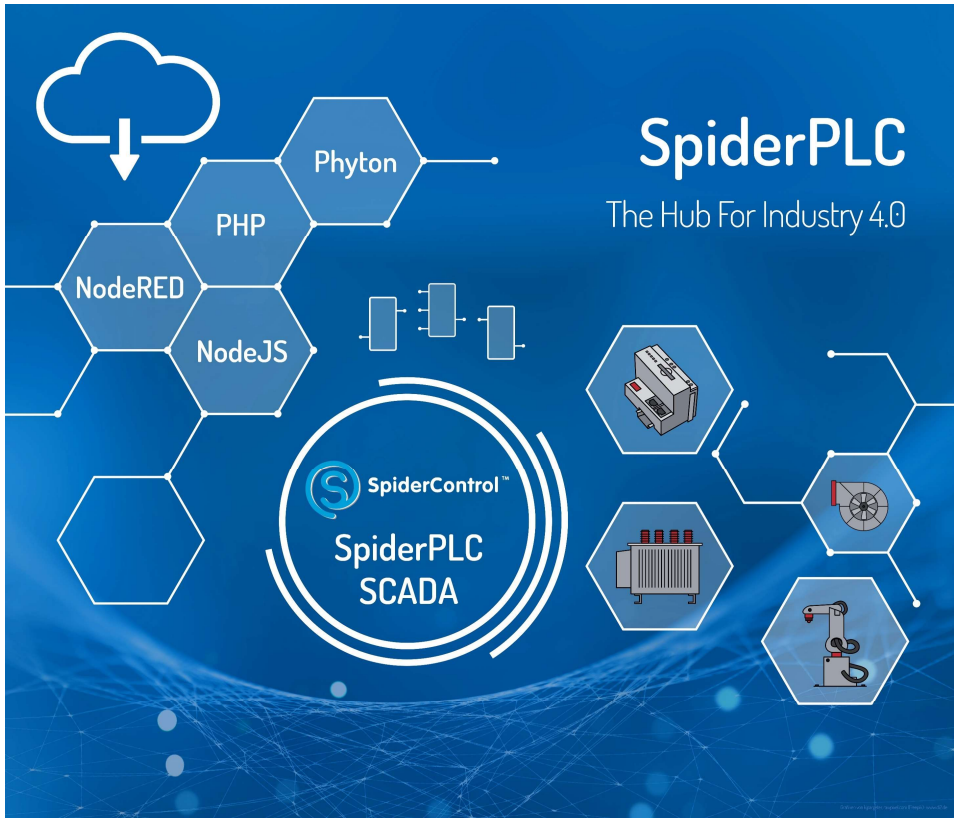
-Kundenspezifische Funktionsbausteine (FB's) rufen externe Script Funktionen

