

DAY 1 / CLASS 1

Manufacturing Digital Twin & Data Architecture

공장의 물리적 시간을 디지털 공간의 데이터로 제어하다



TIME

10:00 ~ 11:00 (60min)



TOPIC

CPS, Protocol, Preprocessing

"여러분, 오늘 오실 때 네이버지도나 T맵 켜고 오셨나요?"



1. Reality (Physical)

실제 도로 상황

⚠ 차가 막히고 사고 발생

❗ 앞길을 알 수 없음



2. App (Cyber)

실시간 데이터 매핑

📍 도로 상황을 색상으로 표시

⚡ 실시간 위치 동기화



3. Prediction

최적 의사결정 지원

⌚ "이 길은 10분 더 걸립니다"

➡ 최적 우회 경로 제안

내비게이션이 바로 가장 대중적인 '**도로의 디지털 트윈**'입니다.

우리는 오늘 이 개념을 공장 (Factory)으로 옮길 것입니다.

From Navigation to Smart Factory

익숙한 내비게이션의 원리를 제조 현장에 그대로 적용해봅시다.

Navigation



차량 위치 & 도로 상황

실제 도로 위의 물리적 위치와 교통 체증



GPS & Traffic Data

수천 대 차량의 신호를 서버로 전송



"10분 뒤 도착 예정"

과거 패턴 기반 도착 시간 및 정체 예측



최적 경로 재탐색

더 빠른 길로 안내하여 시간 절약

Digital Twin



REALITY



설비 상태 (Vibration/Temp)

모터 진동, 베어링 온도, 압력 등 물리적 현상



IoT Sensor & PLC

1초에 100번(100Hz) 진동 데이터 수집 및 전송



AI



"3일 내 고장 확률 80%"

예지보전(PdM): 설비 수명 및 고장 시점 예측



자동 제어 & 정비 지시

설비 속도 조절 또는 정비팀 자동 호출

제조 디지털 트윈의 정의

Definition of Manufacturing Digital Twin



학술적 정의 (CPS)

Cyber-Physical System

물리적 자산과 디지털 모델이 센서와 네트워크로 1:1 연결된 통합 시스템.

- ✓ 물리(Physical) ↔ 가상(Cyber) 양방향 통신
- ✓ 폐루프(Closed-loop) 제어를 통한 최적화
- ✓ 실시간 데이터 동기화 기반



직관적 정의

Digital Mirror & Navigation

현재를 비추는 '**거울**'이자, 미래를 미리 가보고 최적의 길을 알려주는 '**내비게이션**'.

- ✓ 보이지 않는 내부 상태를 가시화 (Monitoring)
- ✓ "만약에(What-if)" 시나리오 모의 실험
- ✓ 과거-현재-미래를 관통하는 데이터 축

핵심 차이

단순 3D 모델 (CAD/CG)

정적 (Static)

보기에만 좋음, 죽은 정보

vs

디지털 트윈 (DIGITAL TWIN)

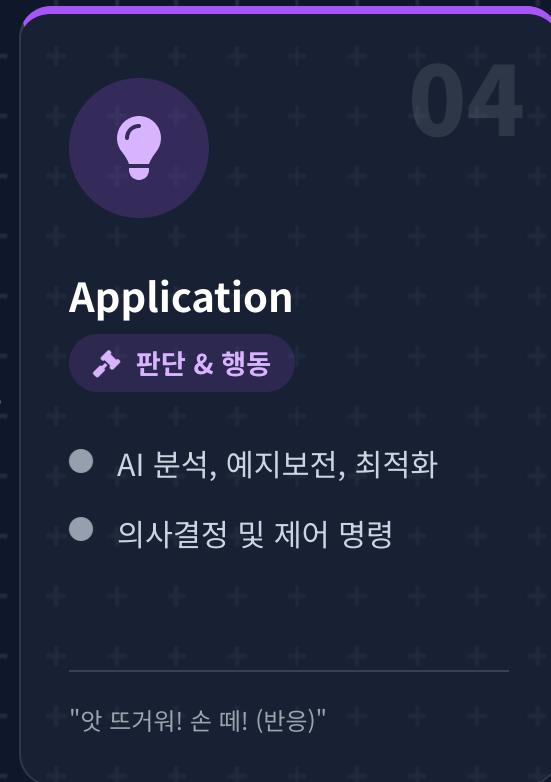
동적 (Dynamic)

살아 숨쉬는 데이터 (Live Data)

"데이터가 흐르지 않으면 트윈이 아니라 그냥 모델입니다."

