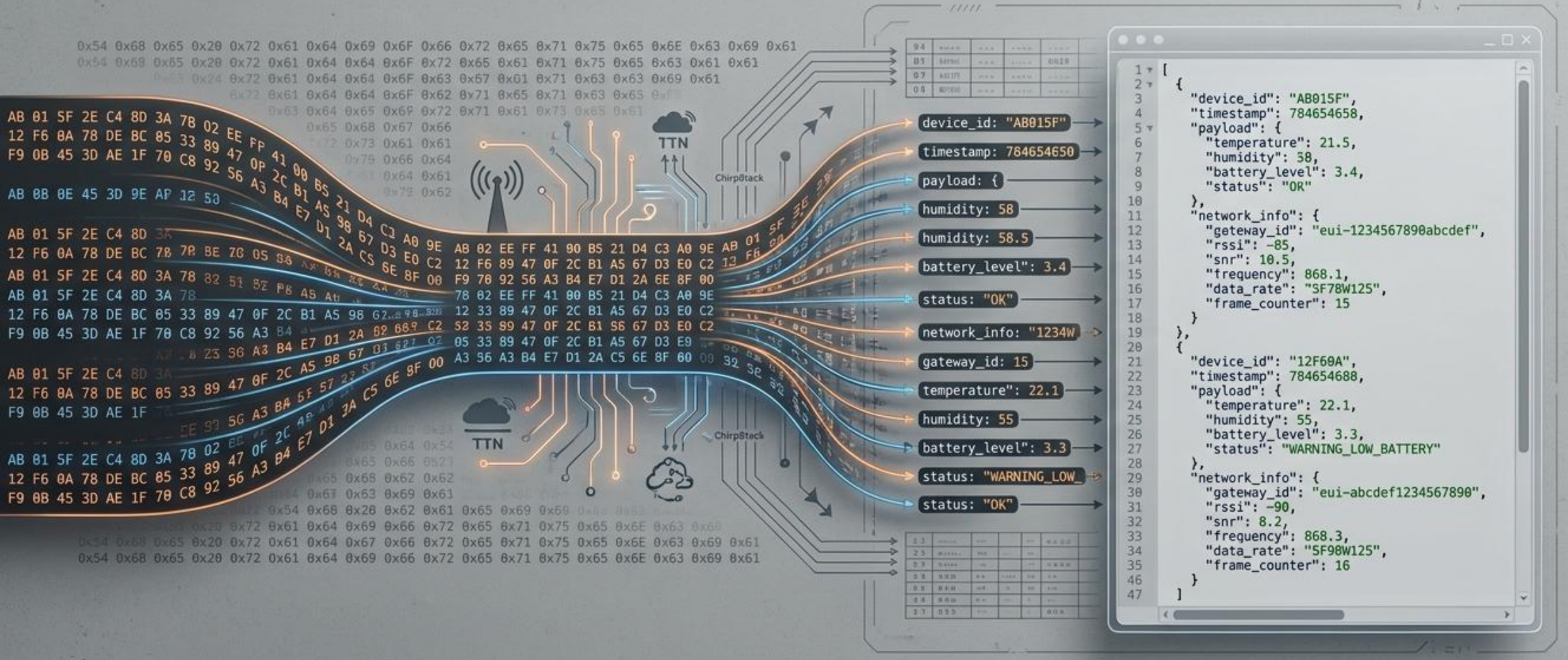
