# ForgeSync "Global Industry 4.0"

**Project Sector:** Precision High-Tech Manufacturing (Automotive & Aerospace)

## 1. Project Overview: The Digital Nervous System

**The Vision**: "**ForgeSync Core**" is a next-generation **On-Premises-First** architecture. Unlike standard IT systems, **manufacturing cannot rely** on a **constant internet connection**. If the **connection** to the **cloud drops**, the **factory cannot stop**.

**The Core Problem: The "Siloed" Factory**

**ForgeSync's 4 Global Mega-Factories** (*Detroit, Stuttgart, Shanghai, São Paulo*) are currently "islands."

- **The 1% OEE Problem**: **Overall Equipment Effectiveness** is stuck at **68%.** Because data isn't synchronized, **managers don't know why machines** are **slowing** down **until the shift ends** (*This means 32% of our time and money is being wasted because machines are breaking down, running too slowly, or making broken parts*).
- **The Value**: **Moving** from 68% **to 69% OEE** is worth **$45 Million/year** in **saved** materials, reduced energy, and higher output.
- **The Constraint**: All **production-critical logic** (*Machine Control, Safety, Quality Vision*) **must** run **On-Premises** at the Factory Edge to **ensure sub-10ms latency** and **99.99 uptime** during **network outages**.

## 2. Business Scenario & Operational Environment

**The "On-Premises" Hierarchy:**

Your **architecture must follow** the **industrial standard** where layers are **separated by function** and **physical location**:

| Layer | Location | Function | Requirement |
|---|---|---|---|
| **Layer 0-2 (Shop Floor)** | On-Premises | PLCs, Sensors, Robotic Arms. | **Ultra-Low Latency:** <10ms response time. |
| **Layer 3 (Factory Edge)** | On-Premises | Manufacturing Execution System (MES), Local Data Historian. | **High Autonomy:** Must run for 48 hours without internet. |
| **Layer 4 (Corporate)** | Global HQ | ERP Integration, Global OEE Analytics, Long-term AI Training. | **Global Visibility:** Aggregated data for executive decision-making. |

### The "Forge" (Hardware Details):

1. **The Robotic Arm (The Worker):** A large **mechanical arm**. **If it moves too fast**, **it vibrates**. If **it rubs against metal**, it screeches. **If** it **works too hard,** it **gets hot**.

2. **The PLC (The Machine's Local Controller):** A rugged, **mini-computer inside** the **machine**. It **executes "Move**" or **"Stop" commands**. Your **software must tell** the **PLC** when to "**STOP" instantly**.

3. **The Sensors (The Senses)**

A **sensor** is a **tiny device** that **converts** a **physical** "feeling" into a **digital number**. You will **monitor 4 main sensors**:

| Sensor Name | What it feels | Digital Data | The "Danger" Sign |
|---|---|---|---|
| **Vibration** | The "shake" of the motor | **Hertz (Hz)** | ○ **12Hz** is a happy hum.<br>○ **85Hz** means a gear is loose. |
| **Acoustic** | High-pitched sound | **Decibels (dB)** | ○ **70dB** is normal.<br>○ **95dB** is a "Grinding" metal sound. |
| **Thermal** | Heat | **Celsius (°C)** | ○ **50°C** is fine.<br>○ **>80°C** means the motor is melting. |
| **Cycle Time** | Speed of work | **Seconds** | ○ **2.1s** is perfect.<br>○ **2.5s** means the robot is lagging. |

### Operational Constraints:

1. **Data Sovereignty:** By law, detailed "**Build Logs**" for aerospace parts produced in **Shanghai** or **Stuttgart** must **stay** on the **factory's physical servers for 10 years**. **Only "Summarized Health Stats"** can **leave** the **building**.

2. **The Firehose (Data Load):** 4 Factories x 12 Lines x 250 Sensors.
   - High-Velocity (Vibration/Acoustic): **100 reports per second** (100 Hz).
   - Low-Velocity (Thermal/Speed): **1 report per second** (1 Hz).

