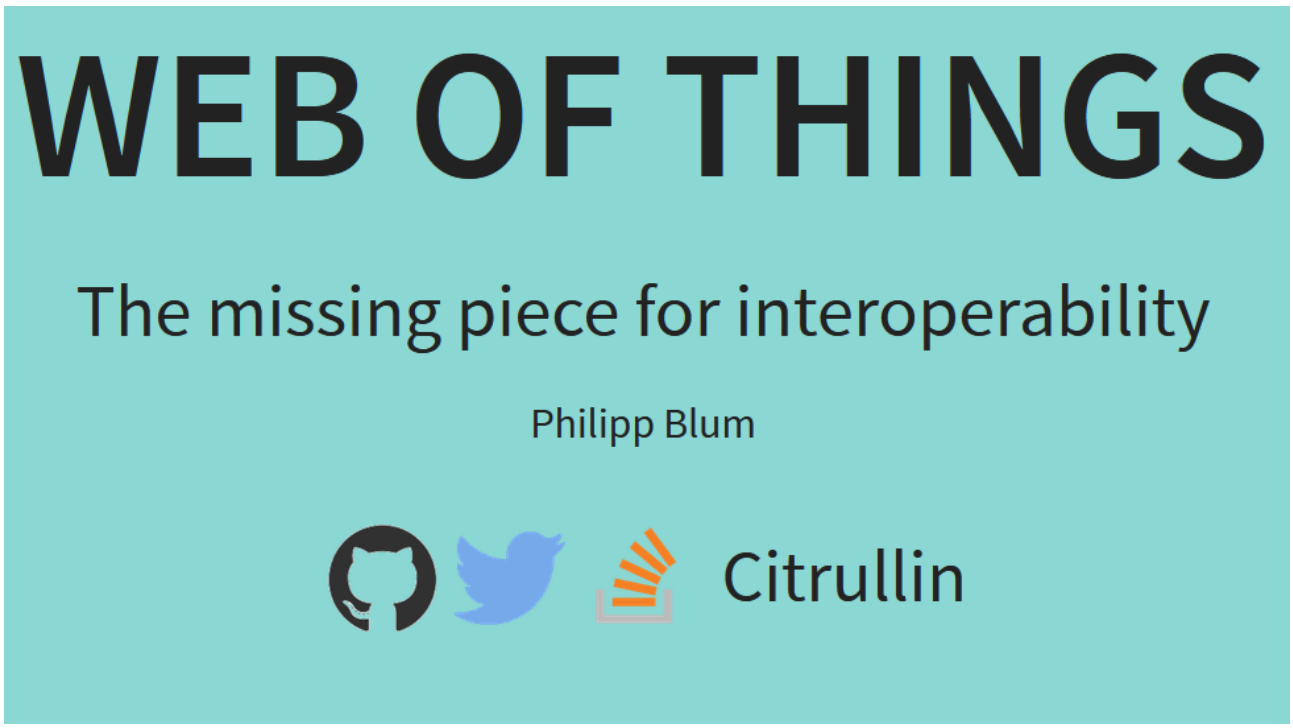
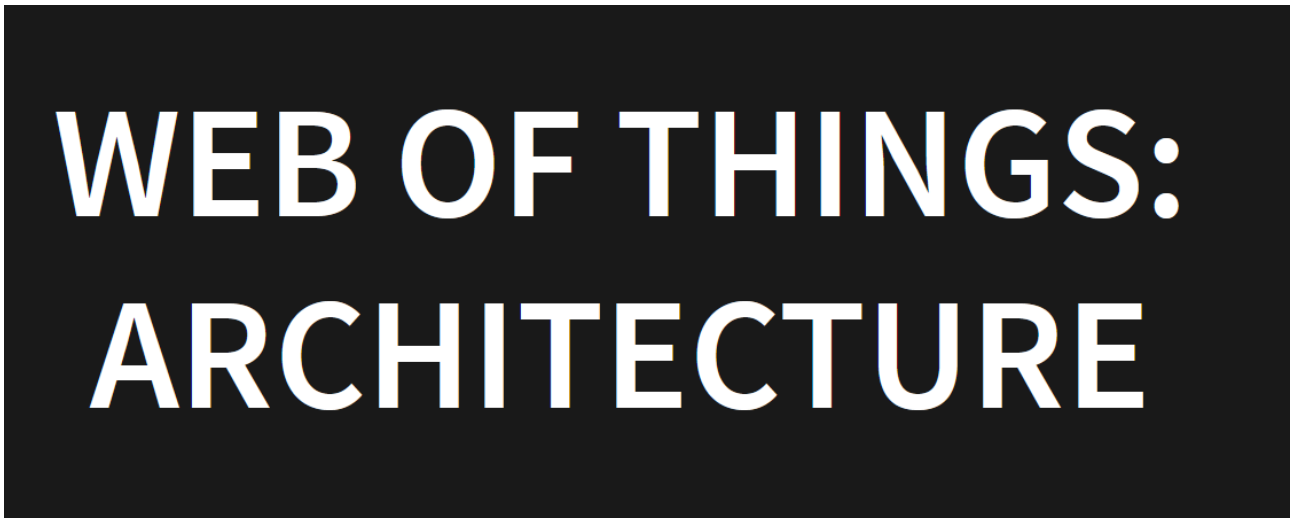
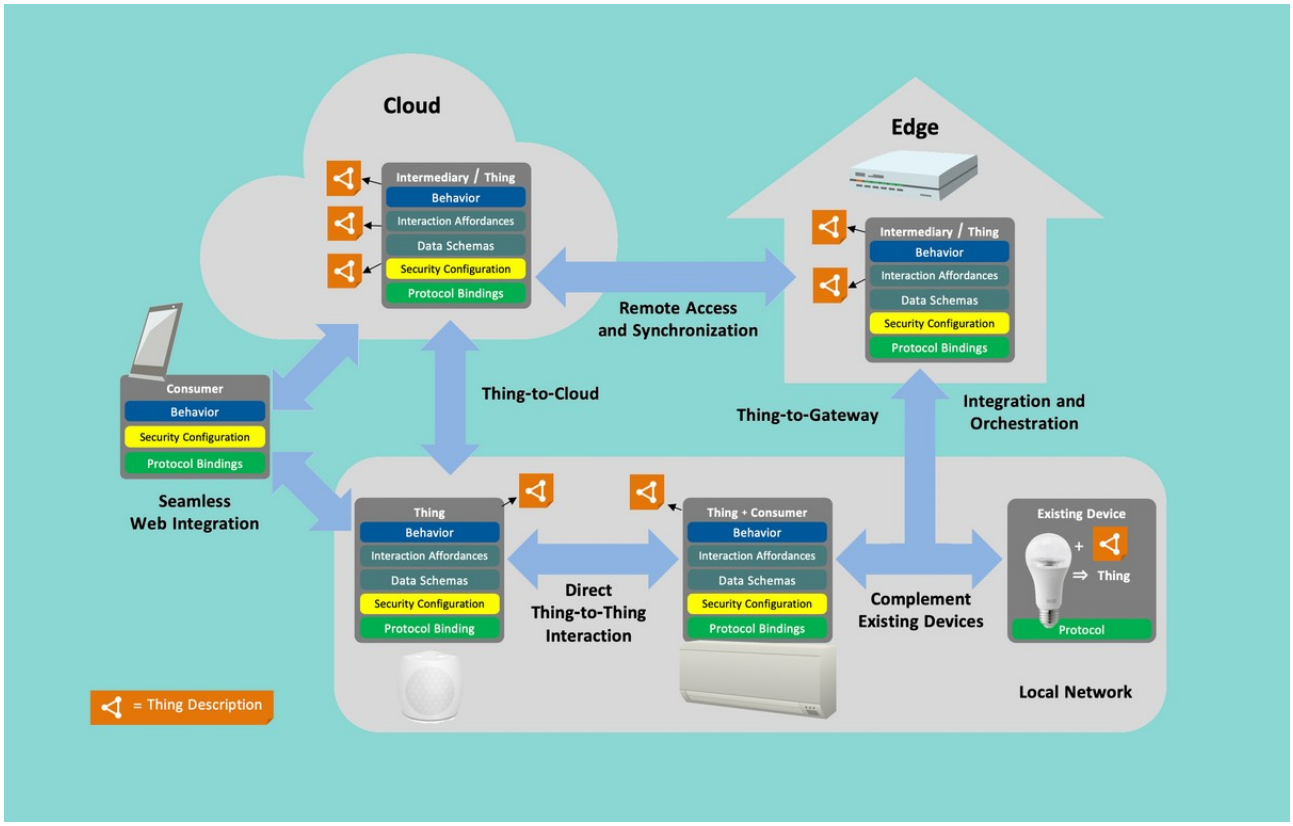
This talk is originally written in Javascript. The code can be found on Github.







