

W3C Web of Things Retail Demo Scenario

Michael McCool

3 May 2022

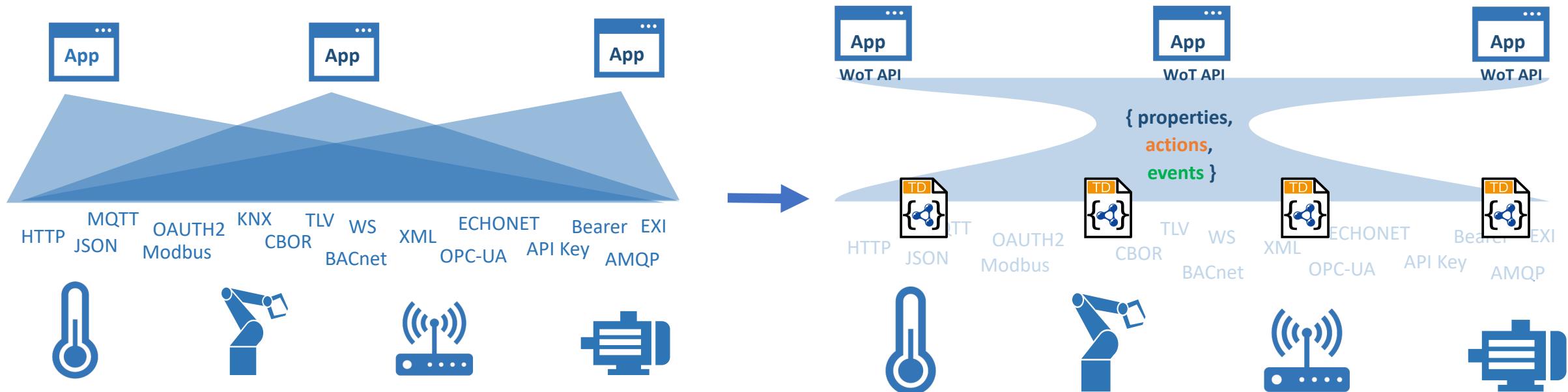
Conexxus Conference

Outline

- What is WoT?
 - Applying and extending web standards for IoT
 - Descriptive interoperability
- Test Scenario: Cooler Monitor
 - Hub architecture
 - Desired behavior and devices
- Implementations
 - Options
 - Home Assistant and Node-RED Prototypes
- Discussion
 - Semantic interoperability

W3C Web of Things (WoT)

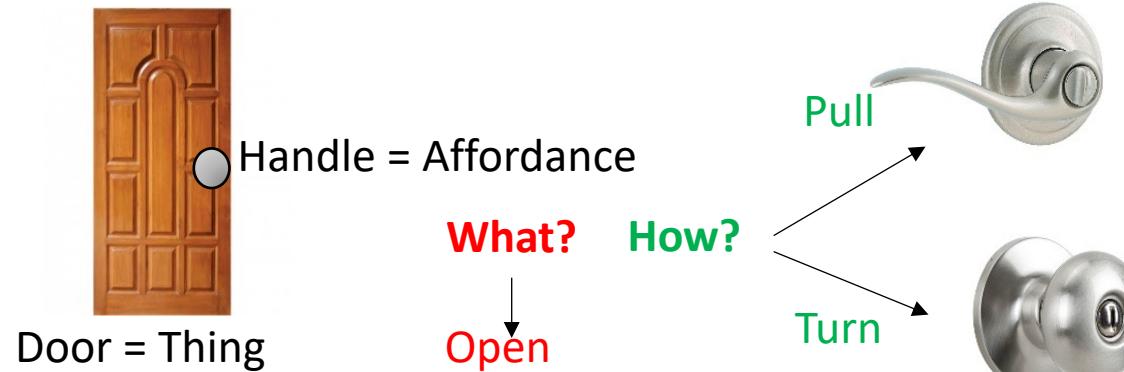
- W3C Working Group goal: Adapting web technologies to IoT
- Already published: Thing Description (TD) metadata format
 - TD describes the available interactions (network API) of a Thing
- New deliverables in progress, including Discovery
 - How does a potential user obtain the TD for a Thing?



Descriptive Interoperability: TDs

WoT Architecture

- Constraints
 - "Things" must have a TD
 - Must use URIs, IANA media types, etc.
- Thing Description Affordances
 - Describes **WHAT** the possible choices are
 - Describes **HOW** to interact with the Thing



WoT Thing Description (TD)

```
{
  "@context": [
    "https://www.w3.org/2022/wot/td/v1.1",
    { "iot": "http://iotschema.org/" }
  ],
  "id": "urn:dev:org:32473:1234567890",
  "title": "MyLEDThing",
  "description": "RGB LED torchiere",
  "@type": [ "Thing", "iot:Light" ],
  "securityDefinitions": {
    "default": { "scheme": "bearer" }
  },
  "security": [ "default" ],
  "properties": {
    "brightness": {
      "@type": [ "iot:Brightness" ],
      "type": "integer",
      "minimum": 0,
      "maximum": 100,
      "forms": [ ... ]
    }
  },
  "actions": {
    "fadeIn": {
      ...
    }
  }
}
```

