



m2m

eclipse.org

**Open Source
building blocks for the
Internet of Things**

Benjamin Cabé
JFokus 2013



Who I am



- Benjamin Cabé
- Open Source M2M Evangelist at Sierra Wireless
- Long-time Eclipse lover



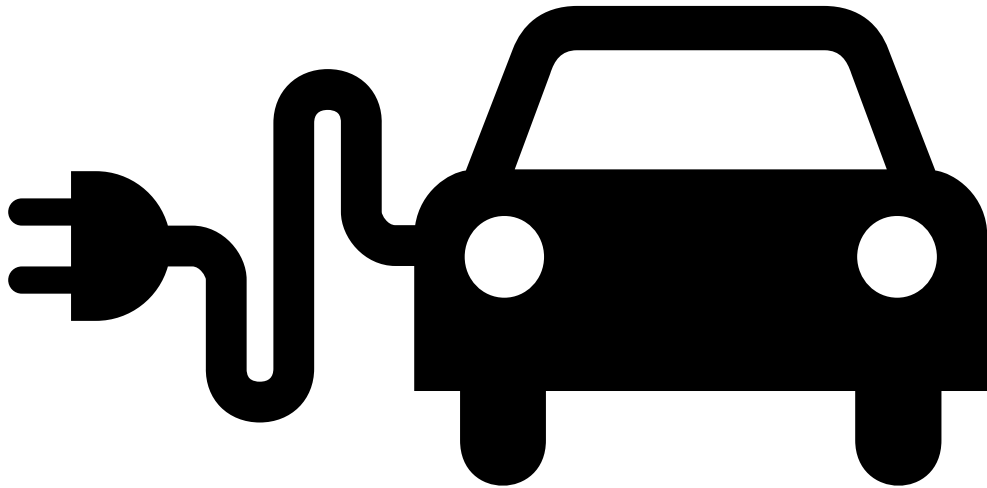
SIERRA
WIRELESS™

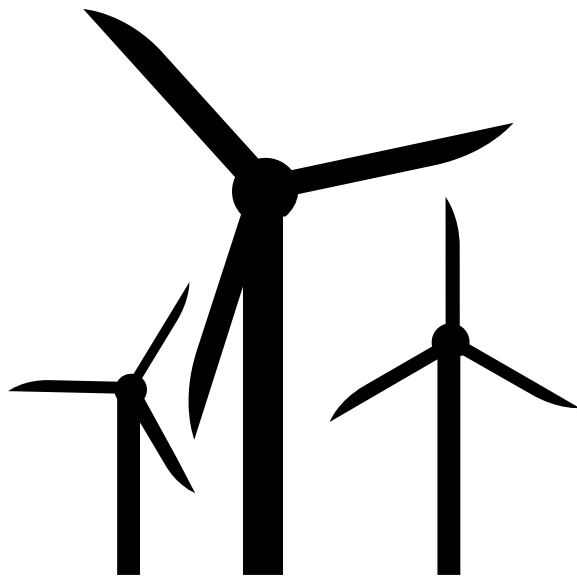
M2M? IoT?

