Introduction to The Web of Things

Niels Olof Bouvin

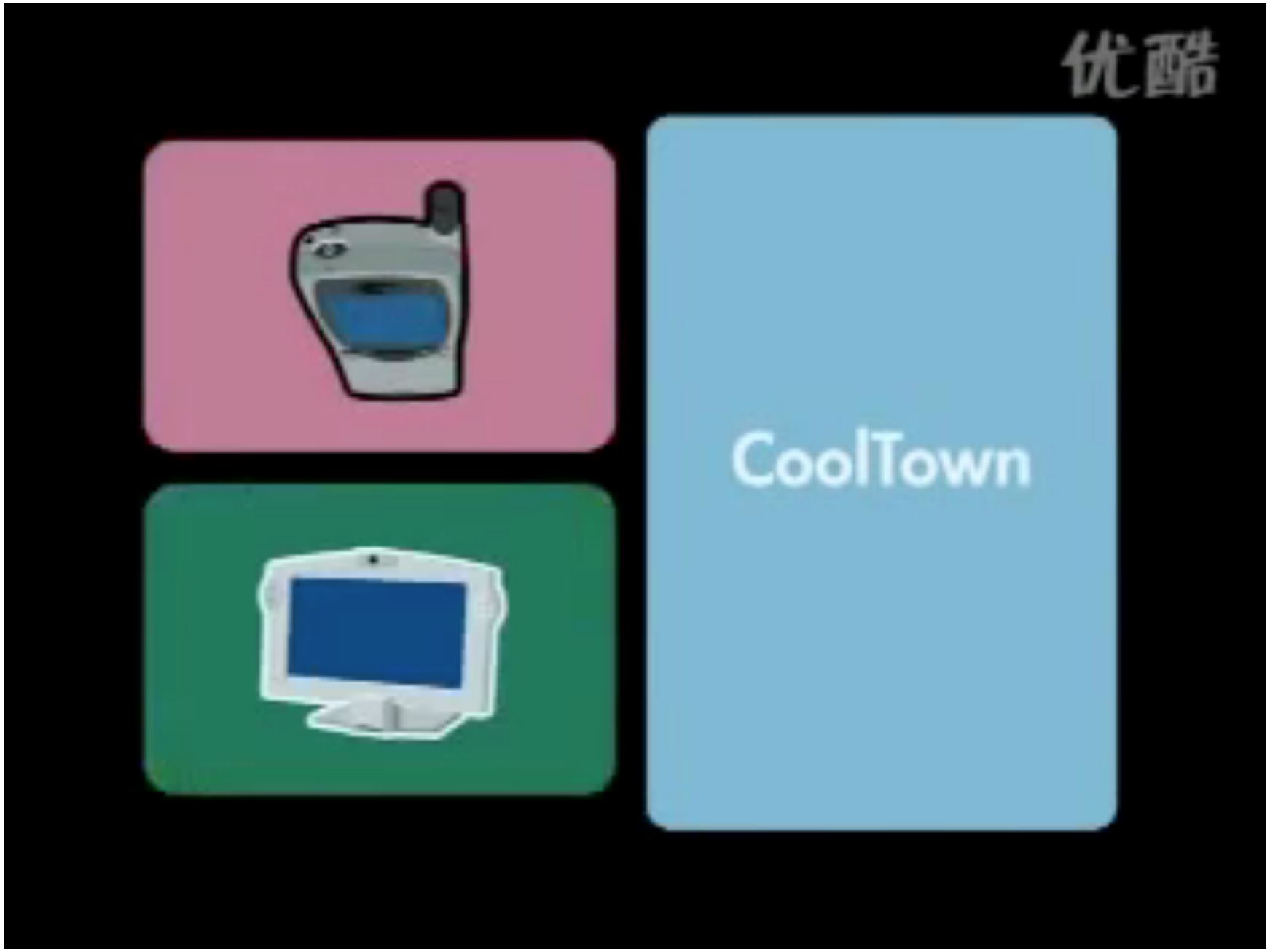
Overview

- **The Proto Web of Things**
- **The challenge of the Internet of Things**
- **The Web as IoT architecture**

