W3C Thing Description Template

Status & SDF Comparison

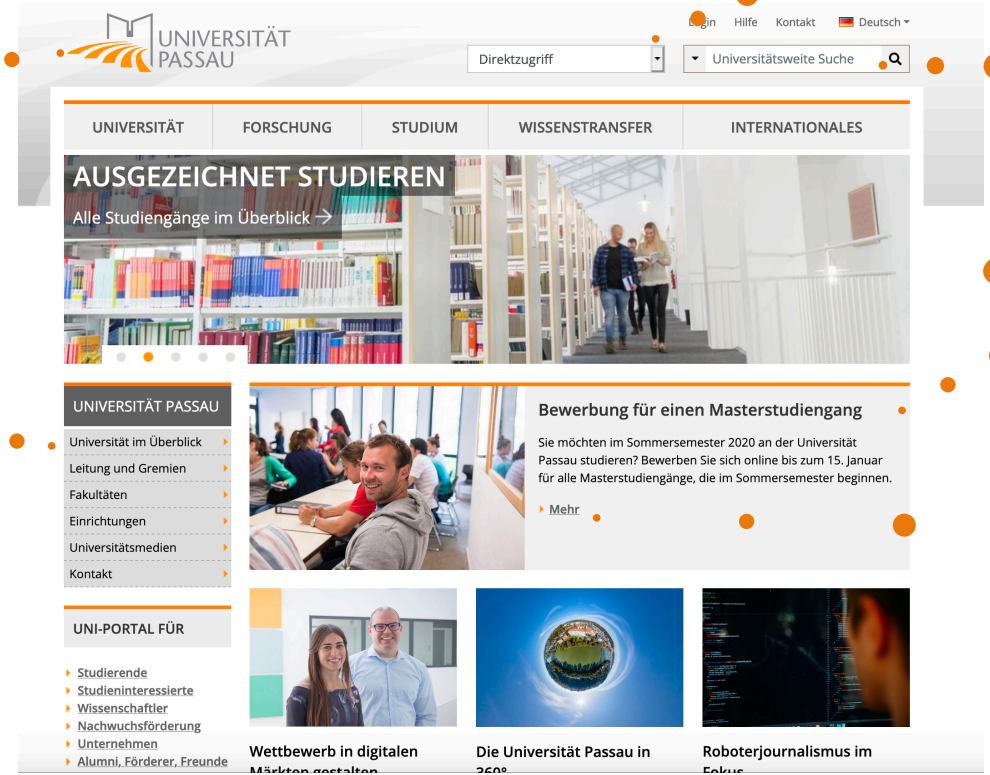
Sebastian Käbisch, WISHI Online Meeting, April 24th, 2020

Websites are intended for Humans
Typically, we know *what* we get and *how*!

Select something

Context of website

Enter data in a web form



Content / Information

Get more information

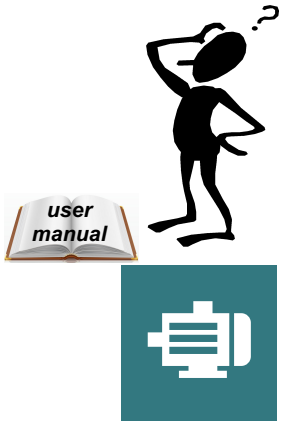
Get more information

How about *Things*?

What kind of data do you serve?

Who are you?

How does the payload structure look like?



Are there some context information (e.g., kind of actuator/sensor, unit)?

How can I access the data/function?

What kind of functions do you have?

What kind of protocols & serializations do you support?

Are there some security constrains?

Do you have other relations to other Things?

Example I/II

5.4.5 Modbus TCP-Port konfigurierbar

Ports sind Kommunikationskanäle, die es ermöglichen, über ein Netzwerk auf ein Modbusfähiges Gerät zuzugreifen.

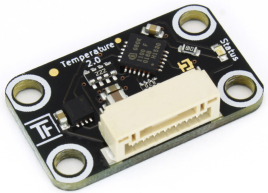
Standard IP-Ports wie Port 502 werden von Port-Scannern oft überprüft. Wird ein offener Port von einem Angreifer entdeckt, kann das Gerät über diesen offenen Port angegriffen werden.