3. **Safety Interlock:** If the **centralized system sends a "Stop" command**, it must **reach** the machine in **under 5ms**.

4. **Hardware: Each factory** has its **own dedicated server room** with **high-availability clusters**.

5. **Latency Constraint (<5ms):** If a **machine** is **"Grinding"** you **cannot send** that **data** to the **Cloud** and **wait** for an **answer**. You **must process** it **locally** in **under 5 milliseconds** to **save** the **robot**.

6. **Autonomy Constraint (100% Uptime):** Even if the **factory loses internet**, the robots and your **software must keep working**.

## 3. The Assignment: 12 Detailed Tasks

### Section 1: Strategic Mapping

- **Assignment 1: Business Vision to Technical Vision.**
- **Assignment 2: Functional & Non-Functional Requirements.**

### Section 2: Architectural Selection

- **Assignment 3: Select Paradigm.**
- **Assignment 4: Select Model.**
- **Assignment 5: Select Architecture Style.**
- **Assignment 6: Select Architecture Pattern.**

### Section 3: Technical Design & Flow

- **Assignment 7: High-Level Design (HLD).**
- **Assignment 8: Low-Level Design (LLD).**
- **Assignment 9: Component & Service Selection.**
- **Assignment 10: Create 3 ADRs (Architectural Decision Records).**

### Section 4: Visualizing the Flow

- **Assignment 11: Create System Flow.**
- **Assignment 12: Final Architecture Picture.**

# Model Answer

## 1. Business Vision to Technical Vision

**Business Goal**: Increase OEE by 1% to capture $45 Million/year in value.

**Technical Vision**: Deploy an On-Premises Digital Twin architecture that provides the "Digital Nervous System" required for high-speed automated decision-making.

| Business Driver | Business Goal | Technical Objective (The "How") | Target Metric (The "Number") |
|---|---|---|---|
| **Availability** | Reduce unplanned machine breakdowns. | **Predictive Health.** Use Vibration/Thermal sensors to predict failure. | **99.99% Availability** of critical machines. |
| **Performance** | Eliminate "micro-stops" and slow cycles. | **Millisecond Tracking.** Real-time tracking machine cycle times. | **<10ms Data Latency** from Sensor to Dashboard. |
| **Quality** | Reduce scrap and wasted materials. | **Automated Interlocks.** Use Acoustics to trigger emergency stops. | **<5ms Response Time** for automated safety shut-offs. |
| **Resilience** | Maintain production during WAN outages. | **Edge Autonomy.** Keep all "Digital Twin" logic On-Premises. | **100% Factory Uptime** during Internet/Cloud outages. |
| **Reliability** | Ensure data for $45M savings is accurate. | **High-Fidelity Sync.** Robust local data storage and synchronization. | **Zero Data Loss (RPO=0)** for production-critical logs. |

## 2. Functional & Non-Functional Requirements (FRs & NFRs)

### a. Functional Requirements (FR): *What the "**ForgeSync Core**" system MUST do.*

| ID | Requirement Name | Description |
|---|---|---|
| FR-1 | **Digital Twin Sync** | The system shall capture and map physical sensor data (Vibration, Thermal, Acoustic, Cycle Time) to a digital record in real-time. |
| FR-2 | **Automated Interlock** | The system must automatically send a "Stop" command to the PLC if Thermal levels exceed 80°C or Acoustics detect grinding. |
| FR-3 | **Local Dashboarding** | The system shall provide a real-time OEE dashboard at the factory floor that updates every minute without requiring internet. |

| | | |
|---|---|---|
| FR-4 | **Data Buffering** | The system shall store all production data locally for up to 48 hours during a WAN/Internet outage. |
| FR-5 | **Predictive Alerting** | The system shall trigger a "Maintenance Alert" when vibration patterns deviate from the baseline (e.g., rising from 12Hz toward 85Hz). |