CPS 4 Layer Architecture

제조 데이터 시스템의 구조를 인체(Human Body)에 비유하여 이해하기



핵심 포인트: 데이터가 끊기거나 느리면(Latency), 뇌는 화상을 입은 뒤에야 반응합니다.

⌚ Real-time ⚡ Sync

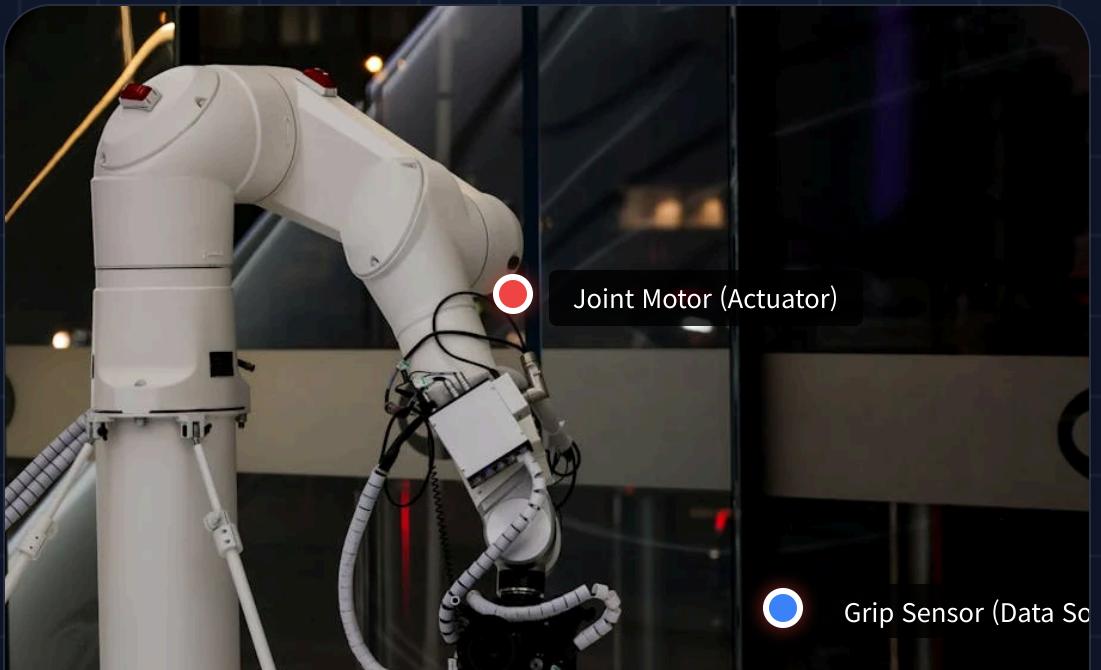
Layer 1 Physical

| 근육과 오감: 물리적 세계의 실행과 감지



Actuators (근육/실행)

로봇 팔, 컨베이어 벨트, 모터, 공구 등
실제로 물건을 옮기고, 자르고, 조립하는 물리적 설비



Sensors & Signals (오감/신호)

진동, 전류, 온도, 위치, 압력, 토크
설비의 상태와 작업 결과를 전기적 신호로 변환



Industry Case: 현대자동차 울산공장

IoT 센서가 내장된 AMR(자율주행 물류로봇) 200여대를 운영하여 물리적 물류 자동화 실현

실시간 장애물 회피

동적 경로 설정

Smart Factory Line #01

Temp

Current

Position

Running

Layer 2 Connectivity & Data

| 신경망: 현장의 신호를 전달하는 연결의 기술



Infrastructure (신경망 구성)

IoT 센서, PLC, 게이트웨이, 5G/Ethernet

현장의 물리적 신호(진동, 온도)를 디지털 패킷으로 변환



Real-time & Latency (반사 신경)

"손가락이 뜨거우면 0.1초 만에 뇌로 전달"

