MANUAL TÉCNICO DE INGENIERÍA — MÓDULO 07

Arquitectura de Contenedores Docker en ARM64, Series Temporales en InfluxDB 2.7, Resiliencia Store-and-Forward y Ergonomía Visual ISA-101

Curso: Internet de las Cosas (IoT) con Arduino y Raspberry Pi — Ingeniería Mecatrobótica

---

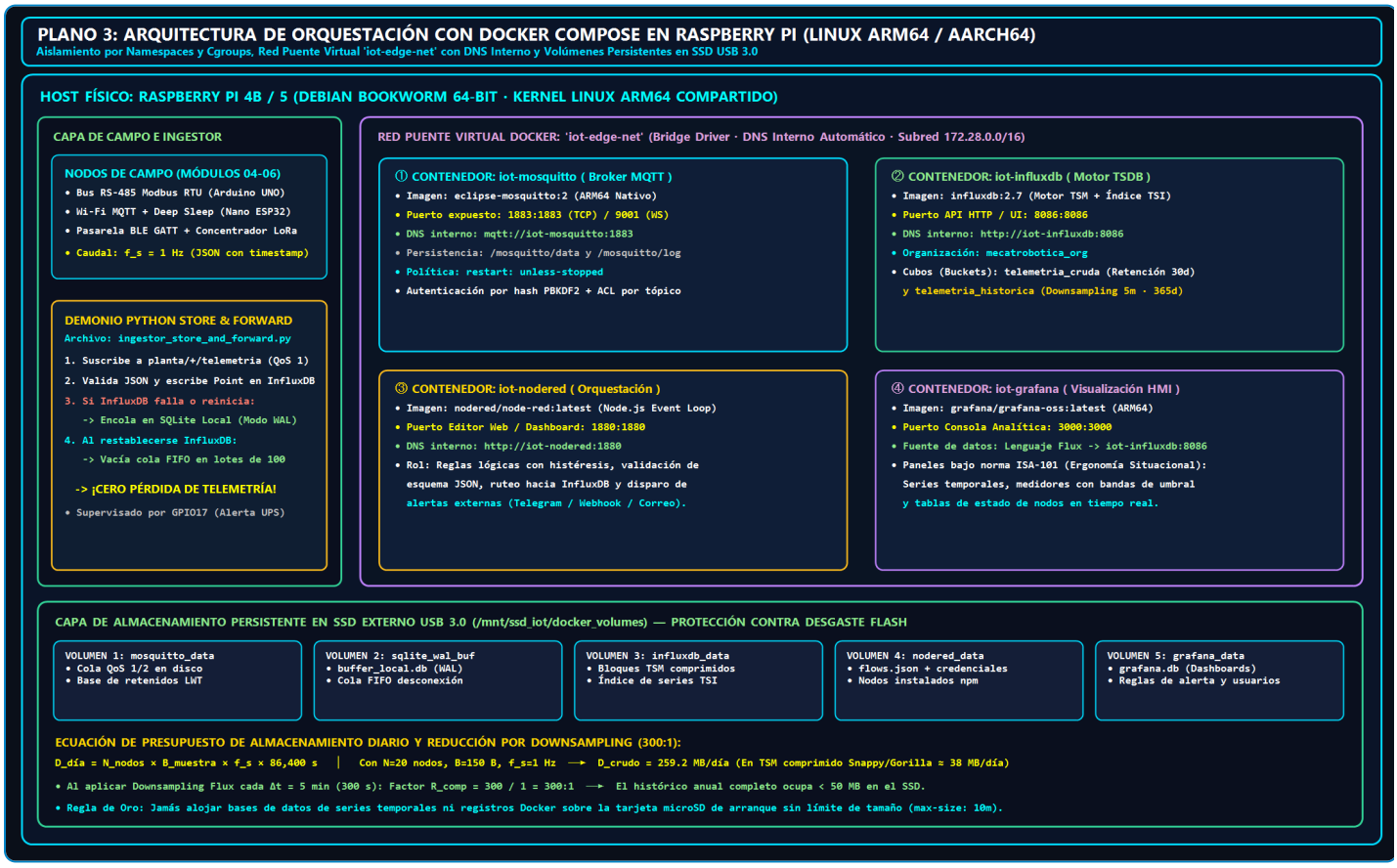
1. Fundamentos de Contenedorización con Docker en Linux ARM64

En despliegues industriales de borde (*Edge Computing*), instalar manualmente demonios de bases de datos, intérpretes de Node.js y servidores web directamente sobre el sistema operativo anfitrión conduce a conflictos de bibliotecas dinámicas y dificulta la recuperación ante desastres.