**b. Non-Functional Requirements (NFR)**: *How the system must perform to protect the **$45 Million** value.*

| ID | Category | Metric/Target | Reason for North Star |
|---|---|---|---|
| NFR-1 | **Latency** | **< 5ms** (*End-to-End*) | **Safety:** To trigger an emergency stop before machine damage occurs. |
| NFR-2 | **Availability** | **99.99%** (*High-Availability*) | **Uptime:** Every minute of system downtime causes a drop in OEE. |
| NFR-3 | **Resilience** | **100% On-Prem Autonomy** | **Reliability:** Factory production must not stop if the Cloud/Internet fails. |
| NFR-4 | **Data Integrity** | **Zero Data Loss** (*RPO=0*) | **Financials:** We cannot calculate the $45M savings if we lose the production logs. |
| NFR-5 | **Scalability** | **1.2M Events/Sec** | **Growth:** The system must handle the massive "Firehose" of vibration and acoustic data from 4 global factories. |

**c. Traceability Matrix**

- *"We have a Functional Requirement to stop the machine if it gets too hot (**FR-2**).*
- *However, that requirement is useless unless we meet the Non-Functional Requirement of **<5ms Latency (NFR-1)**.*
- *If the latency is **2 seconds**, the machine is already destroyed by the time the stop command arrives."*

**d. The Calculation: "The Firehose Math"**

To find the scalability requirement, we look at the **Source** (*The Sensors*) and the **Volume** (*The Factories*).

**i. The Sensor Count (Density)**

- Each of the **4 Global Factories** has **12 Production Lines**.
- Each Production Line has **250+ Sensors**.
- **Total Sensors:** 4 x 12 x 250 = **12,000 sensors globally**.
- **Events per second**: 12,000 sensors x 100 signals/sec = **1,200,000**.

### ii. Why this NFR is "The Firehose"

In your architecture, this number is a **Stress Test**.

If you design a system that can only handle 10,000 events per second, your "Digital Twin" will be "blind" to 99% of the vibration data.

**Without this NFR:**

- You miss the "Micro-shaked" that leads to a breakdown.
- You lose the **Availability** of the machine.
- You fail the **Metric** ($45M savings).

## 3. Select Paradigm

For the ForgeSync platform, the most effective choice is the **Reactive Paradigm** (*specifically using an **Event-Driven** approach*).

Reactive is a way of writing code that focuses on **Data Streams** and the propagation of change.

| Reason | Detail |
|---|---|
| **High Velocity** | ○ The system must process **1.2M events/sec** (Vibration, Acoustics).<br>○ Traditional "Request-Response" (Imperative) would crash under this load. |
| **Responsiveness** | ○ Reactive systems are designed to stay responsive under heavy load by being non-blocking. |
| **Resilience** | ○ In a Reactive paradigm, if one component fails, the system stays alive (Isolation). |
| **Real-time Logic** | ○ The "Digital Twin" needs to react immediately to a "Vibration Spike" or "Thermal Danger." |

### How it Works in ForgeSync (The Model)

In this paradigm, we do not "ask" the machine for its temperature every second.

Instead, the machine **"publishes"** its state, and our system **"reacts"** only when something important happens.

1) **Produce:** The Physical Sensor (Vibration) produces an "Event."
2) **Stream:** The event is pushed into a high-speed stream (e.g., MQTT or Kafka).
3) **React:**
    - **Logic A:** If vibration is **12Hz** (Normal) → Simply store it for OEE reports.
    - **Logic B:** If vibration is **85Hz** → (Danger) → Trigger the **Emergency Stop** immediately.

## 4. Create the Model

### a. The Digital Twin Concept

In the "**ForgeSync**" architecture, the Digital Twin is a **live data object** that mirrors the physical state of a machine.

It consists of three parts: **Identity**, **Telemetry** (The Senses), and **Logic** (The Brain).