Hewlett Packard CoolTown (2001)

- **Everything has a Web-presence**
  - embedded server, or
  - scannable URL (optical or IR based)
- **The state of all things can be inspected**
  - well-defined semantics ⇒ interoperability between devices
- **Devices communicate with each other within the context of their use (e.g., owner's identity)**
  - software agents can work on behalf of their users

HP CoolTown



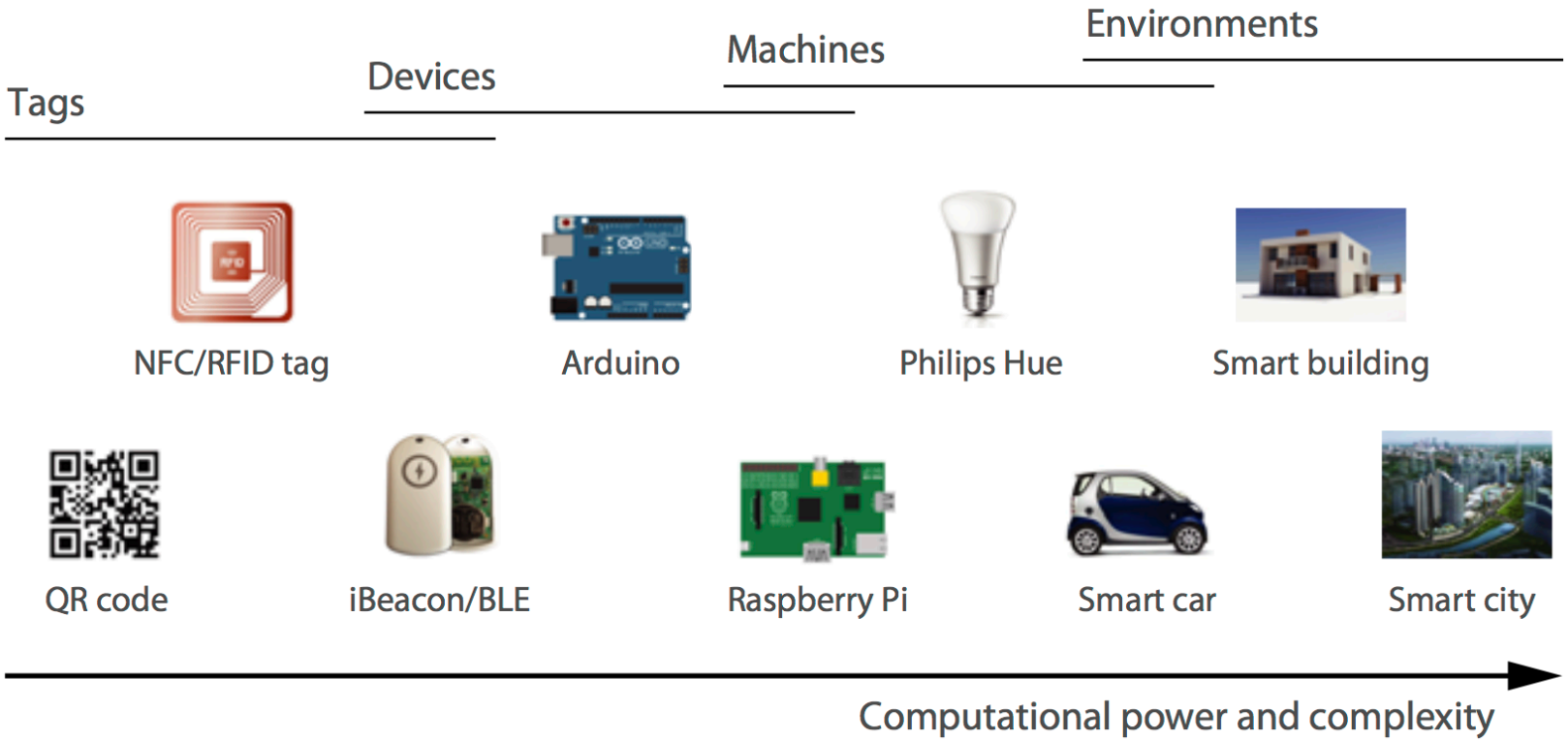
Overview

- The Proto Web of Things
- **The challenge of the Internet of Things**
- **The Web as IoT architecture**

