

Master Universitario Ingeniería de telecomunicación  
**TELEMÁTICA**

# Implementación de un sistema de gestión “In Cloud” de redes WSN

Autor: Francisco Jesús Villaseñor García

Consultor: José López Vicario



# ÍNDICE

- (1) **Problema a resolver**
- (2) **Estándares IoT**
- (3) **Arquitectura**
- (4) **Mensajes CoAP**
- (5) **Extensión de OpenWSN**
- (6) **Funcionamiento de la solución**
- (7) **Conclusiones y líneas futuras**

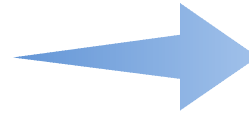
# (1) Problema a resolver

- **Redes de sensores**

Cada vez más habituales

Ubicuas

Múltiples nodos



- **Problemas**

Distribución geográfica

Dificultad de acceso



- **Retos**

Reducción de consumo (autonomía)

Movilidad

- **Solución: gestión remota**

Acceso a la información

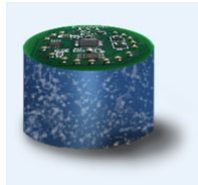
Detección de problemas

Programar mantenimientos

Cambios de funcionamiento centralizados

# (1) Problema a resolver: Otras soluciones analizadas

## Las más conocidas




TinyDB (Berkeley)

- OpenSource
- No tiene una interfaz sino un mecanismo para generarla



MoteWorks (Comercial)

## Inconvenientes

Ligadas a un sistema operativo 

No usan estándares

## Intento de estandarización



Sensor Web Enablement (SWE)

A desarrollar

Servicios SOAP: pesados no es la tendencia actual



# (1) Problema a resolver: Solución desarrollada

## Interfaz web

- ▶ Últimas tendencias (Bootstrap, AngularJS, Single Page Application)
- ▶ Estándares HTML, HTTP-REST, AJAX, JSON



## Servidor web

- ▶ Basado en Flask (PYTHON)
- ▶ Desplegable en la nube
- ▶ Preparado para desplegar sobre Raspberry Pi



