Semantic Interoperability Requires Self-describing Interaction Models

IRTF T2TRG Meeting, San Jose, CA, USA, 2016

Matthias Kovatsch (kovatsch@inf.ethz.ch)
Klaus Hartke (hartke@tzi.org)

Information Model for Interoperability

- Make use of data produced by IoT devices
- Well understood that data must be meaningful

➔ About the “what”

- Domain-specific requirements have led to multiple consortia
- Each consortium has defined their own data model
- Inferred meta model could help to bridge between data models

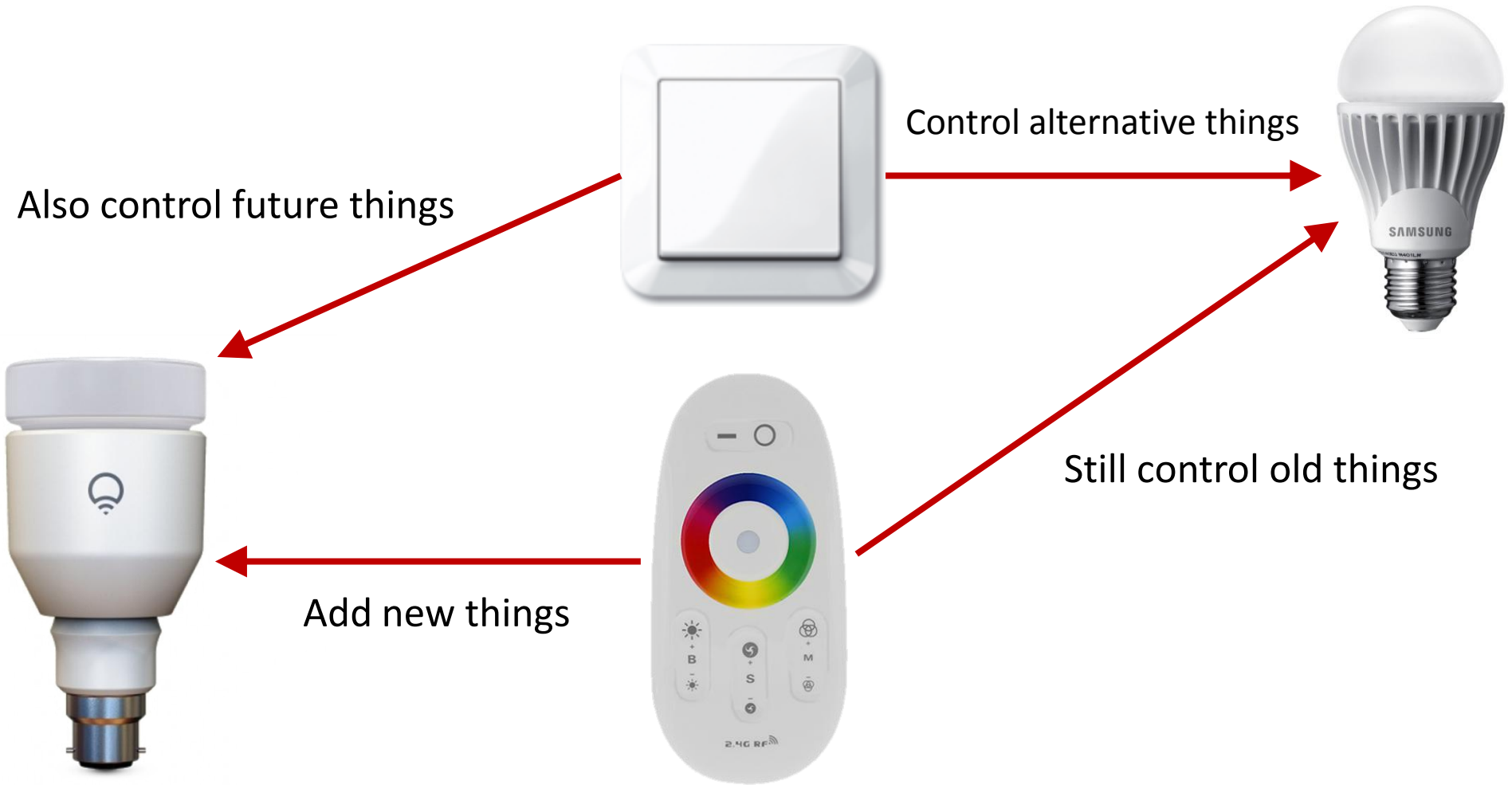
Interaction Model

- Machine-to-machine communication
- Handle change in a global system through automation

➔ About the “how”

- Integration of descriptions on the server side is straight-forward
- Consumption on the client side is challenging
- Missing abstractions have led to hard-coded clients

Change


Control alternative things
Also control future things
Still control old things
Add new things
SAMSUNG

