Internet of Everything DJS22CEC702

By Prof. Nilesh Ghavate

Module 2: IoE Devices

- Introduction IoT Devices
- IoT Software
- IoT Device Management Platform and Software,
- IoT System Architecture,
- IoT connectivity technologies,
- Physical design of IoT,
- Logical design of IoT- IoT Communication model
- IoT Levels and Deployment Templates.

IoT Devices

- IoT devices have sensors that can communicate data over the internet between objects or people.
- Examples include wireless sensors, software, actuators, and computer devices.
- They enable automatic data transmission without human interaction.
- **Example**: A car's IoT system detects traffic and sends a delay notification to a person you are meeting.
- **Example**: A pacemaker can connect to other equipment for improved living convenience.
- IoT devices are remotely managed and operated. They can use technologies like responsive sensors, low-energy wireless, NFC, LTE, Bluetooth, and other wireless protocols.

https://iotdunia.com/top-iot-devices/

Advantages of IoT Devices

- IoT devices enable machine-to-machine interaction, allowing smooth communication between equipment.
- Make daily tasks faster and easier without human interaction (e.g., auto-adjusting room temperature).
- Reduce labor costs and execution time through automation.
- Enable remote access and control of internet-connected devices.
- Provide real-time data for failure detection and corrective action.

Drawbacks of IoT Devices

- Risk of security breaches.
- Loss of manual jobs due to automation.
- Possibility of complex system failures.
- Lack of international standards complicates device construction.

Top IoT Devices

- Google Home Voice Controller:
- Smart IoT gadget to control TV, speakers, alarms, lighting, and other home equipment via voice commands.
- Key features:
    - Make phone calls.
    - Send broadcast messages.
    - Get Google’s real-time responses.
    - Listen to music using built-in speakers.
    - Manage household appliances.
    - Receive weather updates

Top IoT Devices

- **Amazon Echo Voice Controller**

- Its sound cancellation technology aids in differentiating your voice even during music playback.
- It helps in playing the music, initiating phone calls and messaging, setting of alarms, and handling a variety of other tasks.

Top IoT Devices

- Foobot Air Quality Monitor
  - Analyzes pollution levels in homes or businesses.
  - Improves air quality by cleaning the air for healthier breathing.
  - Monitors temperature levels.

Top IoT Devices

- August Smart Lock
  - Enhances home security with remote visitor tracking.
  - Notifies users if doors are secured or not.
  - Key features:
    - Add new visitors with restricted entry.
    - Auto-unlock feature for door opening upon arrival.
    - Create and receive alerts for specific movements or actions.

IoT Software

- **IoT Software** includes platforms, embedded systems, partner systems, and middleware for data gathering, device integration, and real-time analytics, often integrated with business systems.
- **Data Collection:** Manages sensing, measurement, light data protection, and aggregation; uses specialized protocols for M2M communication; collects and transfers data to a central server.
- **Device Integration:** Connects all devices in the IoT network; manages apps, protocols, and restrictions for seamless connectivity and collaboration.
- **Real-Time Analytics:** Translates device data into actionable insights or patterns; enables automation and supports industry decision-making.
- **Application & Process Extension:** Enhances existing systems; integrates specific devices for mobile or technical access, boosting efficiency and accuracy.

IoT Platforms - GCP IoT Reference Architecture

- Overview:

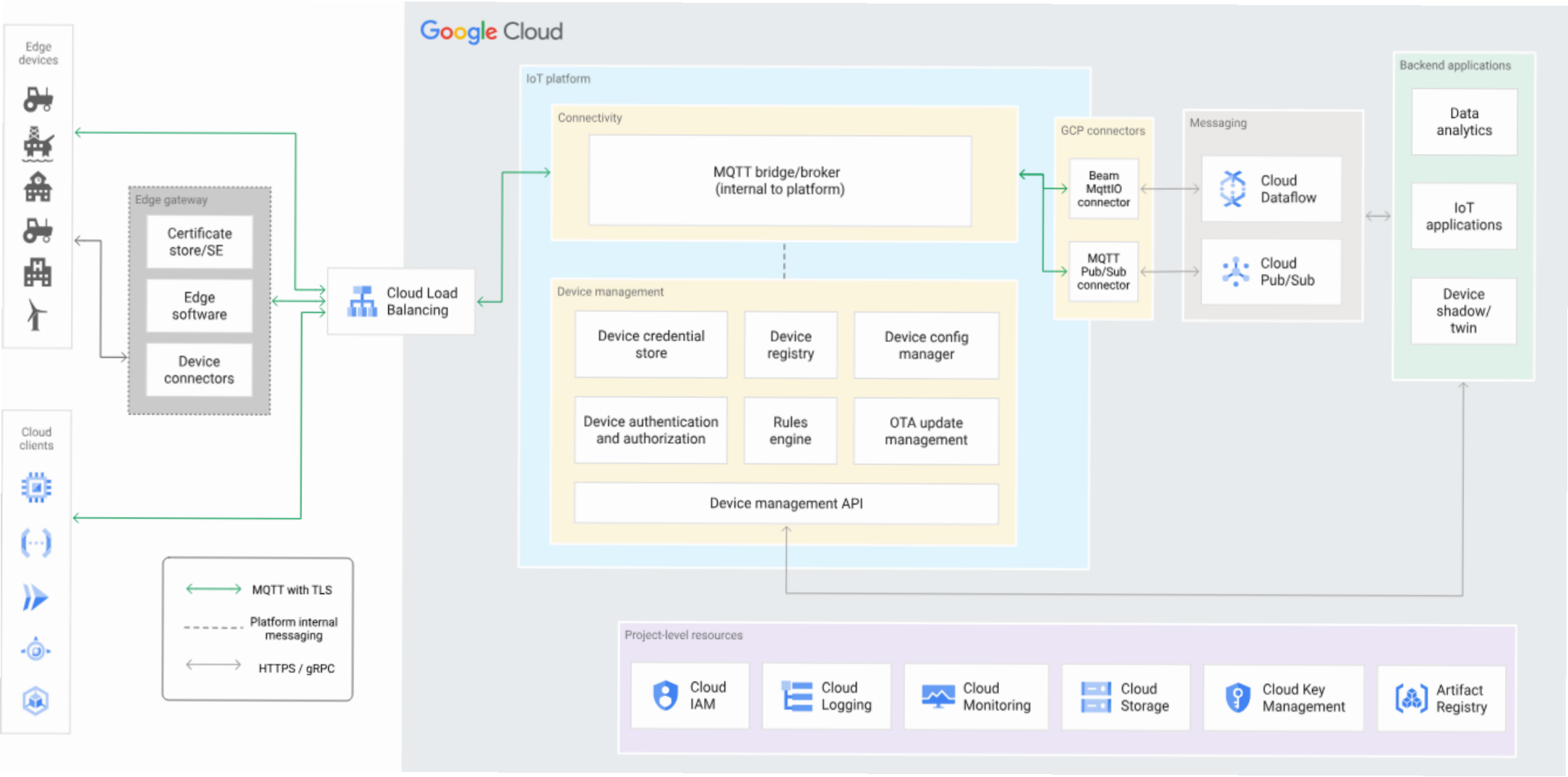
A fully managed set of services by Google for securely connecting, managing, and ingesting data from IoT devices globally.

- Key Features:
  - Cloud IoT Core for device connection & management
  - Integration with BigQuery, Cloud Pub/Sub, and AI tools for analytics
  - Secure communication via MQTT and HTTP

- Example Use Case:

Smart agriculture systems send soil moisture data to Google Cloud for AI-based irrigation recommendations.

IoT Platforms - GCP IoT Reference Architecture



MQTT is a lightweight, open-source messaging protocol

Google Cloud Pub/Sub is a real-time messaging service

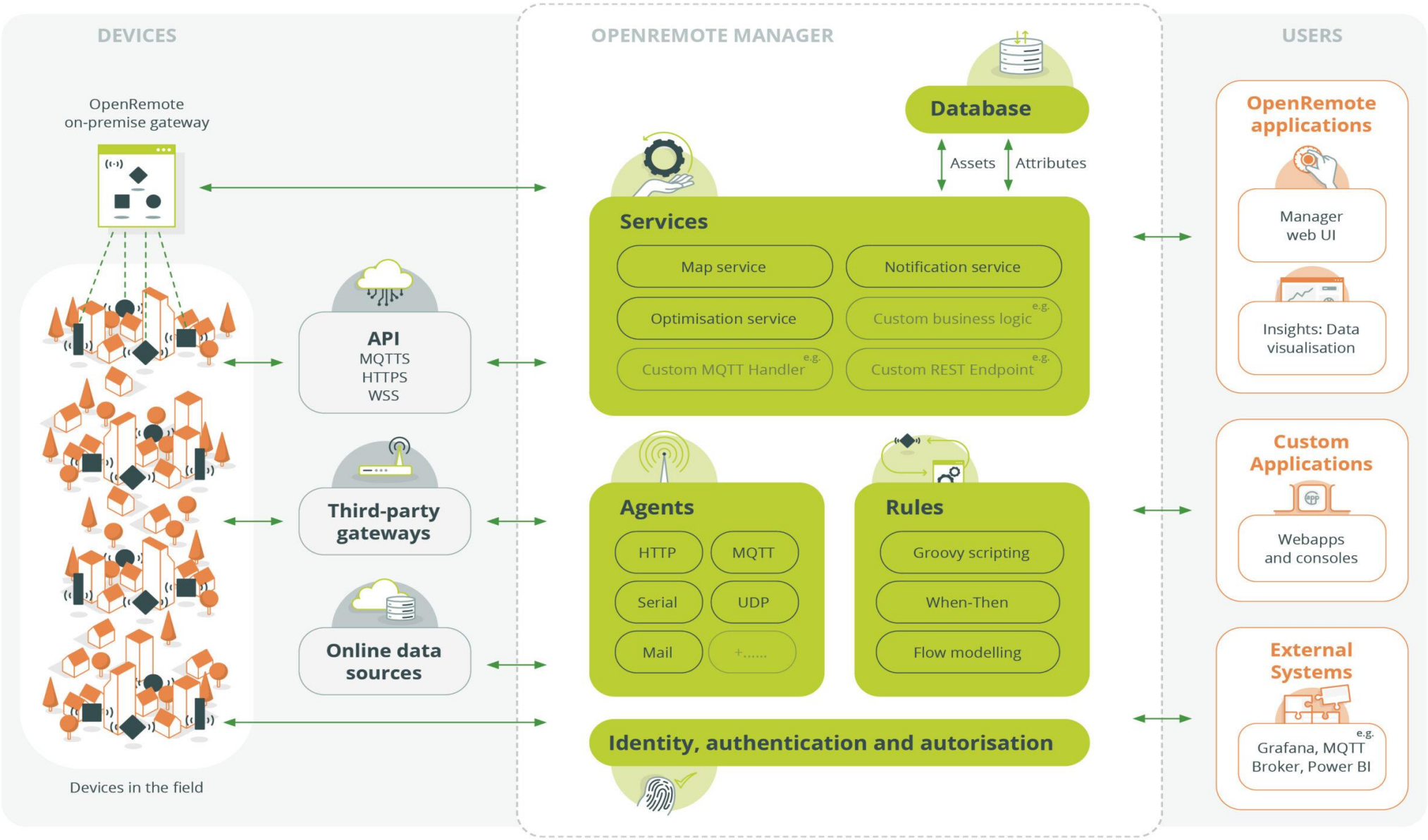
IoT Platforms - GCP IoT Reference Architecture