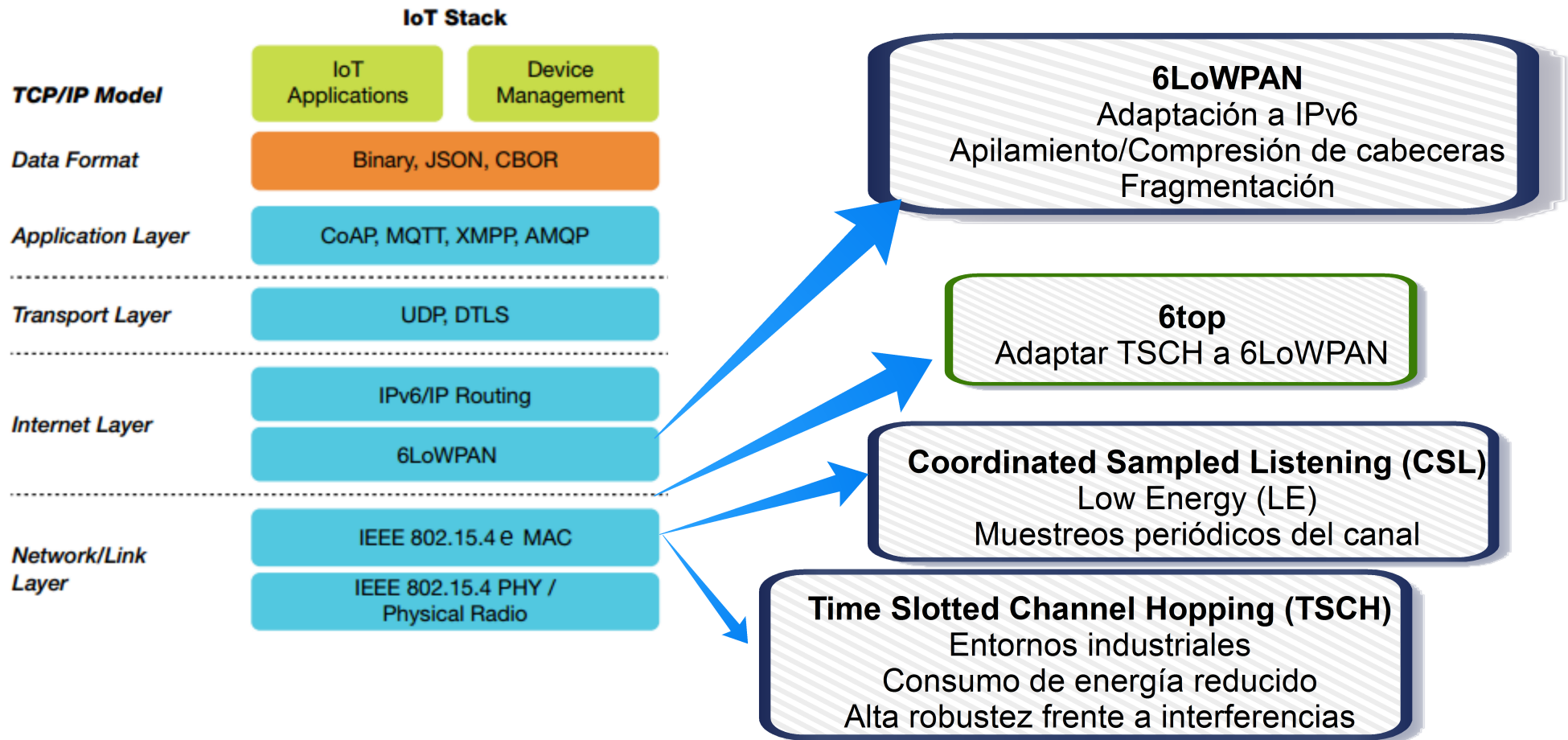
## Mota

- ▶ Protocolos standard IoT
- ▶ Interacción CoAP con la capa del servidor
- ▶ Desplegada en OpenMote-CC2538 y OpenWSN



openmote  
openhardwarefortheinternetofthings

## (2) Estándares IoT



## (2) Estándares IoT: Constrained Application Protocol

CoAP RFC7252 Junio 2014

**Similar a HTTP sobre UDP: ahorro de consumo**

**Transacción CoAP (Token incluido en la cabecera)**

- ◆ CON (confirmado) con piggybacking o sin piggybacking (respuesta en otro mensaje CON)
- ◆ NON (no confirmado)

**Semántica REST: subconjunto de las operaciones HTTP**

- ◆ GET (consulta), seguro e idempotente
- ◆ PUT (modificación/creación) y DELETE (borrado), no seguros e idempotentes
- ◆ POST (creación/modificación/borrado) no seguro y no idempotente

**Códigos de respuesta: subconjunto de los códigos HTTP**

2.xx correcto, 4.xx error cliente, 5.xx error servidor

**URIs CoAP: similares a las HTTP: `coap[s]://host[:puerto]/path[?query]`.**

Puertos por defecto 5683 y 5684

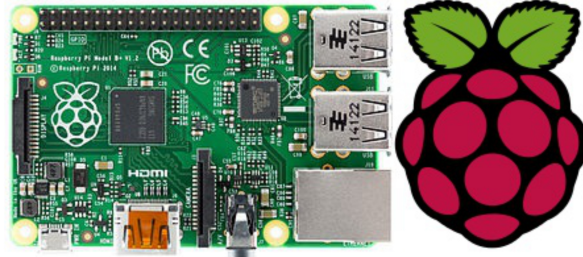
**Compatible con HTTP: proxy CoAP-HTTP directo, proxy HTTP-CoAP sencillo**

