MQTT, CoAP and Java

Find Your Way Through the Internet of Things Protocols Jungle

Benjamin Cabé, Eclipse Foundation
Julien Vermillard, Sierra Wireless

Your devoted presenters :-)

Benjamin Cabé / @kartben

IoT Evangelist at the Eclipse Foundation
Connects all ze thingz!
Traveler and photographer

Your devoted presenters :-)

Julien Vermillard / @vrmvrm

Software Engineer at Sierra Wireless
http://airvantage.net M2M Cloud

Apache member, Eclipse committer on Californium, Wakaama and Leshan

M2M/IoT?

Technology that supports wired or wireless communication between devices

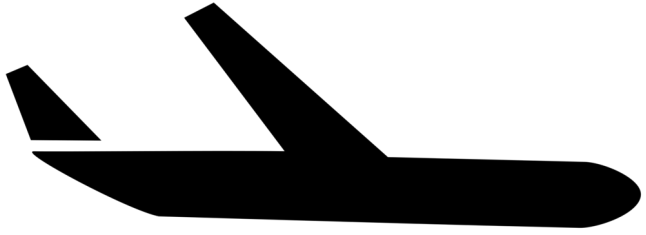
Different needs

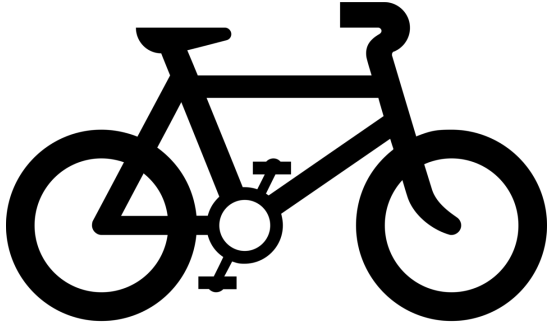
Local communications

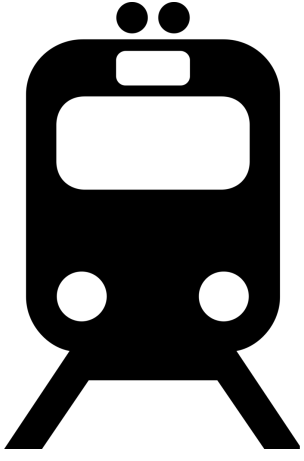
Publishing telemetry values

Device management

Different protocols

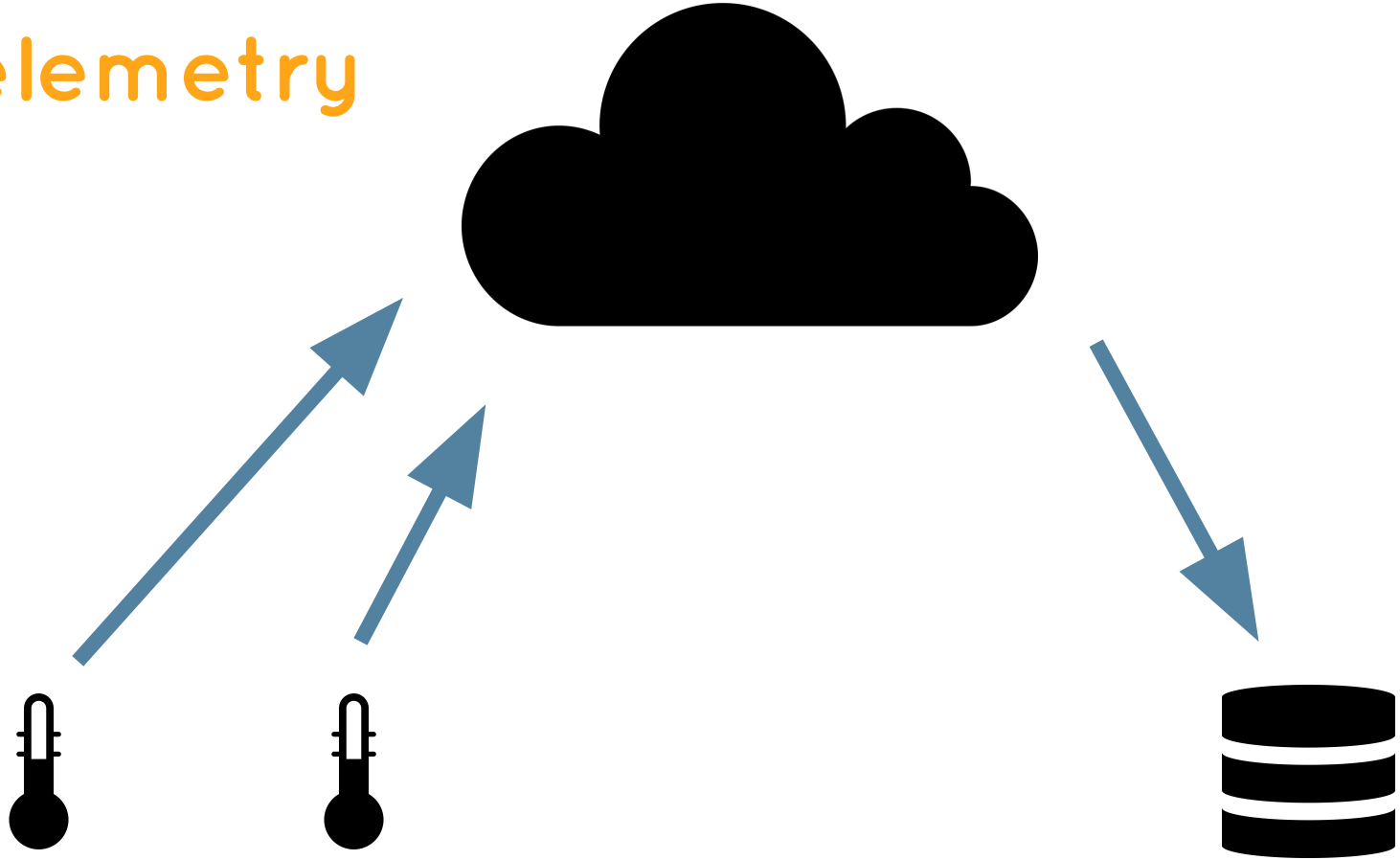







