IE 442 Research Project Report
Monitoring Manufacturing Systems with IOT
Fall 2024/2025

Prepared by: Sude Filiz

Id: 2021402273

Over the past few years, IoT has become one of the most important technologies of the 21st century. The Internet of Things (IoT) is a system that enables physical devices, machines, and other objects to collect and share data through sensors, software, and network connectivity. This technology enhances machine-to-machine communication in manufacturing processes, improving operational efficiency (IBM, 2025)
By means of low-cost computing, the cloud, big data, analytics, and mobile technologies, physical things can share and collect data with little assistance from humans. Digital systems are able to record, track, and modify every interaction between linked things in this hyperconnected world. The physical world meets the digital world and works together.
Industrial IoT (IIoT) is a specialized application of IoT technologies in industrial environments, particularly in manufacturing, where automation and real-time data analysis are made possible by sensors, cloud computing, and machine-to-machine (M2M) communication. IIoT improves operational efficiency, process optimization, and predictive maintenance by combining analytics and machine learning, which reduces downtime and improves productivity. This approach aligns with the principles of Industry 4.0, offering manufacturers the ability to monitor operations remotely, track asset performance, and make data-driven decisions (Oracle, 2025).

IoT in Manufacturing

The integration of IoT in manufacturing has revolutionized industrial operations by enabling real-time monitoring, automation, and predictive maintenance. By leveraging sensor technology, cloud computing, and data analytics, manufacturers can optimize efficiency, reduce downtime, and enhance product quality. IoT-driven predictive maintenance helps detect equipment failures before they occur, minimizing costly disruptions. Additionally, remote monitoring allows manufacturers to oversee production processes from anywhere, ensuring operational flexibility and improved decision-making.

Manufacturing is the **largest segment** of IoT applications, with industrial IoT expected to surpass **$500 billion** in market size by 2025. By 2025, IoT platforms could contribute **$3.7 trillion** annually to the manufacturing sector.

Scope and Methodology: Detailed Implementation

This section provides a structured methodology for designing an IoT-based monitoring system for manufacturing processes. The physical hardware implementation is not possible in this project, data collection is simulated using the 'Production Plant Data for Condition Monitoring' dataset from Kaggle. Since system design and implementation are important stages in the development of IoT-based monitoring systems, a comprehensive and detailed explanation will be provided, encompassing technical components and implementation methods to ensure a complete understanding of the process.

1. System Design

System design is a essential phase in an IoT-based manufacturing monitoring system. It includes identifying the key parameters to be monitored, building the sensor network, and defining the data transmission protocols that will guarantee smooth hardware-cloud infrastructure cooperation.

1.a Identifying Key Manufacturing Parameters to Monitor:

In order to guarantee efficient monitoring and predictive maintenance, the selection of manufacturing parameters is critical. The following parameters are frequently used in industrial IoT (IIoT) applications, based on industry best practices and prior research:(BU KISMI DATA İLE GÜNCELLEMEK GEREKİR Mİ BAK???)

- **Temperature:** Monitoring temperature variations can help detect overheating issues gin machines, which is crucial for preventing breakdowns (Lee, J., "Smart Factory and Industry 4.0," *International Journal of Manufacturing Research*, 2020).
- **Vibration:** Abnormal vibration patterns can indicate mechanical faults in rotating equipment, such as motors and turbines (Kumar, R., *Industrial IoT Applications for Smart Manufacturing*, 2021).
- **Pressure:** Pressure variations in hydraulic or pneumatic systems can provide early warnings about potential failures (Uckelmann & Harrison, *Architecting the Internet of Things*, Springer, 2011).
- **Motor Speed:** Real-time monitoring of motor speed helps optimize operational efficiency and prevents mechanical stress (Krumm, J., *Ubiquitous Computing Fundamentals*, CRC Press, 2016).
- **Load Values:**Analyzing mechanical load can assist in predictive maintenance and operational efficiency analysis.

The selected Kaggle dataset, "Production Plant Data for Condition Monitoring," contains real-world sensor readings for these parameters, making it a suitable proxy for simulated real-time IoT data.

