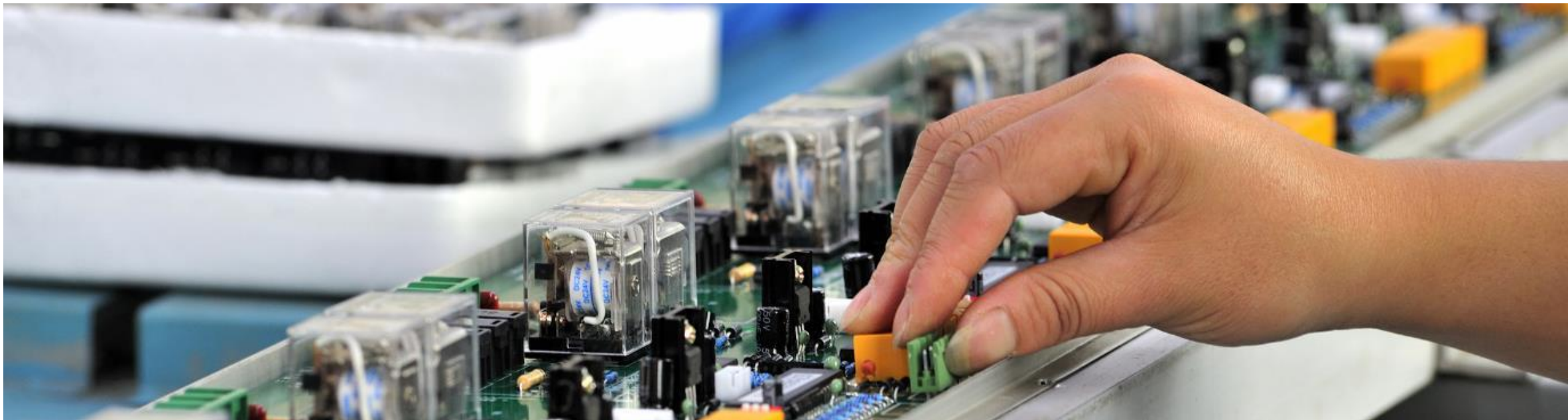


IoT Workshop

TREBING + HIMSTEDT



Niclas und Christoph

Ablauf

17:00 Start

17:00 - 17:45 Uhr Einweisung Hardware und Node-Red Szenario

17:45 - 18:00 Uhr Einweisung SAP Cloud

18:00 - 19:30 Uhr Do It Yourself + Pizza

19:30 Uhr Ergebnisse

Gruppen

Gruppe Stuttgart -> Ziel Sensor bauen für das Stuttgart Büro => Christoph

Gruppe 1 Schwerin -> Ziel Sensor bauen (4 Personen) => Christoph

Gruppe 2 Schwerin -> Ziel Sensor bauen (4 Personen) => Christoph

Gruppe 3 Schwerin -> Ziel Node-Red ertüchtigen (3 Personen) => Christoph

Gruppe 4 Schwerin -> Ziel SAP Cloud ertüchtigen (4 Personen) => Niclas

Bei Fragen kommt gerne auf uns zu!

Viel Spass

Definition

The Internet of things (IoT) is the network of physical devices, vehicles, home appliances and other items embedded with electronics, software, sensors, actuators, and connectivity which enables these objects to connect and exchange data. Each thing is uniquely identifiable through its embedded computing system but is able to interoperate within the existing Internet infrastructure.

Vgl. https://en.wikipedia.org/wiki/Internet_of_things

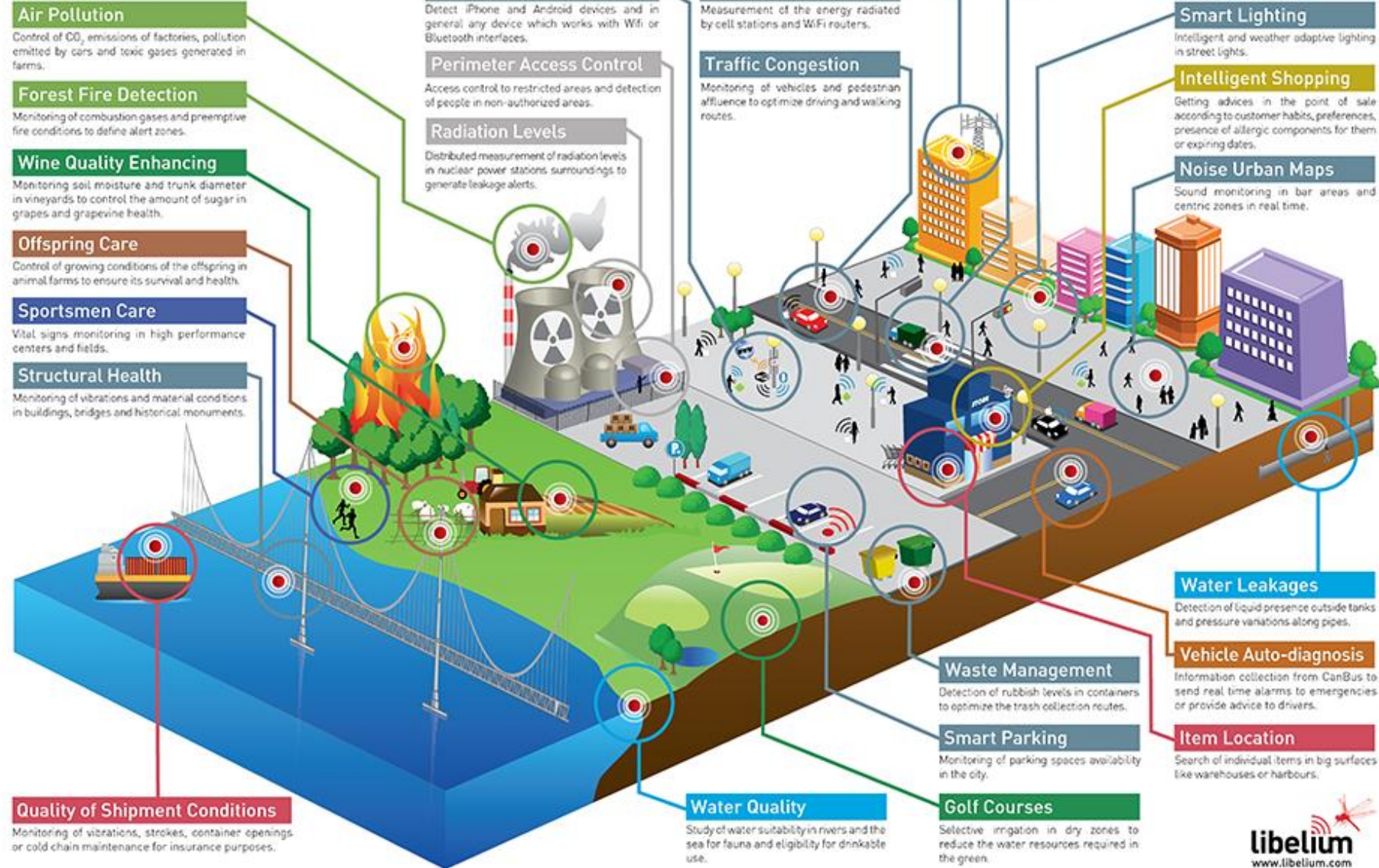
Der Begriff wird für vieles verwendet was mit dem Internet verbunden wird: <https://www.expertenderit.de/blog/iot-definitionen-was-ist-eigentlich-das-internet-der-dinge>

Der Begriff ist nicht neu - erstes Auftreten 1999 - Ideen bereits in den 1980er Jahren.

