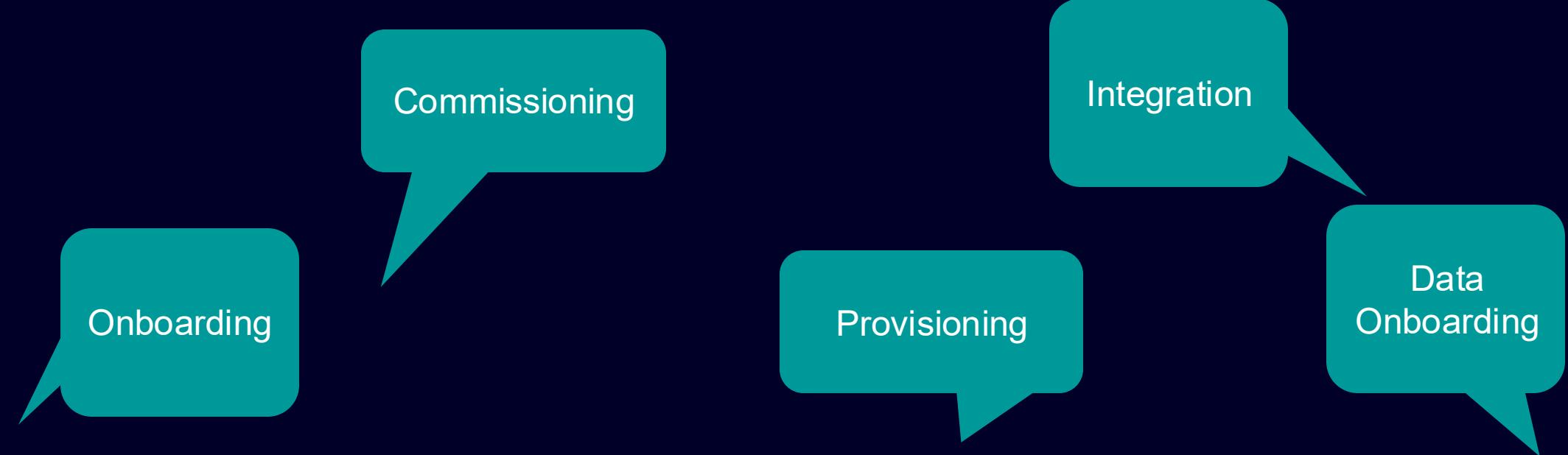


Managing Device Descriptions with the Thing Model Catalog

Ege Korkan

Siemens AG
Smart Infrastructure, CTO Office



Adding a new Device

State of the Art, Focusing on the Modbus TCP Protocol
Home automation forums

Weishaupt WBB, WWP LS und WGB Modbus Configuration

Tutorials & Examples Solutions

B

Blizzard Andreas

1 Oct 2024

[openHAB forum](#)

Hi all,

I wanted to share my Modbus Configuration for reading (and writing) Weishaupt Heatpumps WBB, WWP LS und WGB (according to the documentation I have). Maybe some other people have a Weishaupt Heatpump, too, and can make use of it. You need to enable Modbus in the Expert Section of the Heatpump. This configuration has been running with my heatpump for quite some time.



```
"Weishaupt Wärmepumpe" [ start="41101", length="12", refresh="30000", type="hold":  
    - Konfiguration" [ readStart="41101", valueType="int16" ] // |  
    - Anforderung Typ" [ readStart="41102", valueType="int16" ] // |  
    - Betriebsart" [ readStart="41103", valueType="int16", writeType="int16" ] // |  
    - Pause / Party" [ readStart="41104", valueType="int16", writeType="int16" ] // |  
    - Raumsolltemperatur Komfort" [ readStart="41105", valueType="int16" ] // |  
    - Raumsolltemperatur Normal" [ readStart="41106", valueType="int16" ] // |  
    - Raumsolltemperatur Absenk" [ readStart="41107", valueType="int16" ] // |  
    - Heizkennlinie" [ readStart="41108", valueType="int16" ] // |  
    - Sommer Winter Umschaltung" [ readStart="41109", valueType="int16" ] // |  
    - Heizen Konstanttemperatur" [ readStart="41110", valueType="int16" ] // |  
    - Heizen Konstanttemp Absenk" [ readStart="41111", valueType="int16" ] // |  
    - Kühlen Konstanttemperatur" [ readStart="41112", valueType="int16" ] // |
```

Weishaupt Heatpump integration via modbus

Share your Projects!



tobiasm TobiM

2 Jul 2022

Dear Weishaupt Users,

today i would like to share with you how i have integrated my Weishaupt Biblock Heat pump via modbus in Homeassistant.

I was in touch with some Weishaupt employees and happy that they have shared a List of Datapoints for modbus connectivity. Other ways of integrating were not 100% stable & i have now tested modbus through the last months & it works really great! It is even possible to set e.g. Operating mode or Target warmwater temperature.

Here are the steps to follow:

1. Configure Modbus with static IP on your Weishaupt Device (not possible in WEMPortal - you need to configure it on the device)
2. Configure modbus connectivity - here is my sample yaml:

► Modbus YAML config

```
- name: wpump
  type: tcp
  host: 192.168.X.X
  port: 502
  climates:
    - name: "WP_Warmwasser"
      address: 42103
      input_type: holding
      count: 1
      data_type: int16
      max_temp: 50
      min_temp: 15
      offset: 0
      precision: 1
      scale: 0.1
      target_temp_register: 42103
      temp_step: 1
      temperature_unit: C
  sensors:
#Warmwasser
    - name: WP_Warmwassersolltemperatur
      slave: 1
      address: 32101
      input_type: input
      unit_of_measurement: °C
      state_class: measurement
      count: 1
      scale: 0.1
```

[HomeAssistant Forum](#)

At least Siemens does
better ☺

Or does not...

vaox Valter 1 Nov 2022

Hi, I would like to read from Sentron PAC3200 . I have tried this configuration:

modbus:

```
• name: pac3200
  type: tcp
  host: 192.168.5.3
  port: 502
  binary_sensors:
    • name: "Voltage L1"
      slave: 0
      address: 1
      input_type: holding
      unit_of_measurement: V (this command generate error)
      state_class: measurement (this command generate error)
      count: 1 (this command generate error)
      offset: 0 (this command generate error)
      scale: 0.1 (this command generate error)
      data_type: uint16 (this command generate error)
```

Example of error

Invalid config for [modbus]: [scale] is an invalid option for [modbus]. Check: modbus->modbus->0->binary_sensors->0->scale. (See /config/configuration.yaml, line 17).

This is part of manual that shows how to read data from device.

3.9.3 Modbus measured variables with the function codes 0x03 and 0x04

Measured variables of the SENTRON PAC Power Monitoring Device

The measured variables are provided by the SENTRON PAC Power Monitoring Device. You can use the MODBUS function codes 0x03 and 0x04 on all the measured variables listed below.

nikito7

Change binary_sensors: to sensors:
and
data_type: float32
count: 2

Some more Siemens Devices don't hurt



cloom

Hello @fesklord,

I have a LOGO where input registers are temperatures, they are short integer so if I read 321 it means 32.1°C.

I divide my configuration.yaml so I have to edit 3 files:

- **configuration.yaml:**

```
modbus: !include modbus.yaml
template: !include templates.yaml
```

- **modbus.yaml** (please note the host is your LOGO IP address, I store mine in the secrets.yaml file):

```
- name: "Siemens LOGO"
  type: tcp
  host: !secret siemens_logo_ip
  port: 502
  sensors:
    - name: "MyLOGO1"
      unique_id: mylogo1
      input_type: input
      slave: 1
      address: 0
      count: 8
      data_type: custom
      structure: ">8h"
```

Note: `structure:>8h` means I have reading 8 short number of 2 bytes each with big endian ending. You can read about Python structure [here](#) [21].

Dec 2022

- As I read all 8 input registers at once and then use a template to separate the values, then I have less modbus requests made to the LOGO. So I divide the values in **templates.yaml**:

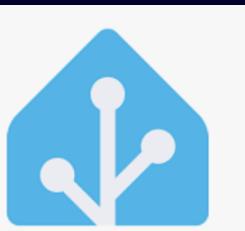
```
- name: Heater temperature
  state: "{{ states('sensor.mylogo1').split(',')[0] | multiply(0.1) }}"
  state_class: measurement
  unit_of_measurement: °C
- name: External temperature
  state: "{{ states('sensor.mylogo1').split(',')[1] | multiply(0.1) }}"
  state_class: measurement
  unit_of_measurement: °C
# and so on until states('sensor.mylogo1').split(',')[7]
```

I hope that helps and do not hesitate to reach out if you have any question.