Implementation Approach

Sensor Placement: Temperature, vibration, and pressure sensors would be physically installed on machinery at key monitoring points in a real-world implementation.

Data Acquisition: Sensors would gather real-time measurements at predefined intervals (e.g., every second or minute) and transmit them to an IoT gateway (Raspberry Pi or an alternative edge computing device).

1.b Designing a Network of IoT-Enabled Sensors to Collect Data

It is necessary to create a distributed network of IoT-enabled sensors to guarantee effective data collection. This network consists of:

Physical Sensors: Devices such as:

- **Temperature sensors** (DHT22, DS18B20) for monitoring heat levels.
- **Vibration sensors** (MPU6050, ADXL345) for detecting abnormal mechanical movement.
- **Pressure sensors** (BMP280, MPX5700) for ensuring fluid and gas flow consistency.

Microcontrollers & Edge Devices:

- **Raspberry Pi** acts as an IoT gateway that aggregates data from multiple sensors.
- **ESP8266/ESP32** for wireless sensor node deployment when remote monitoring is required.

Data Transmission to Cloud: Data must be sent to a cloud server for analysis and storage after it has been collected.

Method of Implementation:

Sensor-to-Gateway Communication: Sensors connect to Raspberry Pi via GPIO, I2C, SPI, or UART interfaces.

Edge Processing: The Raspberry Pi can perform initial data filtering to reduce transmission bandwidth.

Wireless Communication: If required, LoRaWAN, Zigbee, or Wi-Fi can be used to connect multiple sensor nodes over long distances.

"In an IoT-based monitoring system, an efficient sensor network is crucial for real-time data collection and analysis. This system consists of various sensors such as temperature, vibration, and pressure sensors, which continuously monitor different aspects of machine performance. These sensors transmit data to a central processing unit, like a Raspberry Pi, which acts as an IoT gateway. The collected data is then processed and sent to the cloud using communication protocols such as LoRaWAN, Zigbee, or Wi-Fi. This setup enables remote monitoring, predictive maintenance, and operational efficiency, ensuring a smarter and more automated manufacturing process."

1.c Defining Data Transmission Protocols (e.g., MQTT, HTTP):

Once sensor data is collected, it must be efficiently and securely transferred to the cloud. The communication protocol used is determined by the system's latency requirements, bandwidth constraints, and scalability needs.

a. Message Queuing Telemetry Transport (MQTT)

MQTT is a lightweight, publish-subscribe protocol widely used in IoT applications.
Because it uses less bandwidth, making it ideal for real-time monitoring in industrial settings.
To control message exchange between sensors and cloud platforms, the protocol uses a broker (e.g., Eclipse Mosquitto, HiveMQ).

Establishing MQTT Connection with Python

(Figure 1: Code snippet demonstrating MQTT-based message exchange in an IoT system.)

b. Hypertext Transfer Protocol (HTTP)

HTTP is a widely used protocol for REST-based communication.
It is appropriate for less frequent data transmissions, such as periodic cloud updates. But it is not as effective as MQTT for real-time IoT.

Python Script for Cloud Data Transmission

(Figure 2: Sending IoT sensor data to ThingSpeak using HTTP requests in Python.)

2. Implementation

Now that we have established the system design, the next step is its implementation.In IoT-based systems, efficient data transmission is essential to ensure seamless communication between sensors and cloud platforms. Two major protocols used are MQTT and HTTP.
MQTT is a lightweight, publish-subscribe protocol that minimizes bandwidth usage, making it ideal for real-time monitoring. It relies on brokers like Mosquitto for message exchange.
On the other hand, HTTP is commonly used in web-based communication and is more suited for periodic data transfers rather than continuous real-time updates. Depending on system needs, these protocols can be implemented to ensure effective data communication in IoT environments.