Das PAC2200 bietet die Möglichkeit, die Modbus TCP-Ports manuell zu konfigurieren. Das Umschalten von Standard-Port 502 auf einen benutzerdefinierten Port erschwert die Suche nach offenen Ports.

File (FC0x14)	Offset Adresse	Adresse FC0x03 FC0x04	Anzahl der Register	Name	Format	Einheit	Wertebereich	Zugriff
1	25	30025	2	Scheinleistung L2	float	VA		R
1	27	30027	2	Scheinleistung L3	float	VA		R
1	29	30029	2	Wirkleistung L1	float	W		R
1	31	30031	2	Wirkleistung L2	float	W		R
1	33	30033	2	Wirkleistung L3	float	W		R
1	35	30035	2	Blindleistung L1	float	var		R
1	37	30037	2	Blindleistung L2	float	var		R
1	39	30039	2	Blindleistung L3	float	var		R
1	41	30041	2	Leistungsfaktor L1	float	-		R
1	43	30043	2	Leistungsfaktor L2	float	-		R
1	45	30045	2	Leistungsfaktor L3	float	-		R
1	47	30047	2	Frequenz	float	Hz		R
1	49	30049	2	Mittlere Spannung L - N	float	V		R
1	51	30051	2	Mittlere Spannung L- L	float	V		R
1	53	30053	2	Durchschnittsstrom	float	A		R
1	55	30055	2	Gesamtscheinleistung	float	VA		R
1	57	30057	2	Gesamtwirkleistung	float	W		R
1	59	30059	2	Gesamtblindleistung	float	var		R
1	61	30061	2	Gesamtleistungsfaktor	float	-		R
1	257	30257	2	Zeitstempel	unix_ts	-		R
1	259	30259	2	Flags	uint32_t	-	0= UNFLAGGED 1= FLAGGED 2= SAG 4= SWELL 8= POWERFAIL	R
1	261	30261	2	max. Spannung L1 - N	float	V		R
1	263	30263	2	max. Spannung L2 - N	float	V		R
1	265	30265	2	max. Spannung L3 - N	float	V		R
1	267	30267	2	max. Spannung L1 - L2	float	V	R	R
1	269	30269	2	max. Spannung L2 - L3	float	V	R	R
1	271	30271	2	max. Spannung L3 - L1	float	V	R	R

Source: https://support.industry.siemens.com/cs/document/109746835/ger%C3%A4tehandbuch-messger%C3%A4t-pac2200?dti=0&lc=de-DE

Example II/II



API

Every function of the C/C++ bindings returns an integer which describes an error code. Data returned from the device, when a getter is called, is handled via output parameters. These parameters are labeled with the `ret_` prefix.

Possible error codes are:

- E_OK = 0
- E_TIMEOUT = -1
- E_NO_STREAM_SOCKET = -2
- E_HOSTNAME_INVALID = -3
- E_NO_CONNECT = -4
- E_NO_THREAD = -5
- E_NOT_ADDED = -6 (unused since C/C++ bindings version 2.0.0)
- E_ALREADY_CONNECTED = -7
- E_NOT_CONNECTED = -8
- E_INVALID_PARAMETER = -9
- E_NOT_SUPPORTED = -10
- E_UNKNOWN_ERROR_CODE = -11
- E_STREAM_OUT_OF_SYNC = -12
- E_INVALID_UID = -13
- E_NON_ASCII_CHAR_IN_SECRET = -14

as defined in `ip_connection.h`.

All functions listed below are thread-safe.

