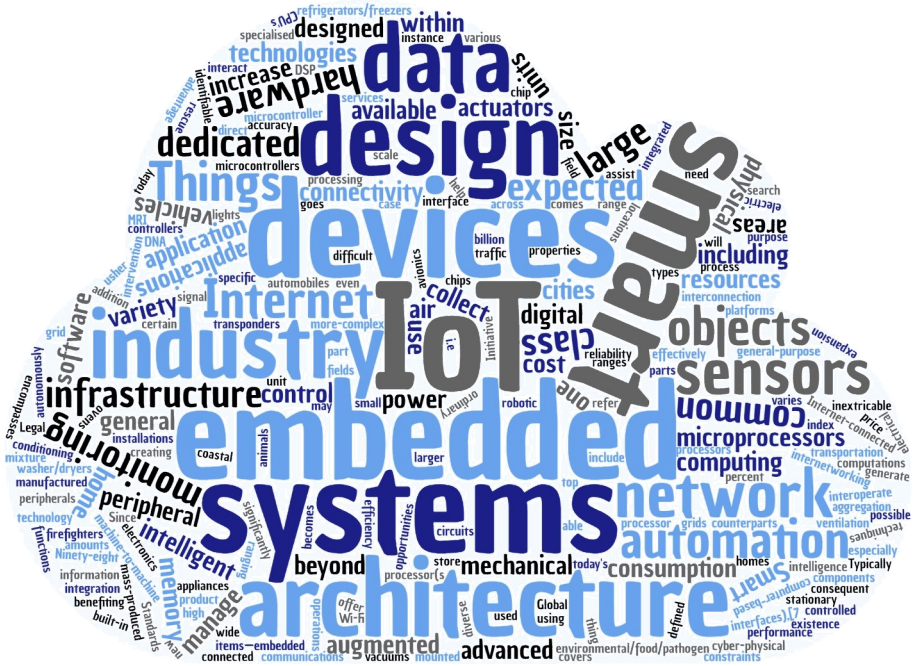
Associate Professorship of Embedded Systems and Internet of Things
Department of Electrical and Computer Engineering
Technical University of Munich

Lecture
IoT Remote Lab

03 – Web of Things

Ege Korkan



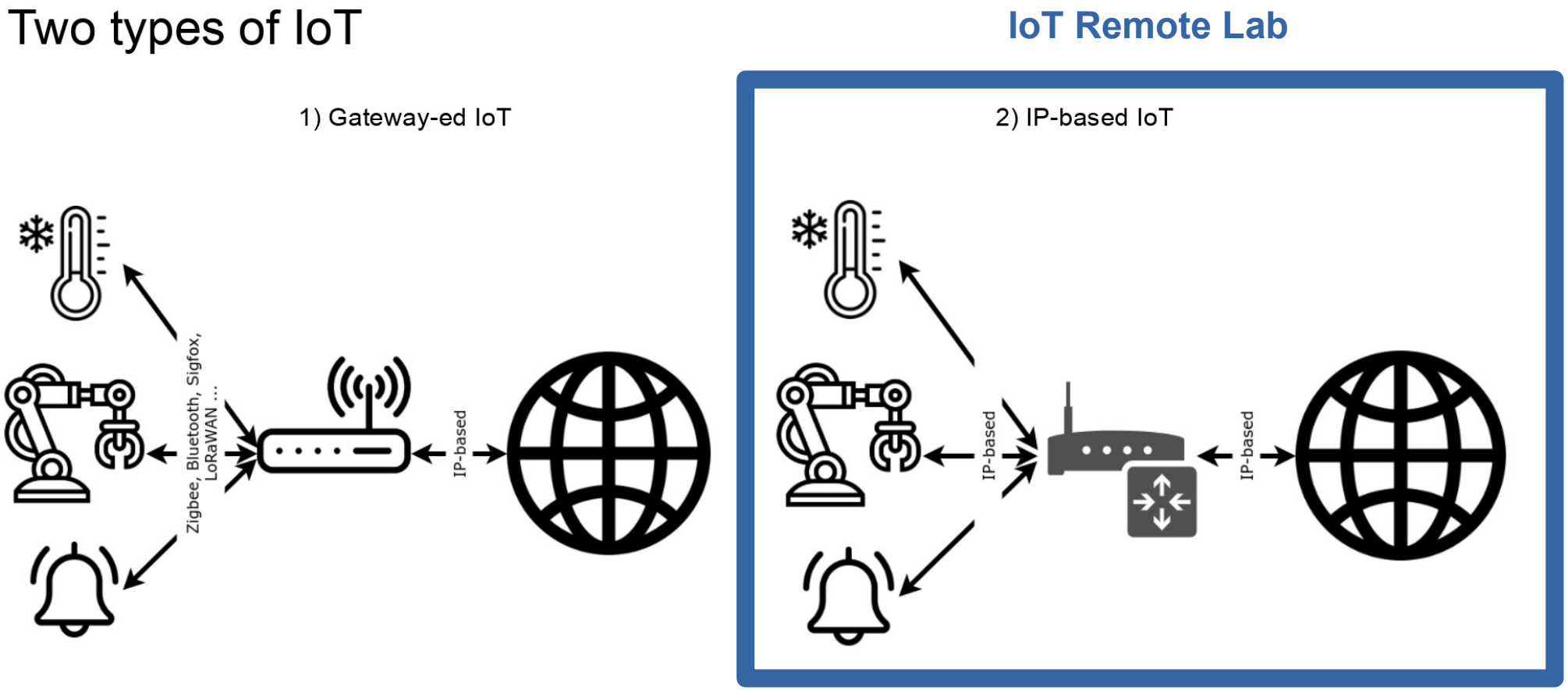
Zoom Guidelines

- The lectures and tutor sessions happen on Zoom meetings following the link sent to you via email.
- Participatition to Zoom sessions is optional
- You can choose a random string for your name
- The Zoom chat will not be recorded and we will not save the chat.
- All the participants except the lecturer is muted. The participants are free to unmute. You can also go to participants, click the hand icon to raise your hand.
- You can also do other things, like asking me to go slower. I have a separate window where I look at the requests from the participants.
- You can ask quick questions in Zoom chat or use the Tweedback link provided in each session.

Tweedback for Real-time Q&A During the Lecture

https://tweedback.de/p3en/chatwall

Recap: IoT Remote Lab



Recap: JSON (application/json)

```json
{
  "productId": 1,
  "productName": "An ice sculpture",
  "price": 12.50,
  "tags": [ "cold", "ice" ],
  "dimensions": {
    "length": 7.0,
    "width": 12.0,
    "height": 9.5
  },
  "warehouseLocation": {
    "latitude": -78.75,
    "longitude": 20.4
  }
}
```