-JS liest einen Wetterdienst

-Python berechnet eine FFT

-NODE-RED bringt den MQTT Anschluss

-PHP speichert in eine SQL DB und macht Statistik



2.3 Anwendungen:
Fog- und Edge-Computing

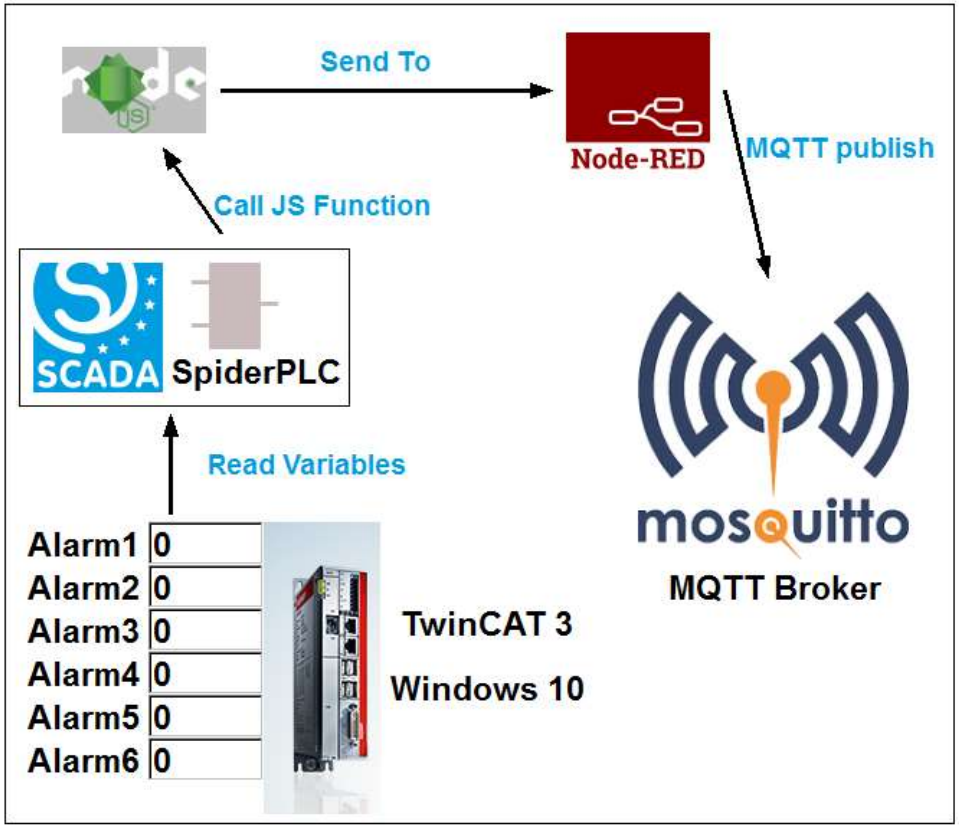
SCADA: Vom Daten sammeln zum Funktionen sammeln

-Kombiniert Daten und Funktionen

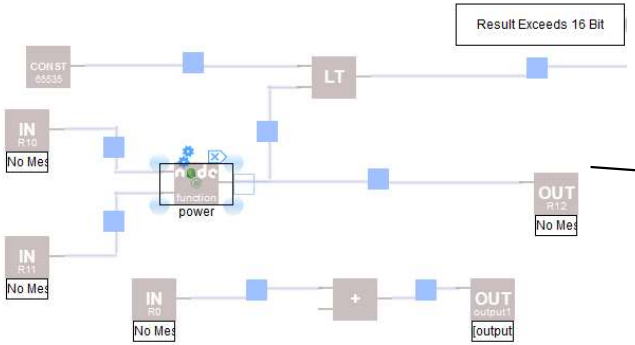
-Treiber können Daten lesen

-Treiber können aber auch Funktionen aus Dritt-systemen aufrufen (dies ist z.B. ein wichtiger Teil der OPC UA Spezifikation)

-SpiderPLC ist das benutzerkonfigurierbare grafische Interface, um Funktionen() aus verschiedenen Sprachen zusammenzubringen

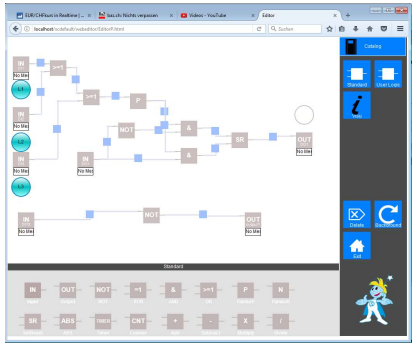


2.3 Anwendungen: Fog- und Edge Computing aus der Sicht des Programmierers:



Lokale Variablen der SPS Treiber werden übergeben

```
function myFunctionDispatcher(request, response) {

        console.log("myFunctionDispatcher");
        var query = querystring.parse(url.parse(request.url).query || "");

        if(query.Param_0 == "power"){
          power(request, response);
        }
        else if(query.Param_0 == "mult"){
          mult(request, response);
        }
        else if(query.Param_0 == "exit"){
          process.exit(0);
        }

}


function power(request, response) {
        var query = querystring.parse(url.parse(request.url).query || "");

        var uniqueID = parseInt(query.Param_1); // Param_1 contains the UID from the call
        var x = parseInt(query.Param_2);
        var y = parseInt(query.Param_3);
        console.log("power");
        var i;
        a = 1;
        for(i = 0; i < y; i++){
          a = x * a;
        }

        response.writeHead(200, {'Content-Type': 'text/html'});
        response.write("<body>");
```