Component	Purpose & Features
Device Connectivity	MQTT/HTTPS ingestion with load balancing and protocol flexibility
Device Management	Secure registry with auth, config, OTA updates, and digital twin capabilities
Data Pipeline	Real-time and batch data processing via Pub/Sub → Dataflow/Functions/Dataproc
Storage & Analytics	BigQuery/Storage for analytics; ML integration for predictive insights
Rules & Extensions	Event-based triggers, alerts, low-code tools, and integration with business systems

IoT Platforms – Open Remote

- **Overview:**
  An open-source IoT platform designed for smart city, energy management, and building automation projects.
- **Key Features:**
  - Device integration through multiple protocols (MQTT, HTTP, LoRa, Modbus)
  - Rule engine for automation
  - Dashboard for visualization and control
- **Example Use Case:**
  City administrators integrate street lighting, parking sensors, and environmental monitors in a single OpenRemote dashboard.

IoT Platform - Open Remote



WSS secure version of the WebSocket

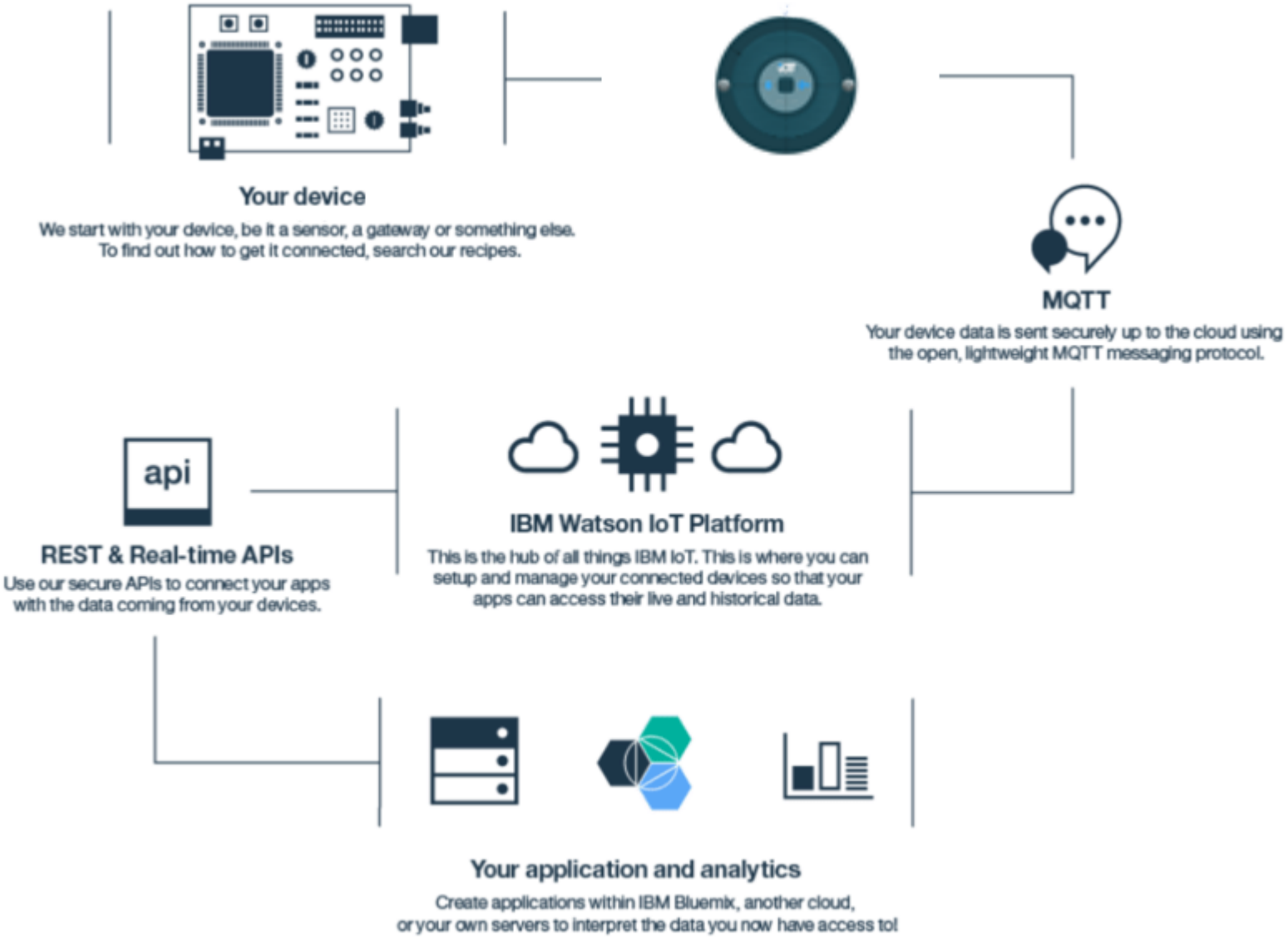
IoT Platforms – Open Remote

Feature Area	Highlights
Asset & Device Modeling	Custom schemas, multi-tenant, dashboards
Protocol Support	Wide-range (HTTP, MQTT, KNX, etc.)
Automation	Drag-and-drop (When-Then, Flow) + scripting
Security	Keycloak, SSL, OAuth, RBAC
Deployment Flexibility	Docker, edge gateways, OTA support
Extensibility	ML, custom UIs, firmware, third-party tools
Developer Ecosystem	AGPL-licensed, documentation, GitHub, community
Use Cases	Cities, energy, industrial monitoring, smart buildings

IoT Platforms – IBM Watson

- **Overview:**
  A cloud-hosted service from IBM designed to connect IoT devices and apply AI-driven analytics.
- **Key Features:**
  - Device registration, connectivity, and lifecycle management
  - AI-powered data analysis via Watson AI
  - Integration with blockchain for secure IoT transactions
- **Example Use Case:**
  Fleet management companies use IBM Watson IoT to track vehicles, predict maintenance, and optimize fuel usage.

IoT Platforms – IBM Watson



IoT Platforms - IBM Watson

Component	Description
Device Connectivity	Secure MQTT/HTTP access; client SDKs & recipes for onboarding
Device Management	Lifecycle control, dashboards, provisioning, and firmware updates
Security	TLS, certs, access control, integration with security partners
Data APIs	MQTT pub-sub + REST/APIs for real-time and historical data access
Analytics & AI	Integration with Watson AI services for predictive insights
Scalability	Cloud-native platform built for growth and extensibility

Why Use Watson IoT Platform?

- Speed of Development: IBM claims up to 40% faster app development leveraging Bluemix and IoT services.
- Strong Security Emphasis: Secure from chip to cloud.
- Seamless IoT → AI Flow: Turn device data into actionable insights with IBM's AI ecosystem.
- Cloud-Native & Scalable: Grows with your deployment needs.

IoT Device Management

- The process of provisioning, authenticating, configuring, managing, monitoring, and maintaining connected devices at scale is known as IoT device management.

- These techniques allow for the administrative monitoring and controlling of the IoT ecosystems by alerting them, who can then remotely resolve problems and carry out system-wide tasks like firmware updates.

Working of IoT Device Managment

- **Provisioning:** It is device registration - the process of adding a device to the system.
- **Authentication:** It establishes the identification of the device and assures its security and trustworthiness in the system.
- **Configuration and control:** It creates firmware properties and security settings. It enables control of devices remotely during and post deployment phase, as well as the ability to change a device's functionality and firmware.
- **Monitoring and diagnostics**: It provides information on the health of the device and sends out alerts for abnormalities and other network problems, enabling administrators to fix the problem promptly and remotely.
- **Maintenance:** It refreshes connected devices and adjusts settings to make the device healthier and enables security in networking capabilities. (OTA)

Working of IoT Device Managment

1. Provisioning

- What it is: The process of onboarding a new IoT device into the network or platform.
- How it works:
  - Assign a unique identity (ID, keys, certificates)
  - Connect it to the correct IoT platform or cloud service
  - Apply initial configurations and firmware

Example: When a smart sensor is shipped to a customer, it is provisioned with network credentials and cloud settings before it starts working.

Working of IoT Device Managment

2. Authentication

• What it is: Verifying that the device is genuine and authorized to connect to the IoT platform.

• How it works:

◦ Uses secure credentials like digital certificates, tokens, or cryptographic keys

◦ Prevents unauthorized devices from accessing the network

• Example: A connected security camera must authenticate with the cloud before transmitting video to ensure it is not a malicious clone.

Working of IoT Device Managment

3. Configuration and Control

- What it is: Setting up and managing the device’s operational parameters.
- How it works:
  - Remote configuration via IoT dashboards or APIs
  - Adjusting settings such as reporting frequency, data thresholds, or operational modes
  - Sending control commands (e.g., restart, change mode)
- Example: Changing the data reporting interval of a smart water meter from once per hour to once every 10 minutes.

Working of IoT Device Managment

4. Monitoring and Diagnostics

- What it is: Continuously tracking the device’s health, status, and performance.
- How it works:
  - Collects metrics like battery level, temperature, connectivity status, and error logs
  - Alerts when devices go offline or show abnormal behavior
- Example: In a factory, IoT temperature sensors send health reports to detect failures before they impact production.