Ready to dive into the jungle?

Telemetry



MQTT

MQ Telemetry Transport

MQTT

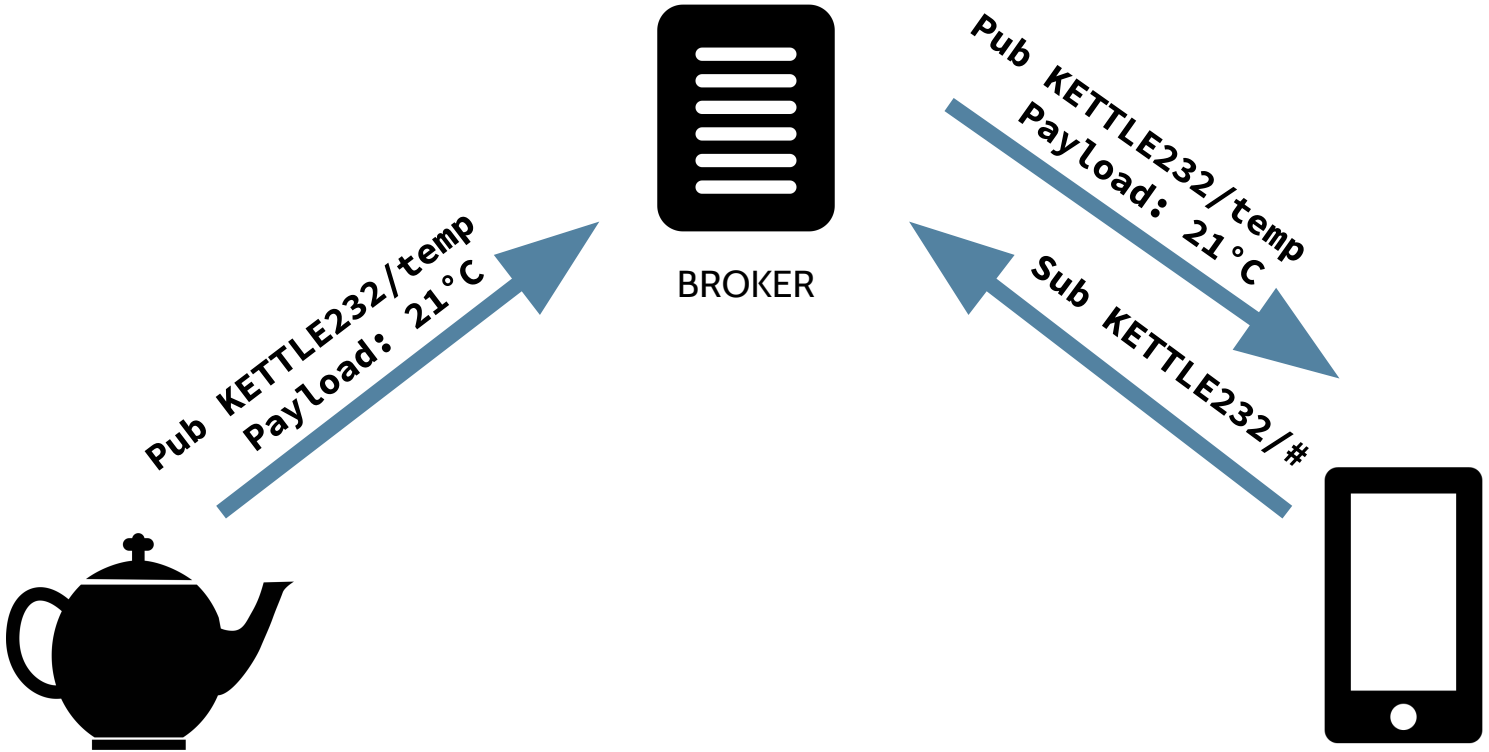
Very simple and light protocol on **TCP**

Invented in 1999 for **telemetry** values

Royalty free since 2010

Now an **OASIS** standard

MQTT Publish & Subscribe



MQTT features

Topics **wildcards**

Quality-of-Service: level 0, 1, 2

Last Will & Testament

Retained messages

MQTT Websockets

WebSockets allow to use MQTT from the web (noBackend)

Ex: real-time web dashboards, HTML5 mobile applications

MQTT-SN

MQTT for Sensor Networks

No dependency on IP (can use e.g Zigbee)

Local sensor network bridged to an MQTT broker via an MQTT-SN gateway

MQTT Security

TLS on top of TCP

Certificate for the broker

Password or certificate for the client

http://eclipse.org/paho

Eclipse Paho

Open-source **MQTT clients**

Pick your language!