# (3) Arquitectura

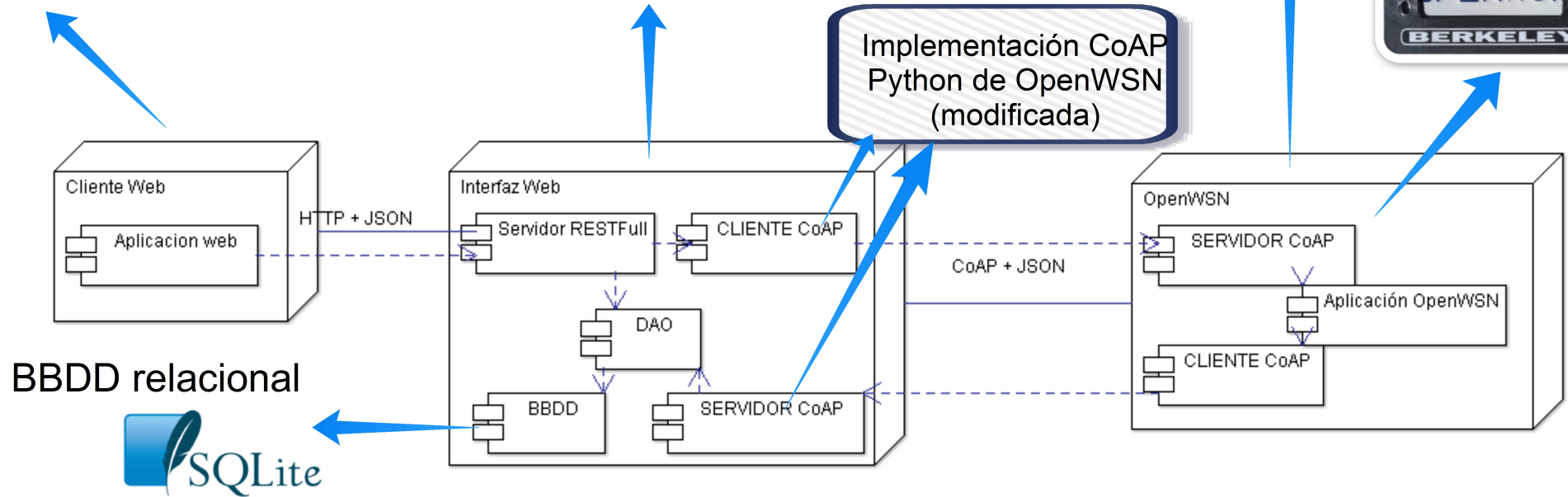
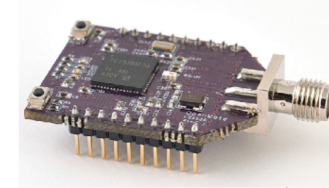
Navegador Web  
(p.e. Chrome)



Servidor  
Raspberry PI



OpenMote-CC2538



BBDD relacional



# (3) Arquitectura: Parámetros en la mota

## Afectan al comportamiento de la mota

1 Servidores [lista de ipv6:puerto]: servidores a los que se desea cargar la información

Servidores  Obligatorio Lista IPv6 separadas por , (255)

```
// metadata
msg->l4_destination_port = uocapp_vars.servidores[iCnt].puerto;
msg->l3_destinationAdd.type = ADDR_128B;
memcpy(&msg->l3_destinationAdd.addr_128b[0], uocapp_vars.servidores[iCnt].ipv6Addr, 16);

// send
outcome = opencoap_send(
    msg,
    COAP_TYPE_NON,
    COAP_CODE_REQ_POST,
    2,
    &uocapp_vars.coap // descripcion CoAP del llamante
);
```

2 Tiempo entre muestras: tiempo entre envíos de información desde la mota

Tiempo entre muestras (segundos)  Obligatorio Numérico (4)

```
uocapp_vars.tarea.timerId = opentimers_start(
    uocapp_vars.tarea.periodo,
    TIMER_PERIODIC,
    TIME_MS,
    &uocapp_tarea_planificada);
```

3 Lista de sensores activos

Lista de sensores activos  Obligatorio Lista <ID> separadas por , (Todos o (N)ninguno (50))

```
for (iCnt = idx; iCnt < MAX_SENSORES; iCnt++) {
    if (uocapp_vars.sensores[iCnt].activo) {
        valor = uocapp_vars.sensores[iCnt].sen->callbackRead();
    }
}
```