Basic Functions

void **`temperature_v2_create`**(TemperatureV2 *temperature_v2, const char *uid, IPConnection *ipcon)

Parameters:	**temperature_v2** – Type: TemperatureV2 * **uid** – Type: const char * **ipcon** – Type: IPConnection *

Creates the device object `temperature_v2` with the unique device ID `uid` and adds it to the IPConnection `ipcon`:

```
TemperatureV2 temperature_v2;
temperature_v2_create(&temperature_v2, "YOUR_DEVICE_UID", &ipcon);
```

```c
#include <stdio.h>

#include "ip_connection.h"
#include "bricklet_temperature_v2.h"

#define HOST "localhost"
#define PORT 4223
#define UID "XYZ" // Change XYZ to the UID of your Temperature Bricklet 2.0

int main(void) {
    // Create IP connection
    IPConnection ipcon;
    ipcon_create(&ipcon);

    // Create device object
    TemperatureV2 t;
    temperature_v2_create(&t, UID, &ipcon);

    // Connect to brickd
    if(ipcon_connect(&ipcon, HOST, PORT) < 0) {
        fprintf(stderr, "Could not connect\n");
        return 1;
    }
    // Don't use device before ipcon is connected

    // Get current temperature
    int16_t temperature;
    if(temperature_v2_get_temperature(&t, &temperature) < 0) {
        fprintf(stderr, "Could not get temperature, probably timeout\n");
        return 1;
    }

    printf("Temperature: %f °C\n", temperature/100.0);

    printf("Press key to exit\n");
    getchar();
    temperature_v2_destroy(&t);
    ipcon_destroy(&ipcon); // Calls ipcon_disconnect internally
    return 0;
}
```


Source: https://www.tinkerforge.com/en/doc/Software/Bricklets/TemperatureV2_Bricklet_C.html