Interaction Model

- Machine-to-machine communication
- Handle change in a global system through automation

➔ About the “how”

- Integration of descriptions on the server side is straight-forward
- Consumption on the client side is challenging
- Missing abstractions have led to hard-coded clients

Semantic Interoperability

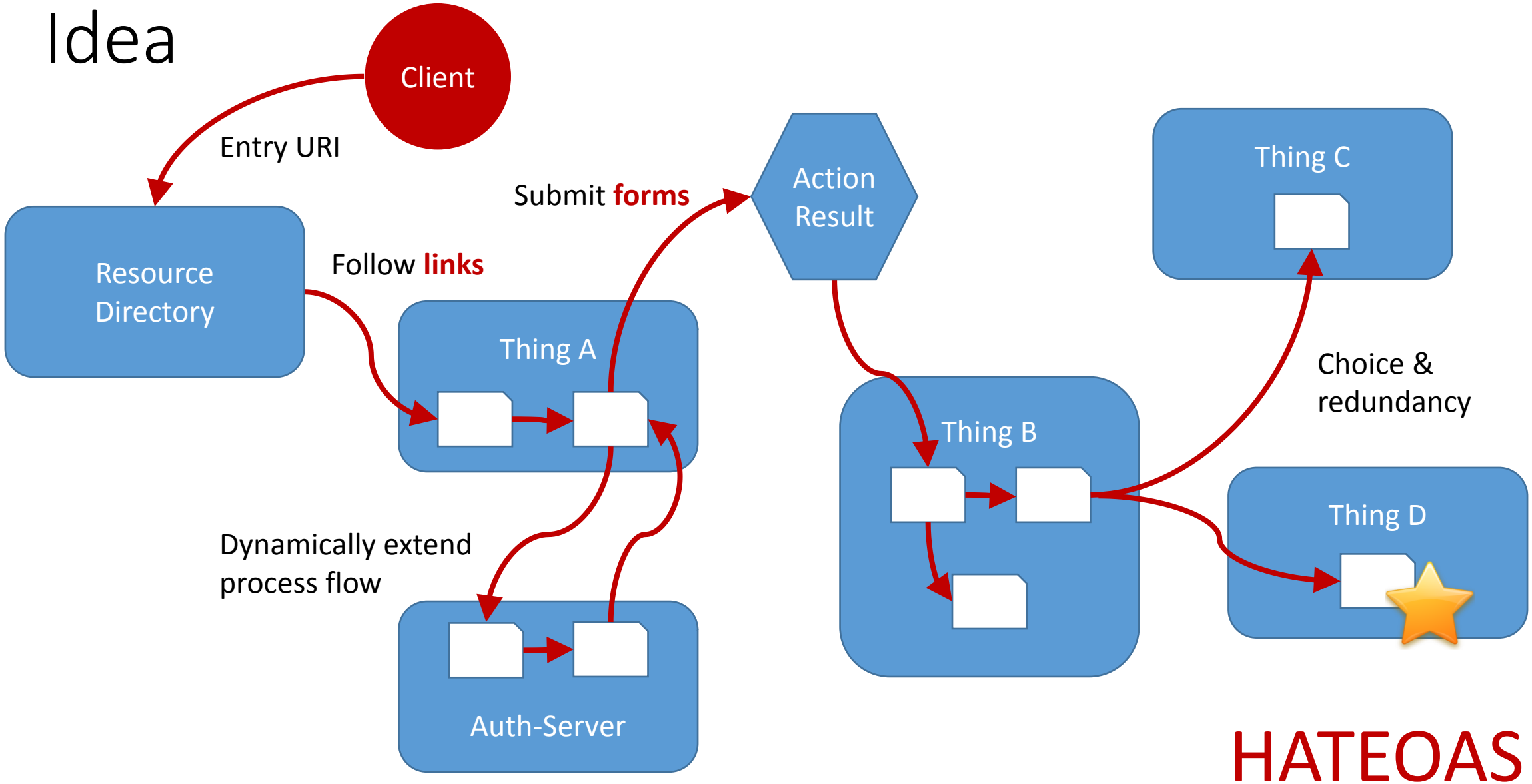
- Information model
  - Describing the exchanged information → vocabulary
  - Must allow for linking data models from different application domains
  - Semantic model such as RDF can span multiple domains/consortia
- Interaction model
  - Describing the possible interactions with a service/thing → vocabulary
  - Must allow for change and diversity
  - Hypermedia-driven REST (HATEOAS)

Interaction Model with HATEOAS

- Hypermedia As The Engine Of Application State

- Model application with atomic interaction steps (request-response)
- Links and forms describe how requests must be formulated
- Relation type vocabulary attaches meaning (shared a priori)
- Publication of links and forms allows for change (URIs shared at runtime)