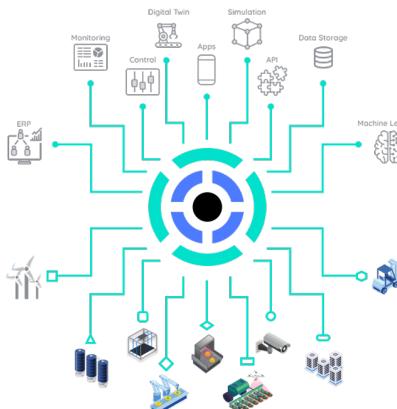
Applications



<https://www.takenaka.co.jp/news/2021/05/02/>

Takenaka Corporation

- CGLL Platform - BIM



<https://netzo.io/>

Netzo

- IoT Data Hub
- Dashboards

2022-05-04

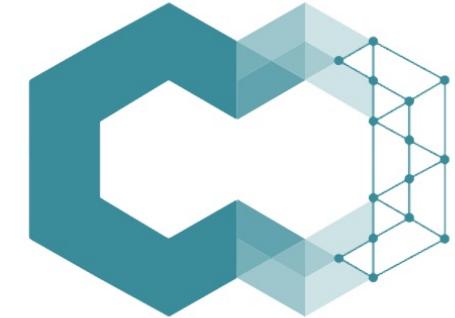


<https://new.siemens.com/global/en/products/buildings/automation/designo.html>

<https://www.evosoft.com/en/digitalization-offering/saywot/>

Siemens

- Desigo CC – BIM
- Say WoT!



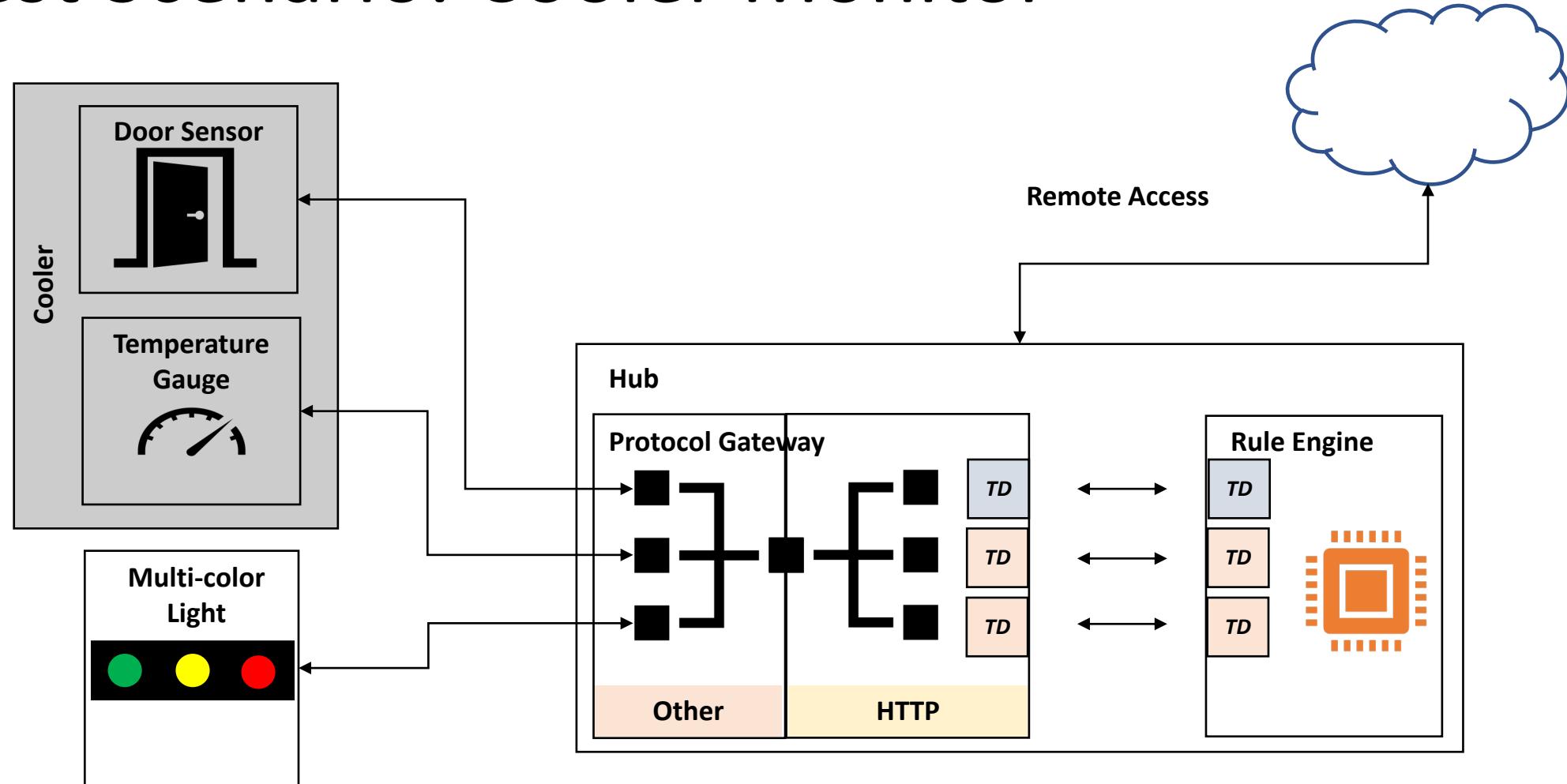
ditto

<https://www.eclipse.org/ditto/2022-03-03-wot-integration.html>

Bosch

- Eclipse Ditto - Digital twin

Test Scenario: Cooler Monitor



Why a Hub?