SIEMENS

What are all these people doing?

```
modbus:  
  - name: hub1  
    type: tcp  
    host: IP_ADDRESS  
    port: 502  
  binary_sensors:  
    - name: my_relay  
      address: 100  
      device_class: door  
      input_type: coil  
      scan_interval: 15
```



Communication protocol *

Modbus TCP Connector

Name*

ModbusServer

IPv4 address*

10.14.6.99

IP Port Number*

502

Slave Address*

1

Layout*

CDAB

Acquisition Mode*

CyclicOnChange

Response timeout (ms)

1000

Zero based addressing

Change bit order

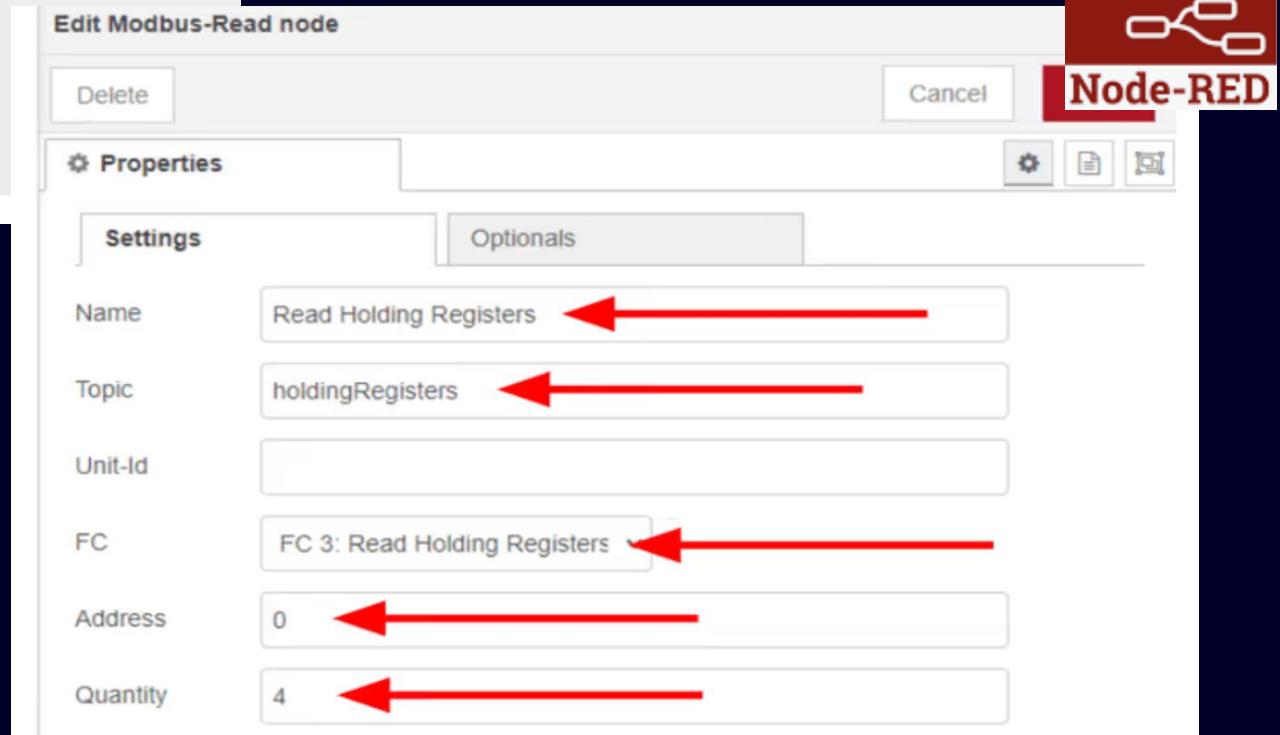
Use single write

Powered by
Siemens
Industrial
Edge

What are all these people doing?



```
<tag>
  <tagDefinition>
    <unitId>0</unitId>
    <startIdx>11</startIdx>
    <readType>HOLDING_REGISTERS</readType>
    <dataType>INT_64</dataType>
  </tagDefinition>
  <tagName>myTag</tagName>
</tag>
```

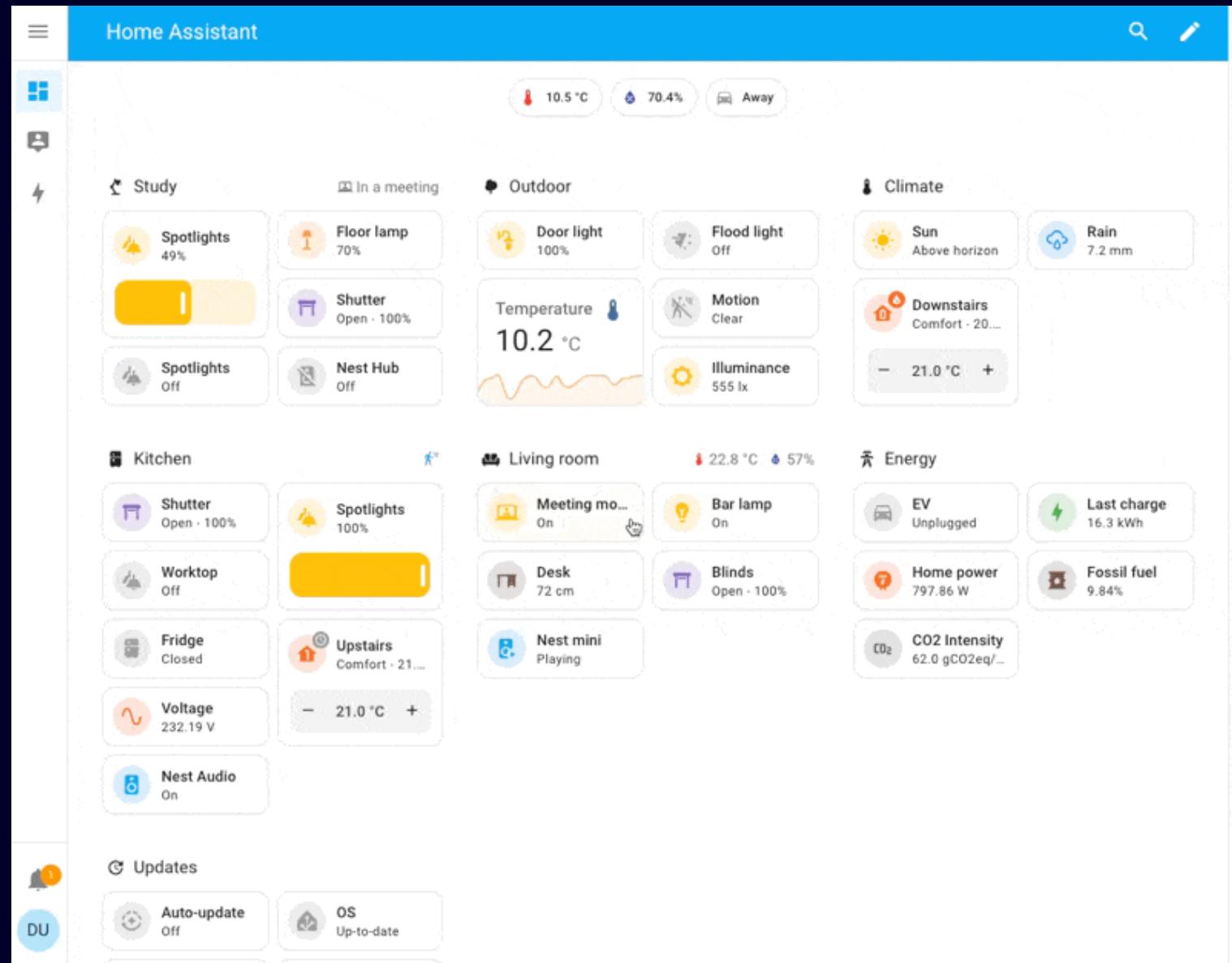


The screenshot shows the "Edit Modbus-Read node" dialog in Node-RED. The "Properties" tab is selected, showing the "Settings" tab. The configuration fields and their values are:

Setting	Value
Name	Read Holding Registers
Topic	holdingRegisters
Unit-Id	
FC	FC 3: Read Holding Registers
Address	0
Quantity	4

Red arrows point from each setting back to its corresponding XML configuration element on the left.

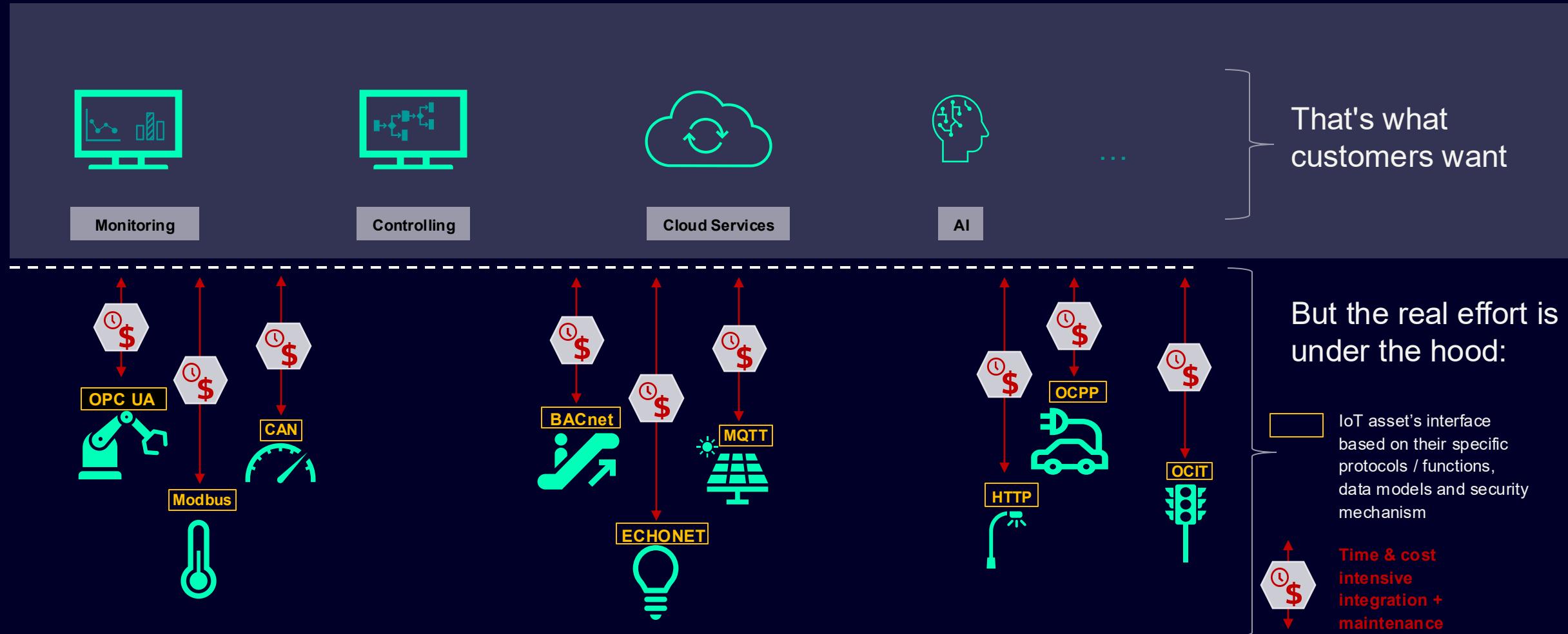
End Goal



End Goal



There is a common problem if you want to use Things smart: Onboarding



Quantifying the Problem



The Modbus integration was introduced in Home Assistant pre 0.7, and it's used by **2.6%** of the active installations.