Java

also JavaScript, Go, Objective C, Lua, Python ...

Eclipse Paho

Also **embedded** C/C++

Arduino, mbed, CC3200, ...

JavaME (e.g Gemalto Concept Board)*

.NET

* if you try it on mbed Freescale FRDM-K64F please tell us!

Show me the code!


```
MqttClient mqttClient = new MqttClient("tcp://iot.eclipse.org",
                                       "myclientId", new MemoryPersistence());
mqttClient.connect();
mqttClient.setCallback(new MqttCallback() {
    @Override
    public void messageArrived(String topic, MqttMessage message)
            throws Exception {
        // ... a message!
    }
}

mqttClient.subscribe("kettle/temp");
```

Moquette

A pure **Java broker**

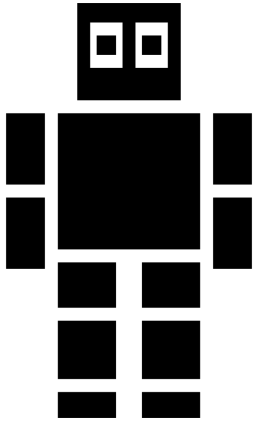
Supports QoS 0, 1, 2

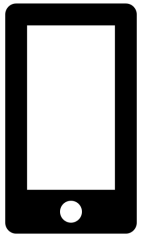
Websockets

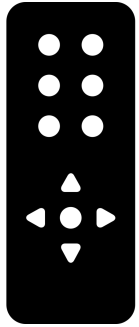
Event-driven: based on Netty and LMAX disruptor

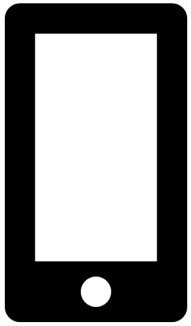
Demo!