Idea



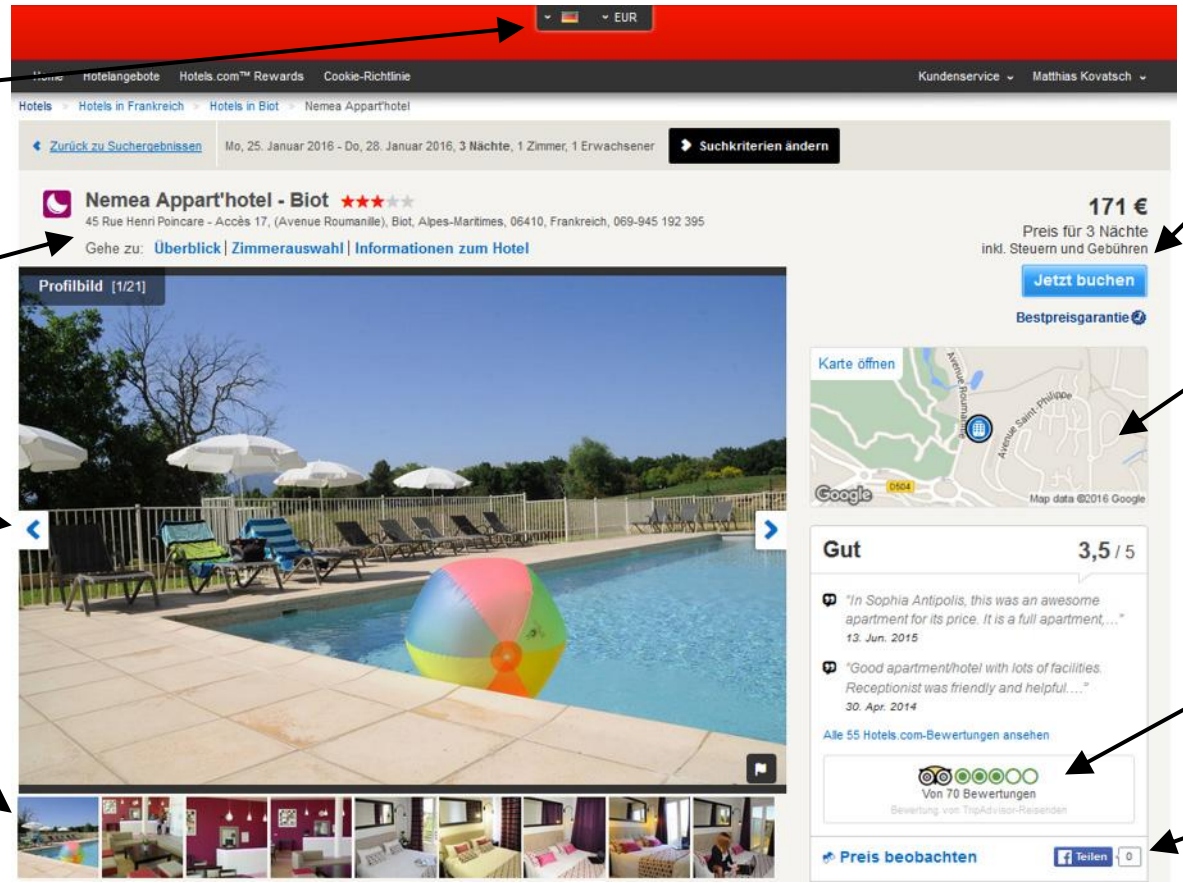
HATEOAS

HATEOAS Summary

- Atomic interactions are described in-band and shape application
  - Links
  - Forms
  - Relation types (shared a priori)
- Loose Coupling
  - Servers are free to define their own resource structure
  - Clients and servers can evolve independently
- Clients can learn applications on the fly
  - Dynamically add new or even proprietary features
  - Clients can adapt to changing environments
- Servers are easy, clients are hard