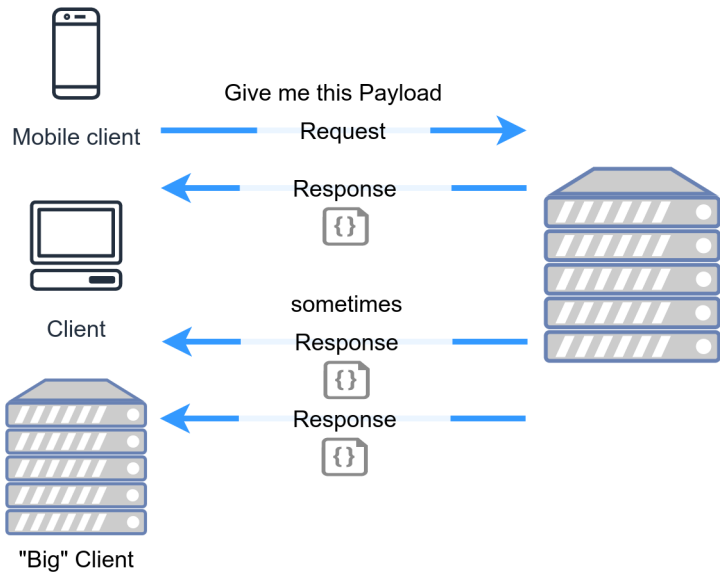
Recap: JSON Schema

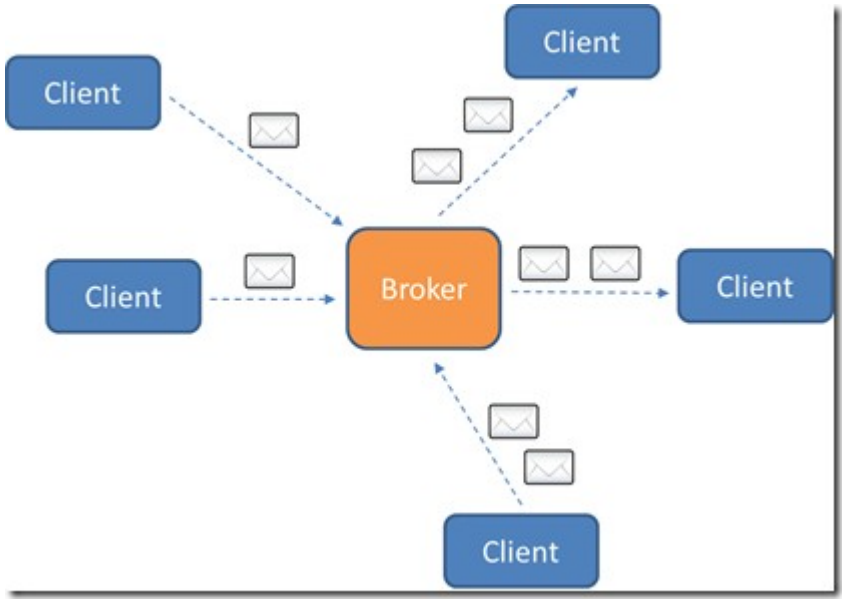
```
{
  "$schema": "http://json-schema.org/draft-07/schema#",
  "$id": "http://example.com/product.schema.json",
  "title": "Product",
  "description": "A product from Acme's catalog",
  "type": "object",
  "properties": {
    "productId": {
      "description": "The unique identifier for a product",
      "type": "integer"
    },
    "productName": {
      "description": "Name of the product",
      "type": "string"
    },
    "price": {
      "description": "The price of the product",
      "type": "number",
      "exclusiveMinimum": 0
    },
    "tags": {
      "description": "Tags for the product",
      "type": "array",
      "items": {
        "type": "string"
      },
      "minItems": 1,
      "uniqueItems": true
    },
```

```
    "dimensions": {
      "type": "object",
      "properties": {
        "length": {
          "type": "number"
        },
        "width": {
          "type": "number"
        },
        "height": {
          "type": "number"
        }
      },
      "required": [ "length", "width", "height" ]
    },
    "warehouseLocation": {
      "description": "Coordinates of the warehouse
      where the product is located.",
      "$ref": "https://example.com/geographical
      -location.schema.json"
    }
  },
  "required": [ "productId", "productName", "price" ]
}
```

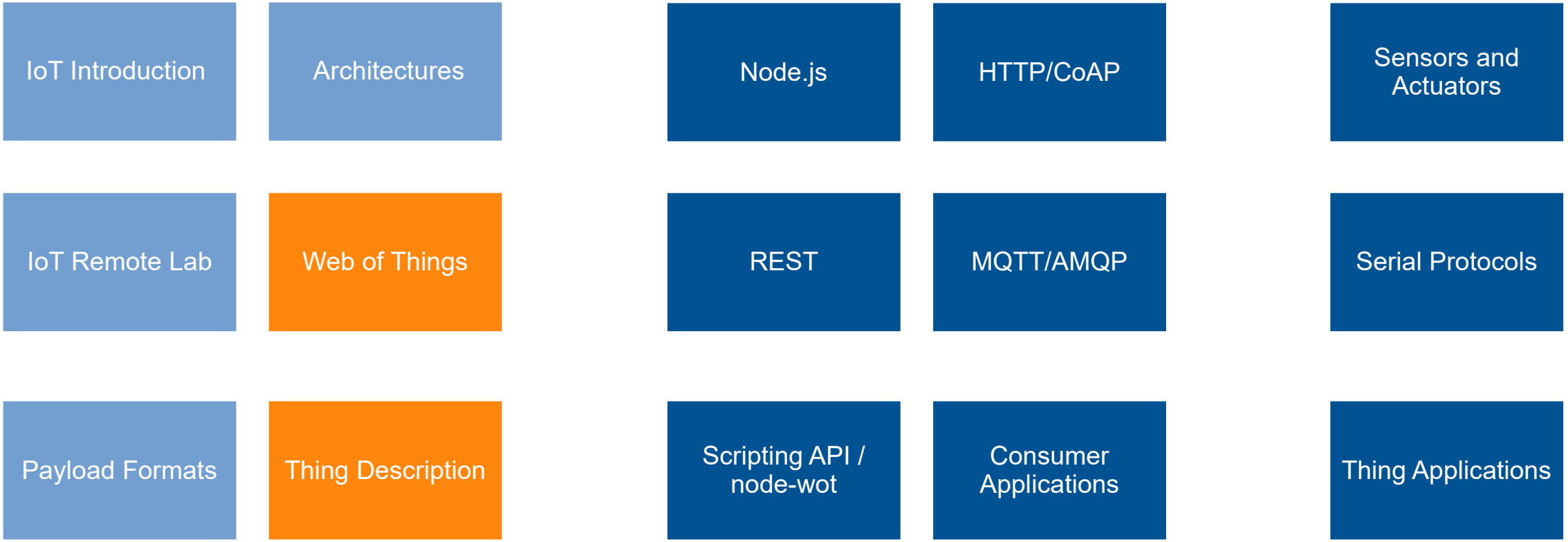
http://json-schema.org/learn/getting-started-step-by-step.html#properties-deeper