🔌 Its IoT class is [Local Polling](#).

[View source on GitHub](#)

[View known issues](#)



13 000 Similar Cases for 1 Protocol in 1 Platform! (2.6% of 500k)
Also: 14.8k in ioBroker

For us, smart home is not the target market,
but it is not that different in more commercial settings

Why not build integrations for these platforms?

Sure it is *possible*!

Two problems, of course 😊

1. Which device?
2. Which platform?

Also, it doesn't work seamlessly when there is no discovery built-in.

Integrations

All Featured Partners

Category	Version	IoT Class
All	All	All

Search integrations...



1-Wire®
Long distance
sensors and devices

1-Wire



17

17TRACK



3 Day Blinds



A.O. Smith.

A. O. Smith



abode

Abode



a caia

Acaia



acomax

Acomax



Actiontec

Actiontec



Activity

Activity



Actron Air

Actron Air



ADAX

Adax



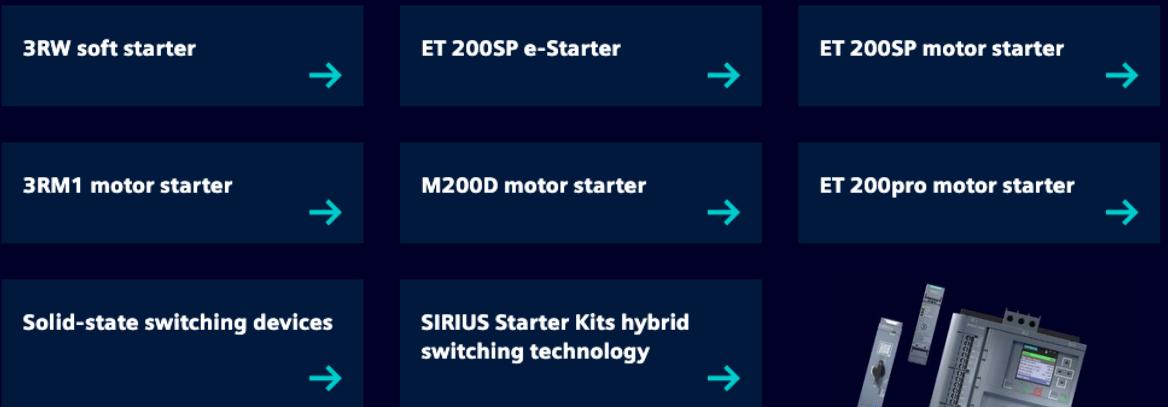
AdGuard Home

AdGuard Home

<https://www.home-assistant.io/integrations/>

1. Which device?

Rather huge device portfolio!



Motor starters



Servo drives



Each is a family with many subtypes

2. Which platform?

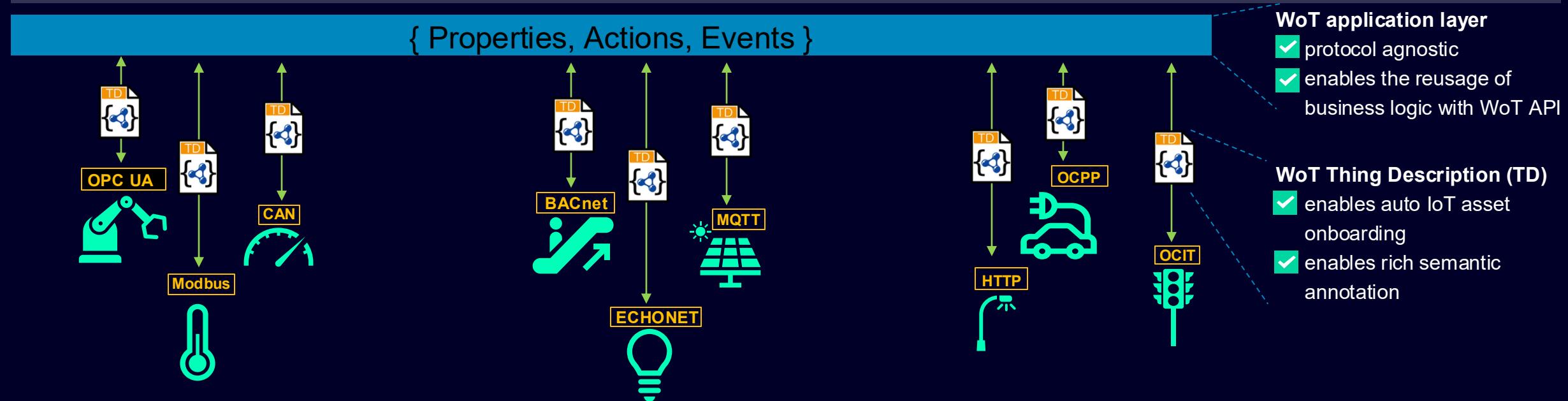
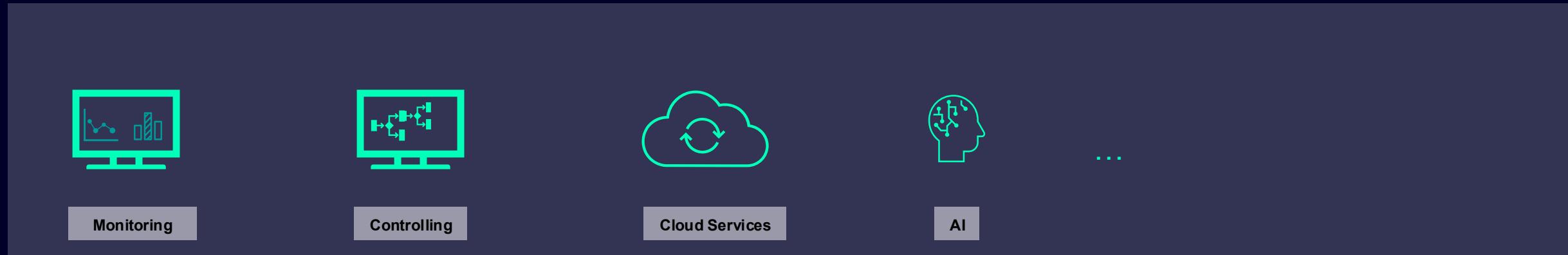
Google search results for "home automation platforms". The results are displayed in a grid format:

Icon	Name	Icon	Name	Icon	Name
	Apple HomeKit		Amazon Alexa		Home Assistant
	SmartThings		Domoticz		works with Google Home
	Nest Labs		ioBroker		Jeedom
	OpenHAB		FHEM		Pimatic
	Ago Control		Calaos		Control4
	Homebridge		MyController		OpenMotics
	Philips Hue		PiDome		SimpliSafe
	Smarthomatic		Sonos		

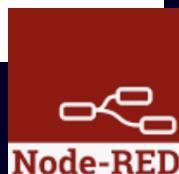
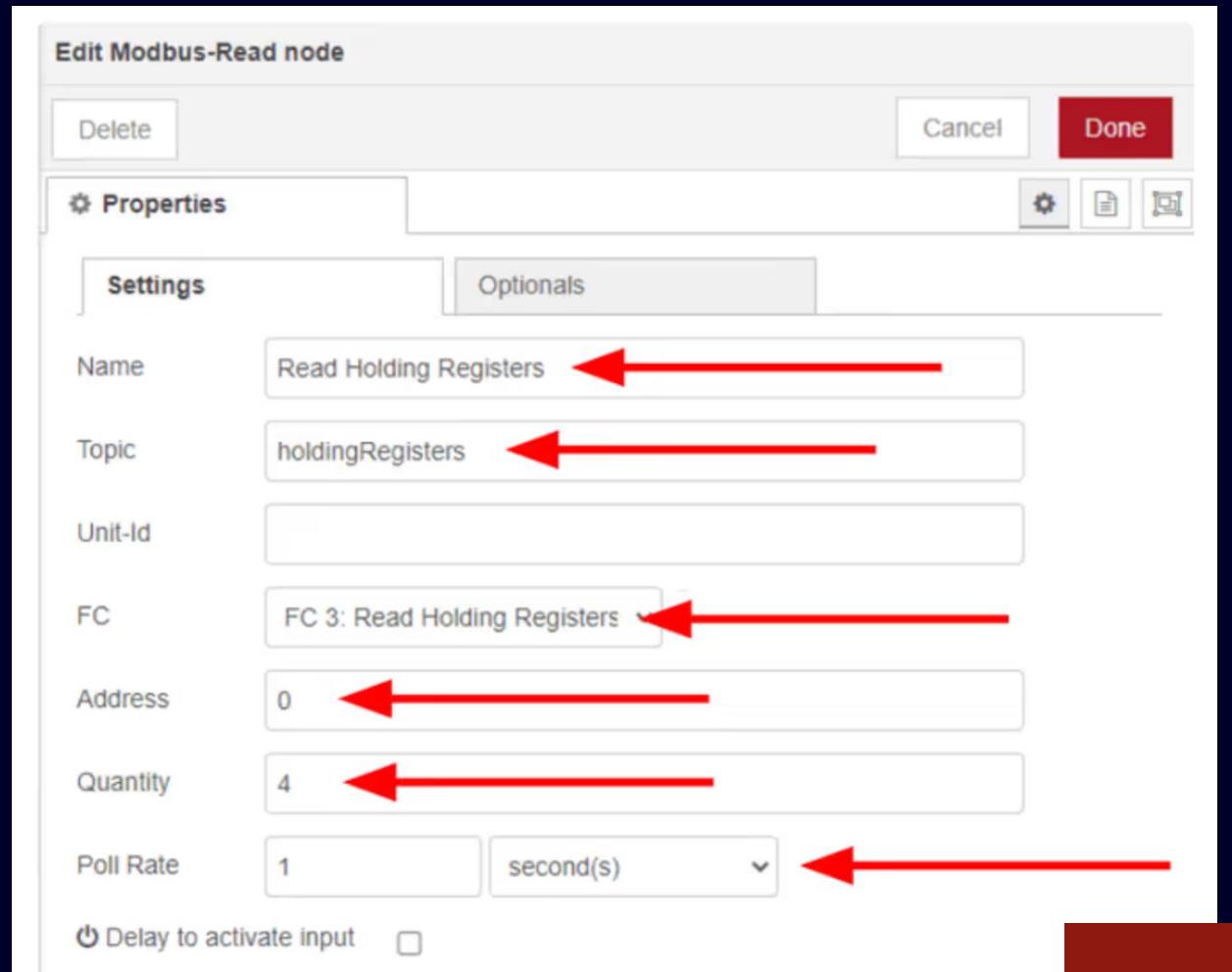
- Not all are platforms
- Still too many
- All with their own device modeling language