# (3) Arquitectura: Parámetros y Descubrimiento

## Parámetros que afectan a la interfaz web

Info. libre	<input type="text" value="-"/>	Obligatorio Texto (255)	◆ Información libre: anotaciones, etc.
Coordenada X	<input type="text" value="0"/>	Obligatorio Numérico (4)	◆ Descripción de la mota: listados
Coordenada Y	<input type="text" value="0"/>	Obligatorio Numérico (4)	◆ X,Y: localización de la mota en el mapa.
Descripción	<input type="text" value="[::1]:55555"/>	Obligatorio Texto (255)	

## Parámetros de estado

Sensores que informan del estado de la mota

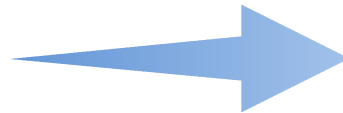
Se define un valor crítico y una comparación para generar una alerta al usuario

- ◆ Estado batería: carga de la batería de la mota en ese momento.
- ◆ Temperatura CPU: temperatura del microcontrolador.

## Descubrimiento

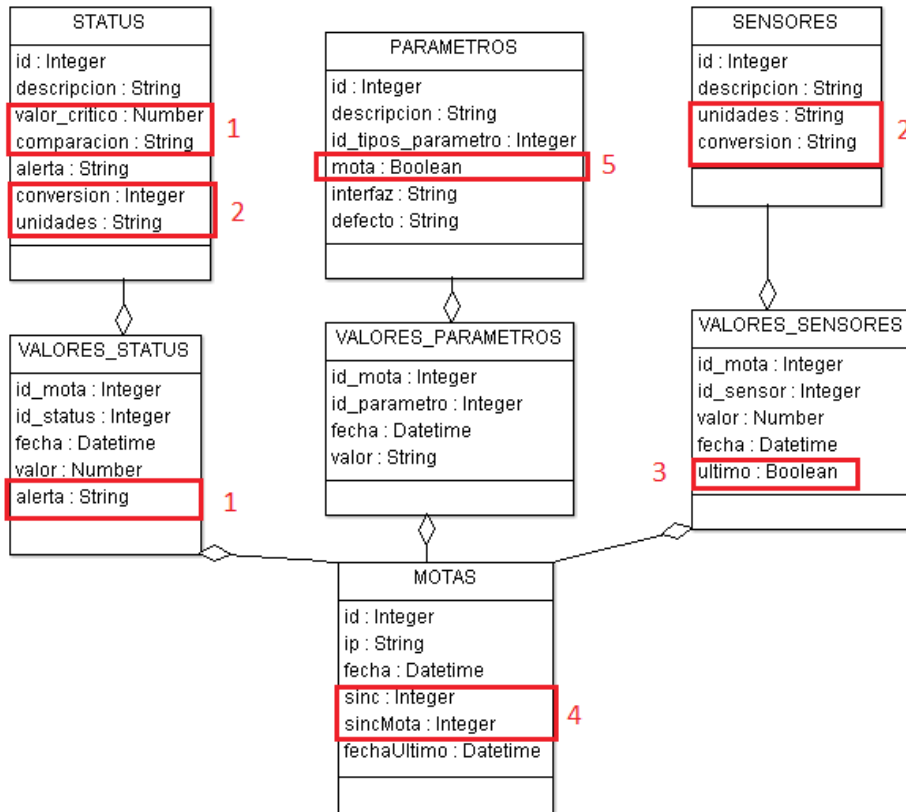
Cómo la Interfaz Web conoce las motas que forman parte del sistema.

Responsabilidad de la mota:  
tiene definidos por defecto un  
valor para los parámetros



La mota envía información a la  
Interfaz Web desde el momento  
que se integra en la red WSN

### (3) Arquitectura: Modelo de datos (resumen)



1/ Si el valor del sensor de estado cumple con la comparación establecida se genera una alerta

2/ Las unidades del valor del sensor y la función de conversión de la lectura del sensor (ADC, entero de 16 bits) al valor de trabajo (°C, %HR, etc.).

3/ Indicador de que es el último valor del sensor recibido

4/ Indicadores de sincronización

- ◆ sinc: indicador enviado a la mota
- ◆ sincMota: indicador que informa la mota que ha recibido

si son distintos es que la mota no ha recibido la actualización de los parámetros.