The WoT Thing Description

The “index.html” for Things – A common language based on JSON-LD / RDF

What kind of data do you serve? ✓

Who are you? ✓

How does the payload structure look like? ✓

Are there some context information (e.g., kind of actuator/sensor, unit)? ✓

How can I access the data/function? ✓

What kind of functions do you have? ✓

What kind of protocols & serializations do you support? ✓

Are there some security constrains? ✓

Do you have other relations to other Things? ✓

The WoT Thing Description

Reuse existing domain knowledge and concepts

What kind of data do you serve? ✓

Who are you? ✓

How does the payload structure look like? ✓

Are there some context information (e.g., kind of actuator/sensor, unit)? ✓

How can I access the data/function? ✓

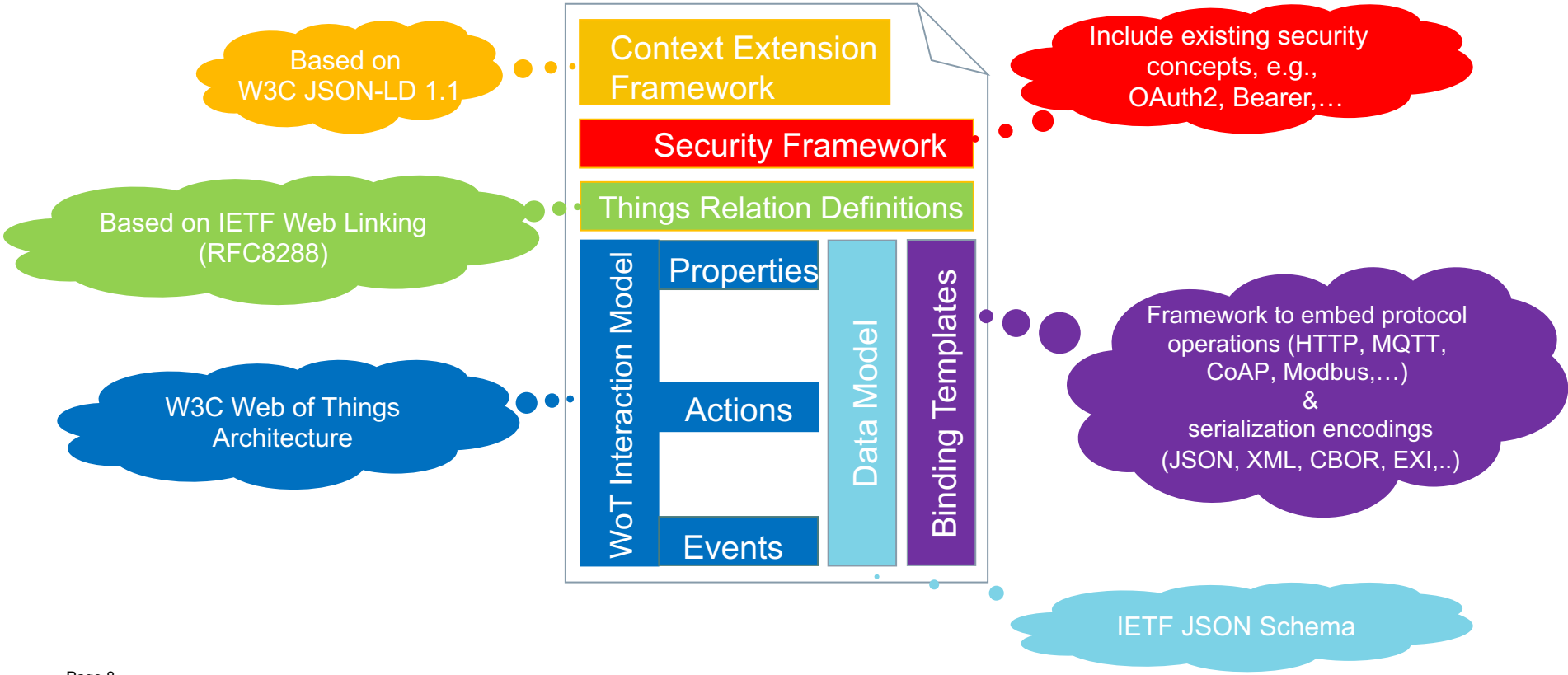
What kind of functions do you have? ✓

What kind of protocols & serializations do you support? ✓

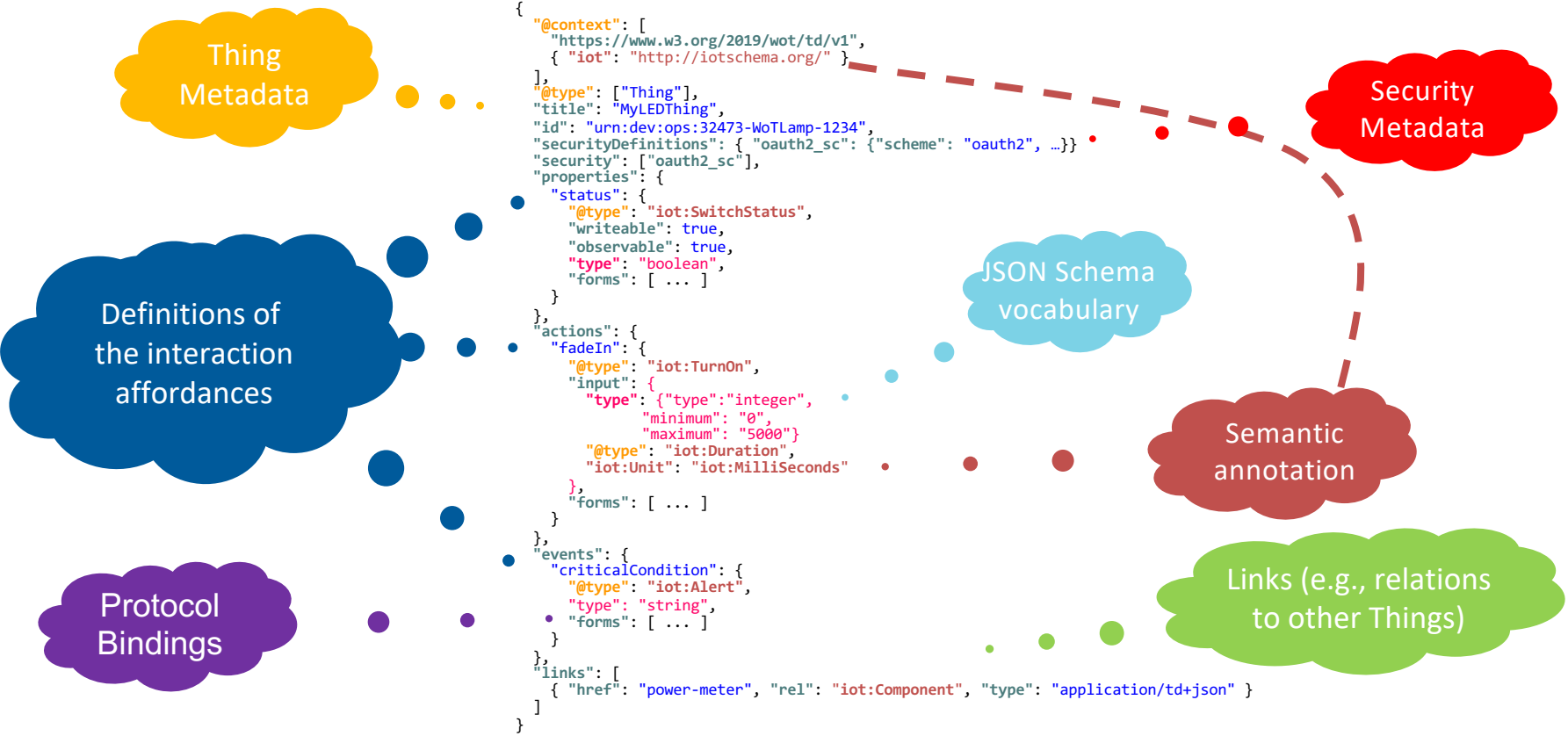
Are there some security constrains? ✓

Do you have other relations to other Things? ✓

Standard Technology Components in WoT Thing Description



WoT Thing Description – JSON-LD based Document Format



WoT Binding Templates –
Uniform Documentation of IoT Protocols (HTTP, MQTT, CoAP, …)

HTTP

```
"properties": {
      ...
      "forms": [
        {
          "op": "readproperty",
          "href": "https://myled.example.com:8080/livingroom/lamp/status",
          "contentType": "application/json",
          "htv:methodName": "GET"
        }
        …
```

MQTT

```
"events": {
      ...
      "forms": [
        {
          "op": "subscribeevent",
          "href": "mqtt://mybroker.example.com:1883/livingroom/lamp/criticalCond",
          "contentType": "application/json",
          "mqv:controlPacketValue": "SUBSCRIBE"
        }
        …
```

MQTT Broker address

MQTT topic

CoAP

```
"actions": {