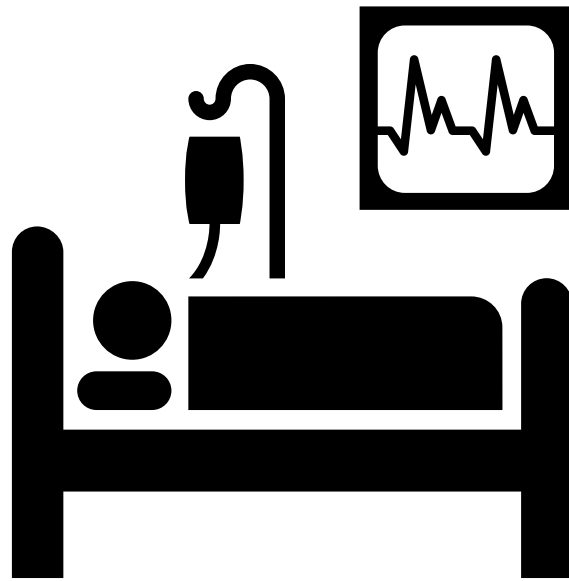


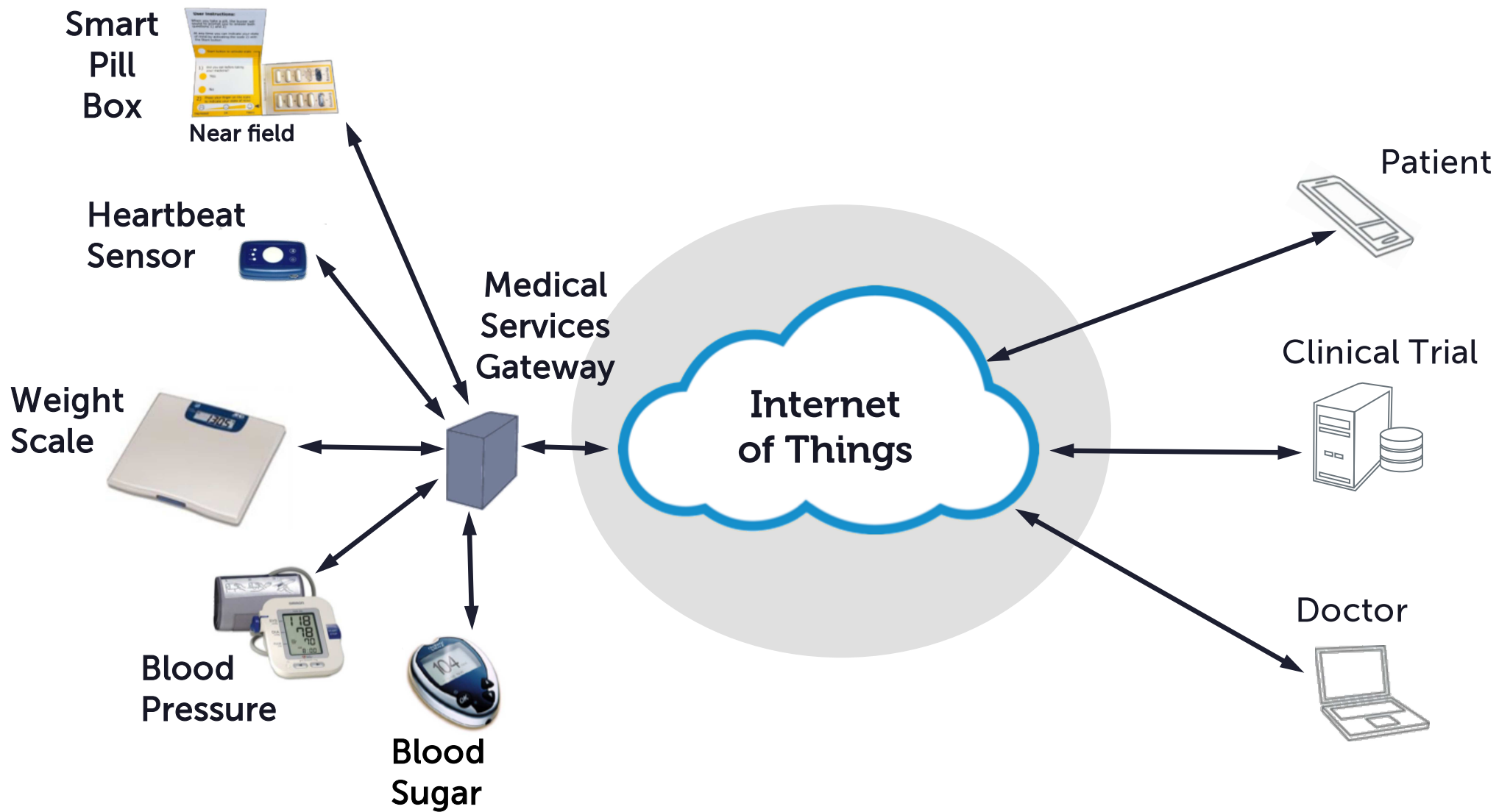
Technology that supports
wired or wireless
communication
between devices



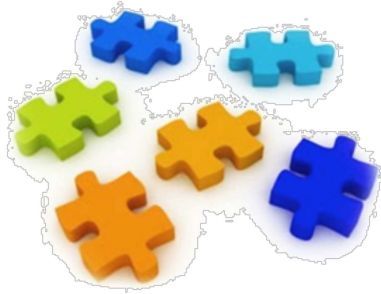








However...