Web Mashups through Open APIs

Internal microservice APIs



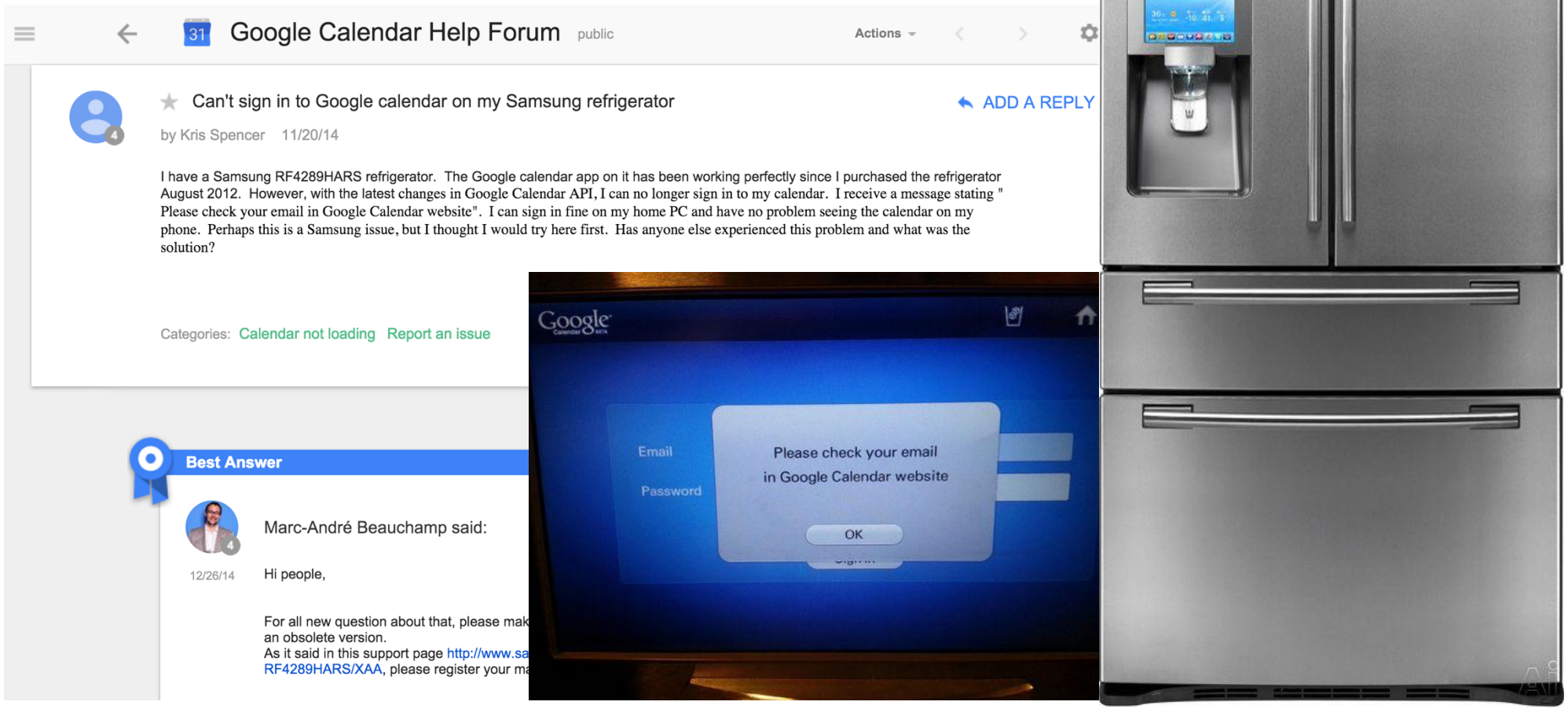
PayPal

tripadvisor®

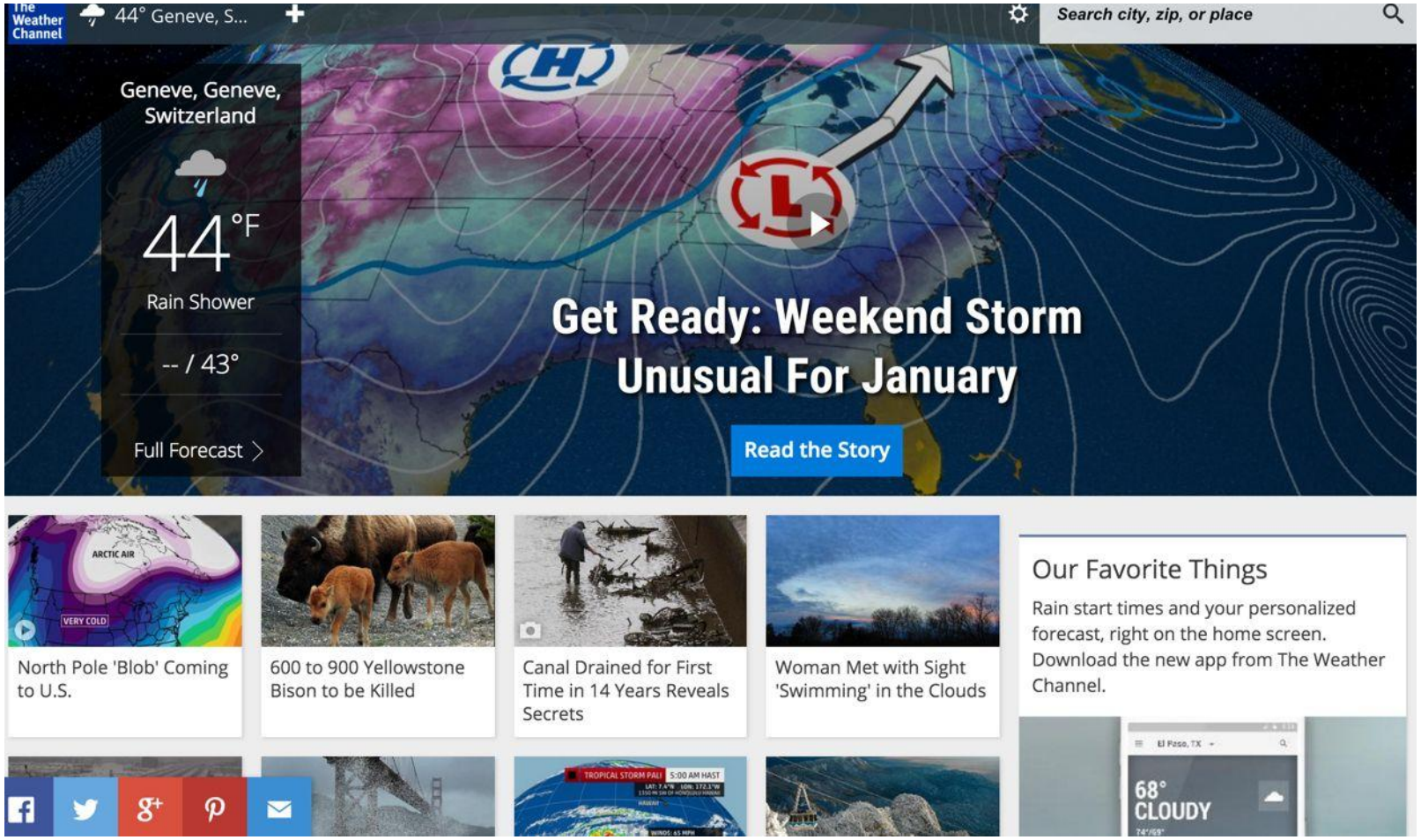
facebook

AccuWeather

… Often Break


Google Calendar Help Forum
public
Actions
Can't sign in to Google calendar on my Samsung refrigerator
ADD A REPLY
by Kris Spencer 11/20/14
I have a Samsung RF4289HARS refrigerator. The Google calendar app on it has been working perfectly since I purchased the refrigerator August 2012. However, with the latest changes in Google Calendar API, I can no longer sign in to my calendar. I receive a message stating " Please check your email in Google Calendar website". I can sign in fine on my home PC and have no problem seeing the calendar on my phone. Perhaps this is a Samsung issue, but I thought I would try here first. Has anyone else experienced this problem and what was the solution?