Web of Things to the rescue

Tackling the onboarding problem



Example TD for the Node-RED example from before



TD Extract

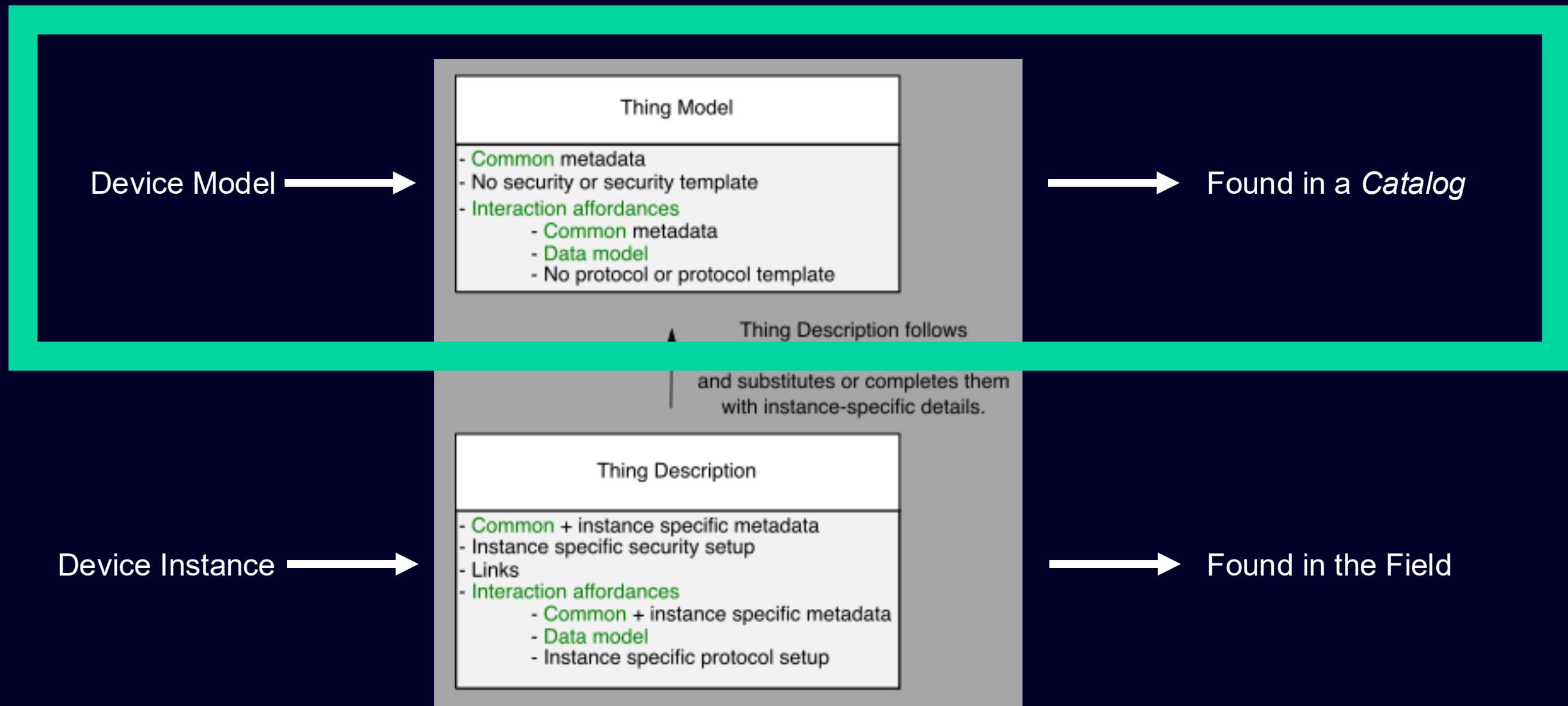
does not repeat

```
"base": "modbus+tcp://192.168.0.10:502",  
...  
{  
    "href": "0?quantity=4",  
    "modv:function": "readHoldingRegisters",  
    "modv:type": "xsd:float",  
    "modv:mostSignificantByte": true,  
    "modv:mostSignificantWord": true  
}
```

repeats for each instance



Concentrating on the Thing Model



What kind of Thing Model though?

Focus of today

Generic device/data model

```
{  
  "@context": ["https://www.w3.org/2022/wot/td/v1.1"],  
  "@type": "tm:ThingModel",  
  "title": "Switchable",  
  "properties": {  
    "on": {  
      "description": "Whether the switch is on or off.",  
      "type": "boolean"  
    }  
  },  
  "actions": {  
    "toggle": {  
      "description": "Toggles/inverts the current 'on' state.",  
      "output": {  
        "title": "New 'on' state",  
        "type": "boolean"  
      }  
    },  
    "switch-on-for-duration": {  
      "description": "Switches the switchable on for a given duration",  
      "input": {  
        "title": "Duration in seconds",  
        "type": "integer",  
        "unit": "time:seconds"  
      }  
    }  
  }  
}
```

[Example from Eclipse Ditto](#)

Specific device, just not instantiated

```
{  
  "@context": ["https://www.w3.org/2022/wot/td/v1.1"],  
  "@type": "tm:ThingModel",  
  "title": "3NACOM_FUSE",  
  "base": "modbus://{{IP}}:{{PORT}}",  
  "securityDefinitions": {"nosec_sc": {"scheme": "nosec"}},  
  "security": "nosec_sc",  
  "properties": {  
    "I_L1_AVERAGE": {  
      "title": "Average Current",  
      "readOnly": true,  
      "type": "number",  
      "unit": "om:ampere",  
      "forms": [  
        {  
          "op": "readproperty",  
          "href": "/",  
          "modbus:unitID": "{{UNITID}}",  
          "modbus:quantity": 2,  
          "modbus:address": 3078,  
          "modbus:type": "number",  
          "modbus:entity": "HoldingRegister",  
          "modbus:zeroBasedAddressing": false,  
          "modbus:pollingTime": 2000  
        }  
      ]  
    }  
  }  
}
```

[Example from Siemens](#)

SIEMENS



Thing Model catalog

Tooling, home and services

<https://github.com/wot-oss/tmc>

Some Design Goals

- **Authors** contributing TMs of **device models** of **manufacturers**
- Finding a TM of a device “in front of you”
- Supporting online and offline deployment
- Git and CI/CD friendly
- No domain-specific requirements
- Federated deployment (merging multiple repositories in one API)

Importantly: Not centralizing TMs in one place.

- Provide the tooling to you to manage your own TM Catalog

You can find more at <https://github.com/wot-oss/proposal/issues/8>

First sightings in the wild:

Pedram Hadjian · 1st
Siemens Software Strategy & Architecture
8mo · Edited ·

It's Beginning to Look a Lot Like Christmas...

We're delighted to announce the release of official W3C Thing Models (TMs) for our Siemens SENTRON Smart Fuses, now available under the Apache 2.0 license! These models make it easier than ever for system integrators to incorporate our devices into any IoT platform.

With these Thing Models, you can accelerate integration and simplify the development of advanced automation solutions.

Explore the models on github:
<https://lnkd.in/eZxT7M-R>

Our Sentron Smart Fuses:
<https://lnkd.in/ecpKF5yC>

If you're new to the Web of Things, have a look at the standard:
<https://lnkd.in/erzQtVaN>

This is our commitment to fostering openness and interoperability in the IoT ecosystem. Let's make buildings smarter and operations smoother together!

#IoT #ThingModels #OpenSource #SiemensSENTRON #BuildingAutomation #



thingmodels / siemens / siemens /

alexbrdn add README.md; remove non-compliant TM for PAC3220

Name
..
3nacom-fuse
3rv2com-msp
5sl6-com-mp-mcb-rcm
5sl6com-mcb-rcm
5sl6com-mcb
5st3com-asfc
5st3com-rca-rcdir
5st3com-rca
5sv6com-afdd
5sv8com-rcm
5ty-com-ecpd
poc1000

SENTRON Smart Fuses

Apache License

Manufacturer Publication

Thing Model Catalog: Tooling

README Apache-2.0 license

Thing Model Catalog CLI

go report A+ release v0.1.2 go.dev docs license scan passing



Find, use and contribute device descriptions for industrial IoT devices!

<https://www.github.com/wot-oss/tmc>

Usage:
tmc [command]

Available Commands:

attachment	Manage TM attachments
check	Check the integrity of all or only named resources in repository's internal index
completion	Generate the autocompletion script for the specified shell
copy	Copy multiple TMs and their attachments from one repository to another
create-si	Create or update search index
delete	Delete a TM by id
export	Export multiple TMs from a catalog and, optionally, their attachments
fetch	Fetch a TM by name or id
help	Help about any command
import	Import a TM or a directory with TMs into a catalog
index	Refresh the repository's internal index, if it has one
list	List TMs in catalog
repo	Manage repositories
search	Search full text of TMs in catalog using bleve search engine
serve	Start a REST API server
validate	Validate a TM before importing
version	Show tmc version information
versions	List available versions of the TM with given name

Also available with a REST API

Getting Started Documentation

Type to search

Thing Model Catalog - Documentation

Installation

Getting Started

- Configure Autocompletion (Optio...)

Browse the Example Catalog

- Configure the Example Reposi...
- List the Contents of the Exam...
- List Versions
- Fetch a Thing Model

Host Your Own Catalog

Concepts

Workflows

Commands

Architecture

Frequently Asked Questions

Browse the Example Catalog

We provide an [example repository](#) for you to get acquainted with `tmc`. The following commands assume that you use the example repository. If your organization hosts a TM catalog for use as a default, you will need to change the commands accordingly.