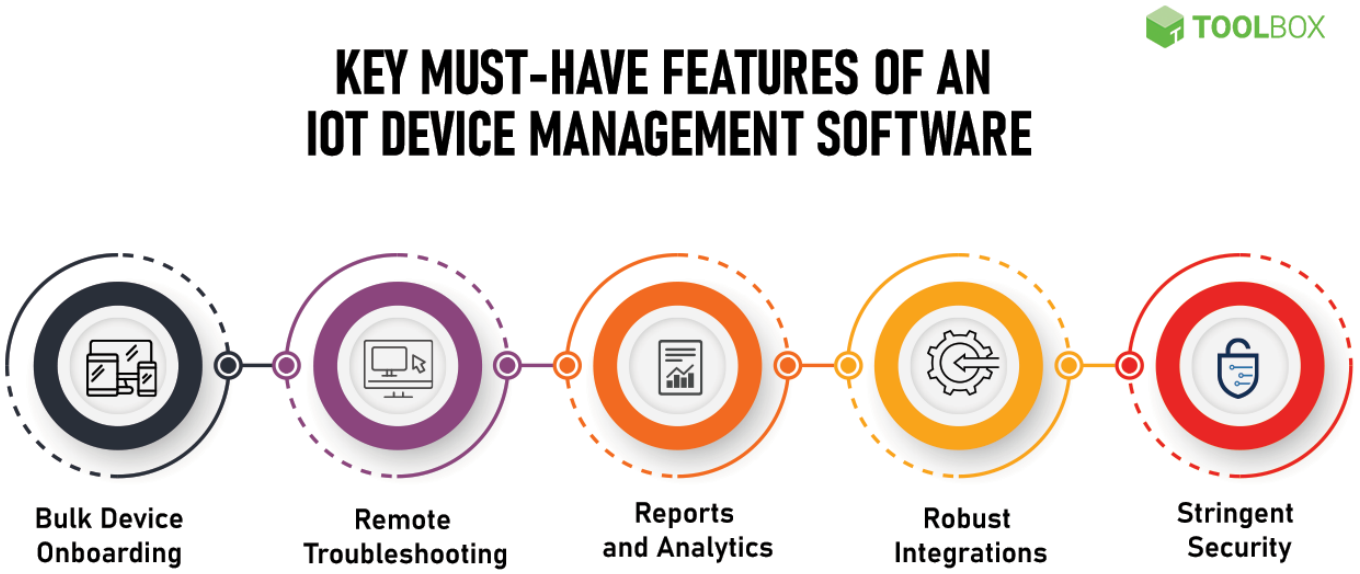
Working of IoT Device Managment

5. Maintenance

- What it is: Keeping the device updated and operational over time.
- How it works:
  - Firmware updates (OTA – Over The Air) to fix bugs or add features
  - Patch vulnerabilities to prevent cyberattacks
  - Replace or recalibrate sensors when needed
- Example: Pushing a security patch to thousands of smart streetlights without physically visiting each one.

IoT Device Management Software

- AWS IoT Core
- Cloud IoT Core
- Azure IoT Hub
- Oracle IoT Cloud
- IBM Watson



Features of AWS IoT Device Management

AWS IoT Core

- Device Onboarding: Bulk registration of devices using templates and provisioning.
- Organization & Grouping: Categorize devices by type, location, or function.
- Secure Communication: Mutual authentication and encryption using AWS IoT Core's security features.
- Remote Management: Update configurations, firmware, and software over the air (OTA).
- Fleet Indexing & Search: Quickly search and filter devices based on attributes or status.
- Jobs & Automation: Schedule and manage remote tasks for single or multiple devices.
- Integration with AWS Services: Works seamlessly with AWS IoT Core, Lambda, S3, DynamoDB, and analytics tools.
- Lifecycle Management Tools: End-to-end support from onboarding to retirement; integration with IoT Core, Fleet Registry, and Jobs.

Pros of AWS IoT Device Management

- Scalability & Reliability – Handles millions of devices and trillions of messages without user-managed infrastructure.
- Strong Security – Uses X.509 certificates, IAM policies, and TLS encryption for secure, fine-grained access control.
- Seamless AWS Ecosystem Integration – Natively works with CloudWatch, Lambda, IoT Analytics, and other AWS services for serverless automation.
- Device Lifecycle Management – Supports provisioning, grouping, OTA updates, and job scheduling.
- Fleet Indexing & Search – Quickly locate devices by attributes or health status.
- Remote Management – Update firmware/configurations across single or multiple devices.

Cons of AWS IoT Device Management

- Can be costly at large scale due to AWS pricing model.
- Steeper learning curve for beginners.
- Internet dependency—offline management capabilities are limited.
- Complex for small/simple IoT deployments.
- Vendor Lock-In Risk

AWS IoT Device Management

You are already using **AWS ecosystem** (Lambda, S3, DynamoDB, SageMaker).
You need **scalable, flexible IoT deployment** (millions of devices).
You need **tight integration with AI/ML, analytics, or serverless apps**.

Strengths: Strong scalability, security (fine-grained IAM), edge computing (Greengrass).

Limitations: Complex pricing, can be overwhelming for small projects.

Features of Google Cloud IoT Core

- Device Manager – Registers, authenticates (IAM), and manages IoT devices with per-device credentials and configurations.
- Protocol Support – Secure communication via MQTT and HTTP.
- Cloud Pub/Sub Integration – Streams device telemetry data to Cloud Pub/Sub for further processing.
- Device State Management – Stores and retrieves the last reported state of a device.
- Secure Authentication – Uses public key-based authentication (RS256 or ES256) with JSON Web Tokens (JWT).
- Scalability – Designed to handle millions of devices with high throughput.
- Seamless GCP Integration – Works with BigQuery, Dataflow, AI Platform, and Cloud Functions for analytics and automation.

Pros - Google Cloud IoT Core

- Fully managed, scalable, and reliable IoT data ingestion platform.
- Strong integration with GCP’s big data and AI/ML tools.
- Secure authentication using industry-standard public key cryptography.
- Supports both batch and streaming analytics workflows.
- Simplifies ingestion of telemetry and event data.

Cons - Google Cloud IoT Core

- Limited to MQTT and HTTP—no native AMQP or WebSocket support.
- Lacks built-in edge computing capabilities (requires extra services like Edge TPU or Anthos).
- Service is deprecated as of August 2023, requiring migration to other solutions.
- Tight coupling with GCP ecosystem—less portable to other clouds.

Google Cloud IoT Core (deprecated-Aug 2023)

Best when:

You want **real-time analytics & AI/ML** via **BigQuery, Dataflow, Vertex AI**.

You are building **data-heavy IoT apps** (predictive maintenance, anomaly detection).

Strengths: Analytics + AI integration, Pub/Sub messaging backbone.

Limitations: IoT Core itself retired → must use partner solutions on Google Cloud.

Features of Azure IoT Hub

- Bi-directional Communication – Secure messaging between cloud and IoT devices using protocols like MQTT, AMQP, and HTTPS.
- Device Twin & Module Twin – Store and sync device state and metadata for configuration and monitoring.
- Mass Device Management – Supports bulk provisioning, updates, and configuration changes across fleets.
- Event Routing & Integration – Connects with Azure Stream Analytics, Functions, Logic Apps, and Data Lake for real-time processing.
- Security – Per-device authentication with symmetric keys, X.509 certificates, and Azure Active Directory integration.
- Edge Computing Support – Integrates with Azure IoT Edge for local processing.
- Monitoring & Diagnostics – Built-in metrics, logging, and health monitoring.

Pros - Azure IoT Hub

- Strong support for hybrid and edge IoT scenarios.
- Rich device state management via twins for synchronization.
- Deep integration with the broader Azure ecosystem for analytics, AI, and storage.
- Flexible protocol support for diverse device connectivity.
- Scales from small prototypes to large enterprise deployments.

Cons - Azure IoT Hub

- Can be complex to set up for beginners.
- Cost structure may be difficult to predict at scale.
- Azure-specific tooling—less portable to non-Microsoft cloud environments.
- More learning curve if not already invested in Azure services.

Azure IoT Hub

You are in a **Microsoft ecosystem** (Power BI, Azure ML, Dynamics 365, Office 365).
You need **enterprise IoT + device twins** (digital representation of devices).
You want **industrial IoT (IIoT)** support (factories, utilities).
Strengths: Bi-directional communication (device ↔ cloud), strong enterprise integration, device twins, edge support.
Limitations: Slightly more enterprise-focused, can be overkill for simple IoT.

Features of Oracle IOT Cloud Service

- Device Connectivity & Management → Secure two-way communication, unique IDs for devices.
- Real-Time Analytics → Process, filter, and analyze data streams instantly.
- Enterprise Integration → Easy connection to enterprise apps via REST APIs.
- Edge & Hybrid Support → Local data processing, digital twins for asset simulation.
- Scalability → Works with Oracle Cloud tools and grows with business needs.

Pros - Oracle IOT Cloud Service

- Oracle Ecosystem Fit → Best for companies already using Oracle products.
- Real-Time Insights → Enables monitoring, predictive maintenance, and faster actions.
- Strong Security → End-to-end encryption, identity management, access control.
- Edge & Digital Twin → Process data locally and simulate assets.
- Scalable → Handles large workloads, enterprise-grade performance.

Cons - Oracle IOT Cloud Service

- Complex Setup → Hard to configure, steep learning curve.
- High Cost → Expensive licensing and pricing is complex.
- Limited Flexibility → Harder to customize with non-Oracle tools.
- Support Issues → Weak documentation and slow customer support.
- Vendor Lock-In → Difficult to migrate away from Oracle.
- Free Tier Problems → Trial accounts may face abrupt issues.

Oracle Cloud IoT Service

Best when:

You are already invested in **Oracle ecosystem** (ERP, SCM, HCM, Oracle Analytics).

You want IoT data feeding **business processes** (supply chain, logistics, manufacturing).

Strengths: Business IoT integration (ERP/SCM), real-time monitoring dashboards, predictive analytics.

Limitations: Smaller developer community, not as popular as AWS/Azure.

IoT Connectivity Technologies

IoT devices rely on various connectivity technologies to communicate data. Choosing the right technology depends on factors like coverage area, battery consumption, bandwidth, cost, and environmental conditions.