Mediante **Docker Engine** y **Docker Compose** sobre arquitecturas `ARM64 (aarch64)`, cada servicio opera aislado mediante dos primitivas nativas del kernel de Linux (sin la sobrecarga de emular hardware de una máquina virtual):

- **Espacios de nombres (`Namespaces`)**: Aíslan el árbol de procesos (`PID`), la pila de red (`NET`) y los puntos de montaje (`MNT`) de cada contenedor.
- **Grupos de control (`Cgroups v2`)**: Limitan y auditan el uso de memoria RAM y ciclos de CPU por servicio.
- **Red puente definida por el usuario (`iot-edge-net`)**: Provee resolución DNS interna automática por nombre de servicio (`iot-mosquitto`, `iot-influxdb`, `iot-nodered`, `iot-grafana`), evitando depender de direcciones IP estáticas frágiles.

📐 Plano 3: Arquitectura Docker Compose ARM64



---

2. Bases de Datos Relacionales (SQL) frente a Series Temporales (TSDB)

Una señal de telemetría mecatrónica es una sucesión discreta ordenada cronológicamente:

$$S = \{(t_1, \mathbf{v}_1), (t_2, \mathbf{v}_2), \dots, (t_k, \mathbf{v}_k)\} \quad \text{con } t_1 < t_2 < \dots < t_k$$

2.1. Anatomía del Protocolo de Línea (*Line Protocol*) en InfluxDB

Cada registro en InfluxDB se compone de cuatro elementos:

```
ambiente,nodo=invernadero_01,ubicacion=sector_norte temperatura=24.65,humedad=61.2,pwm_ventilador=140i
1730505600000000000
```

- **Medición (`Measurement`)**: Contenedor lógico de la magnitud (`ambiente`).
- **Conjunto de Etiquetas (`Tag Set`)**: Pares clave-valor de metadatos **indexados en memoria RAM** (`nodo`, `ubicacion`). Se utilizan en cláusulas de filtrado rápido.
- **Conjunto de Campos (`Field Set`)**: Valores numéricos variables **no indexados** (`temperatura`, `humedad`, `pwm_ventilador`).
- **Marca de Tiempo (`Timestamp`)**: Entero de 64 bits en nanosegundos UTC desde la época Unix.

2.2. Control Matemático de la Cardinalidad de Series

La cardinalidad total de series C_{series} en un cubo de InfluxDB viene dada por el producto cartesiano de los valores únicos de cada etiqueta:

$$C_{series} = |M| \times \prod_{i=1}^{m} |Tag_i| \times |Field_{keys}|$$

> **Regla Crítica de Diseño**: Jamás almacenes valores continuos (temperaturas con decimales), marcas de tiempo ni identificadores únicos por mensaje (`UUID`) dentro de un **Tag**. Hacerlo provoca una explosión de cardinalidad (*Runaway Cardinality*) que agota la memoria RAM de la Raspberry Pi (`OOM Kill`). Los valores variables van siempre en **Fields**.