지연 없는 데이터 흐름이 사고 예방과 정밀 제어의 핵심

INDUSTRY CASE STUDY



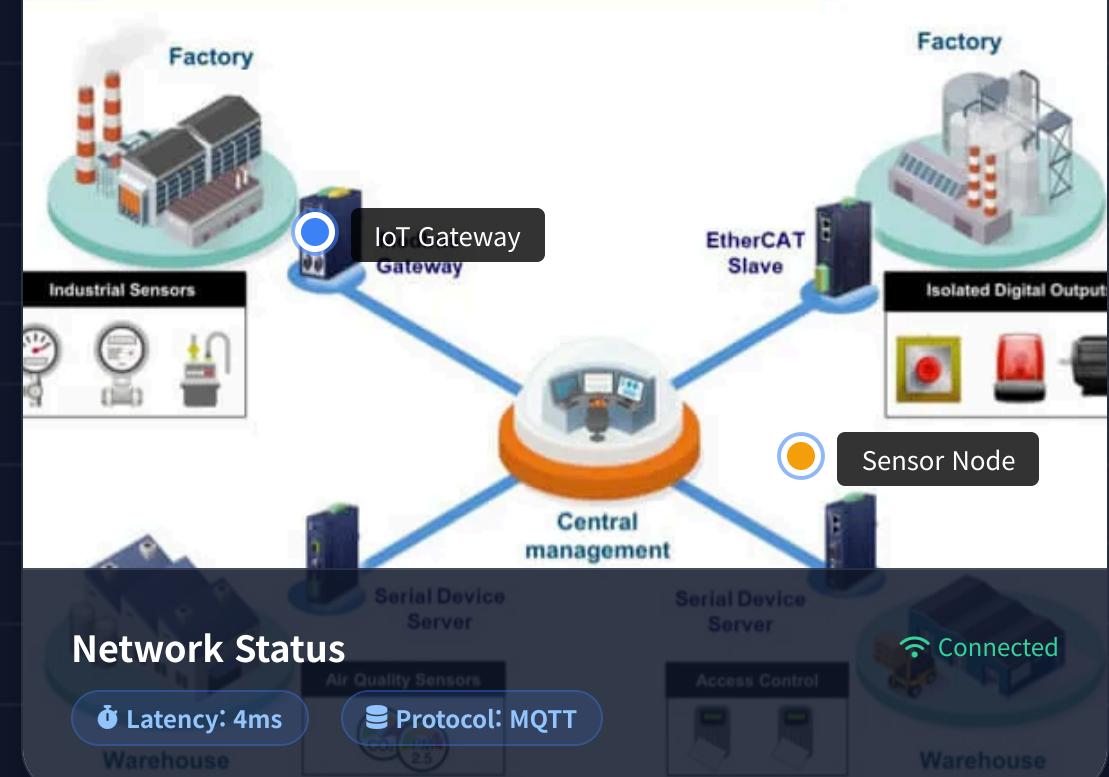
현대오토에버 (Hyundai AutoEver)

IoT Platform & Connectivity Integration

✓ 9개 제조사의 **PLC 장비 17종 프로토콜** 완벽 지원

✓ OPC-UA 등 표준 산업용 **프로토콜** 다수 지원으로 서비스 통합 가속화

Industrial Automation Network



Layer 3 Cyber

| 뇌와 기억: 데이터의 저장과 가상화



Infrastructure (뇌/저장소)