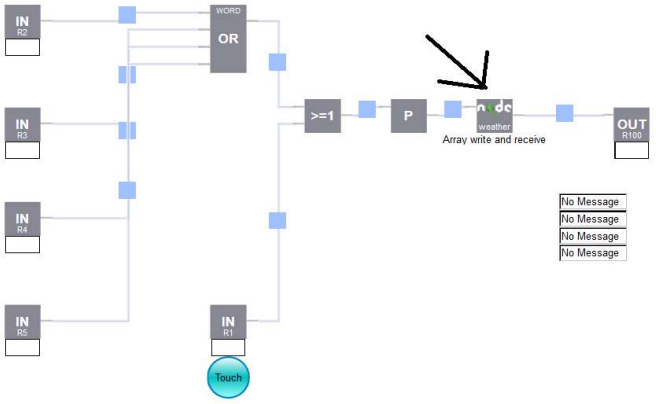
Der Benutzer platziert einen FB, welcher eine JS Funktion aufruft

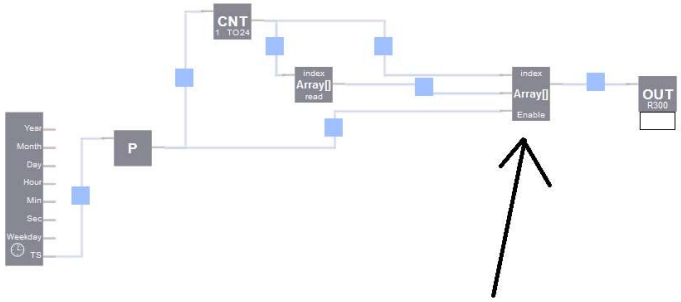
Variablen aus node-js können in SpiderPLC zurückgeschrieben werden

2.3 Anwendungen: Lese Wetter Vorhersage von openweathermap.org und speichere diese in SPS

Set a timer to get the forecast at 12.00 and at 6.00 pm

Call the JS function to call the openweathermap





Copy the result array to variables from a remote PLC

3. Wie programmiert man so etwas ?

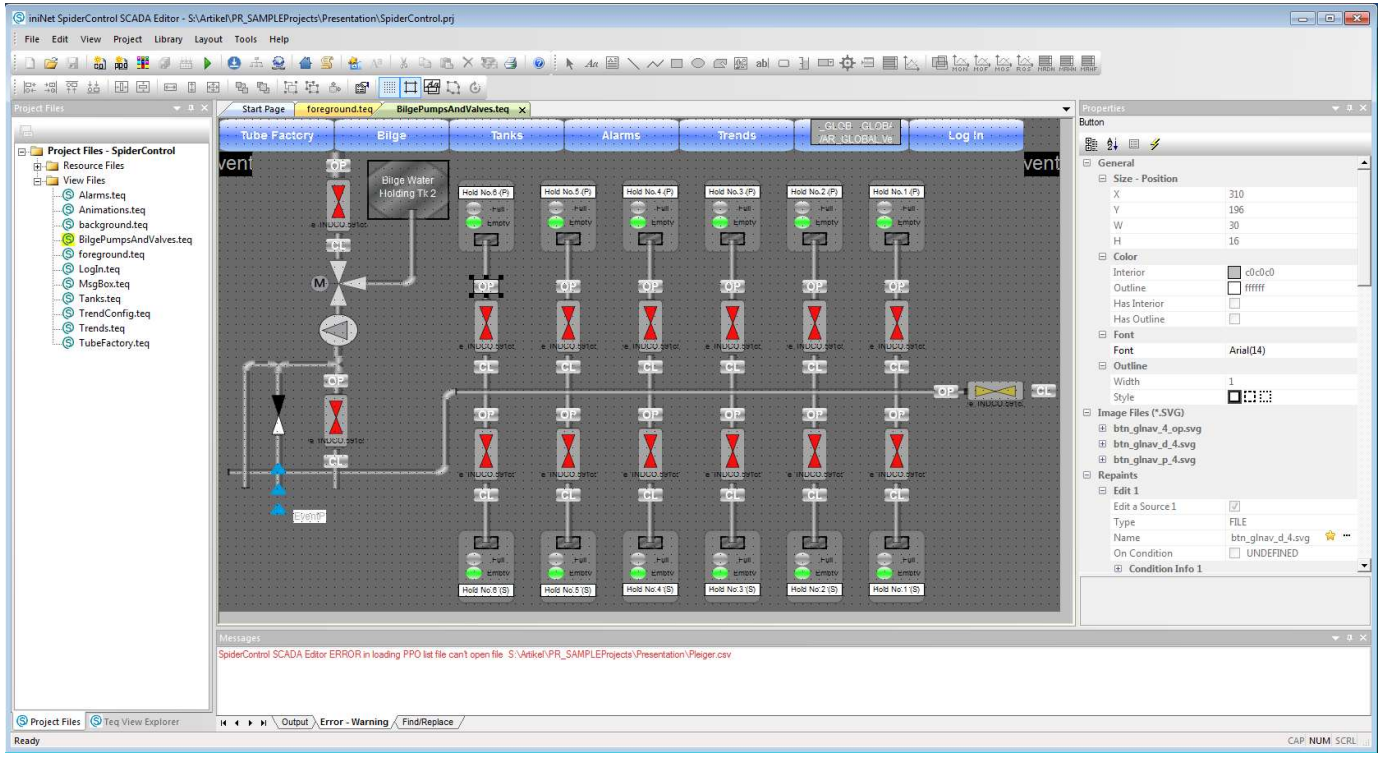
Die Requisiten: SpiderControl Tool Chain