The web-of-things

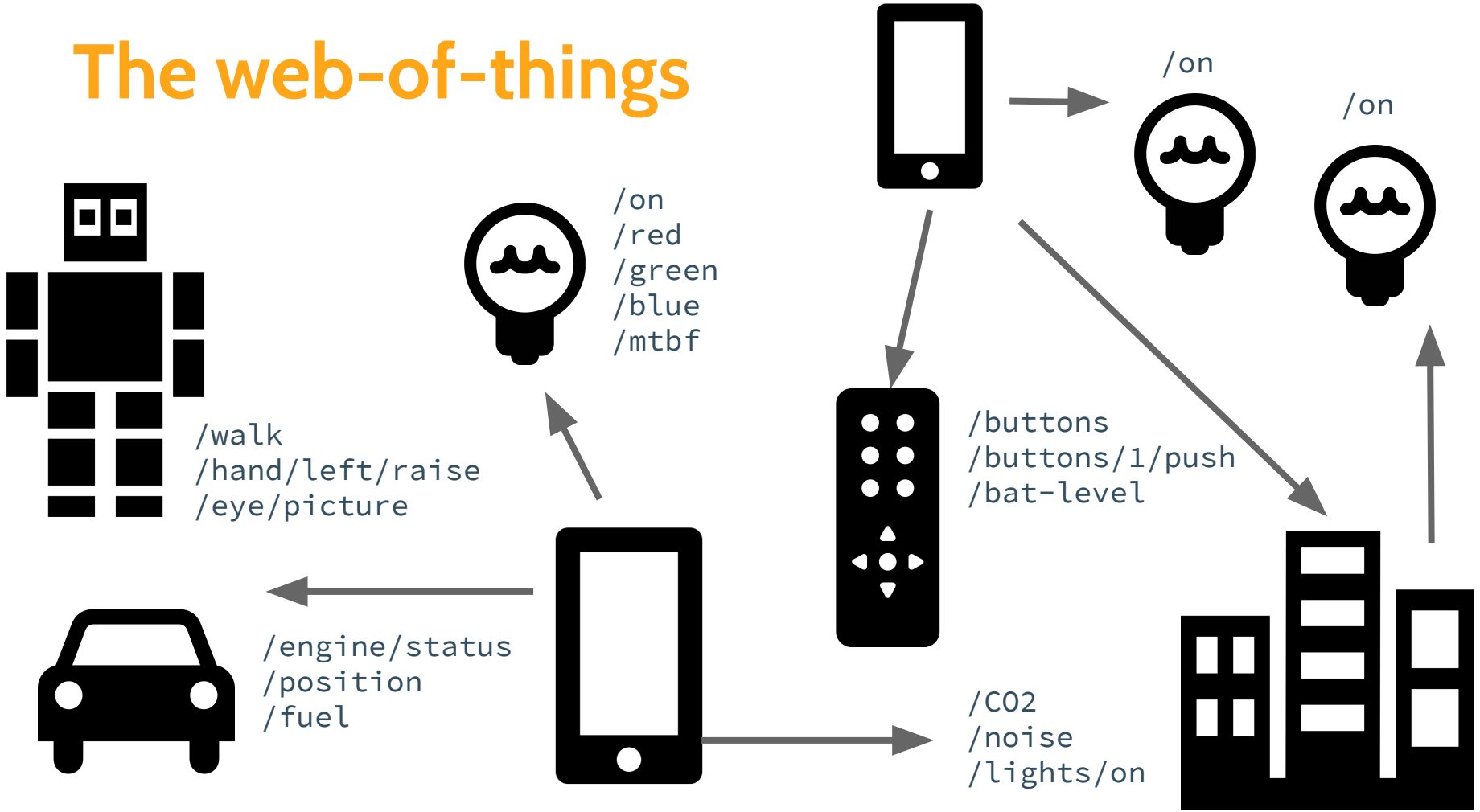







The web-of-things

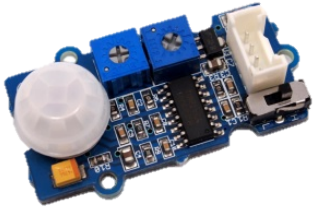


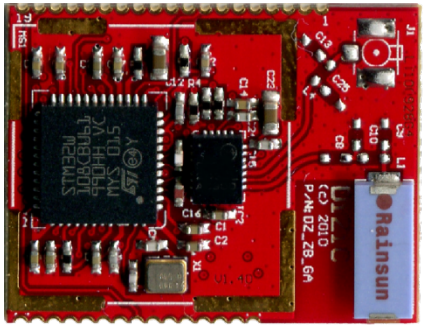
CoAP

Constrained Application Protocol

Tiny resource-constrained devices

Class 1 devices
~100KiB Flash
~10KiB RAM

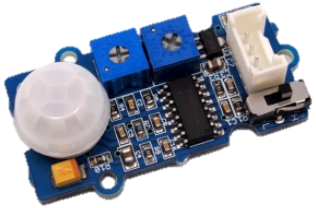


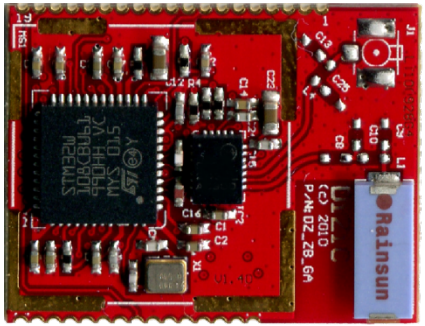


Low-power networks
<100Bytes packets

Tiny resource-constrained devices

Target
