## Q.1] Explain With Suitable diagram Physical Design of IOT - Things in IOT

It is referred to as the Things/Devices and protocols that are used to build an IoT system.

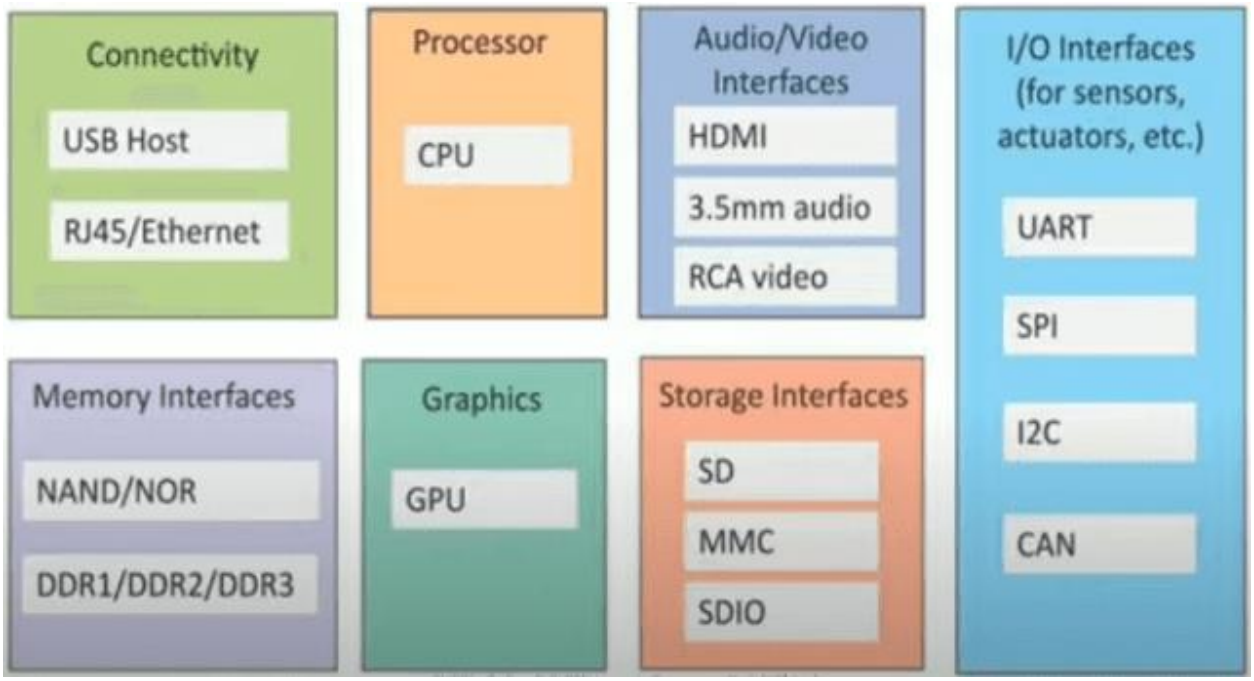
All these things/Devices are called Node Devices and every device has a unique identity that performs remote sensing, actuating, and monitoring work.

# Physical Design of IoT

## Things/Devices

Things/Devices are used to build a connection, process data, provide interfaces, provide storage, and provide graphics interfaces in an IoT system. All these generate data, in a form that can be analyzed by an analytical system and program to perform operations and used to improve the system.

for example temperature sensor that is used to analyze the temperature generates the data from a location and is then determined by algorithms.



**devices in IoT(Internet of things)**

## Connectivity

Devices like USB hosts and ETHERNET are used for connectivity between the devices and the server.

## Processor

A processor like a CPU and other units are used to process the data. these data are further used to improve the decision quality of an IoT system.

## Audio/Video Interfaces

An interface like HDMI and RCA devices is used to record audio and videos in a system.

## Input/Output interface

To give input and output signals to sensors, and actuators we use things like UART, SPI, CAN, etc.