- Protocol translation
- Data translation
- Other services
 - Historical data
 - Dashboards
- Rules engine (automation)
- Shadowing
 - Caching state of sleeping devices
 - Queuing commands to sleeping devices
 - Important for battery-powered devices
- Secure Remote Access

Desired Behavior

Door Open Alert

- If the cooler door has been left open for more than a certain time: **flash light**.
- If the door is closed: **cancel flashing**

Temperature Out of Range

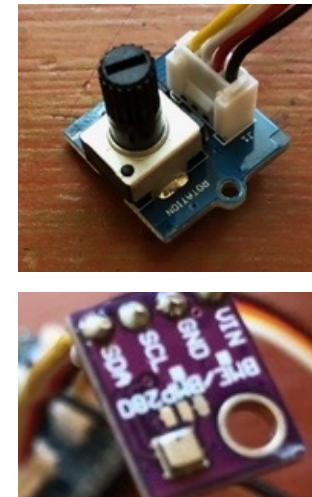
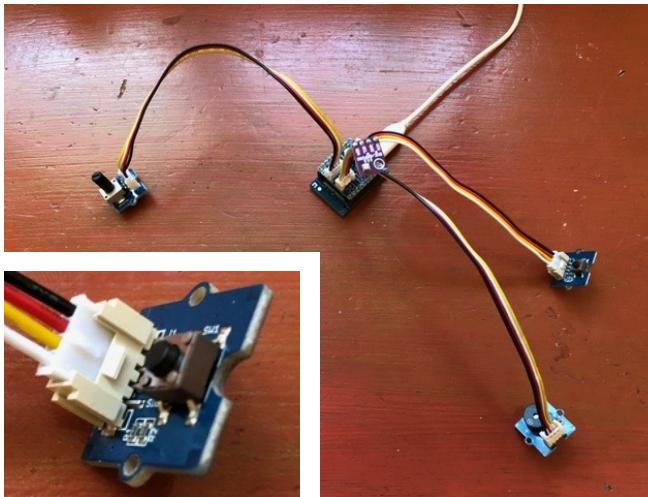
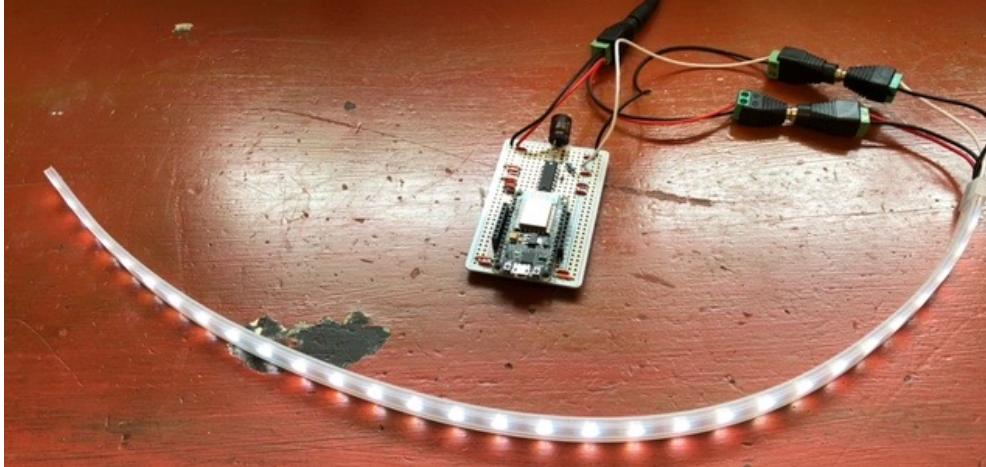
- If internal freezer temp is too low: **turn light BLUE**
- If internal freezer temp is too high: **turn light RED**
- If internal freezer temp is normal: **turn light WHITE**

NOTE:

- Rules should co-exist.
→ If door is open *and* temps are high, light should be **RED** *and* flashing