### b. Physical-to-Digital Mapping Table

This table shows how we translate physical reality into digital data for a single **Robotic Milling Arm**.

| Physical Entity | Sensor Type | Digital Attribute (Field Name) | Normal Range | Trigger Logic (The Brain) |
|---|---|---|---|---|
| **Machine ID** | Asset Tag | twin_id | N/A | **Unique Identifier** (e.g., FORGE-DET-01) |
| **Motor Health** | **Vibration** (Accelerometer) | vibration_hz | 10Hz - 20Hz | **If > 85Hz:** Harmonic Spike Detected (*Alert Maintenance*). |
| **Friction Level** | **Acoustic** (Microphone) | noise_db | 60dB - 75dB | **If > 95dB:** Grinding Detected (*Trigger* ***Emergency Stop***). |
| **Speed/Flow** | **Cycle Time** (Laser) | cycle_time_ms | 2000 - 2100ms | **If > 2300ms:** Performance Lag (*Update OEE Dashboard*). |
| **Motor Heat** | **Thermal** (Thermocouple) | temp_celsius | 40°C - 65°C | **If > 80°C:** Overheating (Trigger **Safety Shutdown**). |

### c. Justification

- **Predictive:** By mapping vibration_hz, we catch **Harmonic Spikes** early, preventing a $500,000 motor failure.
- **Reactive:** By mapping noise_db, we detect **Grinding** and stop the machine in **<5ms**, saving the physical machine from destruction.
- **OEE Tracking:** By mapping cycle_time_ms, we identify exactly where the 1% efficiency loss is happening.

## 5. Select Architecture Style

- **Selected Style: Micro-kernel Architecture (Plug-in Style)**

For the **On-Premises Factory Edge**, we will use a **Micro-kernel** style combined with **Layered Architecture**.

- **Core Logic:** The "**Kernel**" handles mission-critical safety tasks (*Safety Interlocks, PLC communication*). This stays lean to ensure **<5ms latency**.
- **Plug-ins:** The **Digital Twin** models for different machines (*Milling, Casting, Assembly*) are "Plug-ins." This allows us to update one machine's logic without stopping the whole factory.
- **Isolation:** If the "Acoustic Analysis" plug-in crashes, the core "Machine Stop" kernel remains active. This supports **99.99% Availability**.

## 6. Select Architecture Pattern

- **Selected Pattern: Transactional Outbox Pattern**

This pattern is critical for **Edge Autonomy** and **Zero Data Loss**.

### How it works in ForgeSync

When a **Vibration Spike** or **Cycle Time** event occurs:

1. The system saves the event to a **Local Database** (*Outbox*) on the factory floor.
2. A separate "**Relay Service**" constantly **checks** this **Outbox** to **send data** to **Global HQ**.
3. **The WAN Outage Scenario:** If the **internet fails**, the **data** simply **stays** in the **Local Outbox**. When the **internet returns**, the **Relay Service "syncs**" everything in the **correct order**.

## 7. High-Level Design (HLD)

### a. The 3-Layer ForgeSync HLD

The architecture is divided into three physical and logical zones:

- **Zone 1: The Shop Floor (On-Premises - Real Time)**
  - **Components:** Sensors (*Vibration, Acoustic*), PLCs, Edge Gateways.
  - **Flow: *Raw signals → <5ms processing → Emergency Stop commands*.**
- **Zone 2: The Factory Core (On-Premises - Local Control)**
  - **Components:** Local Server Cluster, Digital Twin Registry, Outbox Database.

   - Flow: ***Event Aggregation → OEE Dashboarding → Local storage for 48-hour autonomy***.

- **Zone 3: The Global Command Center (Corporate Cloud)**
   - **Components:** Global Analytics, AI Training, Financial Reporting.
   - **Flow: *Summarized KPIs → Long-term Predictive Maintenance*** (PdM) trends.

### b. Communication Path

We use **MQTT (*Message Queuing Telemetry Transport*)** for **Zone 1** and **Zone 2** because it is **lightweight** and **handles** the "**Firehose**" of **1.2M events efficiently**.

We use **Secure gRPC** or **HTTPS** for **Zone 2** to **Zone 3** communication.

**Note**: **MQTT** (Message Queuing Telemetry Transport) is a lightweight, publish–subscribe messaging protocol designed for low-bandwidth, high-latency, or unreliable networks. It is widely used in IoT, real-time monitoring, and event-driven systems.