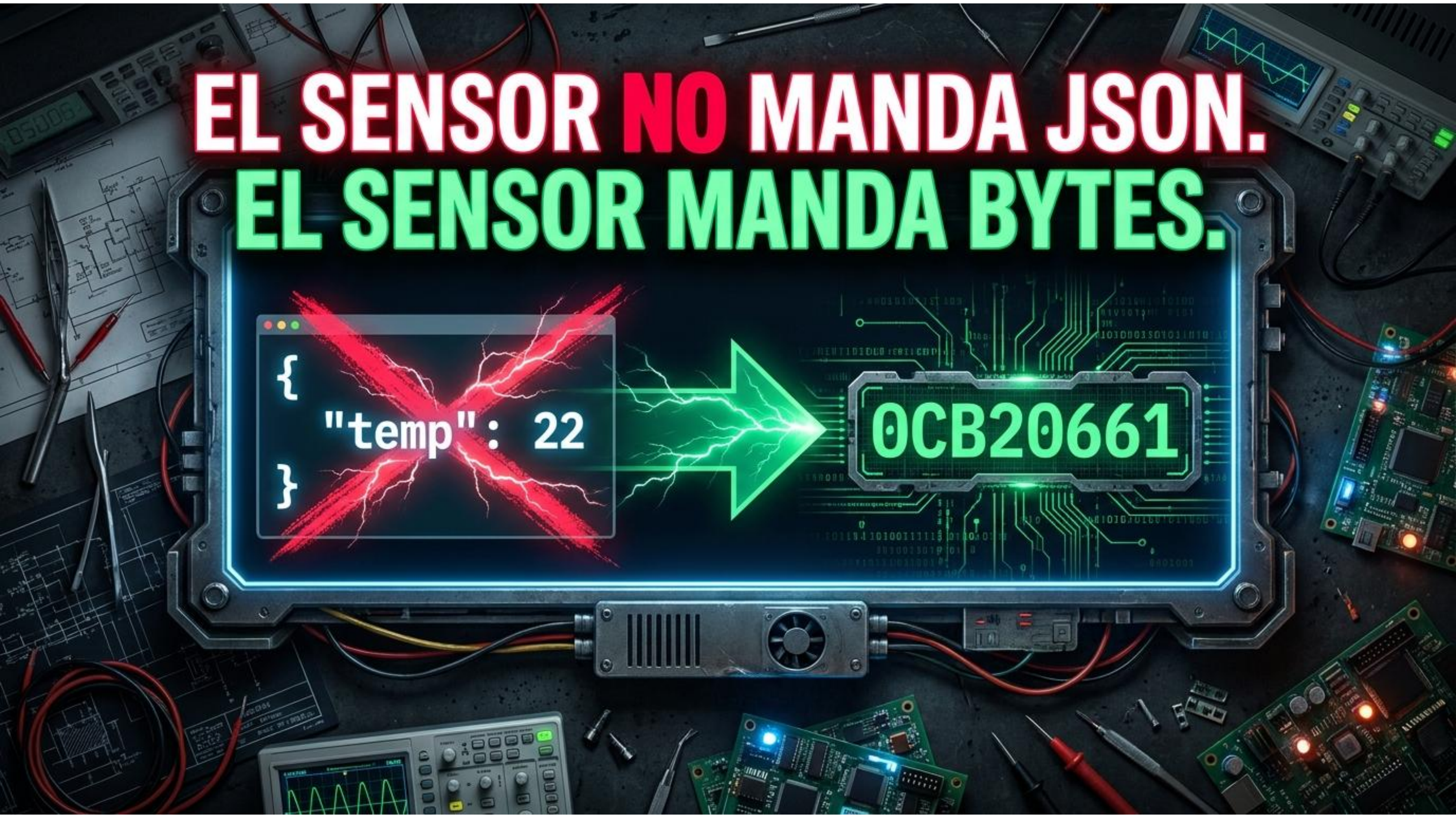
De la Radiofrecuencia al JSON