Categories: Calendar not loading Report an issue
Best Answer
Marc-André Beauchamp said:
12/26/14
Hi people,
For all new question about that, please mak
an obsolete version.
As it said in this support page http://www.sa
RF4289HARS/XAA, please register your ma
Google
Email
Password
Please check your email
in Google Calendar website
OK

Human Web Interaction



CoRE-HAL and Hypermedia Client

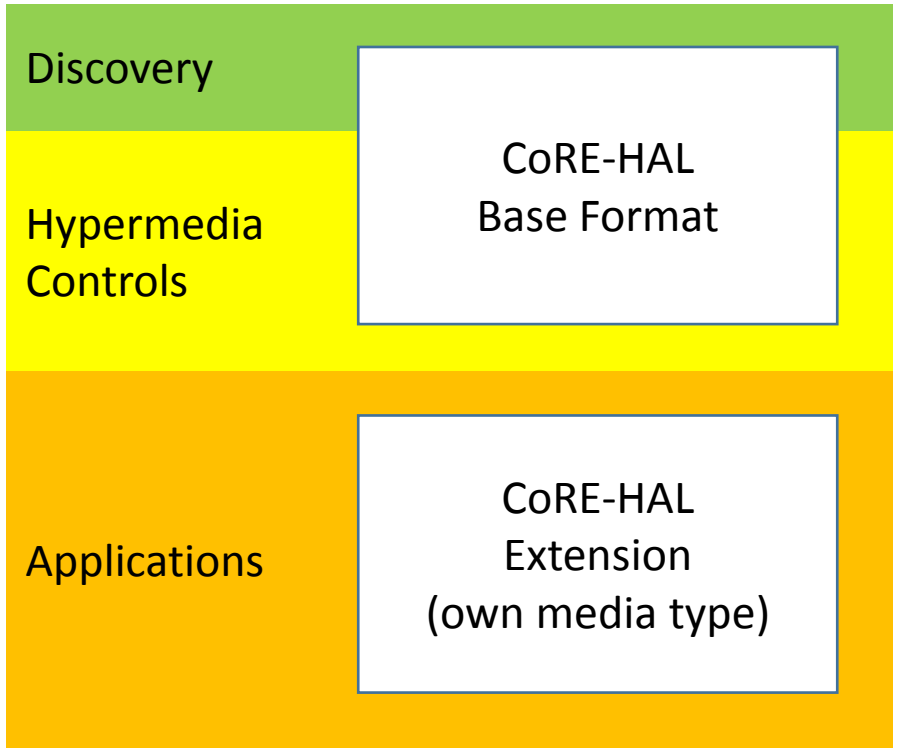
Extensions by Matthias Kovatsch and Yassin Hassan

kovatsch@inf.ethz.ch

Extensible Representation Format

- CoRE-HAL base format
  - For now JSON (without -LD)
  - Hypermedia controls (links and forms)
  - Common descriptions (things, locations)