      ...
      "forms": [
        {
          "op": "invokeaction",
          "href": "coaps://myled.example.com:5684/lr/1/fi",
          "contentType": "application/ocf+cbor",
          "cov:methodName": "POST",
          "cov:options": [ {
            "cov:optionNumber": 2053,          /// OCF-Content-Format-Version
            "cov:optionValue": "1.1.0"
          } ]
```

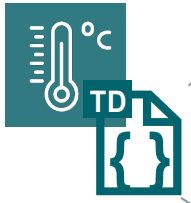
CoAP header settings

…

WebSocket, Modbus, BACnet, OPC-UA, NETCONF, …

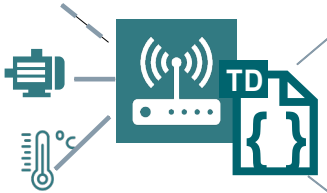
…

Describe *any* Things: Legacy, Edge, Digital Twin, …
Describe what is there!

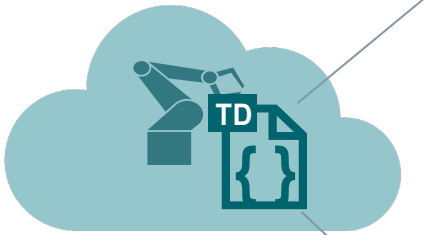



```
{
 "@context": […],
 "id": "urn:dev:ops:13473-temp-12",
 "title": "Temperature",
 "security": { "scheme": "basic" },

 ….
 "events" : {
  "temp": {
     "data" : {    "type": "number",
                   "minimum": -40.2,
                   "maximum": 48.4 },
     "unit": "Celsius",
     "forms": [{
       "href": "mqtt://192.168.0.1:8883/temp/value",
       "contentType": "application/json"
      }]}}}
```


```
{
 "@context": […],
 "id": "urn:dev:ops:42473-engine-12",
 "title": "Engine",
 "security": { "scheme": "basic" },

 ….
 "properties" : {
  "status": {
        …
        "forms": [{
            "href":"modbus+tcp://192.168.0.2:502 ...}]},
   "speed": {...}
 },
 "actions": {…}

 "events": {…}
}
```




```
{
  "@context": […],
  "id": "urn:dev:ops:12473-iot2000",
  "title": "MyEdge",
  "security": { "scheme": "oatuh2" },

   ….
  "properties" : {
    "temp": {...},
    "speed": {...}
  }
}
```




```
{
  "@context": […],
  "@type": "Thing",
  "id": "urn:dev:ops:32473-robot-1",
  "title": "Robotic Arm",
  "security": { "scheme": "oatuh2" },

   ….
 "properties" : {
   "status": {...}
   …
 },
 "actions": {
   "openGrip": {...},
   "closeGrip": {...},
   "moveGripTo": {...}
  }}
```