## 8. Low-Level Design (LLD)

### a. Data Storage Design

To **handle** the **1.2M events/sec** "**Firehose**" we use a **Two-Tier Storage Strategy** at the **Factory Edge**:

- **Tier 1: Time-Series Database (TSDB)**
   - **Purpose:** Stores high-velocity sensor data (*Vibration, Acoustics*).
   - **Schema:** [timestamp, machine_id, sensor_type, value].
   - **Retention:** 7 days (*Raw data is deleted after 7 days to save local space*).
   - **Note:** A **Time-Series Database (TSDB)** is a database optimized to store, query, and analyze data points indexed by time.
- **Tier 2: Relational Database (RDBMS)**
   - **Purpose:** Stores the "**Digital Twin**" metadata (*Identity, Thresholds*) and the **Transactional Outbox**.
   - **Schema:** [machine_id, status, last_maintenance_date, oee_target].

### b. Physical Deployment

- **Hardware: 3-Node High-Availability** (HA) **Server Cluster** located inside the Factory Server Room.
- **Storage:** NVMe SSDs to **support** the **high IOPS** (*Input/Output per second*) required for **1.2M events**.

## 9. Select Component, Feature, & Service

| Component | Selected Technology | Core Purpose | The "Architect's Why" (Deep Dive) |
|---|---|---|---|
| **Edge Gateway** | **Eclipse Kura** | **The Industrial Translator.** It converts physical machine signals into software data. | PLCs speak "Industrial" (Modbus/OPC-UA), but our "Digital Twin" speaks "Cloud." This component bridges that gap in **real-time**. |
| **Messaging Broker** | **EMQX (Enterprise MQTT)** | **The Digital Nervous System.** It carries the 1.2M events/sec throughout the factory. | Traditional protocols like HTTP are too "heavy." MQTT is ultra-lightweight, allowing us to send 100 reports/sec from 12,000 sensors without crashing the network. |
| **Stream Processor** | **Apache Flink (On-Prem)** | **The Intelligent Filter.** It watches the data stream for "Grinding" or "Harmonic Spikes." | We cannot save 1.2M events/sec forever. Flink analyzes the data **in flight**, triggers the <5ms safety stop, and only saves the important parts. |
| **High-Speed Storage (TSDB)** | **InfluxDB** | **The "Pulse" Recorder.** It stores the raw, high-speed sensor readings (Hz, dB, °C). | A standard database (RDBMS) would lock up trying to write at this speed. InfluxDB is designed to handle "Time-Series" data—millions of points with timestamps. |
| **Identity Storage (RDBMS)** | **PostgreSQL** | **The "History" Keeper.** It stores machine IDs, safety thresholds, and the **Outbox**. | We need "ACID" compliance (100% data accuracy). If the power blinks, PostgreSQL ensures the **Transactional Outbox** doesn't lose the data we need for the HQ. |
| **Orchestration** | **K3s (Lightweight K8s)** | **The Factory Brain Manager.** It keeps all our software services running. | If our "Acoustic Monitor" service crashes, K3s restarts it automatically in seconds. This is how we guarantee **99.99% Availability**. |
| **Service Protocol** | **gRPC** | **The Instant Command.** Used for machine-to-machine "Stop" signals. | gRPC is much faster than standard web calls. It allows the software to say "STOP" to the PLC in **under 5ms**, preventing motor melting. |
| **Analytics Tool** | **Grafana** | **The Profit Dashboard.** Displays the OEE health to the human operators. | If the manager can't see the **1% OEE gain**, the project is a failure. Grafana turns the messy data in InfluxDB into a clear "Green/Red" dashboard. |

**Note: K3s** is a lightweight, production-grade Kubernetes distribution created by Rancher (now part of SUSE), designed for resource-constrained environments like edge, IoT, and small on-prem setups**.**

**Note: EMQX** is a high-performance, distributed MQTT broker designed for massive-scale IoT and real-time messaging systems.

