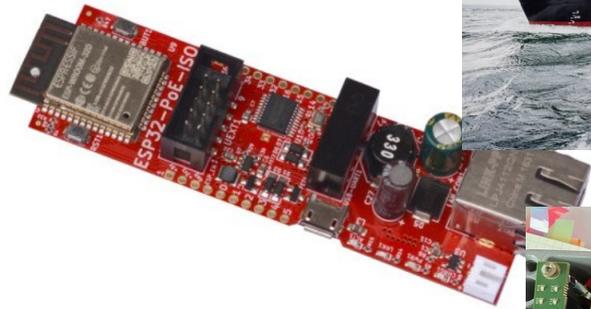


# IOT-Dezentrale Sensoren und Aktoren zentral verwaltet

SEI-Tagung 2023



Christian Jacobsen  
Jörn Plewka

Technikum-Elektronik

# Vortragsgliederung

## **Rückblick Boje, Zielsetzung**

Anderes lokales Konzept, aber u.a. initialer Ideengeber

## **Mikrocontroller, Geräte und Erweiterungen**

Was tun? Kaufen und nutzen, Firmware ersetzen, eigene Geräte bauen?

## **Firmware-Frage**

FreeRTOS, Herstellerfirmware, Tasmota, esphome, was ist zielführend?

## **Automatisierung**

Herausforderung: Zusammenbringen von Geräten und Protokollen, Updates

## **Darstellung und Auswertung**

Mehrere Wege, unser Weg: Influx, Grafana und Co.

## **Standortvernetzung und Sicherheit**

Funkmodems, Router, lokale Dienste, Datenaustausch, Embedded Linux...

## **Fazit und Ausblick**

# Rückblick - Zielsetzung Boje

**Beim Update der Boje ging es seinerzeit darum, ein autonomes Messsystem zu betreiben:**

- Sensoren und Aktoren mit verschiedenen Schnittstellen usw. am Messort erreichbar machen (eigentliche Mess-Sensorik, aber auch Housekeeping)
- Steuerung und Datenübertragung lokal ermöglichen
- Fernzugang ermöglichen
- Daten incl. Telemetrie auf zwei Wegen zum Zentrum übertragen
  1. Livedaten direkt versenden
  2. Messdaten über Dateien
- Gesicherte Infrastruktur für nicht vertrauenswürdige externe Geräte schaffen
- Darstellung und Auswertung anbieten (min. für Quicklook)



# Zielsetzung: Einfaches Housekeeping

Die Boje brachte u.a. eine wesentliche Einschränkung mit, die bei den jetzt betrachteten Systemen eine viel kleinere Rolle spielt: Energie

- Ziel: Einfache dezentrale Sensoren schnell verfügbar machen
- Außerhalb des Zentrums **weiterhin** Messstellen, die die Informationen von verschiedensten Geräten zusammentragen und als Einheit aus dem Zentrum sichtbar sind
- Es fiel auf, dass alles, was ab IP-Ebene schon funktionierte, bereits gut geeignet schien.  
→ tendenziell leichtgewichtiger, aber mehr Teilnehmer

**Mehr Geräte, bis hin zu mehreren PCs, die schon IP (z.B. MQTT) sprechen**

# MQTT als „Nachrichtendienst“

- „Sensor“ der direkt IP (MQTT) spricht
- Gesucht: Mikrocontroller der das spricht & viel mit seiner Hardware bietet

The screenshot shows the MQTT Explorer application. On the left, a tree view lists topics under the IP 192.168.5.12, including 'solar', 'hzig-oben', 'homeassistant', 'sensor', 'switch', 'climate', 'button', 'number', 'binary\_sensor', 'hzig-unten', 'tele', 'tasmota', '\$SYS', and 'sdm630'. The 'sdm630' topic is expanded to show 'Power'. The main pane displays a message history for 'sdm630/Power' with a timestamp of 2023-04-14T22:20:21.782. The message content includes fields for Time, R, S, T, and SUM. Below the history is a line graph titled 'SUM sdm630/Power' showing a fluctuating signal over time.

This screenshot shows a detailed view of a message in MQTT Explorer. The left pane shows the topic tree expanded to 'hzig-unten/temperatur\_wohnzimmer', with the message configuration highlighted: `config = {"dev_cla": "temperature", "unit_of_meas": "°C"}`. The main pane shows the message details, including the topic 'hzig-unten/temperatur\_wohnzimmer', a 'RETAINED' status, and a timestamp of 14.04.2023 22:16:45. The message payload is a JSON object: 

```
{  "dev_cla": "temperature",  "unit_of_meas": "°C",  "stat_cla": "measurement",  "stat_t": "hzig-unten/sensor/temperatur_wohnzimmer/state",  "avty_t": "hzig-unten/status",  "uniq_id": "dallas-5d0119536fcaff28",  "dev": {    "ids": "240ac48011e8",    "name": "hzig-unten",    "sw": "esphome v2023.2.4 Mar 7 2023, 20:06:43",    "mdl": "esp32dev"  }}
```

 The right pane shows a 'Publish' button and a 'Stats' section.