The market is fragmented

- Hardware, software, protocols... all different, independent
- Lack of integration... between devices, to enterprise systems



M2M development is complex

- Many different skills required... Hardware, Embedded, IT network, Telecom, web
- No common architectural guidelines



Current options are closed

- Monolithic solutions... device specific, app specific, market specific
- Proprietary SDKs, protocols, potential vendor lock-in



m2m

eclipse.org

3 projects

Framework

mihini

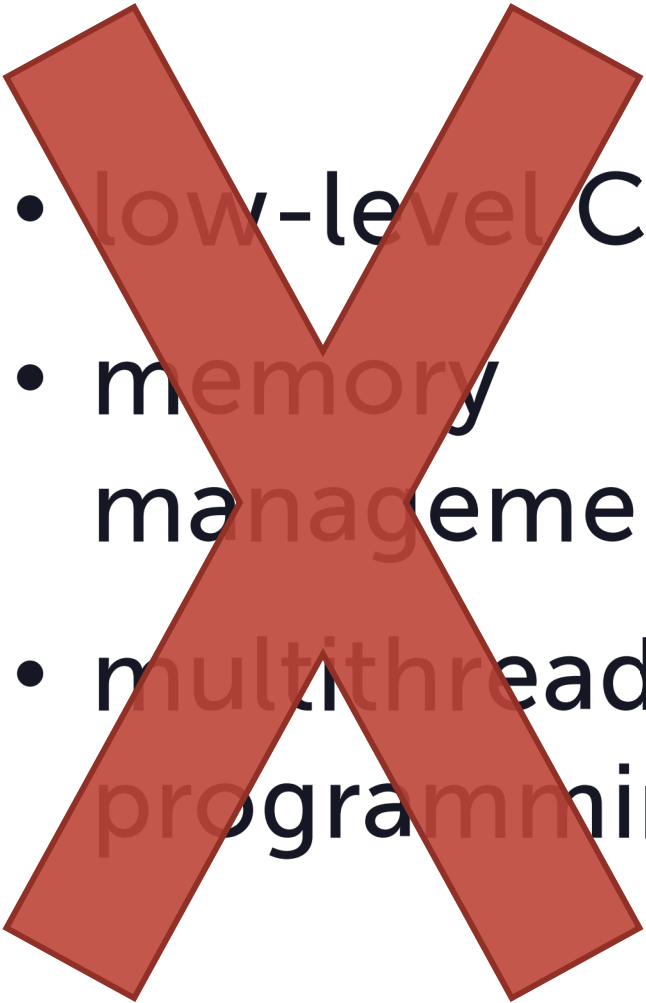
Protocols

paho The paho logo consists of the word "paho" in a lowercase, sans-serif font, followed by a white speech bubble icon on a dark background.

Tools

koneki

M2M embedded programming

- 
- low-level C
 - memory management
 - multithreaded programming

- read sensor values
- control actuators
- consolidate data
- communicate

Example: Sending an SMS

```
int main()
{
    unsigned char char1[10];
    unsigned char char_buf[8]="AT+CS0\r\n";
    // unsigned char sms_buf[20] = "AT+CMGS="xxxxxxxx";

    int wc_fd;
    /******* Init of serial port *****/
    wc_fd = init_wc(wc_fd);
    sleep(3);
    //writing to serial port
    write(wc_fd, char_buf, sizeof(char_buf));
    usleep(40000);
    //reading from serial port
    read(wc_fd, char1, sizeof(char1));

    sleep(2);
    close(wc_fd);

    return 0;
} // end of main

// initialization of serial port
```

```
sms.send(
    '+33612345678',
    'My SMS',
)
```

struct termios options;

```
ttys5_fd = open("/dev/ttyS5", O_RDWR );
if (ttys5_fd < 0)
```

Simplify M2M programming



What is Lua?

- High-level programming language
- Scripting
- Simple
- Extensible
- Portable

Extensible by design

- Small
 - Trivial syntax and reduced keyword set
- Simple but not stupid
 - Simple enough for new users, powerful enough for advanced users (first-class functions, garbage collection, closures, tail calls, coercion, coroutines, metatables)
- Lua core is tiny
 - Compiled size is ~150kB
 - Lua uses libraries for its extensions

Lua vs. other high-level languages

- Same core features as Python, Ruby, Javascript
- Better concurrency management
 - Built-in – doesn't rely on the OS
- Cutting-edge execution technology & performances

Lua vs. other high-level languages

- **Restricted set of libraries**
 - Stay simple, the developer brings his own
- **Designed for C integration**
 - Performance
 - Legacy

Lua for embedded and M2M?