## 10. Architectural Decision Records (ADRs)

- **ADR-01: Choice of On-Premises over Cloud for Level 3**
  - **Decision:** All MES and **Digital Twin logic** will **reside** in the **factory**.
  - **Consequence:** Meets **(*100% Autonomy*)** and **(*<5ms Latency*)**.
- **ADR-02: Choice of MQTT as Primary Protocol**
  - **Decision:** Use **MQTT instead** of **HTTP/REST** for **sensor data**.
  - **Consequence: Massive reduction** in **network overhead**; supports the "Firehose" scalability.
- **ADR-03: Use of Time-Series Database for Telemetry**
  - **Decision: Store sensor data** in **InfluxDB** rather than standard SQL.
  - **Consequence:** Allows for **high-speed analysis** of **Harmonic Spikes** and **Acoustic signatures**.

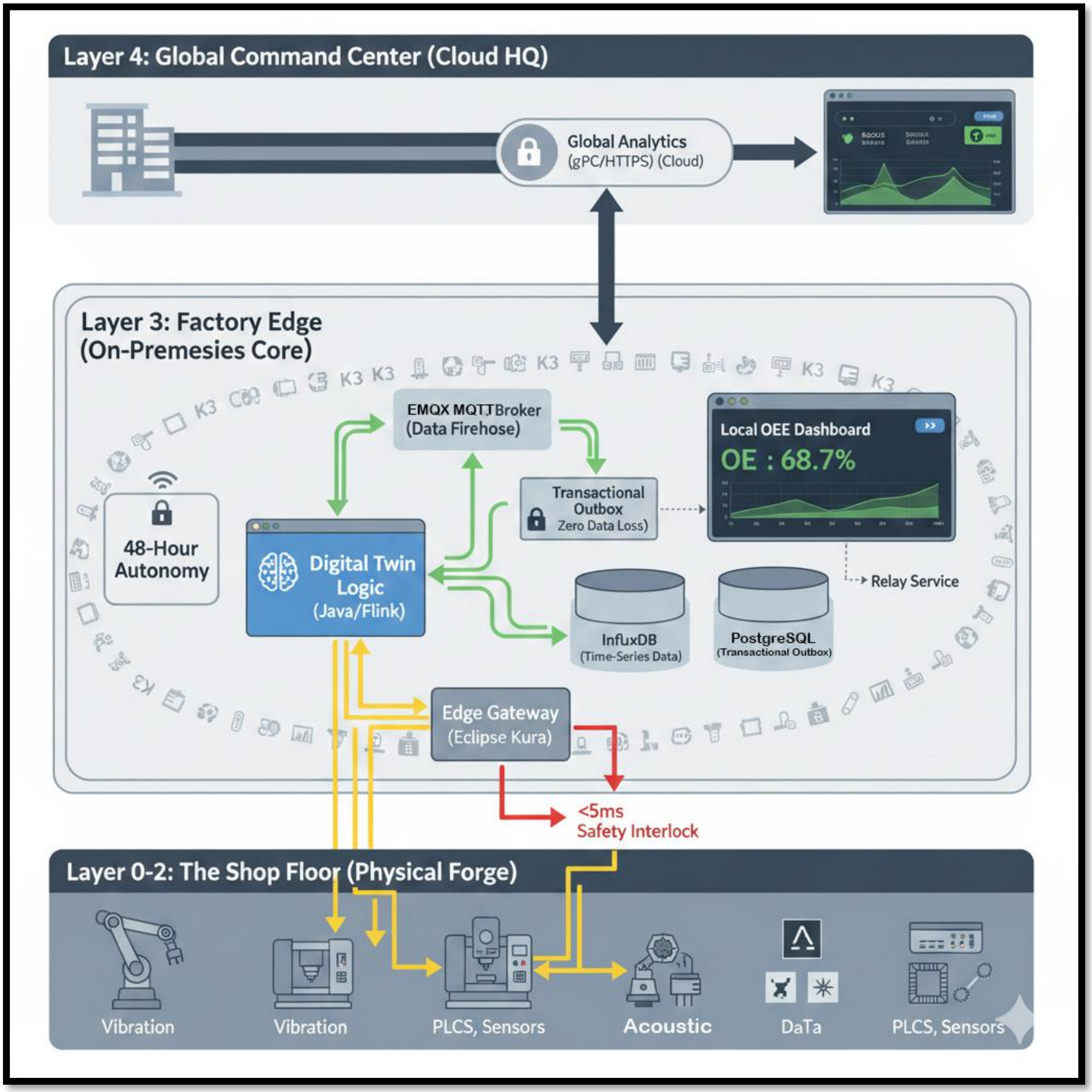
## 11. System Flow (The Vibration Spike)