# Geeigneter IOT-Mikrocontroller

- Mikrocontroller ESP32 ist die unterste Basis für das, was wir nun vortragen
- Platzhirsch, wenn WLAN, LAN und Bluetooth ins Spiel kommen.
- Offene Software und Arduino-Framework, umfangreiche Peripherie und Ressourcen bei geringem Preis usw. haben schnell hohe Verbreitung erzeugt.
- ESP32 wurde von Espressif Systems, einem chinesischen Unternehmen mit Sitz in Shanghai, entwickelt basiert auf einen Cadence-Core, wird von TSMC im 40-nm-Verfahren hergestellt und ist der Nachfolger des ESP8266-Mikrocontrollers

# „Inkarnationen“ des ESP32 und ESP8266

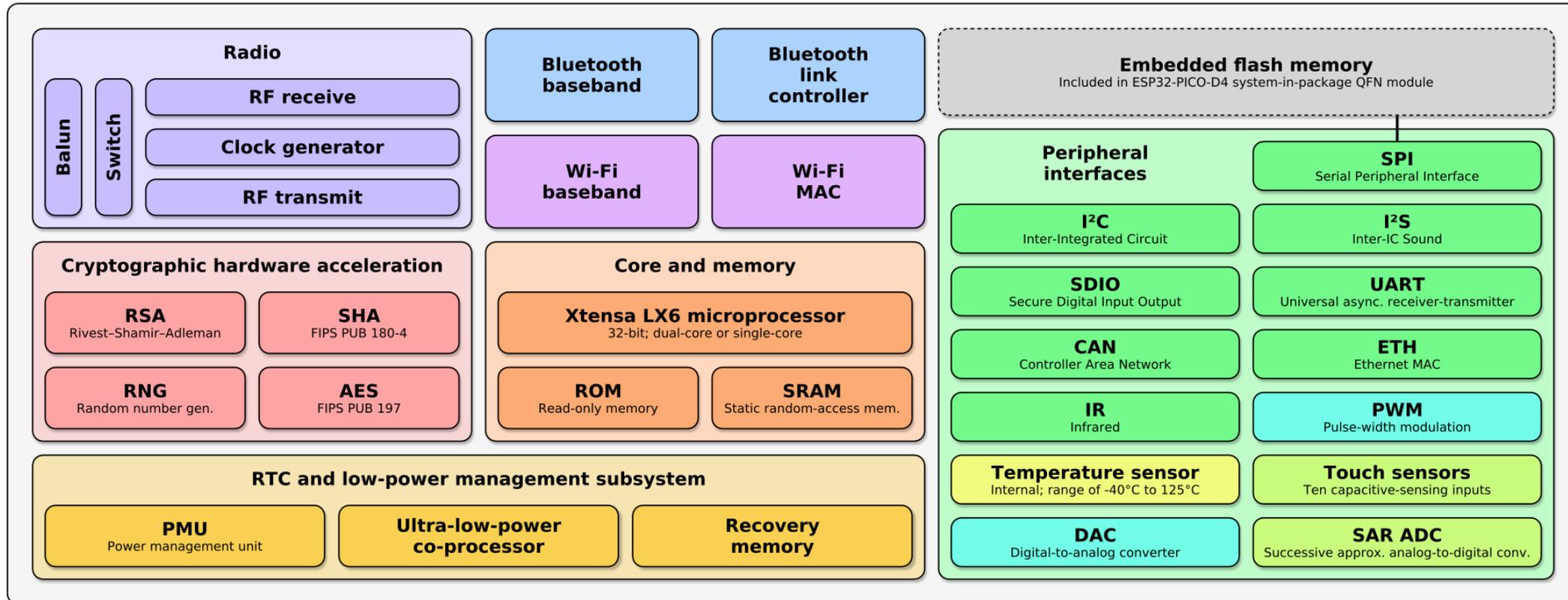


 <p>Shelly Plus 1</p> <p>14,90 € (exkl. MwSt.)</p> <p>Jetzt Kaufen</p>	 <p>Shelly Plus 1PM</p> <p>16,90 € (exkl. MwSt.)</p> <p>Jetzt Kaufen</p>	 <p>Shelly Plus i4</p> <p>11,90 € (exkl. MwSt.)</p> <p>Jetzt Kaufen</p>	 <p>Shelly Plus 2PM</p> <p>25,90 € (exkl. MwSt.)</p> <p>Jetzt Kaufen</p>	 <p>Shelly Pro 2PM</p> <p>82,50 € (exkl. MwSt.)</p> <p>Jetzt Kaufen</p>	 <p>Shelly Pro 4PM</p> <p>97,90 € (exkl. MwSt.)</p> <p>Jetzt Kaufen</p>
---	---	--	---	--	--