Configure the Example Repository

```
tmc repo add -t http example https://raw.githubusercontent.com/wot-oss/example-catalog/main
```

[Copy](#)

List the Contents of the Example Repository

```
tmc list
```

[Copy](#)

The listed names are formatted as follows

```
<author>/<manufacturer>/<model>/[<optional_path>]
```

[Copy](#)

<https://wot-oss.github.io/tmc/gettingstarted>

Web UI (Quite new, not perfect yet)

Thing Model catalog Thing Model Catalog Dashboard Settings Light

Search...

Filters

Protocol +
Manufacturer +
Author +
Repository +

3614 results found Reset filters TMs per page: 20

omnicorp/omnicorp/lightal omnicorp/omnicorp/lightall	omnicorp/omnicorp/lightal omnicorp/omnicorp/lightall-n	omniuser/omnicorp/sense: omniuser/omnicorp/senseall
omnicorp main2, lightall omnicorp/omnicorp/lightall main2	omnicorp main2, lightall-mk2 omnicorp/omnicorp/light... main2	omnicorp main2, senseall omniuser/omnicorp/sens... main2
siemens/siemens/3nacom- siemens/siemens/3nacom-fu	siemens/siemens/3rv2com- siemens/siemens/3rv2com-n	siemens/siemens/5sl6-cor siemens/siemens/5sl6-com-i
Siemens main, 3NACOM_FUSE siemens/siemens/3naco... main	Siemens main, 3RV2COM_MSP siemens/siemens/3rv2co... main	Siemens main, 5SL6_COM_MP_MCB_R... siemens/siemens/5sl6-c... main

- Configurable TMC endpoint
- Customizable Theme
- Follow TMC REST API
- Open in EdiTDoR, Playground

TM Content

- We catalog based on the author (like a namespace), manufacturer and product name (Model Part Number). So schema.org annotations are required

```
"schema:author": {  
    "schema:name": "Siemens"  
,  
"schema:manufacturer": {  
    "schema:name": "Siemens"  
,  
"schema:mpn": "5ST3COM_RCA"
```

- The id you will have in your TM will be overwritten to include a timestamp and a hash based on the content. author/manufacturer/mpn will be also used

```
"id": "siemens/siemens/5st3com-rca/v1.0.0-20240802121832-941071515746.tm.json",
```

- We can sort TMs, guarantee uniqueness and more. See <https://github.com/wot-oss/proposal/issues/10>

Let's see it in action!

Demo 1: Simple CLI and REST API, Federated Demo

1. Install TMC CLI
2. Add a file repository (local on my computer)
3. Do some operations with CLI
4. Do same operations via REST API
5. Add a git repository (on GitHub.com)
6. Show multiple repositories under one roof

How to Contribute TMs?

Authoring Tools: Easier Creation TMs

The screenshot shows the **editTDor** interface, a web-based tool for creating and editing Thing Description (TD) files. The left side of the screen displays a navigation tree and configuration options for a device named "HotelRoom PowerMeter". The right side shows the JSON schema for "http-data-schema-thing" and a large blue "Contribute to Catalog" button with a white upward arrow icon.

JSON Validation ✓
JSON Schema Validation ✓

HotelRoom PowerMeter

Properties (①)

List **Table**

Group Controls

Unit ID (①) **1** **+** **-** **Address Of**

> **device_id**
> **serial_number**
> **creation_date**
> **battery_failure**
> **voltage_constant**
> **light_bulb_power_in_ampere**
> **battery_charge_in_percent**
> **grid_voltage**
> **statistic_value**

Actions (①)

Properties: 9 | Actions: 0 | Events: 0 | Size: 3.363 KiB | Northbound State:

Send TD Contribute to Catalog Share Open Create To TD Download Settings

http-data-schema-thing

```
1  {
2   "@context": ["https://www.w3.org/2019/wot/td/v1"],
3   "id": "urn:PowerMeter0",
4   "securityDefinitions": {
5     "nosec_sc": {
6       "scheme": "nosec"
7     }
8   },
9   "title": "HotelRoom PowerMeter0",
10  "type": "PowerMeter",
11  "base": "modbus://{{IP}}:{{PORT}}",
12  "actions": [
13    {
14      "name": "readProperty",
15      "op": "readProperty",
16      "modbusType": "string",
17      "modbusEntity": "inputRegister"
18    }
19  ],
20  "properties": [
21    {
22      "name": "battery_failure",
23      "title": "battery_failure",
24      "type": "boolean",
25      "const": 0,
26      "forms": [
27        {
28          "name": "checkbox"
29        }
30      ]
31    }
32  ],
33  "events": [
34    {
35      "name": "grid_voltage",
36      "title": "grid_voltage",
37      "type": "number",
38      "const": 0,
39      "forms": [
40        {
41          "name": "range"
42        }
43      ]
44    }
45  ],
46  "links": [
47    {
48      "name": "link1",
49      "target": "http://example.com/resource1"
50    },
51    {
52      "name": "link2",
53      "target": "http://example.com/resource2"
54    }
55  ],
56  "forms": [
57    {
58      "name": "form1",
59      "type": "text"
60    },
61    {
62      "name": "form2",
63      "type": "button"
64    }
65  ],
66  "actions": [
67    {
68      "name": "action1",
69      "op": "readProperty",
70      "modbusType": "string",
71      "modbusEntity": "inputRegister"
72    }
73  ],
74  "properties": [
75    {
76      "name": "voltage_constant",
77      "title": "voltage_constant",
78      "type": "number",
79      "const": 0,
80      "forms": [
81        {
82          "name": "range"
83        }
84      ]
85    }
86  ],
87  "events": [
88    {
89      "name": "battery_charge_in_percent",
90      "title": "battery_charge_in_percent",
91      "type": "number",
92      "const": 0,
93      "forms": [
94        {
95          "name": "range"
96        }
97      ]
98    }
99  ],
100 "links": [
101    {
102      "name": "link3",
103      "target": "http://example.com/resource3"
104    }
105  ],
106  "forms": [
107    {
108      "name": "form3",
109      "type": "button"
110    }
111  ],
112  "actions": [
113    {
114      "name": "action2",
115      "op": "readProperty",
116      "modbusType": "string",
117      "modbusEntity": "inputRegister"
118    }
119  ],
120  "properties": [
121    {
122      "name": "serial_number",
123      "title": "serial_number",
124      "type": "string",
125      "const": "1234567890",
126      "forms": [
127        {
128          "name": "text"
129        }
130      ]
131    }
132  ],
133  "events": [
134    {
135      "name": "creation_date",
136      "title": "creation_date",
137      "type": "date",
138      "const": "2023-01-01T00:00:00Z",
139      "forms": [
140        {
141          "name": "date"
142        }
143      ]
144    }
145  ],
146  "links": [
147    {
148      "name": "link4",
149      "target": "http://example.com/resource4"
150    }
151  ],
152  "forms": [
153    {
154      "name": "form4",
155      "type": "button"
156    }
157  ],
158  "actions": [
159    {
160      "name": "action3",
161      "op": "readProperty",
162      "modbusType": "string",
163      "modbusEntity": "inputRegister"
164    }
165  ],
166  "properties": [
167    {
168      "name": "grid_voltage",
169      "title": "grid_voltage",
170      "type": "number",
171      "const": 0,
172      "forms": [
173        {
174          "name": "range"
175        }
176      ]
177    }
178  ],
179  "events": [
180    {
181      "name": "battery_charge_in_percent",
182      "title": "battery_charge_in_percent",
183      "type": "number",
184      "const": 0,
185      "forms": [
186        {
187          "name": "range"
188        }
189      ]
190    }
191  ],
192  "links": [
193    {
194      "name": "link5",
195      "target": "http://example.com/resource5"
196    }
197  ],
198  "forms": [
199    {
200      "name": "form5",
201      "type": "button"
202    }
203  ],
204  "actions": [
205    {
206      "name": "action4",
207      "op": "readProperty",
208      "modbusType": "string",
209      "modbusEntity": "inputRegister"
210    }
211  ],
212  "properties": [
213    {
214      "name": "device_id",
215      "title": "device_id",
216      "type": "string",
217      "const": "DeviceID001",
218      "forms": [
219        {
220          "name": "text"
221        }
222      ]
223    }
224  ],
225  "events": [
226    {
227      "name": "grid_voltage",
228      "title": "grid_voltage",
229      "type": "number",
230      "const": 0,
231      "forms": [
232        {
233          "name": "range"
234        }
235      ]
236    }
237  ],
238  "links": [
239    {
240      "name": "link6",
241      "target": "http://example.com/resource6"
242    }
243  ],
244  "forms": [
245    {
246      "name": "form6",
247      "type": "button"
248    }
249  ],
250  "actions": [
251    {
252      "name": "action5",
253      "op": "readProperty",
254      "modbusType": "string",
255      "modbusEntity": "inputRegister"
256    }
257  ],
258  "properties": [
259    {
260      "name": "battery_charge_in_percent",
261      "title": "battery_charge_in_percent",
262      "type": "number",
263      "const": 0,
264      "forms": [
265        {
266          "name": "range"
267        }
268      ]
269    }
270  ],
271  "events": [
272    {
273      "name": "grid_voltage",
274      "title": "grid_voltage",
275      "type": "number",
276      "const": 0,
277      "forms": [
278        {
279          "name": "range"
280        }
281      ]
282    }
283  ],
284  "links": [
285    {
286      "name": "link7",
287      "target": "http://example.com/resource7"
288    }
289  ],
290  "forms": [
291    {
292      "name": "form7",
293      "type": "button"
294    }
295  ],
296  "actions": [
297    {
298      "name": "action6",
299      "op": "readProperty",
300      "modbusType": "string",
301      "modbusEntity": "inputRegister"
302    }
303  ],
304  "properties": [
305    {
306      "name": "device_id",
307      "title": "device_id",
308      "type": "string",
309      "const": "DeviceID002",
310      "forms": [
311        {
312          "name": "text"
313        }
314      ]
315    }
316  ],
317  "events": [
318    {
319      "name": "grid_voltage",
320      "title": "grid_voltage",
321      "type": "number",
322      "const": 0,
323      "forms": [
324        {
325          "name": "range"
326        }
327      ]
328    }
329  ],
330  "links": [
331    {
332      "name": "link8",
333      "target": "http://example.com/resource8"
334    }
335  ],
336  "forms": [
337    {
338      "name": "form8",
339      "type": "button"
340    }
341  ],
342  "actions": [
343    {
344      "name": "action7",
345      "op": "readProperty",
346      "modbusType": "string",
347      "modbusEntity": "inputRegister"
348    }
349  ],
350  "properties": [
351    {
352      "name": "device_id",
353      "title": "device_id",
354      "type": "string",
355      "const": "DeviceID003",
356      "forms": [
357        {
358          "name": "text"
359        }
360      ]
361    }
362  ],
363  "events": [
364    {
365      "name": "grid_voltage",
366      "title": "grid_voltage",
367      "type": "number",
368      "const": 0,
369      "forms": [
370        {
371          "name": "range"
372        }
373      ]
374    }
375  ],
376  "links": [
377    {
378      "name": "link9",
379      "target": "http://example.com/resource9"
380    }
381  ],
382  "forms": [
383    {
384      "name": "form9",
385      "type": "button"
386    }
387  ],
388  "actions": [
389    {
390      "name": "action8",
391      "op": "readProperty",
392      "modbusType": "string",
393      "modbusEntity": "inputRegister"
394    }
395  ],
396  "properties": [
397    {
398      "name": "device_id",
399      "title": "device_id",
400      "type": "string",
401      "const": "DeviceID004",
402      "forms": [
403        {
404          "name": "text"
405        }
406      ]
407    }
408  ],
409  "events": [
410    {
411      "name": "grid_voltage",
412      "title": "grid_voltage",
413      "type": "number",
414      "const": 0,
415      "forms": [
416        {
417          "name": "range"
418        }
419      ]
420    }
421  ],
422  "links": [
423    {
424      "name": "link10",
425      "target": "http://example.com/resource10"
426    }
427  ],
428  "forms": [
429    {
430      "name": "form10",
431      "type": "button"
432    }
433  ],
434  "actions": [
435    {
436      "name": "action9",
437      "op": "readProperty",
438      "modbusType": "string",
439      "modbusEntity": "inputRegister"
440    }
441  ],
442  "properties": [
443    {
444      "name": "device_id",
445      "title": "device_id",
446      "type": "string",
447      "const": "DeviceID005",
448      "forms": [
449        {
450          "name": "text"
451        }
452      ]
453    }
454  ],
455  "events": [
456    {
457      "name": "grid_voltage",
458      "title": "grid_voltage",
459      "type": "number",
460      "const": 0,
461      "forms": [
462        {
463          "name": "range"
464        }
465      ]
466    }
467  ],
468  "links": [
469    {
470      "name": "link11",
471      "target": "http://example.com/resource11"
472    }
473  ],
474  "forms": [
475    {
476      "name": "form11",
477      "type": "button"
478    }
479  ],
480  "actions": [
481    {
482      "name": "action10",
483      "op": "readProperty",
484      "modbusType": "string",
485      "modbusEntity": "inputRegister"
486    }
487  ],
488  "properties": [
489    {
490      "name": "device_id",
491      "title": "device_id",
492      "type": "string",
493      "const": "DeviceID006",
494      "forms": [
495        {
496          "name": "text"
497        }
498      ]
499    }
500  ],
501  "events": [
502    {
503      "name": "grid_voltage",
504      "title": "grid_voltage",
505      "type": "number",
506      "const": 0,
507      "forms": [
508        {
509          "name": "range"
510        }
511      ]
512    }
513  ],
514  "links": [
515    {
516      "name": "link12",
517      "target": "http://example.com/resource12"
518    }
519  ],
520  "forms": [
521    {
522      "name": "form12",
523      "type": "button"
524    }
525  ],
526  "actions": [
527    {
528      "name": "action11",
529      "op": "readProperty",
530      "modbusType": "string",
531      "modbusEntity": "inputRegister"
532    }
533  ],
534  "properties": [
535    {
536      "name": "device_id",
537      "title": "device_id",
538      "type": "string",
539      "const": "DeviceID007",
540      "forms": [
541        {
542          "name": "text"
543        }
544      ]
545    }
546  ],
547  "events": [
548    {
549      "name": "grid_voltage",
550      "title": "grid_voltage",
551      "type": "number",
552      "const": 0,
553      "forms": [
554        {
555          "name": "range"
556        }
557      ]
558    }
559  ],
560  "links": [
561    {
562      "name": "link13",
563      "target": "http://example.com/resource13"
564    }
565  ],
566  "forms": [
567    {
568      "name": "form13",
569      "type": "button"
570    }
571  ],
572  "actions": [
573    {
574      "name": "action12",
575      "op": "readProperty",
576      "modbusType": "string",
577      "modbusEntity": "inputRegister"
578    }
579  ],
580  "properties": [
581    {
582      "name": "device_id",
583      "title": "device_id",
584      "type": "string",
585      "const": "DeviceID008",
586      "forms": [
587        {
588          "name": "text"
589        }
590      ]
591    }
592  ],
593  "events": [
594    {
595      "name": "grid_voltage",
596      "title": "grid_voltage",
597      "type": "number",
598      "const": 0,
599      "forms": [
600        {
601          "name": "range"
602        }
603      ]
604    }
605  ],
606  "links": [
607    {
608      "name": "link14",
609      "target": "http://example.com/resource14"
610    }
611  ],
612  "forms": [
613    {
614      "name": "form14",
615      "type": "button"
616    }
617  ],
618  "actions": [
619    {
620      "name": "action13",
621      "op": "readProperty",
622      "modbusType": "string",
623      "modbusEntity": "inputRegister"
624    }
625  ],
626  "properties": [
627    {
628      "name": "device_id",
629      "title": "device_id",
630      "type": "string",
631      "const": "DeviceID009",
632      "forms": [
633        {
634          "name": "text"
635        }
636      ]
637    }
638  ],
639  "events": [
640    {
641      "name": "grid_voltage",
642      "title": "grid_voltage",
643      "type": "number",
644      "const": 0,
645      "forms": [
646        {
647          "name": "range"