5/ Indicador de que el parámetro afecta al funcionamiento de la mota



# (4) Mensajes CoAP

## Interfaz Web a la Mota: POST /uocapp



```
{"s":{"1":..., "2":...}, "e":{"1":...}, "f":'...'}
```

s: información de sensores  
e: información de estado  
f: indicador de sincronización

### NO CONFIRMADO (NON)

- Reducir la sobrecarga de la red
- Información es periódica

## Mota a la Interfaz Web: POST /uocapp



```
{"0":..., "1":..., "2":...}
```

"1" ids parámetros configuración  
"0" corresponde a sincronización

### CONFIRMADO (CON)

- Información crítica
- Esporádico (no sobrecarga la red WSN)

## Limitaciones OpenWSN

OpenWSN soporta paquetes de 127 bytes sin fragmentación  
División lógica de los mensajes anteriores para respetar con estos límites.



# (5) Extensión de OpenWSN

## Implementación CoAP Python

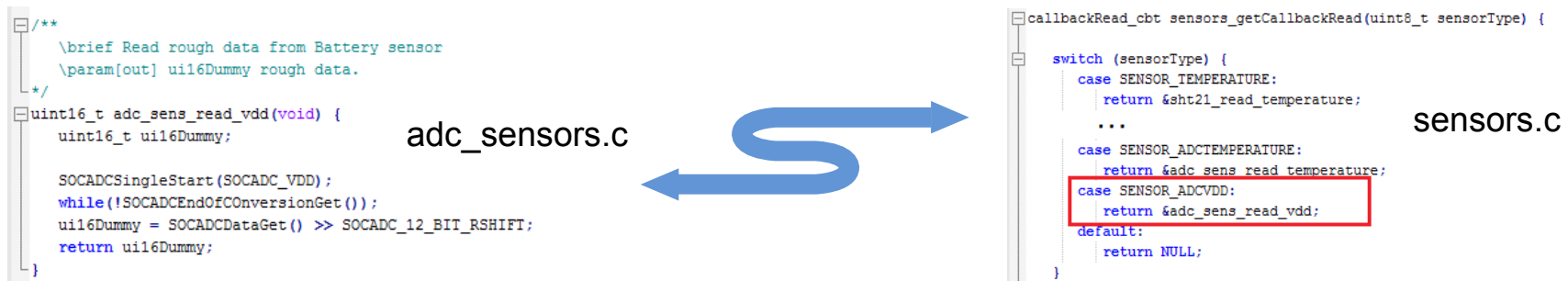
El recurso encargado de atender la petición recibe la información del emisor.  
Así se puede identificar la mota en la red WSN por su IP:puerto



## Lectura del estado de la batería

Parámetro crítico en una red dispersa

Se utilizan las facilidades que proporciona el SoC CC2538



# (6) Funcionamiento de la solución

## Conexión red WSN

The screenshot shows the OpenWSN web interface at localhost:8080. It features a sidebar with navigation options: Motes, Event Bus, Topology, Routing, and Documentation. The main content area is titled 'Motes' and includes a dropdown menu set to 'ed75' and a 'Toggle DAGroot state' button. Below this, there are sections for 'Mote' (Prefix: bb-bb-00-00-00-00-00, EUI-64: 00-12-4b-00-04-33-ed-75) and 'Root Status' (DAG Root? Yes). A 'Data' section has tabs for Network, Schedule, and Neighbors. The Neighbors tab is active, displaying a table with columns: Used, Parent, Stable, Stability, Address, DAG Rank, JP, RSS, RX, TX, TX ACK, Wrap, and ASN.