- High-level languages usually trade hardware resources for development & maintenance resources

Lua allows to reconcile high-level languages accomplishments with embedded constraints

You need an IDE!

- Project structure
- Syntax coloring
- Content assist
- Code navigation
- Code formatting
- Documentation
- Code templates
- Debugger
- Remote development
- Embedded interpreter

koneki



June 2012: first release (0.8)

Dec. 2012: 0.9 release

June 2013: graduate w/ Kepler

50,000+ installations already! (Feb. 2013)

Using Koneki LDT for
remote Lua development

DEMO



koneki **101**

- <http://www.eclipse.org/koneki/ldt>
- Download standalone IDE
- Or install in existing Eclipse
 - from Juno repository (0.8.2)
 - from nightly/milestones repositories (0.9+)
- Execution environments on Koneki wiki
 - <http://goo.gl/f6T80>
- Contact the team
 - <http://eclipse.org/forums/eclipse.koneki>

How do we communicate?

KW FREDERIKST. LUXEMBOG. STAVANG. HALLE. DRESDEN. BRUSSEL. WARSCHAU. LIMOG. DAVENTRY. LEIPZIG. KW
NOZZA. SAARBR. OERUN. STRASSB. HL. VERS. REICHENB. MUNCHEN. FINN. PRAG. NIMOR. BEN.
VATIKAN. BORDEAUX. SZCZECIN. STAGSHAW. KESCHIN. DITSCH. S. DITSCH. S. MOS. EDGE. BERLIN I. POTSDAM.
KRAKOW. KAUNAS. GRENOBLE. BOLOGNA. HAMBURG. ERFURT. SOTTENS. SURE. K.M. STADT. SIMFEROPOL.
NURNBERG. KIEL. BREMEN. KOSICE. PLAIEN. BRUNN I. ROM. HL. VERS. MARSEILLE. SOFIA. BUDAPEST.
HANNOV. MTE. CARLO. CROWD. FALUN. KRASNODAR. LWOW. SOFIA. SCHWER. GREPSW. WIEN. RADIO-DEU.
MW PRAG. KALLINGB. OSLO. DROTW. DITSCH. S. BRASOW. MW
LW MINSK. MOSKAW. LUXEMBURG. RIEW. MOTALA. STRASSBURG. LW