IoT Connectivity Technologies

1. Area Coverage

- Definition: The geographic range that a connectivity technology can reliably cover.
- Examples:
  - Short-range (up to 100 meters): Bluetooth, Zigbee, Wi-Fi
  - Medium-range (hundreds of meters to a few km): LoRaWAN, Sigfox, NB-IoT
  - Long-range (tens of km): Cellular (3G, 4G, 5G), Satellite
- Use Case Impact:
  - Smart home sensors typically use short-range.
  - Agriculture or smart city sensors often need long-range technologies.

IoT Connectivity Technologies

2. Battery Usage

• Definition: The power consumption characteristics of the connectivity method, which impacts device battery life.

• Examples:

◦ Low-power: Zigbee, LoRaWAN, NB-IoT (designed for devices running years on small batteries)

◦ High-power: Wi-Fi, Cellular (require frequent charging or constant power supply)

• Use Case Impact:

◦ Battery-powered IoT devices (wearables, remote sensors) need low-power connectivity.

◦ Devices with continuous power can use higher bandwidth tech like Wi-Fi.

IoT Connectivity Technologies

3. Bandwidth

• Definition: The data rate supported by the connectivity technology (how much data can be sent/received per second).

• Examples:

◦ Low bandwidth: LoRaWAN (0.3–50 kbps), Sigfox (~100 bps)

◦ Medium bandwidth: NB-IoT (up to ~250 kbps), Bluetooth LE (up to 2 Mbps)

◦ High bandwidth: Wi-Fi (up to Gbps), 4G/5G Cellular (up to Gbps)

• Use Case Impact:

◦ Simple sensor data requires low bandwidth.

◦ Video streaming or complex data analysis needs high bandwidth.

IoT Connectivity Technologies

4. Cost

• Definition: The cost of deploying and maintaining the connectivity technology, including hardware, network fees, and infrastructure.

• Examples:

◦ Low cost: LoRaWAN and Sigfox (often community or subscription-based low fees)

◦ Medium cost: Wi-Fi (hardware is cheap but limited range)

◦ High cost: Cellular (SIM cards, data plans, infrastructure)

• Use Case Impact:

◦ Large-scale deployments with budget constraints often choose low-cost LPWAN technologies.

◦ Critical applications needing guaranteed quality may invest in cellular.

IoT Connectivity Technologies

5. Conditions

• Definition: Environmental and operational factors influencing connectivity choice, such as interference, physical obstacles, and mobility.

• Examples:

◦ Indoor vs outdoor: Wi-Fi and Bluetooth better indoors; LoRaWAN and cellular better outdoors

◦ Urban vs rural: Cellular networks better in urban; LPWAN better in remote areas

◦ Mobility: Cellular preferred for moving devices (vehicles, wearables)

• Use Case Impact:

◦ Factory automation may use robust wired or industrial wireless protocols.

◦ Wildlife tracking requires low-power, long-range, and rugged connectivity.

Short Range IoT Connectivity

1. **Wi-Fi**
   - Range: ~50-100 meters indoors
   - Bandwidth: High (up to Gbps)
   - Power Consumption: High (not ideal for battery-powered devices)
   - Use Cases:
     - Smart home devices (cameras, speakers, appliances)
     - Office IoT sensors and devices
   - Pros: High data rates, widely available infrastructure
   - Cons: Power hungry, limited range, interference in crowded networks
2. **Bluetooth**
   - Range: ~10 meters (Bluetooth Classic), up to 100 meters (Bluetooth Low Energy - BLE)
   - Bandwidth: Moderate (~1-3 Mbps)
   - Power Consumption: Low (especially BLE)
   - Use Cases:
     - Wearables, health monitors
     - Proximity sensors, beacons
   - Pros: Low power, easy to pair devices
   - Cons: Limited range, limited device scalability

Short Range IoT Connectivity

3. Mesh Network (e.g., Zigbee, Thread)

- Range: Each node covers ~10-100 meters; mesh extends overall range by relaying data
- Bandwidth: Low to moderate (~250 kbps for Zigbee)
- Power Consumption: Very low
- Use Cases:
  - Smart lighting and home automation
  - Industrial sensor networks
- Pros: Self-healing, scalable, low power
- Cons: Lower bandwidth, complexity in setup

4. Ethernet

- Range: Typically up to 100 meters via cables
- Bandwidth: Very high (up to 10 Gbps and beyond)
- Power Consumption: Powered via cable (PoE available)
- Use Cases:
  - Fixed IoT devices needing stable, high-speed connection
  - Industrial IoT equipment
- Pros: Reliable, secure, high speed
- Cons: Requires physical cabling, not mobile

High Range IoT Connectivity

These technologies enable devices to communicate over long distances, often covering several kilometers or more.

1. LPWAN (Low Power Wide Area Network)

- Range: Several kilometers up to 15+ km (varies by technology and environment)
- Bandwidth: Low (typically <50 kbps)
- Power Consumption: Very low (devices can run years on battery)
- Examples: LoRaWAN, Sigfox, NB-IoT, LTE-M
- Use Cases:
  - Smart agriculture and environmental monitoring
  - Asset tracking and smart metering
  - City-wide IoT deployments (smart parking, lighting)
- Pros: Long range, low power, low cost
- Cons: Low data throughput, limited to small data packets

High Range IoT Connectivity

2) **LoRaWAN (Long Range Wide Area Network)**

- **Range**: 2–15 km (urban/rural).
- **Data rate**: 0.3–50 kbps.
- **Power**: Very low → 5–10 years battery life.
- **Best for**: Agriculture, smart cities, environmental monitoring.

3) NB-IoT (Narrowband IoT)

•Range: 1–10 km.

•Data rate: Up to 250 kbps.

•Power: Low → 5–10 years battery life.

•Best for: Smart meters, healthcare, industrial IoT, smart buildings.

Physical Design of IoT

IoT (Internet of Things) devices have **unique identities** and the ability to perform **remote sensing**, **actuation**, and **monitoring**.

These devices — referred to as **"things"** — can exchange information and communicate with one another using **wired** or **wireless** connections.

Key Elements of Physical Design

Sensor I/O Interfaces

- Used by IoT devices to connect with other devices and gather environmental data.
- Sensors measure parameters like temperature, humidity, motion, light, etc.
- Actuators perform actions like turning a motor, opening a valve, or adjusting brightness.

Connectivity Interfaces for the Internet

- Provide communication between IoT devices and the internet or other networks.
- Can include Ethernet ports, Wi-Fi modules, cellular modems, Zigbee, Bluetooth.

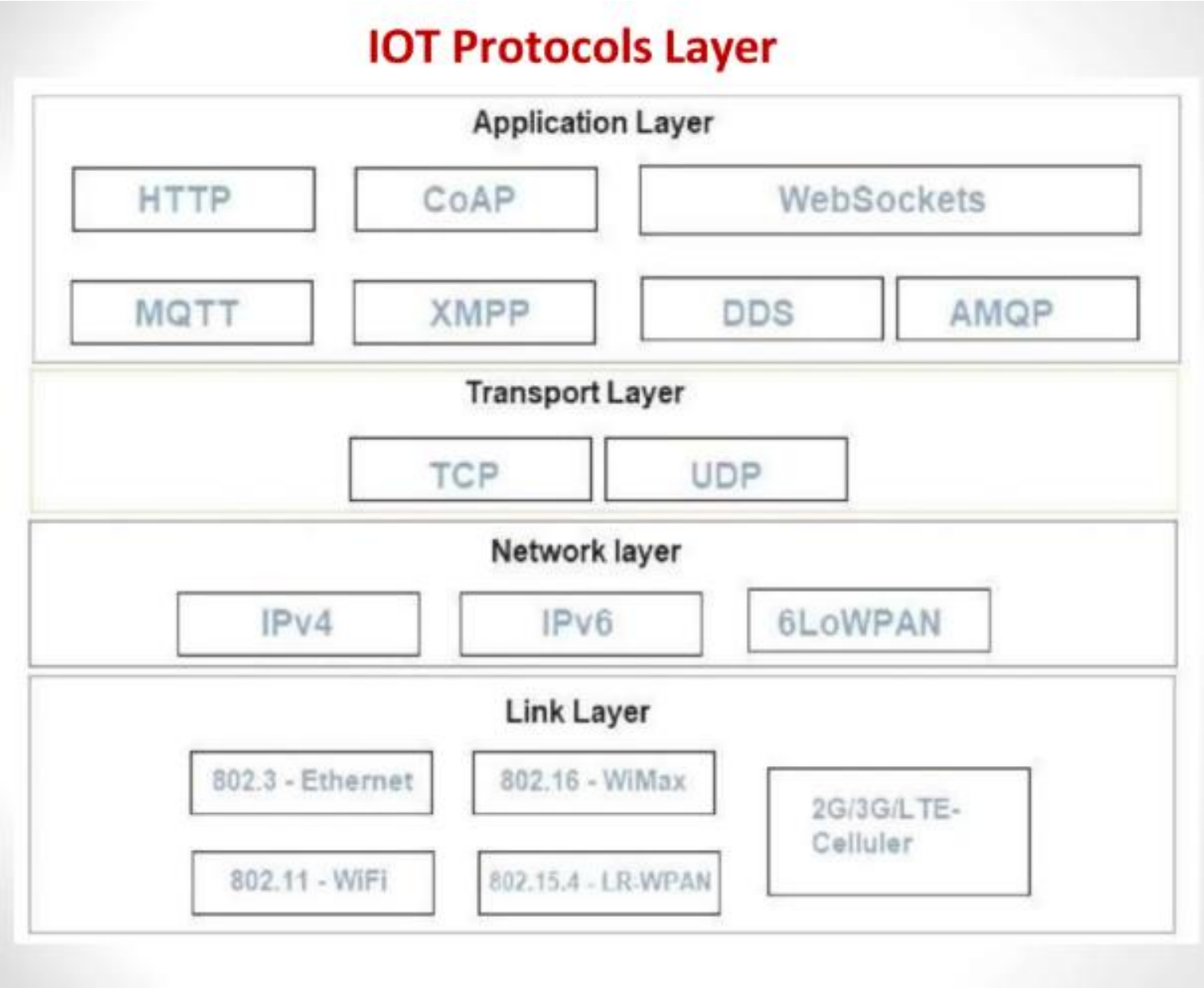
Data Storage and Memory

- Temporary or permanent storage for collected data before processing or transmission.
- Can be local (e.g., SD card, onboard flash memory) or cloud-based.