Devices

Test Devices



2022-05-04

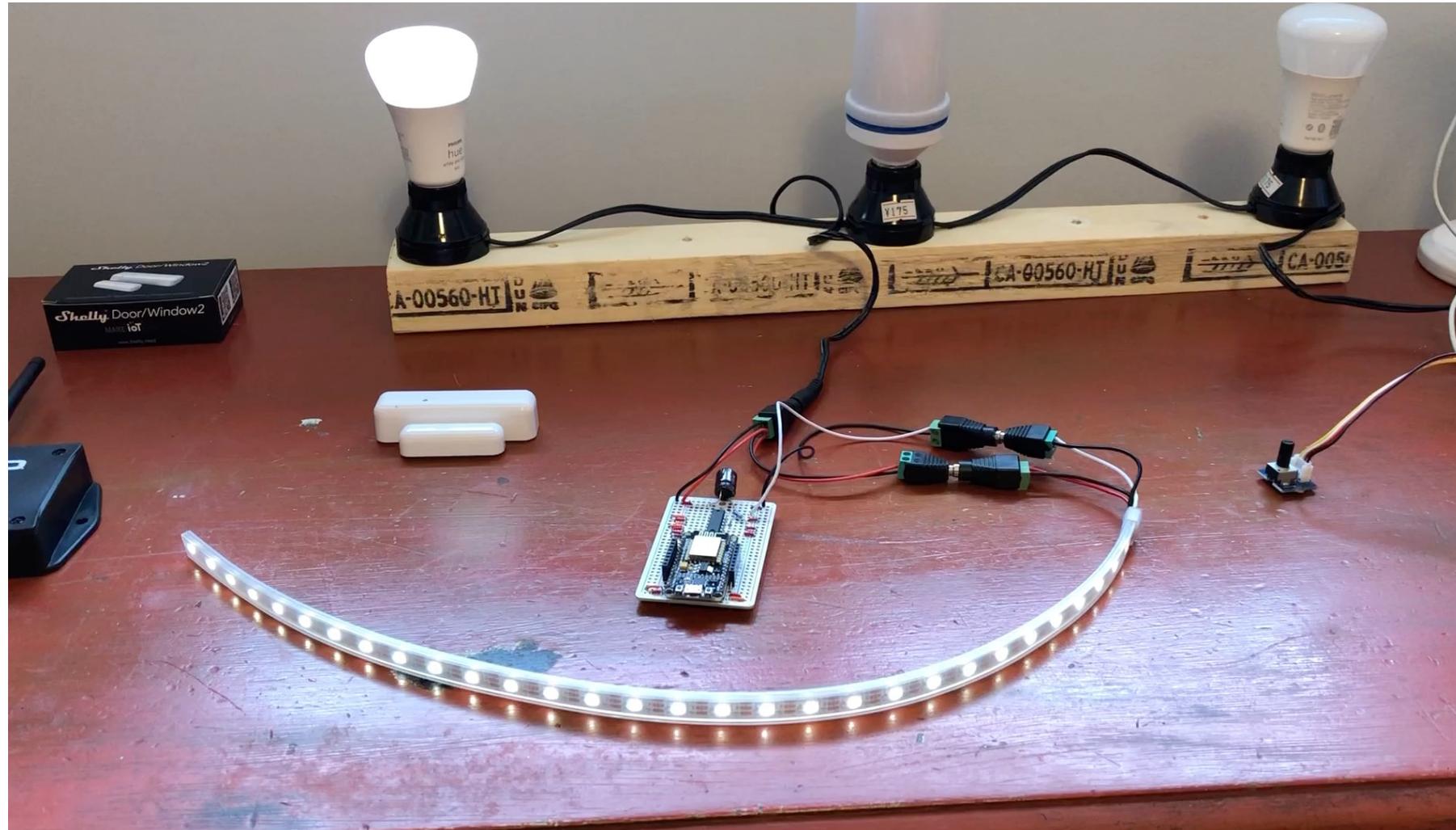
Commercial Devices



W3C Web of Things (WoT) WG/IG

12

Video: System Demo



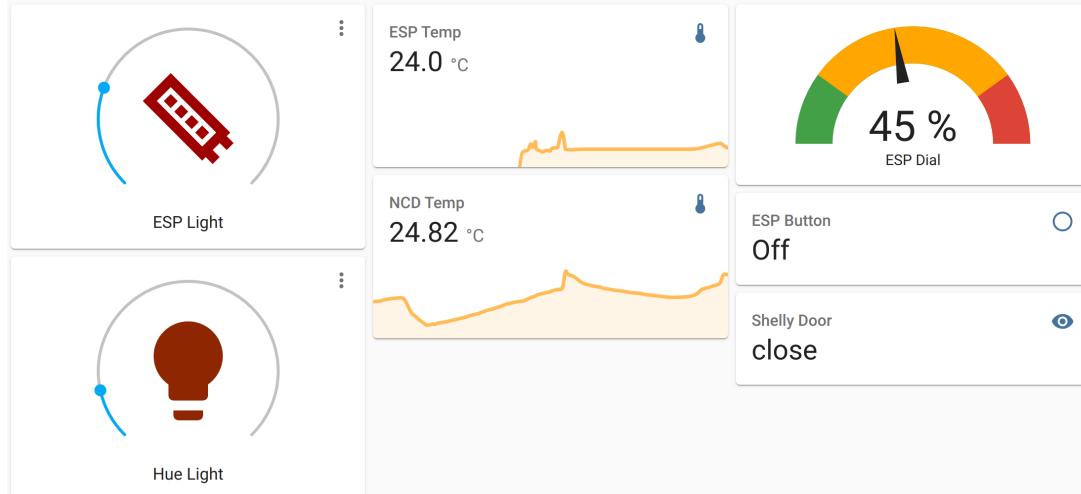
Implementation Approaches

- Proprietary Turnkey System
 - Contract out entire system to a third party
 - Issues: Maintenance, modifications, ownership transfer, business risk
- Proprietary Gateway/Rules Engine (e.g. SmartThings)
 - Integrations limited to those provided by vendor, business risk, portability
- Open-source Gateway/Rules Engine (e.g. Home Assistant, Node-RED)
 - Crowdsourced integrations, variable quality, business risk, portability
- Standards-based Gateway/Rules Engine (e.g. node-wot)
 - Subclass of Proprietary and Open-Sourced Gateways
 - Addresses portability concern, which also helps to address business risk

Two Implementations

Home Assistant Native Rules

- Use built-in rules engine



The screenshot shows the Home Assistant interface with two cards:

- ESP Light:** Displays a red ESP8266 module icon, current temperature (24.0 °C), a line graph of temperature over time, and a dial showing 45%.
- Hue Light:** Displays a red lightbulb icon, current temperature (24.82 °C), a line graph of temperature over time, and a status message "Shelly Door close".

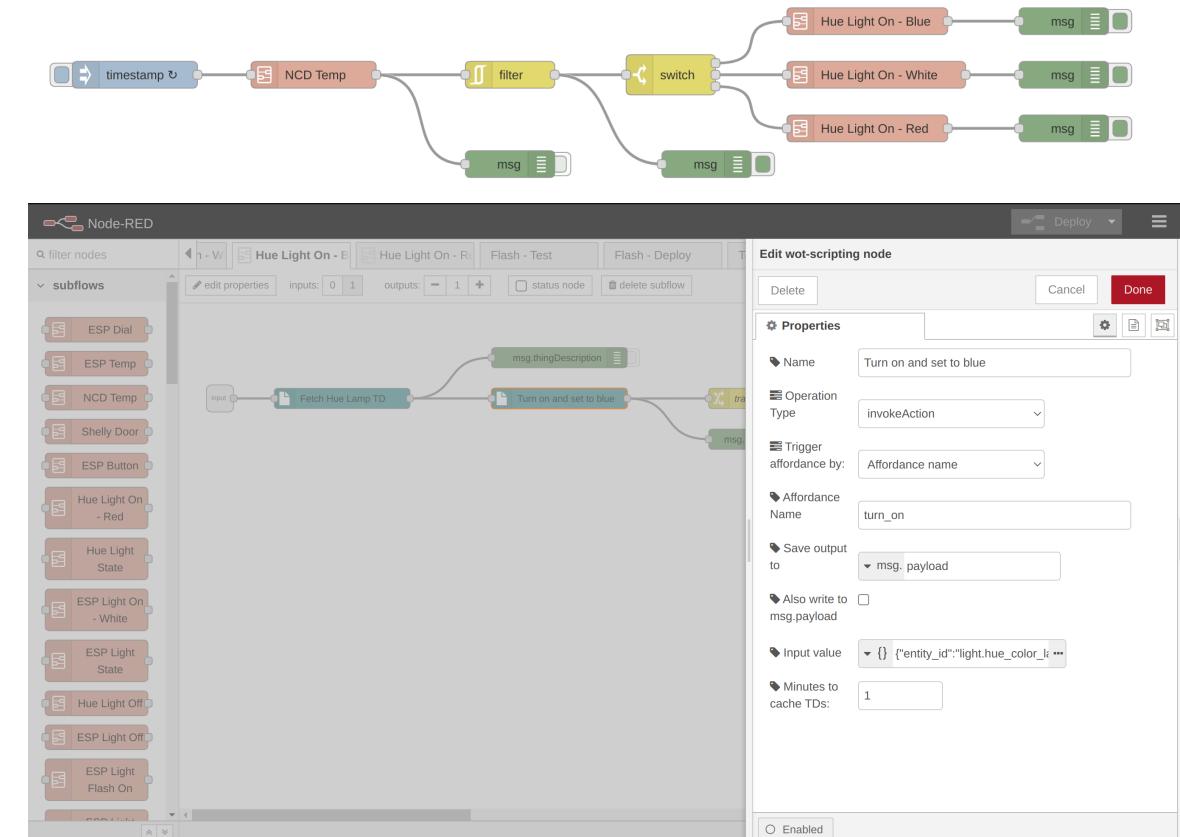
Below the cards is a list of five native rules:

- Demo: Flash Light when Door Open or Button Pressed (April 30, 2022, 23:30)
- Demo: Make Light Blue when Temp/Dial is too low (April 30, 2022, 22:16)
- Demo: Make Light Red when Temp/Dial is too high (April 30, 2022, 22:16)
- Demo: Make Light White when Temp/Dial is neither too high nor too low (April 30, 2022, 22:17)
- Demo: Stop Flashing Light when Door Closed and Button Released (April 30, 2022, 23:30)

Each rule has a "RUN ACTIONS" button and edit icons.

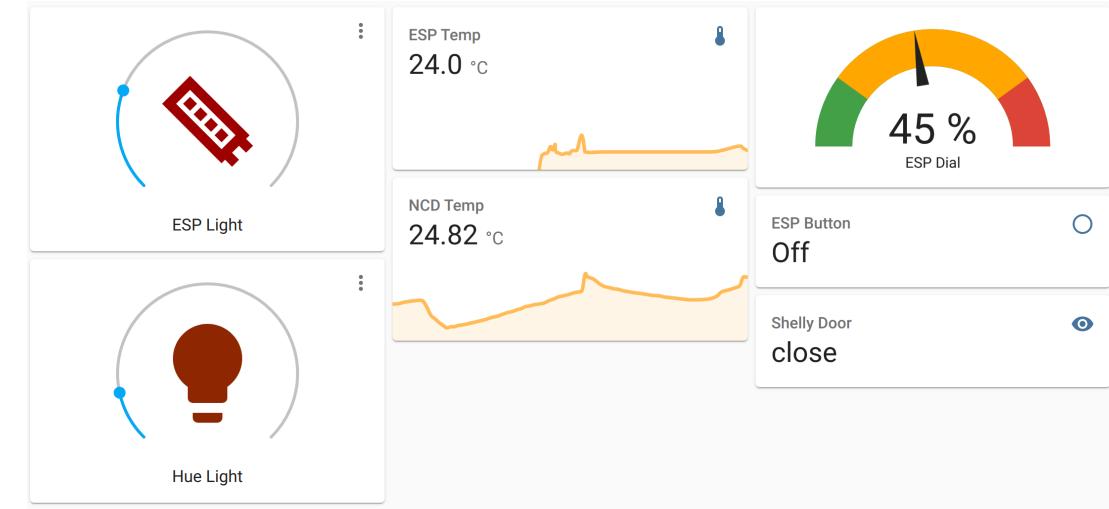
Node RED

- Rules based on WoT TD/node-wot



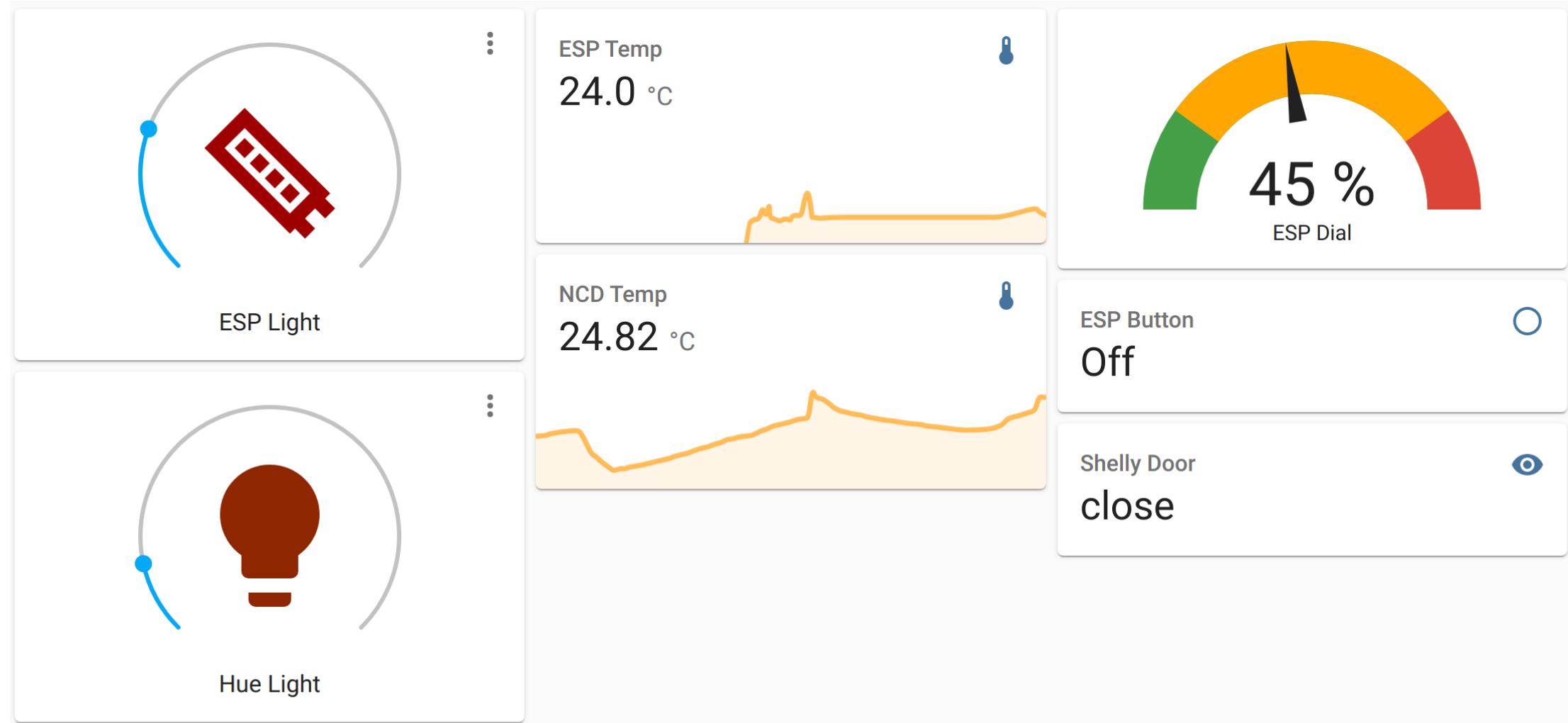
Home Assistant – Native Rule Engine

- HA is used as a "typical" gateway
 - EdgeX Foundry would be more appropriate for retail
- Frontend provides an HTTP API
- Backend supports a variety of device/ecosystem integrations
 - Hue is HTTP (proxy to Zigbee)
 - NCD Temp is MQTT
 - Shelly is MQTT and HTTP
 - Test RGBW strip, dial, button, and temp sensors are MQTT and HTTP, generated via ESPHome



- *Automation rules are based on HA-specific YAML configuration scripts*

Home Assistant – Dashboard



Home Assistant – Rules

<input checked="" type="checkbox"/>	Demo: Flash Light when Door Open or Button Pressed	April 30, 2022, 23:30	RUN ACTIONS	  
<input checked="" type="checkbox"/>	Demo: Make Light Blue when Temp/Dial is too low	April 30, 2022, 22:16	RUN ACTIONS	  
