SIT314/SIT729 – Week 3 Group Activity  
The Impact of IoT on our lives

short line

# Overview

# The aim of this activity is to plan the communications and connectivity for a distinct application.

# Resource mining is a high-cost high-reward industry, which rapidly adopts new technologies to manage those costs. IoT has had a substantial impact on automating mining. Dassault Systems has produced the following image which summarises the use of IoT technologies in mining for their GEOVIA modelling software. For us, this provides a nice case study of the use of IoT in an environment that needs to be resilient and scalable.

![A diagram of a mining site

AI-generated content may be incorrect.]()

# Tasks

For the application `Vehicle Asset Health` identified in the image.

1. What functionality is needed for this application? What should be sensed, what processing is needed and what actuation is possible?

* **Functionality**: this application must be able to detect the current “health status” of a vehicle based on the following factors:
  + ***Vehicle parameters***:
    - Engine temperature.
    - Oil pressure.
    - Fuel levels.
    - Tire pressure.
    - Brake wear.
  + ***Operational data***:
    - Speed.
    - Runtime hours.
    - Load weight.
  + ***Driver behaviours***:
    - Acceleration patterns.
    - Braking frequency.
* **Processed**:
  + ***Data Analysis***: Real-time analysis to detect anomalies (e.g., engine overheating, abnormal vibrations).
  + ***Predictive Maintenance***: Algorithms to predict when maintenance is needed based on sensor data trends.
  + ***Diagnostics***: Fault code generation and prioritization (e.g., critical vs. non-critical issues).
  + ***Aggregation***: Combining data from multiple vehicles for fleet-wide insights.
  + ***Alerts***: Generating notifications for maintenance teams or operators when thresholds are exceeded.
* **Actuated**:
  + ***Alerts/Notifications***: Sending warnings to drivers (e.g., dashboard alerts) or maintenance crews (e.g., via mobile apps or control centers).
  + ***Automated Controls***: Shutting down a vehicle in critical failure scenarios or limiting speed if unsafe conditions are detected.
  + ***Maintenance Scheduling***: Automatically logging maintenance requests in a central system.
  + ***Reporting***: Generating reports for fleet managers on vehicle health status.

1. What web services would you need to structure to implement this?

* Data Collection Service:
  + Purpose: Collects sensor data (e.g., engine temperature, GPS) from vehicles.
  + Implementation: RESTful API (e.g., POST /vehicles/{vehicle\_id}/sensors) to receive data from IoT devices on vehicles.
  + Example: Vehicles send JSON payloads with sensor readings to a cloud endpoint.
* Data Processing Service:
  + Purpose: Processes raw sensor data for anomaly detection and predictive maintenance.
  + Implementation: A microservice that consumes sensor data, runs analytics (e.g., machine learning models), and stores results in a database.
  + Example: POST /analytics/process to trigger analysis and return diagnostic results.
* Notification Service:
  + Purpose: Sends alerts to drivers, maintenance teams, or fleet managers.
  + Implementation: REST API or WebSocket for real-time notifications (e.g., POST /notifications/alert to send emails, SMS, or app notifications).
  + Example: Alerts for critical issues like engine failure.
* Maintenance Scheduling Service:
  + Purpose: Logs and schedules maintenance tasks based on processed data.
  + Implementation: REST API (e.g., POST /maintenance/schedule) integrated with a maintenance management system.
  + Example: Creates a work order for a vehicle needing tire replacement.
* Dashboard Service:
  + Purpose: Provides a user interface for fleet managers to monitor vehicle health.
  + Implementation: GET /dashboard/vehicles to retrieve real-time and historical data for visualization.
  + Example: Displays vehicle status, alerts, and maintenance schedules.
* Authentication/Authorization Service:
  + Purpose: Secures access to services and data.
  + Implementation: OAuth 2.0-based API (e.g., POST /auth/token) to authenticate users and devices.
  + Example: Ensures only authorized personnel access sensitive vehicle data.
* Assumptions: Services are cloud-based, scalable, and use standard protocols like HTTPS for security.

1. What hierarchical MQTT topics would you use?

* For Vehicle Asset Health, topics should reflect the mining site, vehicle, and data type.
* Possible topic structure:

“mining/{site\_id}/{vehicle\_id}/{data\_type}/{sensor\_type}”

* Where:
  + site\_id: Unique identifier for the mining site (e.g., siteA).
  + vehicle\_id: Unique vehicle identifier (e.g., truck001).
  + data\_type: Category of data (e.g., sensors, alerts, commands).
  + sensor\_type: Specific sensor or action (e.g., engine\_temp, tire\_pressure, maintenance).
* **Example Topics**:
  + **Sensor Data**:
    - mining/siteA/truck001/sensors/engine\_temp: Publishes engine temperature readings.
    - mining/siteA/truck001/sensors/tire\_pressure: Publishes tire pressure data.
  + **Alerts**:
    - mining/siteA/truck001/alerts/critical: Publishes critical alerts (e.g., engine failure).
    - mining/siteA/truck001/alerts/warning: Publishes non-critical warnings (e.g., low fuel).
  + **Commands**:
    - mining/siteA/truck001/commands/shutdown: Sends a command to shut down the vehicle.
    - mining/siteA/truck001/commands/schedule\_maintenance: Triggers a maintenance request.
* Wildcards for Filtering:
  + **Single-level (+):** Subscribe to mining/siteA/+/sensors/engine\_temp to receive engine temperature data from all vehicles at siteA.
  + **Multi-level (#):** Subscribe to mining/siteA/truck001/# to receive all messages for truck001 (sensors, alerts, commands).
  + **Fleet-wide Monitoring:** Subscribe to mining/+/+/sensors/# to monitor sensor data across all sites and vehicles.

