Unit 4

Sensors & Actuator Networks:

1) Sensors: These help to find what is happening in the surrounding environment. sensors are becoming smarter, using less energy & can be fit into any system

2) Actuators: These take action based on what sensors detect.

3) Sensor Network: Wireless Sensor Network (WSN) are formed by many sensor nodes, where every sensor detect specific thing
Body Sensor Network (BSN) are used to monitor health

4) Cloud : It manages sensor data, performs analysis ~~and takes decision~~.

Sensor to Cloud Integration:

Data collected by sensors & actions taken by actuators must be sent to cloud for storage & analysis. There are various cloud options available

1) Public cloud: Used to store large amount of data.
2) Edge/Fog cloud: These are closer to where data is generated i.e.) cloud should be near to sensors.

Wireless Sensor Network	Cloud
1) Group of tiny sensors all work to collect data	1) Data warehouse to store info
2) Has less range	2) Can collect data from anywhere
3) They provide real time data	3) They make analysis on data.
4) They collect data from environment	4) They collect data from sensors

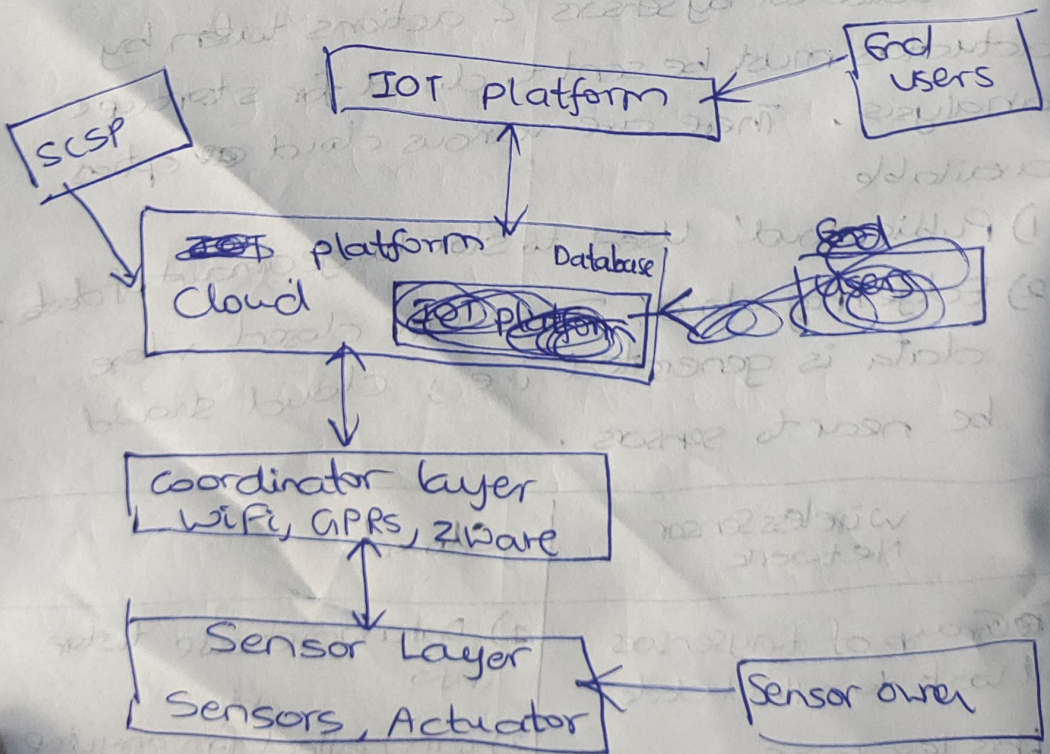
Actors in sensor-cloud:

1) End users: The users who will benefit from data collected by sensors & analysed by cloud. They take analysis & perform actors.
   Ex: Doctors monitoring patient remotely

2) Sensor owner: Responsible for deploying & managing sensors. They have control over the data.
   Ex: Farmers using sensors for soil moisture
   Farmers will only have data