<input checked="" type="checkbox"/>	Demo: Make Light Red when Temp/Dial is too high	April 30, 2022, 22:16	RUN ACTIONS	  
<input checked="" type="checkbox"/>	Demo: Make Light White when Temp/Dial is neither too high nor too low	April 30, 2022, 22:17	RUN ACTIONS	  
<input checked="" type="checkbox"/>	Demo: Stop Flashing Light when Door Closed and Button Released	April 30, 2022, 23:30	RUN ACTIONS	  

Home Assistant – Temp High Rule

Trigger type Device ⋮

Device esp-grove-1 X ⋮

Trigger esp_grove_1 Temperature temperature changes ⋮

Above 27 °C

Below

Duration (optional)
hh mm ss
0 : 00 : 05

Trigger type Device ⋮

Device

Home Assistant – Temp High Rule

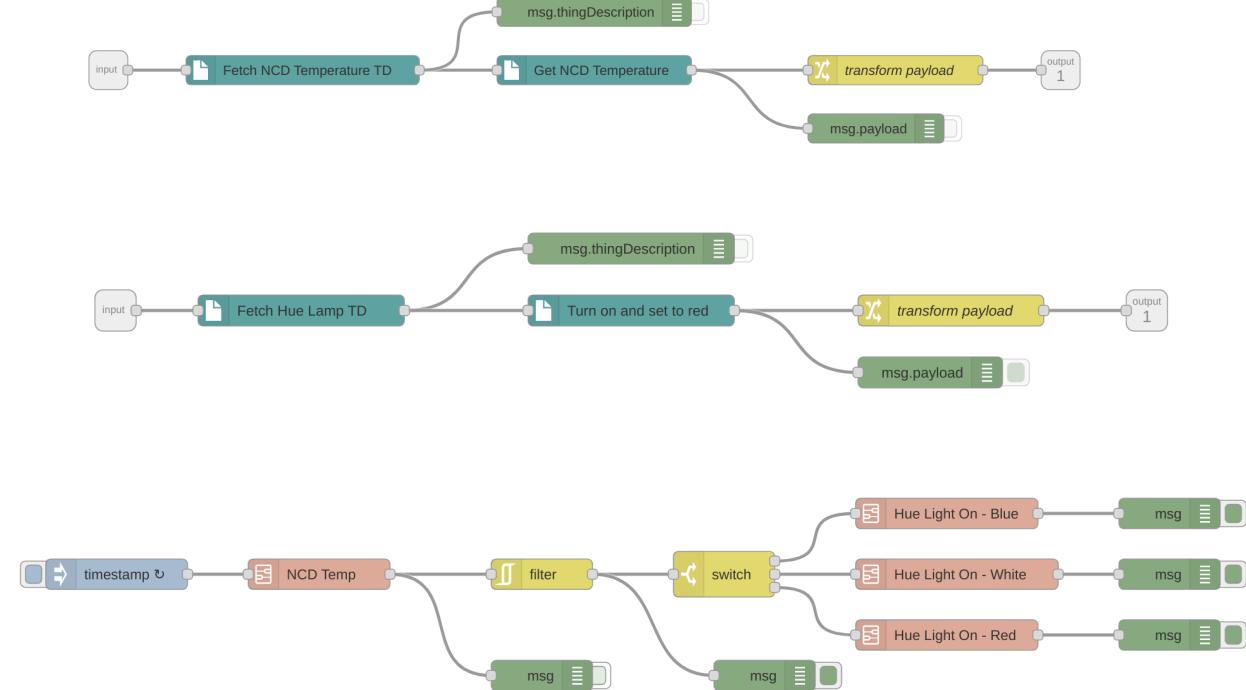
```

1 alias: 'Demo: Make Light Red when Temp/Dial is too high'
2 description: ''
3 trigger:
4   - type: temperature
5     platform: device
6     device_id: 57e2b74dc8beaf0fcb16979d8f1d8dc0
7     entity_id: sensor.esp_grove_1_temperature_2
8     domain: sensor
9     above: 27
10    for:
11      hours: 0
12      minutes: 0
13      seconds: 5
14    - type: voltage
15      platform: device
16      device_id: 57e2b74dc8beaf0fcb16979d8f1d8dc0
17      entity_id: sensor.esp_grove_1_dial
18      domain: sensor
19      above: 80
20 condition: []
21 action:
22   - service: light.turn_on
23     data:
24       rgb_color:
25         - 246
26         - 4
27         - 4
28     target:
29       entity_id: light.esp_rabw_1_rabw_stri_2