2.a Hardware: Using Raspberry Pi as the IoT Gateway

Raspberry Pi is the name of a series of single-board computers made by the Raspberry Pi Foundation, a UK charity that aims to educate people in computing and create easier access to computing education. The Raspberry Pi serves as a versatile IoT gateway, capable of handling sensor data processing and communication efficiently (Opensource.com, n.d.) Sensors communicate with Raspberry Pi through GPIO, I2C, SPI, or UART, enabling efficient data transfer and preprocessing. The role of edge computing in Industrial IoT (IIoT) has been explored extensively (IEEE Transactions on Industrial Informatics, 2021)

2.b Data Transmission: Utilizing MQTT Protocol

MQTT is an OASIS standard messaging protocol for the Internet of Things (IoT). It is designed as an extremely lightweight publish/subscribe messaging transport that is ideal for connecting remote devices with a small code footprint and minimal network bandwidth (MQTT.org, n.d.).

MQTT brokers (such as Mosquitto) can be deployed locally or in the cloud for message management.

2.c Cloud Integration: AWS IoT Core / ThingSpeak

AWS IoT Core: AWS IoT provides device software that can help you integrate your IoT devices into AWS IoT-based solutions (AWS, n.d.). AWS IoT Core offers advanced features such as device shadowing, rule-based processing, and machine learning analytics.

ThingSpeak: ThinkSpeak provides a simple RESTful API for real-time sensor data visualization.

In conclusion; The implementation of an IoT-based monitoring system involves hardware selection, data transmission, and cloud integration. As an IoT gateway, the Raspberry Pi efficiently processes and transmits sensor data through various communication interfaces. **MQTT**, with its lightweight publish/subscribe model, ensures efficient real-time messaging. For cloud integration, ThingSpeak offers a more straightforward RESTful API for real-time data visualization, which is perfect for smaller projects, while AWS IoT Core offers scalability, advanced analytics, and security, making it suitable for large-scale industrial applications.

Establishing MQTT Connection with Python

```python
import paho.mqtt.client as mqtt

broker_address = "mqtt.eclipse.org"
client = mqtt.Client("ManufacturingMonitor")
client.connect(broker_address)

# Publish sensor data
client.publish("factory/data", "Temperature: 75°C, Vibration: 2.1 m/s²")
```

(Figure 1: Code snippet demonstrating MQTT-based message exchange in an IoT system.)

b. Hypertext Transfer Protocol (HTTP)
HTTP is a widely used protocol for REST-based communication.
It is appropriate for less frequent data transmissions, such as periodic cloud updates. But it is not as effective as MQTT for real-time IoT.

Python Script for Cloud Data Transmission

```python
import requests

api_key = "YOUR_API_KEY"
data = {"field1": 75, "field2": 2.1}

requests.post(f"https://api.thingspeak.com/update?api_key={api_key}",
              params=data)
```

(Figure 2: Sending IoT sensor data to ThingSpeak using HTTP requests in Python.)

2. Implementation

Now that we have established the system design, the next step is its implementation.

2.a Hardware: Using Raspberry Pi as the IoT Gateway

Raspberry Pi is the name of a series of single-board computers made by the Raspberry Pi Foundation, a UK charity that aims to educate people in computing and create easier access to computing education. The Raspberry Pi serves as a versatile IoT gateway, capable of handling sensor data processing and communication efficiently (Opensource.com, n.d.) Sensors communicate with Raspberry Pi through GPIO, I2C, SPI, or UART, enabling efficient data transfer and preprocessing. The role of edge computing in Industrial IoT (IIoT) has been explored extensively (IEEE Transactions on Industrial Informatics, 2021)

2.b Data Transmission: Utilizing MQTT Protocol

MQTT is an OASIS standard messaging protocol for the Internet of Things (IoT). It is designed as an extremely lightweight publish/subscribe messaging transport that is ideal for connecting remote devices with a small code footprint and minimal network bandwidth (MQTT.org, n.d.).
MQTT brokers (such as Mosquitto) can be deployed locally or in the cloud for message management.

2.c Cloud Integration: AWS IoT Core / ThingSpeak

AWS IoT Core: AWS IoT provides device software that can help you integrate your IoT devices into AWS IoT-based solutions (AWS, n.d.). AWS IoT Core offers advanced features such as device shadowing, rule-based processing, and machine learning analytics.
ThingSpeak: ThinkSpeak provides a simple RESTful API for real-time sensor data visualization.