The definition of IoT by Guinard & Trifa

The Internet of Things is a system of physical objects that can be discovered, monitored, controlled, *or* interacted *with by electronic devices that communicate over various networking interfaces and eventually can be connected to the wider internet.*

The scope of the Internet of Things



- **A wide range of uses & devices—from tags to cities**
  - all connected to the Internet in one form or another
- **A large set of associated technologies, data, and communication standards, companies, and stakeholders**

The Intranet of Things

- **The set of technologies and uses is too diverse to have one single standard of communication across the entire stack**
- **Companies are inclined to prefer their own solutions**
- **Thus, we face the Intranet of Things**
  - islands of devices
- **If a company goes out of business, is bought, or abandons a 'smart' product…**
  - the device may well stop functioning

An arbitrary example

Philip Hue	4 apps, hardware controllers	Accessible through HomeKit and Siri
AirTunes	Own app, iTunes	
Elgato sensors and switches	Own app	
Squeezebox Radio	Own app (by third party)	
Anova Sous Vide	Own app	
SmartHalo Bike Nav	Own app	
AV Equipment	Own app, Harmony remote	
Pebble watch	Own app	HealthKit

Overview

- The Proto Web of Things
- The challenge of the Internet of Things
- **The Web as IoT architecture**

The Web of Things

- **So…**
  - a myriad of devices from different vendors generating huge amounts of data
  - multitudes of users accessing and manipulating these devices and their data from a similarly broad palette of systems
- **If only we had some system that managed just that…**
- **…also known as the World Wide Web**
- **Why invent new technologies and protocols, when we have adequate, *extremely* widespread systems already?**

A return to the Cool part of Town…

- Use the web for **application layer communication**
- Use web browsers to inspect devices and their capabilities (M2H)
- Use standard web protocols, naming conventions, and data formats to access, explore, and control devices (M2M)
- Not necessary to reinvent security once again
- Better than an app for every device…

One size fits all?

- **No. There is still a need for specialised communication standards for, e.g., energy constrained devices**
- **But, as long as we maintain an application layer based on web standards, it does not matter what is underneath**
- **The web has continued to evolve and grow since its inception, but it is still highly interoperable**
  - especially since web standards became something to follow 5-10 years ago
- **Loose coupling helps ensuring continuity**

A WoT Thing

My WoT Camera | Home | Book | Code

Device: My WoT Camera!

This is a WoT Device (URL: http://devices.webofthings.io/camera/).

Description: A simple WoT-connected camera..

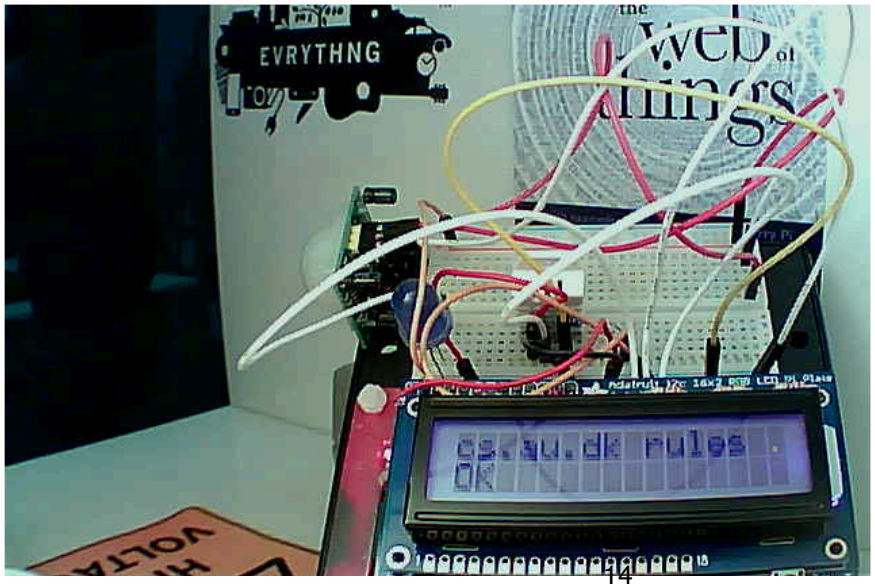
Tags: ["camera","WoT"].

