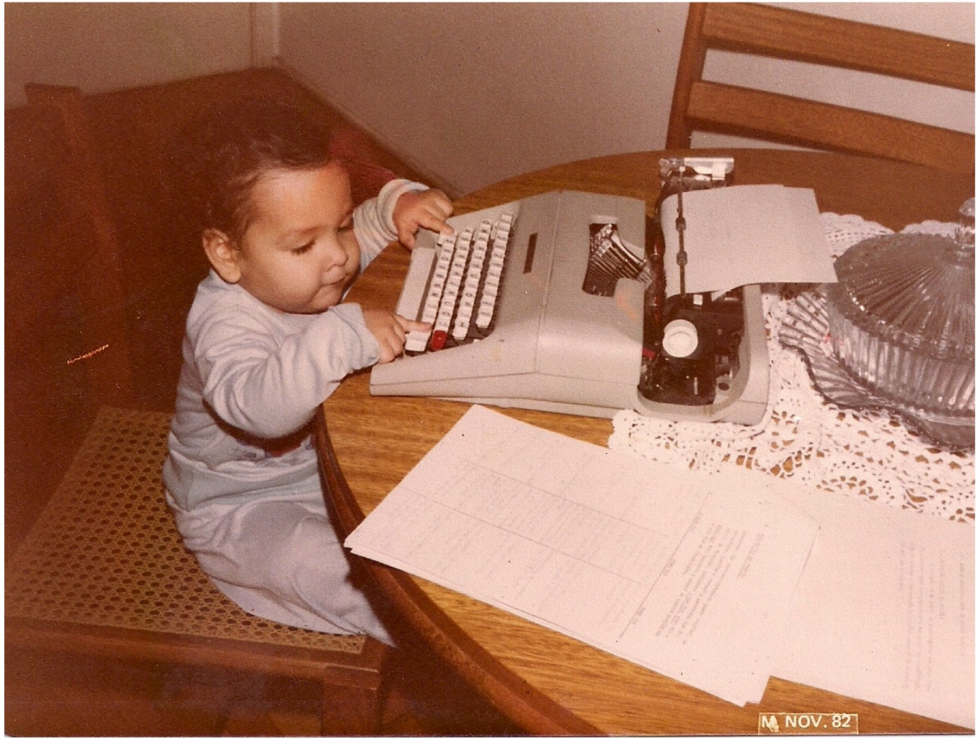
IoT API: A Vision for 2020

Thiago Macieira

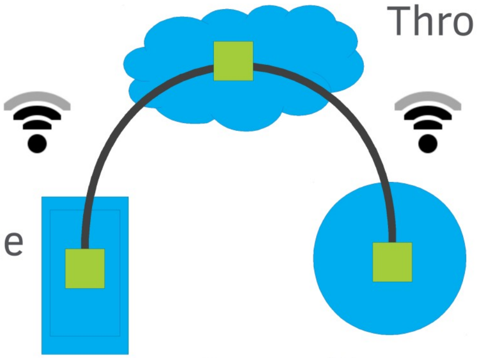
Embedded Linux Conference / OpenIoT Summit – April, 2016

Who am I?



What is the Internet of Things?

“Cloud orchestration”



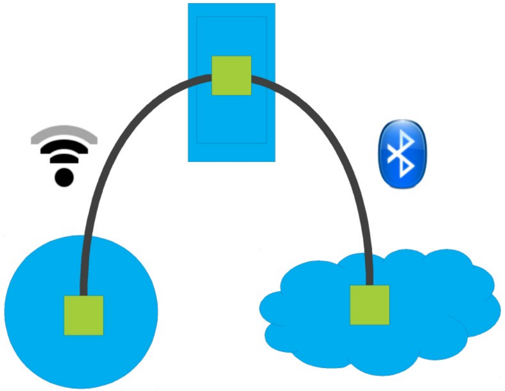
Through the cloud

An app on a phone

Controlling one “thing”

“Phone orchestration”