Interfaces for Audio and Video

- For multimedia IoT devices such as smart cameras, voice assistants, or video doorbells.
- Includes microphones, speakers, and camera modules.

IOT Protocols Layer



IoT Protocols:

1) Link Layer :

❖Protocols determine how data is physically sent over the network's physical layer or medium.

❖Local network connect to which host is attached. Hosts on the same link exchange data packets over the link layer using link layer protocols.

❖Link layer determines how packets are coded and signaled by the h/w device over the medium to which the host is attached.

- **802.3-Ethernet**:
- **802.11-WiFi:**
- **802.16** - WiMax:
- **802.15.4-LR-WPAN:**
- **2G/3G/4G-** Mobile Communication:

Protocols:

(i)802.3-Ethernet: IEEE802.3 is collection of wired Ethernet standards for the link layer. Eg: 802.3 uses co-axial cable; 802.3i uses copper twisted pair connection; 802.3j uses fiber optic connection; 802.3ae uses Ethernet overfiber.

(ii)802.11-WiFi: IEEE802.11 is a collection of wireless LAN(WLAN) communication standards including extensive description of link layer. Eg: 802.11a operates in 5GHz band, 802.11b and 802.11g operates in 2.4GHz band, 802.11n operates in 2.4/5GHz band, 802.11ac operates in 5GHz band, 802.11ad operates in 60Ghzband.

(iii)802.16 - WiMax: IEEE802.16 is a collection of wireless broadband standards including exclusive description of link layer. WiMax provide data rates from 1.5 Mb/s to 1Gb/s.

(iv)802.15.4-LR-WPAN: IEEE802.15.4 is a collection of standards for low rate wireless personal area network(LR-WPAN). Basis for high level communication protocols such as ZigBee. Provides data rate from 40kb/s to250kb/s.

(v)2G/3G/4G- Mobile Communication: Data rates from 9.6kb/s(2G) to up to100Mb/s(4G).

2) Network/Internet Layer:

Responsible for sending IP datagrams from source n/w to destination n/w. Performs the host addressing and packet routing. Datagrams contains source and destinationaddress.

Protocols:

- **IPv4:** Internet Protocol version4 is used to identify the devices on a n/w using a hierarchical addressing scheme. 32 bit address. Allows total of 2**32addresses.

- **IPv6:** Internet Protocol version6 uses 128 bit address scheme and allows 2**128 addresses.

- **6LOWPAN:**(IPv6overLowpowerWirelessPersonalAreaNetwork)operatesin 2.4 GHz frequency range and data transfer 250 kb/s.

3) Transport Layer: This layer provides functions such as error control, segmentation, flow control and congestion control. So this layer protocols provide end-to-end message transfer capability independent of the underlying network.

Protocols:

(i)TCP: ***Transmission Control Protocol*** used by web browsers(along with HTTP and HTTPS), email(along with SMTP, FTP). Connection oriented and stateless protocol. IP Protocol deals with sending packets, TCP ensures reliable transmission of protocols in order. Avoids n/w congestion and congestion collapse.

(ii)UDP: ***User Datagram Protocol*** is connectionless protocol. Useful in time sensitive applications, very small data units to exchange. Transaction oriented and stateless protocol. Does not provide guaranteed delivery.

4) Application Layer:

Application layer protocols define how the applications interface with the lower layer protocols to send over the network.

Protocols:

(i) HTTP: Hypertext Transfer Protocol (HTTP) is an application-layer protocol for transmitting hypermedia documents, such as HTML. It was designed for communication between web browsers and web servers, but it can also be used for other purposes. Follow request- response model Stateless protocol.

(ii)CoAP: ***Constrained Application Protocol*** for machine-to-machine(M2M) applications with constrained devices, constrained environment and constrained n/w. The protocol is especially targeted for constrained hardware such as 8-bits microcontrollers, low power sensors and similar devices that can't run on HTTP or TLS.

(iii)Web Socket: allows full duplex communication over a single socket connection.

(iv) MQTT: ***Message Queue Telemetry Transport*** is light weight messaging protocol based on publish-subscribe model. Uses client server architecture. Well suited for constrained environment

(v)XMPP: Extensible Message and Presence Protocol for real time communication and streaming XML data between network entities. Support client-server and server-server communication.

(vi)DDS: ***Data Distribution Service*** is data centric middleware standards for device-to-device or machine-to-machine communication. Uses publish-subscribe model.

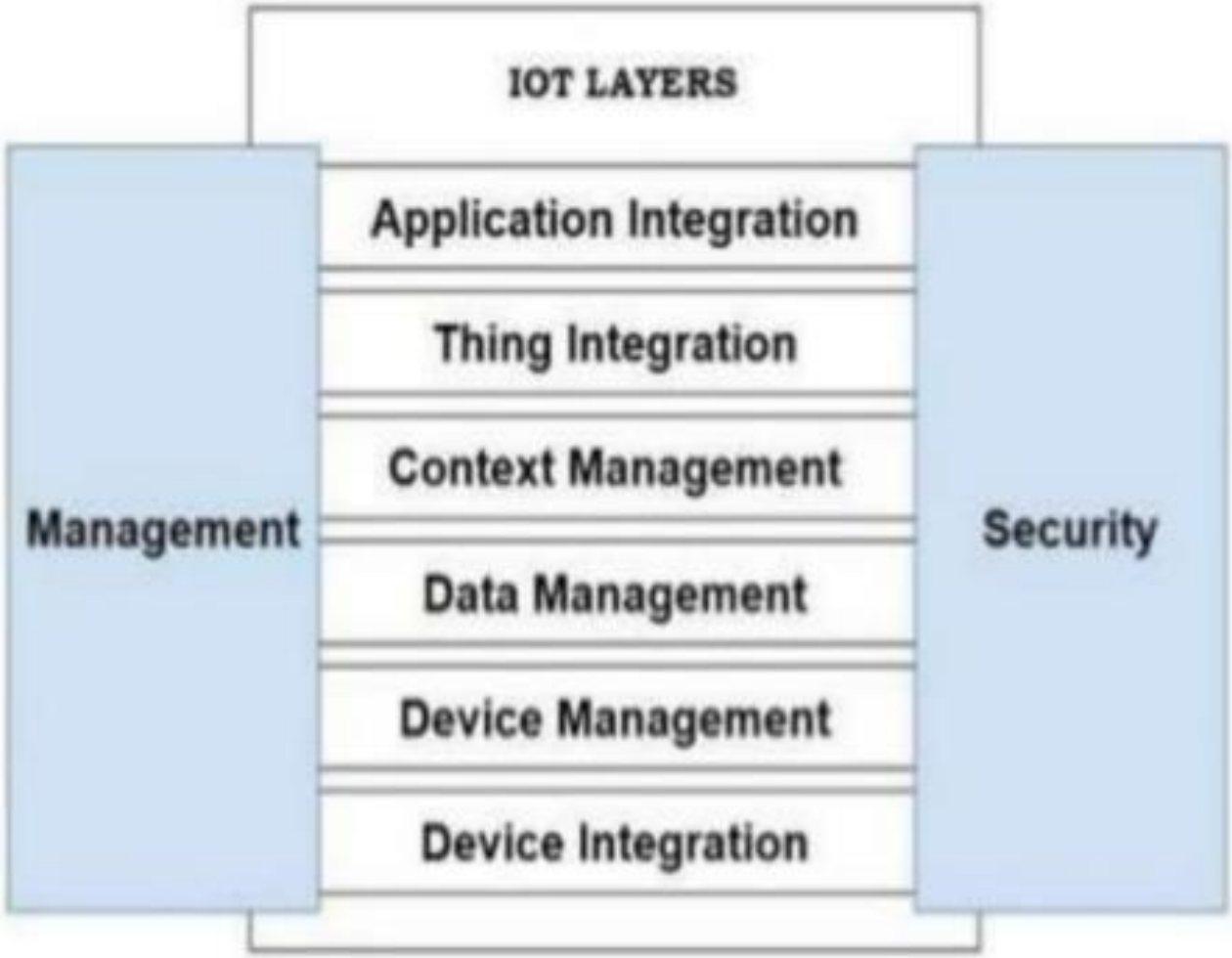
(vii)AMQP: ***Advanced Message Queuing Protocol*** is open application layer protocol for business messaging. Supports both point-to-point and publish-subscribe model

LOGICAL DESIGN of IOT

Logical design of IoT system refers to an abstract representation of the entities & processes without going into the low-level specifies of the implementation. For understanding Logical Design of IoT, we describes given below terms.