Sensor: Camera Sensor

Description: Takes a still picture with the camera.

1. Type: image
2. Recorded at: 2017-09-06T21:33:00.751Z
3. Value: http://devices.webofthings.io:9090/snapshot.cgi?user=snapshots&pwd=4MXfTSr0gH

Sensor Value



URIs for smart devices

- **Devices have resources**
  - such as sensors that can take measurements,
  - states, that can be read or set, or
  - rules, that can be modified
- **These can be named systematically using URIs**
  - `http://…/sunspots/spot1/sensors/` (all sensors provided by spot1)
  - `http://…/sunspots/spot1/sensors/temperature` (spot1's temperature sensor)
  - `http://…/sunspots/spot1/actuators/leds/2` (spot1's second led)
- **Hierarchical structured, easily readable for humans, and easily transversed by machine**
  - pages contain links to parents and children

Representing resources

- HTTP provides the accept header to signify what data formats the recipient prefers/can handle
- This allows for flexible representations of the same resources, increasing levels of interoperability
- When a device is being queried by a web browser (i.e., a human), it should return HTML describing the state of the resource
- When a device is being queried by another device, JSON or XML is much better

Different representations for different purposes

Operating on a device

- **HTTP supports four main methods:**
- GET
  - retrieve the state of a resource — don't change the resource
- PUT
  - update existing resource, or create new resource with an identifier
- POST
  - create new resource, do not specify identifier
- DELETE
  - remove a resource

(A few) HTTP Status Codes

- **`200 OK`**
  - Standard response for successful HTTP requests. The actual response will depend on the request method used. In a GET request, the response will contain an entity corresponding to the requested resource. In a POST request the response will contain an entity describing or containing the result of the action
- **`201 Created`**
  - The request has been fulfilled and resulted in a new resource being created
- **`301 Moved Permanently`**
  - This and all future requests should be directed to the given URI
- **`404 Not Found`**
  - The requested resource could not be found but may be available again in the future.

Many clients, weak device?