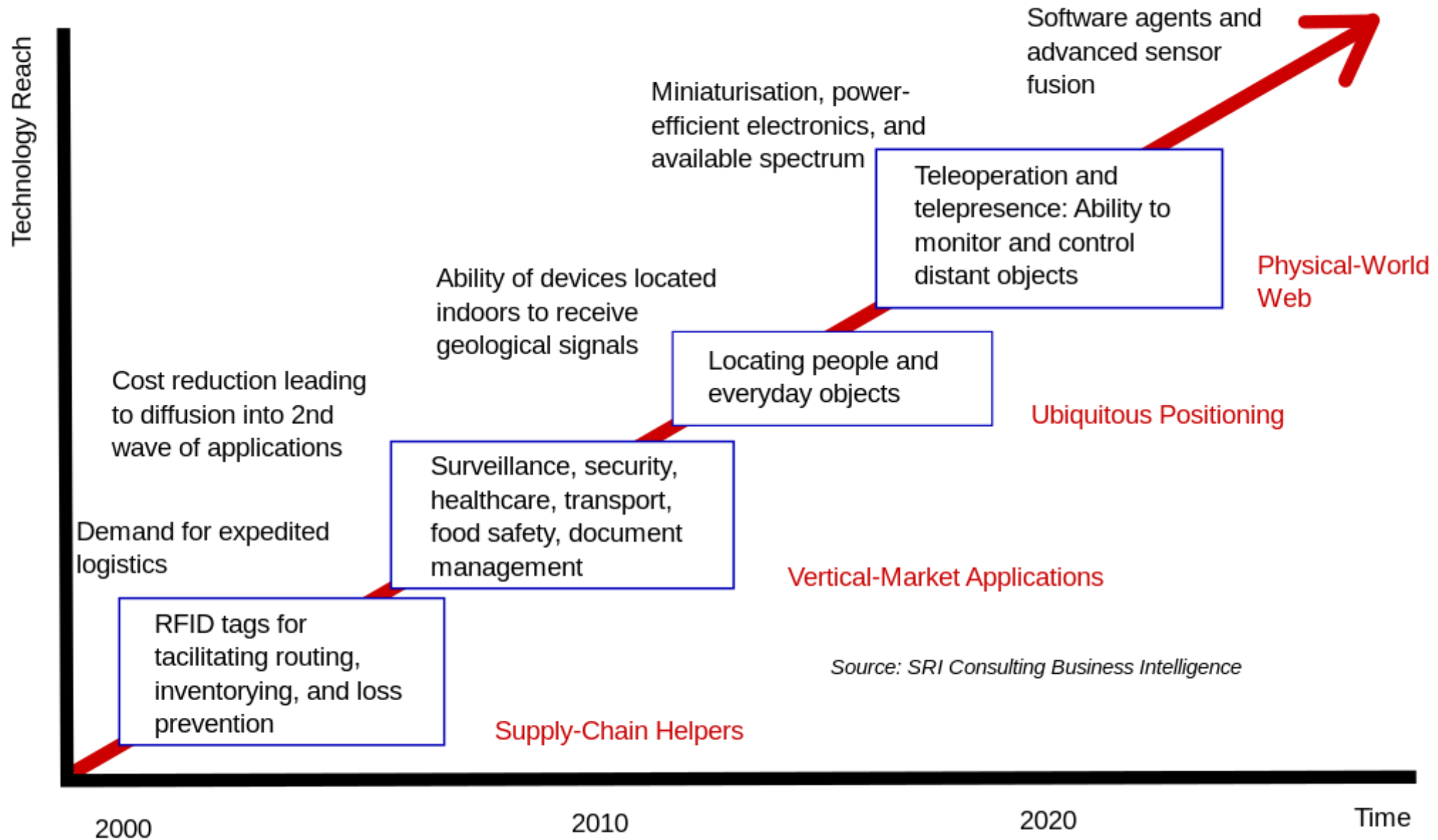
Neu ist die Menge und Leichtigkeit mit der die Geräte verbunden werden können.

Der Begriff wird auch gerne für das Marketing verwendet um mit dem Internet verbundene Dienste zu bezeichnen.

Libelium Smart World



Technology roadmap: The Internet of Things



Beispiele für IoT Geräte

Amazon Dashbutton zeitweise für 2 € mit WLAN / Bluetooth
Super zum Steuern von einfachen Aktionen

Nur für Prime Mitglieder :-)

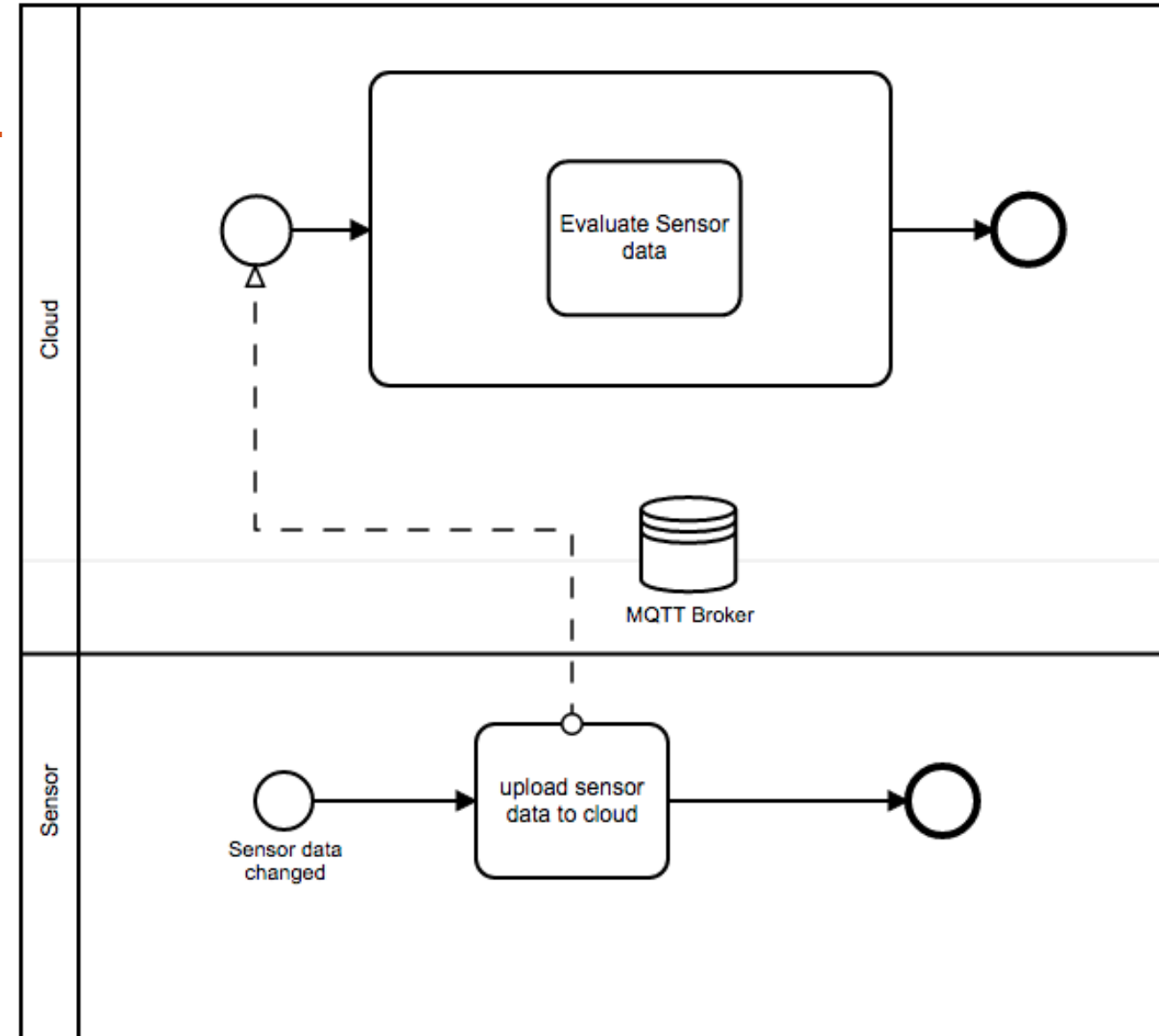
CCC Test und Debugging für Sicherheit bestanden



Was wollen wir machen

Sensordaten auslesen und in die Cloud schreiben.

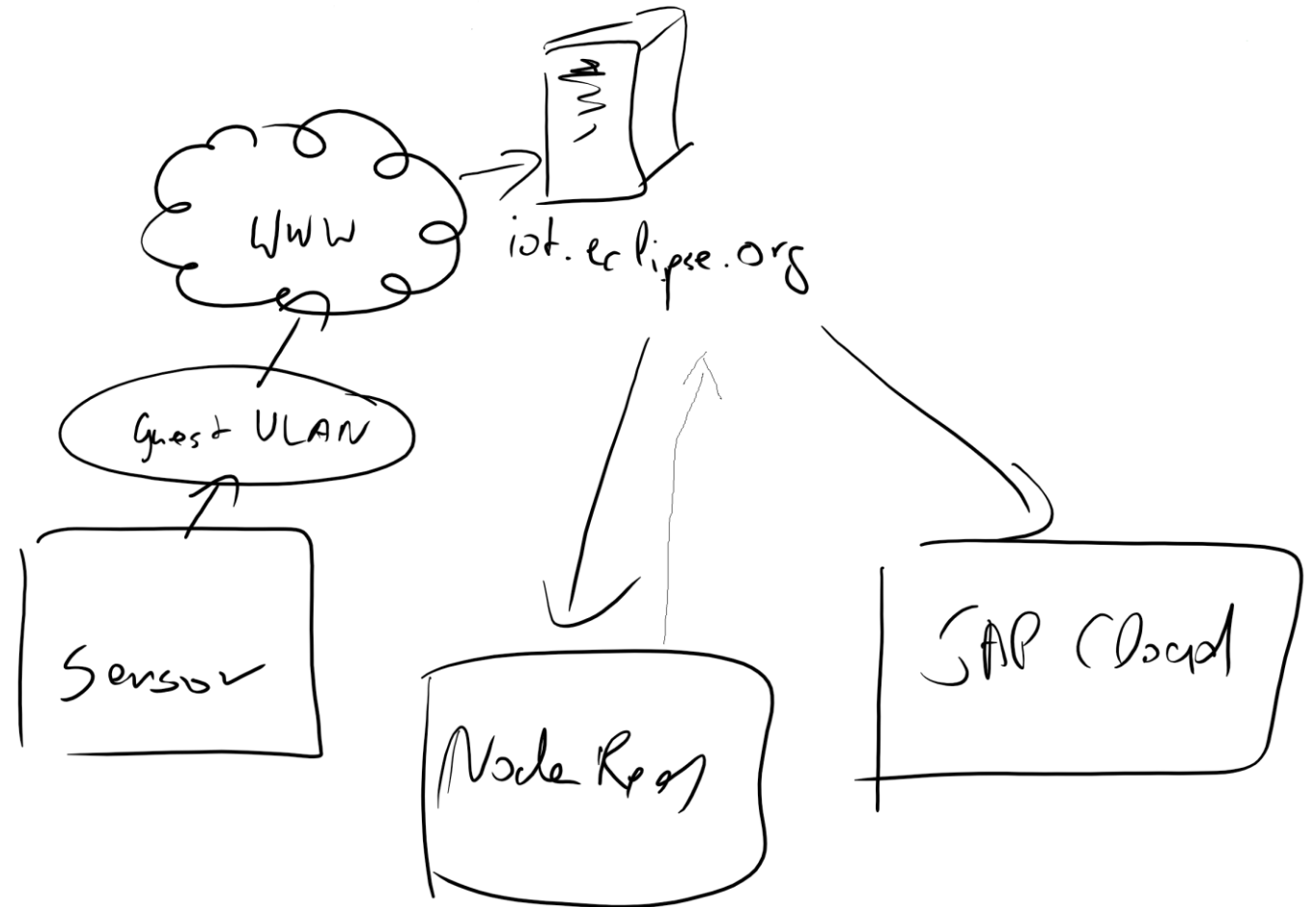
Sensordaten visualisieren.