1) HMI Editor für HTML5 SCADA Projekte

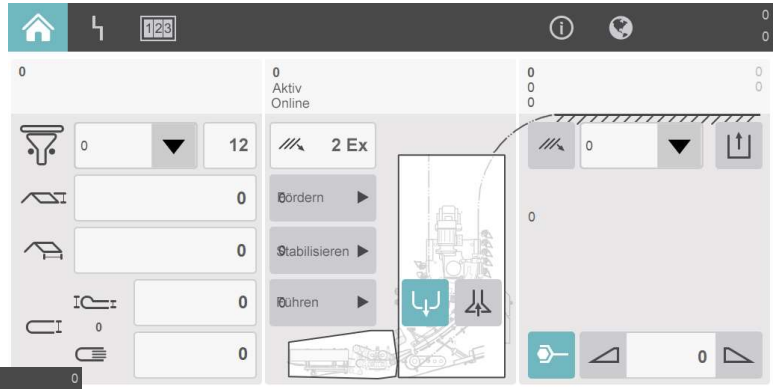
2) SCADA Server portiert auf alle wichtigen OS: WindowsXY, Linux, Raspian, Android

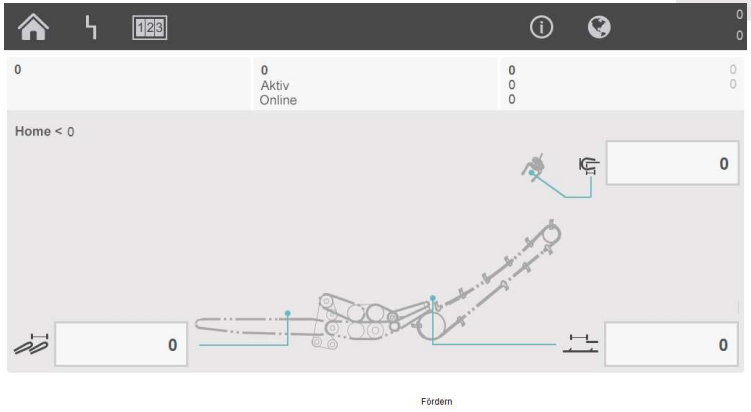
3) SpiderPLC: Programmieren mit dem Browser