클라우드(AWS/Azure), 온프레미스, 데이터 레이크
수집된 방대한 '기억(데이터)'을 저장하고 처리하는 공간

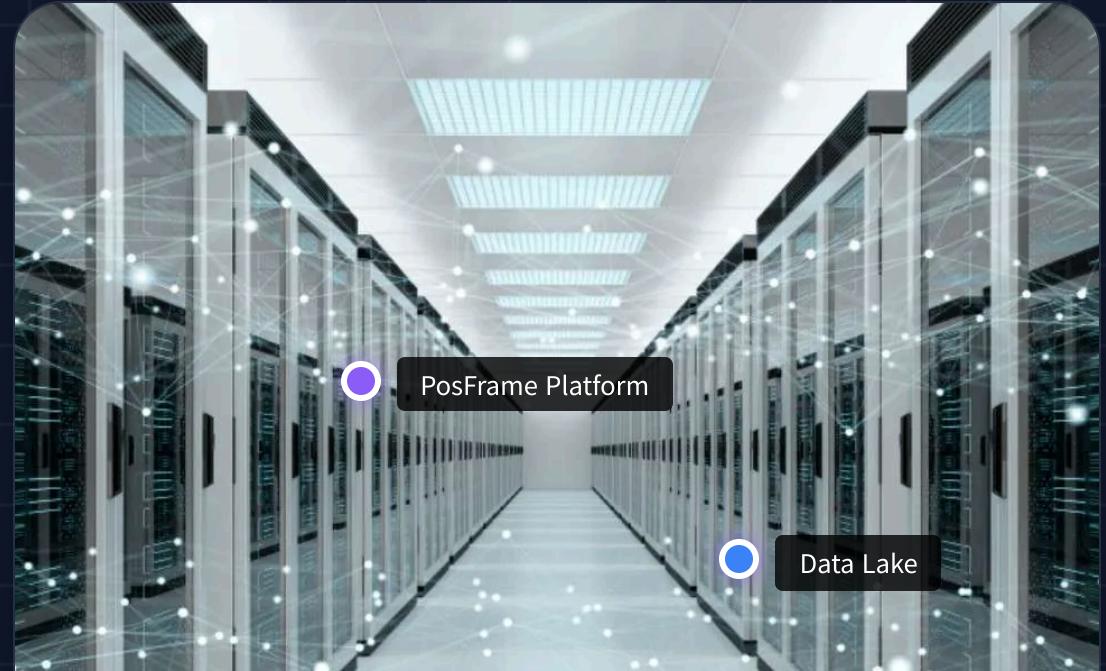


Digital Thread (기억/연결)

3D 가상공간 매핑, 데이터 정제 및 카탈로그
설계-생산-운영 데이터를 하나로 퀘어 맥락(Context) 부여

■ 산업 적용 사례: POSCO (PosFrame)

- ⚡ **초고속 데이터 처리:** 정형/비정형 및 마이크로(센서)/매크로(조업) 데이터를 실시간 수집·통합 분석
- ☞ **Digital Thread 구현:** 연속 공정의 조업 데이터를 하나로 연결하여 선후 공정 간 품질 결함 요인 추적 및 제어



Smart Factory Data Hub



Big Data



Analytics



Steelmaking

Layer 4 Service/Application

| 판단과 행동: 데이터 기반 의사결정과 제어



Analysis & Insight (판단)

이상 탐지, 예지 보전, 품질 최적화

"왜 고장났지?", "언제 고장날까?"를 AI가 판단



Action & Control (행동)

"앗 뜨거워, 손 떼!" → 제어 명령 전송

분석 결과를 바탕으로 즉시 서비스 제어 및 작업 지시



CASE STUDY LG전자 창원공장 (LG Smart Park)