Práctica de decodificación con TTN y arquitectura de red con ChirpStack




EL SENSOR NO MANDA JSON.
EL SENSOR MANDA BYTES.
{
"temp": 22
}
0CB20661

Entorno Favorable
(Domótica: Wi-Fi / Zigbee)

- Distancias cortas
- Alimentación eléctrica accesible
- Ancho de banda alto
- El reto es conectar dispositivos cercanos.

Entorno Hostil
(Territorio: LoRaWAN)

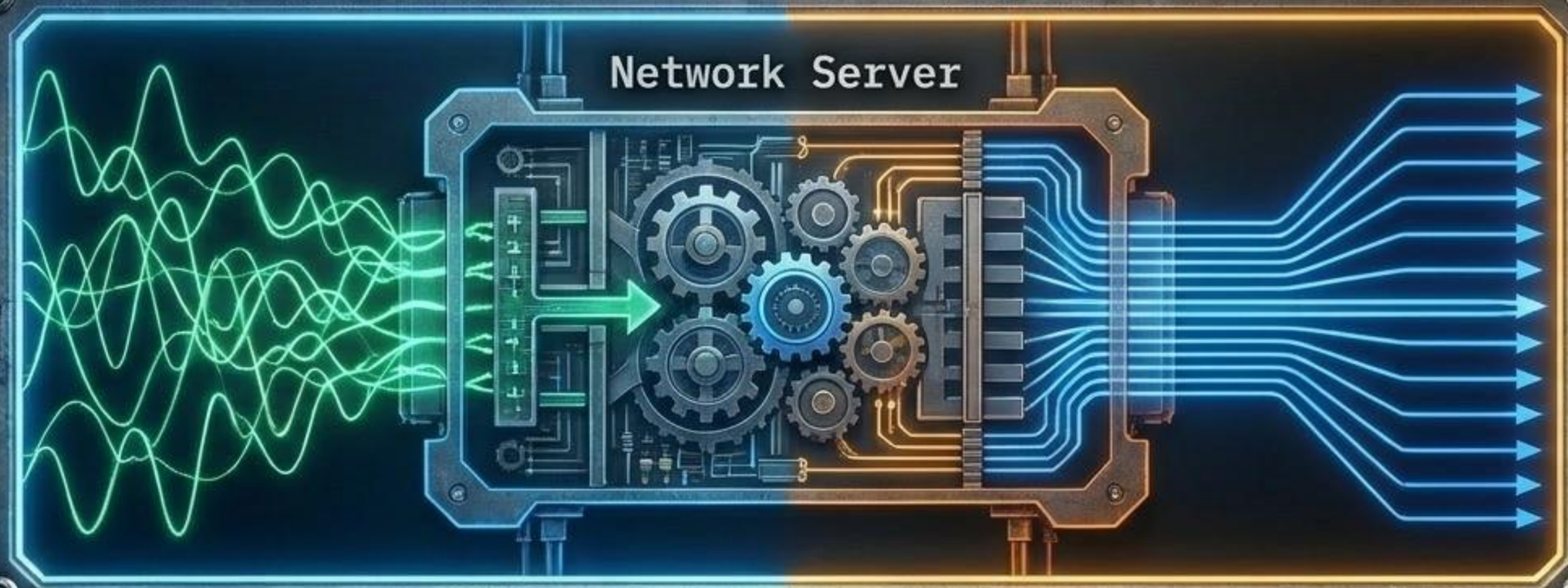
- Distancias kilométricas
- Alimentación por batería
- Intemperie y condiciones extremas
- El reto es mantener sensores vivos durante años lejos de la red eléctrica.

La Arquitectura LoRaWAN

La temperatura no es LoRaWAN. LoRaWAN es solo la forma en la que viaja el dato.

El Rol del Network Server

El motor de la arquitectura LoRaWAN



- Gestión de sesiones (Join requests, OTAA)
- Deduplicación de mensajes (Múltiples gateways escuchan un solo nodo)
- Control ADR (Adaptive Data Rate)
- Decodificación de Payloads (Bytes -> JSON)

La Encrucijada Arquitectónica

Ambos reciben uplinks. La diferencia radica en quién controla la infraestructura y la soberanía del dato.

The Things Stack (TTN Sandbox)

El entorno de aprendizaje y validación.

- Consola web amigable (No requiere instalación de servidor).
- Permite crear aplicaciones y simular uplinks sin hardware.
- Ideal para probar payload formatters y decoders.
- Limitación: Dependencia de infraestructura externa y menor control sobre el backend.