3.1 HMI Editor für HTML5 SCADA Projekte

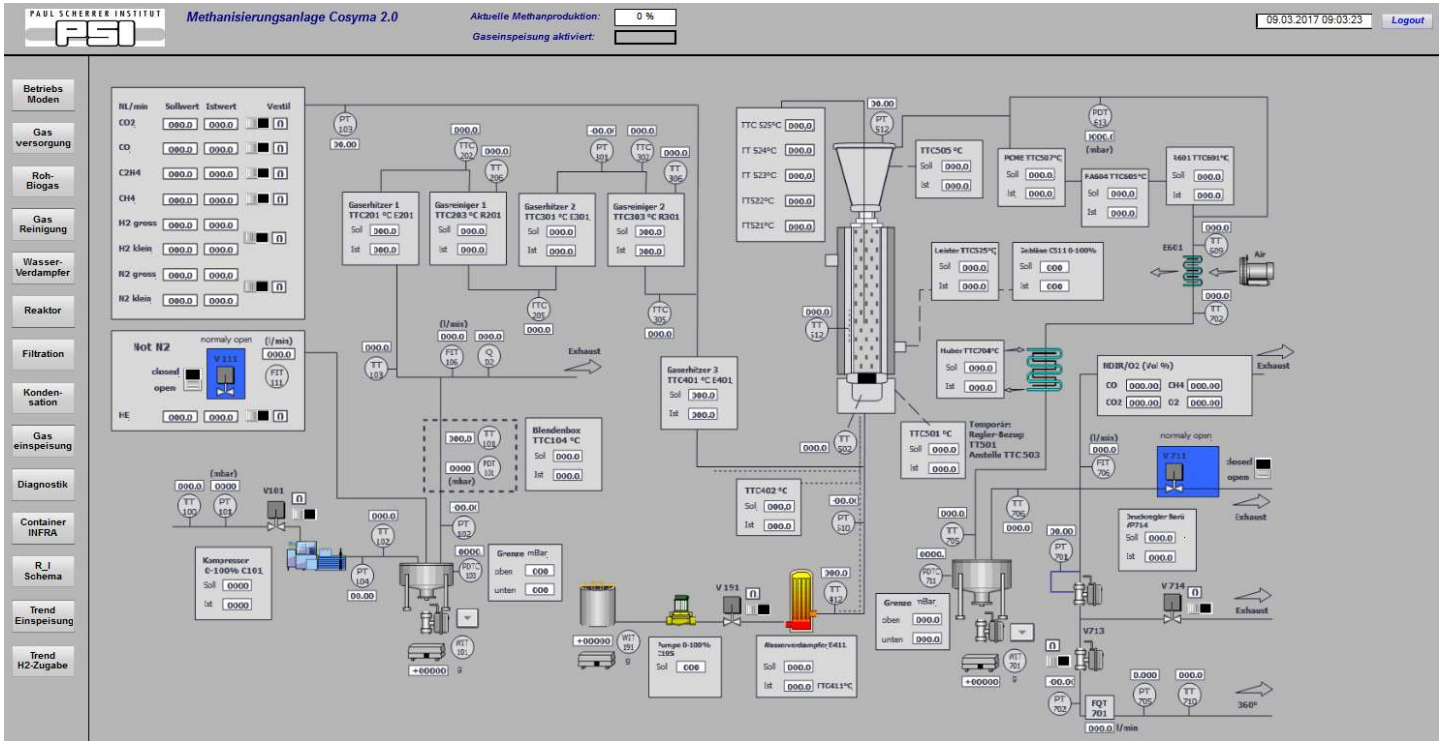


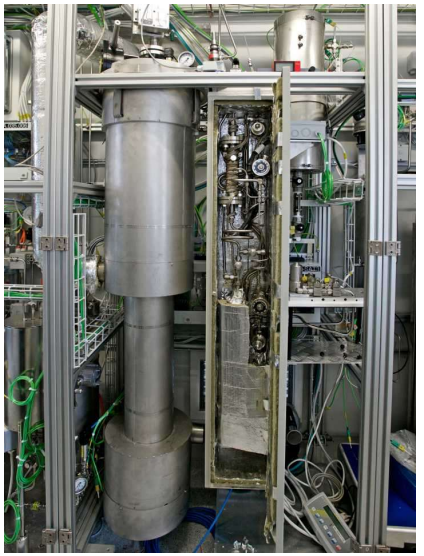
3.1 Industrial HTML5 HMI Beispiel: Maschinen



