



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

Benjamin Cabé



# Who I am



- Benjamin Cabé
- Open Source M2M Evangelist at Sierra Wireless
- Eclipse M2M IWG chairperson



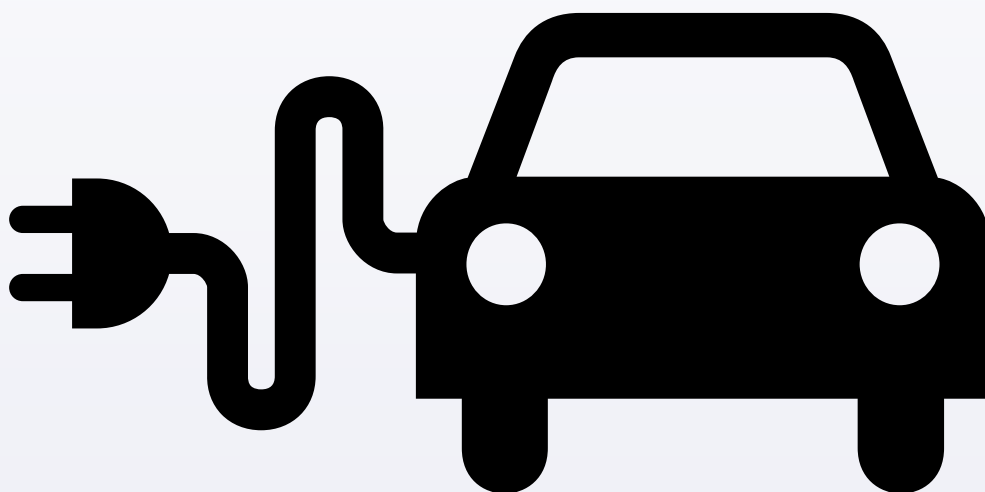
SIERRA  
WIRELESS™

# M2M? IoT?

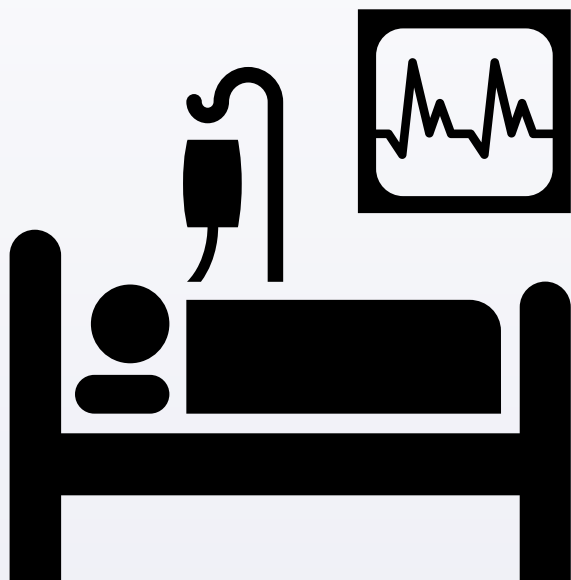


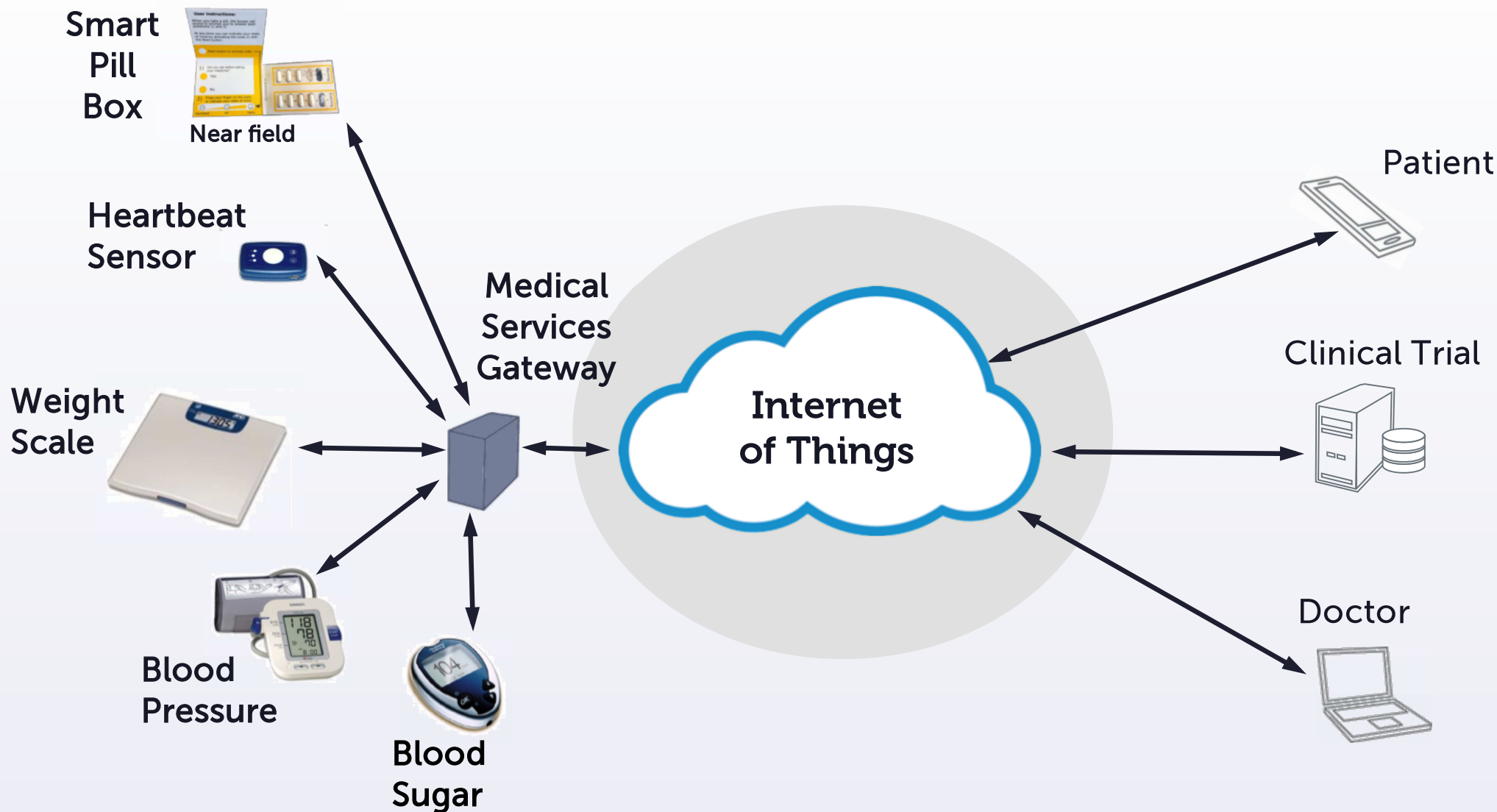
Technology that supports  
wired or wireless  
communication  
between devices



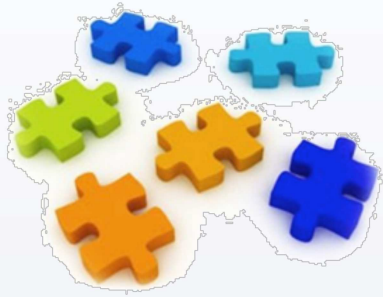








# However...



## 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

# Eclipse M2M IWG



## **Interoperability**

Promote open interoperability between the M2M gateways and M2M server, and between M2M servers and IT/enterprise servers.