Was wollen wir machen

Sensordaten auslesen und in die Cloud schreiben.

Sensordaten visualisieren.



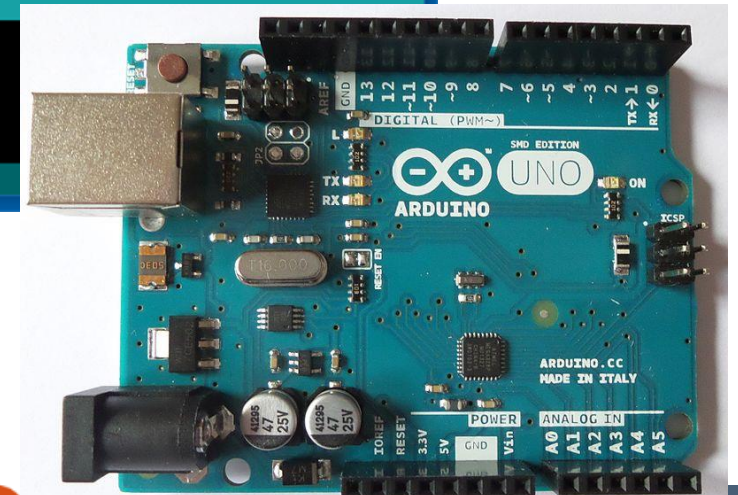
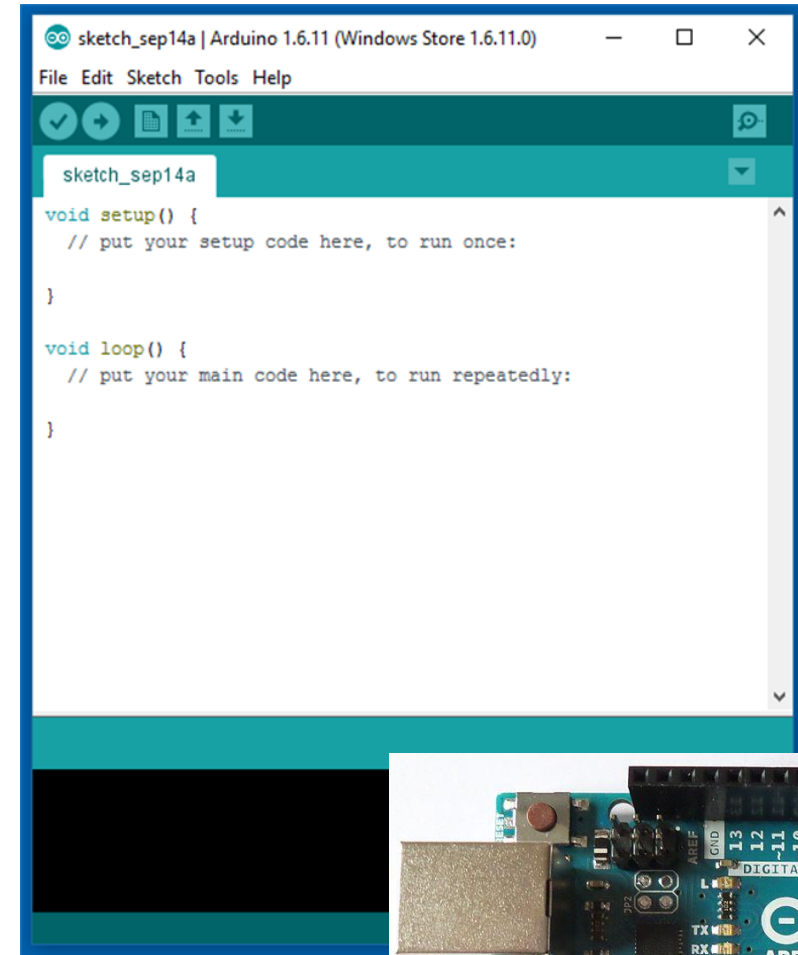
Grundlagen für Alle

Grundlagen Software

Arduino IDE - Entwicklungsumgebung für die Entwicklung von Programmen für Arduinos

MQTT - offenes Nachrichtenprotokoll für Machine-to-Machine-Kommunikation (M2M). OASIS Standard.

<https://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.html>



Grundlagen Hardware

NodeMCU / ESP8266 (2-3 €)

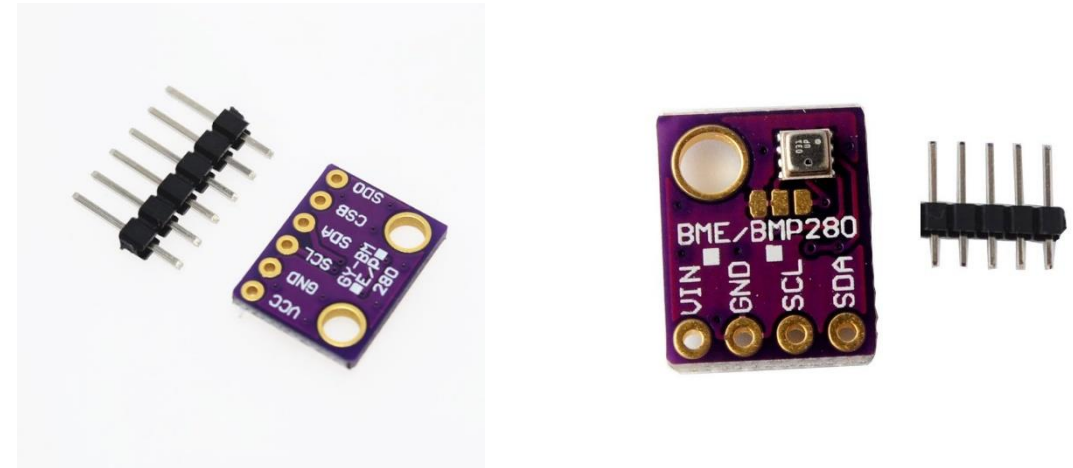
Wifi 32bit Microcontroller



Bosch BME280 (3 €)

Temperatur / Luftfeuchte / Luftdruck

-40...+85 °C, 0...100 % rel. humidity, 300...1100 hPa



Achtung auch als BMP280 mit nur Temperatur und Luftdruck – Sorry Max ☹

Grundlagen Hardware

NodeMCU / ESP8266 - 32bit Microcontroller mit Wifi

Lua-basierte interaktive Programmierung (NodeMCU)

Micropython (Pythonbasierte interaktive Programmierung)

