HITACHI
Inspire the Next

Plugfest Result

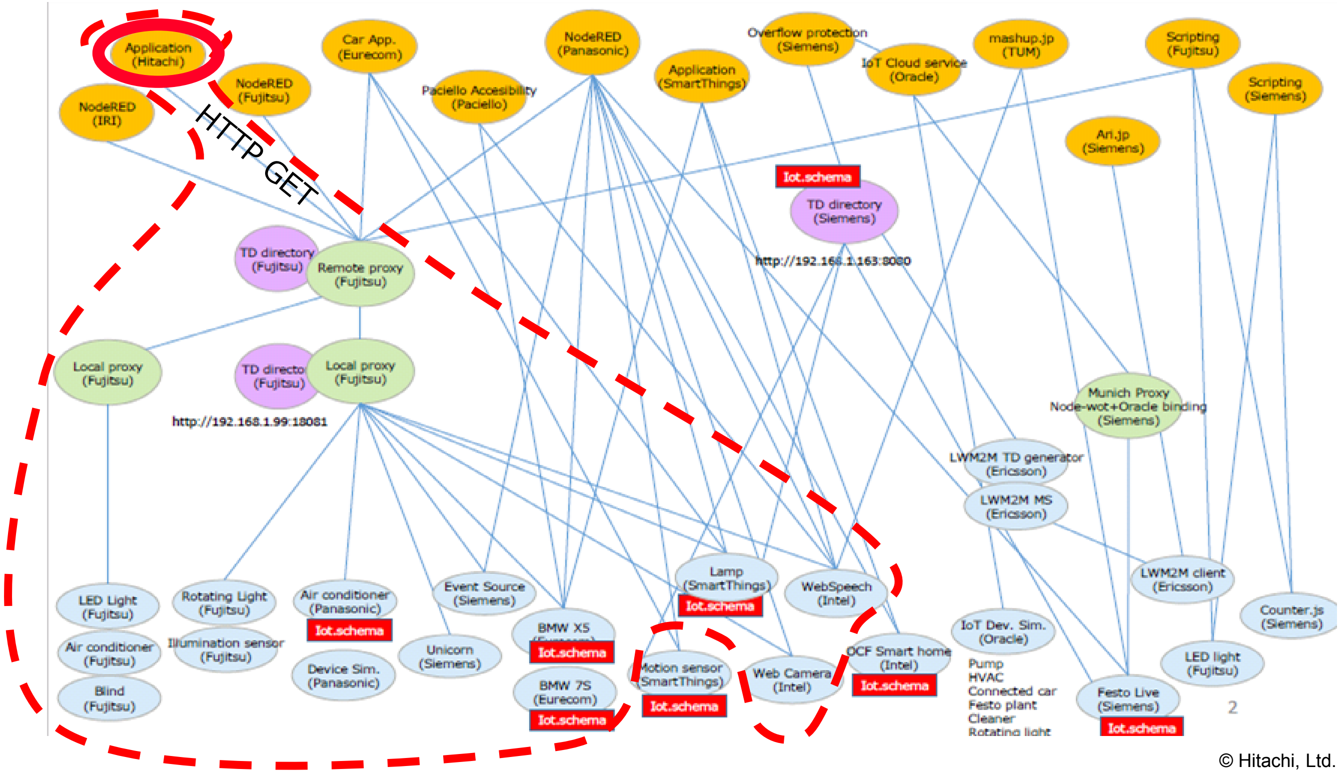
W3C Web of Things IG/WG F2F meeting @ Prague

Kunihiko Toumura, Hitachi Ltd.
Kunihiko.toumura.yv@hitachi.com

Apr. 25, 2018

Plugfest Summary (1/2)

- We've implemented two Application Servients.
  - Generate configuration files for commonly-used IoT Tools (Logstash, Node-RED)
  - Retrieving (HTTP GET) properties of Things via Fujitsu's Remote Proxy.