RIOT OS & WEB OF THINGS

It's all about Things Description (TD)

(Pseudocode)

```
{
    "@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/st
        }
    },
```

WOT SCRIPTING API

A standard API to interact with Things

(Pseudocode)

```
fetch("https://tds.mythings.org/sensor11")
    .then(res => new ConsumedThing(res.json()))
    .then(thing => {

        thing
            .subscribeProperty(
                "status",
                value => { console.log("Status of the lamp: "
            );

        thing
            .subscribeEvent(
                "overheating",
                data => { console.log("Ready; index: "+ data
```

WOT BINDING TEMPLATES

"Binding Templates enable a Thing Description to be adapted to the specific protocol or data payload usage across the different standards."

Binding can be defined in the context of a TD

(Pseudocode, invalid)

```
{
    "@context": [
        "https://www.w3.org/2019/wot/td/v1",
        {
            "cov": "http://www.example.org/coap-binding#"
        }
    ],
    "id": "urn:dev:ops:32473-WoTLamp-1234",
    "title": "MyLampThing"
}
```

And use it in an Interaction Affordance

(Pseudocode, invalid)

```
{
    "actions": {
        "toggle": {
        "description" : "Turn on or off the lamp",
        "forms": [{
                "href": "coaps://mylamp.example.com/toggle",
                "cov:methodName" : "POST"
            }]
        }
    }
}
```

INTEGRATION INTO RIOT OS

Modules: wot and wot_coap

MODULE: WOT

Structure and API to manipulate it

```
typedef struct {
    json_ld_context_t *context;
    wot_td_type_t *type;
    wot_td_uri_t *id;
    wot_td_multi_lang_t *titles;
    wot_td_multi_lang_t *descriptions;
    wot_td_version_info_t *version;
    wot_td_date_time_t *created;
    wot_td_date_time_t *modified;
    wot_td_uri_t *support;
    wot_td_uri_t *base;
    wot_td_prop_affordance_t *properties;
    wot_td_action_affordance_t *actions;
    wot_td_event_affordance_t *events;
```

MODULE: WOT_COAP

WoT API on top of nanocoap

```
typedef struct {
    coap_resource_t *coap_resource;
    wot_td_prop_affordance_t *affordance;
} wot_td_coap_prop_affordance_t;

typedef struct {
    coap_resource_t *coap_resource;
    wot_td_action_affordance_t *affordance;
} wot_td_coap_action_affordance_t;

typedef struct {
    coap_resource_t *coap_resource;
    wot_td_event_affordance_t *affordance;
} wot_td_coap_event_affordance_t;
```

FEATURES FOR THE FUTURE:

- WoT persistence: Module to cache the TD on a persistent storage.
- WoT SAUL/eSAUL: Module to generate a TD from SAUL/eSAUL

WANT TO SUPPORT?

"Tracker: WoT Thing Description" on Github

Financial contributions are welcomed to focus on the development.

QUESTIONS?