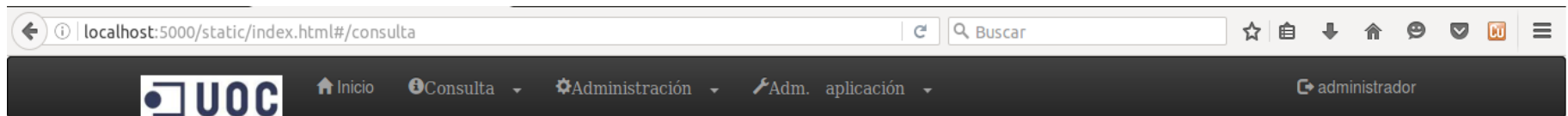
Used	Parent	Stable	Stability	Address	DAG Rank	JP	RSS	RX	TX	TX ACK	Wrap	ASN
1	0	1	0	00-12-4b-00-06-0d-9e-e1 (64b)	65535	0	-33 dBm	7	0	0	0	0x00000060cf
0	0	0	0	(None)	0	0	0 dBm	0	0	0	0	0x0000000000

OpenVisualizer



# (6) Funcionamiento de la solución

## Consultas



Consulta de los últimos valores obtenidos de las motas

Listado público

IPv6 ↑	Descripción ↑	Fecha alta ↑
[bbbb::12:4b00:60d:9ee1]:5683	[bbbb::12:4b00:60d:9ee1]:5683	2016-06-08 06:17:04



Al pulsar se ven los valores de los sensores

[bbbb::12:4b00:60d:9ee1]:5683 ([bbbb::12:4b00:60d:9ee1]:5683) ×

Sensor ↑	3	Valor
Aceleración X (ADXL346)		24 -
Aceleración Y (ADXL346)		4294967276 -
Aceleración Z (ADXL346)		293 -
Humedad (SHT21)		37.266 %HR
Luz (MAX44009)		2.34 Lux
Temperatura (SHT21)		26.606 °C

Cerrar

Log de la aplicación

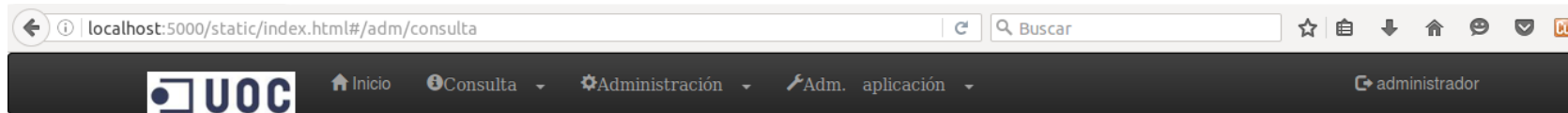
```
openwsn@openwsn-VirtualBox: ~/InterfazWeb/log 2
[DEBUG] 2016-06-08 06:17:04,575 uocapp: Recibido [{"s":{"1":27396,"2":22684,"3":820}}] de ('bbbb::12:4b00:60d:9ee1', 5683)
[DEBUG] 2016-06-08 06:17:04,740 uocapp: Recibido [{"s":{"4":24,"5":4294967276,"6":293}}] de ('bbbb::12:4b00:60d:9ee1', 5683)
[DEBUG] 2016-06-08 06:17:05,399 uocapp: Recibido [{"e":{"1":1439,"2":1591}}] de ('bbbb::12:4b00:60d:9ee1', 5683)
[DEBUG] 2016-06-08 06:17:08,367 uocapp: Recibido [{"f":0}] de ('bbbb::12:4b00:60d:9ee1', 5683)
[DEBUG] 2016-06-08 06:17:35,534 uocapp: GET http://localhost:5000/login [administrador]
[DEBUG] 2016-06-08 06:17:35,535 uocapp:
[DEBUG] 2016-06-08 06:17:36,037 uocapp: GET http://localhost:5000/motas [administrador]
[DEBUG] 2016-06-08 06:17:36,038 uocapp:
```

Se ven los mensajes CoAP recibidos de la mota



# (6) Funcionamiento de la solución

## Administrador (I)



### Estado de motas en el sistema

Buscar

IPv6	Descripción	Fecha alta	Fecha utl. notificación	Estado
[bbbb::12:4b00:60d:9ee1]:5683	[bbbb::12:4b00:60d:9ee1]:5683	05/06/2016 14:41:02	05/06/2016 14:51:07	 