Recap: Architectures





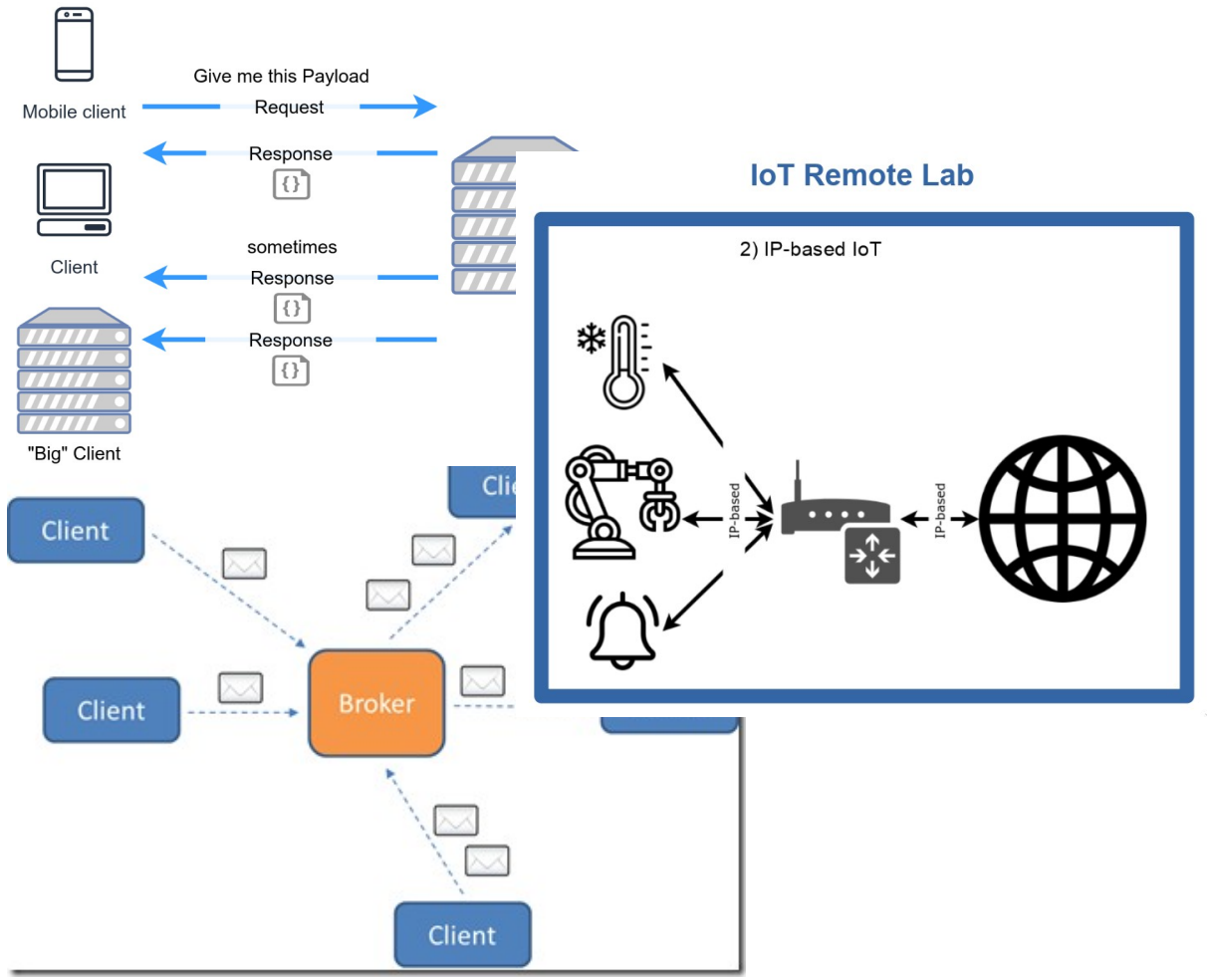
Course Contents



Deliverable 1

- Part 1 of Deliverable 1 is created (or almost online) on Artemis. You will receive an email.
- You should register yourself.
- Deadline of Deliverable 1 is **16.12.2020 23:59 (11:59 pm)**.
- Late submissions will not be accepted at all, you simply lose 30% of your grade.

Web of Things



Web of Things in general

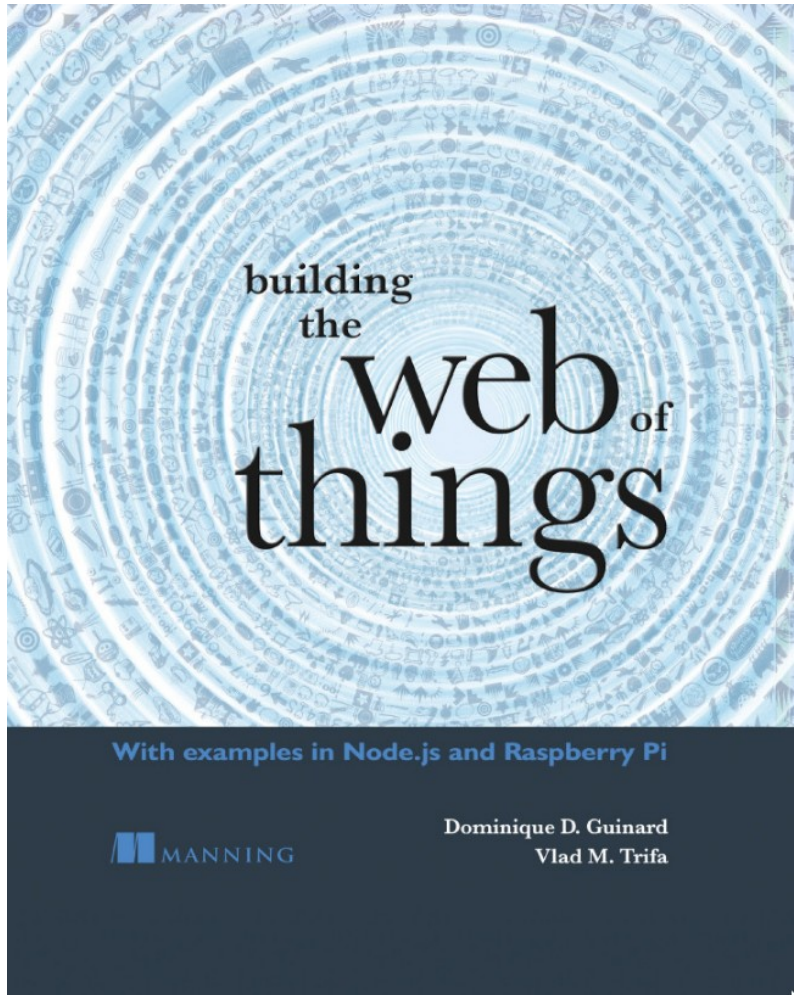
```
{
  "productId": 1,
  "productName": "An ice sculpture",
  "price": 12.50,
  "tags": [ "cold", "ice" ],
  "dimensions": {
    "length": 7.0,
    "width": 12.0,
    "height": 9.5
```

```
"type": "object",
"properties": {
  "productId": {
    "description": "The unique identifier for a product",
    "type": "integer"
  },
  "productName": {
    "description": "Name of the product",
    "type": "string"
  },
  "price": {
    "description": "The price of the product",
    "type": "number",
    "exclusiveMinimum": 0
  },
```

Thing Description

What is Web of Things

- A concept coined around 2008, allowing interacting with devices over the Internet, analogously similar to interacting with *Web* APIs.



Towards the Web of Things: Web Mashups for Embedded Devices

Dominique Guinard
SAP Research Zurich and
Institute for Pervasive Computing
ETH Zurich, Switzerland
dguinard@guinard.org