648        }
649      ]
650    }
651  ],
652  "links": [
653    {
654      "name": "link15",
655      "target": "http://example.com/resource15"
656    }
657  ],
658  "forms": [
659    {
660      "name": "form15",
661      "type": "button"
662    }
663  ],
664  "actions": [
665    {
666      "name": "action14",
667      "op": "readProperty",
668      "modbusType": "string",
669      "modbusEntity": "inputRegister"
670    }
671  ],
672  "properties": [
673    {
674      "name": "device_id",
675      "title": "device_id",
676      "type": "string",
677      "const": "DeviceID010",
678      "forms": [
679        {
680          "name": "text"
681        }
682      ]
683    }
684  ],
685  "events": [
686    {
687      "name": "grid_voltage",
688      "title": "grid_voltage",
689      "type": "number",
690      "const": 0,
691      "forms": [
692        {
693          "name": "range"
694        }
695      ]
696    }
697  ],
698  "links": [
699    {
700      "name": "link16",
701      "target": "http://example.com/resource16"
702    }
703  ],
704  "forms": [
705    {
706      "name": "form16",
707      "type": "button"
708    }
709  ],
710  "actions": [
711    {
712      "name": "action15",
713      "op": "readProperty",
714      "modbusType": "string",
715      "modbusEntity": "inputRegister"
716    }
717  ],
718  "properties": [
719    {
720      "name": "device_id",
721      "title": "device_id",
722      "type": "string",
723      "const": "DeviceID011",
724      "forms": [
725        {
726          "name": "text"
727        }
728      ]
729    }
730  ],
731  "events": [
732    {
733      "name": "grid_voltage",
734      "title": "grid_voltage",
735      "type": "number",
736      "const": 0,
737      "forms": [
738        {
739          "name": "range"
740        }
741      ]
742    }
743  ],
744  "links": [
745    {
746      "name": "link17",
747      "target": "http://example.com/resource17"
748    }
749  ],
750  "forms": [
751    {
752      "name": "form17",
753      "type": "button"
754    }
755  ],
756  "actions": [
757    {
758      "name": "action16",
759      "op": "readProperty",
760      "modbusType": "string",
761      "modbusEntity": "inputRegister"
762    }
763  ],
764  "properties": [
765    {
766      "name": "device_id",
767      "title": "device_id",
768      "type": "string",
769      "const": "DeviceID012",
770      "forms": [
771        {
772          "name": "text"
773        }
774      ]
775    }
776  ],
777  "events": [
778    {
779      "name": "grid_voltage",
780      "title": "grid_voltage",
781      "type": "number",
782      "const": 0,
783      "forms": [
784        {
785          "name": "range"
786        }
787      ]
788    }
789  ],
790  "links": [
791    {
792      "name": "link18",
793      "target": "http://example.com/resource18"
794    }
795  ],
796  "forms": [
797    {
798      "name": "form18",
799      "type": "button"
800    }
801  ],
802  "actions": [
803    {
804      "name": "action17",
805      "op": "readProperty",
806      "modbusType": "string",
807      "modbusEntity": "inputRegister"
808    }
809  ],
810  "properties": [
811    {
812      "name": "device_id",
813      "title": "device_id",
814      "type": "string",
815      "const": "DeviceID013",
816      "forms": [
817        {
818          "name": "text"
819        }
820      ]
821    }
822  ],
823  "events": [
824    {
825      "name": "grid_voltage",
826      "title": "grid_voltage",
827      "type": "number",
828      "const": 0,
829      "forms": [
830        {
831          "name": "range"
832        }
833      ]
834    }
835  ],
836  "links": [
837    {
838      "name": "link19",
839      "target": "http://example.com/resource19"
840    }
841  ],
842  "forms": [
843    {
844      "name": "form19",
845      "type": "button"
846    }
847  ],
848  "actions": [
849    {
850      "name": "action18",
851      "op": "readProperty",
852      "modbusType": "string",
853      "modbusEntity": "inputRegister"
854    }
855  ],
856  "properties": [
857    {
858      "name": "device_id",
859      "title": "device_id",
860      "type": "string",
861      "const": "DeviceID014",
862      "forms": [
863        {
864          "name": "text"
865        }
866      ]
867    }
868  ],
869  "events": [
870    {
871      "name": "grid_voltage",
872      "title": "grid_voltage",
873      "type": "number",
874      "const": 0,
875      "forms": [
876        {
877          "name": "range"
878        }
879      ]
880    }
881  ],
882  "links": [
883    {
884      "name": "link20",
885      "target": "http://example.com/resource20"
886    }
887  ],
888  "forms": [
889    {
890      "name": "form20",
891      "type": "button"
892    }
893  ],
894  "actions": [
895    {
896      "name": "action19",
897      "op": "readProperty",
898      "modbusType": "string",
899      "modbusEntity": "inputRegister"
900    }
901  ],
902  "properties": [
903    {
904      "name": "device_id",
905      "title": "device_id",
906      "type": "string",
907      "const": "DeviceID015",
908      "forms": [
909        {
910          "name": "text"
911        }
912      ]
913    }
914  ],
915  "events": [
916    {
917      "name": "grid_voltage",
918      "title": "grid_voltage",
919      "type": "number",
920      "const": 0,
921      "forms": [
922        {
923          "name": "range"
924        }
925      ]
926    }
927  ],
928  "links": [
929    {
930      "name": "link21",
931      "target": "http://example.com/resource21"
932    }
933  ],
934  "forms": [
935    {
936      "name": "form21",
937      "type": "button"
938    }
939  ],
940  "actions": [
941    {
942      "name": "action20",
943      "op": "readProperty",
944      "modbusType": "string",
945      "modbusEntity": "inputRegister"
946    }
947  ],
948  "properties": [
949    {
950      "name": "device_id",
951      "title": "device_id",
952      "type": "string",
953      "const": "DeviceID016",
954      "forms": [
955        {
956          "name": "text"
957        }
958      ]
959    }
960  ],
961  "events": [
962    {
963      "name": "grid_voltage",
964      "title": "grid_voltage",
965      "type": "number",
966      "const": 0,
967      "forms": [
968        {
969          "name": "range"
970        }
971      ]
972    }
973  ],
974  "links": [
975    {
976      "name": "link22",
977      "target": "http://example.com/resource22"
978    }
979  ],
980  "forms": [
981    {
982      "name": "form22",
983      "type": "button"
984    }
985  ],
986  "actions": [
987    {
988      "name": "action21",
989      "op": "readProperty",
990      "modbusType": "string",
991      "modbusEntity": "inputRegister"
992    }
993  ],
994  "properties": [
995    {
996      "name": "device_id",
997      "title": "device_id",
998      "type": "string",
999      "const": "DeviceID017",
1000      "forms": [
1001        {
1002          "name": "text"
1003        }
1004      ]
1005    }
1006  ],
1007  "events": [
1008    {
1009      "name": "grid_voltage",
1010      "title": "grid_voltage",
1011      "type": "number",
1012      "const": 0,
1013      "forms": [
1014        {
1015          "name": "range"
1016        }
1017      ]
1018    }
1019  ],
1020  "links": [
1021    {
1022      "name": "link23",
1023      "target": "http://example.com/resource23"
1024    }
1025  ],
1026  "forms": [
1027    {
1028      "name": "form23",
1029      "type": "button"
1030    }
1031  ],
1032  "actions": [
1033    {
1034      "name": "action22",
1035      "op": "readProperty",
1036      "modbusType": "string",
1037      "modbusEntity": "inputRegister"
1038    }
1039  ],
1040  "properties": [
1041    {
1042      "name": "device_id",
1043      "title": "device_id",
1044      "type": "string",
1045      "const": "DeviceID018",
1046      "forms": [
1047        {
1048          "name": "text"
1049        }
1050      ]
1051    }
1052  ],
1053  "events": [
1054    {
1055      "name": "grid_voltage",
1056      "title": "grid_voltage",
1057      "type": "number",
1058      "const": 0,
1059      "forms": [
1060        {
1061          "name": "range"
1062        }
1063      ]
1064    }
1065  ],
1066  "links": [
1067    {
1068      "name": "link24",
1069      "target": "http://example.com/resource24"
1070    }
1071  ],
1072  "forms": [
1073    {
1074      "name": "form24",
1075      "type": "button"
1076    }
1077  ],
1078  "actions": [
1079    {
1080      "name": "action23",
1081      "op": "readProperty",
1082      "modbusType": "string",
1083      "modbusEntity": "inputRegister"
1084    }
1085  ],
1086  "properties": [
1087    {
1088      "name": "device_id",
1089      "title": "device_id",
1090      "type": "string",
1091      "const": "DeviceID019",
1092      "forms": [
1093        {
1094          "name": "text"
1095        }
1096      ]
1097    }
1098  ],
1099  "events": [
1100    {
1101      "name": "grid_voltage",
1102      "title": "grid_voltage",
1103      "type": "number",
1104      "const": 0,
1105      "forms": [
1106        {
1107          "name": "range"
1108        }
1109      ]
1110    }
1111  ],
1112  "links": [
1113    {
1114      "name": "link25",
1115      "target": "http://example.com/resource25"
1116    }
1117  ],
1118  "forms": [
1119    {
1120      "name": "form25",
1121      "type": "button"
1122    }
1123  ],
1124  "actions": [
1125    {
1126      "name": "action24",
1127      "op": "readProperty",
1128      "modbusType": "string",
1129      "modbusEntity": "inputRegister"
1130    }
1131  ],
1132  "properties": [
1133    {
1134      "name": "device_id",
1135      "title": "device_id",
1136      "type": "string",
1137      "const": "DeviceID020",
1138      "forms": [
1139        {
1140          "name": "text"
1141        }
1142      ]
1143    }
1144  ],
1145  "events": [
1146    {
1147      "name": "grid_voltage",
1148      "title": "grid_voltage",
1149      "type": "number",
1150      "const": 0,
1151      "forms": [
1152        {
1153          "name": "range"
1154        }
1155      ]
1156    }
1157  ],
1158  "links": [
1159    {
1160      "name": "link26",
1161      "target": "http://example.com/resource26"
1162    }
1163  ],
1164  "forms": [
1165    {
1166      "name": "form26",
1167      "type": "button"
1168    }
1169  ],
1170  "actions": [
1171    {
1172      "name": "action25",
1173      "op": "readProperty",
1174      "modbusType": "string",
1175      "modbusEntity": "inputRegister"
1176    }
1177  ],
1178  "properties": [
1179    {
1180      "name": "device_id",
1181      "title": "device_id",
1182      "type": "string",
1183      "const": "DeviceID021",
1184      "forms": [
1185        {
1186          "name": "text"
1187        }
1188      ]
1189    }
1190  ],
1191  "events": [
1192    {
1193      "name": "grid_voltage",
1194      "title": "grid_voltage",
1195      "type": "number",
1196      "const": 0,
1197      "forms": [
1198        {
1199          "name": "range"
1200        }
1201      ]
1202    }
1203  ],
1204  "links": [
1205    {
1206      "name": "link27",
1207      "target": "http://example.com/resource27"
1208    }
1209  ],
1210  "forms": [
1211    {
1212      "name": "form27",
1213      "type": "button"
1214    }
1215  ],
1216  "actions": [
1217    {
1218      "name": "action26",
1219      "op": "readProperty",
1220      "modbusType": "string",
1221      "modbusEntity": "inputRegister"
1222    }
1223  ],
1224  "properties": [
1225    {
1226      "name": "device_id",
1227      "title": "device_id",
1228      "type": "string",
1229      "const": "DeviceID022",
1230      "forms": [
1231        {
1232          "name": "text"
1233        }
1234      ]
1235    }
1236  ],
1237  "events": [
1238    {
1239      "name": "grid_voltage",
1240      "title": "grid_voltage",
1241      "type": "number",
1242      "const": 0,
1243      "forms": [
1244        {
1245          "name": "range"
1246        }
1247      ]
1248    }
1249  ],
1250  "links": [
1251    {
1252      "name": "link28",
1253      "target": "http://example.com/resource28"
1254    }
1255  ],
1256  "forms": [
1257    {
1258      "name": "form28",
1259      "type": "button"
1260    }
1261  ],
1262  "actions": [
1263    {
1264      "name": "action27",
1265      "op": "readProperty",
1266      "modbusType": "string",
1267      "modbusEntity": "inputRegister"
1268    }
1269  ],
1270  "properties": [
1271    {
1272      "name": "device_id",
1273      "title": "device_id",
1274      "type": "string",
1275      "const": "DeviceID023",
1276      "forms": [
1277        {
1278          "name": "text"
1279        }
1280      ]
1281    }
1282  ],
1283  "events": [
1284    {
1285      "name": "grid_voltage",
1286      "title": "grid_voltage",
1287      "type": "number",
1288      "const": 0,
1289      "forms": [
1290        {
1291          "name
```

