“Fieldware: So wird das IoT programmiert!”

iniNet Solutions GmbH
Peter Brügger
Geschäftsleitung

SpiderControl™

Virtual Private Infrastructure

“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.

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.

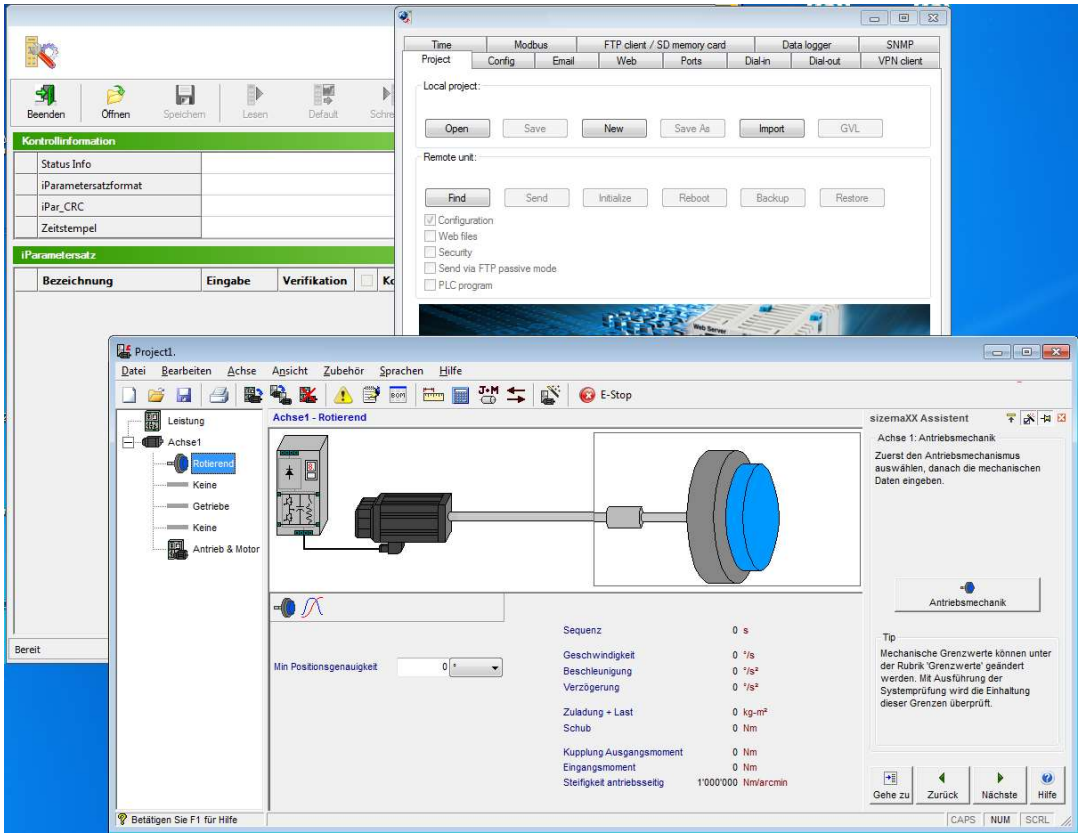
- Die 3. Ebene ist die Feldebene. I4.0 und IoT verlangen vermehrt nach einer Möglichkeit zur einfachen Programmierung bei der Inbetriebnahme.

Anwendungen:
Wer braucht das?

Markt: Geräte und Systeme der Automation

Bisher:
PC Software
Konfigurationsseiten
Per Browser

Neue Anforderungen aus IoT/I4.0 & Co.:
Mobile Devices
Cloud-Anbindung
Virtualisierung
Vereinfachung



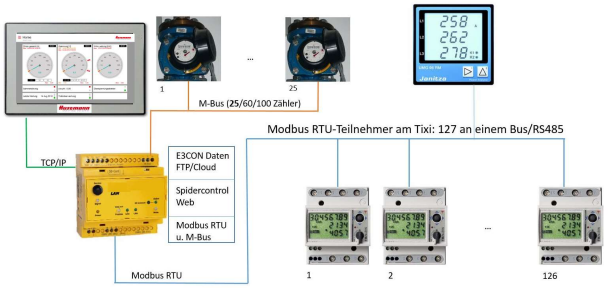
SpiderControl™

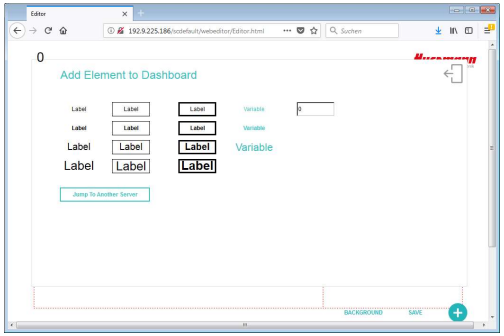
Virtual Private Infrastructure

Beispiel 1: Energiemanagement

Die “Firmware”:

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





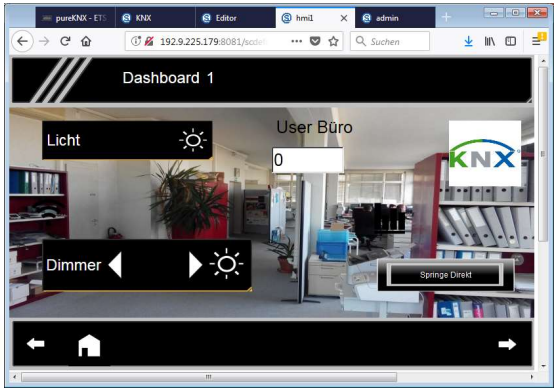
Die “Fieldware”: Programmiert mit SpiderPLC

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

SpiderControl™

Virtual Private Infrastructure

Beispiel 2:
ETS KNX Touch Panel

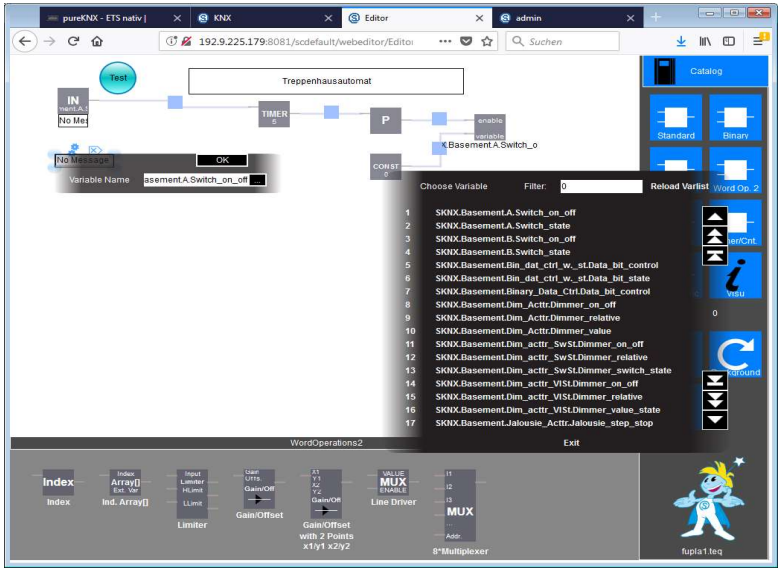


Die “Firmware”:

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.



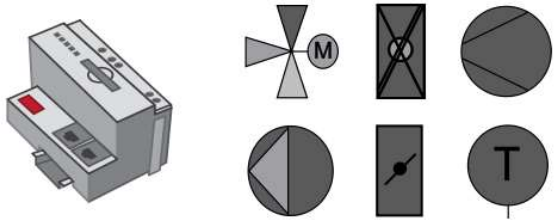
SpiderControl™

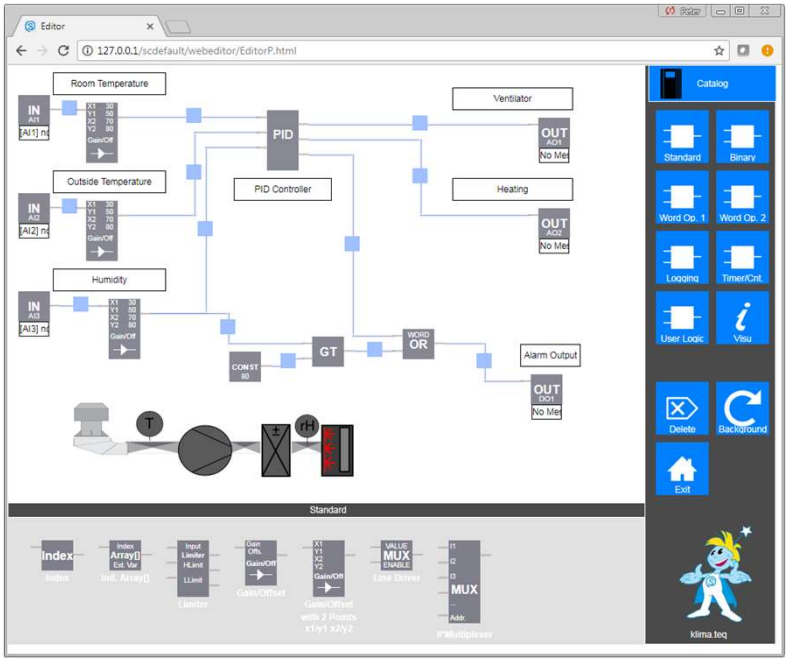
Virtual Private Infrastructure

Beispiel 3:
Industrieregler Gebäudeautomation

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 zusamengebaut hat.