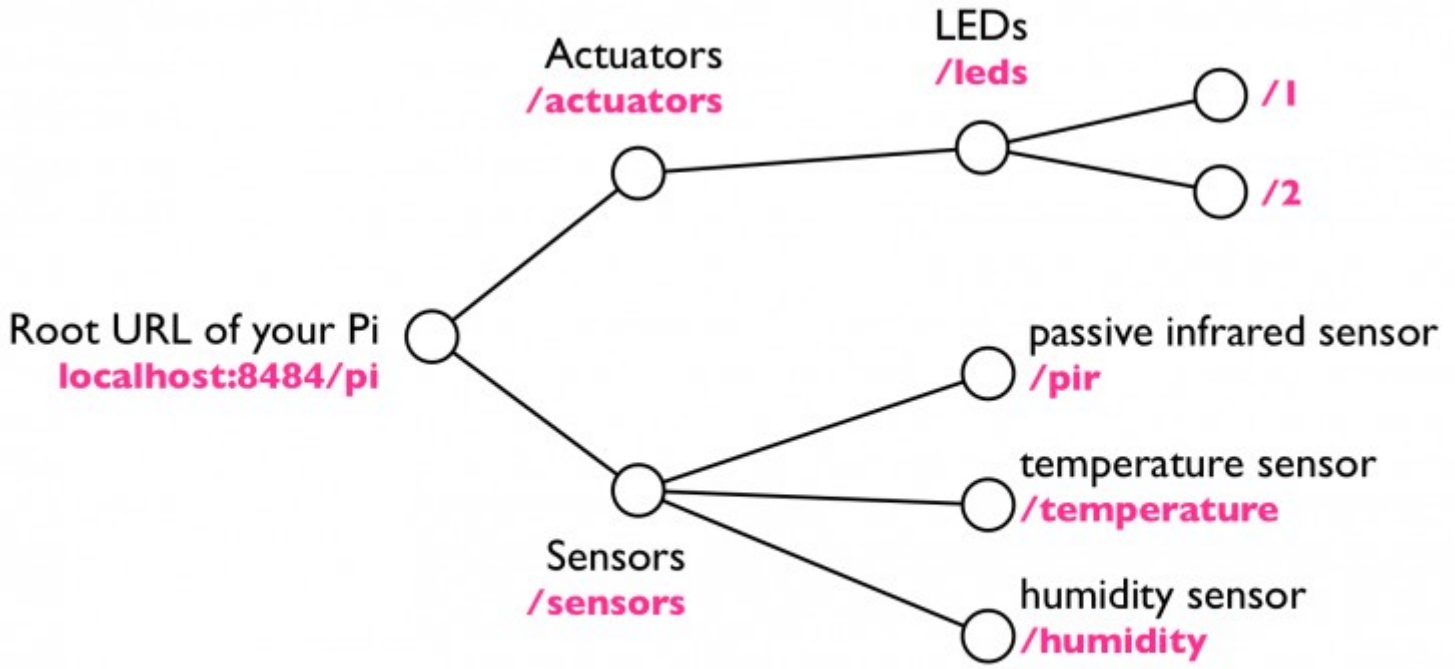
Vlad Trifa
SAP Research Zurich and
Institute for Pervasive Computing
ETH Zurich, Switzerland
vlad.trifa@ieee.org

5 From the Internet of Things to the Web of Things: Resource-oriented Architecture and Best Practices

Dominique Guinard[1,2], Vlad Trifa[1,2], Friedemann Mattern[1], Erik Wilde[3]

[1]Institute for Pervasive Computing, ETH Zurich
[2]SAP Research, Zurich
[3]School of Information, UC Berkeley

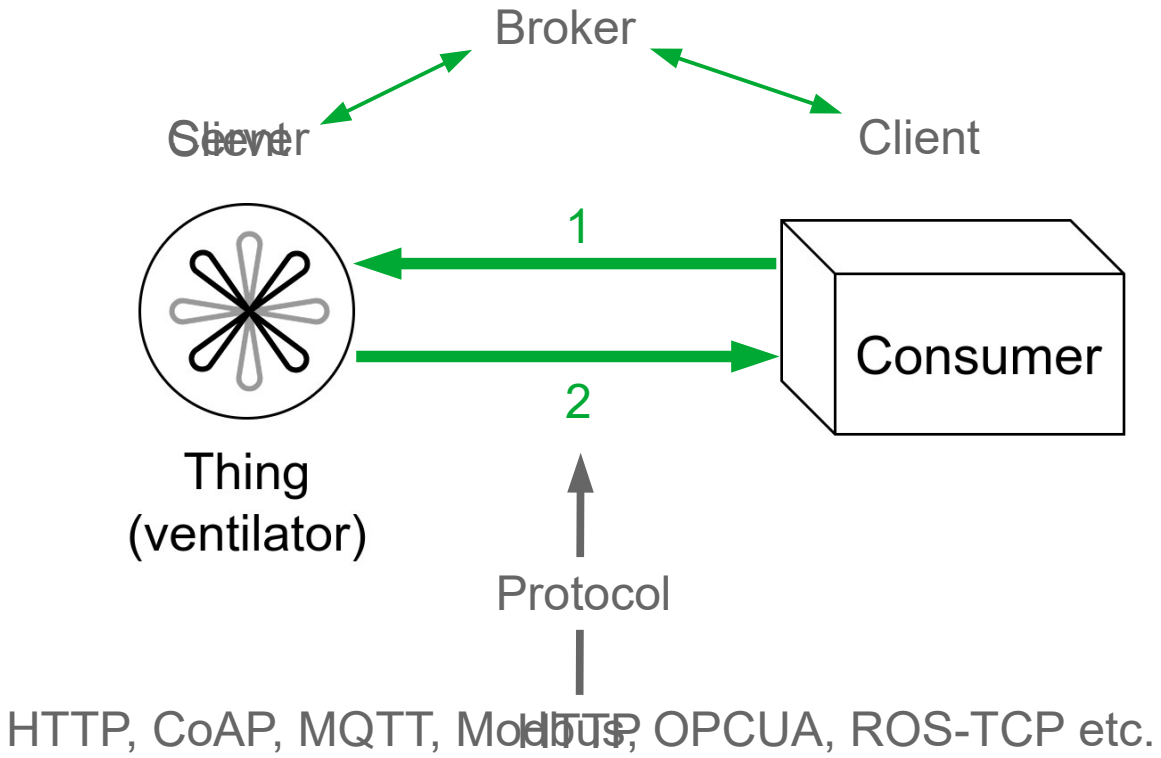
What is Web of Things



Source: Building the Web of Things: book.webofthings.io
Creative Commons Attribution 4.0

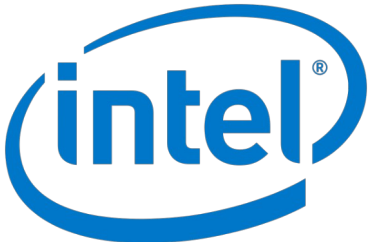
What is Web of Things

- Defining the devices by their interactions over the network



Standardization of WoT in the World Wide Web Consortium (W3C)

- **Architecture**
- **Thing Description**
- Scripting API
- Binding Templates
- *Discovery*