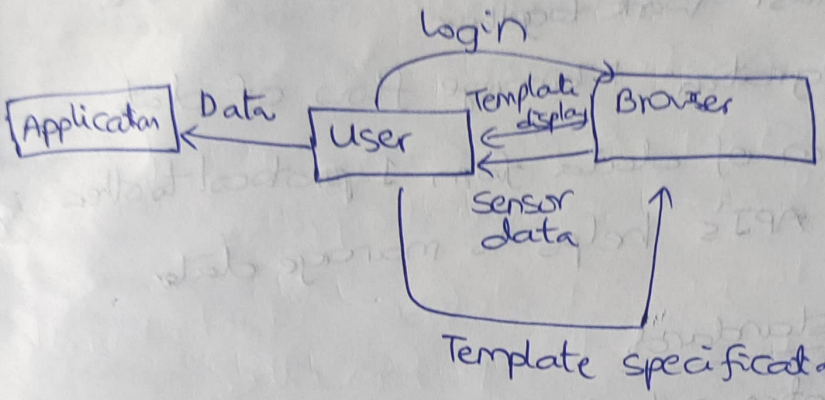
3) Sensor Cloud service Provider: (SCSP)
   They provide platforms & services for storing, analyzing & processing data.
   Ex: Azure, Google Cloud, AWS

Layered Architecture for sensor-cloud integration

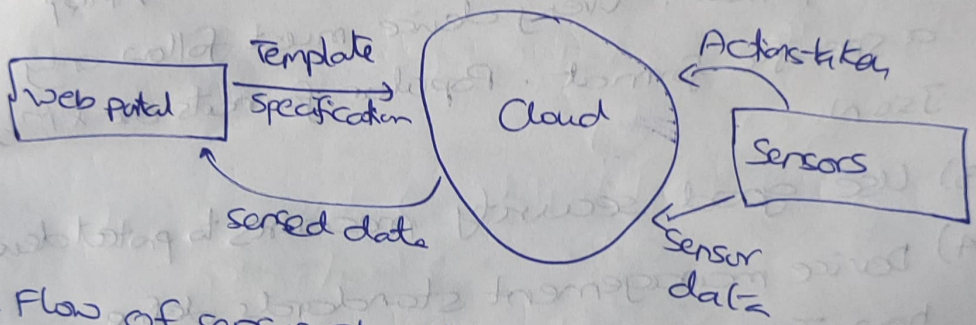


End users registers themselves in the platform.
Sensor owner deploys sensors.

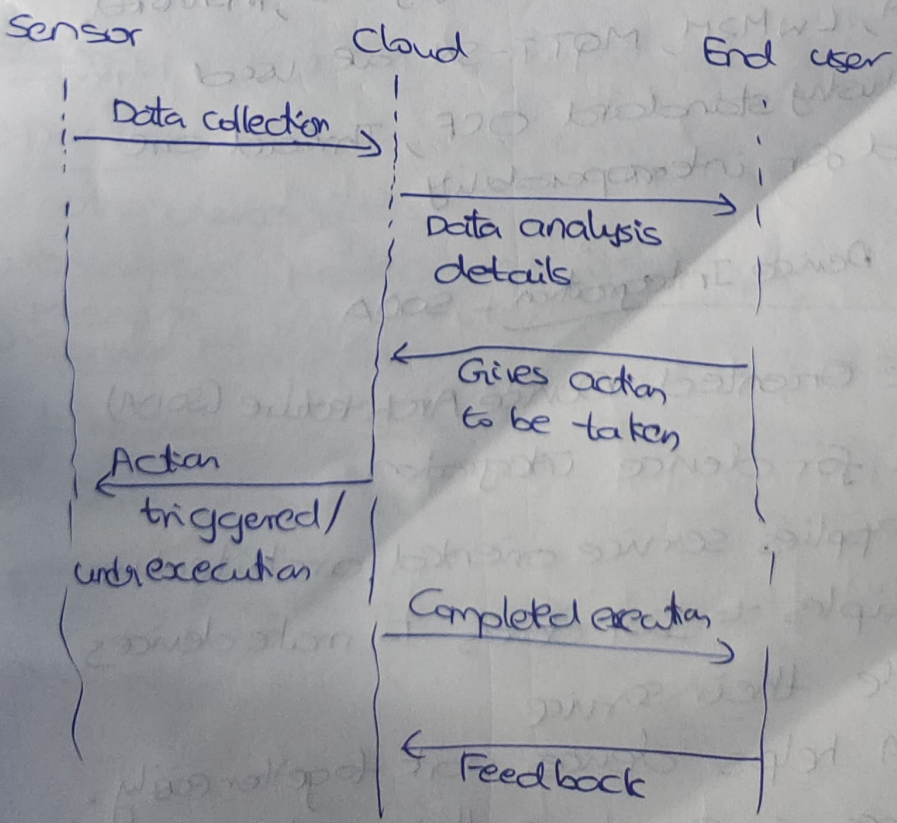
User view:



Real view:



Work Flow of sensor cloud:



IOT device integration:

To integrate we need the following

1) Interoperability: Diff IOT devices, platforms & systems must work together
2) Data integration: Manage flow of data
3) Conversion of data from 1 protocol to other
4) Usage of API's helps to manage data.

IOT device standards:

1) Communication protocols: uses MQTT, CoAP, HTTP, AMQP
2) Data format: Every device must follow only a specific format. Popular formats are XML, JSON
3) Use good security measures to protect devices
4) Device management standards define how IOT devices are configured, monitored. OMA, LWM2M, MQTT-SN are used
5) Industry standard OCF, Thread are used for interoperability

IOT Device Integration - SODA

Service Oriented Device Architecture (SODA)

→ Used for device integration

→ It applies service oriented architecture principles to IOT devices to make devices provide their service

→ SODA helps devices work together easily.

→ Devices can share their abilities with each other.

→ Devices can communicate using SOAP and also connect to internet & cloud.

Ex: Smart Home: Motion sensors detect when someone enters room & turn on light.

Device Profile For Web Services (DPWS): Set of Rules which help devices to talk to each other

Components in DPWS:

1) Discovery: Allows devices to find each other on a network.
2) Description: once device is found, it needs to describe what to be done using WSDL.
3) Messaging: DPWS defines how messages to be sent. It uses SOAP, WS addressing, MTOM, XOP
4) Eventing: Devices would react to events.

Example: Integrating Smart Heater:

1) Discovery: Heater tells that it is in network & other devices finds it
2) Description: Heater sends its description like its temp.
3) Messaging Service: Smart Home can send message to heater via SOAP to heat the water
4) Data is sent securely & errors are handled
4) As we add more devices to smart home they must work good.

Open Source Gateway Initiative (OSGi):

A framework for developing modular Java apps.

modules are called bundles.

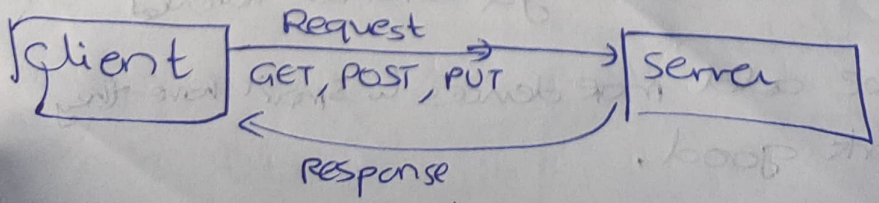
1) Bundles: Each bundle ~~is independent &~~ does a specific service
2) Service Registry: A place where bundles can tell what service they provide
3) Module Layer: It defines which bundle req what things from others
4) OSGi provides tools to configure & manage them.

Ex: Smart Home

1) Bundle: Control light, security, heater
2) Service registry:
   Control light → turn on, off
   heater → boil water
   ~~Security~~
3) Module layer:
   control light requires time & brightness information in the house to turn on or off.
4) Even we can manage all devices

Advantages: Modularity, Dynamic updates, working independent modules.

REST: Representational state Transfer.