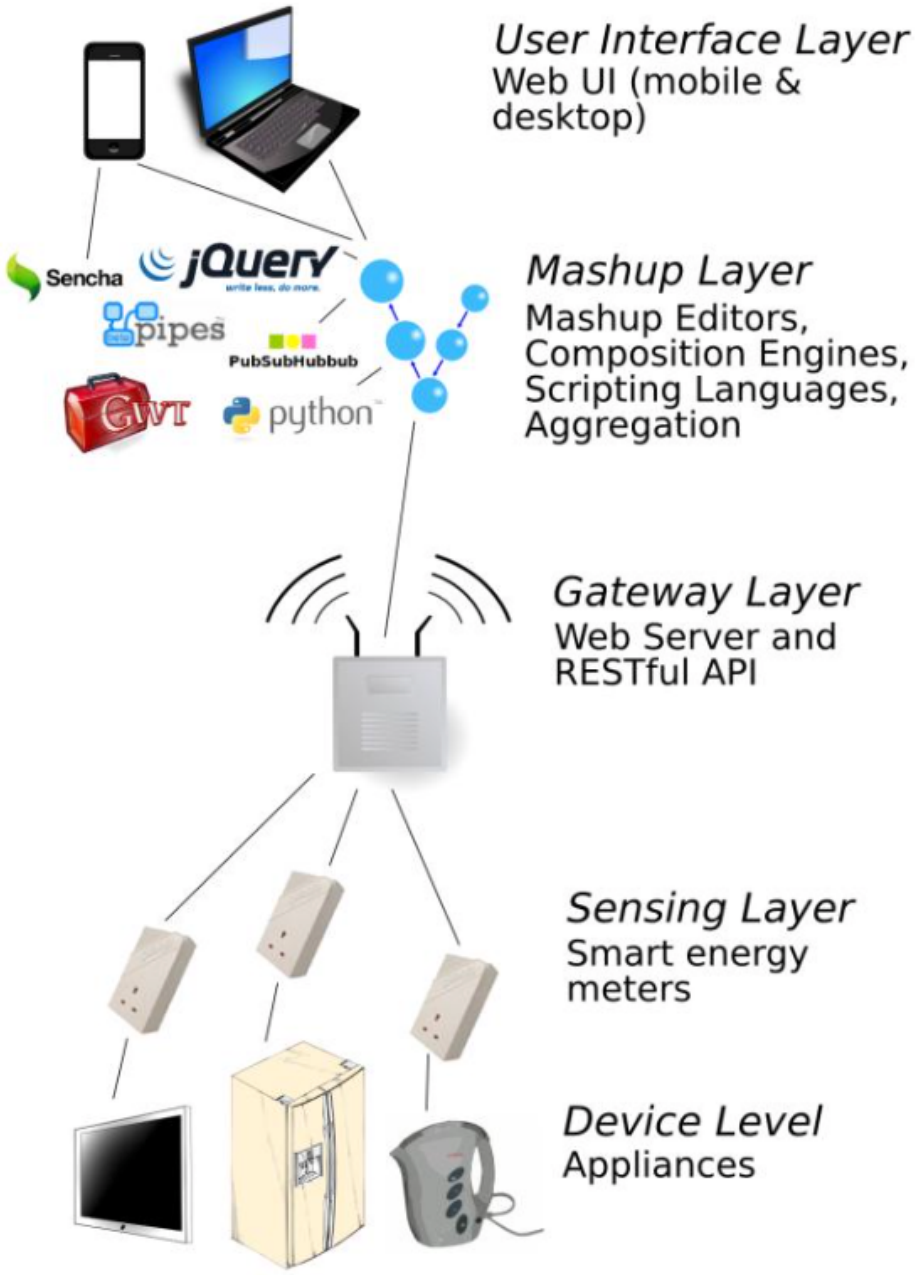
- Register device with Atom feed on server
- Whenever a specified state changes on the device, it is pushed to the server
- The resource is published by the server, conserving the device's resources

Streaming data?

- Connect through a WebSocket
- Pass data back and forth as needed, possibly through an intermediary

Wrapping existing systems

- Devices not adhering to the RESTful approach can be handled through the use of a gateway
- Thus, proprietary systems can be wrapped and used in a wider context



Advantages of Web of Things

- **Existing Web based tools, frameworks, and methodologies just work**
  - including proxies and caching
  - and, crucially, security
- **Simple to do "mash-ups", connecting data sources with other tools**
- **Easy to do development by exploration**
- **If a device has a Web browser, it can be used to explore the Internet of Things**

Disadvantages of the Web of Things

- **HTTP is a fairly heavy protocol for small devices**
  - though small devices are getting more powerful
- **Streaming (sensor) data is not what HTTP was originally designed to do**
  - Web Sockets helps, as will HTTP/2
- **Discovery**
  - devices may be described using micro formats, which can be systematically indexed
- **Security**
  - use existing frameworks, including authentication through social network sites
  - hide devices behind a proxy that requires proper authentication from, e.g., Facebook

The *Internet* of Things

- The premise and success of the Internet is founded on open standards
- Open, shared standards are required for the Internet of Things to succeed outside of specialist or vendor-specific domains
- Hopefully, if no shared standards can be agreed upon in advance, market realities will force interoperability

Milestone 1 notes

- **Programming language**
  - anything you want, but choose something with a low memory footprint
- **Frameworks**
  - Not ok to use a P2P framework; perfectly fine to use, e.g., Express for REST purposes
- **Discovery**
  - the first Kademlia peer is very useful
- **Live demo: Show me what you've got**
- **Bring a hardcopy of your API, and explain it**
- **Next Thursday afternoon and Friday**