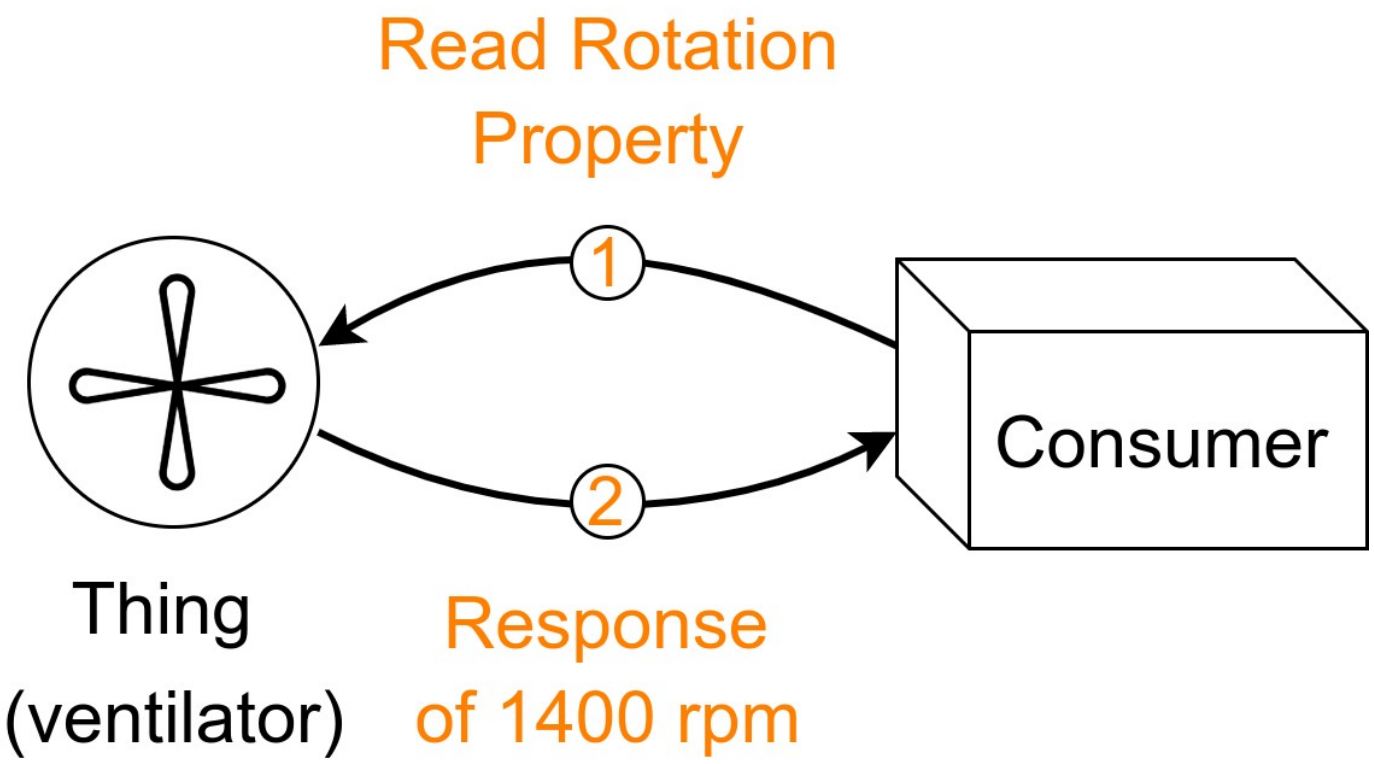


W3C WoT Architecture

- Details different types of use cases and architectures covered, Interaction Affordances, WoT operations.
- *Textual* reference point for other documents

- Interaction Affordances:
  - Property
  - Action
  - Event

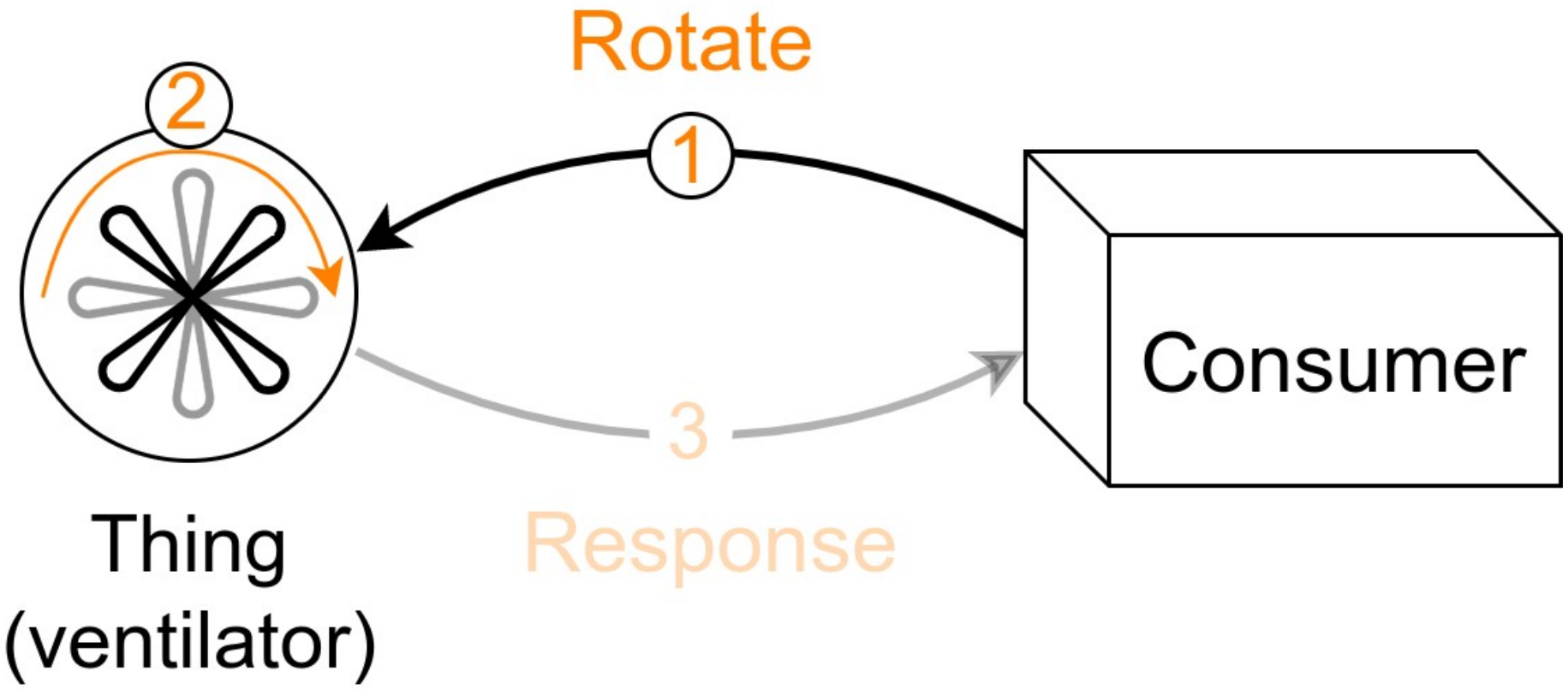
Interaction Affordances: Property



Reading, Writing a Property (readproperty, writeproperty)

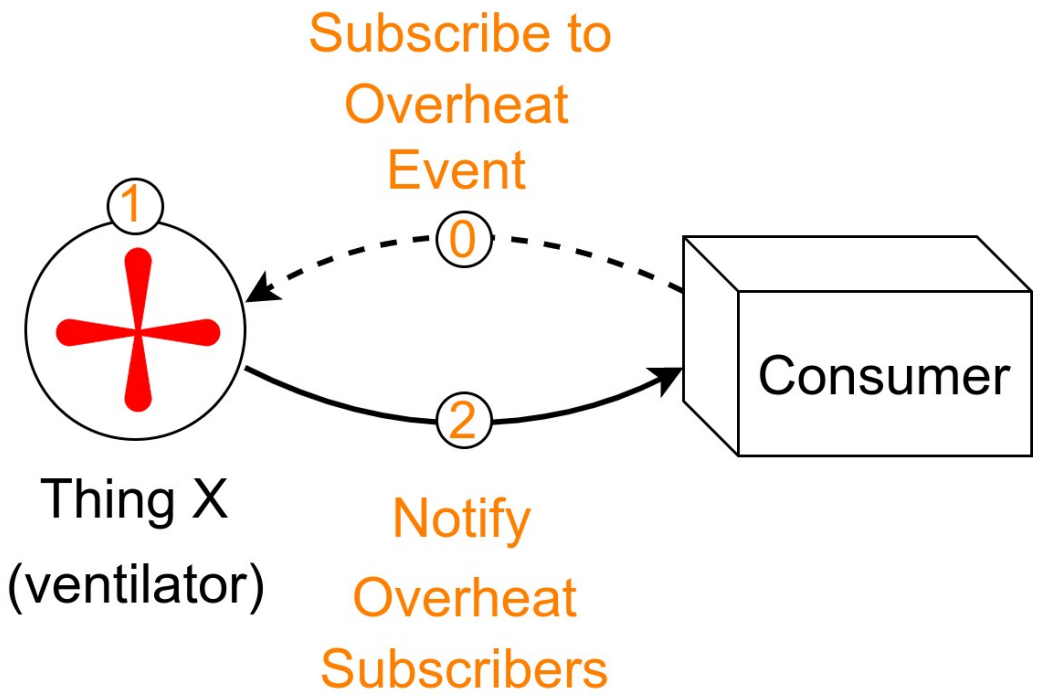
Observing, unobserving a Property (observeproperty, unobserveproperty)