An app on a phone



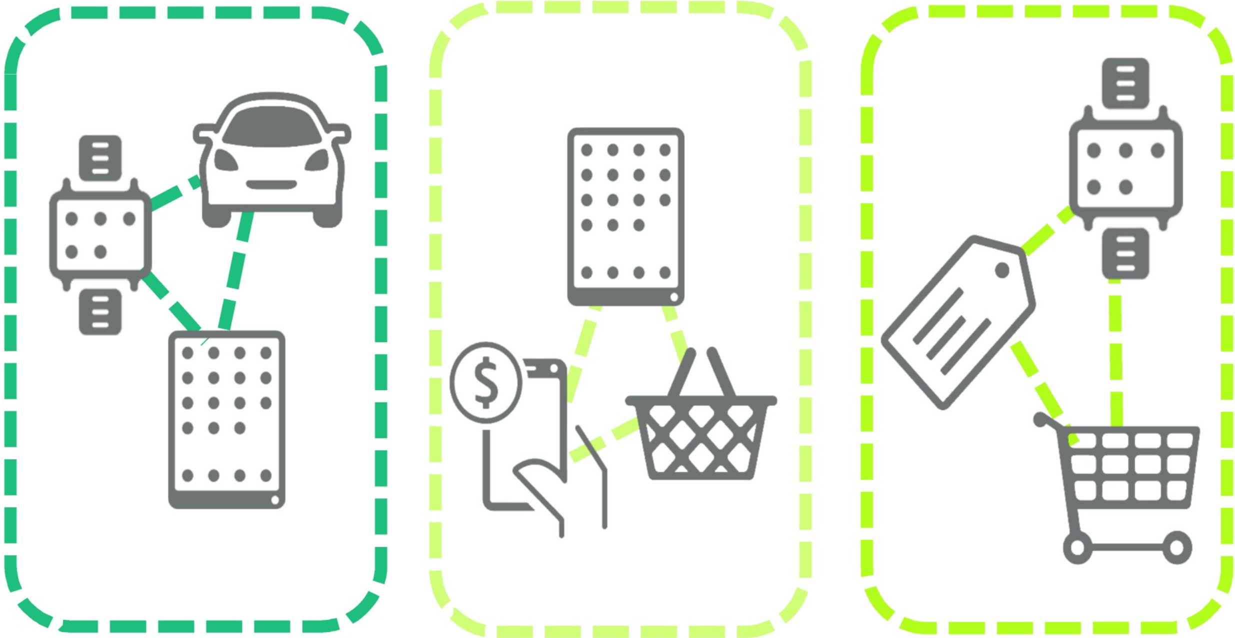
Reading from one “thing”