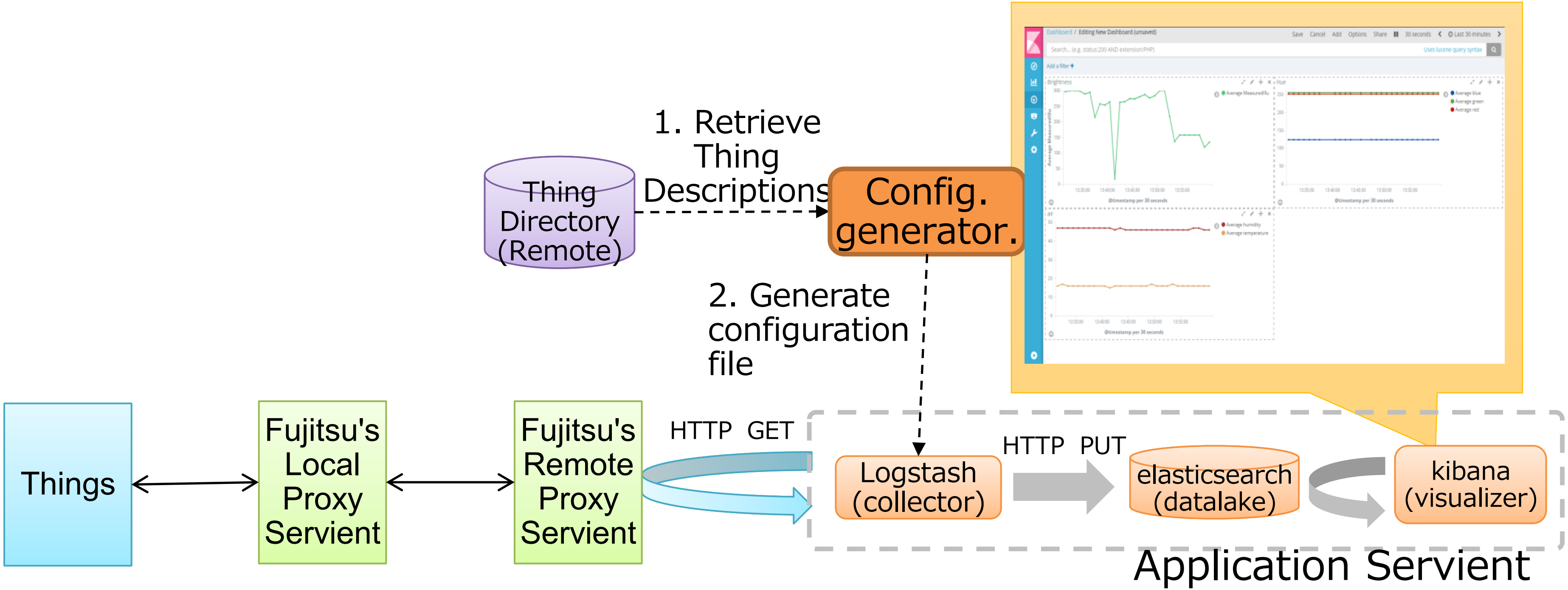
Plugfest Summary (2/2)

Lessons Learned ("Checking point for the plugfest" from PlugfestSummary180418.pdf by Matsukura-san)

1. **Connect with remote/local proxy (narrow waist model)**
   - It is useful for application developer (on the Internet) to aggregate all local device servient access.
2. **Application servient**
   - We can easily connect to WoT device using IoT tools that support HTTP REST API call.
   - It might be a good idea to check connectivity of other existing IoT tools for broader adoption of WoT.
3. **Connect with node-wot**
   - (future work)
4. **Scripting API**
   - (future work)
5. **Thing Directory**
   - We just crawl all TDs in Fujitsu's directory.
   - Using search functions from application servient is future work.
6. **Many kinds of device servients**
   - Our application only collect properties on these devices. Not yet utilized each characteristic of devices...
   - Utilizing each device's characteristic by semantic annotations in application is future work.
7. **Semantic discovery**
   - (future work)
8. **Security, Accessibility**
   - Using bearer token. HTTPS is not yet tested.
9. **Event handling with long polling**
   - (future work)
10. **Device simulators**
    - (future work)

Application Servient (1/2): using ELK Stack

- Use Thing Description to generate configuration of existing IoT data collector solutions.
  - generate a configuration file for Logstash



Thing Directory (Remote)

1. Retrieve Thing Descriptions

Config. generator.