- **Detection:** A physical **Vibration Sensor detects** a jump from 12Hz to **85Hz**.
- **Ingestion: Data** is **published** to the **On-Premises MQTT Broker** (*Latency: <1ms*).
- **Reaction:** The **Reactive Stream Processor** (*Flink*) identifies this as a "*Harmonic Spike*".
- **Action:** A **command** is **sent** via **gRPC** to the **PLC Interface** to **trigger** an **Emergency Stop**.
- **Synchronization:** The **event** is **written** to the **Transactional Outbox** (*PostgreSQL*).
- **Reporting:** Once the **WAN returns**, the **event** is **synced** to **HQ** for long-term AI training to prevent future spikes.

## 12. Final Architecture Picture

**The Diagram should depict:**

- **Bottom Layer:** The **physical** "**Forge**" with **Vibration/Acoustic sensors**.
- **Middle Layer (On-Premises):** The **MQTT Broker**, **Digital Twin Logic**, and **Local OEE Dashboard**. (The "Heart" of the system).
- **Side Layer:** The **Transactional Outbox** ensuring **data integrity**.
- **Top Layer (Global Cloud):** The **HQ receiving summarized OEE reports** via a secure tunnel.




1. **Layer 0-2: The Shop Floor (The Physical Forge)**

This is the bottom layer where the actual work happens.

- **The Senses:** These machines are equipped with **Vibration** and **Acoustic** sensors.
- **The Connection:** The yellow lines represent raw electrical signals and data being pulled from the machines and sent up to the "brain" (Layer 3).

2. **Layer 3: Factory Edge (The On-Premises Core)**

This is the "Heart" of your project. It is located physically inside the factory to ensure the system never stops.

- **Edge Gateway (Eclipse Kura):** This is the first stop. It translates machine language into digital data.
- **Safety Interlock (<5ms):** The **Red Arrow** pointing back down to the machines. This represents the ultra-fast emergency stop. If the Digital Twin hears "Grinding," it bypasses the cloud and stops the machine instantly.
- **The Firehose (EMQX MQTT Broker):** This handles the massive stream of 1.2M events per second.
- **Digital Twin Logic (Java/Flink):** This is the "Brain" where the virtual copy of the machine lives, evaluating health thresholds in real-time.
- **The Two Databases:**
  - **InfluxDB:** Storing the high-speed "Time-Series" waves (Vibration/Sound).
  - **PostgreSQL:** Storing the machine's "Identity" and the **Transactional Outbox**.
- **Local OEE Dashboard:** A screen showing **68.7% OEE**. This allows the local manager to see the $45M improvement progress without needing an internet connection.
- **48-Hour Autonomy:** The "lock" icon on the left indicates that this entire gray box can run perfectly even if the internet cable is cut.

3. **Layer 4: Global Command Center (Cloud HQ)**

This is the top layer, representing the Corporate HQ.

- **The Secure Tunnel:** The thick gray pipes with a lock icon represent a secure connection (VPN/gRPC) that only sends **summarized** data.
- **Global Analytics:** HQ doesn't see every single vibration; they only see the "Health Stats" and the total OEE across all 4 factories.
- **ERP/AI Training:** This is where long-term trends are analyzed to improve the entire global fleet.

4. **The Side Layer: The Transactional Outbox**

Located in the middle-right, the **Transactional Outbox** acts as a "Data Safety Net."

- If the connection to the Top Layer (HQ) breaks, the data is safely queued here.
- The **Relay Service** (dotted line) waits patiently for the internet to return, then "pushes" the saved data up to HQ so that no production logs are ever lost.