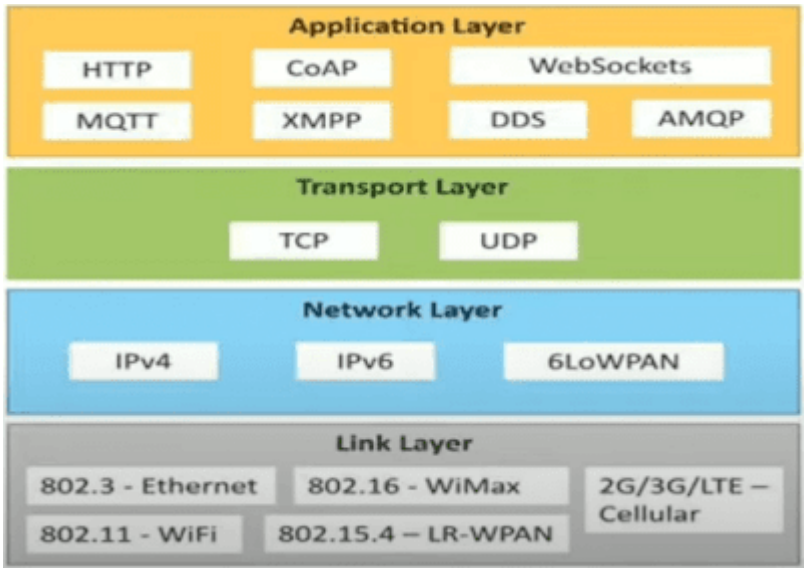
## Storage Interfaces

Things like SD, SDIO, MMC are used to store the data generated from an IoT device.

Other things like DDR and GPU are used to control the activity of an IoT system.

# IoT Protocols

These protocols are used to establish communication between a node device and a server over the internet. it helps to send commands to an IoT device and receive data from an IoT device over the internet. we use different types of protocols that are present on both the server and client side and these protocols are managed by network layers like application, transport, network, and link layer.




IoT(Internet of Things) protocols

## Application Layer protocol

In this layer, protocols define how the data can be sent over the network with the lower layer protocols using the application interface. these protocols include HTTP, WebSocket, XMPP, MQTT, DDS, and AMQP protocols.

### *HTTP*

Hypertext transfer protocol is a protocol that presents an application layer for transmitting media documents. it is used to communicate between web browsers and servers.

### *WebSocket*

This protocol enables two-way communication between a client and a host that can be run on an untrusted code in a controlled environment. This protocol is commonly used by web browsers.

### *MQTT*

It is a machine-to-machine connectivity protocol that was designed as a publish/subscribe messaging transport.

## Transport Layer

This layer is used to control the flow of data segments and handle error control. also, these layer protocols provide end-to-end message transfer capability independent of the underlying network.

### *TCP*

***Transmission Control Protocol (TCP) is a communications standard that enables application programs and computing devices to exchange messages over a network. It is designed to send packets across the internet and ensure the successful delivery of data and messages over networks***

### *UDP*

a user datagram protocol is part of an internet protocol called the connectionless protocol. this protocol is not required to establish the connection to transfer data.

## Network Layer

This layer is used to send datagrams from the source network to the destination network. we use IPv4 and IPv6 protocols as host identification that transfers data in packets.

### *IPv4*

This is a protocol address that is a unique and numerical label assigned to each device connected to the network. an IP address performs two main functions host and location addressing. IPv4 is an IP address that is 32-bit long.

### IPv6

It is a successor of IPv4 that uses 128 bits for an IP address.

## Link Layer

Link-layer protocols are used to send data over the network's physical layer. it also determines how the packets are coded and signaled by the devices.

### *Ethernet*

Ethernet is a widely used networking technology that allows devices to communicate and share data over a local area network (LAN).

### *WiFi*

Wi-Fi is a wireless technology that enables devices to connect to the internet and local networks without the need for physical wired connections.