- Application-specific extensions
  - Descriptions for atomic use cases
  - Grouping of semantic vocabulary
    - Information model
    - Link and form relation types



CoRE-HAL Lighting State Example

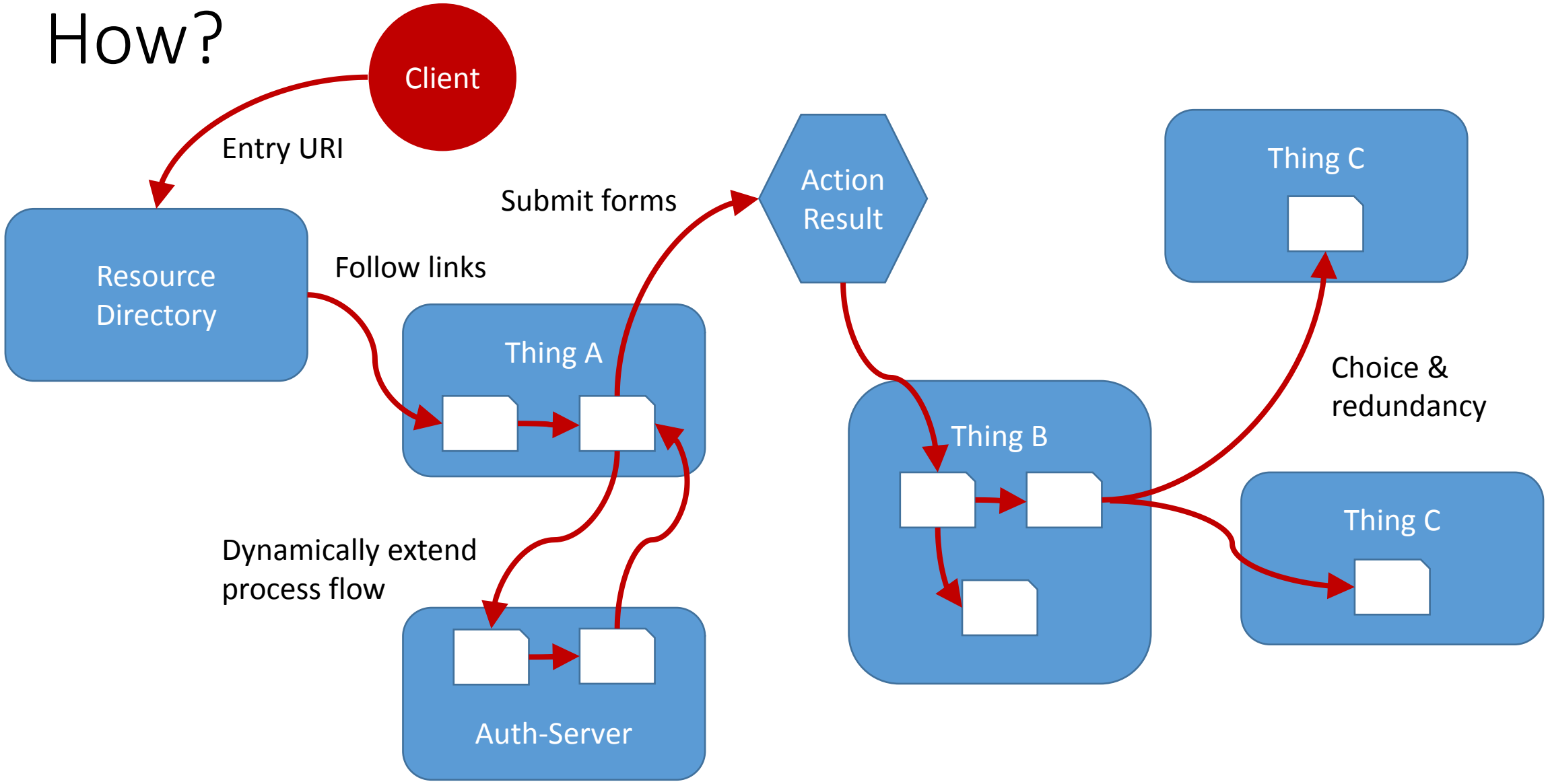
```
{
    "value": {"r":255, "g":0, "b":0},
    "mode": "rgb",
    "_links": {
       "same-as": {
          "href": "/brightness",
          "type": "application/x.lighting-state+json"
       }
    },
    "_forms": {
       "edit": {
          "method": "PUT",
          "href": "/light",
          "accepts": "application/x.lighting-state+json"
       }
    }
}
```