2. Generate configuration file

Things

Fujitsu's Local Proxy Servient

Fujitsu's Remote Proxy Servient

HTTP GET

Logstash (collector)

HTTP PUT

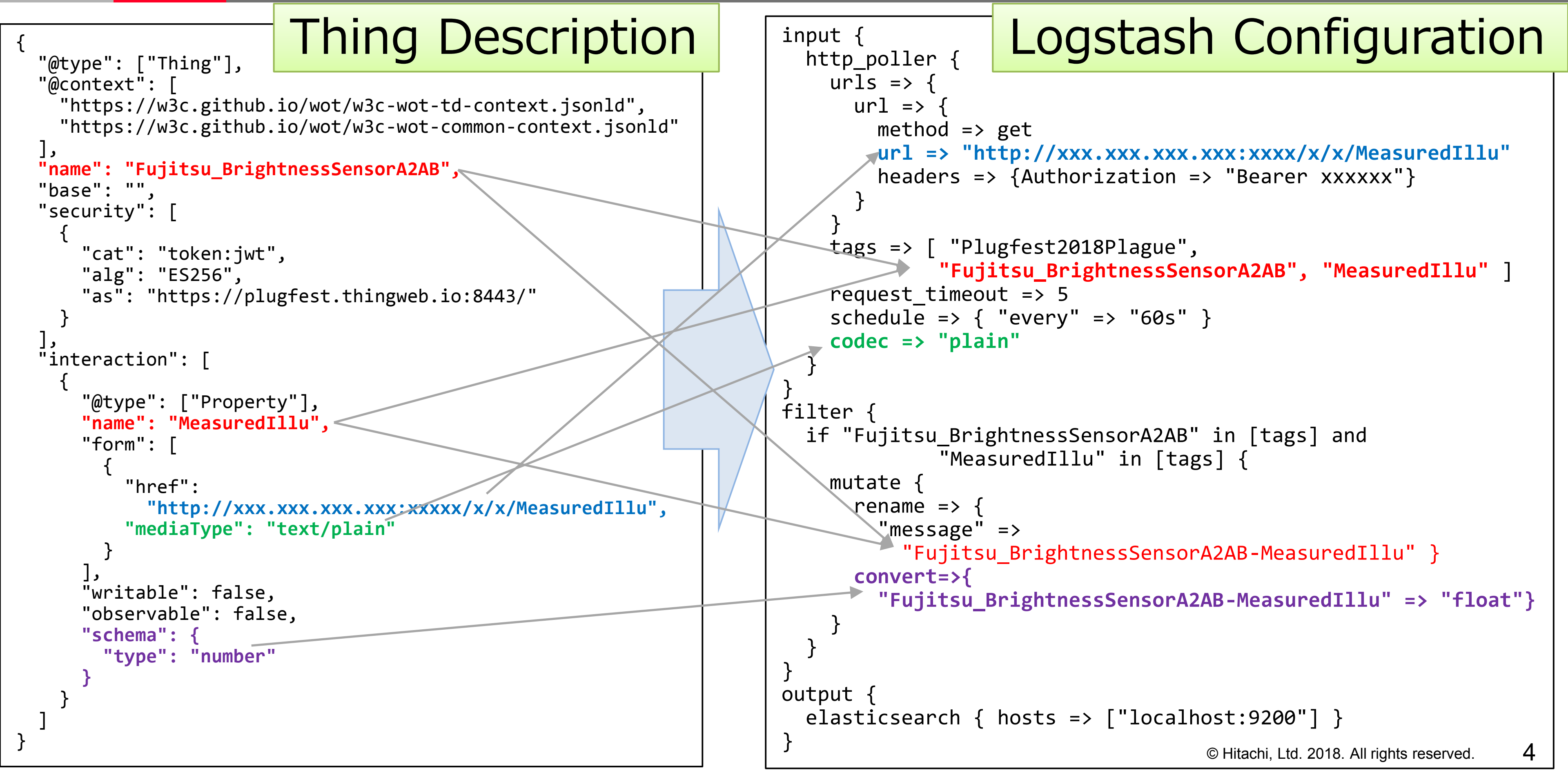
elasticsearch (datalake)