In conclusion; The implementation of an IoT-based monitoring system involves hardware selection, data transmission, and cloud integration. As an IoT gateway, the Raspberry Pi efficiently processes and transmits sensor data through various communication interfaces. **MQTT**, with its lightweight publish/subscribe model, ensures efficient real-time messaging. For cloud integration, ThingSpeak offers a more straightforward RESTful API for real-time data visualization, which is perfect for smaller projects, while AWS IoT Core offers scalability, advanced analytics, and security, making it suitable for large-scale industrial applications.

Now that we have designed the system, the next step is its implementation. The hardware selection plays a crucial role in IoT-based monitoring, and Raspberry Pi serves as an efficient gateway for processing and transmitting sensor data. It connects with multiple sensors using various interfaces such as GPIO, I2C, and UART, allowing real-time data collection.
For data transmission, we use MQTT, a lightweight protocol ideal for real-time communication. Unlike HTTP, which is better suited for periodic updates, MQTT enables low-bandwidth, real-time messaging using a broker system.
Finally, cloud integration is essential for data visualization and advanced analytics. While AWS IoT Core offers scalability and AI-driven insights, ThingSpeak provides a simpler API for smaller projects. This combination of hardware, protocols, and cloud services ensures an optimized IoT-based monitoring system.

3. Data Analysis and Visualisation: After the implementation, the obtained data is analyzed. At this stage, the Production Plant Data for Condition Monitoring dataset, obtained from the Kaggle platform, will be used.
The dataset contains various sensor readings collected from a production plant, including temperature, vibration, pressure, motor speed, load values, current, flow rate, and torque, which are crucial for monitoring machine health and predicting failures. Specifically, the parameters analyzed include Temperature (simulated), Vibration (L2), Pressure (C2), Motor Speed (B2), Load Values (L3), Current (B1), Flow Rate (C1), and Torque (B4). Initially, the dataset is preprocessed by handling missing values, converting timestamps into a usable format, and performing data cleaning to remove anomalies. Afterward, different visualization techniques, such as time series plots, histograms, and smoothed trends, are applied to observe patterns and detect irregularities in the sensor data. Additionally, advanced statistical methods, including anomaly detection using Z-score and forecasting using ARIMA models, are implemented to predict future sensor readings and potential failures. These insights are then integrated into an interactive dashboard using Streamlit, allowing for real-time monitoring and analysis of the production plant's condition.

Each monitored parameter plays a crucial role in understanding the production plant's condition. Below is an assessment of each sensor variable based on its trends, anomalies, and predictive insights.

- **Temperature (Simulated):**
  The temperature data displayed expected fluctuations within operational ranges. However, occasional outliers were observed, likely due to external environmental influences or short-term equipment malfunctions. The anomaly detection method successfully identified these spikes, preventing potential overheating issues.
- **Vibration (L2):**
  The vibration sensor data showed periodic increases, corresponding to operational loads and machine activity. Through Z-score-based anomaly detection, unexpected peaks were identified, indicating possible mechanical misalignments or wear-and-tear in rotating components.
- **Pressure (C2):**
  Pressure variations were generally stable, with minor fluctuations aligning with operational cycles. However, sudden drops in pressure were flagged as anomalies, suggesting potential system leaks or temporary reductions in flow rate. The smoothing technique helped highlight long-term trends without noise interference.
- **Motor Speed (B2):**
  The motor speed analysis confirmed consistent performance, though occasional irregularities were identified. These fluctuations were correlated with changes in load values, emphasizing the importance of optimizing speed based on real-time operational demands.
- **Load Values (L3):**
  The load parameter exhibited fluctuations that corresponded to varying production conditions. Smoothed visualization provided better clarity, helping predict stress points where machinery might require maintenance.
- **Current (B1):**
  Current levels followed expected patterns, but a few anomalies were detected where current consumption exceeded normal ranges. Such occurrences suggest inefficiencies in the electrical system or increased power demands during specific operational phases.
- **Flow Rate (C1):**
  Flow rate remained within an optimal range but presented a few inconsistencies over time. These deviations could indicate obstructions or inconsistencies in pipeline pressure, which warrant further investigation.

- **Torque (B4):**
  The torque analysis revealed significant variations corresponding to machine operations and applied loads. Forecasting techniques using the ARIMA model provided future torque predictions, helping anticipate maintenance requirements and minimize downtime.