## **Tools**

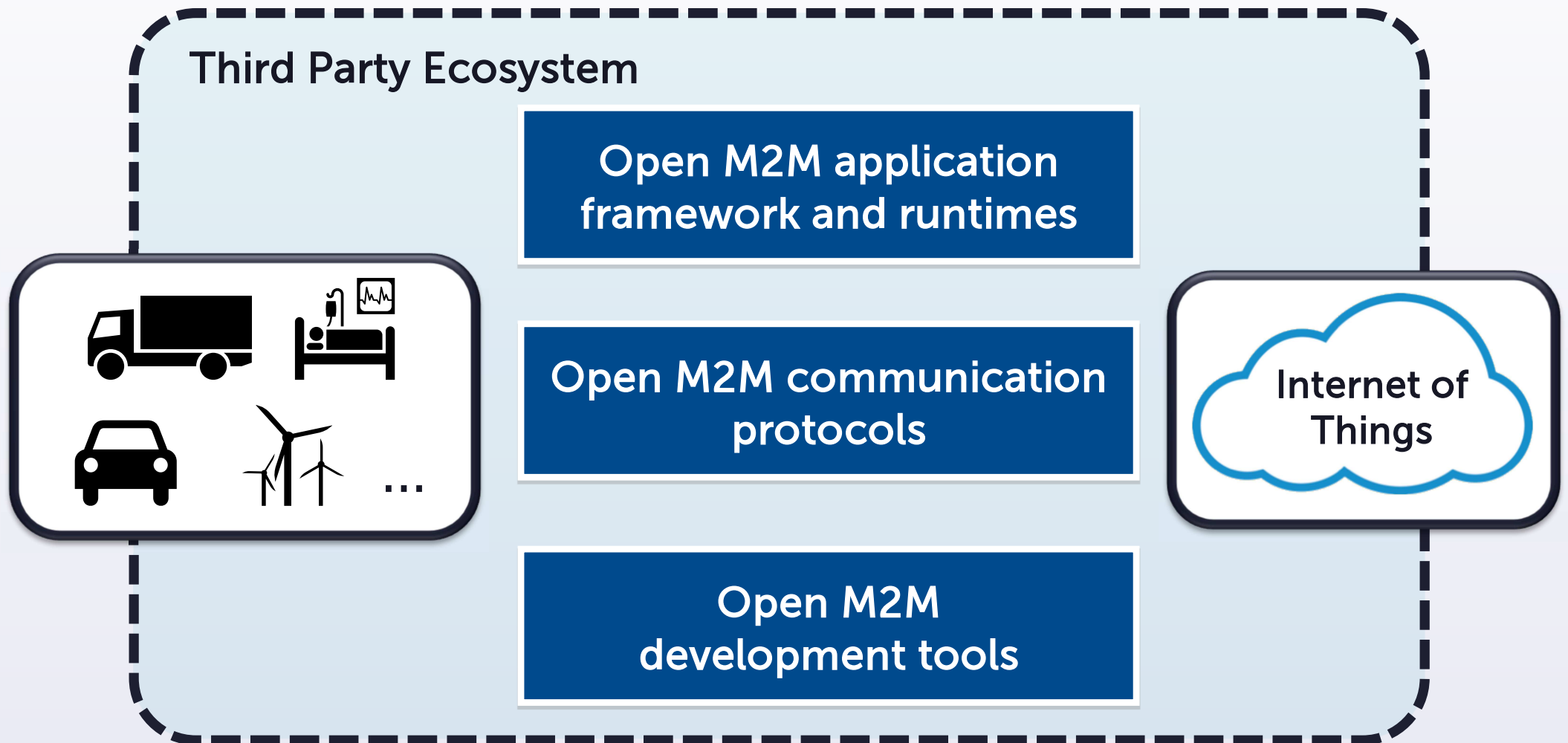
Provide tooling for M2M gateways development including integration with M2M servers



## **User/Developer Experience**

Provide samples, examples, testing environments and technical documentation via a developer hub.

# Open ecosystem for M2M





m2m

eclipse.org

# 3 projects

**Framework**

mihini

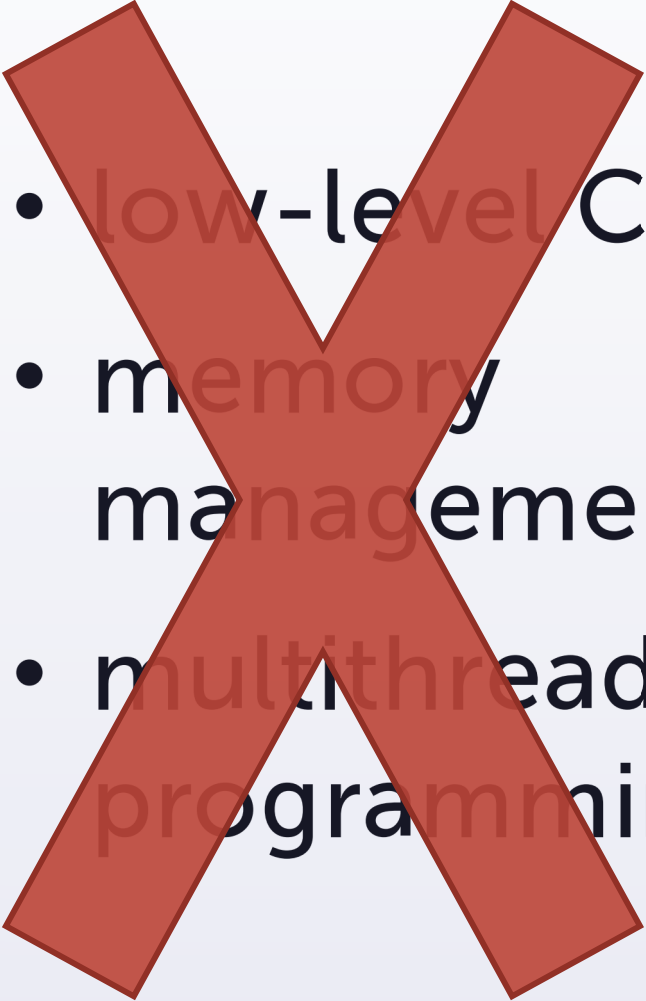
**Protocols**

paho

**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+CSQ\r\n";
    // unsigned char sms_buf[20] = "AT+CMGS="xxxxxxxxx";