Arduino/C++ basierte Programmierung

AT-Command für die Nutzung als Seriell-zu-WLAN-Schnittstelle

ESP Easy zur Ansteuerung von Sensoren/Aktoren über Wlan

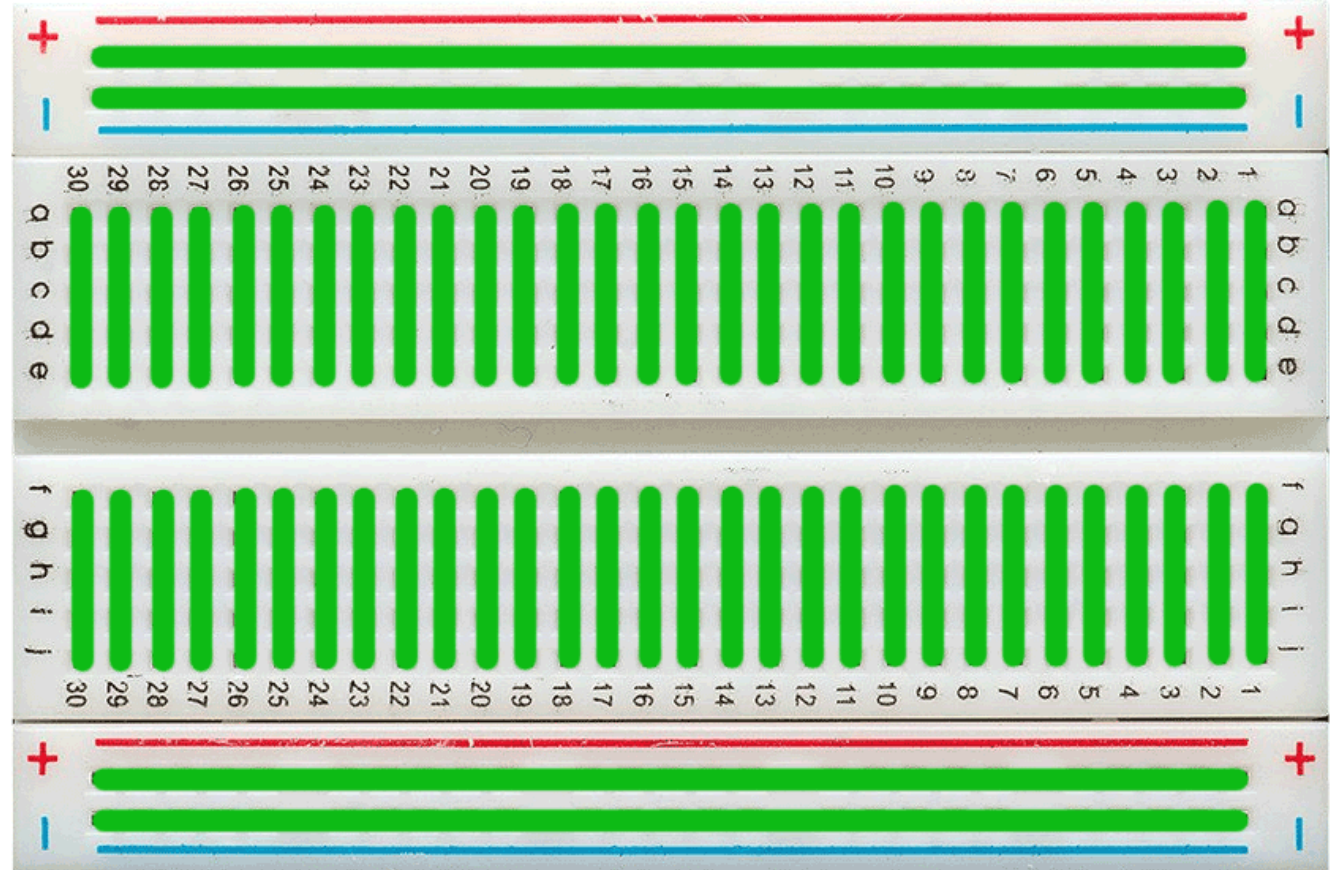
ESP Basic

Breadboard - Steckbrett

+ - sind jeweils von links nach rechts verbunden

das Steckfeld in der Mitte ist spaltenweise verbunden

+ - und 3,3Volt und 5 Volt nicht verwechseln



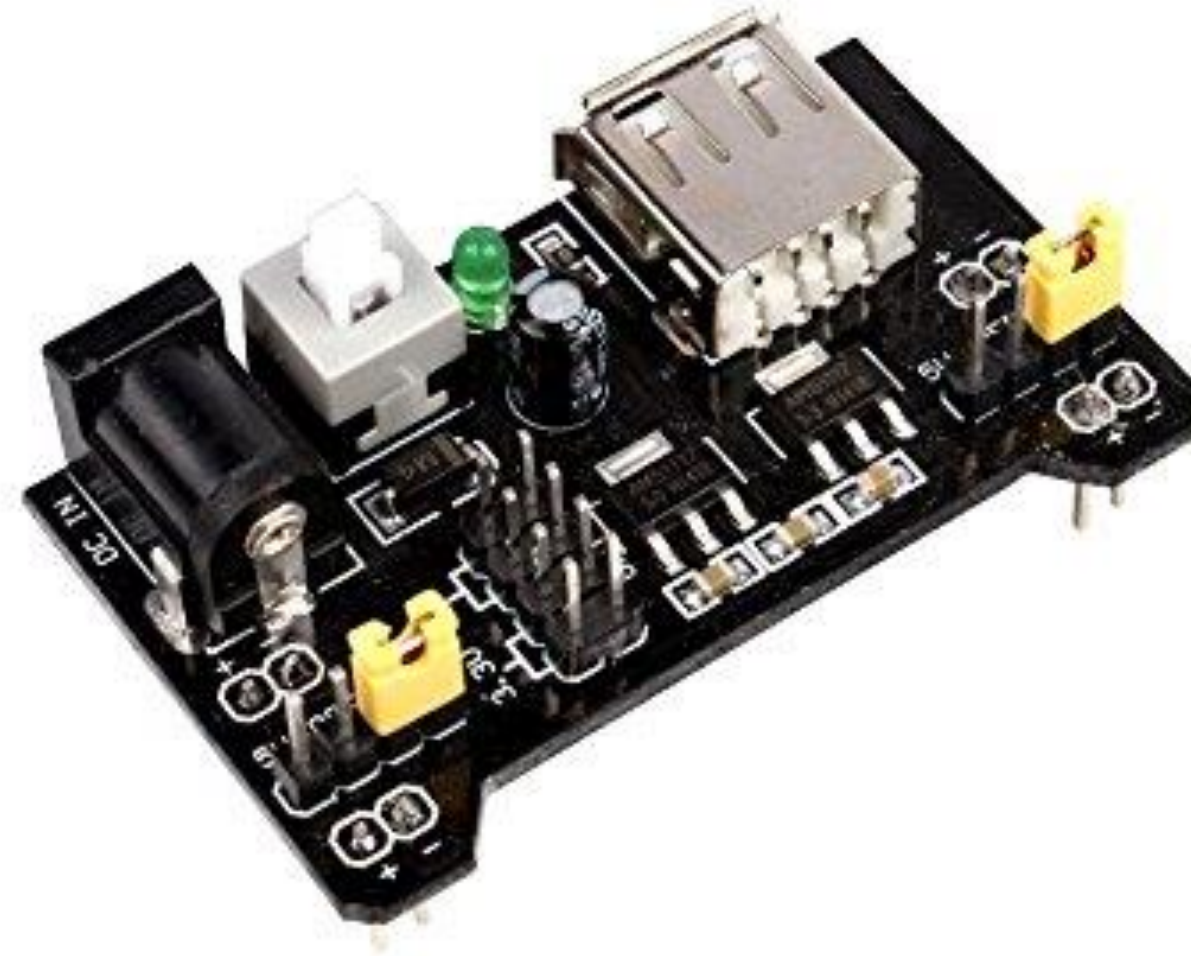
Stromversorgung

bitte nur Arbeiten wenn Netzteil ausgezogen ist

Wichtig, entweder Stromversorgung über das Netzteil oder über USB

+ - und 3,3Volt und 5 Volt nicht verwechseln

ggf. gemeinsame Masseleitung verwenden



Grundlagen Netzwerkdienste

<http://iot.eclipse.org> bietet MQTT Server als Public Dienst an. Wir verwenden diesen hier als Datendrehscheibe.

Achtung: Alle Daten sind öffentlich sichtbar!

Alternativ können die Geräte auch direkt HTTP uvm.

Node-Red

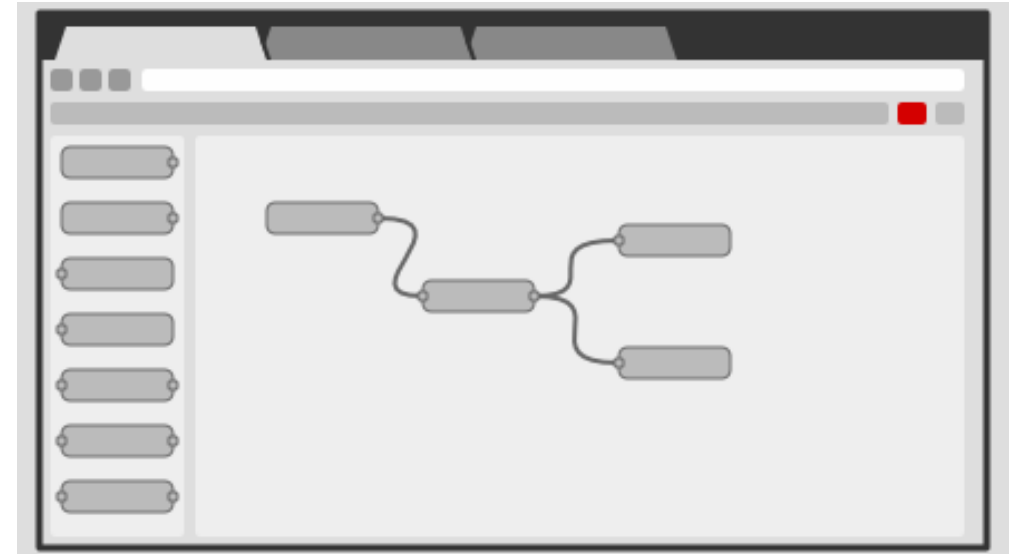
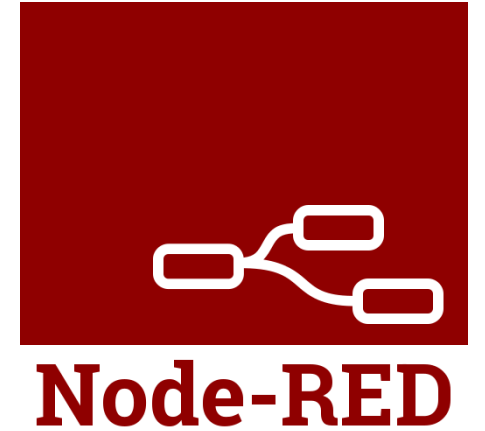
Node-Red