```

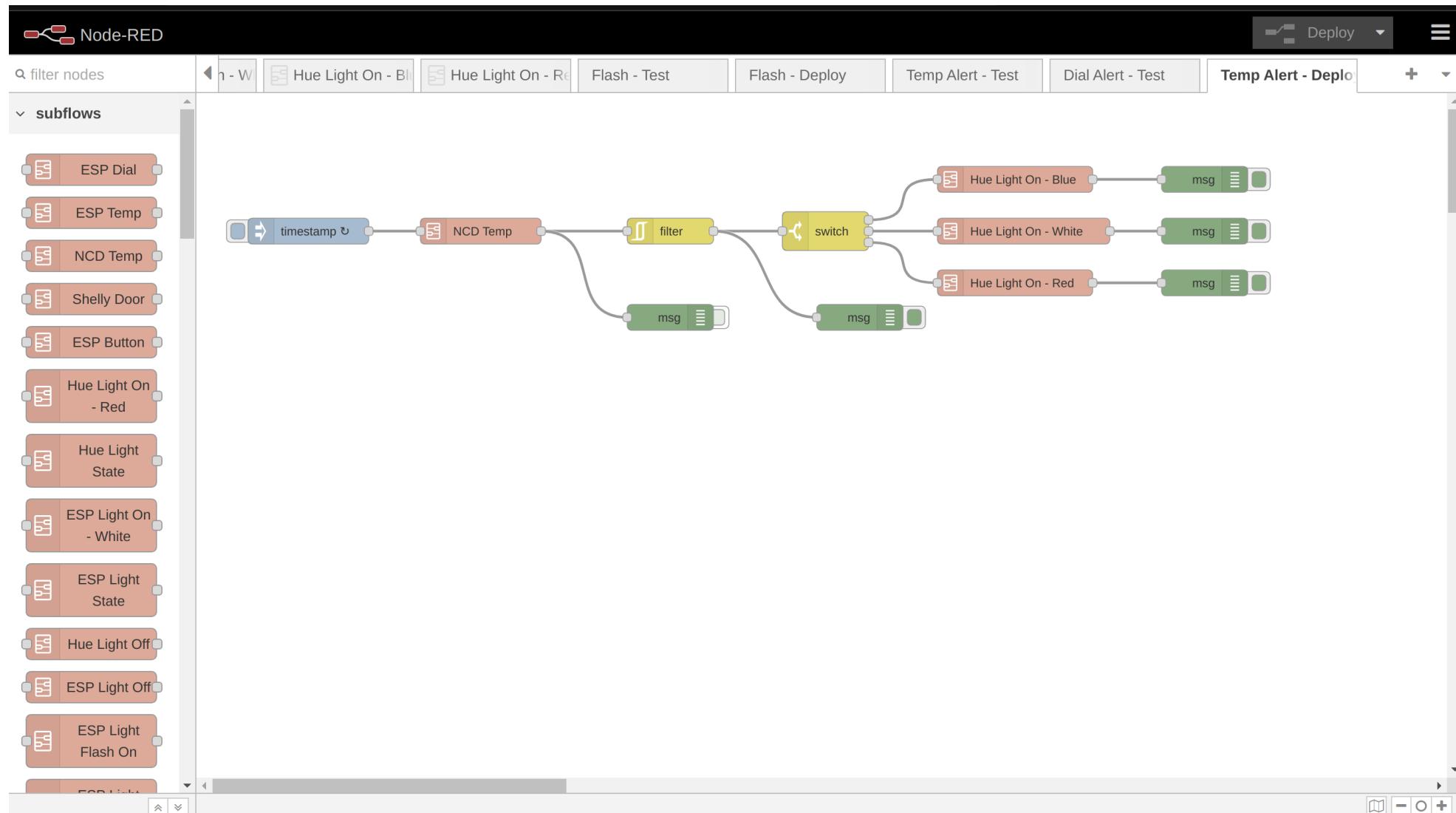
Node-RED/WoT Discovery

- Node-RED is an open-source rule engine
- An extension can be used to discover WoT TDs and interface to what they describe
- In the demo we use HA as a gateway
- **HOWEVER**, we *could* also talk directly to devices via MQTT (for example), given a suitable WoT TD