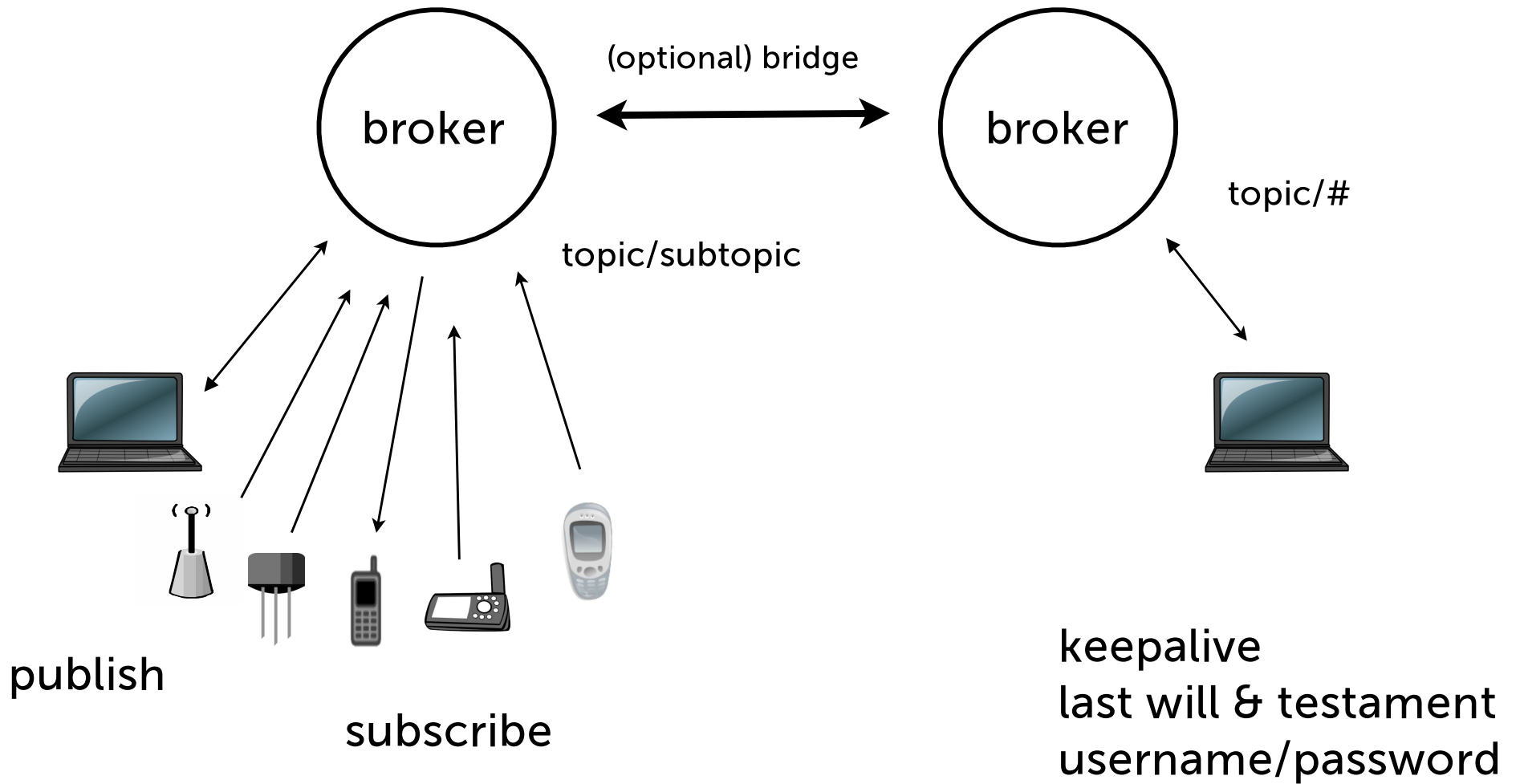
UKW AUS 0 LW MW KW UKW **UKW**





- Messaging protocol
- Low-bandwidth / Low-power
- Payload agnostic
- Adjustable QoS
- Large ecosystem

MQTT





paho 101

- <http://eclipse.org/paho/>
- Eclipse Paho delivers clients for MQTT in C and Java
 - <http://eclipse.org/paho/download.php>
- Lua client available soon
 - https://github.com/geekscape/mqtt_lua
- MQTT view in Eclipse
 - <http://git.eclipse.org/c/paho/org.eclipse.paho.esf.git/>
- Free broker hosted at Eclipse: **m2m.eclipse.org**
- Contact the team
 - [paho-dev](#) mailing-list



Lua VM + MQTT client to go?

Application framework for M2M

- Set of libraries providing building blocks to develop M2M applications:
 - Serial and I/O management,
 - Networking (FTP, HTTP, e-mail, ...),
 - GPS,
 - Cryptography,
 - Modbus,
 - Local storage
 - etc.

mihini

<http://www.eclipse.org/mihini>

Smart agent for M2M

- M2M data queues
- Network bearers
- Device management
- Application container
- Application configuration

mihini

<http://www.eclipse.org/mihini>

Asset management

- User applications use an API to communicate with Mihini
 - Send data or events
 - Register listeners to handle data writing or commands
- The Mihini agent takes care of network connection, buffering and reliable storage of unsent data, etc.

Asset management

```
local gh_asset = asset_mgt.newAsset("greenhouse")
```

```
gh_asset:start()
```

```
gh_asset:pushdata("sensors.temperature", 22, "now")
```

```
gh_asset:pushdata("sensors.humidity", 89, "hourly")
```

Device management

- A Tree Manager presents device's data as
 - variables,
 - organized in a hierarchical tree,
 - that can be read, written, and monitored for changes by user applications.
- Standard paths for modem, network settings, ...