TMC Contributions further simplified with EdiTDoR

Contribute your TM to a TM Catalog - Metadata

Follow the steps below to contribute your TM to a Catalog specified in the last step

1 2 3

The following fields will be added in the background to your TM for cataloging purposes to ensure quality and discoverability of Thing Models.

Model*
powermeter3000

Author*
EgeKorkan

Manufacturer*
ege inc.

License
URL of the license, e.g., <https://www.apache.org/licenses/LICENSE-2.0.txt>

Copyright Year
e.g. 2024...

Copyright Holder
Organization holding the copyright of the TM...

Validate

✓ TM is valid

Click to copy the full Thing Model

Close Next

TMC Contributions further simplified with EdiTDoR

Contribute your TM to a TM Catalog - Interaction

Follow the steps below to contribute your TM to a Catalog specified in the last step

1 2 3

If you want to verify the correctness of your Thing Model, you can interact with a device instance here. To do so, please configure the proxy (northbound, southbound, valuepath) and provide instance-specific information such as IP address.

> 2.1 Instance

> 2.2 Gateway

> 2.3 Value Verification

Read property values from device instance

Property Name	Title	Preview Value
device_id	device_id	"7KG7750" ⓘ
serial_number	serial_number	"BF0703100052" ⓘ
creation_date	creation_date	"05082020" ⓘ
battery_failure	battery_failure	false ⓘ
voltage_constant	voltage_constant	false ⓘ
light_bulb_power_in_ampere	light_bulb_power_in_ampere	-0.20791169081775987 ⓘ
battery_charge_in_percent	battery_charge_in_percent	100 ⓘ
grid_voltage	grid_voltage	-0.20791169081775987 ⓘ
statistic_value	statistic_value	1.2534438669959154e-263 ⓘ

Test All Properties

> 2.4 Saving results

Previous Close Next

TMC Contributions further simplified with EdiTDoR

Contribute your TM to a TM Catalog - Submission

Follow the steps below to contribute your TM to a Catalog specified in the last step

Add the TM Catalog Endpoint and Repository URL

TM Catalog Endpoint
http://localhost:8082

Name of the Repository
example

Submit

✓ TM submitted successfully!

Copy TM id

egekorkan/ege-inc/powermeter3000/v0.0.0-20251204121622-f126a0719eeb.tm.json

Open in new tab

http://localhost:8082/thing-models/egekorkan/ege-inc/powermeter3000/v0.0.0-20251204121622-f126a0719eeb.tm.json

Previous **Close**

Demo 2: Authoring Workflow Supported with Eclipse EdiTDor

1. Start EdiTDor locally
2. Start a Thing (we will use a Modbus Thing for the demo)
 1. Start a gateway in this case
3. Write its TM
4. Start Contribution Workflow
5. Optionally test all the endpoints
6. Contribute to the specified catalog

In general, all parameters typed here can be sent via query parameters or stored in browser local storage.

Using the TMC at Siemens

Extract from Meetup 20

Smart Edge Connector runs on an Edge Device

The screenshot shows the Siemens Smart Edge Connector interface. The left sidebar has a 'Things' section selected, and the main area displays a table of provisioned things. The table columns are: Name, Types, Protocol, and Status. There are two entries: 'POC1000' (Thing, Modbus+TCP, Online) and 'Siemens SENTRON PAC4200' (tm:ThingDescription, Modbus, Online). A search bar is at the top of the main area.

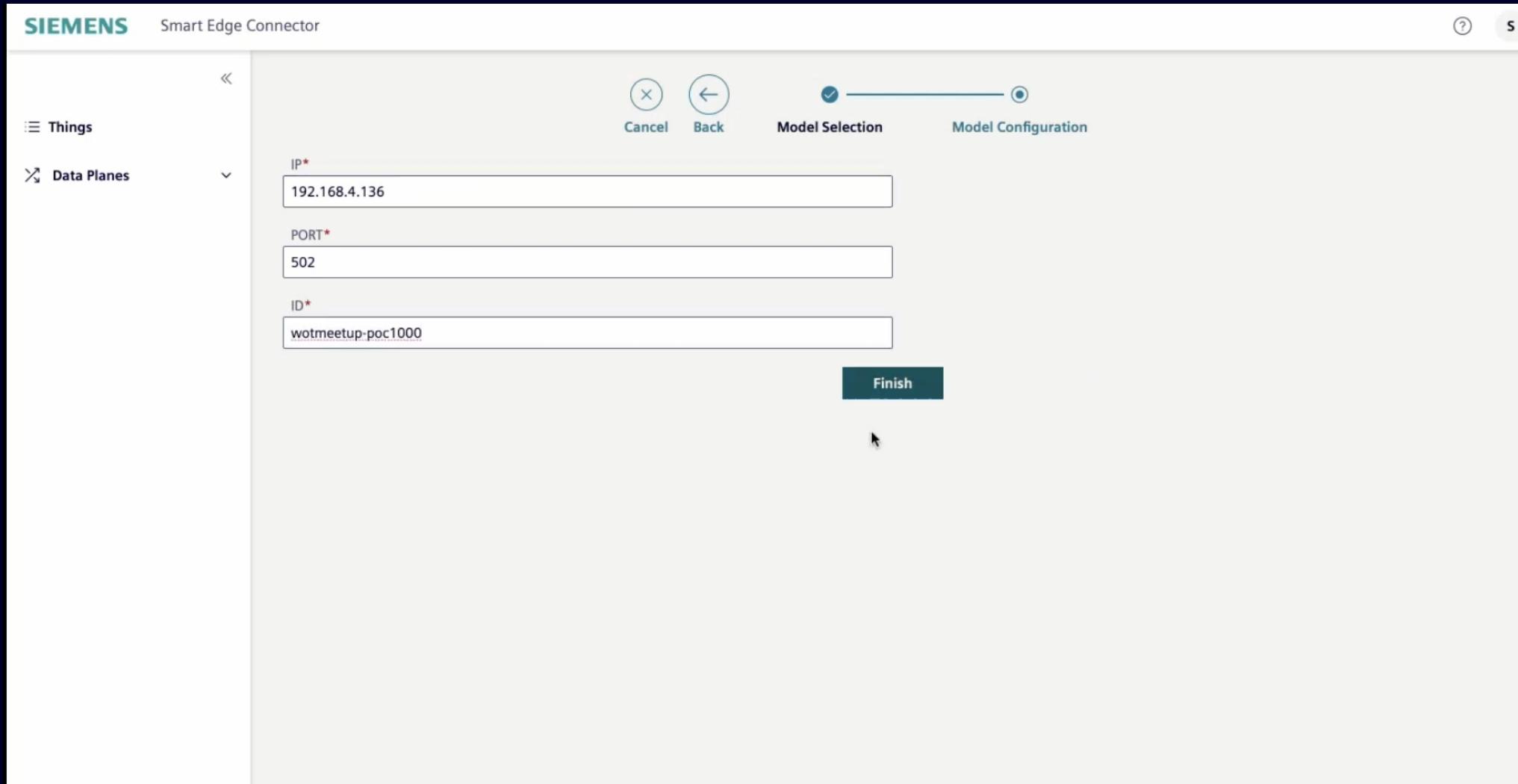
<input type="checkbox"/>	Name	Types	Protocol	Status		
<input type="checkbox"/>	POC1000	Thing	Modbus+TCP	Online		
<input type="checkbox"/>	Siemens SENTRON PAC4200	tm:ThingDescription	Modbus	Online		

We search through a Catalog

The screenshot shows the Siemens Smart Edge Connector interface. On the left, there's a sidebar with 'Things' and 'Data Planes'. The main area has a search bar with 'Manufacturer siemens' typed in. Below it is a table with columns: Manufacturer, Series, and Author. The 'Manufacturer' column contains entries like 'Siemens' and 'siemens'. The 'Series' column contains various product codes such as 'SSL6COM', 'GH180', 'P5320R', etc. The 'Author' column contains names like 'nexus-x', 'siemens', and 'systemx'. There are navigation buttons at the top: 'Cancel', 'Model Selection' (which is selected), 'Model Configuration', and 'Next'.

Manufacturer	Series	Author
Siemens	SSL6COM	nexus-x
Siemens	GH180	nexus-x
Siemens	P5320R	nexus-x
Siemens	PAC2200	nexus-x
Siemens	PAC4200	nexus-x
siemens	3NACOM-FUSE	siemens
siemens	5ST3COM-ASFC	siemens
siemens	5SV6COM-AFDD	siemens
siemens	POC1000	siemens
siemens	7KM2200-2EA30-1DA1	systemx

We instantiate a TD from TM



Device is onboarded!

The screenshot shows the Siemens Smart Edge Connector interface. The top navigation bar includes the Siemens logo, the title "Smart Edge Connector", and a search bar. Below the navigation is a breadcrumb trail: "Devices > POC1000". On the left, there's a sidebar with sections for "Things" and "Data Planes". The main content area has tabs for "Details" (which is selected), "Properties 122", and "Northbound 0". A search bar is at the top of the list table. The table lists ten entries under "Details":

Name	Title	Northbound	Value Preview
idenT_DEVICE_STATUS_UNIT_ID24	Device Status (24)	0	Updated 28/10/2024, 14:30:32
idenT_DEVICE_STATUS_UNIT_ID3	Device Status (3)	3	Updated 28/10/2024, 14:30:32
idenT_DEVICE_STATUS_UNIT_ID4	Device Status (4)	3	Updated 28/10/2024, 14:30:32
idenT_DEVICE_STATUS_UNIT_ID5	Device Status (5)	3	Updated 28/10/2024, 14:30:32
idenT_DEVICE_STATUS_UNIT_ID6	Device Status (6)	3	Updated 28/10/2024, 14:30:32
idenT_DEVICE_STATUS_UNIT_ID7	Device Status (7)	0	Updated 28/10/2024, 14:30:32
idenT_DEVICE_STATUS_UNIT_ID8	Device Status (8)	0	Updated 28/10/2024, 14:30:32
idenT_DEVICE_STATUS_UNIT_ID9	Device Status (9)	0	Updated 28/10/2024, 14:30:32
idenT_FW_COM	FW Version Communication Contr...	"\u0002\u0000\u0002\u0000..."	Updated 28/10/2024, 14:30:32

Each row includes edit and delete icons. The "Properties" tab shows 122 items, and the "Northbound" tab shows 0.

CG Challenge

Hacktoberfest, Advent of Code, now...

We will host a small event for the community to write TMs for the IoT devices around themselves.

As TMC can pull TMs from different repositories, you don't even need to commit them to the CG repository. We will just run a simple bash script to pull your repositories at the end of the challenge.

Instructions, also on [our repo](#):

- Create an empty repository under your own GitHub username
- Install TMC CLI and create a new repository
- Write/generate a TM however you like
- Do `tmc import yourTM.json`
- Your repository will be updated with the TM, table of contents.
- Push the changes
- Create a PR to the CG repository with the `tmc repo add` commands
 - Issues or Discord messages are also fine! We will add a new line with your repo.
- That's it!

My challenge ☺
Tado X Thermostat with Matter



Contact

Ege Korkan
SI CTO SSA
ege.korkan@siemens.com