kibana (visualizer)

Application Servient

ELK: Elasticsearch, Logstash, and Kibana. (https://www.elastic.co/elk-stack)

Detail of Configuration file generation



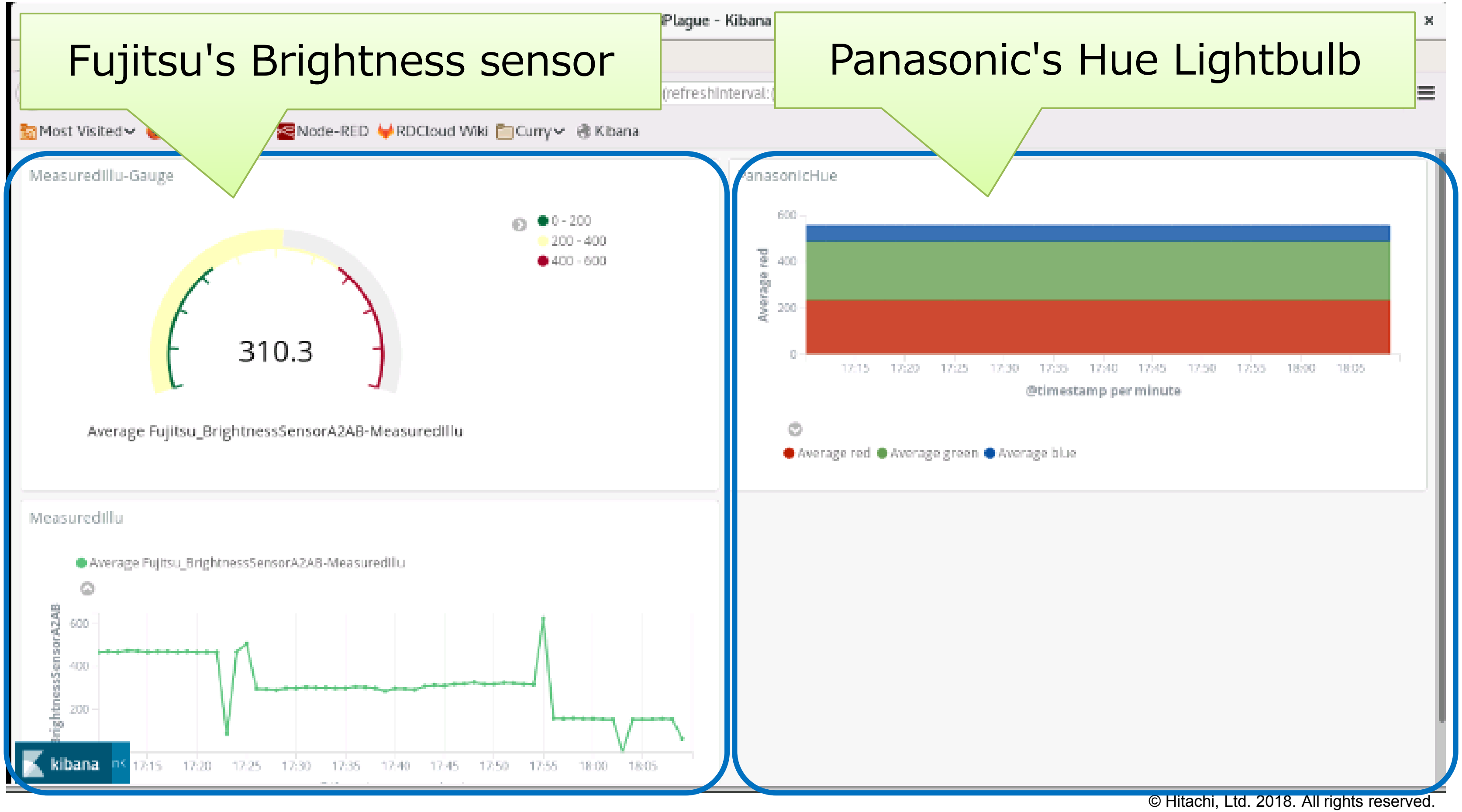
Thing Description

```
{
  "@type": ["Thing"],
  "@context": [
    "https://w3c.github.io/wot/w3c-wot-td-context.jsonld",
    "https://w3c.github.io/wot/w3c-wot-common-context.jsonld"
  ],
  "name": "Fujitsu_BrightnessSensorA2AB",
  "base": "",
  "security": [
    {
      "cat": "token:jwt",
      "alg": "ES256",
      "as": "https://plugfest.thingweb.io:8443/"
    }
  ],
  "interaction": [
    {
      "@type": ["Property"],
      "name": "MeasuredIllu",
      "form": [
        {
          "href":
            "http://xxx.xxx.xxx.xxx:xxxxx/x/x/MeasuredIllu",
          "mediaType": "text/plain"
        }
      ],
      "writable": false,
      "observable": false,
      "schema": {
        "type": "number"
      }
    }
  ]
}
```