Thing Description Template – Current Status

A *Thing Description Template* is a description for a *class* of Things. It describes the properties, actions, events and common metadata that are shared for an entire group of Things, to enable the common handling of thousands of devices by a cloud server, which is not practical on a per-Thing basis. The Thing Description Template uses the same core vocabulary and information model from § 5. TD Information Model.

EXAMPLE 36: MyLampThingTemplate serialized in JSON

```
{
    "@context": ["https://www.w3.org/2019/wot/td/v1"],
    "@type" : "ThingTemplate",
    "title": "Lamp Thing Description Template",
    "description" : "Lamp Thing Description Template",
    "properties": {
        "status": {
            "description" : "current status of the lamp (on|off)",
            "type": "string",
            "readOnly": true
        }
    },
    "actions": {
        "toggle": {
            "description" : "Turn the lamp on or off"
        }
    },
    "events": {
        "overheating": {
            "description" : "Lamp reaches a critical temperature (overheating)",
            "data": {"type": "string"}
        }
    }
}
```

https://www.w3.org/TR/wot-thing-description/#thing-templates

Thing Description Template – Motivation

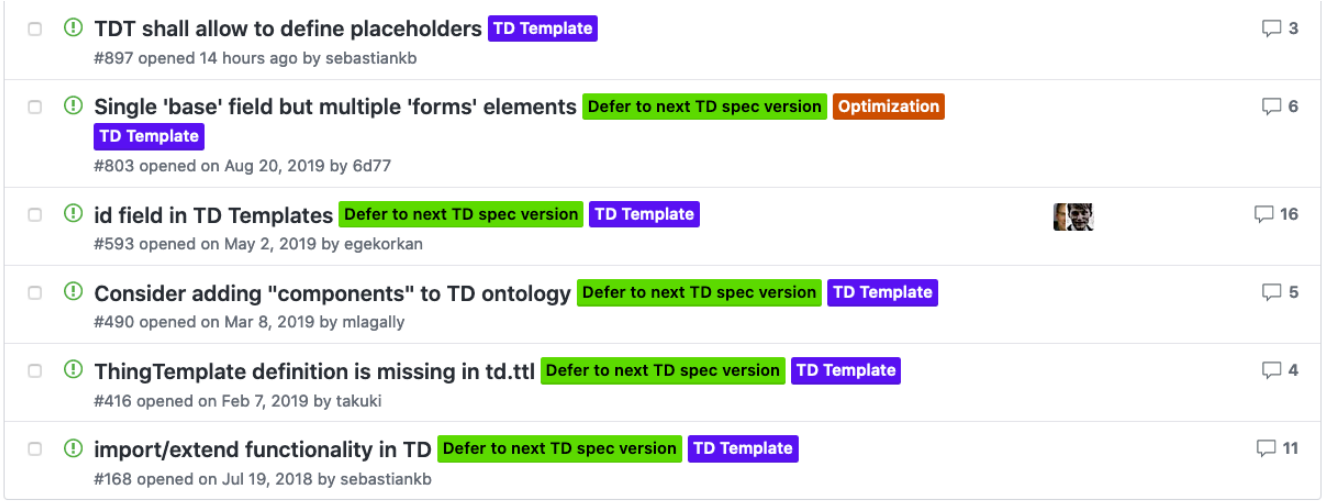
- Mass production of devices
- Digital Twins
- Simulation & Testing
- Combining different models
- …

Formal Definition (Draft)

Arch call 1 on 26.3.:
Proposal:
A Thing Description Template (TDT) is a description for a class of Things that have the same capabilities.
It describes the properties, actions, events and common metadata that are shared for an entire group of Things. A TDT does not contain enough information to identify or interact with a Thing instance.
(potentially add the following: Note: A TD is an instance of one or more TDTs.)

https://github.com/w3c/wot-architecture/issues/454#issuecomment-604272883

Features Requests



https://github.com/w3c/wot-thing-description/labels/TD%20Template

Some Open Questions

- If a TDT is exchanged, is it self-contained or may a client collect all dependencies that may provided in the TDT?