Open-Source webbasierte Oberfläche um Flow-basierte Programmierung auszuführen

Konzept aus den 70er Jahren von J. Paul Morrison

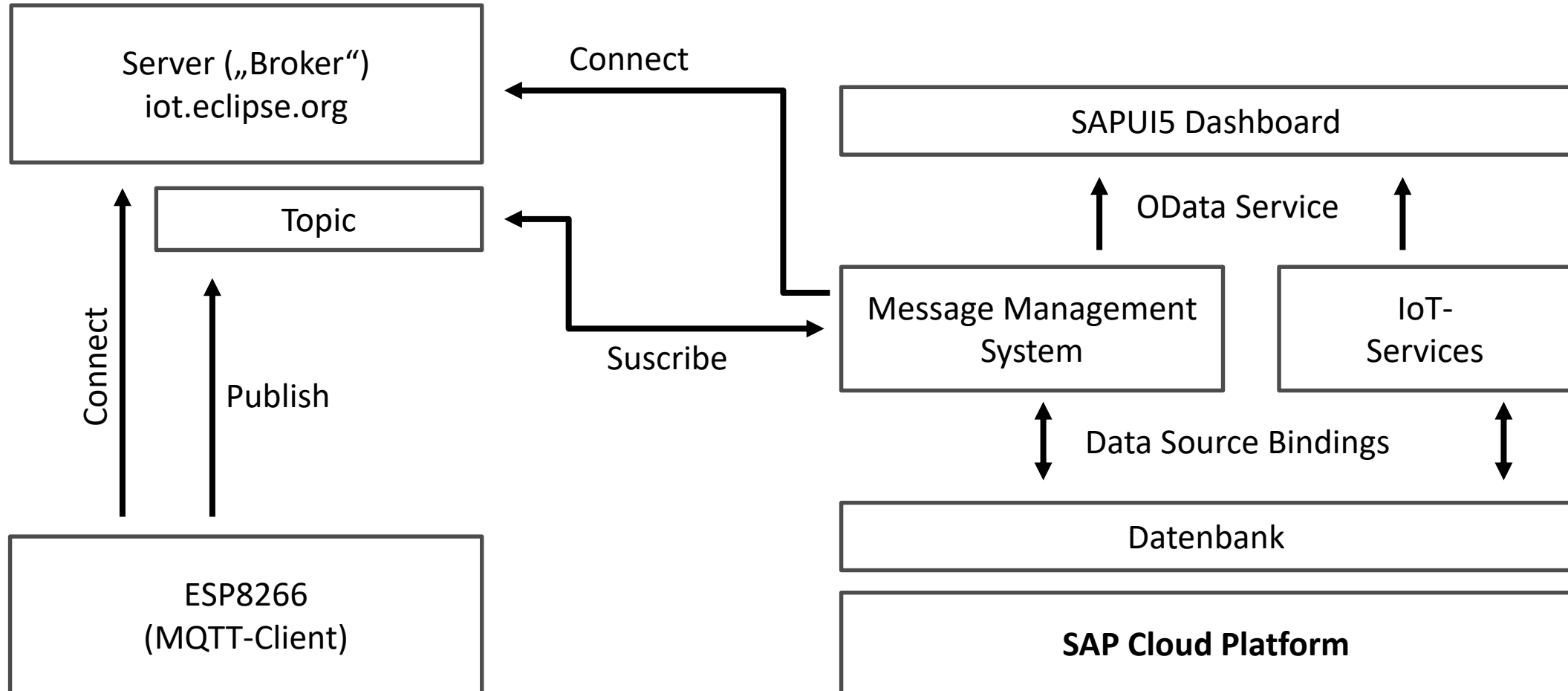
Ursprünglich als IBM Projekt gestartet, jetzt von der JS Foundation verwaltet

<https://nodered.org/>



SAP Cloud

Architektur



Absprache Dashboard und Bastel-Team

In der SCP wird angelegt

- Account
 - UserID

In den IoT-Services werden angelegt

- Message Type
 - MessageID
 - Fields / Felder
- Device Type
- Device
 - DeviceID

Das ESP8266 Published an das Topic

- `iot/data/iotmms<UserID>trial/v1/<DeviceID>`
die folgende Nachricht:

```
{  
    "mode": "async",  
    "messageType" : Guid.raw(),  
    "messages":  
        [{"tempC": msg.tempC,  
         "humidity": msg.humidity,  
         "pressure": msg.pressure}]  
}
```


FRAGEN

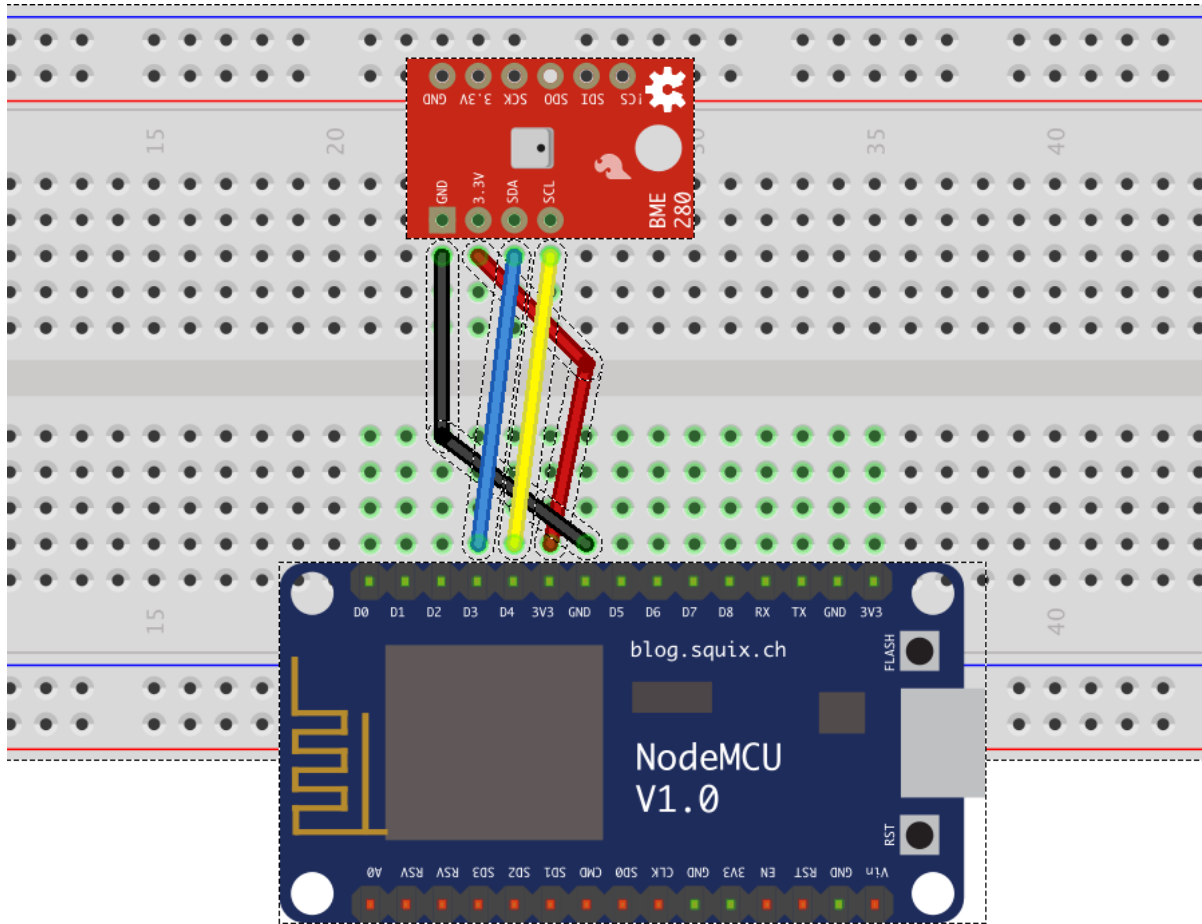
DO IT YOURSELF

Gruppen: Sensor bauen

Ziel

Sensor Daten in der Cloud per MQTT abliefern

Anleitung Verkabelung



Rot - 3,3V -> VCC/VIN

Schwarz - G -> GND = Masse

Blau - D3 -> SDA

Gelb - D4 -> SCL

Raspberry Pis

Ist ein Linux Rechner aber nicht schwer zu bedienen:

<https://www.raspberrypi.org/blog/kids-and-their-raspberry-pis/>

Auf jedem Desktop gibt es ein Verzeichnis `iotworkshop`. Dies enthält die Libraries und die einzelnen Schritte mit dem jeweiligen Code.

Der Code kann über das `update.sh` Kommando aktualisiert werden.

Die Arduino IDE befindet sich als Link auf dem Desktop.

Bitte unter Tools den USB Port auswählen.

Programmierung Wifi Verbindung herstellen

Lesson1.ino öffnen

Codestellen:

```
const char* ssid = ".....";
```

```
const char* password = ".....";
```

Mit WLAN SSID Guest und passendem Passwort anfüllen.

Auf das Gerät laden und danach den Seriellen Monitor öffnen.

Programmierung Wifi Verbindung herstellen

```
#include <ESP8266WiFi.h>
```

```
#include <PubSubClient.h>
```

```
// Update these with values suitable for your network.
```

```
const char* ssid = ".....";
```

```
const char* password = ".....";
```

```
const char* mqtt_server = "iot.eclipse.org";
```

```
WiFiClient espClient;
```

```
PubSubClient client(espClient);
```

```
long lastMsg = 0;
```

```
void setup() {
```

```
    Serial.begin(115200);
```

```
    setup_wifi();
```

```
    client.setServer(mqtt_server, 1883);
```

```
}
```


Programmierung Wifi Verbindung herstellen

```
void setup_wifi() {  
    delay(10);  
    // We start by connecting to a WiFi network  
    Serial.println();  
    Serial.print("Connecting to ");  
    Serial.println(ssid);  
  
    WiFi.begin(ssid, password);  
  
    while (WiFi.status() != WL_CONNECTED) {  
        delay(500);  
        Serial.print(".");  
    }  
  
    Serial.println("");  
    Serial.println("WiFi connected");  
    Serial.println("IP address: ");  
    Serial.println(WiFi.localIP());  
}
```

Programmierung Wifi Verbindung herstellen

```
void loop() {  
    long now = millis();  
    if (now - lastMsg > 30000) {  
        lastMsg = now;  
    }  
}
```

Programmierung Lesson 2 Upload

Code auf den NodeMCU laden und prüfen.