Interaction Affordances: Action



Invoking an Action (invokeaction)

Interaction Affordances: Event



Subscribing, Unsubscribing to/from an Event (subscribeevent, unsubscribeevent)

W3C WoT Thing Description

- A **model** of an **individual** Thing from the Consumer point of view
  - Interaction Affordances
  - Descriptive meta information for humans
  - Security Requirements
  - Versioning
  - Extendable by use of other vocabularies

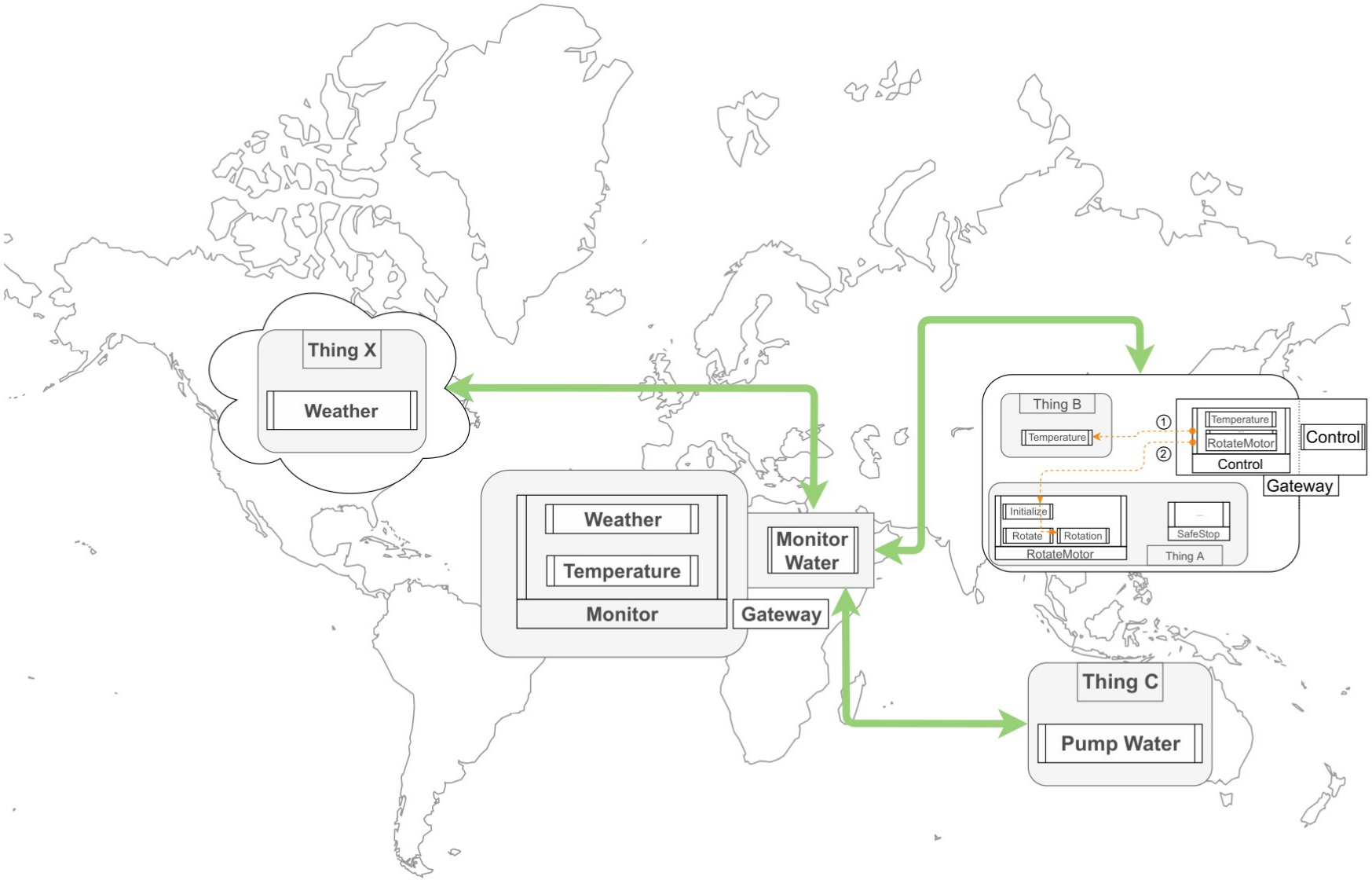
What for?

W3C WoT Thing Description

https://www.youtube.com/watch?v=BxV14h0kFs0

W3C WoT Thing Description

W3C WoT Thing Description



W3C WoT Thing Description

But isn‘t it the same as API descriptions and documentations?

PHILIPS

hue PERSONAL WIRELESS LIGHTING

1.6. Set light state

URL	`/api/<username>/lights/<id>/state`
Method	`PUT`
Version	1.0
Permission	Whitelist

1.6.1. Description

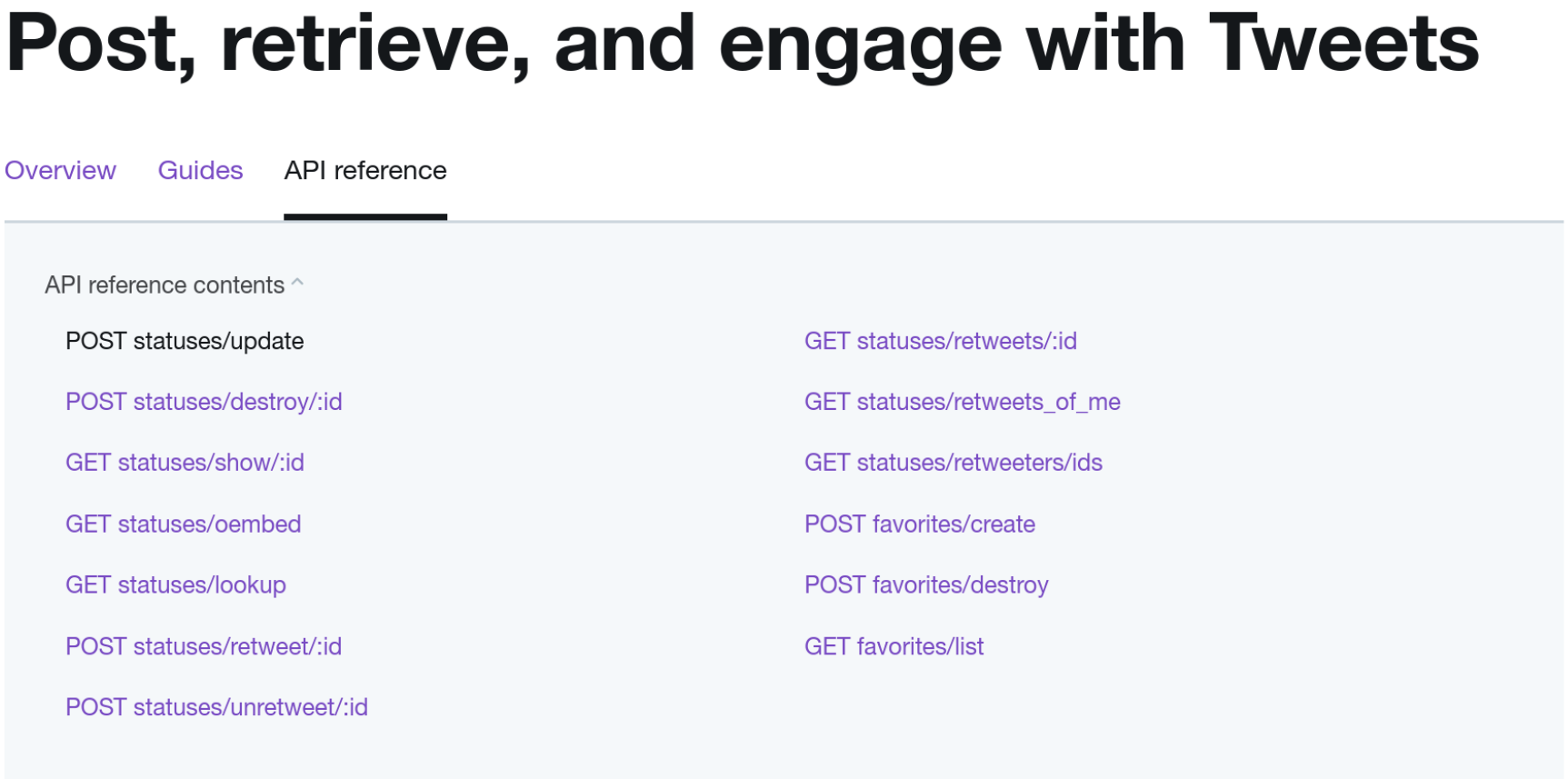
Allows the user to turn the light on and off, modify the hue and effects.