Sending to the cloud

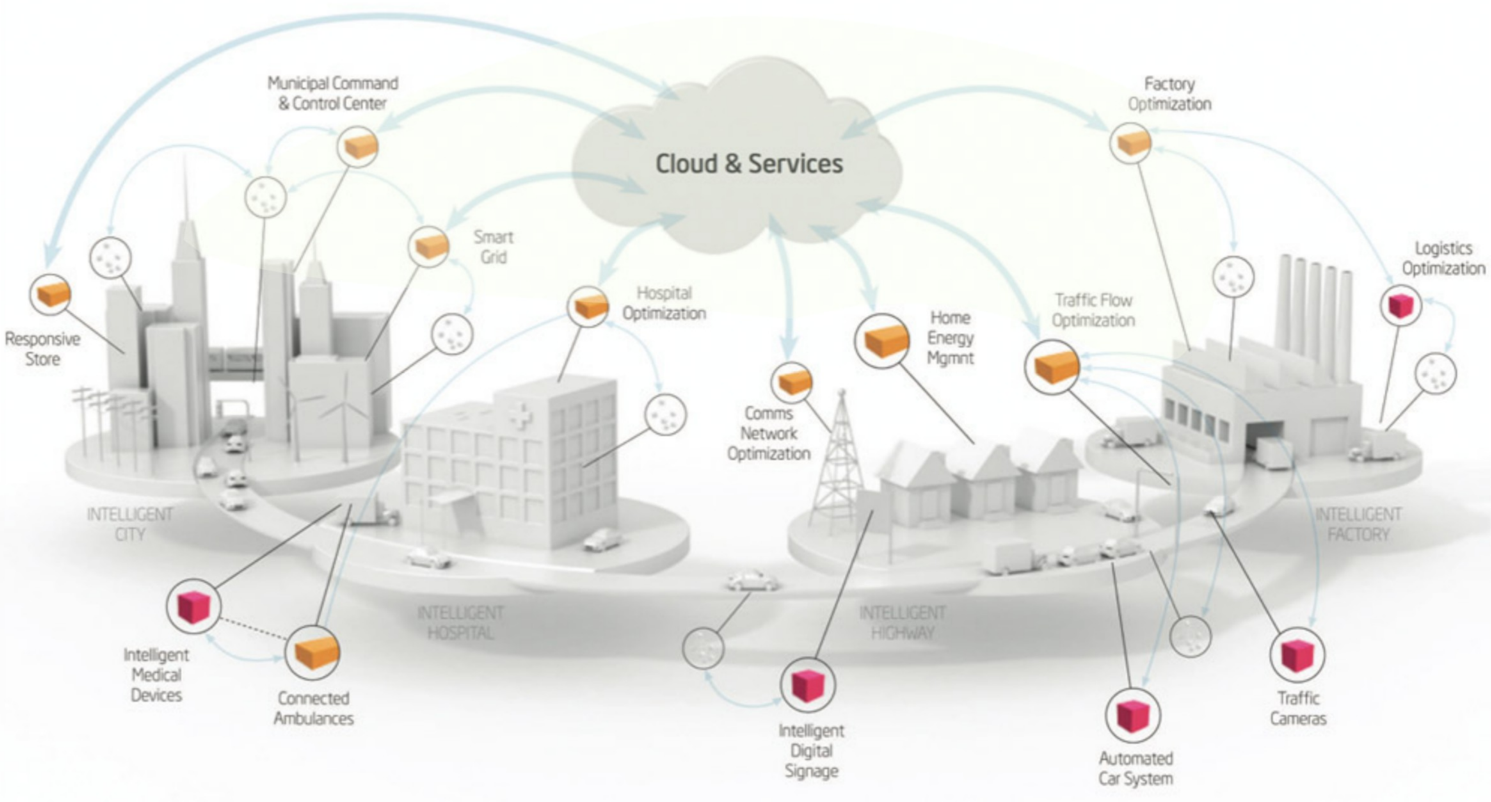
The challenge of IoT communications

- The Internet of Things is currently evolving as "Isolated Islands of Things."

- Severely limits the value of IoT.



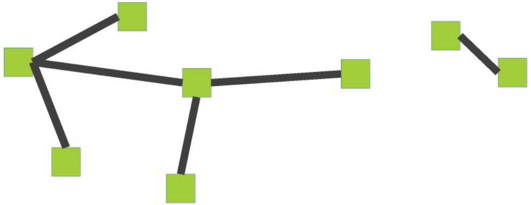
IoT can’t deliver potential without the cloud



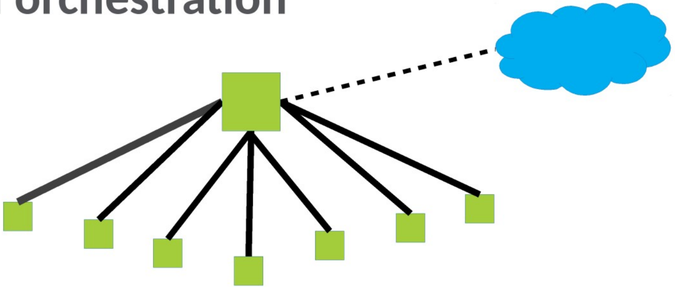
source: http://m.eet.com/media/1174500/internet-of-things-lg.jpg

What are we leaving on the table?