Programmierung MQTT Verbindung herstellen

lesson2.ino als Kopievorlage verwenden

```
void reconnect() {  
    // Loop until we're reconnected  
    while (!client.connected()) {  
        Serial.print("Attempting MQTT connection...");  
        // Attempt to connect  
        String clientId = "ESP8266Client-";  
        clientId += String(random(0xffff), HEX);  
        if (client.connect(clientId.c_str())) {  
        } else {  
            Serial.print("failed, rc=");  
            Serial.print(client.state());  
            Serial.println(" try again in 5 seconds");  
            // Wait 5 seconds before retrying  
            delay(5000);  
        }  
    }  
}
```


Programmierung MQTT Verbindung herstellen

lesson2.ino als Kopievorlage verwenden

```
void loop() {  
    if (!client.connected()) {  
        reconnect();  
    }  
    client.loop();  
  
    long now = millis();  
    if (now - lastMsg > 30000) {  
        lastMsg = now;  
        sendMessage();  
    }  
}
```

Programmierung Lesson 2 Upload

Code auf den NodeMCU laden und prüfen.

Programmierung MQTT Verbindung herstellen

lesson2.ino als Kopievorlage verwenden

```
void sendMessage() {  
    double tempC=23.0;  
    double humidity=50;  
    double pressure=1023.0;  
  
    String jsonString = "{\"@c\": \".BME280\", \"tempC\": "+String(tempC)+", \"humidity\": " +  
String(humidity) + ", \"pressure\": "+String(pressure)+", \"device\": \"Group1\"}";  
  
    char charBuf[jsonString.length() + 1];  
    jsonString.toCharArray(charBuf, jsonString.length() + 1);  
    client.publish("t-h.de/sensor", charBuf);  
}
```

Wir liefern erstmal Beispieldaten => ab jetzt sollten die Kollegen im Node-RED etwas sehen.

Wichtig Group1 gegen einen eigenen Namen ersetzen.

Programmierung Sensordaten lesen

lesson3.ino als Kopievorlage verwenden

Passende Header (Bibliotheken) einbinden

```
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_BME280.h>
```

```
Adafruit_BME280 bme;
```

Stuttgart bitte hier die entsprechende Teile aus lesson3bmp.ino verwenden.
BME durch BMP ersetzen und 0 für die Humidity fix vorgeben.

Programmierung Sensordaten lesen

lesson3.ino als Kopievorlage verwenden

setup() Methode erweitern:

```
Wire.begin(0, 2);  
bool status;  
// default settings  
// (you can also pass in a Wire library object like &Wire2)  
status = bme.begin();  
if (!status) {  
  Serial.println("Could not find a valid BME280 sensor, check wiring!");  
  while (1);  
}
```

Programmierung Sensordaten lesen

lesson3.ino als Kopievorlage verwenden

sendMessage() Methode anpassen:

```
double tempC = bme.readTemperature();  
double humidity = bme.readHumidity();  
double pressure = bme.readPressure() / 100.0F;
```


Programmierung Lesson 3 Upload

Code auf den NodeMCU laden und prüfen.

Stromversorgung anschliessen

VIN und Masse (G) am NodeMCU wie folgt verbinden:

Power Supply -> NodeMCU

5V + -> VIN

- -> G

Passende Kabel gibt es bei Christoph

Gruppe Node-Red

Ziele

Sensordaten entgegen nehmen

Sensordaten an SAP Cloud weiterleiten

Sensordaten visualisieren

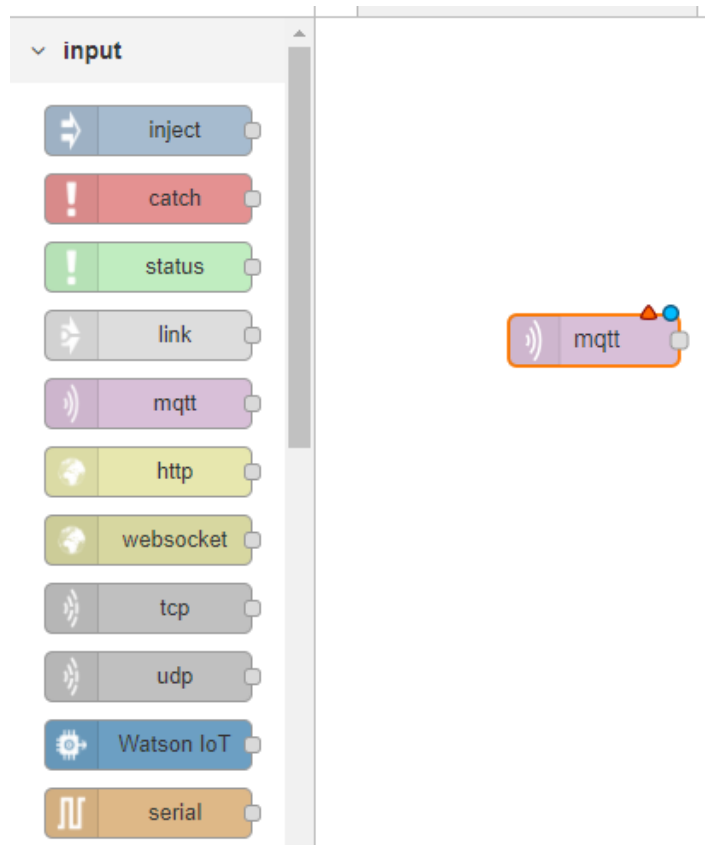
Programmierung Node-Red

Starten über das Start Menü auf dem Raspberry

Webbrowser öffnen und <http://localhost:1880> laden

ein leere Flow ist bereits vorhanden

Programmierung Node-Red Daten von MQTT lesen



Programmierung Node-Red Daten von MQTT lesen

mqtt in > Add new mqtt-broker config node

Cancel Add

Connection Security Birth Message Will Message

Server Port

☐ Enable secure (SSL/TLS) connection

Client ID

Keep alive time (s) ☒ Use clean session

☒ Use legacy MQTT 3.1 support

Programmierung Node-Red Daten von MQTT lesen

Edit mqtt in node


Delete

Cancel

Done

▼ node properties

🌐 Server

iot.eclipse.org:1883 ▼ 

📄 Topic

/t-h.de/sensor

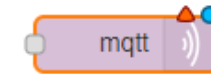
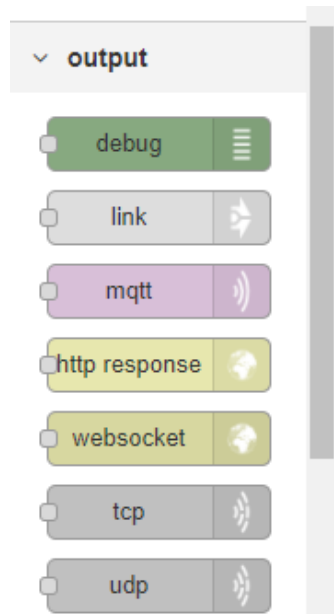
⚙️ QoS