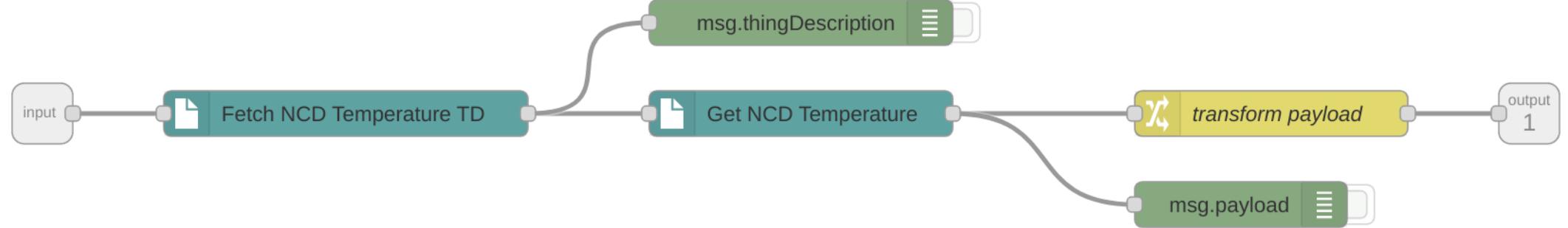


- *Automation rules are based on Node-RED "flows"*

Node-RED – Temp Rule

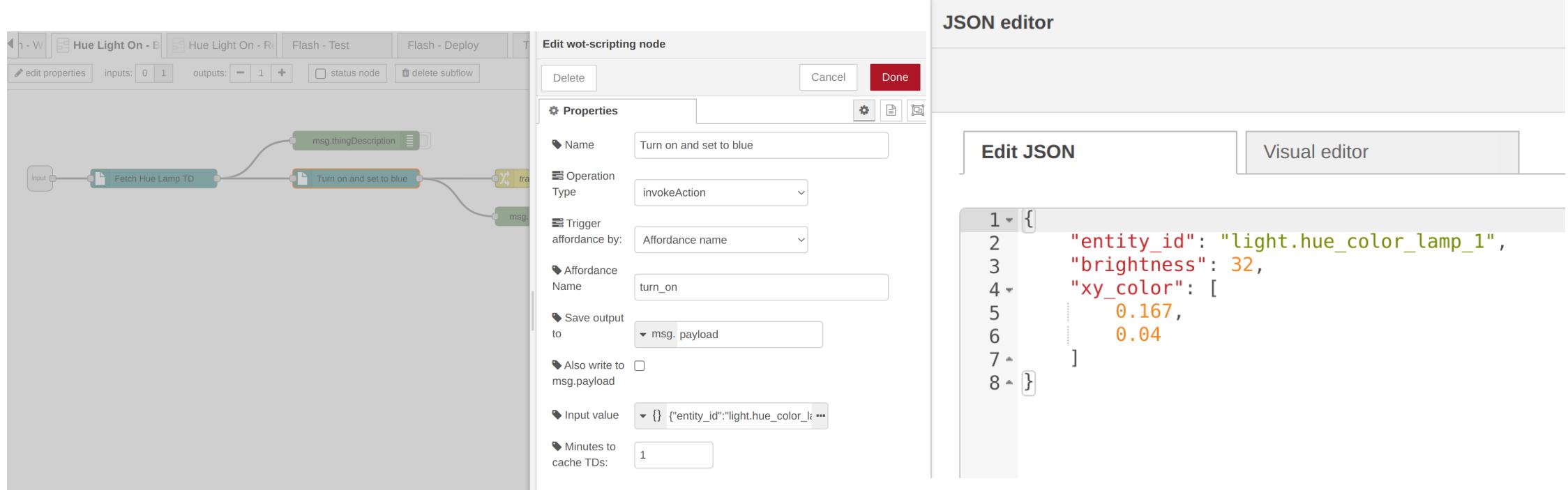


Node-RED Subflows



Interoperability

- WoT TDs describe device interfaces in a standard way
 - Interactions are now protocol and payload encoding independent
- HOWEVER, devices that do the "same" thing may still have different data models and behavior...



The image shows a screenshot of a WoT (Web of Things) development environment. On the left, there is a subflow editor window titled "Edit wot-scripting node". It displays a flowchart with nodes like "Fetch Hue Lamp TD", "Turn on and set to blue", and "msg.thingDescription". The "Properties" tab is open, showing settings such as Name ("Turn on and set to blue"), Operation Type ("invokeAction"), Trigger affordance by ("Affordance name"), Affordance Name ("turn_on"), Save output to ("msg. payload"), and Input value ("{"entity_id": "light.hue_color_lamp_1"}"). On the right, there is a "JSON editor" window with tabs for "Edit JSON" and "Visual editor". The "Edit JSON" tab shows a JSON object with properties: "entity_id": "light.hue_color_lamp_1", "brightness": 32, and "xy_color": [0.167, 0.04]. The "Visual editor" tab is also visible.

Semantic Interoperability

- WoT Thing Descriptions
 - Currently address "syntactic interoperability"
 - Transform multiple protocols into a common interaction abstraction
 - Transform multiple payload encodings into a common format (JSON)
- Next Steps
 - Annotating Thing Description with semantics
 - Transform data into a standardized data model
 - WoT TDs are encoded using JSON-RD, which provides for *semantic annotation* and inferencing
 - Still need (**Conexxus can contribute here**):
 - Use cases driving required operations and behavior
 - Common semantic vocabularies
 - Standardized data models

Resources and Contacts

<https://www.w3.org/WoT>

Dr. Michael McCool
Principal Engineer

Intel
Technology Pathfinding

michael.mccool@intel.com

Dr. Sebastian Kaebisch
Senior Key Expert

Siemens
Technology

sebastian.kaebisch@siemens.com