96% ↓

설비 중단 시간 감소

AI 예지보전 적용

3분 / 8h

품질 예측 시간 단축

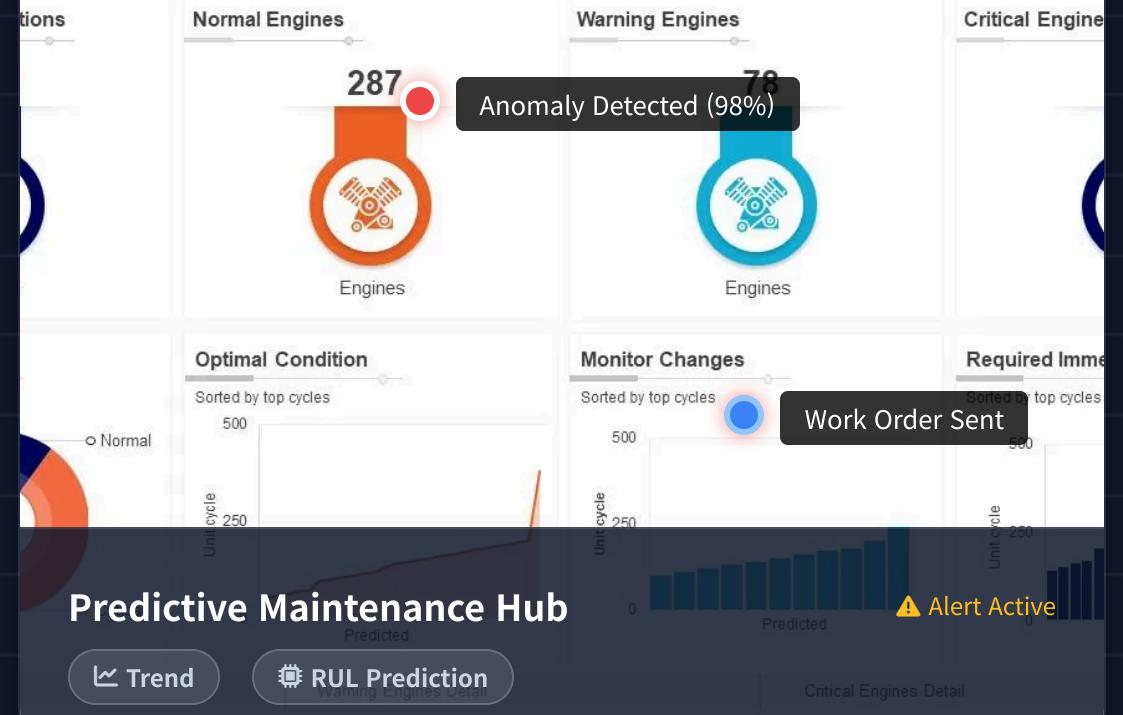
99% 효율 향상



✓ WEF 등대공장 선정

Maintenance Dashboard with Total Engines in Operations

This maintenance dashboard such as normal, warning and critical engines with total engine status, optimal condition, monitor changes, requires immediate maintenance.



The Factory of Babel (바벨탑의 공장)

⚠ 표준이 없으면 기계들은 서로 '외계어'를 쓴다

💡 "Ich bin bereit!" (Siemens PLC)

💡 "準備完了" (Mitsubishi Robot)

💡 "Hello World" (Analysis)



CURRENT PROBLEM

Protocol Fragmentation

- ✖ 상호 운용성 부재: 장비 제조사마다 독자적인 통신 규격 사용 (S7, MELSEC, Modbus 등)
- ✖ Data Silo: 데이터가 각 장비 안에 고립되어 통합 분석 불가능
- ✖ 비용 증가: 장비 연결할 때마다 전용 드라이버 개발 필요 (N:N 연결)



THE SOLUTION

Standard Protocols

모든 기계가 이해할 수 있는 '국제 공용어(English)'를 도입해야 합니다.

MQTT

OPC-UA

AAS

Protocol Battle: MQTT vs OPC-UA

어떤 상황에 무엇을 써야 할까? 특징과 용도 비교



MQTT

Message Queuing Telemetry Transport

⚡ 특징: 초경량, 고속

헤더가 작고(2byte), 네트워크 대역폭을 최소화함.

📻 방식: Publish / Subscribe

구독형 모델. "나 온도 50도야!"라고 뿌리면 필요한 놈이 받아감.

📱 용도: IoT 센서, 모바일

네트워크가 불안정한 현장이나 배터리 구동 기기에 적합.



ANALOGY

트위터 / 카카오톡: 가볍고 빠르지만, 내용 형식은 자유로움(Text, Img).

VS



OPC-UA

Unified Architecture

➊ 특징: 정보 모델링, 강력한 보안

데이터의 의미(메타데이터)까지 정의함. 무겁지만 확실함.

🤝 방식: Client / Server

요청-응답 모델. 인증서 기반의 엄격한 연결 관리.

💻 용도: 설비 제어, MES/ERP 연동

PLC와 상위 시스템 간의 신뢰성 있는 데이터 교환.



ANALOGY

외교 문서 / 계약서: 형식이 복잡하고 엄격하지만, 오해의 소지가 없음.

AAS (Asset Administration Shell)

설비의 디지털 여권 (Digital Passport)



Physical Asset + Digital Shell = I4.0 Component



Concept: 디지털 신분증

"나는 ○○사 로봇이고, 스펙은 △△입니다."
기계의 모든 정보(ID, 제조사, 기술사항)가 담긴
표준화된 디지털 파일.



Role: Plug & Play

PC에 마우스를 꽂으면 드라이버가 자동 설치되듯,
공장 네트워크에 서비스를 연결하는 즉시 시스템
이 인식함.



Structure: Submodels

기능별로 정리된 정보 서랍.
[명판], [기술 데이터], [매뉴얼], [운영 상태],
[유지보수 이력] 등으로 구성.



Effect: 상호운용성

제조사가 달라도 컱데기(Shell) 규격이 같으므로,
별도의 통역사(Converter) 없이 데이터 교환
가능.