Application Data

Links

Forms

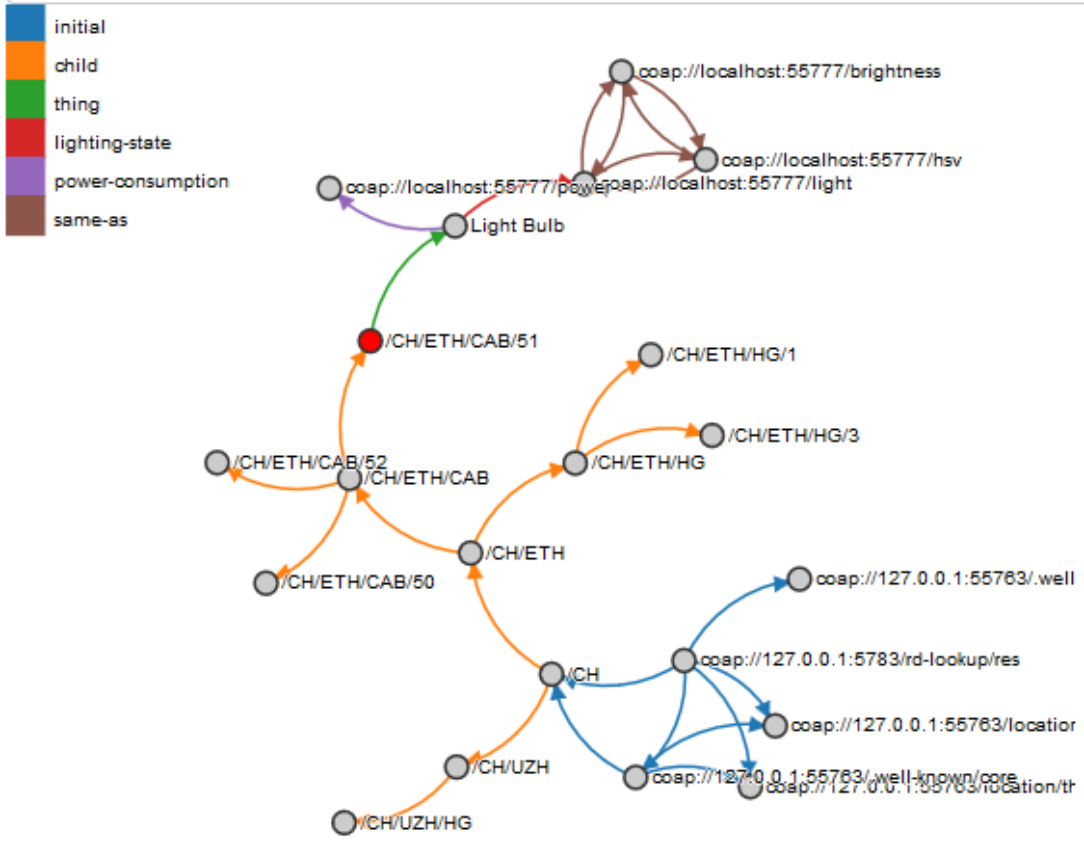
How?


Client
Entry URI
Resource Directory
Follow links
Thing A
Submit forms
Action Result
Thing C
Choice & redundancy
Thing B
Thing C
Dynamically extend process flow
Auth-Server

CoRE-HAL Explorer

coap://127.0.0.1:5783/rd-lookup/res

Run!



coap://localhost:55768/location


```
{
   location: "/CH/ETH/CAB/51",
 - _links: {
     - thing: {
          href: "coap://localhost:55775/thing",
          type: "application/x.thing-description+json"
       }
   },
 - _forms: {
     - remove-thing: {
          href: "/location/things{?_self}",
          method: "DELETE",
          accepts: "application/x.thing-description+json",
          templated: true
       },
     - add-child: {
          href: "/location/children",
          method: "POST",
          accepts: "application/x.location-description+json"
       },
     - add-thing: {
          href: "/location/things",
          method: "POST",
          accepts: "application/x.thing-description+json"
       },
     - edit: {
          href: "",
          method: "POST",
          accepts: "application/x.location-description+json"
       }
   },
```

Hypermedia Client

- High-level path description to resource based on link relation types
- Actual (dynamic) URIs are retrieved from representations

```
entry = new HypermediaClient("coap://home.local"); // entry point
light = entry.follow("lighting"); // link relation type
state = light.follow("state"); // link relation type
```

Hypermedia Client Futures

- Lazy loading of resource representations
- Only request representations (i.e., transmit data) when used

```
entry = new HypermediaClient("coap://home.local");
light = entry.follow("lighting"); // returns Future
state = light.follow("state"); // returns Future
representation = state.get(); // lazy evaluation (not a GET)
```