# IOT-Mikrocontroller ESP32

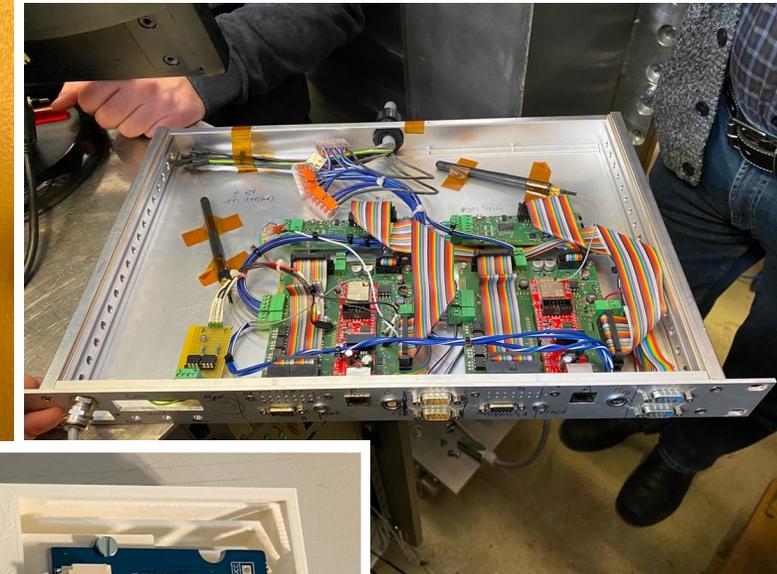
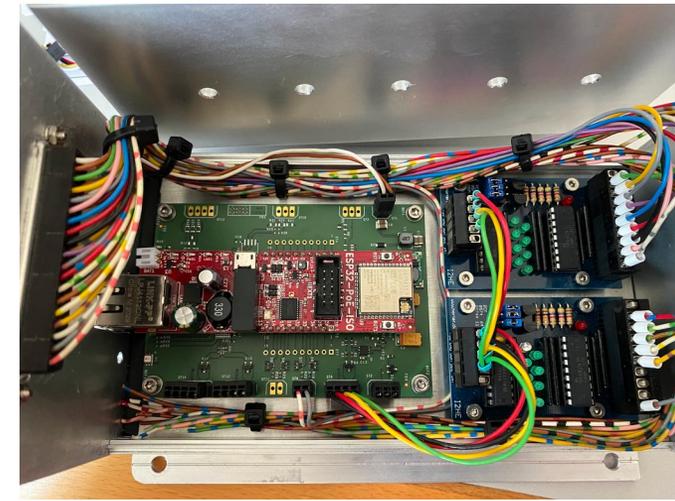
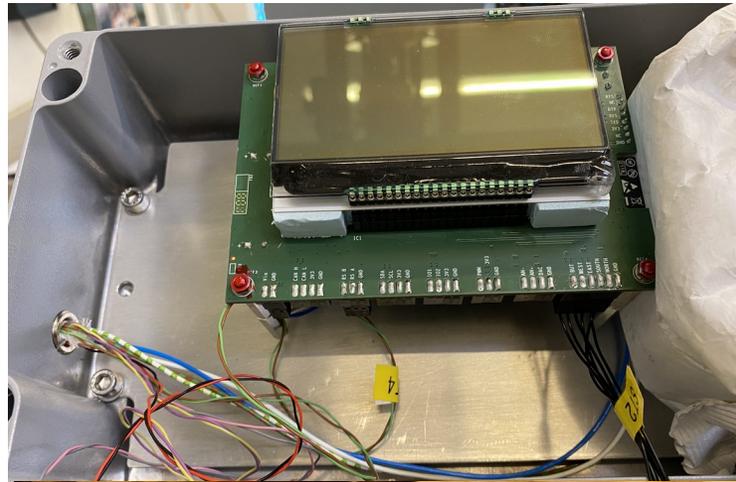
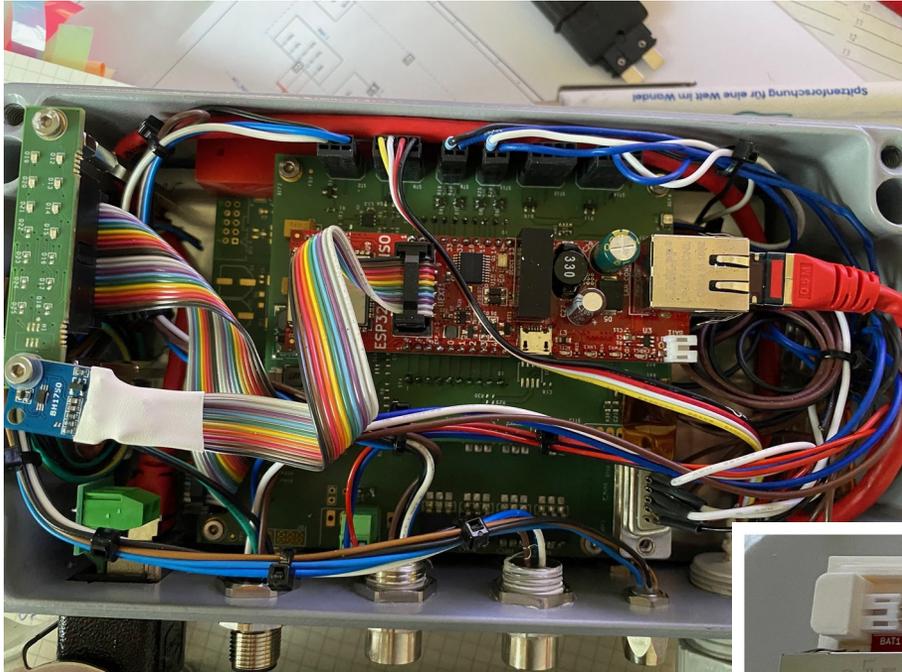
- Datenblatt zeigt noch mehr Fähigkeiten als Block-Diagramm
- „Feuerwerk“ an Funktionalität – ABER: u.a. recht wenige Pins

Epressif ESP32 Wi-Fi & Bluetooth Microcontroller – Function Block Diagram



# Eigene Hardware

EvalBoard bzw.  
direkt Prozessormodul  
(teils Prototypen)

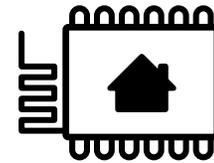
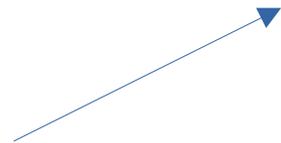


# Firmware

## „Raus aus der Cloud“:

- Erzwungene Hersteller-Cloud (für uns) inakzeptabel
- Alternativen:
  - Direkt - FreeRTOS IDF, Arduino
  - Erweitert - Tasmota, espHome, (MongooseOS)
- Funktionalität von Tasmota und ESPHome sehr umfangreich, ausgereift, quelloffen  
Eigene Lösung viel Arbeit → es muss gute Gründe geben.