Al pulsar se ve el detalle

Mota [bbbb::12:4b00:60d:9ee1]:5683 3

Param. estado	Fecha	Valor	Valor critico
Estado batería	08/06/2016 07:15:32	2.738 V	<= 2.2
Temperatura CPU	08/06/2016 07:15:32	29.625 °C	>= 100

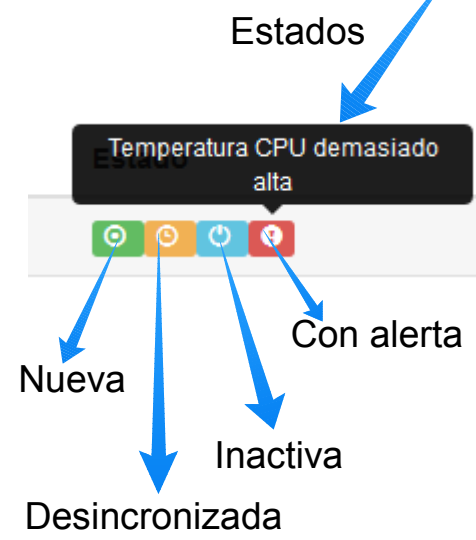
Cerrar

### Listado del administrador

Eliminar la mota de la lista

¿Desea eliminar la mota?

Confirmar Cancelar



### Estado de motas en el sistema

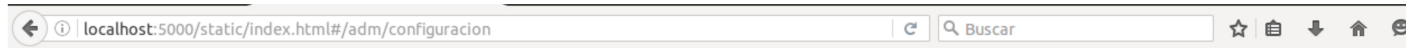
Buscar

IPv6 Descripción Fecha

Mota eliminada correctamente

# (5) Funcionamiento de la solución

## Administración (II)



### Configuración de motas en el sistema

IPv6	Descripción	Fecha alta	Fecha utl. notificación
[bbbb::12:4b00:60d:9ee1]:5683	[bbbb::12:4b00:60d:9ee1]:5683	06/06/2016 19:47:34	06/06/2016 19:47:35

Ejemplo modificación tiempo entre muestras a 30 segundos

**Servidores** [BBBB::1]:5683 Obligatorio Lista IPv6 separadas por (255)

**Tiempo entre muestras (segundos)** 300 Obligatorio Numérico (4)

**Lista de sensores activos** T Obligatorio Lista <ID> separadas por (T)odos o (N)inguno (50)

**Info. libre** - Obligatorio Texto (255)

**Coordenada X** 0 Obligatorio Numérico (4)

**Coordenada Y** 0 Obligatorio Numérico (4)

**Descripción** [bbbb::12:4b00:60d:9ee1]:5683 Obligatorio Texto (255)

**Guardar** **Sincronizar** Cancelar

### Histórico de valores obtenidos de una mota

Esta es la mota 9ee1 ([bbbb::12:4b00:60d:9ee1]:5683) Temperatura (SHT21)

Fecha	Valor
2016-06-06 19:47:34	26.038 °C
2016-06-06 19:50:42	26.124 °C
2016-06-06 19:51:12	26.124 °C
2016-06-06 19:51:43	26.124 °C
2016-06-06 19:52:13	26.124 °C
2016-06-06 19:52:43	26.124 °C

Ejemplo modificación descripción

### Consulta de los últimos valores obtenidos de las motas

Buscar

IPv6	Descripción
[bbbb::12:4b00:60d:9ee1]:5683	Esta es la mota 9ee1

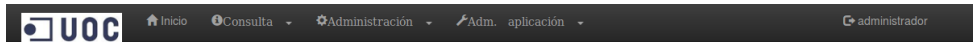
Cambios en la mota seleccionada

Cambios en todas las motas; sólo los 3 primeros parámetros



# (6) Funcionamiento de la solución

## Otras funcionalidades



Histórico de valores obtenidos de una mota

[bbbb::12:4b00:60d:9ee1]:5683 ([bbbb::12:4b00:60d:9ee1]:5683) Temperatura (SHT21) Registros 15