2 ▼

🏷️ Name

Name

Programmierung Node-Red Daten nach MQTT publizieren



Programmierung Node-Red Daten nach MQTT publizieren

Edit mqtt out node

Delete

Cancel

Done

▼ node properties

🌐 Server

iot.eclipse.org:1883

▼

✎

📋 Topic

to be filled in with SAP Team

⚙️ QoS

▼

🔄 Retain

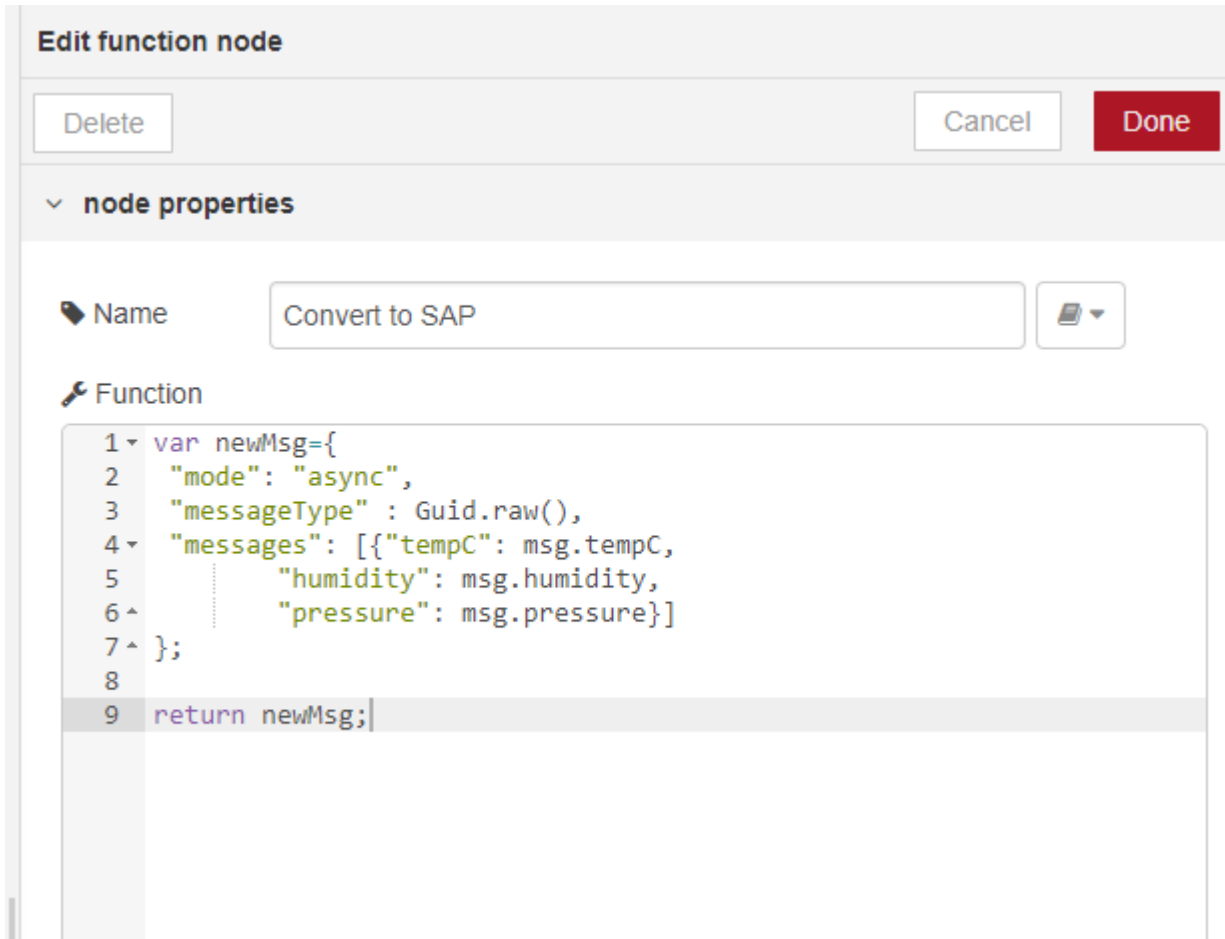
▼

🏷️ Name

Name

Tip: Leave topic, qos or retain blank if you want to set them via msg properties.

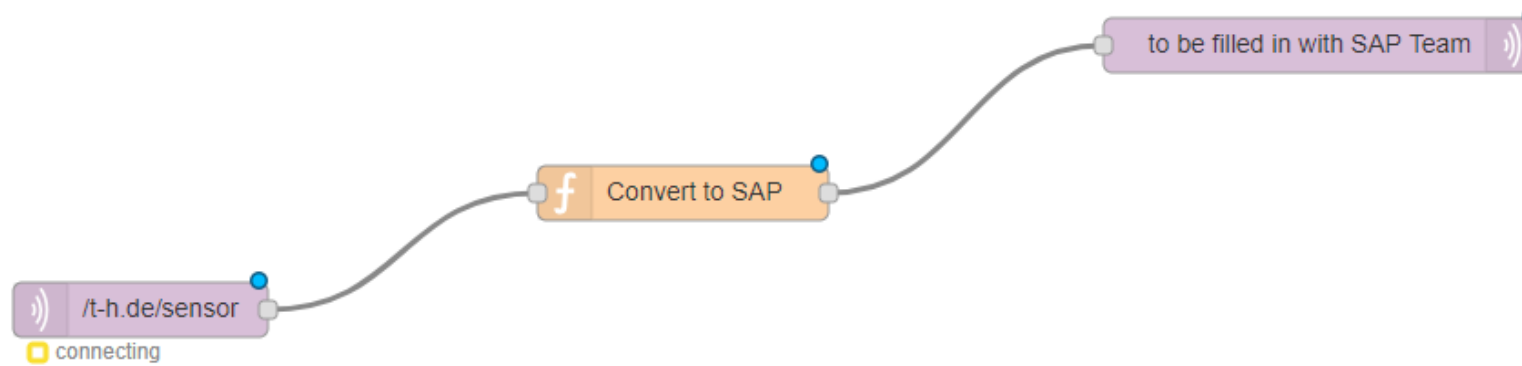
Programmierung Node-Red Daten nach MQTT publizieren



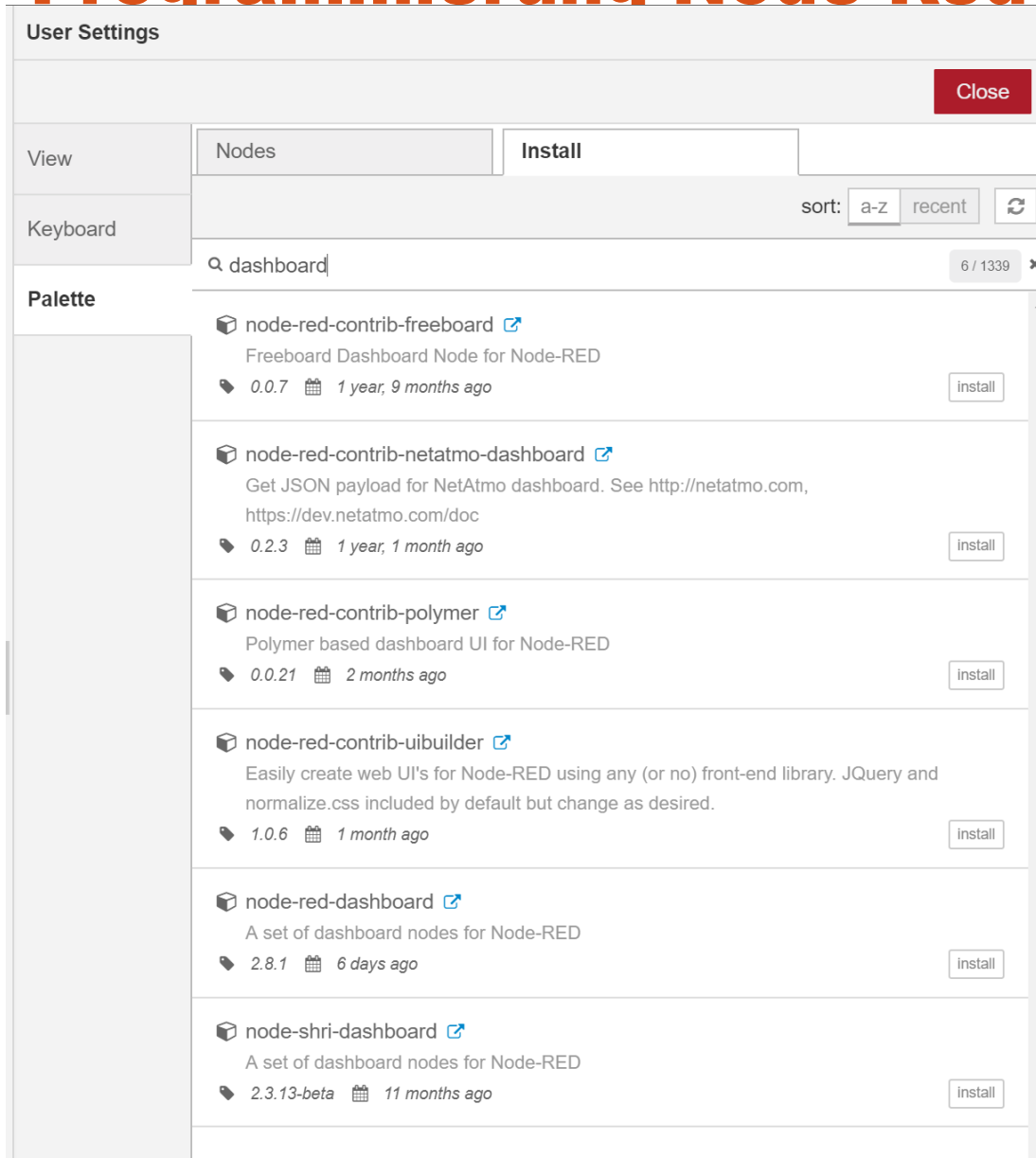
```
var newMsg={
  "mode": "async",
  "messageType" : Guid.raw(),
  "messages": [
    {"tempC": msg.tempC,
     "humidity": msg.humidity,
     "pressure": msg.pressure}]];
return newMsg;
```