  
**TASMOTA**



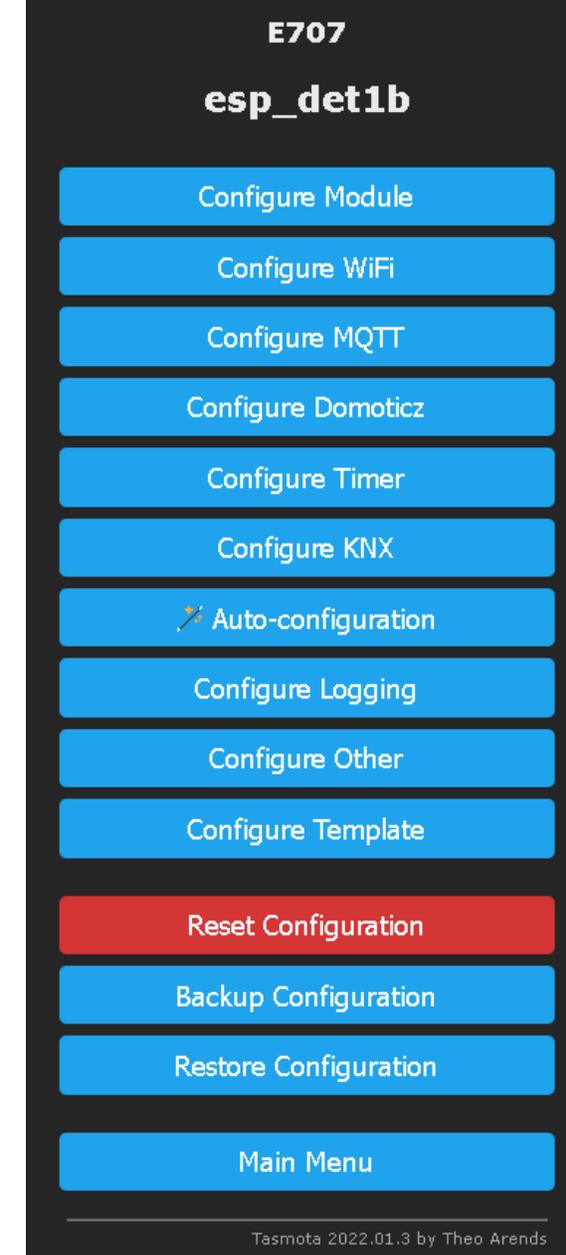
**ESPHome**



# Firmware Tasmota

- Open Source Firmware für Espressif ESP8266, ESP32, ESP32-S oder ESP32-C3 Chipsatz basierte Geräte
- Ursprüngliches Ziel war es, ESP8266-basierte ITEAD-Sonoff-Geräte mit MQTT und "Over the Air"- oder OTA-Firmware auszustatten
- Abwandlung eines cloudbundenen Sonoff Basic (eines der ersten günstigen und zugänglichen Smart-Home-Geräte auf dem Markt) in ein lokal gesteuertes Gerät, hat sich zu einem vollwertigen Ökosystem für praktisch jedes ESP8266-basierte Gerät entwickelt.
- Später kamen die neueren Prozessoren hinzu, ESP8266 hat starke Beschränkungen, weshalb der Umgang mit Speicher vor neuen Funktionen stand/steht.
- Idee ist eine einzige Firmware, die alles kann.

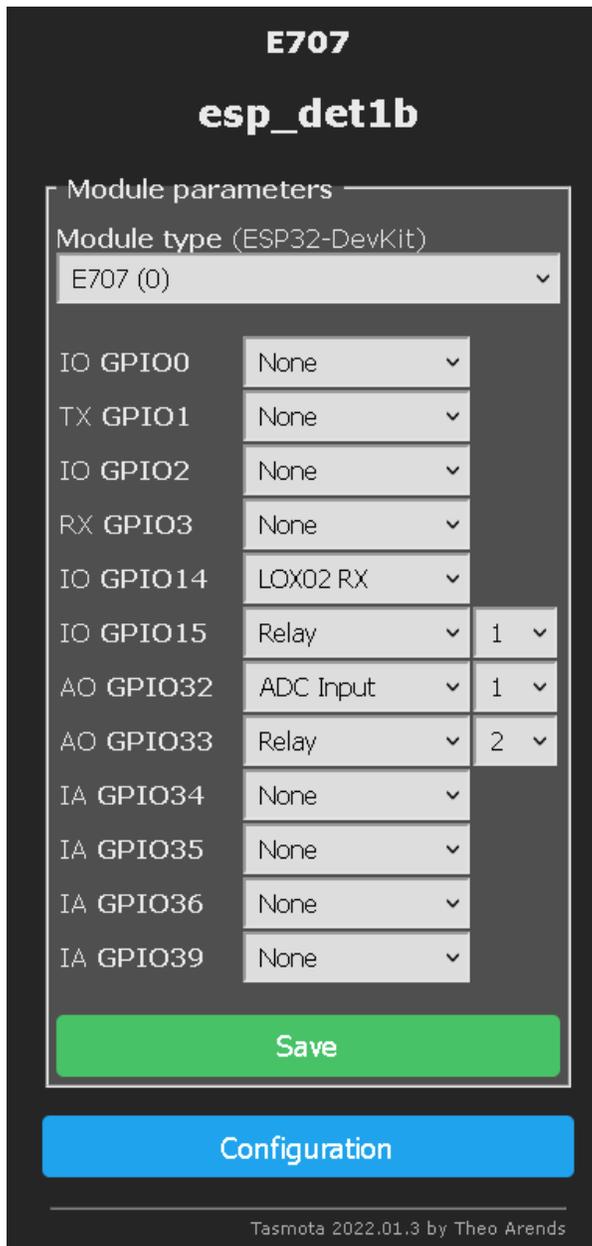
→ **Bild: Geräte öffnen initial Accesspoint für Einstellungen - später auch per Webseite sowie MQTT, Konsole usw. möglich**