ChirpStack

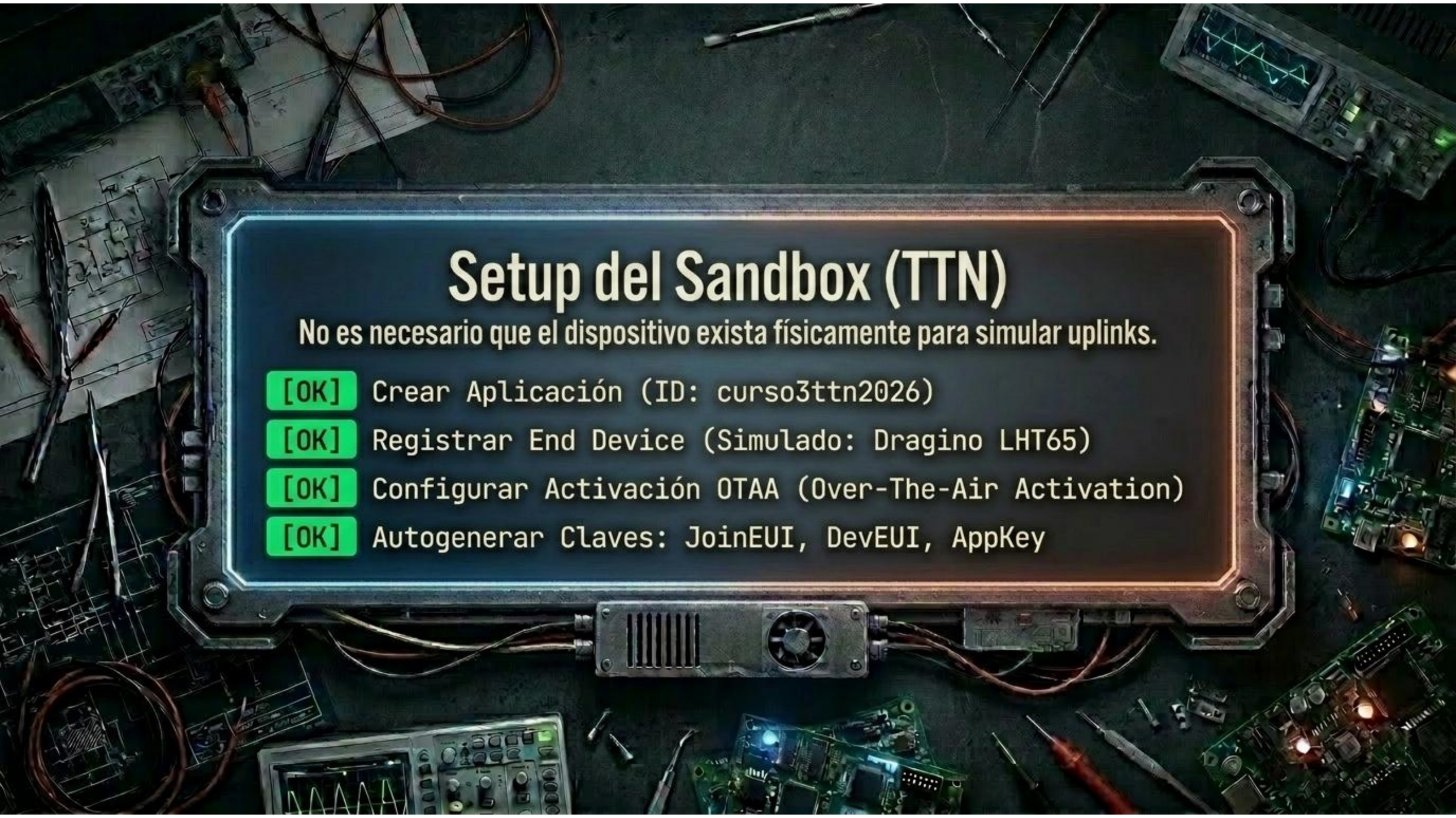
La infraestructura soberana y privada.