- IoT Functional Blocks
- IoT Communication Models
- IoT Communication APIs

Logical Design of IoT

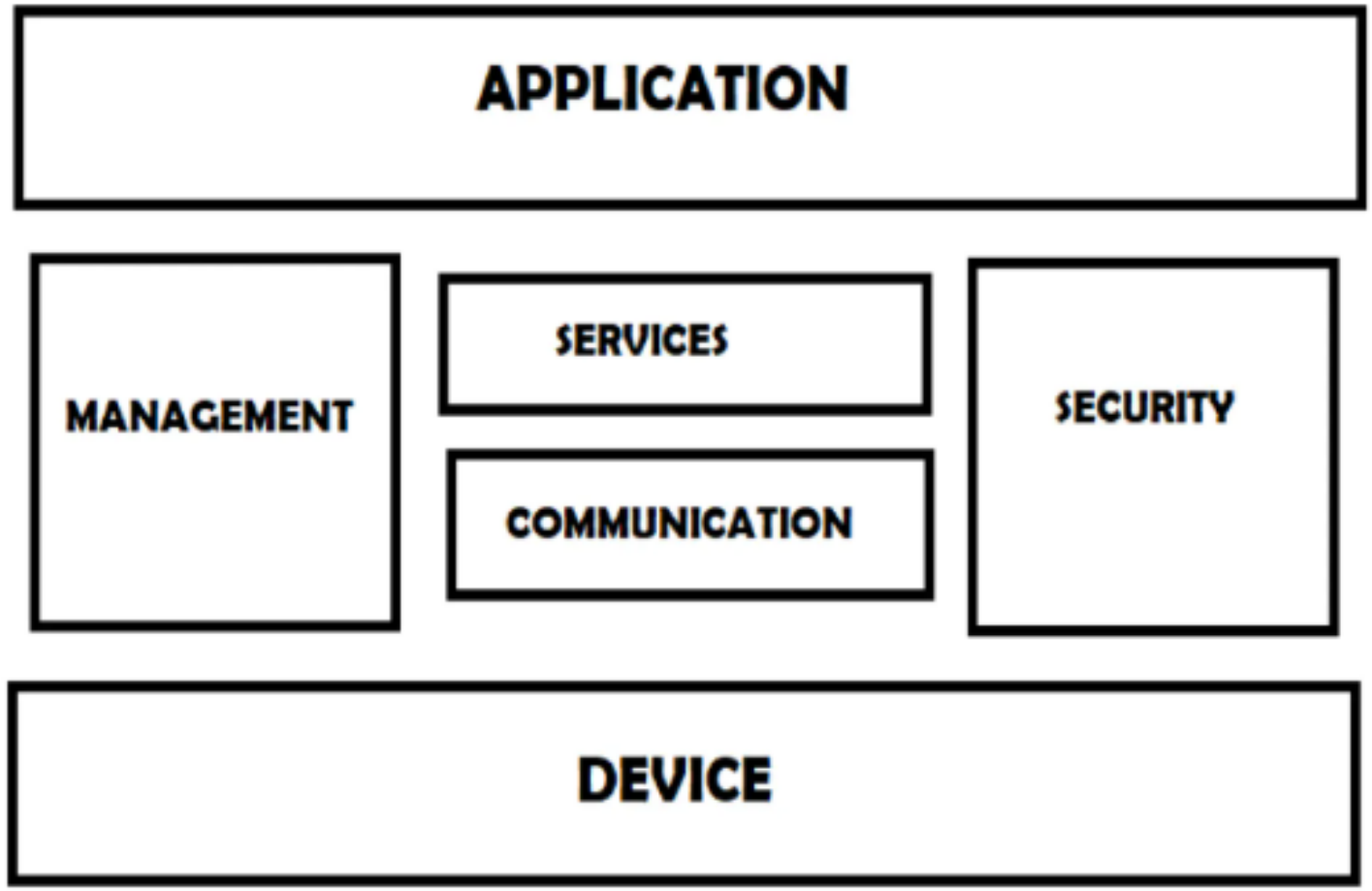


IoT Functional Block

- An IoT system comprises of a number of functional blocks that provide the system the capabilities for identification, sensing, actuation, communication, and management
  - Device
  - Communication
  - Services
  - Management
  - Security
  - Application

functional blocks are:

Device: An IoT system comprises of devices that provide sensing, actuation, monitoring and control functions.
Communication: Handles the communication for the IoT system.
Services: services for device monitoring, device control service, data publishing services and services for device discovery.
Management: this blocks provides various functions to govern the IoT system.
Security: this block secures the IoT system and by providing functions such as authentication , authorization, message and content integrity, and data security.
Application: This is an interface that the users can use to control and monitor various aspects of the IoT system. Application also allow users to view the system status and view or analyze the processed data.


APPLICATION
MANAGEMENT
SERVICES
COMMUNICATION
SECURITY
DEVICE

IoT Communication Models

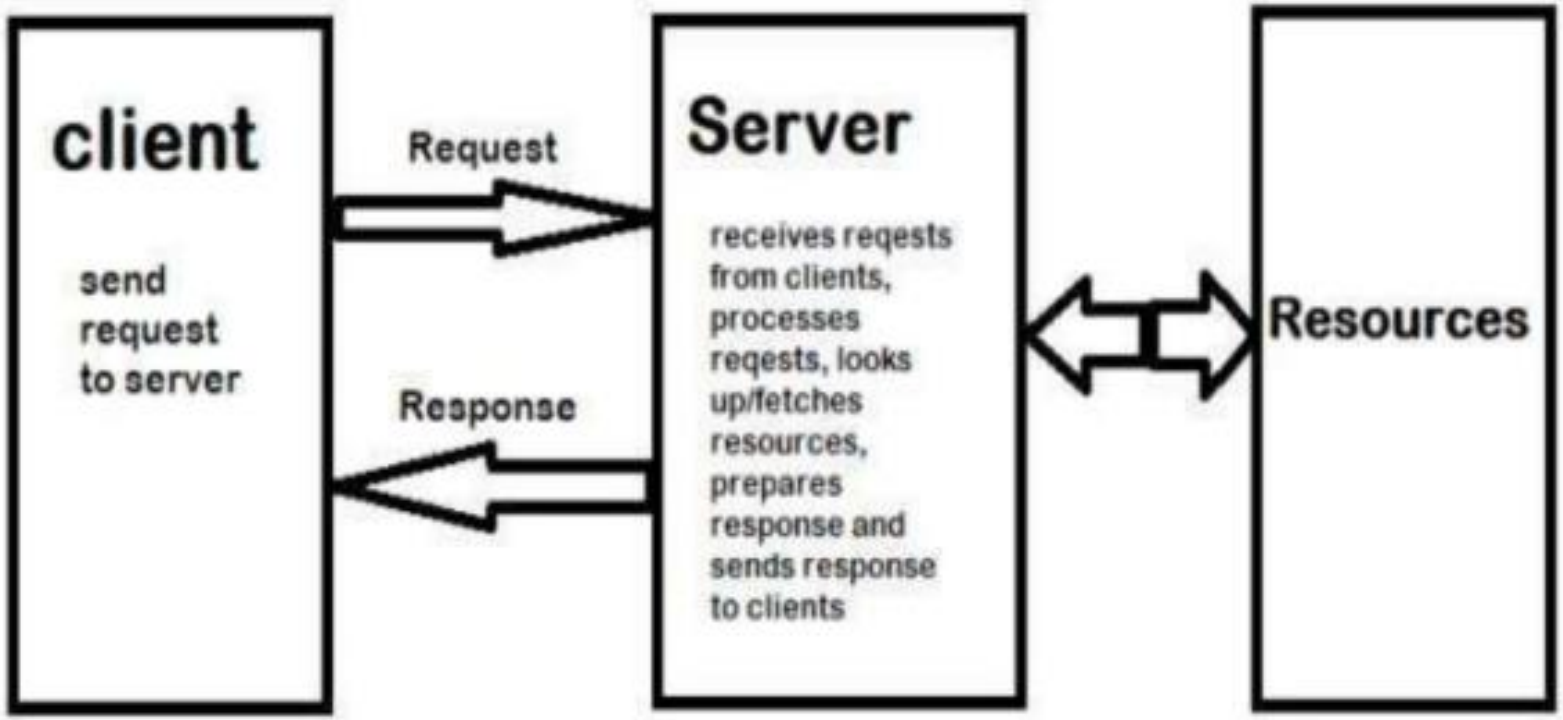
Request-Response Model

Publish-Subscribe Model

Push-Pull Model

Exclusive Pair Model

Request-Response Model



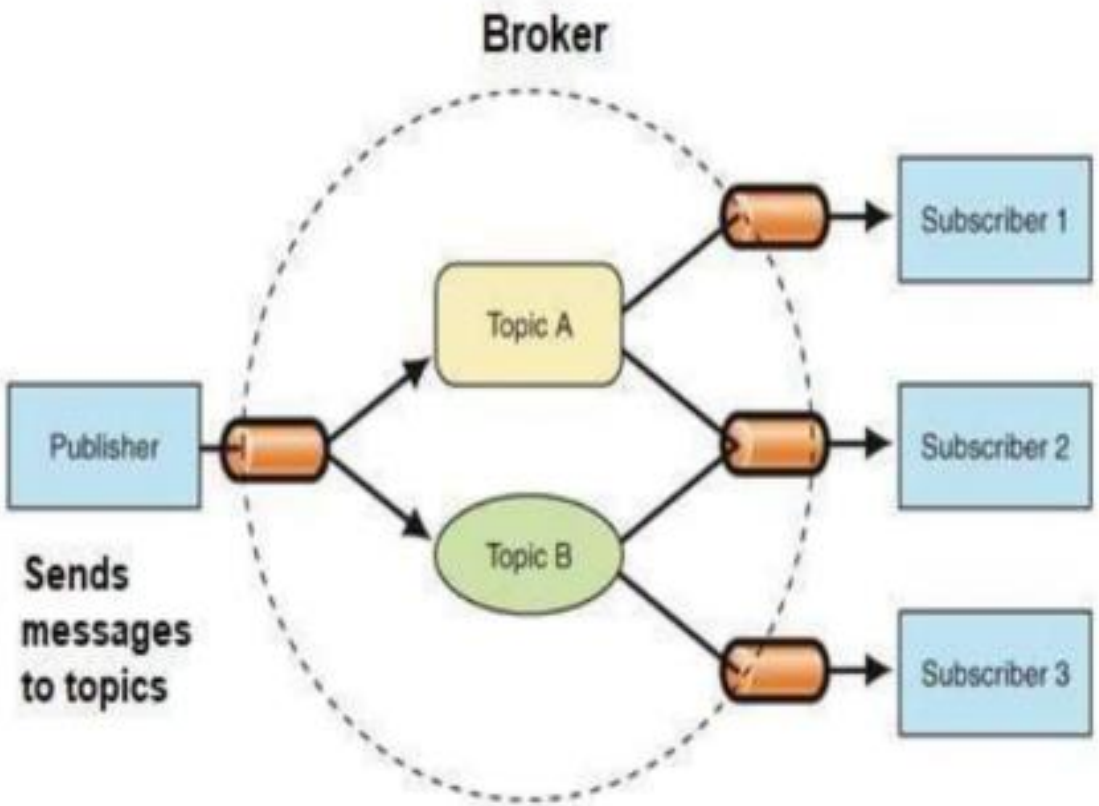
Request-Response Communication Model

•***Request-response model*** is communication model in which the client sends requests to the server and the server responds to the requests. When the server receives a request, it decides how to respond, fetches the data, retrieves resource representation, prepares the response, and then sends the response to the client. Request-response is a stateless communication model and each request-response pair is independent of others.

•HTTP works as a request-response protocol between a client and server. A web browser may be the client, and an application on a computer that hosts a web site may be the server.

Example: A client (browser) submits an HTTP request to the server; then the server returns a response to the client. The response contains status information about the request and may also contain the requested content.