---

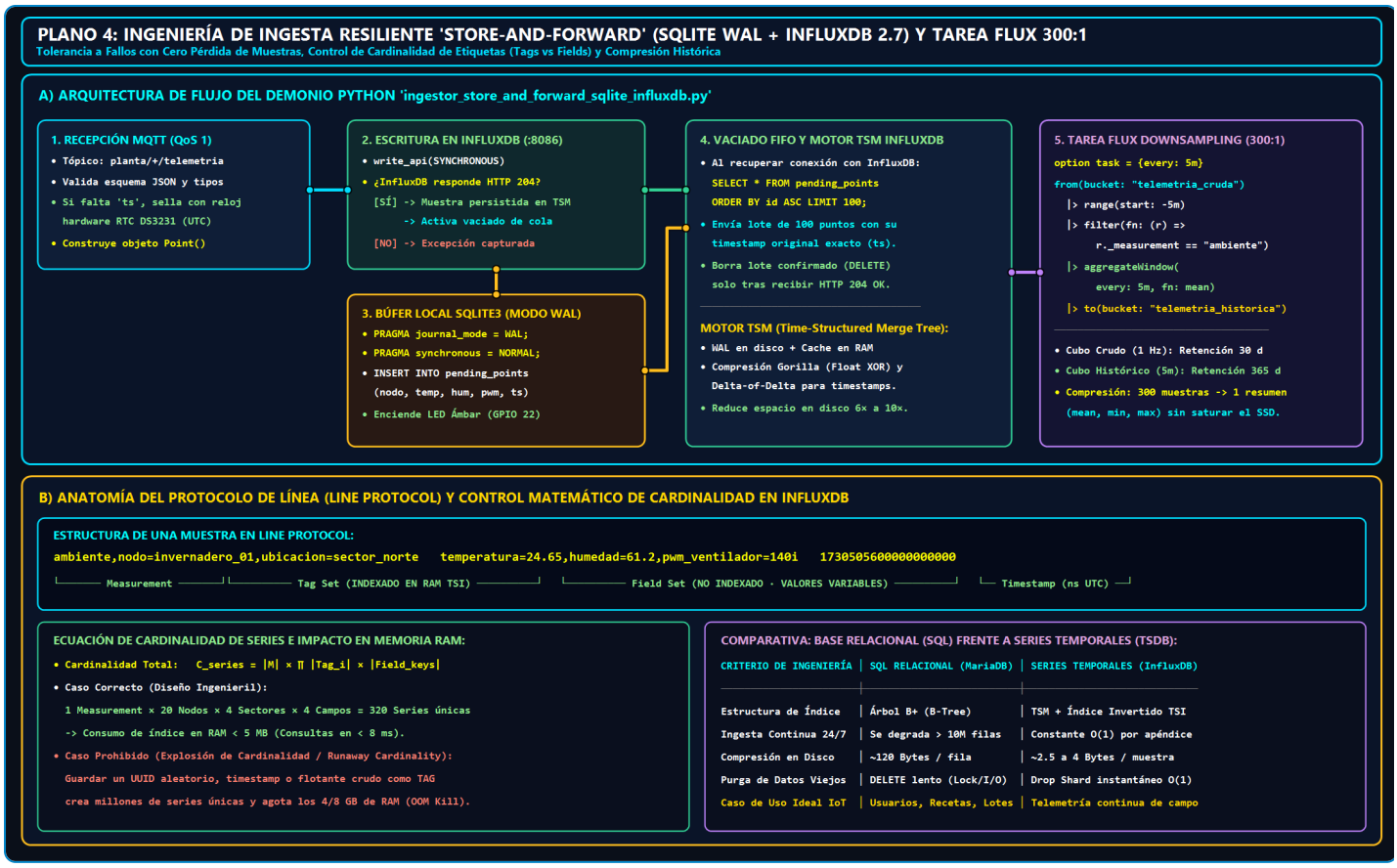
3. Patrón de Resiliencia *Store-and-Forward* con SQLite3 en Modo WAL

Para alcanzar una pérdida de datos igual a cero ($\text{Data Loss} = 0\%$) cuando el contenedor de InfluxDB se reinicia por mantenimiento o el enlace hacia un servidor central cae, nuestro demonio en Python implementa un búfer local en **SQLite3** activando la directiva:

```
PRAGMA journal_mode = WAL;
PRAGMA synchronous = NORMAL;
```

El modo **Write-Ahead Logging (`WAL`)** escribe las transacciones entrantes en un archivo secuencial de apéndice (`buffer_local_wal.db-wal`) sin bloquear las lecturas concurrentes y reduciendo drásticamente las operaciones de escritura aleatoria sobre las celdas Flash del SSD.

📐 **Plano 4: Flujo Store-and-Forward y Tarea Flux 300:1**



---

4. Ergonomía Visual bajo Norma ANSI/ISA-101 en Grafana

Un panel de control industrial no se evalúa por cuántos colores brillantes exhibe, sino por la velocidad con la que un operador humano detecta una condición anormal a las tres de la madrugada:

- **Contexto neutro de bajo contraste**: Fondos oscuros mate (`#0A101C` y `#121E36`) que reducen el deslumbramiento ocular.
- **Reserva cromática para anomalías**: Las curvas normales utilizan tonos cian o verde esmeralda moderados; el color **ámbar (`#FFBE1E`)** indica advertencia preventiva y el color **rojo (`#FF5050`)** se reserva exclusivamente para alarmas críticas activas.
- **Jerarquía de tres niveles**: Nivel 1 (KPIs globales y estado de salud), Nivel 2 (tendencias por subsistema con bandas sombreadas) y Nivel 3 (diagnóstico detallado de tramas, RSSI y latencias).