SpiderControl™

Virtual Private Infrastructure

Beispiel 4:
Kleinsteuerung mit LCD

SmartPLC

Die neuen SmartPLCs werden als Kompakt SPS eingesetzt und übernehmen Steuerungsaufgaben, Regelungstechnik, Datenerfassung oder sind für den Einsatz als Gateway für Industrie 4.0-Anwendungen entwickelt. Die Erstellung des SPS-Programms und der HMI-Oberfläche basiert auf HTML5 und ist mit jedem aktuellen Standard-Browser per PC oder Tablet möglich. Dafür ist die Funktionsplan-Programmierung (FuP) als Web-Applikation auf der SPS verfügbar. Es sind keine externen Engineeringtools erforderlich – das SPS-Programm und die Entwicklungsumgebung sind auf der Steuerung abgelegt. Die HMI-Oberfläche ist als Web-Visualisierung konzipiert und kann mit jedem HTML5-Browser bedient werden, auch per Tablet oder Smartphone. Der Zugriff mit den SPS-Bedienterminals der XS-Serie ist ebenfalls möglich.

Es sind zwei Bauformen mit integriertem 4,3-Zoll-Touchdisplay verfügbar und eine weitere Box-Variante zur Montage auf der Hutschiene.

SpiderControl™

Virtual Private Infrastructure

Kampfwertsteigerung

Der Kunde hat ein System, welches für eine spezifische Aufgabenstellung (Nische) eingesetzt werden kann und dort klare Vorteile (USP's) gegenüber anderen Lösungen aufweist.

Durch den Zugewinn an Flexibilität durch eine einfache Programmierung 'mit Bordmitteln' vergrössert sich die Anzahl möglicher Projekte signifikant.

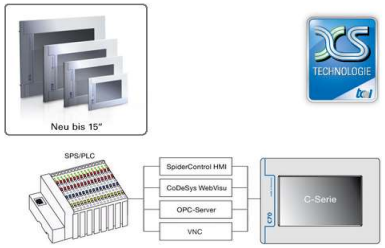
→ Das selbe Produkt kann an doppelt so viele Kunden verkauft werden.

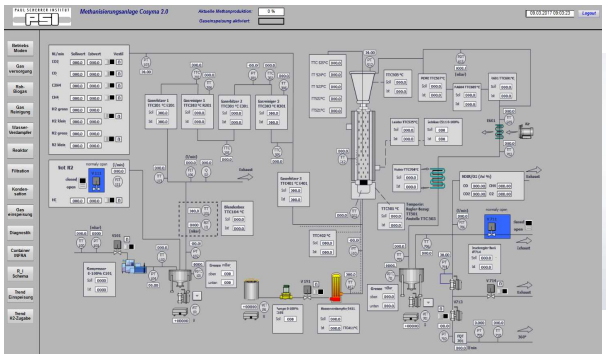
Was muss so ein Tool können?

- **Programmiersprache muss von Elektrikern verstanden werden: Funktionsplan**
- **Integrierbar auf embedded HW und SPS**
- **Aufrufen von externen Funktionen aus IEC61131, Java-Script, PHP, usw.**
- **Einfaches 'Customizing' in Funktion und Design**
- **Möglichst einfaches Interface zum Endbenutzer: “Es spricht seine Sprache”.**

Das SpiderPLC Konzept

- Weiterentwicklung aus einer industriellen Web-HMI Tool-Chain SpiderControl™
- Verwendet die selben Bausteine
- Ist bereits auf vielen SPS und den meisten OS verfügbar, auch embedded (Cortex M0/3/4)
- Der 'Editor-Editor': Einfache Entwicklung eines eigenen Programmiertools