## Q.2 Logical Design of IoT | Communication Models | APIs | Functional Blocks

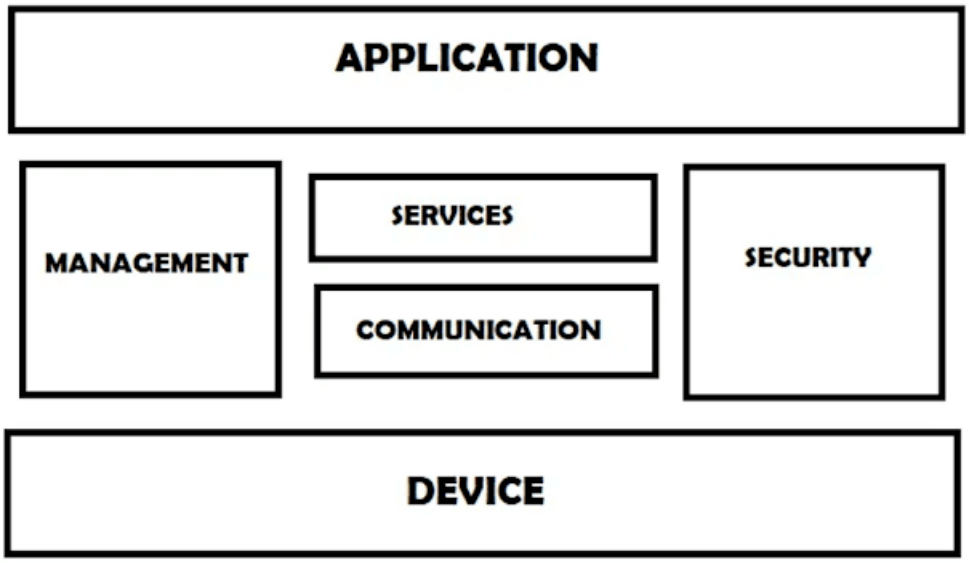
**The logical design** of an **IoT** system refers to an abstract representation of entities and processes without going into the low-level specifies of implementation. it uses **Functional Blocks**, **Communication Models**, and **Communication APIs** to implement a system.

## Logical Design of the Internet of Things(IoT)

1. IoT Functional Blocks
2. IoT Communication Models
3. IoT Communication APIs

# IoT Functional blocks

An IoT system consists of a number of functional blocks like Devices, services, communication, security, and application that provide the capability for sensing, actuation, identification, communication, and management.




**IoT functional blocks**

## Application

An application serves as an interface that empowers users to access and control the system, enabling them to view real-time status updates and analyze its performance.

## Management

This functional block provides various functions that are used to manage an IoT system.

## Services

This functional block provides some services like monitoring and controlling a device and publishing and deleting the data and restoring the system.

## Communication

This block handles the communication between the client and the cloud-based server and sends/receives the data using protocols.

## Security

This block is used to secure an IoT system using some functions like authorization, data security, authentication, 2-step verification, etc.

## Device

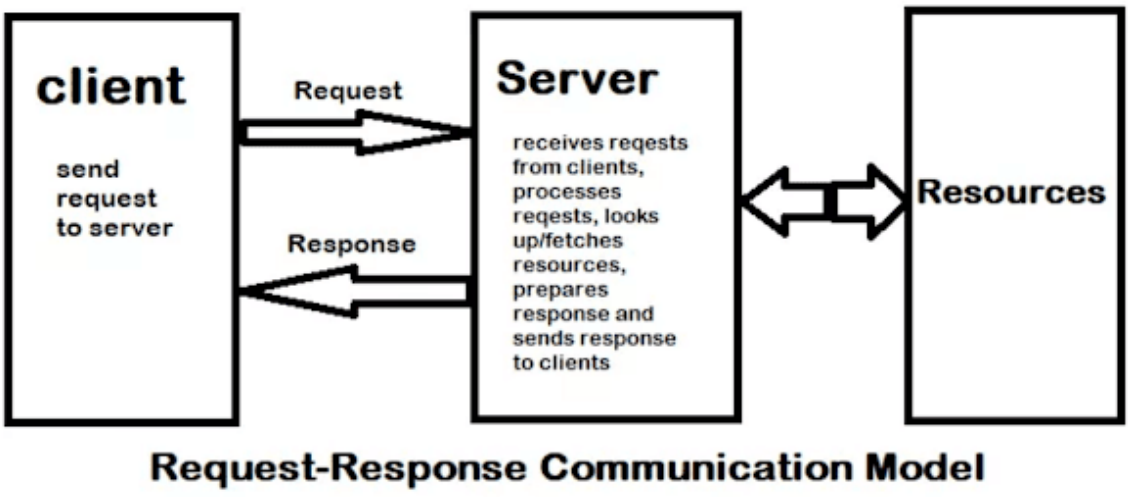
These devices are used to provide sensing and monitoring control functions that collect data from the outer environment.