of less than $1
for IoT SoC





TCP and HTTP
are not a good fit

Constrained Application Protocol

RESTful protocol designed from scratch

- URIs, Internet Media Types
- GET, POST, PUT, DELETE

Transparent mapping to HTTP

Additional features for M2M scenarios

- Observe

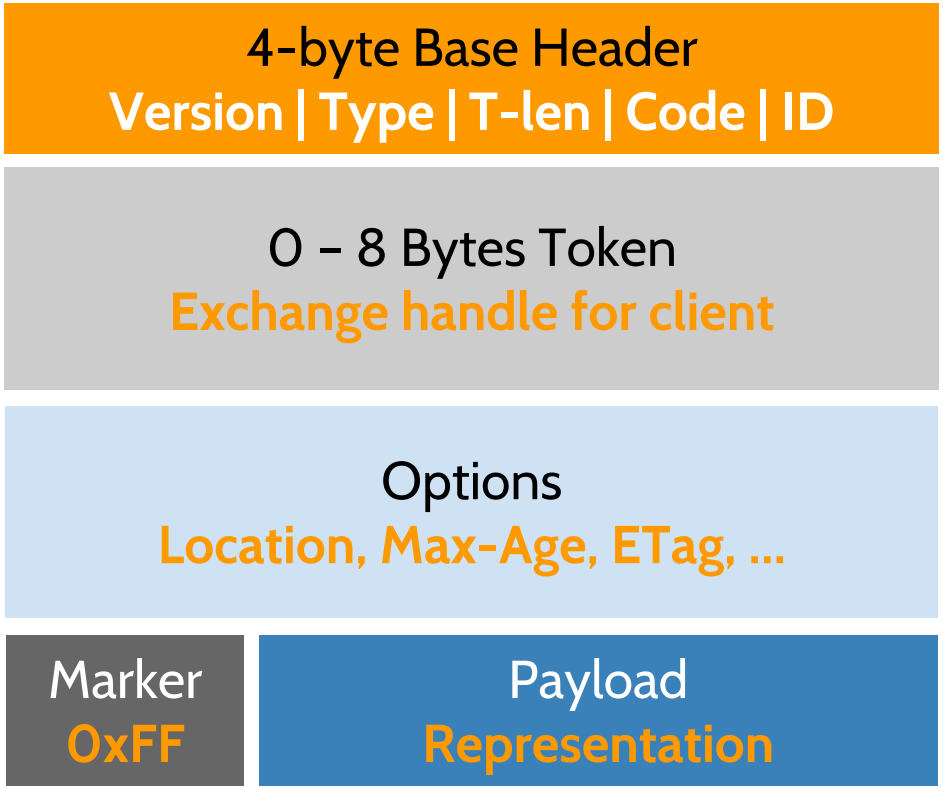
Constrained Application Protocol

Binary protocol

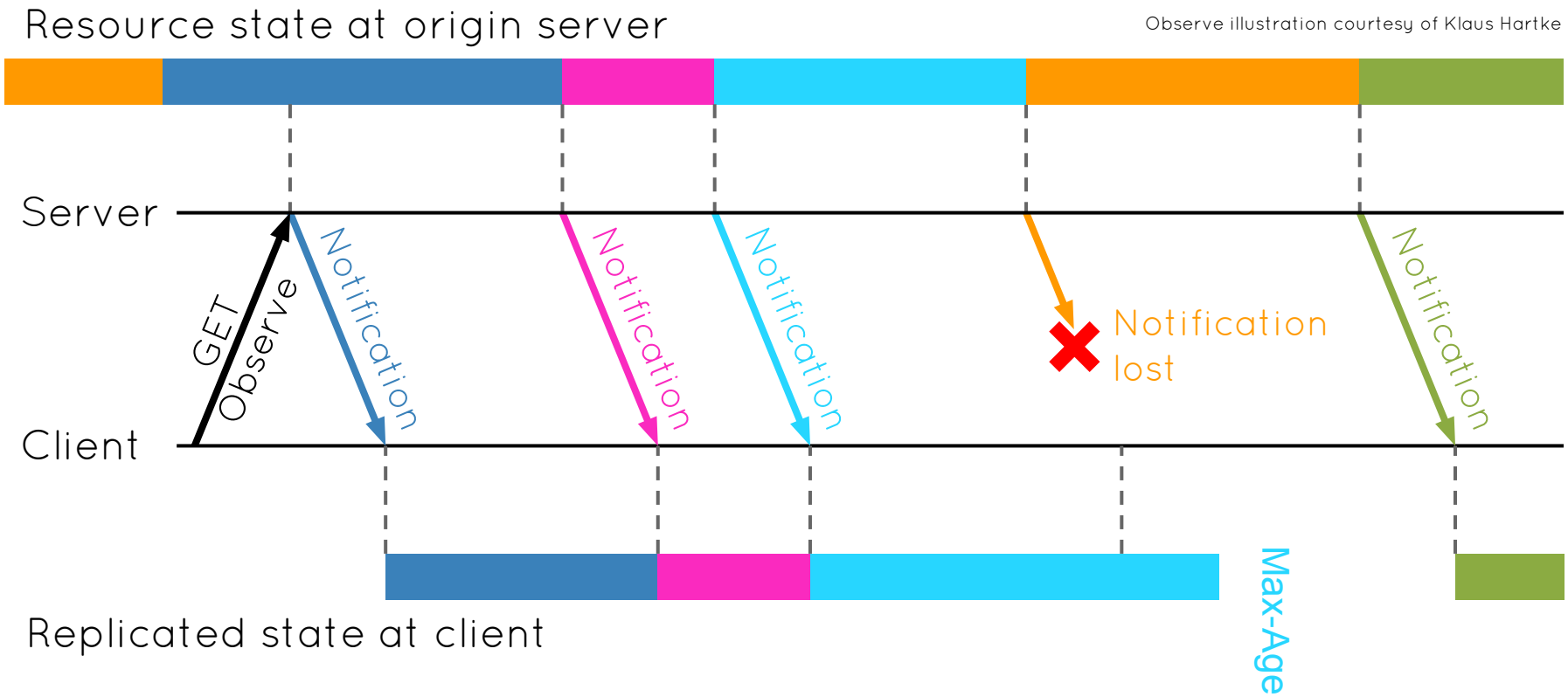
- Low parsing complexity
- Small message size