- What is the minimum requirement to be a TDT?

```
{
}
```

or

```
{
"@context": "https://www.w3.org/2019/wot/td/v1"
}
```

or

```
{
"@context": "https://www.w3.org/2019/wot/td/v1",
"title": "TDT example"
}
```

or

```
{
"@context": "https://www.w3.org/2019/wot/td/v1",
"@type" : “TDT"
}
```

TDT & SDF

```
{
  "info": {
    "title": "Example file for ODM Simple JSON Definition Format",
    "version": "20190424",
    "copyright": "Copyright 2019 Example Corp. All rights reserved.",
    "license": "http://example.com/license"
  },
  "namespace": {
    "st": "http://example.com/capability/odm"
  },
  "defaultNamespace": "st",
  "odmObject": {
    "Switch": {
      "odmProperty": {
        "value": {
          "type": "string",
          "enum": [
            "on",
            "off"
          ]
        }
      },
      "odmAction": {
        "on": {},
        "off": {}
      }
```

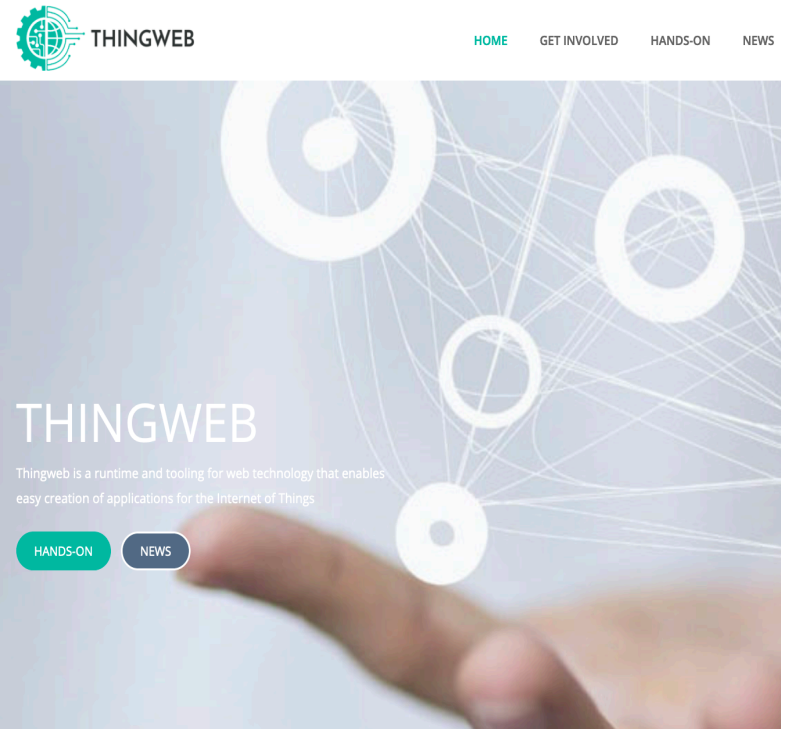
TDT: allows to integrate any metadata. title & version already exists in the TD namespace

TDT: @context from JSON-LD → does the namespace handling

TDT: Seems to be identical with WoT properties; relies on JSON Schema terminology

TDT: Seems to be identical with WoT actions

Thank you for your attention



W3C Web of Things Co-Chair

Dr. Sebastian Kaebisch

Sebastian.Kaebisch@siemens.com

Visit

https://www.w3.org/TR/wot-thing-description/

https://www.w3.org/TR/wot-architecture/

https://www.w3.org/2020/04/pressrelease-wot-rec.html.en

https://github.com/eclipse/thingweb.node-wot/