📐 Plano 5: Ergonomía Visual ISA-101 y Flujo Node-RED

PLANO 5: DISEÑO DE INTERFACES HMI BAJO NORMA ISA-101 EN GRAFANA Y FLUJO DE REGLAS EN NODE-RED

Conciencia Situacional para Operadores de Planta, Jerarquía de 3 Niveles, Histéresis de Alerta y Disponibilidad Operativa (SLA)

A) PANEL DE SUPERVISIÓN INDUSTRIAL GRAFANA (:3000) DISEÑADO BAJO ESTÁNDAR ANSI/ISA-101.01

TEMPERATURA PROCESO (NODO 01)	HUMEDAD RELATIVA Y PRESIÓN	PRE-ALARMA PREVENTIVA (ADVERTENCIA)	ALARMA CRÍTICA ISA-18.2 (PRIORIDAD 1)
24.6 °C	61.2 %RH | 1013 hPa	29.4 °C (Sector Sur)	0 ALARMAS ACTIVAS
Banda Nominal: [18.0 .. 28.0 °C] (OK)	Sensor Bosch BME280 (I2C 0x76)	Supera umbral preventivo > 28.0 °C	Disponibilidad SLA: A = 99.982% (Online)

CURVA DE SERIE TEMPORAL CON BANDA OPERATIVA SOMBREADA (CONSULTA FLUX aggregateWindow)



REGLAS DE ORO DE ERGONOMÍA VISUAL ISA-101:

1. Prohibido el efecto 'Árbol de Navidad': Si todo brilla en rojo, verde y amarillo saturado, el operador sufre fatiga visual y no detecta fallos.
2. Fondos oscuros mate (#0A101C / #121E36) y rejillas de bajo contraste para descanso ocular en turno 24/7.
3. Reserva cromática estricta:
   - Cian / Esmeralda -> Operación dentro de consigna
   - Ámbar (#FFBE1E) -> Desviación preventiva
   - Rojo (#FF5050) -> Alarma crítica que exige acción
4. Incluir siempre unidades físicas (°C, %RH, mA, dBm).

B) FLUJO DE ORQUESTACIÓN EN NODE-RED (:1880) CON HISTÉRESIS DE ALARMA Y CÁLCULO DE DISPONIBILIDAD (SLA)

① MQTT In
planta/+/telemetria
(QoS 1 · JSON)

② JSON Parser &
Schema Validator
(Verifica llaves)

③ Function Node
Filtro Histéresis
(T_on=30, T_off=28.5)

④ RBE / Filter
Report by Exception
(Bloquea spam)

⑤ Salida Dual:
• InfluxDB Batch Out
• Alerta Telegram/MQTT

LÓGICA JAVASCRIPT CON MEMORIA DE CONTEXTO E HISTÉRESIS EN NODE-RED:

```
const T_ALARMA_ON  = 30.0; // Umbral de disparo (°C)
const T_ALARMA_OFF = 28.5; // Umbral de reposición (Banda ΔT = 1.5 °C)
let enAlarma = context.get('enAlarma') || false;
if (!enAlarma && msg.payload.temp >= T_ALARMA_ON) {
    enAlarma = true;  context.set('enAlarma', true);
    return [msg, { payload: { alerta: 'ALTA_TEMP', valor: msg.payload.temp } }];
} else if (enAlarma && msg.payload.temp <= T_ALARMA_OFF) {
    enAlarma = false; context.set('enAlarma', false);
}
```

INGENIERÍA DE CONFIABILIDAD Y DISPONIBILIDAD DEL SISTEMA (MTBF / MTTR):

1. Ecuación de Disponibilidad Estacionaria (Availability):
   A = MTBF / (MTBF + MTTR) = T_operativo / T_total

2. Efecto de Docker 'restart: unless-stopped' + Healthchecks:
   Reduce el Tiempo Medio de Reparación (MTTR) de horas (intervención humana manual) a menos de 3.5 segundos (reinicio automático).

3. Combinado con el búfer SQLite WAL (Store-and-Forward):
   La disponibilidad efectiva de datos alcanza el 100.0% frente a microcortes de red o reinicios de mantenimiento de contenedores.