Options

- Binary HTTP-like headers



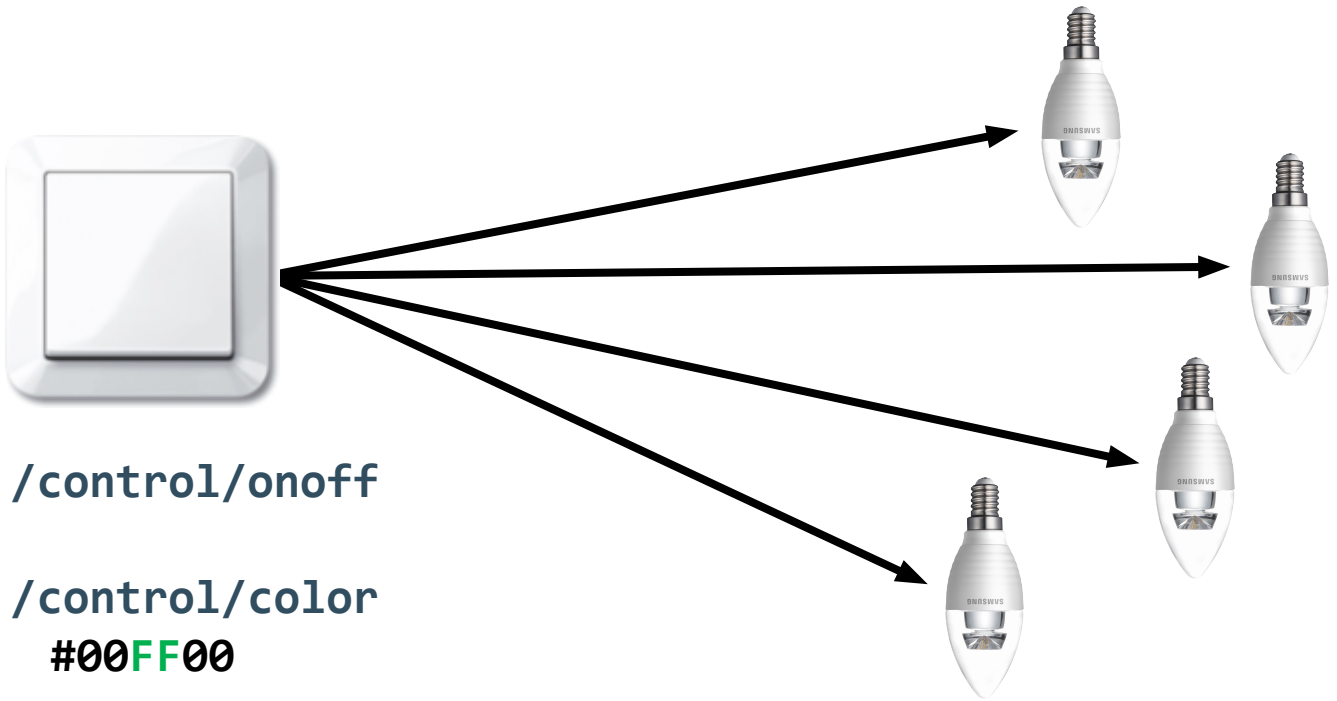
Observing resources



RESTful group communication

```
GET /status/power
```

```
all-lights.floor-d.example.com
```



```
PUT /control/onoff
```

```
PUT /control/color
      #00FF00
```

Resource discovery

Based on **Web Linking** and **Core Link Format** RFC6690

```
GET /.well-known/core
```

```
</config/groups>;rt="core.gp";ct=39,
</sensors/temp>;rt="ucum.Cel";ct="0 50";obs,
</large>;rt="block";sz=1280,
</device>;title="Device management"
```

Multicast discovery or **Resource Directories**

Alternative transports

Short Message Service (**SMS**)

Unstructured Supplementary Service Data (**USSD**)

Could power up subsystems for IP connectivity after SMS signal

Security

Based on **DTLS** (TLS/SSL for Datagrams)

Focus on Elliptic Curve Cryptography (**ECC**)

Pre-shared secrets, certificates, or raw public keys

Hardware acceleration in IoT devices

Security

IETF is currently working on

- Authentication/authorization (**ACE**)
- DTLS profiles (**DICE**)

Status of CoAP

Proposed Standard since 15-Jul-2013 – **RFC 7252**

Next working group documents in the queue

- Observing Resources
- Group Communication
- Blockwise Transfers
- Resource Directory
- HTTP Mapping Guidelines

iot
eclipse.org

Californium CoAP Framework

Unconstrained CoAP implementation

- Written in **Java**
- Focus on **scalability** and **usability**

Targetting

- IoT cloud services
- Beefier IoT devices (Java SE or SE Embedded)

Create a CoAP server

```
public static void main(String[] args) {

    CoapServer server = new CoapServer();
    server.add(new MyResource("hello"));
    server.start(); // does all the magic
}
```

A synchronous CoAP client

```
public static void main(String[] args) {
    CoapClient client1 = new CoapClient("coap://iot.eclipse.org:5683/multi-format");
    String text = client1.get().getResponseText(); // blocking call
    String xml = client1.get(APPLICATION_XML).getResponseText();
    CoapClient client2 = new CoapClient("coap://iot.eclipse.org:5683/test");
    CoapResponse resp = client2.put("payload", TEXT_PLAIN); // for response details
    System.out.println( resp.isSuccess() );
    System.out.println( resp.getOptions() );

    client2.useNONs();  // use autocomplete to see more methods
    client2.delete();
    client2.useCONs().useEarlyNegotiation(32).get(); // it is a fluent API
}
```