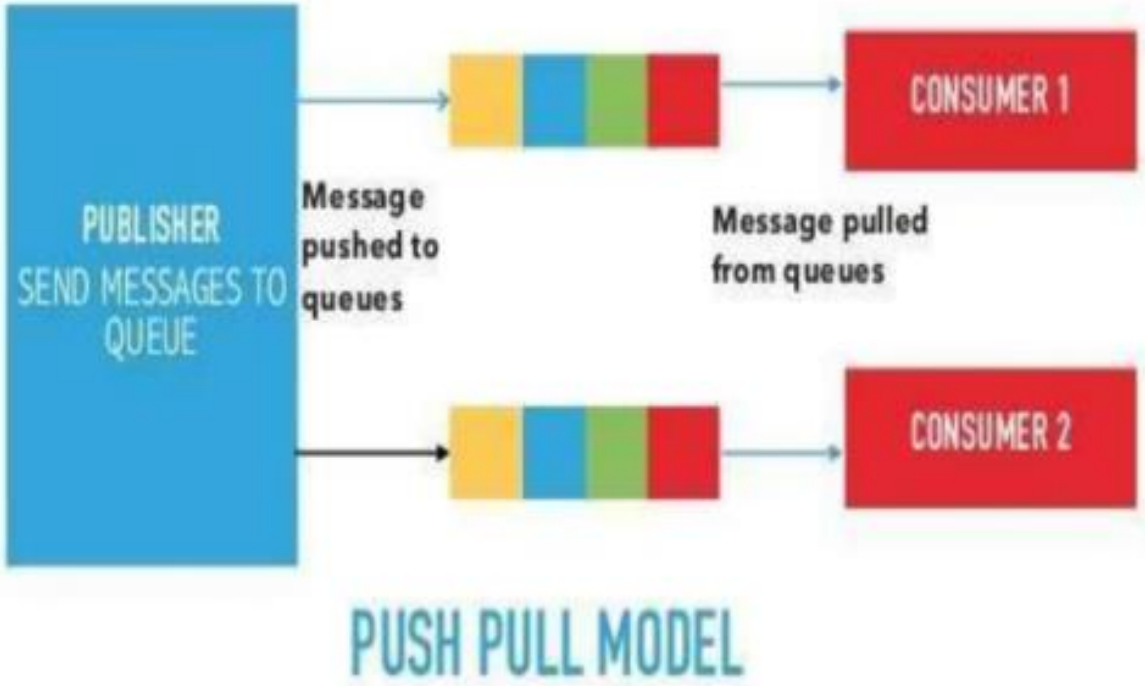
Publish-Subscribe Model



•Publish-Subscribe is a communication model that involves publishers, brokers and consumers. Publishers are the source of data. Publishers send the data to the topics which are managed by the broker.

•Publishers are not aware of the consumers. Consumers subscribe to the topics which are managed by the broker. When the broker receive data for a topic from the publisher, it sends the data to all the subscribed consumers.

Push-Pull Model



- Push-Pull is a communication model in which the data producers push the data to queues and the consumers Pull the data from the Queues.

- Producers do not need to be aware of the consumers. Queues help in decoupling the messaging between the Producers and Consumers.

- Queues also act as a buffer which helps in situations when there is a mismatch between the rate at which the producers push data and the rate rate at which the consumer pull data.

Exclusive Pair Model

- Exclusive Pair is a bidirectional, fully duplex communication model that uses a persistent connection between the client and server. Connection is setup it remains open until the client sends a request to close the connection.

- Client and server can send messages to each other after connection setup. Exclusive pair is stateful communication model and the server is aware of all the open connections.

IoT Levels

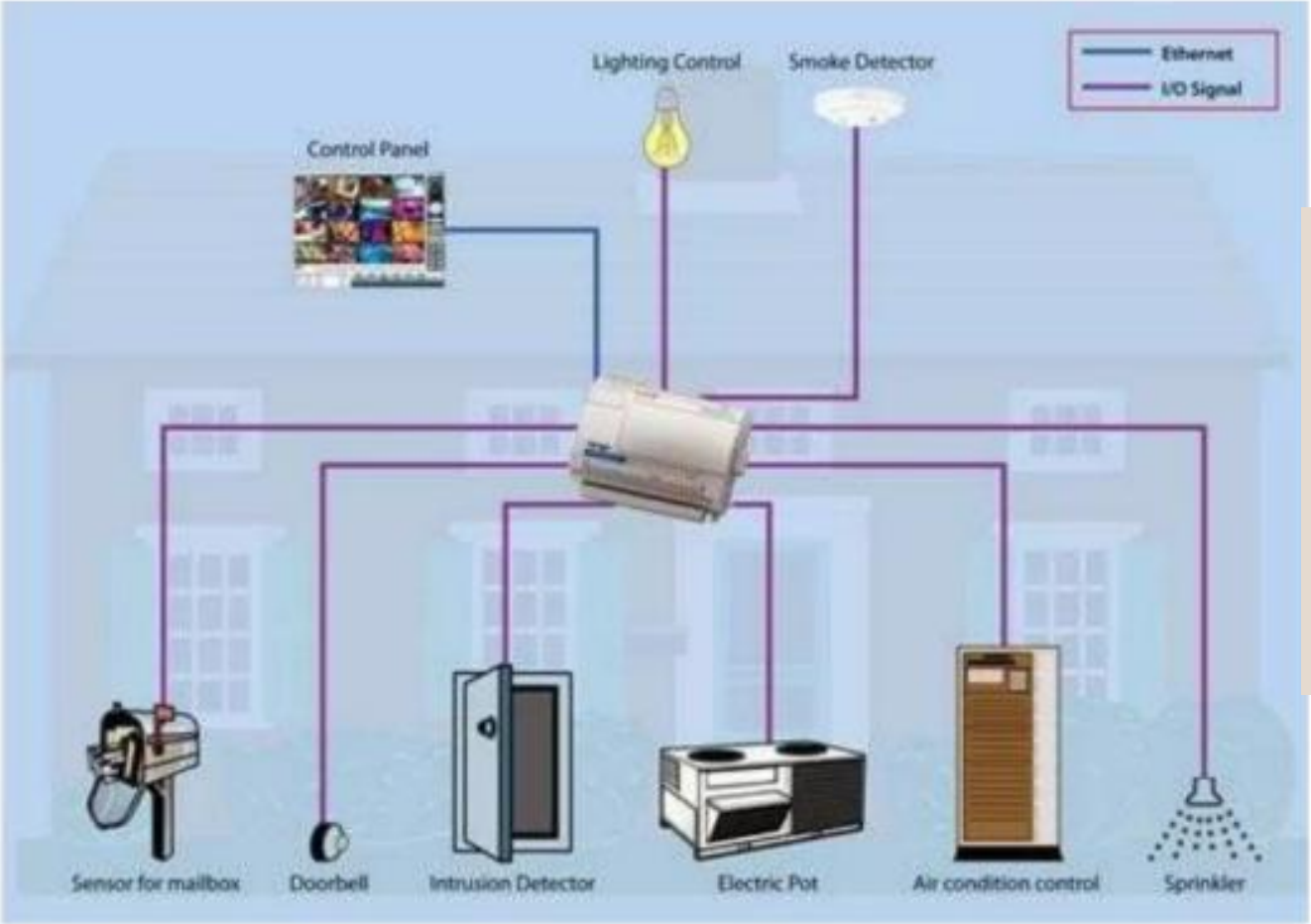
IoT Levels

IOT LEVEL 1

- IoT systems have a single device that performs sensing or actuation, stores and analyses it, and hosts the application

- IoT system-level-l is the best example for modeling low complexity and low-cost solution where the analysis requirement is not comprehensive and the data involved is not big.

IoT – Level 1 Example : Home Automation System



EXAMPLE

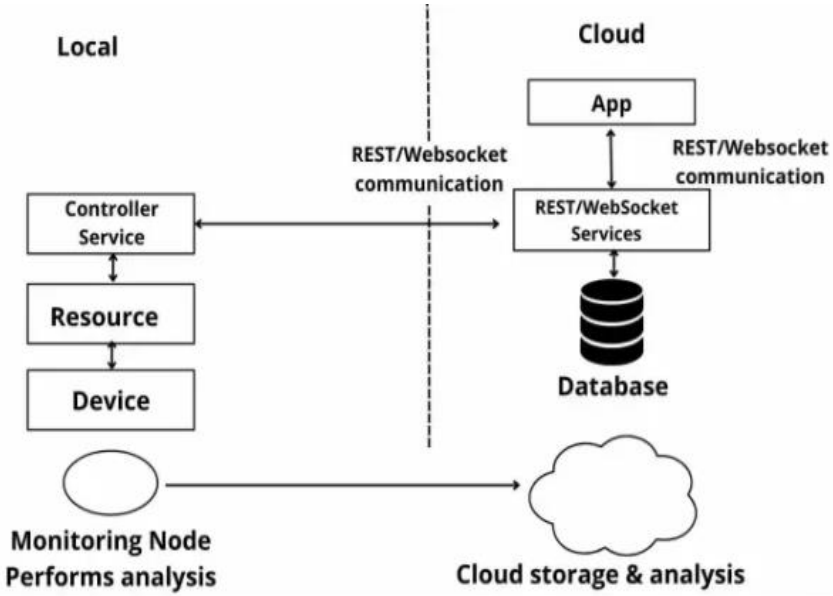
- Room temperature is monitored using temperature sensor and data is stored/analyzed locally.
- Based on analysis made, control action is triggered using mobile app or it can just help in status monitoring.

IOT LEVEL 2:

- This level-2 is complex compare to level-1. Moreover rate of sensing is faster compare to level-1.
- This level has voluminous size of data. Hence cloud storage is used.
- Data analysis is carried out locally. Cloud is used for only storage purpose.
- Based on data analysis, control action is triggered using web app or mobile app.

Examples: Agriculture applications, room freshening solutions based on odour sensors etc.

IoT – Level 2 Example: Smart Irrigation



IOT LEVEL 3:

- • Data here is voluminous i.e. big data. Frequency of data sensing is fast and collected sensed data is stored on cloud as it is big.
  • Data analysis is done on the cloud side and based on analysis control action is triggered using mobile app or web app.

- **Examples:** Agriculture applications, room freshening solutions based on odour sensors etc.