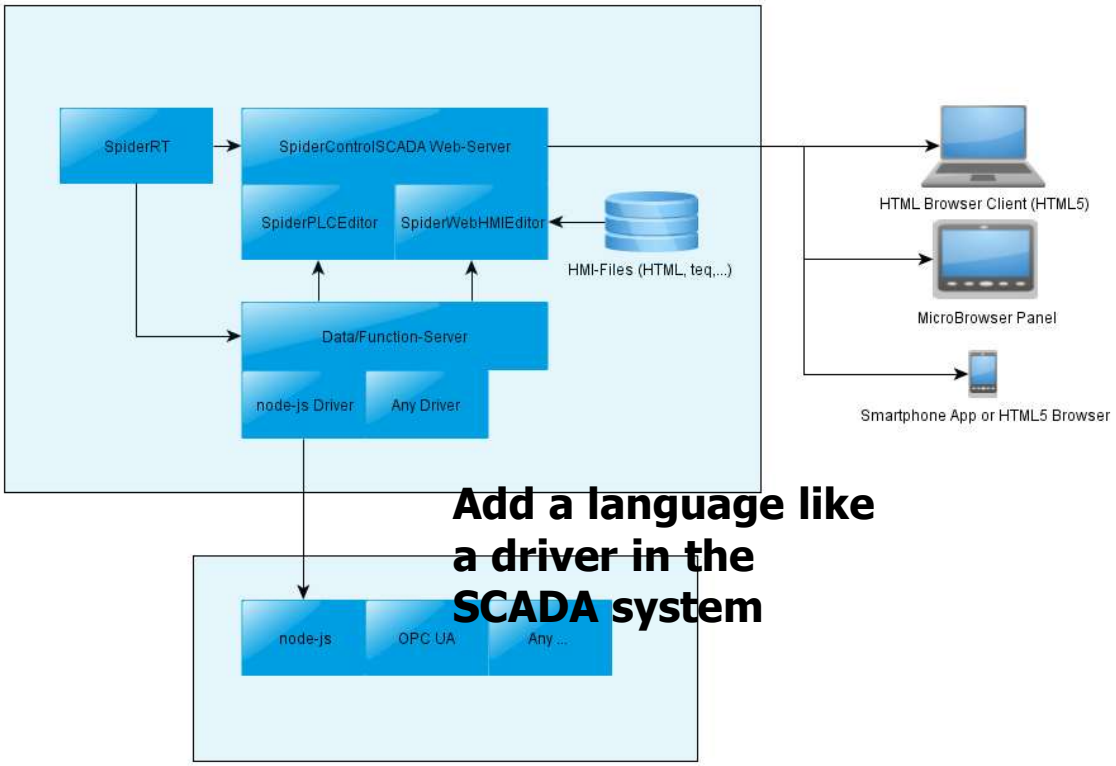




SpiderControl™

Virtual Private Infrastructure

Integrierbar auf jede HW: Teil des SCADA Frameworks

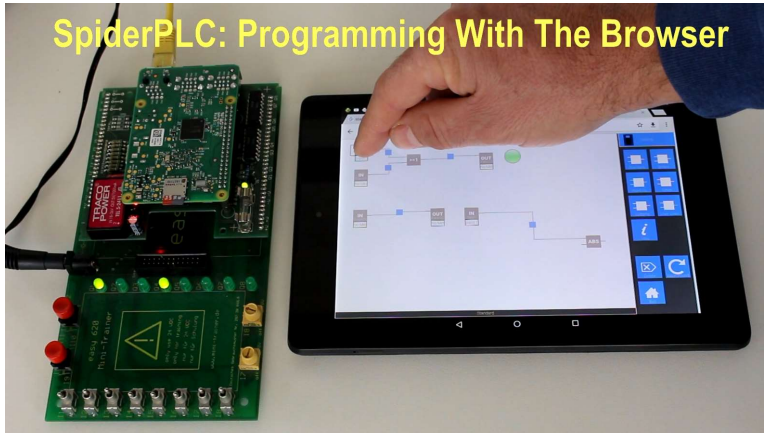


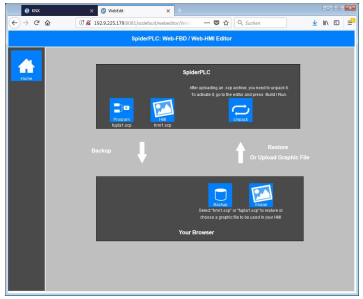
Add a language like a driver in the SCADA system

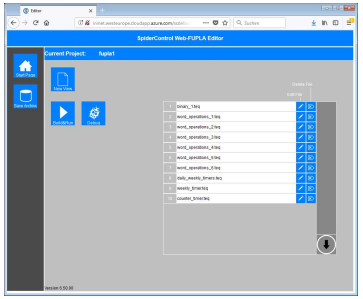
Die Zukunft von SCADA:
-Kombiniere Daten und Funktionen
-Treiber können Daten lesen
-Treiber können aber auch Funktionen aus Dritt-Systemen aufrufen (dies ist z.B. ein wichtiger Teil der OPC UA Spezifikation)
-SpiderPLC ist das benutzerkonfigurierbare grafische Interface, um Funktionen() aus verschiedenen Sprachen zusammenzubringen

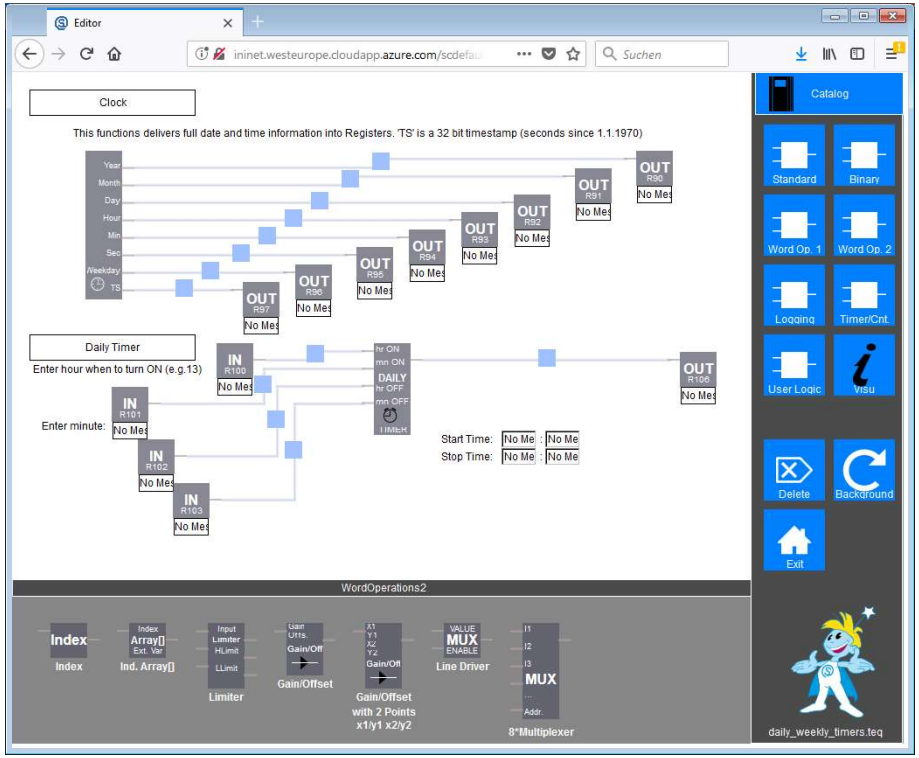
SpiderPLC: FUP/HMI -> Einfache SPS Programmierfunktionen