Asynchronous CoAP client

```
public static void main(String[] args) {
    CoapClient client = new CoapClient("coap://iot.eclipse.org:5683/separate");
    client.get(new CoapHandler() { // e.g., anonymous inner class

        @Override public void onLoad(CoapResponse response) { // also error
    resp.
            System.out.println( response.getResponseText() );
        }

        @Override public void onError() { // I/O errors and timeouts
            System.err.println("Failed");
        }
    });
}
```

Observe!

```
public static void main(String[] args) {
     CoapClient client = new CoapClient("coap://iot.eclipse.org:5683/obs");
     CoapObserveRelation relation = client.observe(new CoapHandler() {
          @Override public void onLoad(CoapResponse response) {
               System.out.println( response.getResponseText() );
          }

          @Override public void onError() {
               System.err.println("Failed");
          }
        });
     relation.proactiveCancel();
}
```

More Californium

Scandium (Sc) DTLS (TLS/SSL for UDP)

Californium (Cf) Proxy HTTP/CoAP proxy

Californium (Cf) RD Resource directory

Demo!

Device management

Monitor, **configure**, **secure**, and **update** your devices

All you need for **operating a fleet** of IoT devices

Independent of your application(s)

Interoperability is the key

You don't know yet what hardware will power your IoT projects on the field

...but you **MUST** be able to do management in a consistent way

OMA-DM

Open Mobile Alliance
Device Management

OMA-DM

Open Mobile Alliance standard for Device Mgmt

Targets mobile phones but can be used for M2M

Meant to be used by mobile network operators

OMA-DM features

Read, **write** configuration or monitoring nodes.

Trigger remote **commands** (Exec)

FUMO – Firmware Update Management Object

SCOMO – Software Component Mgmt Object

OMA-DM flaws

Phone oriented, and somewhat chatty

OMA-DM Hello world!

```
<?xml version="1.0" encoding="UTF-8"?>
<SyncML xmlns="SYNCML:SYNCML1.2">
  <SyncHdr>
    <VerDTD>1.2</VerDTD>
    <VerProto>DM/1.2</VerProto>
    <SessionID>D101</SessionID>
    <MsgID>1</MsgID>
    <Target>
    <LocURI>http://na.airvantage.net</LocURI>
    </Target>
    <Source><LocURI>IMEI:1234567890</LocURI></Source>
    <Meta>
```

```
        <MaxMsgSize>20480</MaxMsgSize>
  <MaxObjSize>512000</MaxObjSize>
  </Meta>
</SyncHdr>
<SyncBody>
  <Alert>
  <CmdID>1</CmdID>
  <Data>1201</Data>
  </Alert>
  <Replace>
  <CmdID>2</CmdID>
  <Item>
      <Source><LocURI>./DevInfo/DevId</LocURI></Source>
      <Data>IMEI:1234567890</Data>
  </Item>
```

```
    <Item><Source><LocURI>./DevInfo/Man</LocURI></Source>
       <Data>Sierra Wireless</Data>
    </Item>
    <Item><Source><LocURI>./DevInfo/Mod</LocURI></Source>
        <Data>SL6087</Data>
    </Item>
    <Item>
       <Source><LocURI>./DevInfo/DmV</LocURI></Source>
       <Data>Sierra Wireless OMC v2.0</Data>
    </Item>
    <Item>
    <Source><LocURI>.
/DevInfo/Lang</LocURI></Source><Data>en</Data>
    </Item>
    </Replace>
    <Final />
  </SyncBody>
</SyncML>
```

OMA-DM: Hello world server reply

```
<?xml version='1.0' encoding='UTF-8' standalone='no' ?>
<SyncML xmlns="SYNCML:SYNCML1.2">
  <SyncHdr><VerDTD>1.2</VerDTD><VerProto>DM/1.
2</VerProto><SessionID>D101</SessionID>
    <MsgID>1</MsgID>
    <Target><LocURI>IMEI:1234567890</LocURI></Target>
    <Source><LocURI>http://na.airvantage.net</LocURI>
    <LocName>AIRVANTAGE-SERVER</LocName>
    </Source>
  </SyncHdr>
  <SyncBody>
```

⚠ **XML QUOTA EXCEEDED...**

OMA-DM

But works :)

OMA-DM flaws: security

Weak security (MD5-HMAC)

- Use full HTTPS for higher grade security

Complex to implement correctly, no streaming due to HMAC

Unnecessary complex protocols is the shortest path to security holes

OMA Lightweight M2M

OMA Lightweight M2M

A reboot of OMA-DM but for M2M

Built on top of **CoAP**

REST API for device management

Lightweight M2M: REST API

Security, Device, Server

Connectivity monitoring

Connectivity statistics

Location, Firmware Upgrade

But objects have a **numerical identifier**

Lightweight M2M: URL

`/{object}/{instance}/{resource}`

Examples:

- "`/6/0`" the whole position object (binary record)
- "`/6/0/2`" only the altitude