# Tasmota Anpassung

- Inzwischen als Platzgründen verschiedene Firmwareimages, mit verschiedenen funktionalen Gewichtungen
- Verschiedene Wege, über Webdienste bis hinunter zu #defines im Quellcode festlegen, welche Funktionalitäten in das individuelle Image eingebaut werden
- Funktionsumfang durch Vielzahl von Sensoren/Aktoren, Protokollen usw. schier "erdrückend"
- Arduino-Abstammung für Quellcode sehr hilfreich
- Möglichkeit, Änderungen selber einzubauen und in das Projekt aufnehmen zu lassen - Codestruktur macht es aber recht schwer Änderungen zu pflegen

← **Bild: Zuordnung von HW-Funktionen auf Basis von IO-Pins**

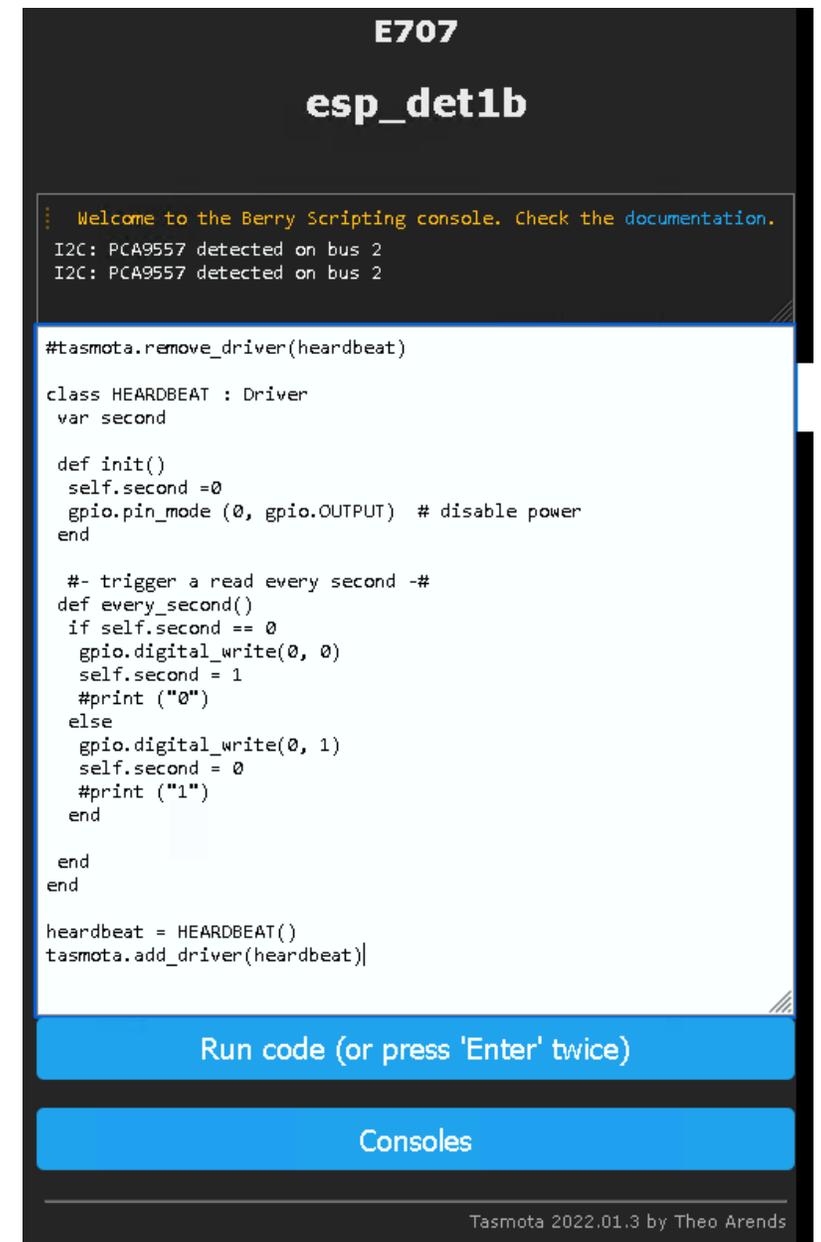


# Firmware Tasmota

- Images für ESP32 bieten sogar einen On-the-Fly Interpreter/Compiler (Berry)

→ auf dem IOT-Gerät per Webseite sogar Treiber z.B. für Sensoren mit serieller Schnittstelle oder I2C schreiben, Daten verschicken oder Abläufe steuern