https://www.youtube.com/watch?v=mbtmtWpSeTc
https://www.youtube.com/watch?v=e5icGf0eBbc





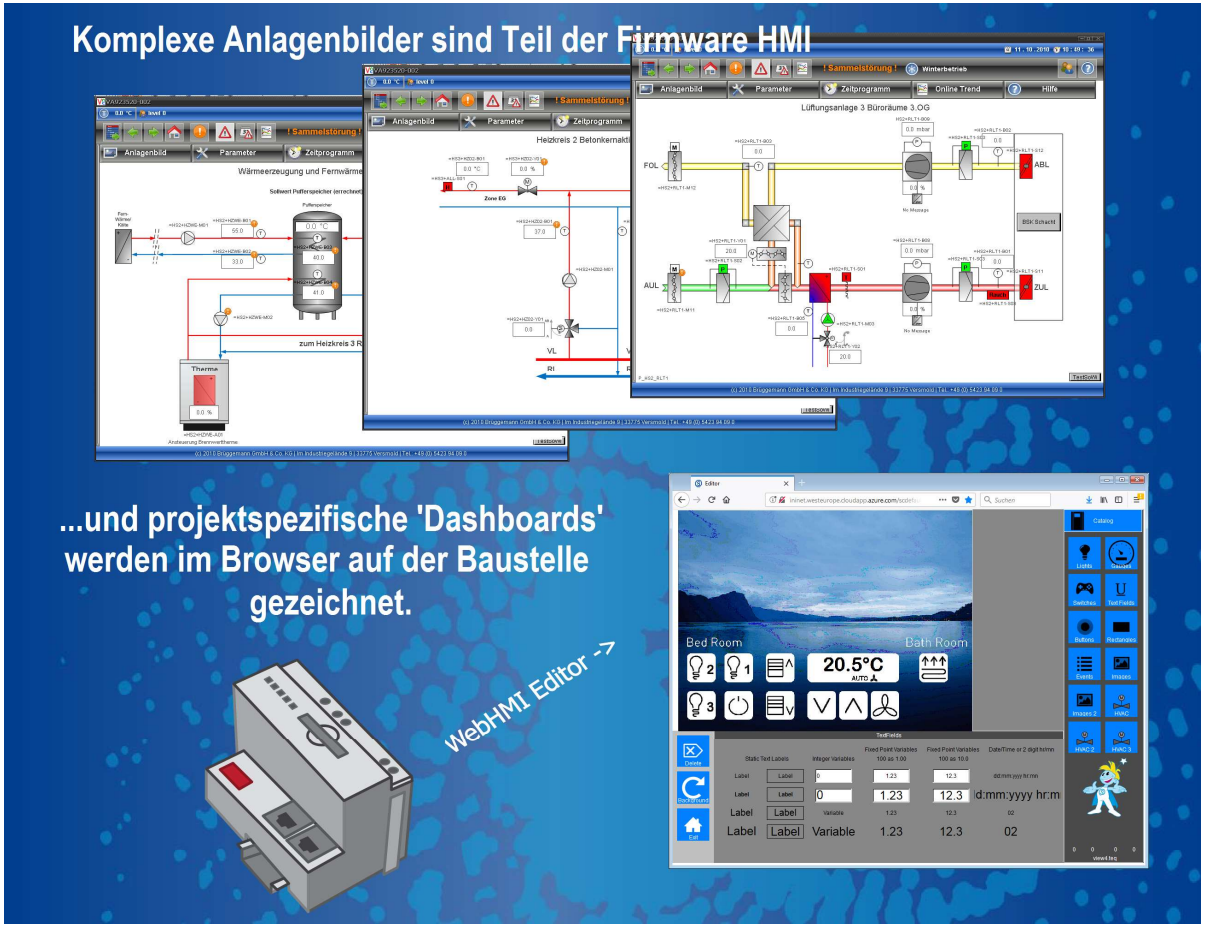




SpiderControl™

Virtual Private Infrastructure