센서 데이터의 민낯 (Raw Data)

"여러분이 상상한 깔끔한 엑셀 파일은 학교에만 있습니다."



Signal Noise (노이즈)

전기적 간섭, 진동으로 인한 자잘한 떨림.
값의 신뢰도를 떨어뜨림.



Spike / Outlier (이상치)

순간적인 전압 튀김이나 센서 오류.
평균값을 왜곡시키는 주범.



Missing Data (결측)

네트워크 지연, 배터리 방전 등으로
데이터가 수집되지 않은 구간.



Sensor Drift (드리프트)

센서 노후화로 인해 영점(Zero point)이
서서히 틀어지는 현상.

Vibration Sensor #404 (Raw Stream)

Freq: 100Hz

Status: Anomaly Detected



“현장 데이터는 ‘흙 묻은 감자’와 같습니다.

이를 씻고 다듬는 **전처리(Preprocessing)** 과정 없이는 어떤 AI도
작동하지 않습니다.

🐍 Python Preprocessing Practice

pandas와 scipy를 활용한 실제 센서 데이터 세척 코드 예제

● ● ● 🗃 preprocessing.py

```
import pandas as pd
from scipy.signal import medfilt

# 1. Load Raw Sensor Data
df = pd.read_csv('sensor_log.csv')

# 2. Fill Missing Values (Interpolation)
df['value'] = df['value'].interpolate(method='linear')

# 3. Filter Noise (Median Filter)
# kernel_size=5 means window of 5 points
df['clean'] = medfilt(df['value'], kernel_size=5)

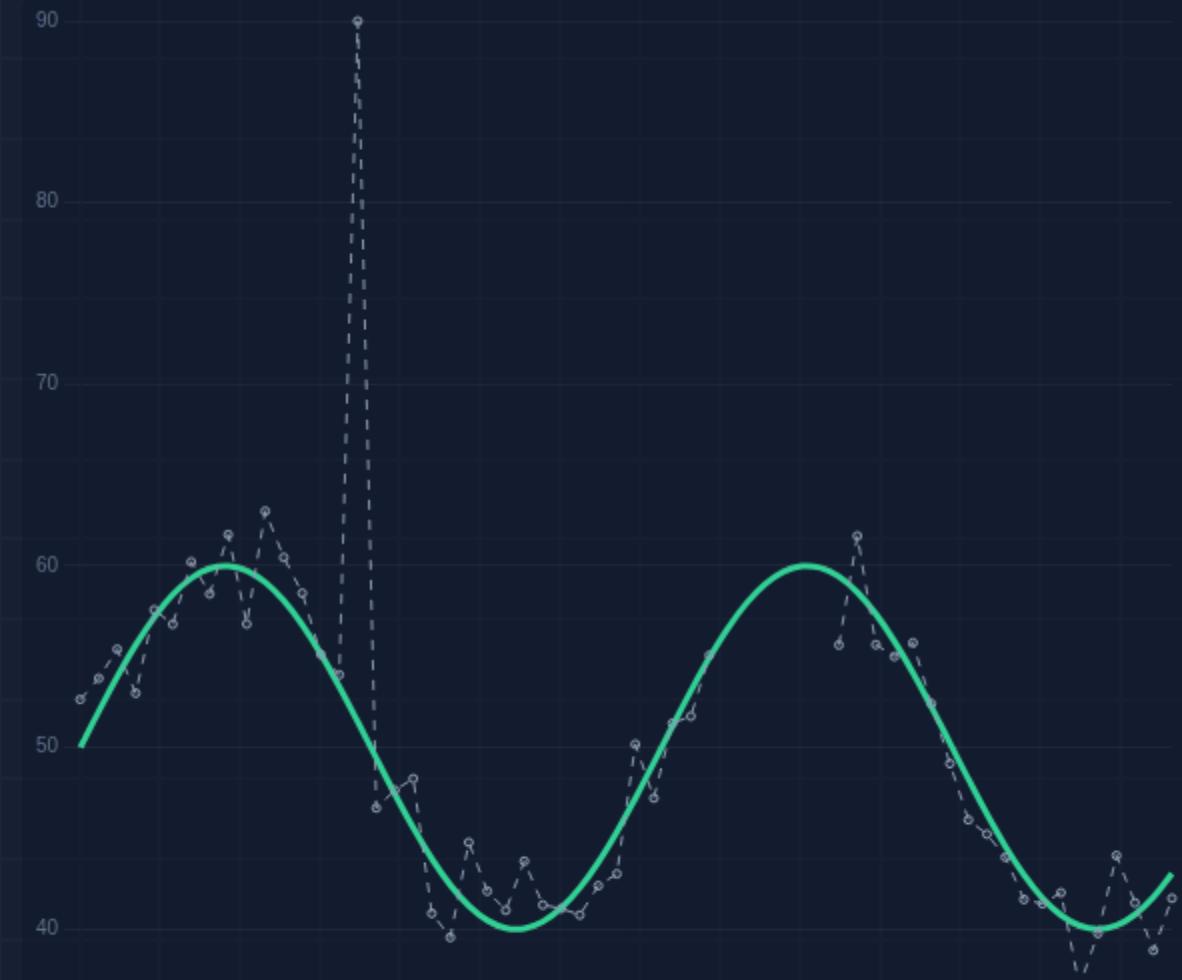
# 4. Remove Outliers (Z-Score > 3)
mean = df['clean'].mean()
std = df['clean'].std()

# Filter rows within 3 standard deviations
df_final = df[abs(df['clean']) - mean] < 3 * std

print(f"Data Cleaned: {len(df)} -> {len(df_final)} rows")
# Save result
df_final.to_csv('cleaned_data.csv', index=False)
```