IoT – Level 3 Example: Tracking Package Handling

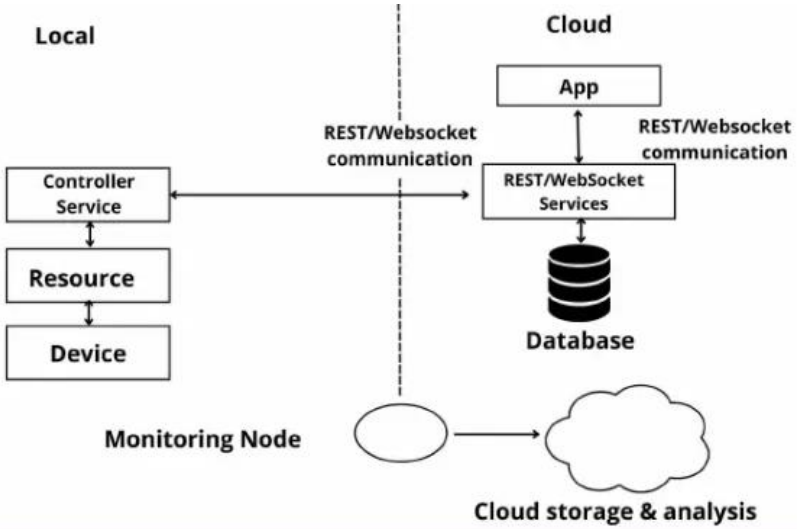
Sensors used

Accelrometer

sense movement or vibrations

Gyroscope

Gives orientation info

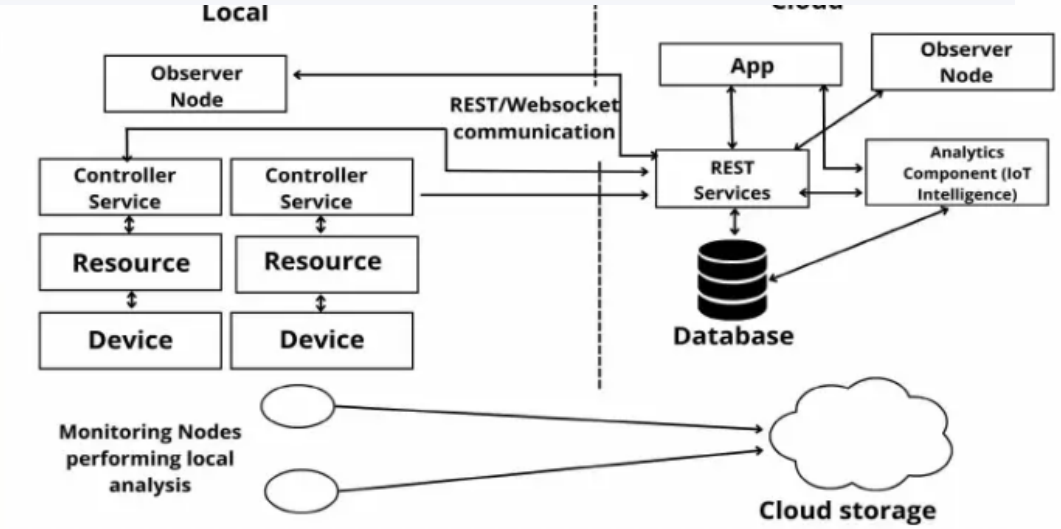


IOT LEVEL 4:

- This level consists of multiple sensors, data collection and analysis and control & monitoring app.
- At this level-4, multiple sensors are used which are independent of the others.
- The data collected using these sensors are uploaded to the cloud separately. The cloud storage is used in this level due to requirement of huge data storage.
- The data analysis is performed on the cloud and based on which control action is triggered either using web app or mobile app.

IoT – Level 4 Example: Noise Monitoring

Sound Sensors are used

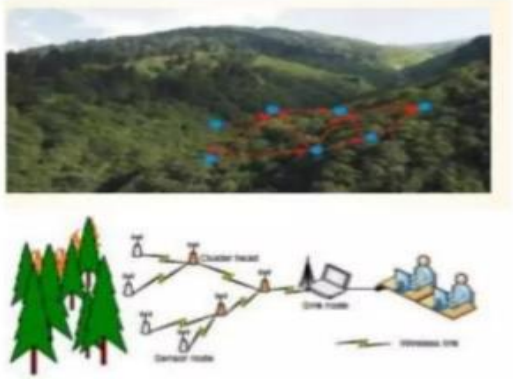


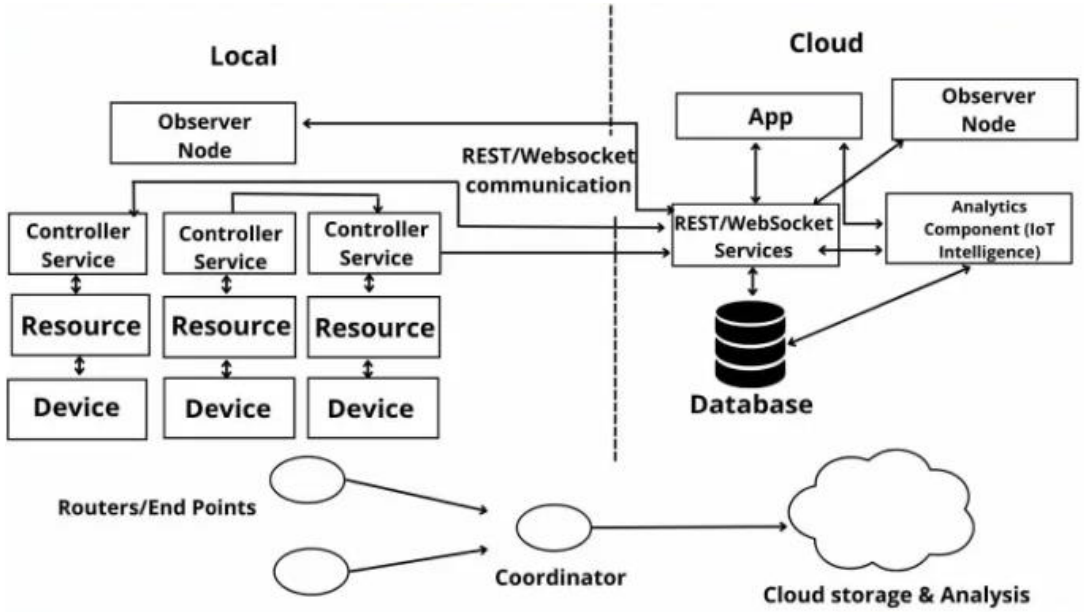
IOT LEVEL 5:

- This level consists of multiple sensors, coordinator node, data collection and analysis and control & monitoring app.
- This level is similar to level-4 which also has huge data and hence they are sensed using multiple sensors at much faster rate and simultaneously.
- The data collection and data analysis is performed at the cloud level.
- Based on analysis,control action is performed using mobile app or web app.

IoT – Level 5 Example: Forest Fire Detection

Detect forest fire in early stages to take action while the fire is still controllable.
Sensors measure the temperature, smoke, weather, slope of the earth, wind speed, speed of fire spread, flame length





IOT LEVEL 6:

- It has several separate end nodes used for monitoring and/or transferring data to the cloud. Our database is built on the Cloud.
- The analytics portion analyses and stores data in the cloud database and displays cloud-based results.
- The centralized controller knows all nodes' status and delivers command signals to the nodes.

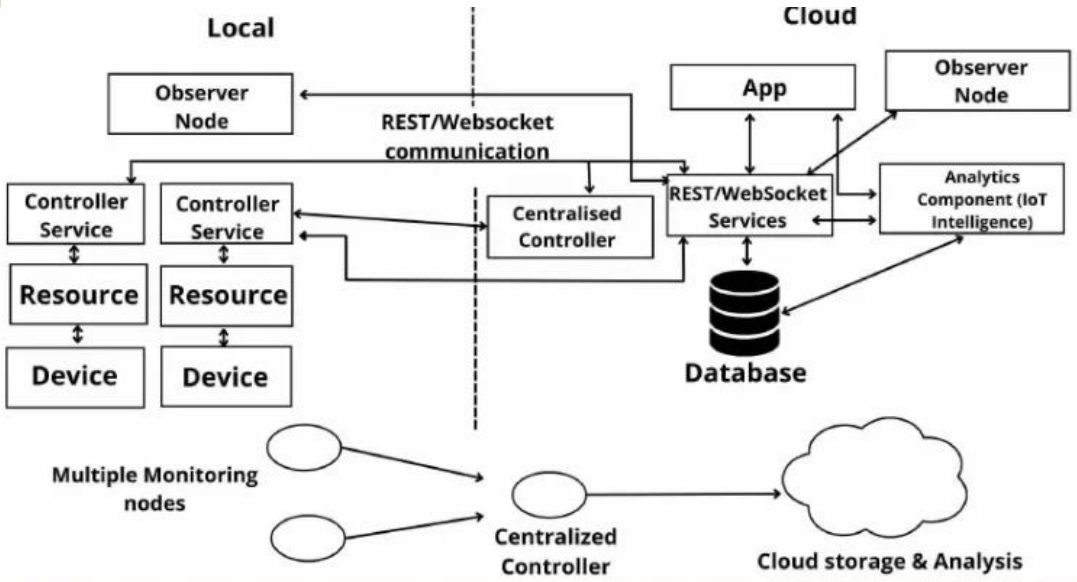
- **Example**
- Take an example of a system of weather surveillance.
- Numerous temperature, humidity, stress, etc. sensors are contained in this system

IoT – Level 6 Example: Weather Monitoring System

Sensors used

Wind speed and direction
Solar radiation
Temperature (air, water, soil)
Relative humidity

Precipitation
Snow depth
Barometric pressure
Soil moisture



Developing an **IoT Level Template** system consists of the following components:

- **Device:**

These may be sensors or actuators capable of identifying, remote sensing, or monitoring.

- **Resources:**

These are software components on IoT devices for accessing and processing.

- **Controller Service:**

It is a service that runs on the device and interacts with web services.

- **Database:**

Stores data generated from the device

Web Service:
It provides a link between IoT devices, applications, databases, and analysis components.

Analysis Component:
It performs an analysis of the data generated by the device and generates results in a form which are easy for the user to understand.

Application:
It provides a system for the user to view the system status and view product data.