# IoT Communication Models

There are several different types of models available in an IoT system that is used to communicate between the system and server like the request-response model, publish-subscribe model, push-pull model, exclusive pair model, etc.

## Request-Response Communication Model

This model is a communication model in which a client sends the request for data to the server and the server responds according to the request. when a server receives a request it fetches the data, retrieves the resources and prepares the response, and then sends the data back to the client.




**Request response communication model**

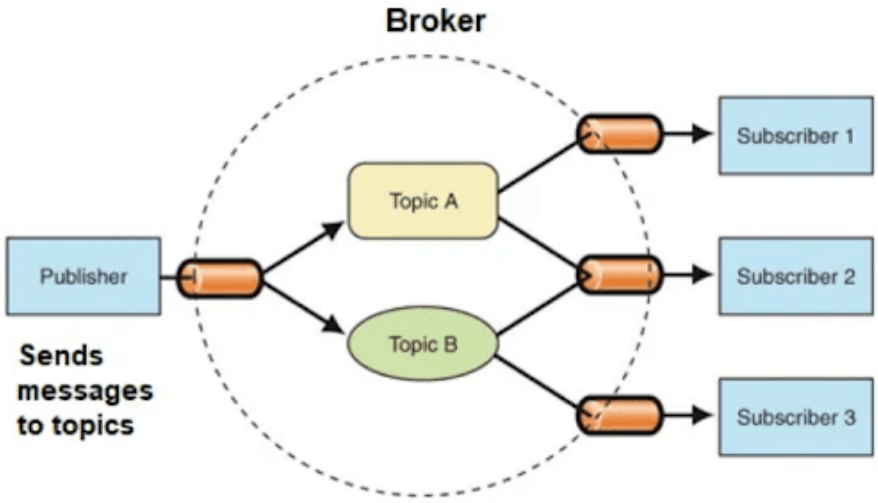
In simple terms, we can say that in the request-response model, the server sends the response equivalent to the request of the client. in this model, HTTP works as a request-response protocol between a client and server.

### *Example*

When we search a query on a browser then the browser submits an HTTP request to the server and then the server returns a response to the browser(client).

## Publish-Subscribe Communication Model

In this communication model, we have a broker between the publisher and the consumer. here publishers are the source of data but they are not aware of consumers. they send the data managed by the brokers and when a consumer subscribes to a topic that is managed by the broker and when the broker receives data from the publisher it sends the data to all the subscribed consumers.