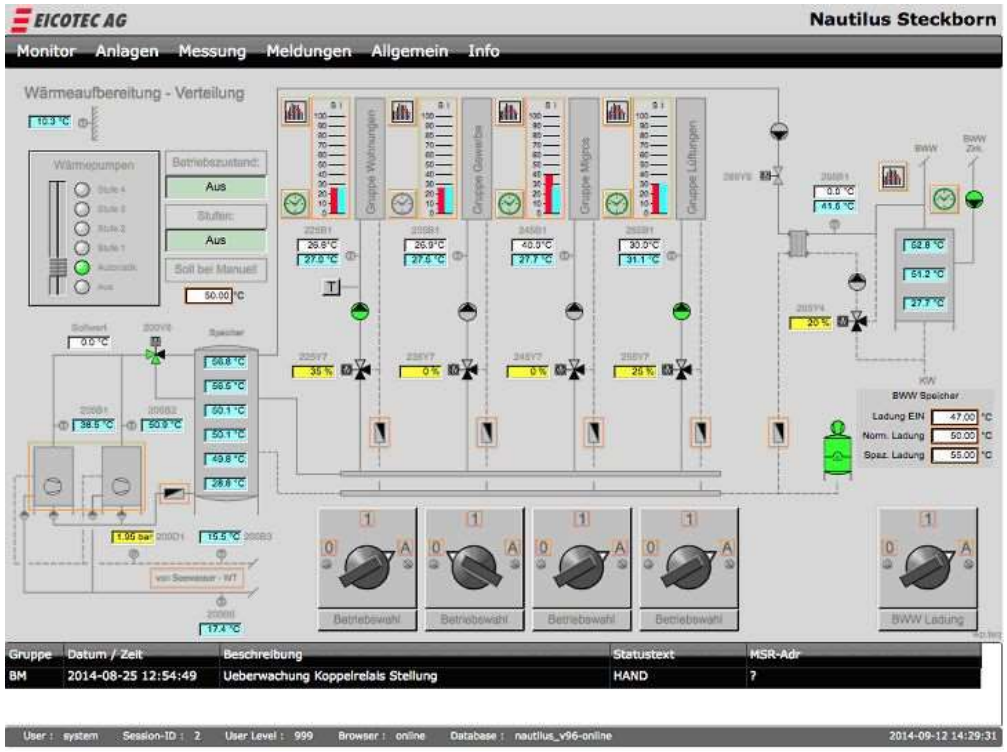


3.1 Industrial HTML5 HMI Beispiel: Prozess

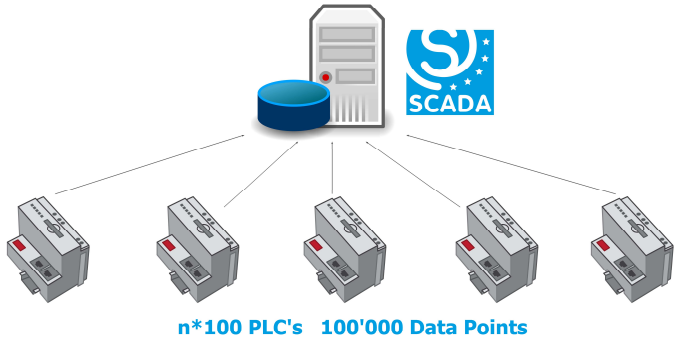


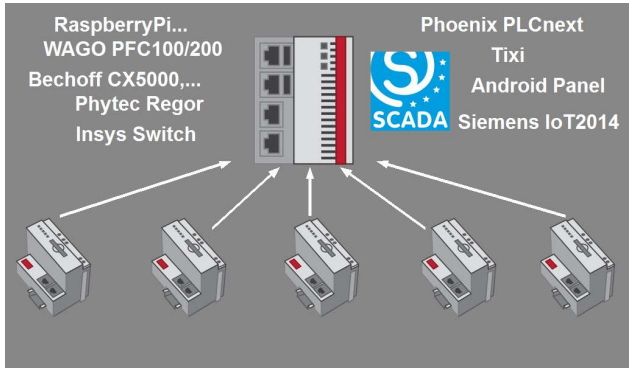


3.1 Industrial HTML5 HMI Beispiel: Building

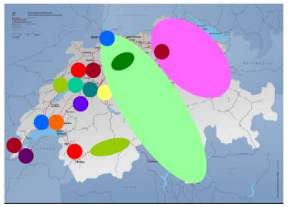


3.2 SCADA Server auf allen Plattformen