Logstash Configuration

```
input {
  http_poller {
    urls => {
      url => {
        method => get
        url => "http://xxx.xxx.xxx.xxx:xxxx/x/x/MeasuredIllu"
        headers => {Authorization => "Bearer xxxxxx"}
      }
    }
    tags => [ "Plugfest2018Plague",
         "Fujitsu_BrightnessSensorA2AB", "MeasuredIllu" ]
    request_timeout => 5
    schedule => { "every" => "60s" }
    codec => "plain"
  }
}
filter {
  if "Fujitsu_BrightnessSensorA2AB" in [tags] and
           "MeasuredIllu" in [tags] {
    mutate {
      rename => {
        "message" =>
         "Fujitsu_BrightnessSensorA2AB-MeasuredIllu" }
      convert=>{
        "Fujitsu_BrightnessSensorA2AB-MeasuredIllu" => "float"}
    }
  }
}
output {
  elasticsearch { hosts => ["localhost:9200"] }
}
```

Visualization result

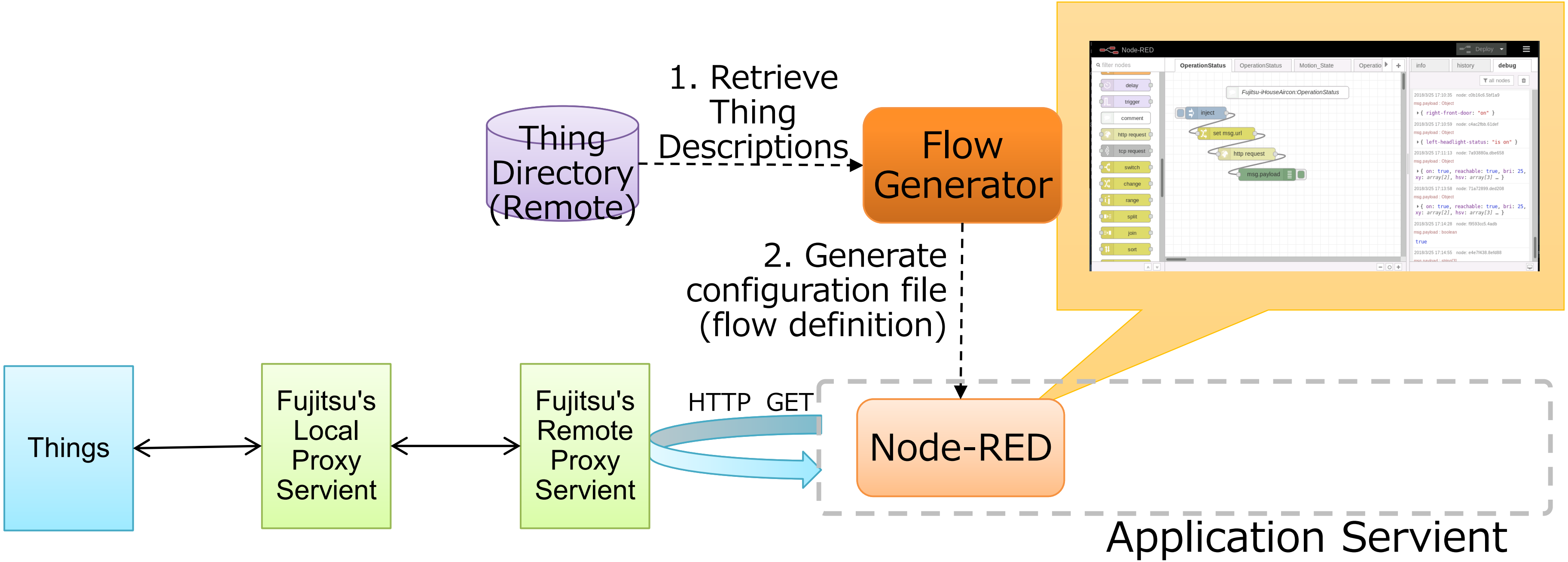


Fujitsu's Brightness sensor

Panasonic's Hue Lightbulb

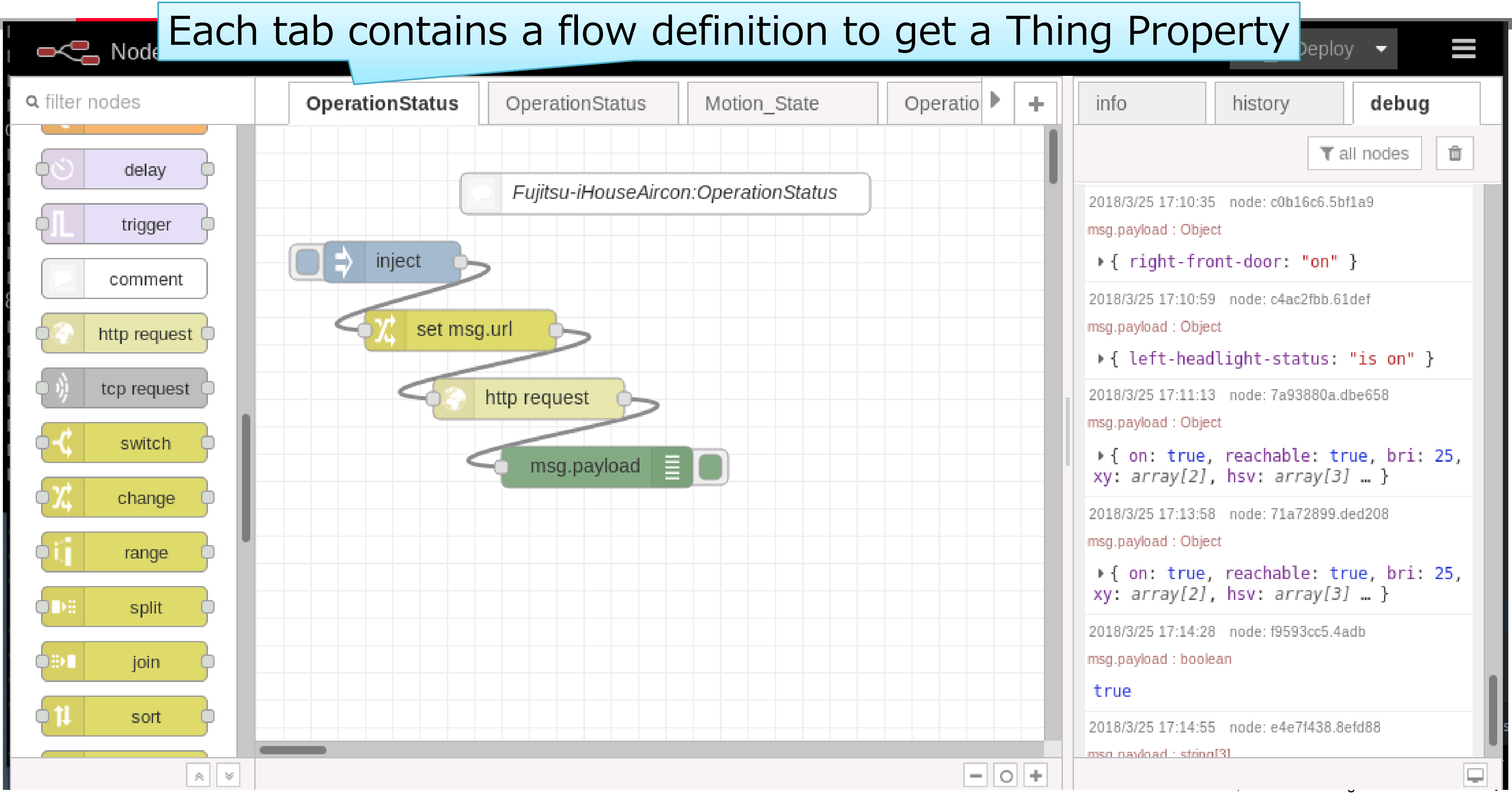
Application Servient (2/2): using Node-RED

- Use Thing Description to generate program (Node-RED flow)
  - generate skeleton flows for retrieve each property of Things



```
2018/3/25 17:10:35   node: c0b16c6.5bf1a9
msg.payload : Object
▸{ right-front-door: "on" }
2018/3/25 17:10:59   node: c4ac2fbb.61def
msg.payload : Object
▸{ left-headlight-status: "is on" }
2018/3/25 17:11:13   node: 7a93880a.dbe658
msg.payload : Object
▸{ on: true, reachable: true, bri: 25,
xy: array[2], hsv: array[3] … }
2018/3/25 17:13:58   node: 71a72899.ded208
msg.payload : Object
▸{ on: true, reachable: true, bri: 25,
xy: array[2], hsv: array[3] … }
2018/3/25 17:14:28   node: f9593cc5.4adb
msg.payload : boolean
true
2018/3/25 17:14:55   node: e4e7f438.8efd88
msg.payload : string[3]
```

Example of Generated Flow Skeleton



Each tab contains a flow definition to get a Thing Property

```
2018/3/25 17:10:35  node: c0b16c6.5bf1a9
msg.payload : Object
▸{ right-front-door: "on" }
2018/3/25 17:10:59  node: c4ac2fbb.61def
msg.payload : Object
▸{ left-headlight-status: "is on" }
2018/3/25 17:11:13  node: 7a93880a.dbe658
msg.payload : Object
▸{ on: true, reachable: true, bri: 25,
xy: array[2], hsv: array[3] … }
2018/3/25 17:13:58  node: 71a72899.ded208
msg.payload : Object
▸{ on: true, reachable: true, bri: 25,
xy: array[2], hsv: array[3] … }
2018/3/25 17:14:28  node: f9593cc5.4adb
msg.payload : boolean
true
2018/3/25 17:14:55  node: e4e7f438.8efd88
msg.payload : string[3]
```

HITACHI

Inspire the Next