**Published-subscribe communication model**

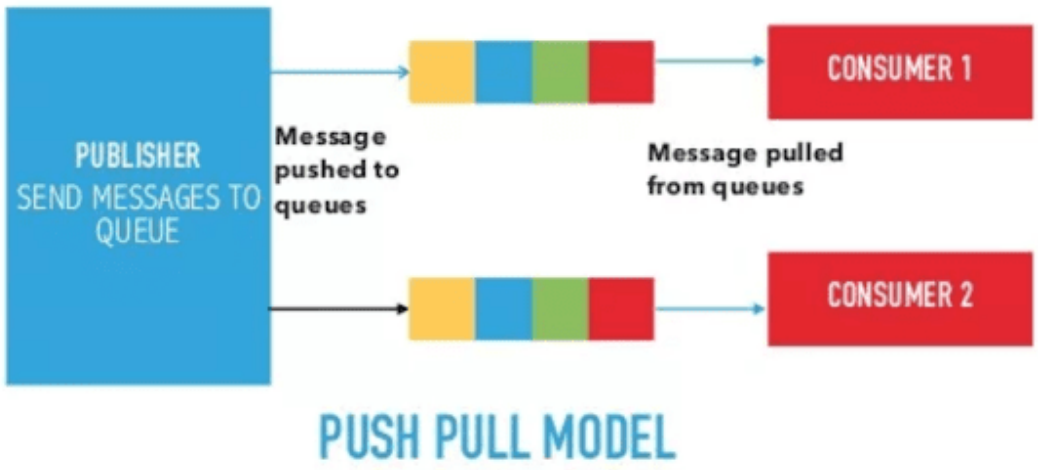
### *Example*

On the website many times we subscribed to their newsletters using our email address. these email addresses are managed by some third-party services and when a new article is published on the website it is directly sent

to the broker and then the broker sends these new data or posts to all the subscribers.

## Push-Pull Communication Model

It is a communication model in which the data push by the producers in a queue and the consumers pull the data from the queues. here also producers are not aware of the consumers.




**Push Pull Model**

### *Example*

When we visit a website we saw a number of posts that are published in a queue and according to our requirements, we click on a post and start reading it.

## Exclusive Pair Communication Model

It is a bidirectional fully duplex communication model that uses a persistent connection between the client and server. here first set up a connection between the client and the server and remain open until the client sends a close connection request to the server.


**Exclusive Pair communication model**

# IoT communication APIs

These APIs like REST and WebSocket are used to communicate between the server and system in IoT.

# REST-based communication APIs

Representational state transfer(REST) API uses a set of architectural principles that are used to design web services. these APIs focus on the systems' resources that how resource states are transferred using the request-response communication model. This API uses some architectural constraints.

## *Client-server*

Here the client is not aware of the storage of data because it is concerned about the server and similarly the server should not be concerned about the user interface because it is a concern of the client. and this separation is needed for independent development and updating of the server and client. no matter how the client is using the response of the server and no matter how the server is using the request of the client.

## *Stateless*

It means each request from the client to the server must contain all the necessary information to understand the server. because if the server can't

understand the request of the client then it can't fetch the requested data in a proper manner.

### *Cacheable*

In response, if the cache constraints are given then a client can reuse that response in a later request. it improves the efficiency and scalability of the system without loading extra data.

A RESTful web API is implemented using HTTP and REST principles.