- Pila Open Source para redes privadas.
- Control total sobre dispositivos, usuarios e integraciones (MQTT/HTTP).
- Diseñado para despliegues municipales, industriales o criticos.
- Responsabilidad: Requiere mantenimiento propio, gestión de servidores, seguridad y backups.

Matriz de Decisión: TTN vs. ChirpStack

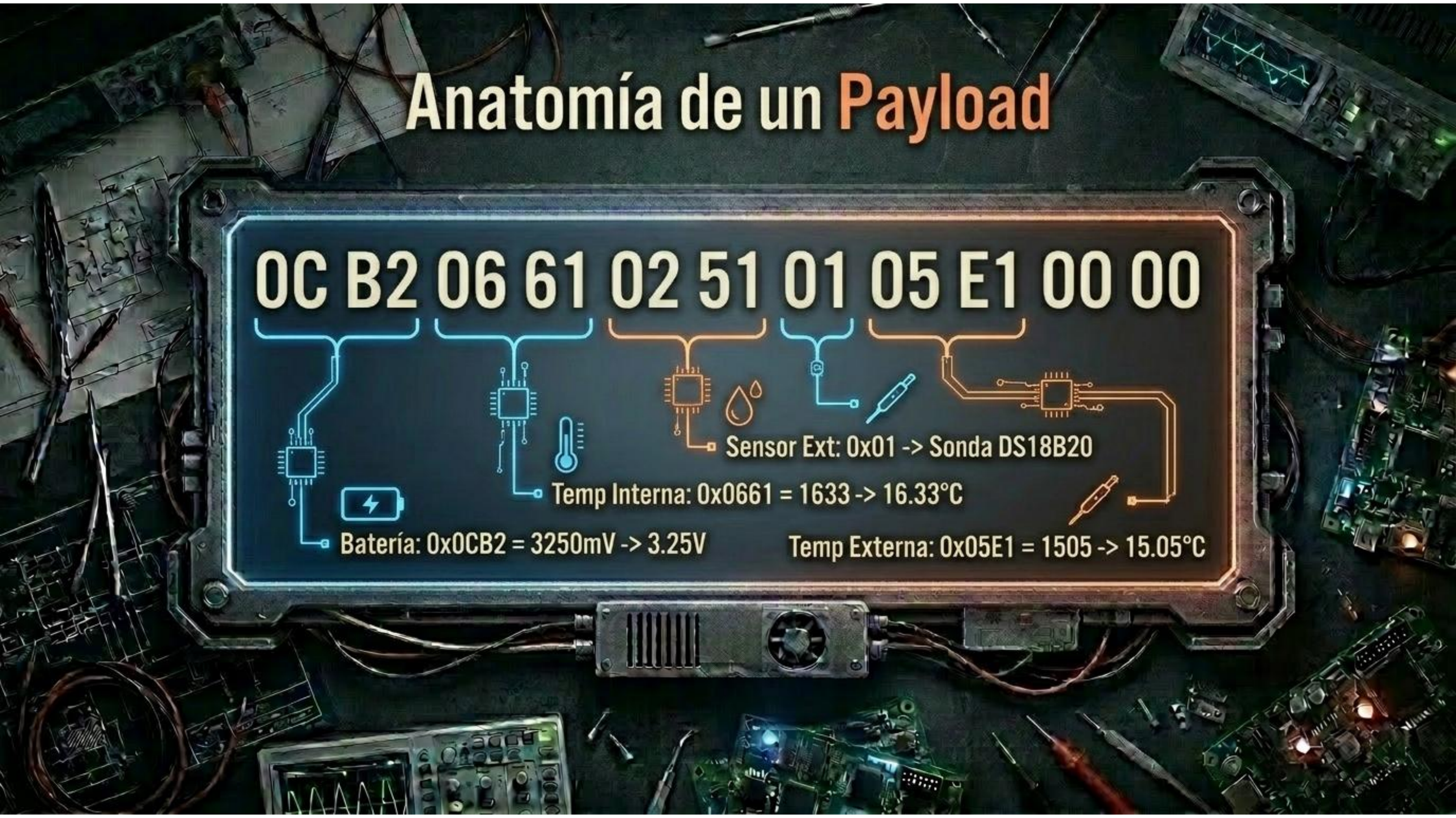
Dimensión	TTN Sandbox	ChirpStack
Soberanía del Dato	Limitada (Nube pública)	Absoluta (On-premise/Privada)
Infraestructura	Externa / Gestionada	Propia / Desplegada por cliente
Complejidad Operativa	Baja	Media / Alta
Mantenimiento	Mínimo	Gestión de servidores y backups
Mejor Caso de Uso	Aulas, Prototipos, Demos	Territorio, Industria, Smart City