HMI Anpassung im Feld

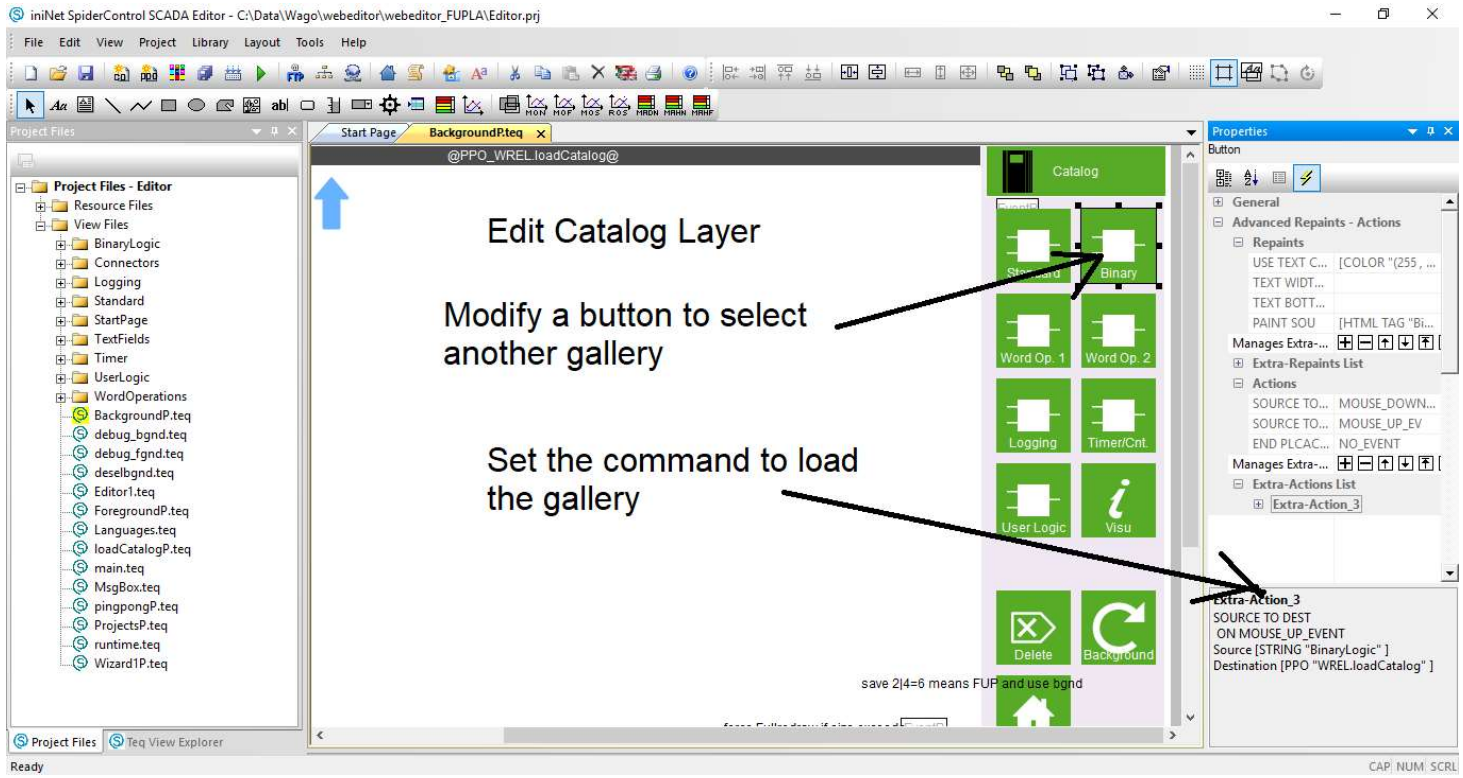


SpiderControl™

Virtual Private Infrastructure

Der Editor

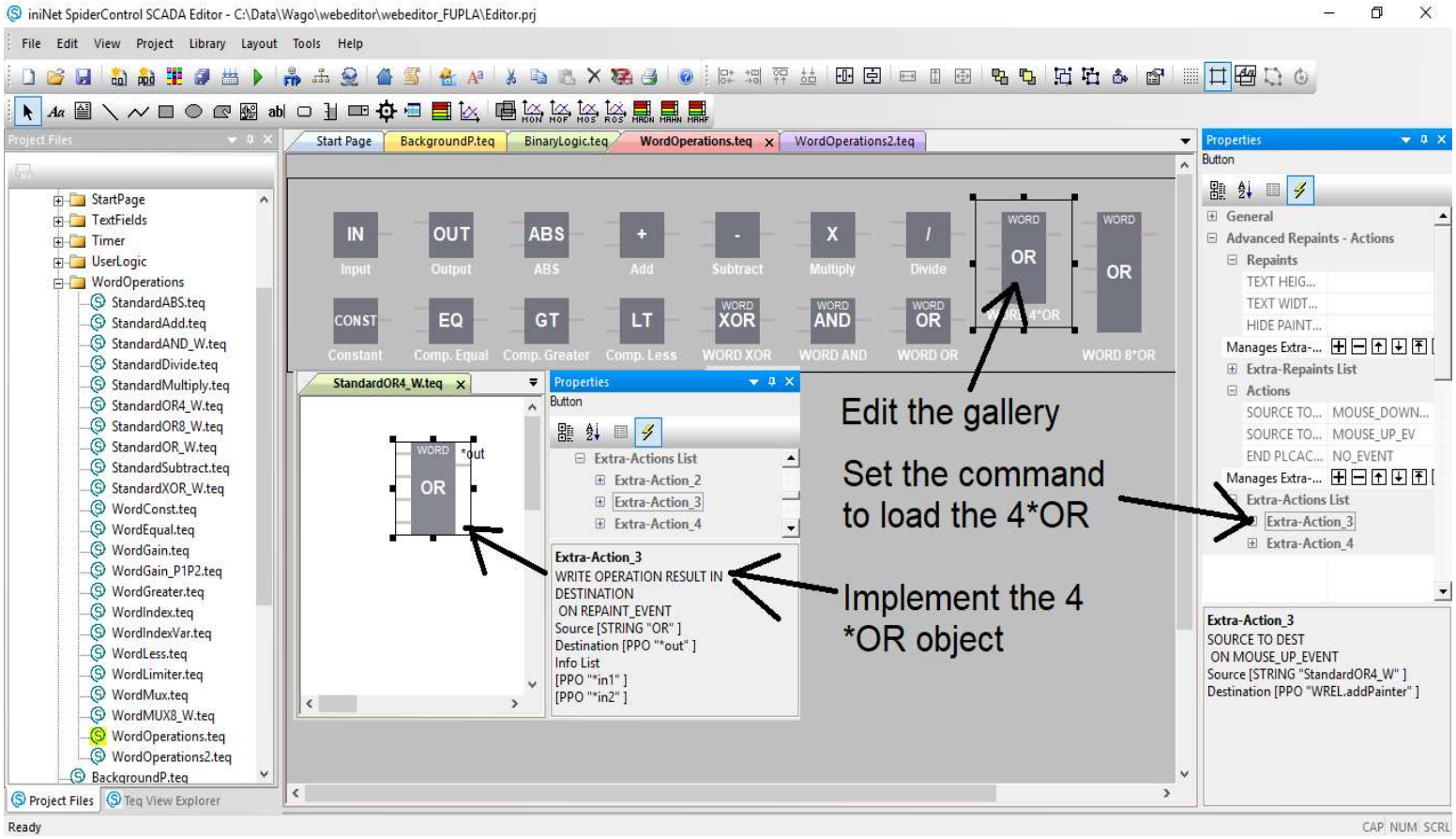