## WebSocket-based communication API

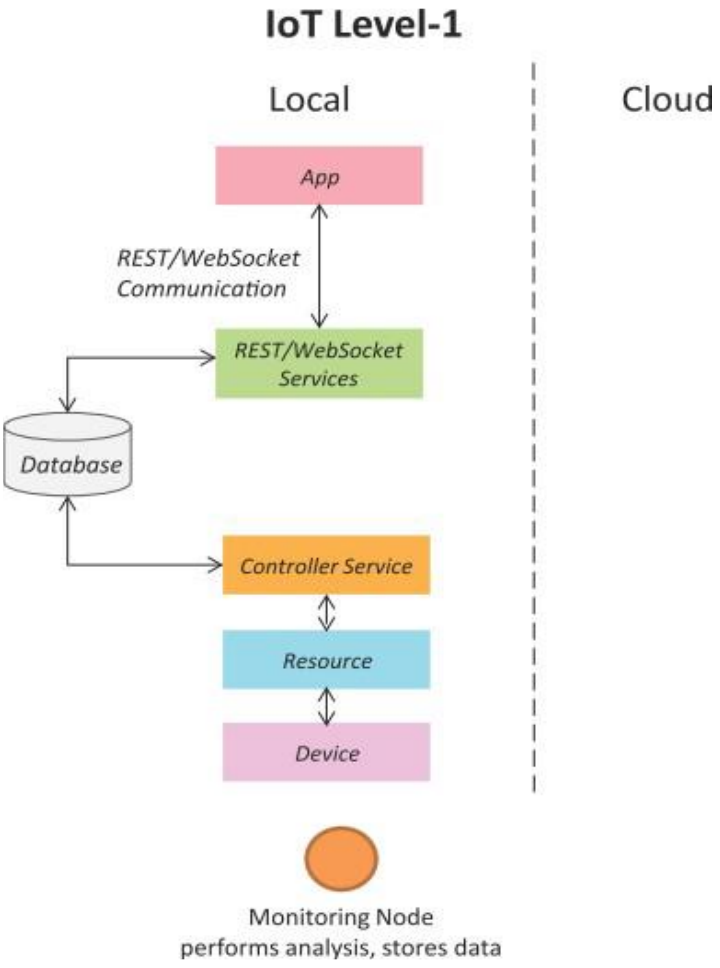
This type of API allows bi-directional full-duplex communication between server and client using the exclusive pair communication model. This API uses full-duplex communication so it does not require a new connection setup every time when it requests new data. WebSocket API begins with a connection setup between the server and client and if the WebSocket is supported by the server then it responds back to the client with a successful response after the setup of a connection server and the client can send data to each other in full-duplex mode.

this type of API reduces the traffic and latency of data and makes sure that each time when we request new data it cannot terminate the request.

# IOT Level 1

## IOT Level 1:

1. Level 1 is the simplest of all Levels.
2. A single node/device in a level-1 IoT system performs sensing and/or actuation, stores data, performs analysis, and hosts the application.
3. It is appropriate for modeling low-cost and low-complexity approaches where the data involved is minimal and the analysis criteria are not particularly challenging.
4. The data sensed is processed locally.
5. Data processing is performed locally.
6. Monitoring and control are performed using a mobile app or a web app.



## Example of Level 1 IOT:

At IOT Level 1, imagine you have a smart home app that allows you to control a few home appliances and lights.

The app connects to a single smart device (IoT computer) in your home, which is connected to the lights and appliances through relay switches.

The app communicates with the smart device using a special service (REST service), which allows you to see the current status of the lights and appliances and change their settings.

A controller service running on the smart device oversees this entire process, ensuring smooth communication between the app, the local database, and the relay switches.

All of this happens within your home, and you can control everything using a user-friendly graphical interface provided by the app.

## IOT LEVEL 2

1. It consists of a single node that performs sensing, actuation, and local analysis (IoT Device and collected data).
2. In this IoT Stage, a database and framework are set up in the cloud.
3. It is useful for solutions where the data is large, but the primary analysis criterion is not computationally intensive and can be performed locally.

### Let us consider an example of Smart irrigation System.

A single node monitors soil moisture and controls the irrigation system.

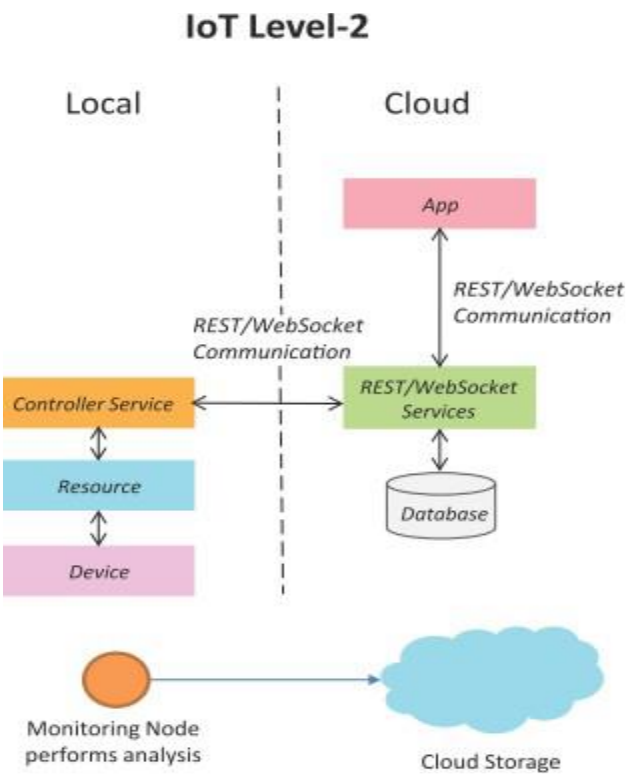
If the moisture level falls below the prescribed predefined threshold, the irrigation system is enabled.