SCADA Server auf: WindowsXY, Linux, Raspian, Android…

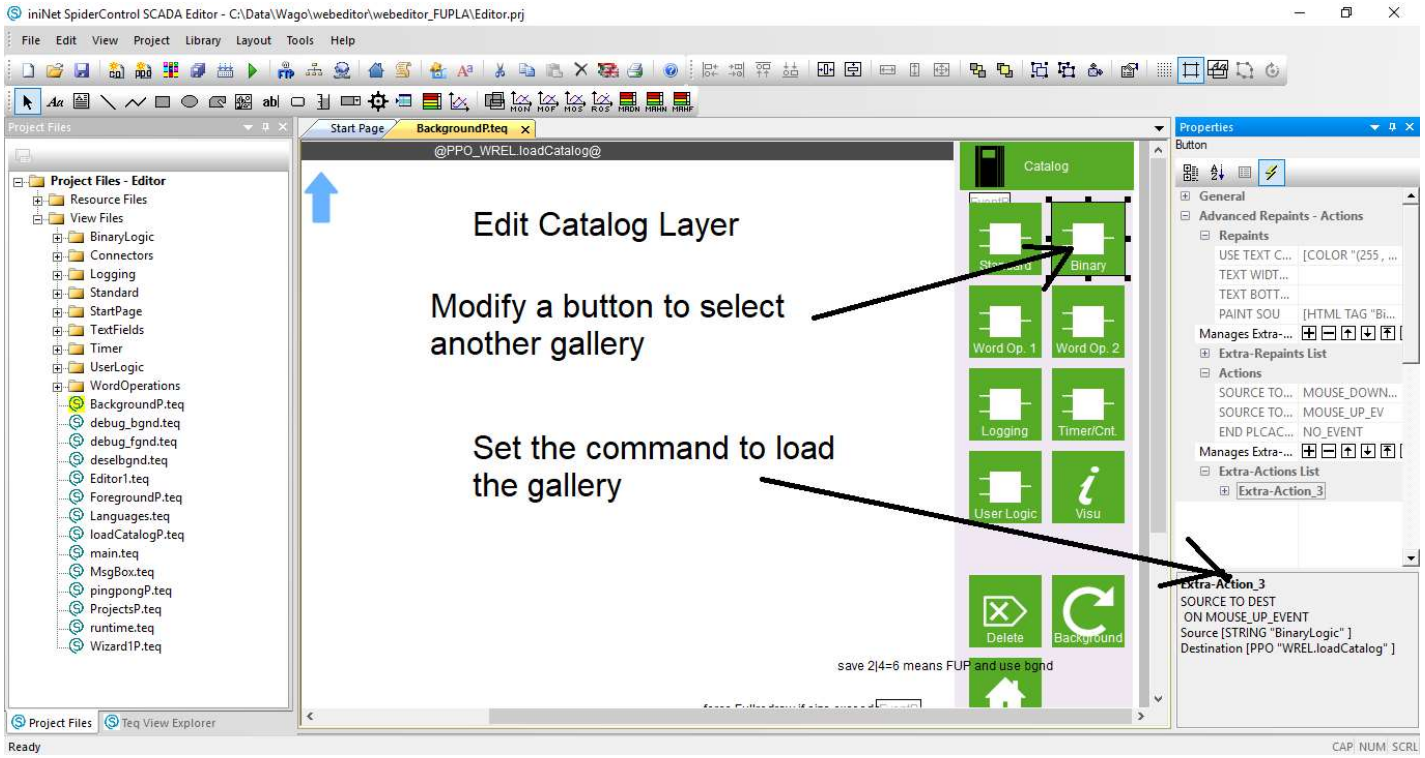


SpiderControl™ is a brand of iniNet

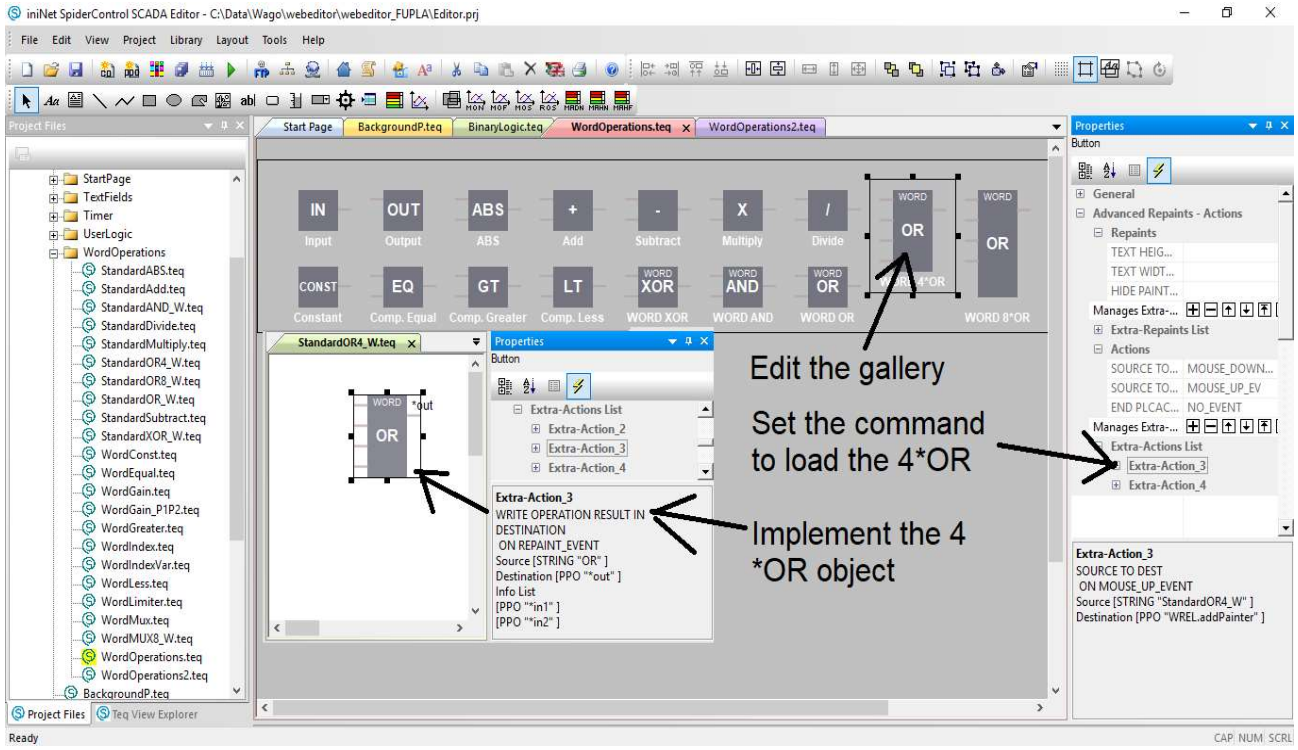
3.3 Ihr eigenes Programmiertool ist ein SCADA HMI

Der “Editor-Editor”:

PC HMI Editor zeichnet SpiderPLC UI



…und definiert Funktionen und Objekte!



SpiderPLC zwei mal nominiert!

embeddedworld2018
Exhibition&Conference
... it's a smarter world

20 Jahre E Elektronik
Produkte des Jahres
2018 NOMINIERT

Vielen Dank für Ihre Aufmerksamkeit.

iniNet Solutions GmbH
Fichtenhagstr. 2
CH – 4132 Muttenz

Tel: +41 61 716 96 26
Fax: +41 61 716 96 17
E-Mail: info@ininet.ch
Web: www.ininet.ch