1.6.2. Body arguments

Name	Typre	Description	
on	bool	On/Off state of the light. On=true, Off=false	Optional
bri	uint8	The brightness value to set the light to.Brightness is a scale from 1 (the minimum the light is capable of) to 254 (the maximum). Note: a brightness of 1 is not off. e.g. “brightness”: 60 will set the light to a specific brightness	Optional
hue	uint16	The hue value to set light to.The hue value is a wrapping value between 0 and 65535. Both 0 and 65535 are red, 25500 is green and 46920 is blue. e.g. “hue”: 50000 will set the light to a specific hue.	Optional
sat	uint8	Saturation of the light. 254 is the most saturated (colored) and 0 is the least saturated (white).	Optional

W3C WoT Thing Description

But isn‘t it the same as API descriptions and documentations?



W3C WoT Thing Description

Modbus Functions

Read Device Identification

ID	Name / Description	Type
0x00	VendorName	ASCII String
0x01	ProductCode	ASCII String
0x02	MajorMinorRevision	ASCII String

Example

Default values to be detailed

Request

Slave no.	2B	Type of MEI 0E	ReadDeviceId 01	Object Id 00	CRC16 Lo	CRC16 Hi
1 byte	1 byte	1 byte	1 byte	1 byte	2 bytes	

Response

Slave no.	2B	Type of MEI 0E	ReadDeviceId 01	Degree of conformity 02	-------
1 byte	1 byte	1 byte	1 byte	1 byte	

-------	Number of additional frames 00	Next object Id 00	Number of objects 03	-------
	1 byte	1 byte	1 byte	

-------	Id of object no. 1 00	Length of object no. 1 12	Value of object no. 1 “Schneider Electric”	-------
	1 byte	1 byte	18 bytes	

W3C WoT Thing Description

But isn‘t it the same as API descriptions and documentations? **No**

Classical API documentations are:

- Only human readable (if you can read and understand it)
- Protocol specific
- No-overarching vocabulary accross APIs
- Not instance specific
- Hardly reusable and extendable

W3C WoT Thing Description

```json
{
    "@context": "https://www.w3.org/2019/wot/td/v1",
    "id": "urn:dev:ops:32473-WoTLamp-1234",
    "title": "MyLampThing",
    "securityDefinitions": {
        "basic_sc": {"scheme": "basic", "in":"header"}
    },
    "security": ["basic_sc"],
    "properties": {
        "status" : {
            "type": "string",
            "forms": [{
                "href": "https://mylamp.example.com/status",
                "htv:methodName":"GET"
            }]
        }
    },
    "actions": {
        "toggle" : {
            "forms": [{
                "href": "https://mylamp.example.com/toggle",
                "htv:methodName":"POST"
            }]
        }
    },
    "events":{
        "overheating":{
            "data": {"type": "string"},
            "forms": [{
                "href": "https://mylamp.example.com/oh",
                "htv:methodName":"GET",
                "subprotocol": "longpoll"
            }]
        }
    }
}
```

HTTP based Thing

```json
{
    "@context": [
        "https://www.w3.org/2019/wot/td/v1",
        {"mqv": "http://www.example.org/mqtt-binding#"}
    ],
    "id": "urn:dev:ops:32473-WoTLamp-1234",
    "title": "MyLampThing",
    "securityDefinitions": {
        "nosec_sc": {"scheme": "nosec"}
    },
    "security": ["nosec_sc"],
    "properties": {
        "status" : {
            "type": "string",
            "forms": [{
                "href": "mqtt://mylamp.example.com/status",
                "mqv:controlPacketValue": "SUBSCRIBE"
            }]
        }
    },
    "actions": {
        "toggle" : {
            "forms": [{
                "href": "mqtt://mylamp.example.com/toggle",
                "mqv:controlPacketValue": "PUBLISH"
            }]
        }
    },
    "events":{
        "overheating":{
            "data": {"type": "string"},
            "forms": [{
                "href": "mqtt://mylamp.example.com/oh",
                "mqv:controlPacketValue": "SUBSCRIBE"
            }]
        }
    }
}
```

MQTT based Thing

W3C WoT Thing Description

```
"properties": {
  "coilProp": {
    "title": "coilProp",
    "description": "A coil state",
    "type": "boolean",
    "readOnly": false,
    "writeOnly": false,
    "observable": false,
    "forms": [
      {
        "href": "modbus://127.0.0.1:60000",
        "contentType": "application/octet-stream;length=1",
        "op": [
          "writeproperty"
        ],
        "modbus:function": 5,
        "modbus:range": [0, 1],
        "modbus:unitID": 1
      },
      {
        "href": "modbus://127.0.0.1:60000",
        "contentType": "application/octet-stream;length=1",
        "op": [
          "readproperty"
        ],
        "modbus:function": 1,
        "modbus:range": [0, 1],
        "modbus:unitID": 1
      }
    ]
  }
}
```

Modbus based Thing

W3C WoT Scripting API

- Defines a programming interface for:
  - interacting with Things as Consumer: Can be run in browser or native
  - developing a Thing for Consumer: Must be run native
- It is protocol independent
- Portable
- *Programming language agnostic*

W3C WoT Scripting API

Thing Description

```
{
    "@context": "https://www.w3.org/2019/wot/td/v1",
    "title": "TemperatureController",
    "description":"A temperature controlling device with alarm functionality",
    "securityDefinitions": {
        "basic_sc": {"scheme": "basic", "in":"header"}
    },
    "security": ["basic_sc"],
    "properties": {
        "temperature" : {
            "type": "number",
            "readOnly":true,
            "forms": [{"href": "http://mycontroller.example.com/temperature"}]
        }
    },
    "actions": {
        "increase" : {
            "input":{
                "type":"number"
            },
            "forms": [{"href": "coaps://mycontroller.example.com/inc"}]
        }
    },
    "events":{
        "overheating":{
            "data": {
                "type": "number"
            },
            "forms": [{"href": "mqtt://broker.com/controller/highTemperature"}]
        }
    }
}
```

Control Application over Network

```
WoTHelpers.fetch("coap://temperatureController.com/td").then( async (td) => {
    let thing = await WoT.consume(td);
    let targetTemperature = 28;
    let currentTemperature = await thing.readProperty("temperature");
    if(targetTemperature<currentTemperature){
        await thing.invokeAction("increase",currentTemperature-targetTemperature);
    }
    thing.subscribeEvent("highTemperature",
        x => {
            targetTemperature = 0;
        }
    );
});
```

https://www.youtube.com/watch?v=lt_P2BU8e3I

Let’s dig deeper!