4. System Performance Evaluation

To evaluate the effectiveness of the IoT-based monitoring system we implemented, several key performance criteria were considered.

Real-time Monitoring Efficiency:
The Streamlit dashboard played a crucial role in enabling real-time visualization of sensor data. This allowed us to continuously monitor manufacturing conditions and detect anomalies as they occurred, ensuring better operational control.

Anomaly Detection Accuracy:
By applying the Z-score method, we effectively identified outliers across different parameters. This helped us flag unusual patterns in the data, allowing for early detection of potential equipment failures and reducing the risk of unexpected breakdowns.

Predictive Maintenance Insights:
ARIMA models also incorporated for time-series forecasting, which provided valuable trends in sensor behavior. This predictive capability allowed us to anticipate potential failures in advance, supporting proactive maintenance strategies and minimizing unplanned downtime.

Data Cleaning & Smoothing Techniques:
To enhance data clarity and reliability, rolling window smoothing and resampling techniques were applied. These methods helped filter out noise from the dataset, improving trend analysis and making decision-making more precise and data-driven.

Operational Improvements:
From an operational perspective, the system demonstrated several key benefits. Transparency was significantly improved by providing clear visibility into sensor data, leading to a better understanding of manufacturing conditions. Downtime was reduced through anomaly detection and predictive insights, helping prevent sudden equipment failures. Additionally,

efficiency was optimized by monitoring energy consumption and balancing workload distribution across machines, ultimately enhancing overall productivity.

Through this performance evaluation, we can see that an IoT-based monitoring system provides significant value in industrial environments. It enables smarter maintenance strategies, minimizes risks, and ensures continuous improvement in manufacturing operations.

5. Conclusion

To conclude, the integration of IoT in manufacturing has revolutionized industrial operations by enabling real-time monitoring, automation, and predictive maintenance. this project successfully demonstrated how an IoT-based monitoring system can be implemented to enhance industrial efficiency, predictive maintenance, and data-driven decision-making.

By leveraging real-time sensor data, we were able to detect anomalies, analyze trends, and forecast future machine performance. The anomaly detection methods, particularly the Z-score approach, allowed us to identify outliers and prevent potential system failures. Additionally, using the ARIMA forecasting model, we predicted future fluctuations in key manufacturing parameters, enabling proactive maintenance strategies and reducing unexpected downtime.

Furthermore, the integration of a cloud-based dashboard using Streamlit provided a clear and interactive visualization of the data, ensuring seamless real-time monitoring of industrial processes. This not only enhanced transparency but also helped optimize energy consumption and operational efficiency.

Moving forward, there are several ways this system can be improved. Implementing advanced machine learning models, such as LSTM or Random Forest, could enhance predictive accuracy. Expanding sensor coverage to monitor additional factors like humidity or acoustic emissions would provide deeper insights into production conditions. Lastly, integrating an automated alert system could allow for immediate notifications when anomalies are detected, ensuring a more responsive manufacturing environment.

In summary, this project highlights how IoT and data analytics can significantly improve manufacturing operations by reducing downtime, optimizing efficiency, and enabling smarter decision-making. By adopting such IoT-based solutions, industries can enhance reliability, reduce costs, and move towards a fully data-driven production system.

REFERENCES:

IBM. (2025). *What is the IoT?* IBM. Retrieved January 10, 2025, from https://www.ibm.com/think/topics/internet-of-things

Oracle. (2025). *What is industrial IoT?* Retrieved January 10, 2025, from https://www.oracle.com/internet-of-things/

Opensource.com. (n.d.). *Raspberry Pi Resources*. Retrieved January 12, 2025, from https://opensource.com/resources/raspberry-pi

IEEE Transactions on Industrial Informatics. (2021). *Edge Computing for Industrial IoT*. IEEE.

MQTT.org. (n.d.). *MQTT: The Standard for IoT Messaging*. Retrieved January 12, 2025, from https://mqtt.org/

AWS. (n.d.). *What is AWS IoT?* Retrieved January 12, 2025, from https://docs.aws.amazon.com/iot/latest/developerguide/what-is-aws-iot.html