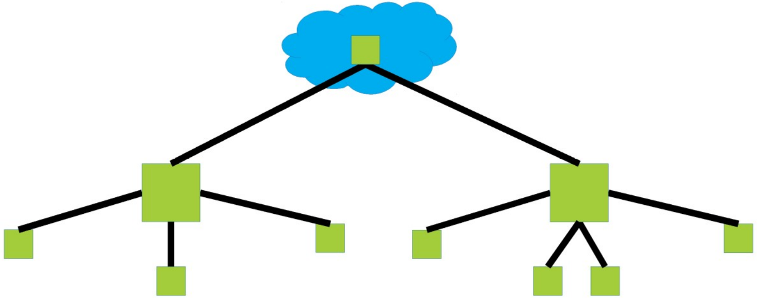
Distributed "thing" to "thing"



"Local orchestration"



Private (locally-deployed) clouds



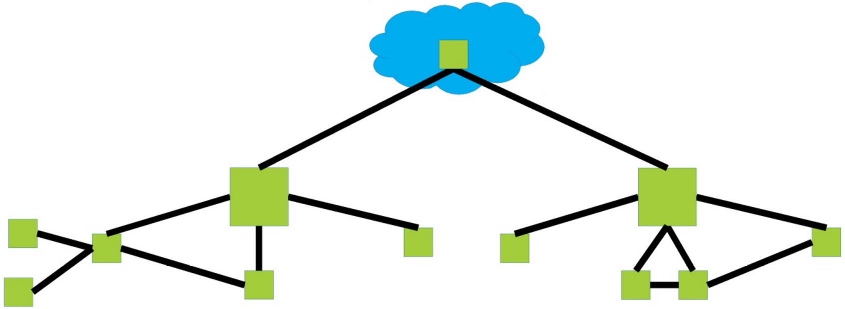
And a lot more

- Smart grids
- Intermittent access
- …

The vision for IoT

- **Small, medium and big devices**
- **Local orchestration**
- **Cloud for extra services**

Added value, on top of what devices do by themselves



Vision: challenges

- **Easy to use**
  - Cloud integration
- **Flexible**
- **Code reuse (prototype → product)**

Programming the “things”

- Operating system (including drivers)
- Device management frameworks
- Communication frameworks
- Ease-of-development frameworks

The Zephyr* Project

- Small OS for MCU-class devices
- Open Source, a Linux Foundation collaborative project
- Professionally developed

https://zephyrproject.org

Zephyr sample application

```
#include <zephyr.h>
/* RGB LED states - inverse of logic value */
#define LED_ON   0
#define LED_OFF  1
/* RGB LED pins */
#define LED_R    22  /* Port B pin */
#define LED_B    21  /* Port B pin */
#define LED_G    26  /* Port E pin */
/* SW3 states- inverse of logic value */
#define SW_ON    0
#define SW_OFF   1
/* SW3 pin */
#define SW3       4  /* Port A pin */
int main(void)
{
    /* Replace this with user task code */
    while (1) {
        /* set to next LED state if switch is not pressed */
        gpio_pin_read(gpio_a, SW3, &sw_value);
        if (sw_value == SW_OFF) {
            if (++led_value > 7)
                led_value = 0;
        }
        switch (led_value) {
            case 0: // OFF
                PRINT("LED is OFF\n");
                gpio_pin_write(gpio_b, LED_R, LED_OFF);
                gpio_pin_write(gpio_b, LED_B, LED_OFF);
                gpio_pin_write(gpio_e, LED_G, LED_OFF);
                break;
```

Soletta™ Project

- **Higher level API for developing devices**
  - I/O
  - Comms (CoAP, HTTP, MQTT, LWM2M, OIC/OCF)
  - Persistence, storage
  - Device management
- **For Linux, RIoT, Contiki, Zephyr**