    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
```

```
struct termios options;
```

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

```
if (ttys5_fd < 0)
```

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

# **Simplify M2M programming**



# **Lua?**

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

# **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

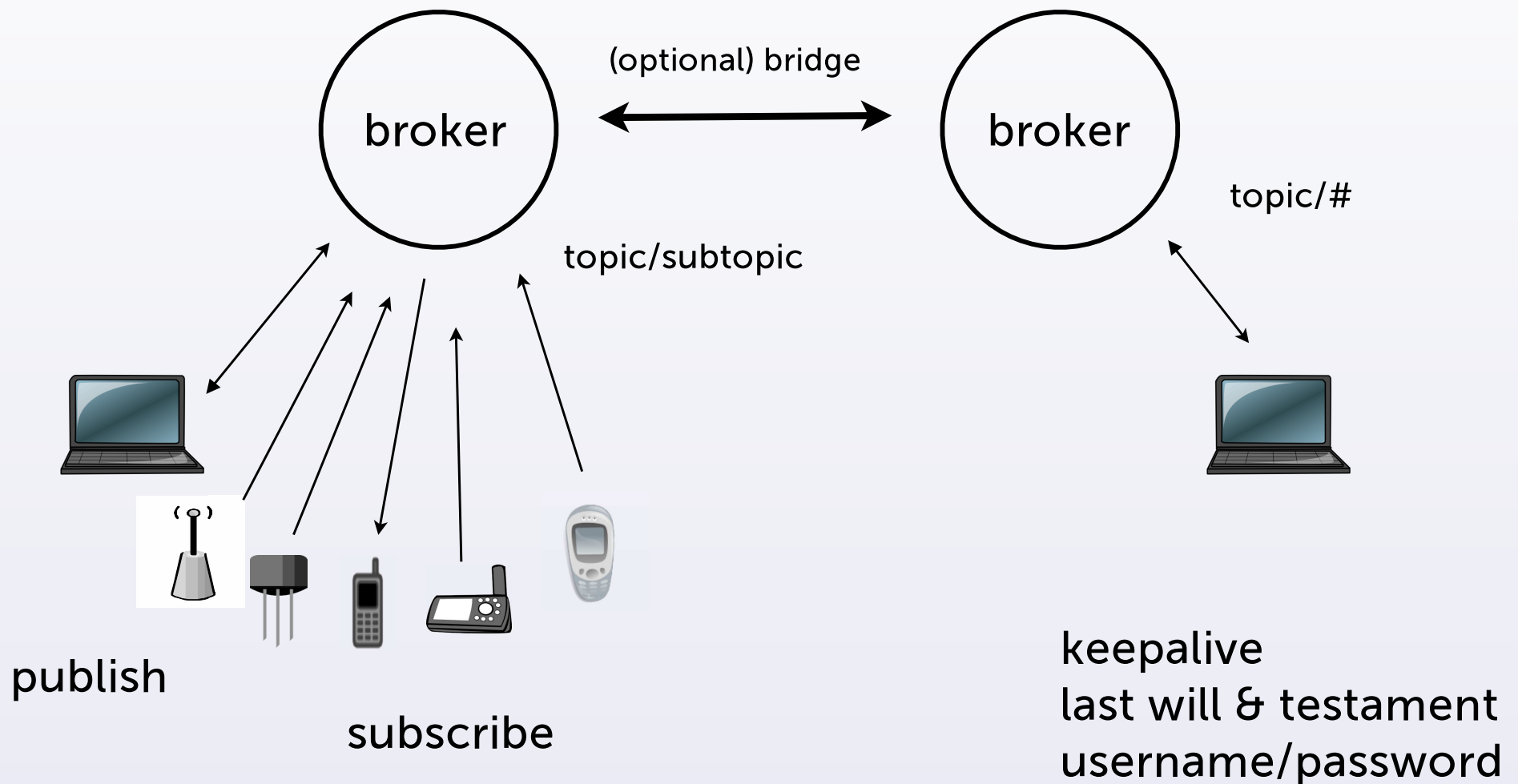
**60,000+** installations already! (Apr. 2013)

# How do we communicate?





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



# M3DA

**Compact** ... *because in the wireless world, data overhead costs money*

- Efficient in the transport of binary M2M data

**Interoperable** ... *because the M2M communication chain is heterogeneous*

- Language-independent
- Tolerant to data schema changes
- Agnostic to transport layer (TCP, HTTP, SMS, ...)