Result Visualization

● Raw Data ● Cleaned



전처리(Preprocessing) 전략

▼ Garbage In, Garbage Out: 데이터 세척의 5단계 프로세스



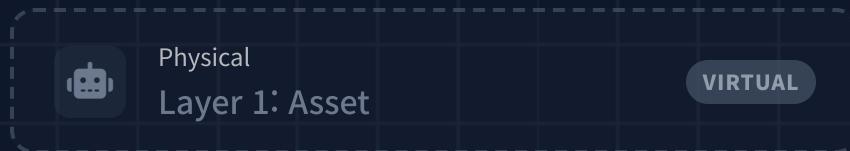
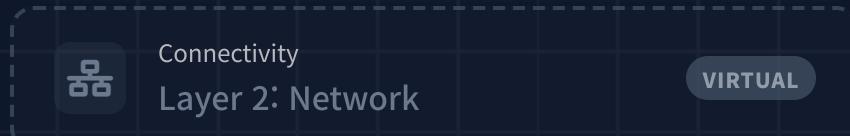
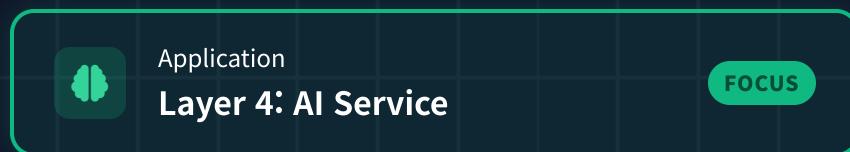
Coming Up Next: Practice Session

"잠시 후 2교시 실습 때, 여러분이 직접 파이썬으로 이 '흙 묻은 데이터'를 씻게 될 겁니다."

Hands-on Lab

오늘 우리의 목표 (Today's Goals)

선택과 집중: 가상 시뮬레이터와 실전 AI 분석



Key Achievements

Data Pipeline 구축

Raw Data(센서)에서 DB 저장까지의 **전체 데이터 흐름**을 직접 코드로 구현합니다.

분석 대시보드 Sketch

Streamlit을 활용하여 실시간으로 들어오는 데이터를 **시각화(Chart)**하는 방법을 익힙니다.

LLM Service Chain Demo

간단한 RAG(검색 증강 생성)를 통해 AI가 "**기계 매뉴얼**"을 읽고 답변하는 데모를 체험합니다.

Protocol 이해 (MQTT)

가상의 센서가 MQTT 브로커로 메시지를 **발행(Publish)**하고 수신하는 과정을 이해합니다.

Q & A

💬 오늘 학습한 내용에 대해 무엇이든 물어보세요.

SUGGESTED DISCUSSION TOPICS



디지털 트윈 설계 사례



프로토콜 선택 가이드



데이터 전처리 팁

Next Session: **Hands-on Lab (11:10~)**

Break Time: 10 mins