Firmware Entwickler designen ihr eigenes Tool mit PC-HMI Editor



SpiderControl™

Virtual Private Infrastructure

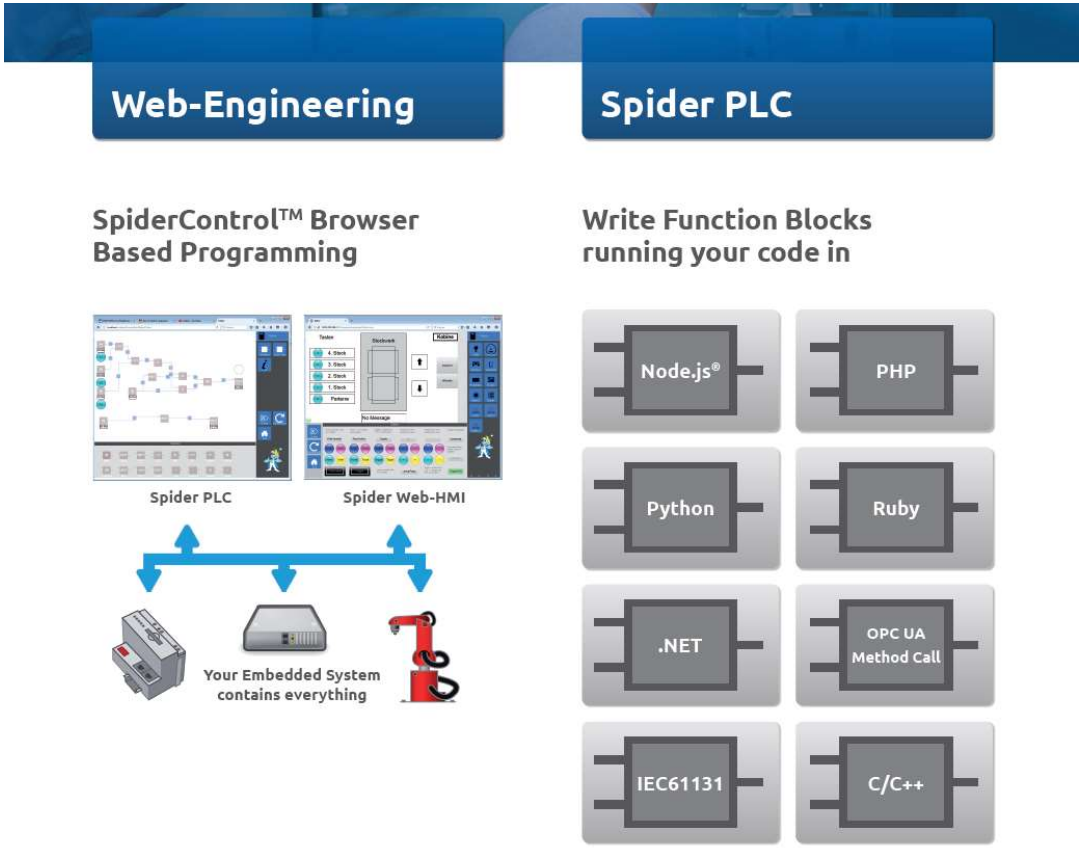
...und definieren die Objekte und Funktionen