“TTN es perfecto para aprender el camino.”
“ChirpStack es para controlar la carretera.”
Para esta práctica, entraremos al Sandbox
de TTN para entender la alquimia de los datos.


Setup del Sandbox (TTN)
No es necesario que el dispositivo exista físicamente para simular uplinks.
[OK] Crear Aplicación (ID: curso3ttn2026)
[OK] Registrar End Device (Simulado: Dragino LHT65)
[OK] Configurar Activación OTAA (Over-The-Air Activation)
[OK] Autogenerar Claves: JoinEUI, DevEUI, AppKey

El Problema del Ancho de Banda

Transmitir menos bytes respeta el duty cycle y garantiza la autonomía de la batería por años.

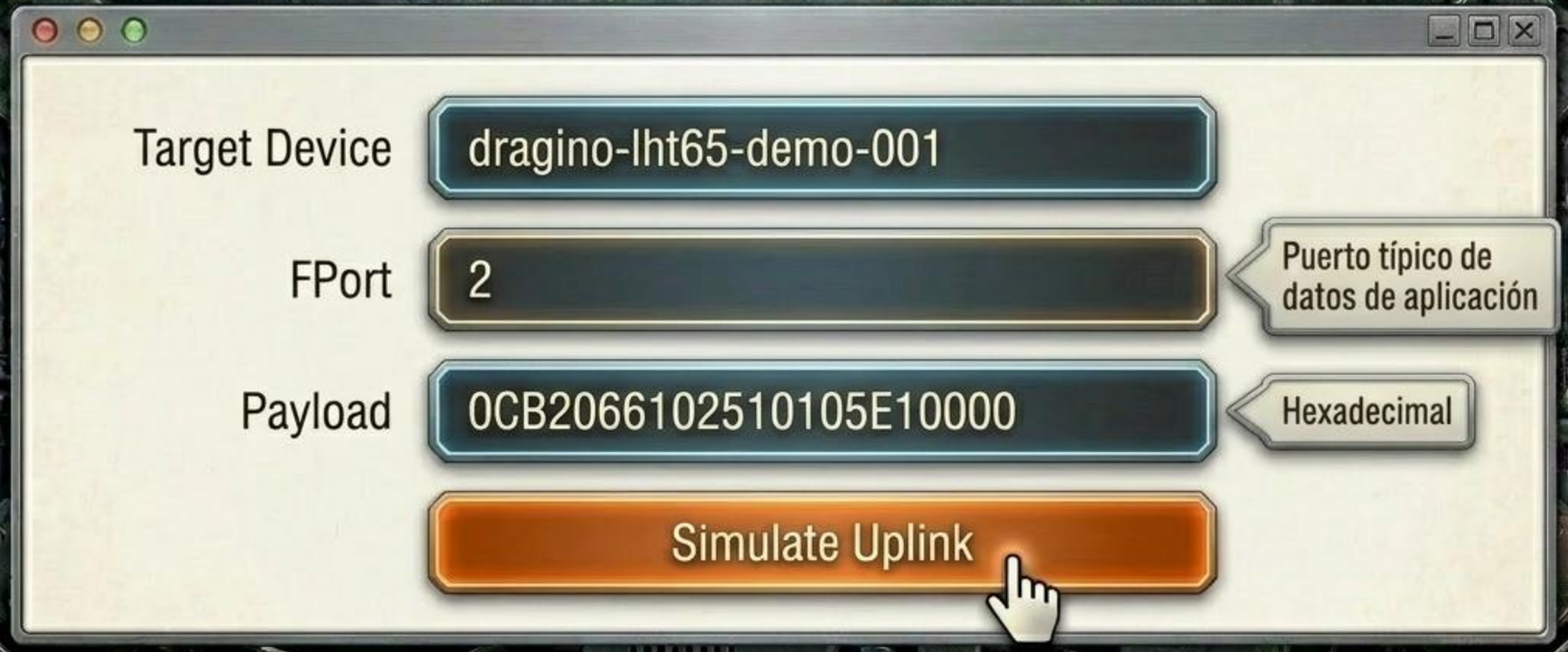

Anatomía de un Payload
0C B2 06 61 02 51 01 05 E1 00 00
Batería: 0x0CB2 = 3250mV -> 3.25V
Temp Interna: 0x0661 = 1633 -> 16.33°C
Sensor Ext: 0x01 -> Sonda DS18B20
Temp Externa: 0x05E1 = 1505 -> 15.05°C

El Decoder: Custom Javascript Formatter

```javascript
function decodeUplink(input) {
  var bytes = input.bytes;

  // Bytes 0-1: Batería
  var rawBattery = (bytes[0] << 8) | bytes[1];
  var batteryMv = rawBattery & 0x3FFF;

  // Bytes 2-3: Temperatura Interna
  var internalTempRaw = readInt16BE(bytes, 2);