An IoT system detects soil moisture, and the controller service tracks it and sends the data to the cloud.

Moisture levels are shown to users in an application, which can be used to create an irrigation schedule.

This level has a voluminous size of data. Hence cloud storage is used.

Data analysis is carried out locally. Cloud is used for only storage purposes.




## IOT Level 3:

It has a single node.

Data is stored and analyzed in the Cloud and it's a cloud-based application.

It is appropriate for solutions where there is large amounts of data and the primary analysis criterions are computationally intensive.

**IoT Level-3 Example**

As an example, consider package monitoring in a distribution system.

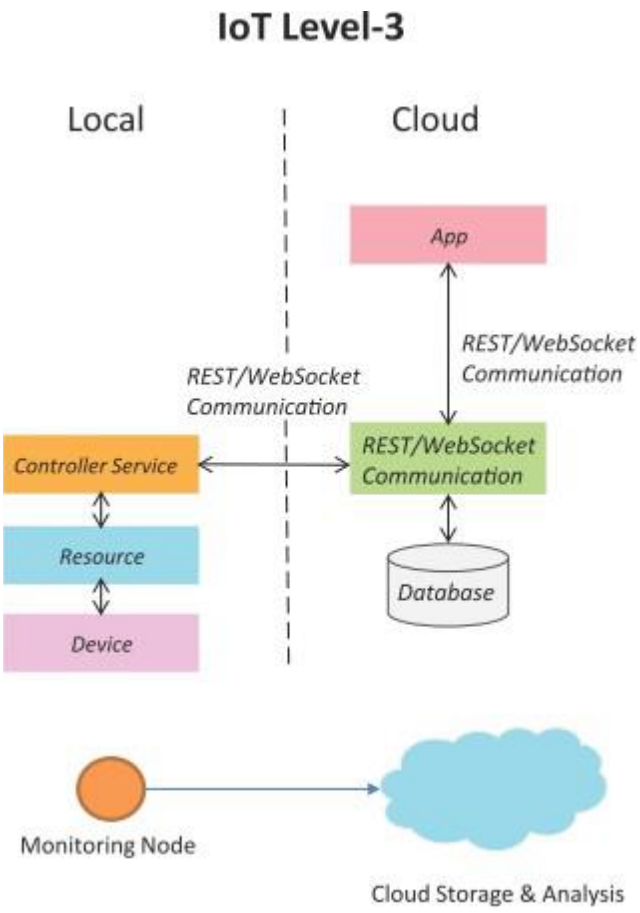
The movements that occur to the package are reviewed here. If they exceed the threshold, an alarm is triggered.

To detect these movements, the IoT system has gyroscope and accelerometer sensors.

The controller service uses Websocket API to send real-time data to the cloud, which is useful in real-time applications due to its low overhead.

The data is voluminous, i.e. large data, in this case. The data sensing frequency is high, and the collected sensed data is stored on the cloud because it is large

Data is analyzed in the cloud, and control actions are activated using a mobile app or a web app based on the results of the analysis.


IoT Level-3
Local
Cloud
App
REST/WebSocket
Communication
REST/WebSocket
Communication
REST/WebSocket
Communication
Controller Service
Resource
Device
Database
Monitoring Node
Cloud Storage & Analysis