Programmierung Node-Red Daten nach MQTT publizieren

Ggf. mehrere Ziele durch
direktes verbinden angeben



Programmierung Node-Red Daten visualisieren



STRG-SHIFT-P drücken

Install Reiter auswählen

Nach „dashboard“ suchen

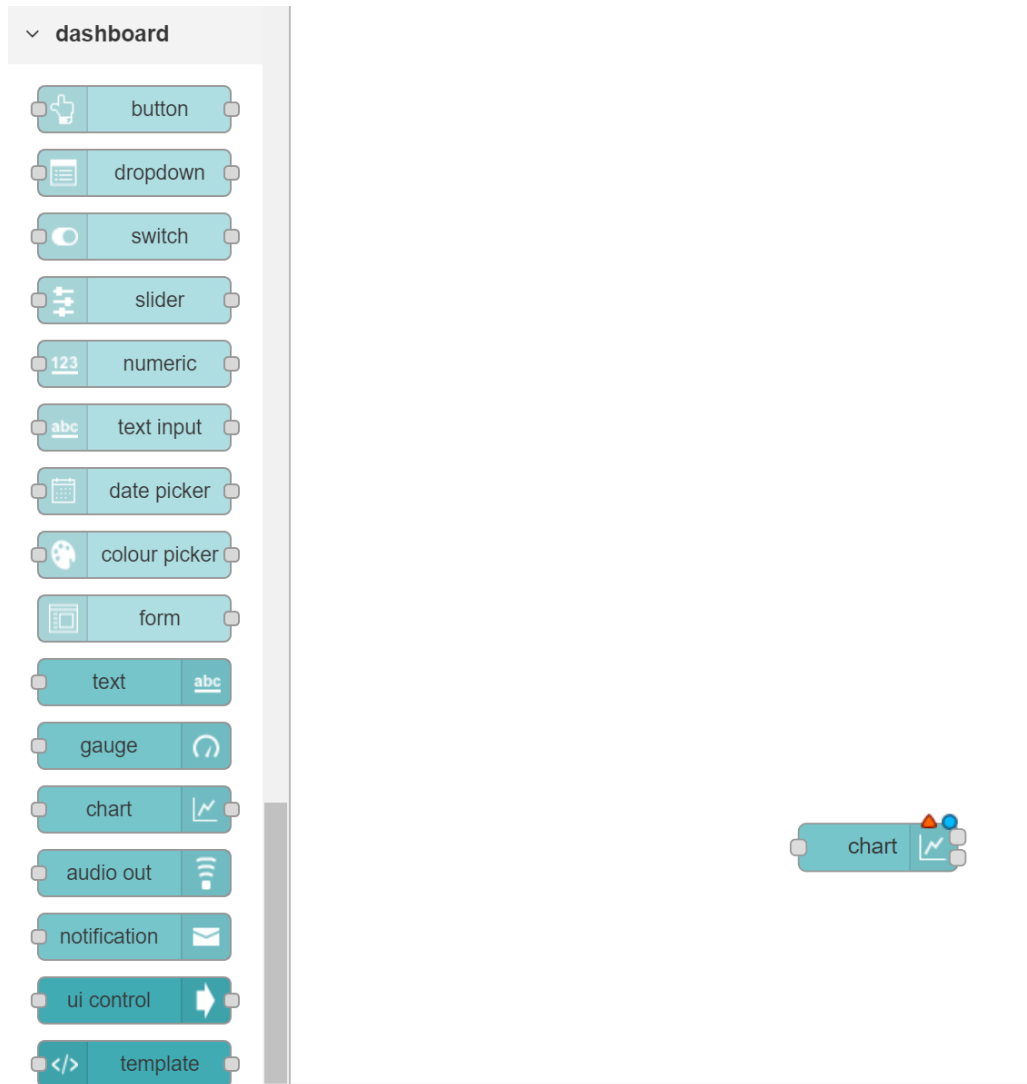
Node-red-dashboard wählen

Install drücken

Dialog mit Install bestätigen

Programmierung Node-Red Daten visualisieren

Chart auf den Flow ziehen



Programmierung Node-Red Daten visualisieren

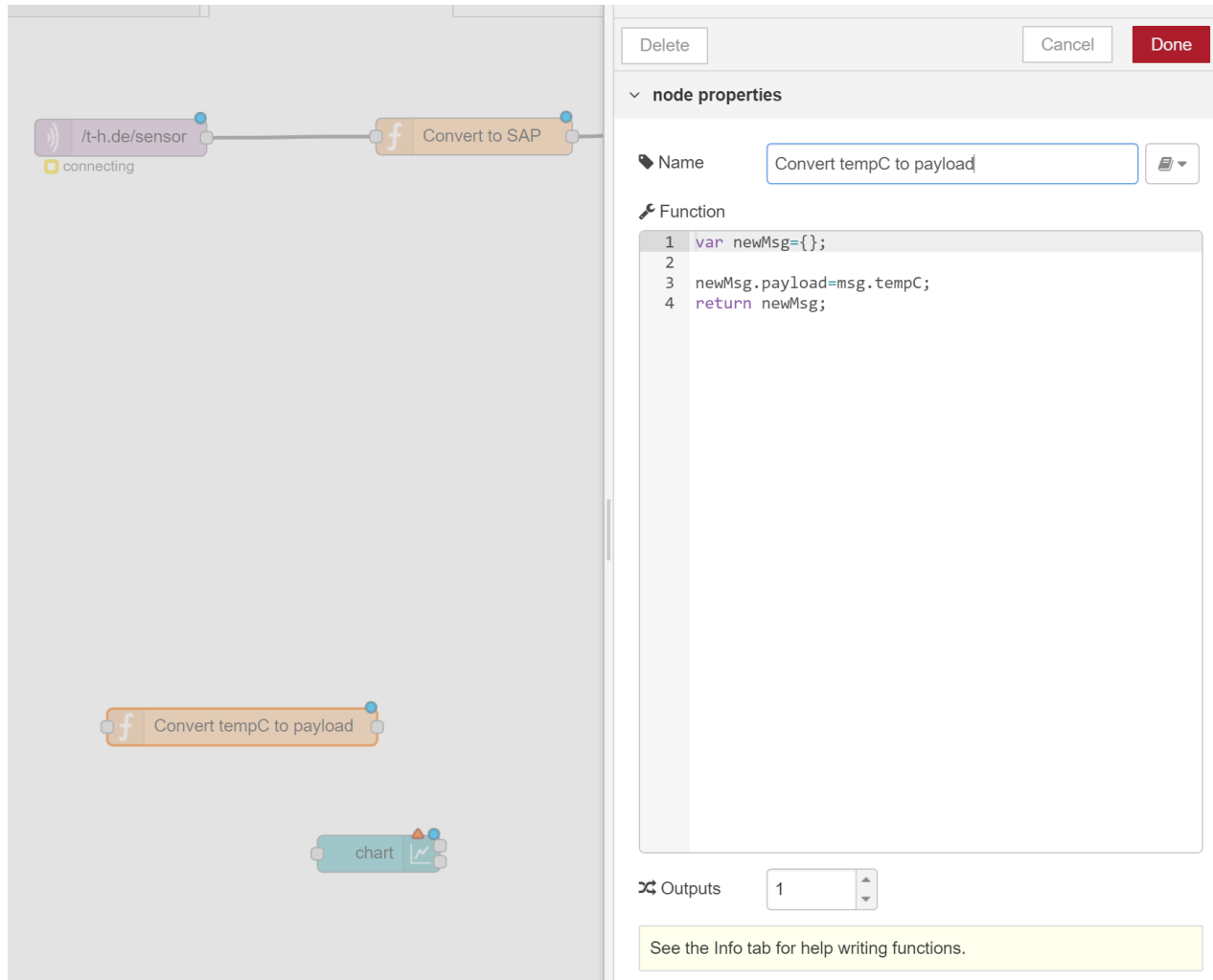
The image shows the Node-Red web interface. On the left is the 'dashboard' sidebar with various widgets: button, dropdown, switch, slider, numeric, text input, date picker, colour picker, form, text, gauge, chart, audio out, notification, ui control, and template. The 'chart' widget is selected. On the right is the 'Edit chart node' configuration panel. It includes a 'Delete' button, 'Cancel', and 'Done' buttons. The 'node properties' section contains: 'Group' (Add new ui_group...), 'Size' (auto), 'Label' (chart), 'Type' (Line chart), 'X-axis' (last 1 hours OR 1000 points), 'X-axis Label' (HH:mm:ss), 'Y-axis' (min/max), 'Legend' (None), 'Interpolate' (linear), 'Series Colours' (a grid of color swatches), 'Blank label' (display this text before valid data arrives), 'Use deprecated (pre 2.5.0) data format.' (checkbox), and 'Name'.

Chart editieren

Neue UI Gruppe „Sensor Data“ anlegen

Label für das Chart e.g. Temperature, Humidity, Pressure vergeben

Programmierung Node-Red Daten visualisieren



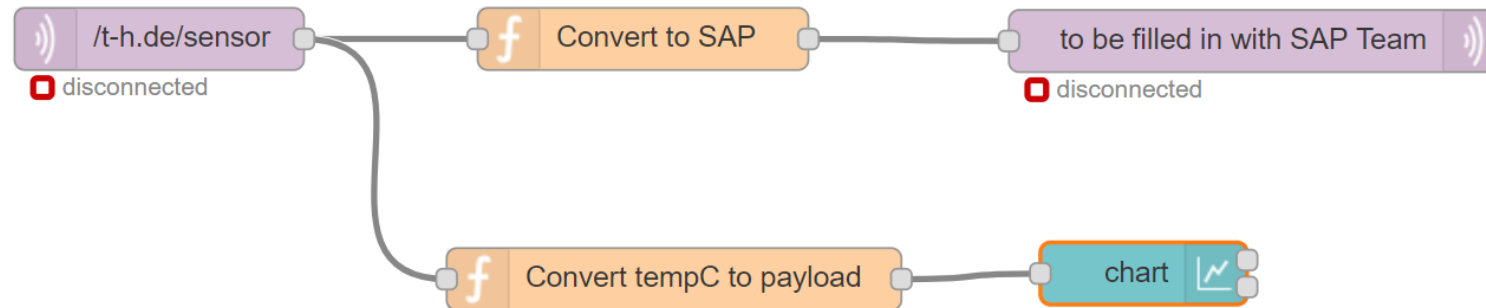
Convert function template
hinzufügen

```
var newMsg={};  
newMsg.payload=msg.temp  
C;  
return newMsg;
```

Programmierung Node-Red Daten visualisieren

Verdrahtung für den unteren Pfad aufnehmen

Für Humidity und Pressure die Convert und Chart Schritte wiederholen



Gruppe SAP Cloud



TO

DO

Vielen Dank

Niclas und Christoph