W3C WoT

- Below are the documents we will use/follow:
  - Thing Description Standard: You will always return to it, bookmark it!
  - Architecture: Read it in its entirety once
  - Scripting API: You will always return to it, bookmark it!
  - Binding Templates: Now read quickly, more on it later
- Difference between editor‘s draft and published versions
- Software
  - Interactable virtual Thing
  - TD Playground: To validate Thing Descriptions
  - Reference Scripting API implementation

Now I will go through the standard's web page. Please tell me if I am scrolling too fast.

Web of Things (WoT) Thing Description

W3C Recommendation 9 April 2020

This version:
https://www.w3.org/TR/2020/REC-wot-thing-description-20200409/

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

Latest editor's draft:
https://w3c.github.io/wot-thing-description/

Implementation report:
https://w3c.github.io/wot-thing-description/testing/report.html

Previous version:
https://www.w3.org/TR/2020/PR-wot-thing-description-20200130/

Editors:
Sebastian Kaebisch (Siemens AG)
Takuki Kamiya (Fujitsu Laboratories of America)
Michael McCool (Intel)
Victor Charpenay (Siemens AG)
Matthias Kovatsch (Huawei)

Participate:
GitHub w3c/wot-thing-description
File a bug
Commit history
Pull requests

Contributors:
In the GitHub repository

Repository:
We are on GitHub
File a bug

Separation of Application Logic from Protocols

- Outside of the forms there should not be any information about the protocol or payload format, OR any assumption of it.
- In the forms, there should not be information that has to be understood for the application logic.

```
"actions": {
    "fade" : {
        "title": "Fade in/out",
        "description": "Smooth fade in and out animation.",
        "input": {
            "type": "object",
            "properties": {
                "from": {
                    "type": "integer",
                    "minimum": 0,
                    "maximum": 100
                },
                "to": {
                    "type": "integer",
                    "minimum": 0,
                    "maximum": 100
                },
                "duration": {"type": "number"}
            },
            "required": ["to","duration"],
        },
        "output": {"type": "string"},
        "forms": [...]
    }
}
```

```
"forms": [{
    "op": "invokeaction",
    "href" : "http://mytemp.example.com:5683/temp",
    "contentType": "application/json",
    "htv:methodName": "POST"
}]
```

Complex Forms

- One form, multiple op
- Multiple forms, same protocol, different op
- Multiple forms, same protocol, same op, different URIs
- Multiple forms, same protocol, same op, different contentType
- Multiple forms, different protocols, same op

```
"properties": {
        "brightness": {
            "description": "The current brightness",
            "type": "integer",
            "minimum": -64,
            "maximum": 64,
            "observable": false,
            "forms": [
                {
                    "href": "http://example.org:9191/api/brightness",
                    "op": [
                        "readproperty", "writeproperty"
                    ]
                }
            ]
        }
```

```
"properties": {
        "brightness": {
            "description": "The current brightness setting",
            "type": "integer",
            "minimum": -64,
            "maximum": 64,
            "observable": true,
            "forms": [
                {
                    "href": "http://example.org:9191/api/brightness",
                    "op": ["readproperty"],
                    "htv:methodName": "GET"
                },
                {
                    "href": "http://example.org:9191/api/brightness",
                    "op": ["writeproperty"],
                    "htv:methodName": "POST"
                },
                {
                    "href": "http://example.org:9191/api/brightness/observe",
                    "op": ["observeproperty"],
                    "htv:methodName": "GET",
                    "subProtocol": "longpoll"
                }
            ]
        }
```

```
"properties": {
        "brightness": {
            "description": "The current brightness setting",
            "type": "integer",
            "minimum": -64,
            "maximum": 64,
            "observable": true,
            "forms": [
                {
                    "href": "http://example.org:9191/api/brightness",
                    "op": ["readproperty"],
                    "htv:methodName": "GET"
                },
                {
                    "href": "http://192.168.0.101:8080/api/brightness",
                    "op": ["readproperty"],
                    "htv:methodName": "GET"
                }
            ]
        }
```

```
"properties": {
        "photo": {
            "description": "The photo of the room",
            "forms": [
                {
                    "href": "http://example.org:9191/image",
                    "op": ["readproperty"],
                    "htv:methodName": "GET",
                    "contentType":"image/jpeg"
                },
                {
                    "href": "http://example.org:9191/image",
                    "op": ["readproperty"],
                    "htv:methodName": "GET",
                    "contentType":"image/png"
                }
            ]
        }
```

```
"properties": {
        "brightness": {
            "description": "The current brightness setting",
            "type": "integer",
            "minimum": -64,
            "maximum": 64,
            "observable": true,
            "forms": [
                {
                    "href": "http://example.org:9191/api/brightness",
                    "op": ["readproperty"],
                    "htv:methodName": "GET"
                },
                {
                    "href": "mqtt://192.168.0.101:1883/api/brightness",
                    "op": ["readproperty"],
                    "mqv:ControlPacketValue": "SUBSCRIBE"
                }
            ]
        }
```

Event or Observable Property

- What makes something a Event Affordance and what makes it a Property Affordance that is observable?
  - Complex subsription and unsubscription mechanism → Event
  - Not readable → Event
  - Readable → Property or Event and Property (not clean though)
  - No payload → Event (e.g. buttonPressed)

Web Linking

- Expressing relations to other Things and documents
  - UI of the Thing
  - Documentation pages
  - The location to find the TD
  - The Thing that is my controller
- Possible relation types: https://www.iana.org/assignments/link-relations/link-relations.xhtml

```
"links": [{
    "rel": "controlledBy",
    "href": "https://servient.example.com/things/lampController",
    "type": "application/td+json"
},
{
    "rel": "self",
    "href": "https://servient.example.com/things/lamp/td",
    "type": "application/td+json"
},
{
    "rel": "index",
    "href": "https://servient.example.com/things/lamp/interface",
    "type": "application/html"
}]
```

Semantic Annotations

- How would you describe:
  - Accuracy of your temperature property?
  - Give information that this temperature is of an iron forge?
  - Give information about its static location?

```
{
    "@context": [
        "https://www.w3.org/2019/wot/td/v1",
        {
            "v": "http://www.example.org/versioningTerms#",
            "saref": "https://w3id.org/saref#",
            "om": "http://www.ontology-of-units-of-measure.org/resource/om-2/"
        }
    ],
    "version": {
        "instance": "1.2.1",
        "v:firmware": "0.9.1",
        "v:hardware": "1.0"
    },
    "@type": "saref:TemperatureSensor",
    "properties": {
        "temperature": {
            "description": "Temperature value of the weather station",
            "type": "number",
            "minimum": -32.5,
            "maximum": 55.2,
            "unit": "om:degree_Celsius",
            "forms": [...]
        },
        ...
    },
    ...
}
```

Wrap-Up

- Web of Things is for applications of IoT devices.
- Thing Description gives information about a device regarding its network facing interactions
- Scripting API allows Thing or Consumer to be written in a standardized fashion.

- Additional WoT Introduction Videos:
  - https://www.youtube.com/watch?v=It_P2BU8e3I
  - https://www.youtube.com/watch?v=ySLuAIoz0Dg