Lightweight M2M: custom objects

You can define your own objects

Discoverable using CoAP Link Format

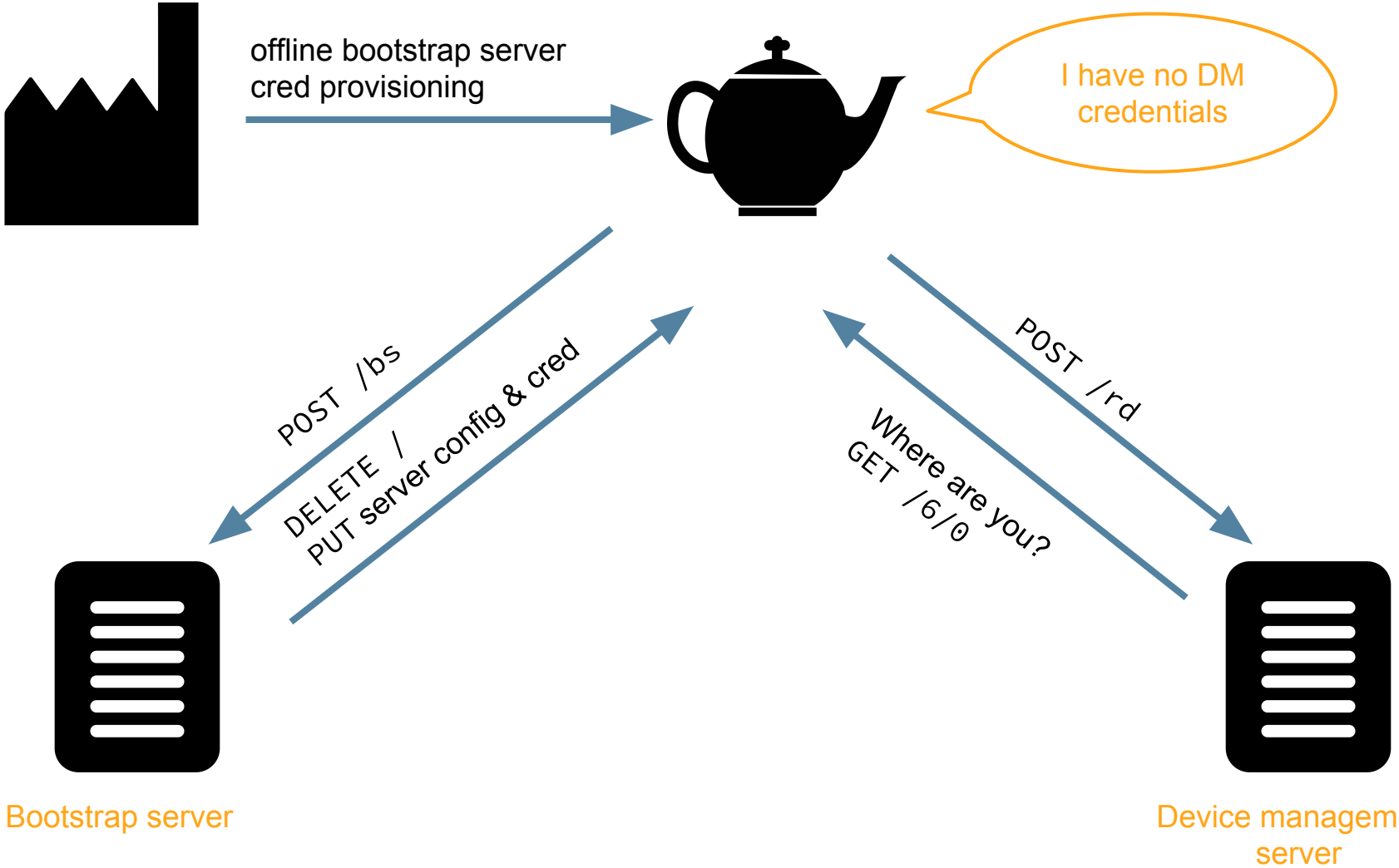
IPSO Alliance Smart Objects

Lightweight M2M: bootstrap

Well defined security keying lifecycle

How to update credential and security scheme

Keying from factory, Smart Card, or over-the-air


offline bootstrap server
cred provisioning
I have no DM credentials
POST /bs
DELETE /
PUT server config & cred
POST /rd
Where are you?
GET /6/0
Bootstrap server
Device management server

Lightweight M2M: bootstrap

Factory only knows the **bootstrap credentials**

Bootstrap server can be really **robust**

Easy path for **re-keying**

Leshan

Java implementation of Lightweight M2M

Built on top of Eclipse Californium

Under proposal phase @ Eclipse IoT

→ https://projects.eclipse.org/proposals/leshan

→ https://github.com/jvermillard/leshan

Leshan

An OSGi-friendly library for implementing server

A standalone Device Management server with a web user interface

Leshan (work in progress)

A bootstrap server

A client for your non constrained devices

Demo!

Conclusion

One protocol to rule them all is a fantasy

- IoT applications are very diverse
- Other emerging protocols: AllJoyn, XMPP, …

Lots of devices and servers are already talking **MQTT**

CoAP and **LWM2M** are well-positioned for WSNs

Open source Java implementations at **iot.eclipse.org**