**Secure** ... *because security is #1 concern for M2M adopters*

- Ensure integrity and confidentiality of customer data
- Message ticket-id to enable acknowledgement

**Open** ... *because vendor lock-in hinders M2M adoption*

# 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>

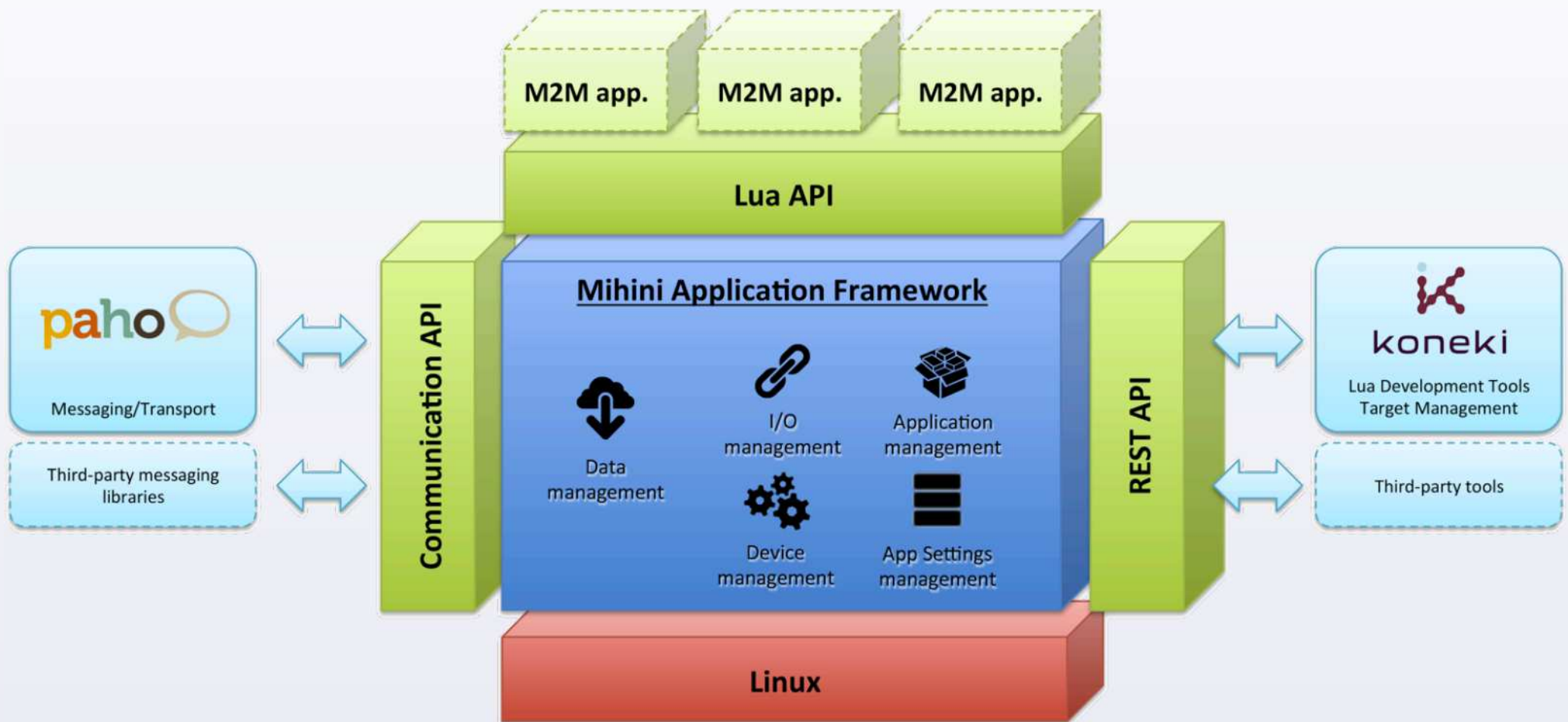
# Smart agent for M2M

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

**mihini**

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

# Overall architecture



# **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.

# 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.

# **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

# Roadmap

- REST API
  - Ease the communication of 3rd party apps with the Agent
  - Provide better tooling
- Polyglot framework
  - C and Java on their way

# http://m2m.eclipse.org

## 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**