SpiderControl™

Virtual Private Infrastructure

SpiderPLC ruft 3rd Party Code auf



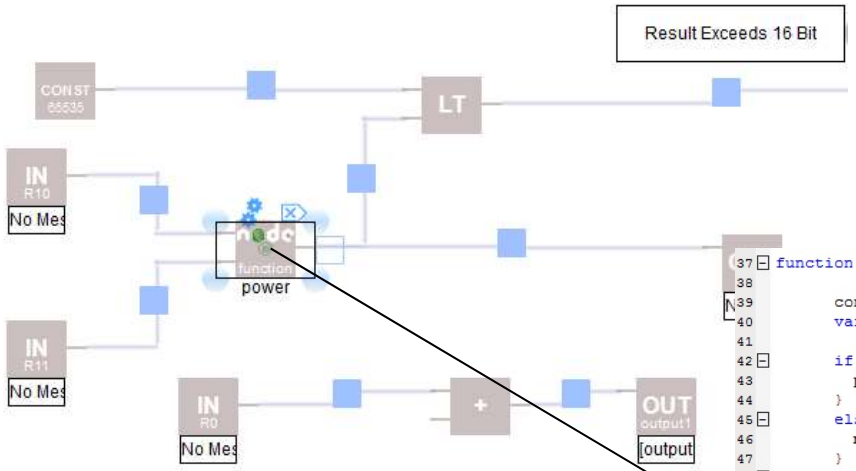
Function Block Programming:

Einfache Logik wird in SpiderRT ausgeführt
Kundenprogrammierte Funktionsbausteine rufen Funktionen aus externen Laufzeiten (Node.js, Python, PHP, C/C++, IEC61131, etc.)

SpiderControl™

Virtual Private Infrastructure

Aus der Sicht des Programmierers



Lokale Variablen der SPS oder von Treibern 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, {'
        response.write("<body>");
```

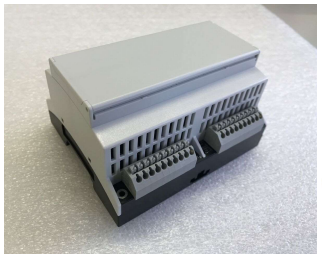
Der Anwender platziert ein FB, welcher eine Funktion in js ausführt

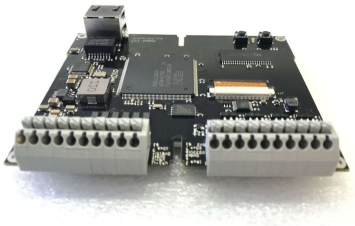
Variablen aus node-js können in SpiderPLC sichtbar werden

SpiderControl™

Virtual Private Infrastructure

SpiderPLC:
Erste Plattformen





SpiderControl™

Virtual Private Infrastructure

SpiderPLC gleich zwei mal nominiert!

20 Jahre E Elektronik
Produkte des Jahres
2018 NOMINIERT

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

CERTIFICATE

Nominee of the

embeddedaward2018

iniNet Solutions GmbH

Category
Tools

Product
SpiderPLC

Nuremberg, 23 January 2018

Prof. Dr.-Ing. Matthias Sturm
Chairman of the Jury

Dr. Roland Fleck
CEO NürnbergMesse Group

Peter Ottmann
CEO NürnbergMesse Group

DESIGN & ELEKTRONIK
KNOW-HOW FÜR ENTWICKLER

NÜRNBERG MESSE

SpiderControl™

Virtual Private Infrastructure

Vielen Dank für Ihre Aufmerksamkeit!

iniNet Solutions GmbH
Fichtenhagstrasse 2
4132 Muttenz
Schweiz

Tel: +41 61 716 96 26
Fax:+ 41 61 716 96 17

E-Mail: info@ininet.ch
Web: www.ininet.ch

SpiderControl™

Virtual Private Infrastructure