**Bild: Berry lässt IO/LED als Heartbeat blinken →**



The screenshot shows the Berry Scripting console for a Tasmota device (E707, esp\_det1b). The console output displays the following code and its execution results:

```

Welcome to the Berry Scripting console. Check the documentation.
I2C: PCA9557 detected on bus 2
I2C: PCA9557 detected on bus 2

#tasmota.remove_driver(heartbeat)

class HEARDBEAT : Driver
  var second

  def init()
    self.second = 0
    gpio.pin_mode (0, gpio.OUTPUT) # disable power
  end

  #- trigger a read every second -#
  def every_second()
    if self.second == 0
      gpio.digital_write(0, 0)
      self.second = 1
      #print ("0")
    else
      gpio.digital_write(0, 1)
      self.second = 0
      #print ("1")
    end
  end

end

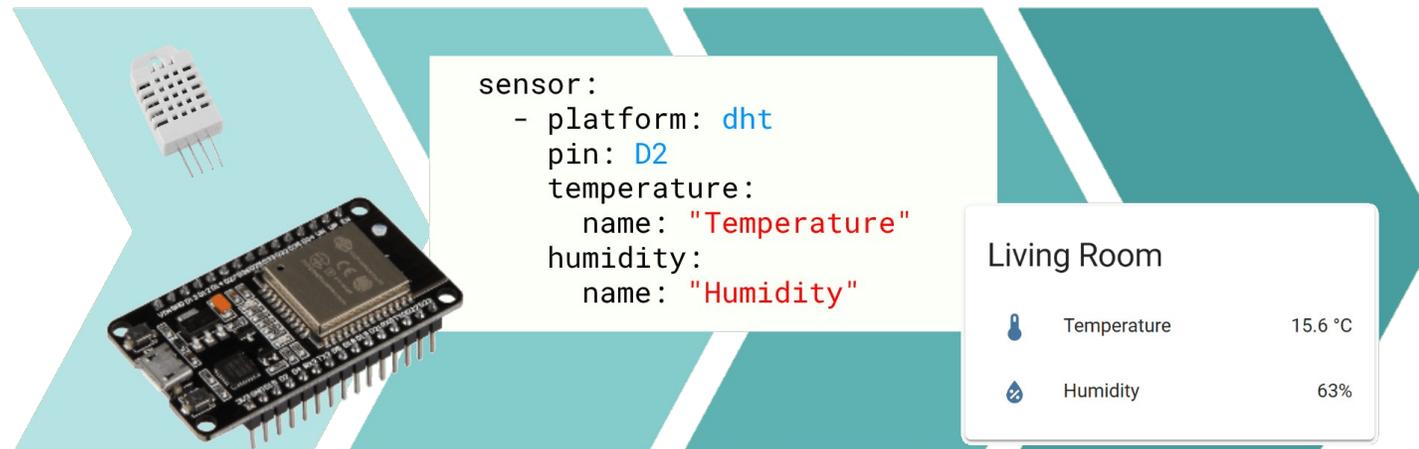
heartbeat = HEARDBEAT()
tasmota.add_driver(heartbeat)

```

At the bottom of the console, there are two buttons: "Run code (or press 'Enter' twice)" and "Consoles". The footer of the console indicates "Tasmota 2022.01.3 by Theo Arends".

# Firmware ESPHome

- Die Beschreibung Homepage von ESPHome beschränkt sich auf einen Satz:  
"ESPHome ist ein System, mit dem Sie Ihren ESP8266/ESP32 durch einfache, aber leistungsfähige Konfigurationsdateien steuern und über Home Automation Systeme fernsteuern können."
- nach Bedarf auch Darstellung einfacher Webseite, aber **nicht zur Konfiguration**
- Ausrichtung auf **unterste Ebene einer größeren Automatisierung**
- Firmware nutzt bei den Konfigurationsdateien den "Look-and-Feel" von Home-Assistant und bringt ebenfalls eine Integration zu HomeAssistant mit



The image illustrates the ESPHome configuration and its user interface. On the left, there is a photograph of a DHT22 temperature and humidity sensor and an ESP8266-based development board. In the center, a code block shows the configuration for a sensor:

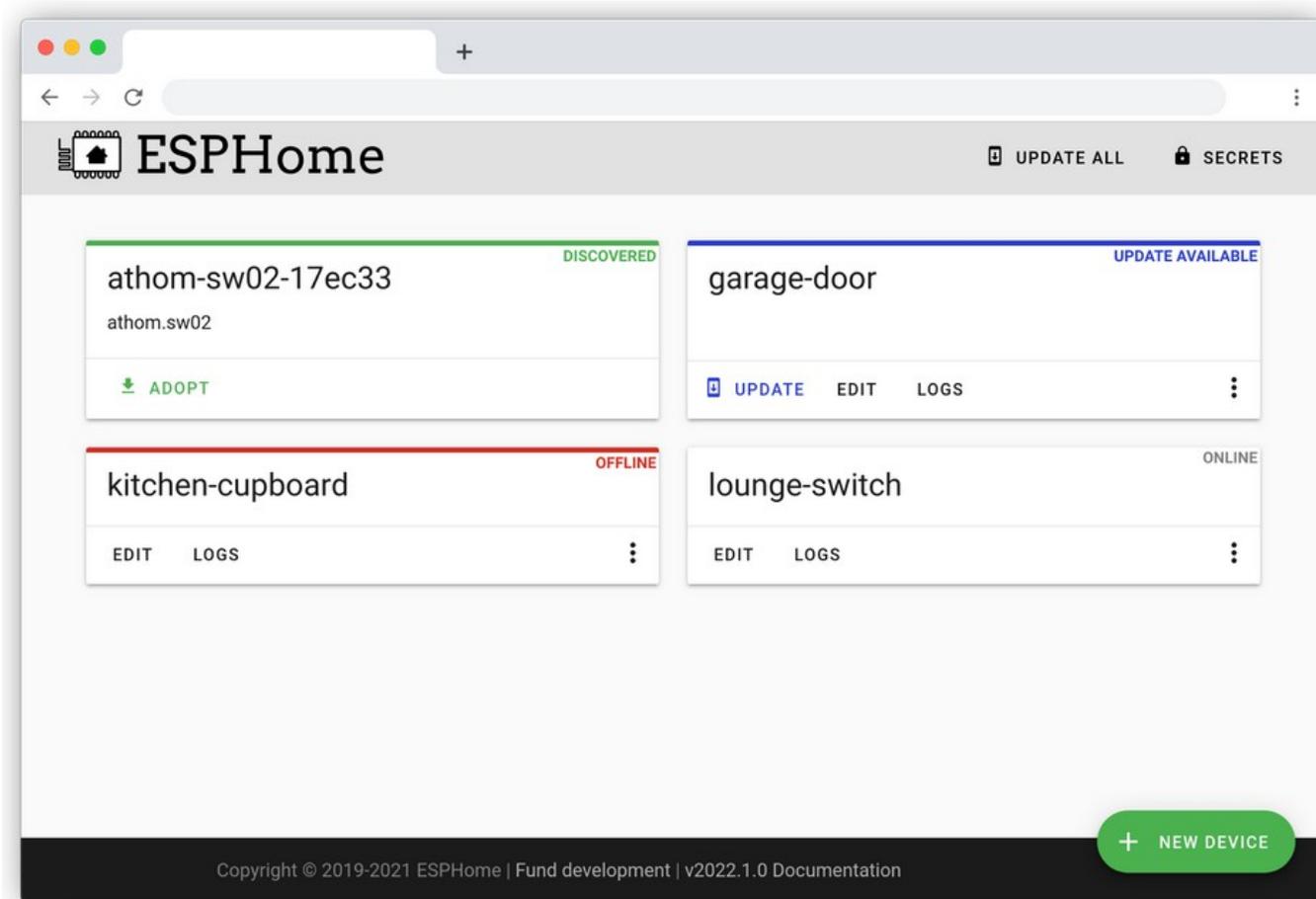
```
sensor:  
- platform: dht  
  pin: D2  
  temperature:  
    name: "Temperature"  
  humidity:  
    name: "Humidity"
```