Soletta™
Project

https://github.com/solettaproject

Soletta sample application

```c
#include <soletta.h>

static bool discover_resource(struct remote_light_context *ctx)
{
    struct sol_coap_packet *req;
    struct sol_network_link_addr cliaddr = { };

    req = sol_coap_packet_request_new(SOL_COAP_METHOD_GET, SOL_COAP_TYPE_CON);
    if (!req) {
        SOL_WRN("Looks like we have no space");
        return false;
    }

    sol_coap_add_option(req, SOL_COAP_OPTION_URI_PATH, "a", sizeof("a") - 1);
    sol_coap_add_option(req, SOL_COAP_OPTION_URI_PATH, "light", sizeof("light") - 1);

    cliaddr.family = SOL_NETWORK_FAMILY_INET6;
    sol_network_addr_from_str(&cliaddr, "ff02::fd");
    cliaddr.port = DEFAULT_UDP_PORT;

    sol_coap_send_packet_with_reply(ctx->server, req, &cliaddr, discover_reply_cb, ctx);

    return true;
}
```

Also JavaScript and Flow programming

```
btn(Button)
led(LED)
tog(boolean/toggle)

btn OUT -> IN tog OUT -> IN led
```

MRAA

- **I/O library for Linux**
  - Sensors, actuators
  - GPIO, I^{2}C, AIO, PWM, SPI, UART, …
- **C, C++, Python, JS**

https://github.com/intel-iot-devkit/mraa

The big guns: Node.js*

- Very easy to develop with
- Incredible wealth of modules

Intel® IoT Services Orchestration Layer

- **JavaScript-based (Node.js, Node-RED engine)**
- **Graphical editor for Flow Based Programming**
- **Distributed management**

https://github.com/01org/intel-iot-services-orchestration-layer

IoTivity* Project

- Communications, communications, communications
- Services
- Data models from the Open Connectivity Foundation

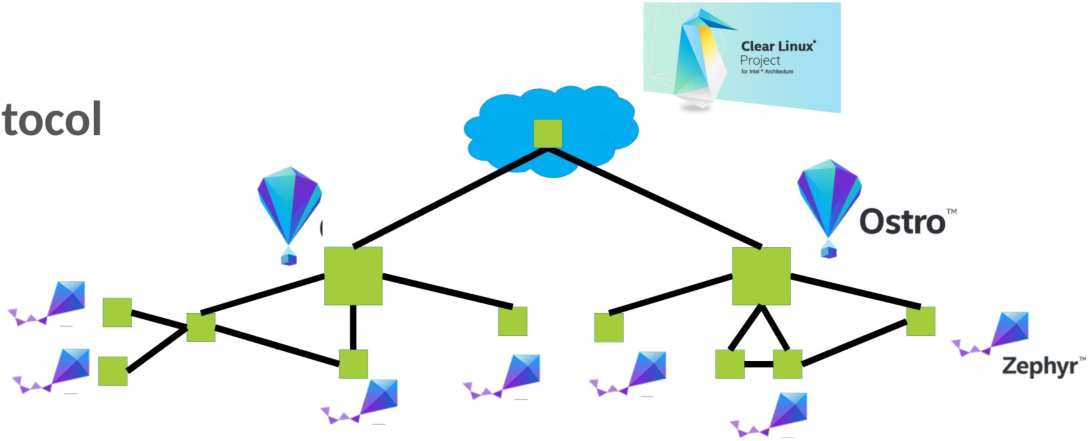
https://iotivity.org

IoTivity bindings

- C core
- C++ higher level API
- Java (Android)
- JavaScript (Node.js)

Implementing the vision

- **Small devices running Zephyr and Soletta**
- **Bigger devices running Linux, Soletta and/or SOL**
  - Downloadable “apps”
  - Management by the user
- **Communicating via OCF protocol**



KILL ALL
HUMANS
YOU MEATBAGS HAD YOUR CHANCE
Bender
2016

Thiago Macieira
thiago.macieira@intel.com
http://google.com/+ThiagoMacieira