Based on this, continue with the following.

1. Create a high-level block diagram of the scenario, include the all the people, places and things that are involved in the scenario.

Decide what wireless communications technology to use.

1. Create a simple data flow diagram showing all the needed communication in this system.

A black screen with white text

AI-generated content may be incorrect.

Diagram 1 – Flow diagram for both question 4 & 5.

Design two approaches to this application:

1. Plan a web service-based approach to implementing the application-level communications for this application. What services would you need?

 **Services Needed**

* **Sensor Data Ingestion**:
  + **Endpoint**: POST /vehicles/{vehicle\_id}/sensors
  + **Function**: Receives JSON payloads with sensor data (e.g., { "engine\_temp": 85, "tire\_pressure": 30 }).
  + **Security**: Authenticated via OAuth 2.0 device tokens.
* **Analytics Service**:
  + **Endpoint**: POST /analytics/process/{vehicle\_id}
  + **Function**: Runs predictive maintenance algorithms and generates diagnostics.
  + **Output**: Stores results in a database and triggers alerts if needed.
* **Alert Service**:
  + **Endpoint**: POST /notifications/alert
  + **Function**: Sends notifications via email, SMS, or push notifications to users.
  + **Example**: { "vehicle\_id": "truck001", "message": "Engine overheating", "priority": "critical" }.
* **Maintenance Service**:
  + **Endpoint**: POST /maintenance/schedule
  + **Function**: Creates work orders in a maintenance system.
  + **Integration**: Connects to existing enterprise maintenance software.
* **Dashboard Service**:
  + **Endpoint**: GET /dashboard/vehicles/{site\_id}
  + **Function**: Returns vehicle health data for visualization (e.g., JSON with status, alerts, and maintenance history).
* **Command Service**:
  + **Endpoint**: POST /vehicles/{vehicle\_id}/commands
  + **Function**: Sends commands to vehicles (e.g., shutdown, speed limit).
  + **Security**: Restricted to authorized users (e.g., fleet managers).

 **Architecture**:

* **Microservices**: Each service is independently deployable, using a cloud platform like AWS or Azure.
* **Database**: A NoSQL database (e.g., MongoDB) for storing sensor data and analytics results.
* **API Gateway**: Manages routing, authentication, and rate limiting for all services.
* **Scalability**: Services scale horizontally to handle large fleets.
* **Security**: HTTPS, OAuth 2.0, and role-based access control (RBAC) for users and devices.

1. Plan a hierarchical MQTT application-level communications protocol. Think about the identity of users and devices in the system and plan single-level or multi-level wildcards for message filtering.

 **Identity Management**:

* **Devices**:
  + Each vehicle has a unique ID (e.g., truck001) embedded in MQTT topics.
  + IoT Gateways have IDs (e.g., gatewayA) for aggregating data from multiple vehicles.
  + Devices authenticate with the MQTT broker using client certificates or username/password pairs.
* **Users**:
  + **Drivers**: Subscribe to vehicle-specific alerts (e.g., mining/siteA/truck001/alerts/#).
  + **Maintenance Crews**: Subscribe to site-wide alerts (e.g., mining/siteA/+/alerts/#).
  + **Fleet Managers**: Subscribe to fleet-wide data (e.g., mining/+/+/sensors/#).
  + User authentication is handled via a separate authentication service (e.g., OAuth 2.0 tokens mapped to MQTT subscriptions).

 **Topic Hierarchy**:

“mining/{site\_id}/{vehicle\_id}/{data\_type}/{sensor\_type\_or\_action}”

* **Examples**:
  + Sensor data: mining/siteA/truck001/sensors/engine\_temp
  + Alerts: mining/siteA/truck001/alerts/critical
  + Commands: mining/siteA/truck001/commands/shutdown
* **Gateway Topics**:
  + mining/siteA/gatewayA/sensors/aggregated: Publishes aggregated data from all vehicles at siteA.
* **Control Center Topics**:
  + mining/siteA/control\_center/commands: Publishes commands to all vehicles at siteA.

 **Wildcard Usage**:

* **Single-level (+)**:
  + Maintenance crews subscribe to mining/siteA/+/alerts/critical to receive critical alerts from all vehicles at siteA.
  + Drivers subscribe to mining/siteA/truck001/+/critical to receive critical alerts for their vehicle.
* **Multi-level (#)**:
  + Fleet managers subscribe to mining/+/+/sensors/# to monitor all sensor data across all sites.
  + Control centers subscribe to mining/siteA/# for all messages from siteA.
* **Security Considerations**:
  + Use Access Control Lists (ACLs) in the MQTT broker to restrict subscriptions (e.g., drivers can only subscribe to their own vehicle’s topics).
  + Encrypt payloads with TLS to protect sensitive data.

 **Message Flow**:

* **Publish**: Vehicles publish sensor data to mining/siteA/truck001/sensors/\*.
* **Subscribe**: Maintenance crews subscribe to mining/siteA/+/alerts/# for real-time alerts.
* **Command**: Control centers publish to mining/siteA/truck001/commands/\* to send instructions.
* **Aggregation**: Gateways publish aggregated data to mining/siteA/gatewayA/sensors/aggregated.