On the right, a UI card titled "Living Room" displays the sensor data:

Living Room		
🌡️	Temperature	15.6 °C
💧	Humidity	63%

# Verwaltung ESPHome

- keine "Default-Images", die auf mehrere Geräte gespielt werden...incl. der initialen Firmware
- Konfiguration per yaml-Beschreibung - wird erst "komponiert" und dann individuell für jedes Gerät kompiliert und über ein serielles Kabel (direkt am Server, oder mit Serial Redirect per Chrome) oder online aufgespielt
- ESPHome-Serverdienst baut tatsächlich für **jeden Client einzeln** die Firmware!



→ **Bild: Verwaltungsdienst mit ganz anderer Ausrichtung als bei TasmAdmin u.a. mit Key-Management**

# Hardware/Firmware Fazit

- Geräte mit Standardfunktionalität nutzen:  
Tasmota wohl gut geeignet, auch das Closed-Source-System von Shelly (MoongooseOS) ist ggf. direkt geeignet und benötigt nicht zwingend Shellys Cloud-Verbindung
- Eigene oder Geräte anpassen und miteinander verbinden:  
Tasmota schwergewichtiger, ESPHome leichtgewichtiger und eleganter
- Wer die Prinzipien moderner Softwareentwicklung schätzt, wird eher mehr Freude durch Eleganz mit ESPHome erfahren.
- Sobald das Gerät MQTT spricht, ist der Weg für alles frei!



# Übergeordnete Automatisierung

- Was erwartet man von so einem System für unser Housekeeping?

Vergleich von HomeAssistant, IoBroker, openHAB, FHEM, openWallbox usw. → Internet



HomeAssistant: 2019 das zehntgrößte Opensource-Projekt mit >63000 Mitwirkenden, hohe Veränderungsrate, stabiles Fundament

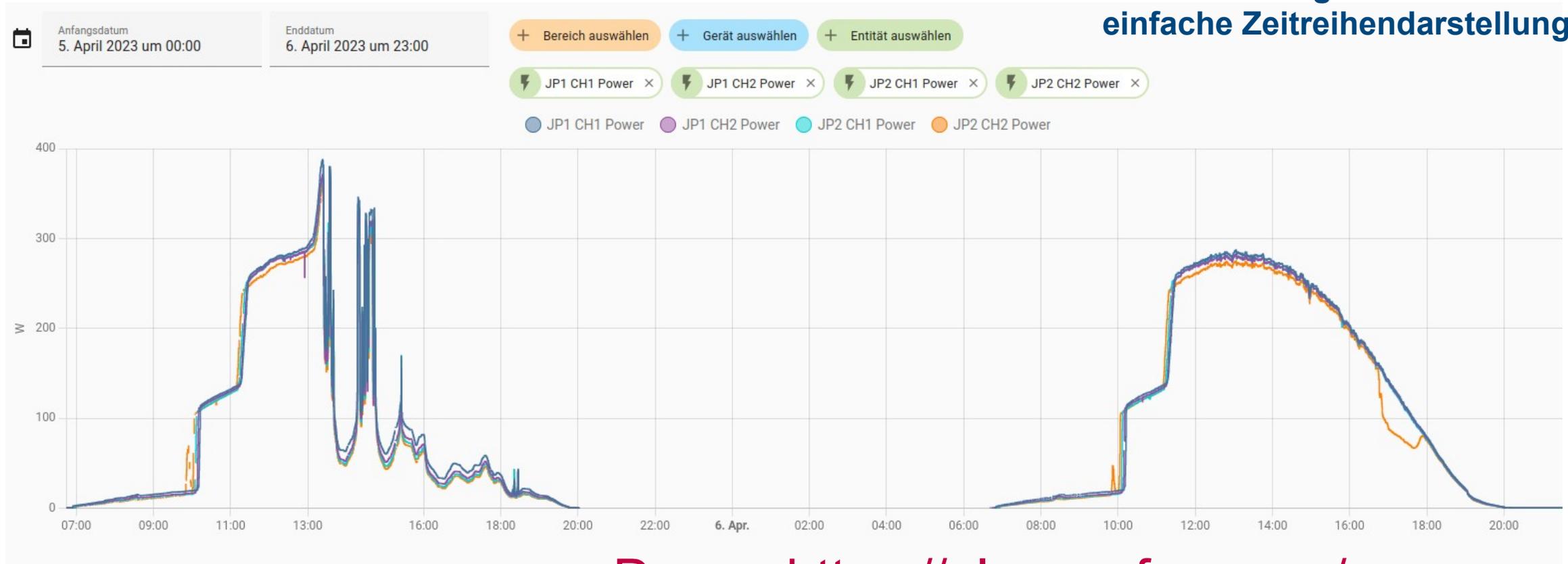
- Technologisch, vom Anwender verborgen:  
Virtual Appliance aus Docker-Containern, eigenes Image für RaspberryPi
- Fokus zwar nicht auf Industrieanwendungen, aber:
  - vorherrschende Programmiersprache ist Python
  - Integration völlig verschiedener Systeme mit unterschiedlichen Bussen, Protokollen