Hypermedia Client Futures

- Reloadable resource representation in the Future
- Transparently handles cache control

```
entry = new HypermediaClient("coap://home.local");
light = entry.follow("lighting");
state = light.follow("state"); // returns Future
representation = state.get(); // lazy evaluation
/* Max-Age expires */
representation = state.get(); // retransmission of representation
```

Hypermedia Client Futures

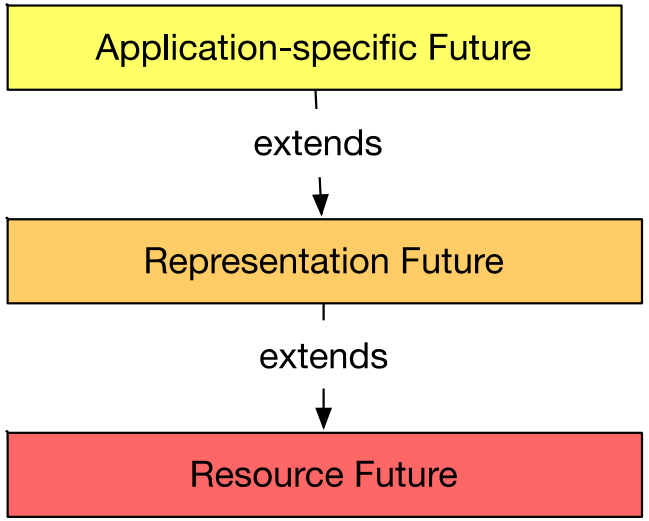
- Bookmark support
- On error discovery is re-triggered
  to recover from unavailable/replaced resources/devices

```
/* thing is replaced, address and resource path changes */
data = state.get(); // resource described in Future is re-discovered
```

Hypermedia Client Programming API

- Programmatically provide application-specific operations
- Allow developer to use the IDE auto-completion feature

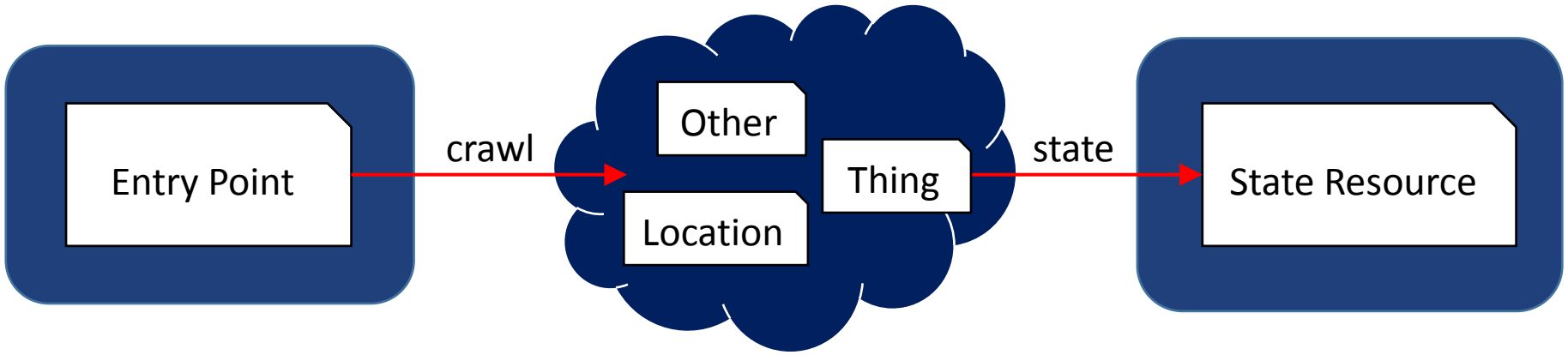
```
public class LightingStateFuture
       extends CoREHalResourceFuture<LightingState> {

    public void setRGB(int r, int g, int b)  {
       LightingState representation = new LightingState();
       representation.setValue(new RGBValue(r,g,b));
       submitForm("edit", representation);
    }

}
```



Hypermedia Crawlers

- Abstract arbitrarily long link chains
- Can include metadata (and data) in link selection decision

```
                                        // defines algorithm
thing = client.links().use(new ThingCrawler()) // returns crawler
                      .findLocation("/CH/ETH/CAB/51") // metadata
                      .findFirstWith("state") // link relation
```



Links

- http://mkovatsc.github.io/iot-hypermedia/
- http://mkovatsc.github.io/core-hal-explorer/
- https://github.com/eclipse/californium.tools/tree/master/cf-polyfill

HATEOAS Discussion

- Costs
  - Higher design effort
  - More Roundtrips
  - Larger representation size
- Optimizations
  - Caching
  - Bookmarks
  - Reduced representations