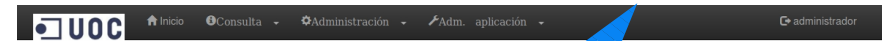
Fecha	Valor
2016-06-05 14:41:02	25.813 °C
2016-06-05 14:46:04	25.459 °C
2016-06-05 14:51:07	25.459 °C



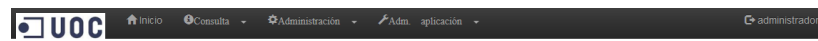
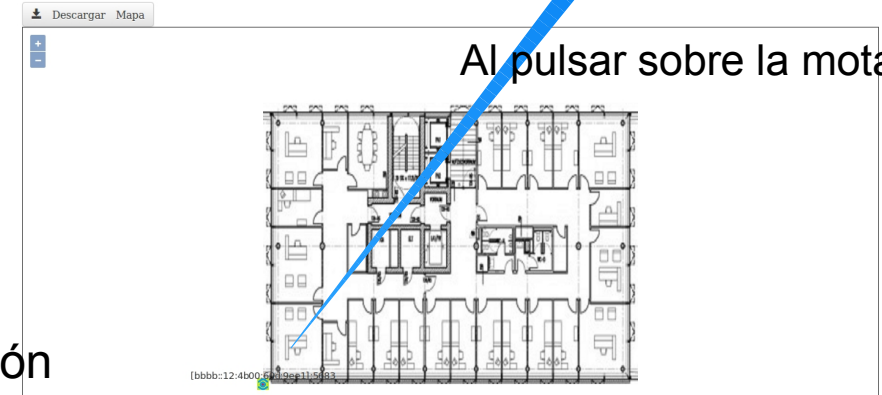
(1) Histórico de valores

(2) Posición sobre un mapa

Sensor ID	Valor
Aceleración X (ADXL346)	873 -
Aceleración Y (ADXL346)	27 -
Aceleración Z (ADXL346)	4294967276 -
Humedad (SHT21)	34.718 %HR
Luz (MAX44009)	4.86 Lux
Temperatura (SHT21)	25.459 °C



Localización de las motas



Administración de tablas

Seleccione la tabla: tipos\_parametro

Buscar [input type="text"] [input type="submit" value="Q"]

ID	Descripcion	Longitud	
1	Número	4	[icon]
2	Texto	255	[icon]
3	Lista IPv6 separadas por .	255	[icon]
4	Lista <ID> separadas por . (Todos o (N)ninguno	50	[icon]

ID [input type="text"] Obligatorio Número 3

Descripcion [input type="text"] Obligatorio Texto 100

Longitud [input type="text"] Obligatorio Número 3

[input type="button" value="Guardar"] [input type="button" value="Cancelar"]

(3) Administración de la aplicación



# (7) Conclusiones y líneas futuras

## Origen:

- ◆ Las redes inalámbricas de sensores WSN son cada vez más habituales.
- ◆ Deseable una aplicación de gestión remota de la red.
- ◆ IEEE 802.15.4 estándar inalámbrico de baja tasa y bajo consumo para aplicaciones IoT.
- ◆ Soluciones tradicionales limitadas a un S.O. sin utilizar estándares.

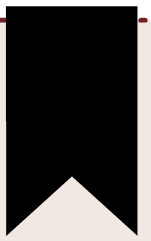
## Objetivo:

- ◆ Desarrollar solución basada en estándares IoT como CoAP.
- ◆ Uso de configuración a nivel de servidor: cualquier mota sobre cualquier S.O. puede integrarse en la herramienta de gestión
- ◆ Seguir tendencias actuales de aplicaciones web (SPA, comunicaciones REST)

## Líneas futuras:

- ◆ Modificación dinámica del software (programación over-the-air OTA)
- ◆ Extensión a otras motas y otros S.O. (Contiki, etc.).

# Implementación de un sistema de gestión “In Cloud” de redes WSN



## Agradecimientos

A José López Vicario, consultor del proyecto, por su colaboración y ayuda a lo largo de todo el desarrollo del mismo.

A Pere Tuset-Peiró, por su ayuda en lo referente a OpenMote

*Francisco Jesús Villaseñor García <fvillasenor@uoc.edu>*