Demo: <https://demo.home-assistant.io>

# HA und Datendarstellung

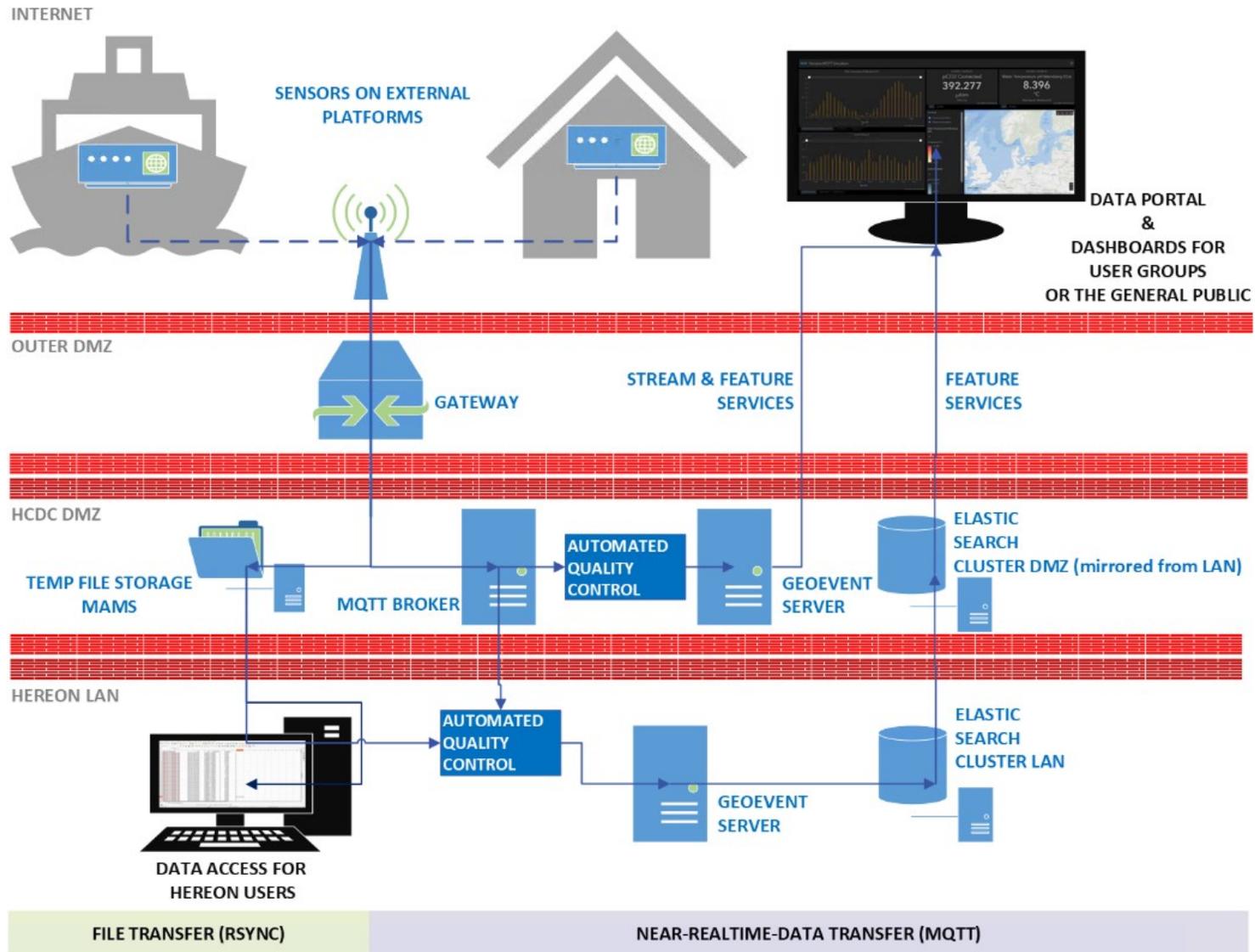
HomeAssistant als Blackbox gedacht:  
Basisdienst kann viele Subdienste integrieren, auch Grafana zur Darstellung.

**Bild im Hintergrund:  
einfache Zeitreihendarstellung in HA**



Demo: <https://play.grafana.org/>

# Scientific Sensor Data Flow (HCDC-Team)



Das Bild zeigt die Datenflüsse von wissenschaftlichen Daten ins Hereon.

Die Sensoren in den Außenstellen werden durch „HELMI“ angebunden (→ nächste Folie).

Der wissenschaftliche Fokus liegt auf Geo-Position, unserer eher auf Zeitserien.

Wir bieten eher Monitoring von Housekeeping an und verzweigen beim MQTT-Cluster z.B. auf Grafana

# „Helmi“ - Zugangspunkt in den Außenstellen

Zugangspunkte mit zwei Varianten, einmal als Plug-and-Play Hardware mit Embedded Linux und einmal als Software – mit dem Standard-Router.

→ Ganz frisch: auch ESPHome spricht WireGuard

## HELMI



[https://www.phytec.de/fileadmin/phytec\\_base/pdfs/flyer/phyGATE-Tauri.pdf](https://www.phytec.de/fileadmin/phytec_base/pdfs/flyer/phyGATE-Tauri.pdf),  
accessed 2021/06/17



## HELMI light



© Hereon/Joost Hemmen

# Danke für die Aufmerksamkeit und fürs Fragen stellen!

## Ausblick

Weiter schnell wachsende Nachfrage durch kurze Bereitstellungszeit u.a. für Energiemessung...