  return { data: { battery_v: batteryMv / 1000 } };
}
```

Operaciones bit a bit para reconstruir el dato numérico original.

Simulación de Uplink en TTN



La Transformación

El decoder transforma el ruido radioeléctrico en contexto de negocio.

frm_payload (Dato Crudo)

```
{
  "frm_payload": "DLIGYQJRAQXhAAA="
}
```

Codificado en Base64 para el transporte en la red.

decoded_payload (Dato Interpretado)

```
{
  "battery_v": 3.25,
  "temperature_sht_c": 16.33,
  "humidity_sht_pct": 59.3,
  "external_temperature_c": 15.05
}
```

La Primera Frontera de Calidad

Un pipeline profesional debe detectar valores incoherentes antes de que lleguen al dashboard.

⚠ REPORTE DE ERROR - VALIDACIÓN DE PAYLOAD ⚠

1. Payload Sospechoso inyectado:

`0CB206610B010205E1`

2. Extracción de bytes de humedad (0B 01):

`0x0B01 = 2817`

2. Extracción de bytes de humedad (0B 01):

`0x0B01 = 2817`

3. Aplicación de fórmula:

`2817 / 10 = 281.7%` ⚠

4. Lógica de Validación:

```
if (humidity > 100) {
    warnings.push("Humedad fuera de
        rango físico.");
}
```

¡ERROR CRÍTICO DETECTADO!
Humedad fuera de rango físico. Datos descartados.