Device management

```
local devicetree = require 'devicetree'  
local APN        = 'system.cellular.apn.apn'  
local RSSI       = 'system.cellular.link.rssi'  
  
local apn = devicetree.get (APN)  
log(LOG_NAME, "INFO", "Configure APN: %s", apn)  
  
local function print_callback (values)  
    for name, value in pairs(values) do  
        log (LOG_NAME, "INFO",  
            " - Variable %s changed: %s", name, value)  
    end  
end  
  
devicetree.register(RSSI, print_callback)  
  
devicetree.set(APN, 'foo')
```

Application Management

- Language-agnostic application container
 - install/uninstall
 - start/stop, auto-start on boot
 - restart on failure
- Agent handles over-the-air software download and update mechanism
- Remote script execution



mihini **101**

- <http://www.eclipse.org/proposals/technology/mihini>
- <http://eclipse.org/mihini>
- Code will be available (very) soon
 - Initial contribution is pending IP review at Eclipse



**Let me
show you!**

A very common use case

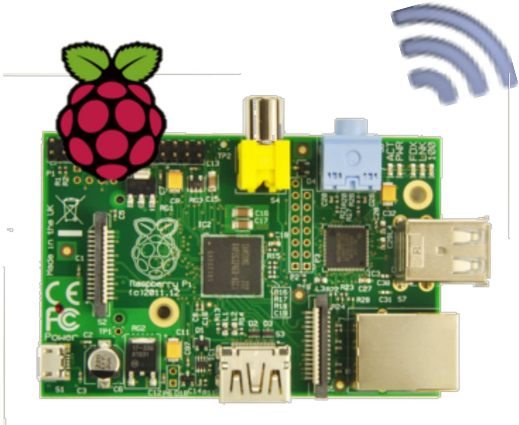
- **Greenhouse business**
 - Connect gardening equipment
 - Remote monitoring of sensors
 - Remote control
- **M2M Gateway not selected yet, neither is the rest of the equipment (PLCs)**



MQTT broker

Raspberry Pi

Mobile phone

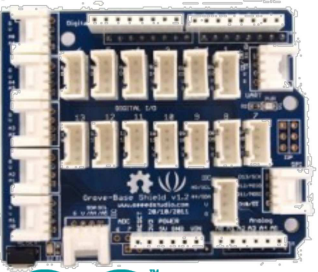


Modbus RTU

humidity
temperature
illuminance



light ON/OFF



Two Lua applications

Embedded

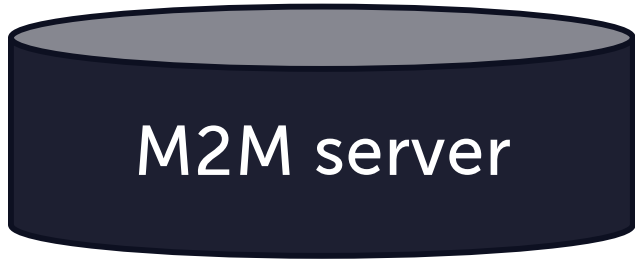
mihini

- Uses Modbus library to communicate w/ Arduino
- Collects sensor data/ controls actuators
- Publishes MQTT messages
- Subscribe to commands

Mobile

Corona SDK

- Subscribes to MQTT messages
- Displays sensor data with a fancy UI
- Publish command to switch on/off the light



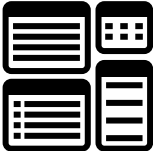
etc...



Rugged wireless gateways



Mobile phones



Web applications



IT applications

...

Modbus

control sensors & actuators



Roadmap

- **Protocols**
 - M3DA (Micro M2M Data Access) – see the spec. at http://wiki.eclipse.org/Mihini/M3DA_Specification
- **REST API**
 - Ease the communication of 3rd party apps with the Agent
 - Provide better tooling
- **Polyglot framework**
 - C and Java on their way

Join the party!

m2m.eclipse.org

m2m.eclipse.org is where you can learn about the technologies developed at [Eclipse](#) to make Machine-to-Machine (M2M) development simpler.

These technologies aim at establishing an open, end-to-end, M2M stack.

< mihini >

Mihini

Mihini will deliver an embedded runtime running on top of Linux, exposing high-level Lua API for building M2M applications.

Frameworks



Deliver an embedded extensible runtime enabling M2M vertical applications.

In order to enable the creation of M2M apps on communicating embedded devices, we provide a complete framework enabling device management, software updates, ...

[More »](#)

Protocols



Provide Open Source implementations of standard M2M protocols.

Currently, we provide tools and libraries for:

- [MQTT](#) messaging protocol
- [OMA-DM](#) Device Management protocol

[More »](#)

Tools



Package a "one-stop shop" IDE for M2M developers.

We believe that Lua is a language very well-tailored for M2M, therefore the first component we deliver is an IDE for Lua development, called [Lua Development Tools](#).

[More »](#)

Thank you!



m2m

eclipse.org