El Flujo Completo: Listo para la Explotación

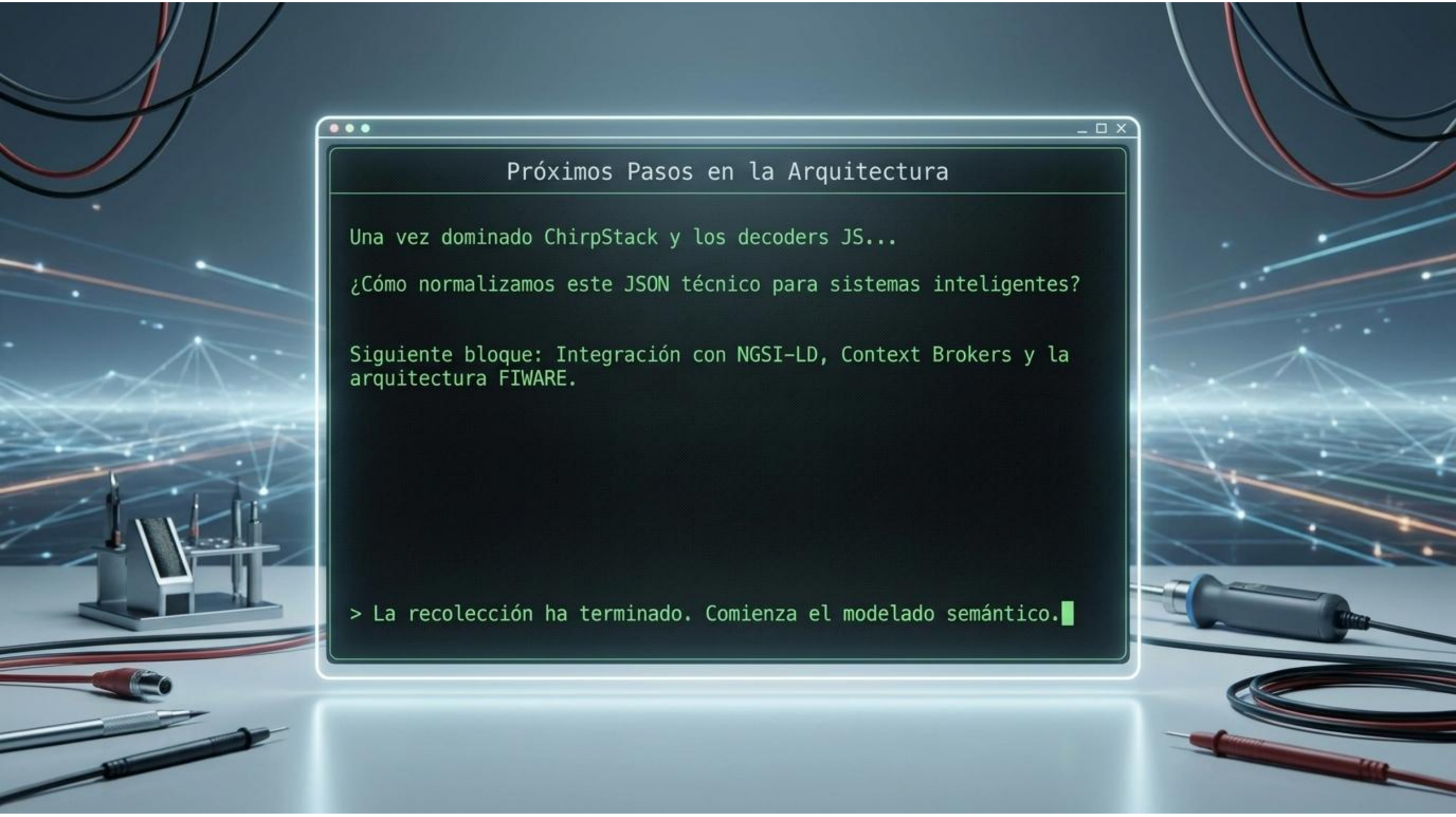
El JSON técnico validado ya está listo para alimentar bases de datos, Context Brokers y dashboards territoriales.

Síntesis Arquitectónica

> El sensor manda bytes. El JSON lo construimos nosotros en el borde.

> Elegir conectividad es elegir el menor coste operativo compatible con la fiabilidad necesaria.

> La soberanía del dato empieza antes del dashboard: empieza en la red.


Próximos Pasos en la Arquitectura
Una vez dominado ChirpStack y los decoders JS...
¿Cómo normalizamos este JSON técnico para sistemas inteligentes?
Siguiente bloque: Integración con NGSI-LD, Context Brokers y la arquitectura FIWARE.
> La recolección ha terminado. Comienza el modelado semántico.