Elements of REST API:

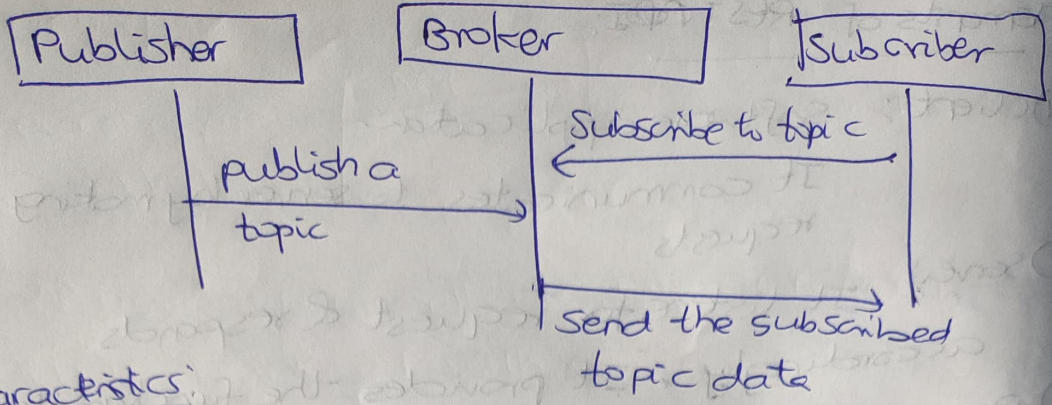
1) Client: Software application.
   It communicates to server by making requests
2) Server: It listens for request & responds accordingly. It provides the functionalities
3) Resource: Info provided by server to client.
   Each resource is identified by unique URL

Rules/Characteristics of REST:

1) client Server Architecture: Ask a request & you will get a response with required info.
2) statelessness: Every request contains all the information needed to process it.
3) Cacheability: Response from server can be cacheable / non cacheable
4) Uniform Interface: REST uses uniform & consistent way to interact with resources
5) REST has layered architecture such as security layer, load balancing, caching layers. These layer are not known to client / server.
6) Every resource can be identified by URL.

MQTT – Message Queuing Telemetry ~~Content~~ Transport:

→ Machine to machine connectivity protocol.
→ Based on Publish - Subscribe model.
→ Publisher sends messages a/c to topic to broker
→ Broker accepts messages from publisher & send them to specific subscriber
→ subscriber receives message from broker & specific topics



Characteristics:

1) MQTT is light weight:
   1) Uses less bandwidth, less power
2) Methods used: connect, disconnect, subscribe, unsubscribe, publish
3) Quality of Service levels:

First message publisher → MQTT server
second " MQTT server → Subscriber

1) Here publisher sets QOS level
2) Here subscriber sets QOS level.

3 levels of QOS:

QOS-0 (At most once): Message is delivered at most once & loss can occur

QOS-1 (At least once): Message is delivered atleast once so that it reaches receiver

QOS-2 (Exactly once): Message is delivered exactly once, no duplicates

4) Persistent connection: Broker allows clients to resume from where they left.

AMQP - Advanced Message Queuing Protocol

→ Standard protocol for messaging between applications. Uses Publish Subscribe model.

→ 2 reasons to use AMQP are
1) Reliability 2) Interoperability

Characteristics: 1) Reliability 2) Interoperability
3) Flexibility: Supports diff messaging patterns
4) Security: Includes authentication, authorization
5) Flow control: uses token based mechanism
6) Supports Publish Subscribe mode for point to point & multipoint communication

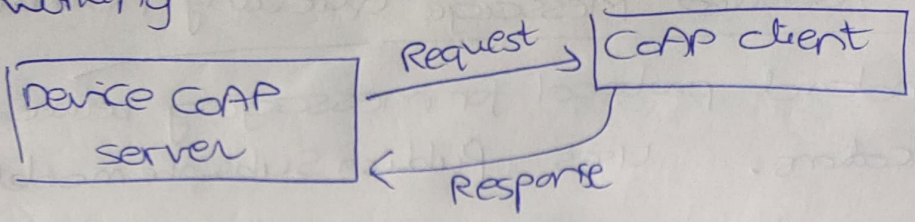
Components:
1) Producer – send msg to broker
2) Consumer – Receive msg from broker
3) Broker – store & forward messages

Message delivery guarantee service:
1) At Most once: Msg delivered atmost once, loss possible
2) Atleast once: At least once msg delivered, duplicates possible
3) Exactly once: No loss/duplicates

CoAP: Constrained Application Protocol.
used for Machine to Machine communication.

Characteristics:
1) Lightweight: Less overhead
2) CoAP follows RESTful architecture so it will be easy to integrate with web
3) It uses UDP
4) Uses Request Response model
5) Supports asynchronous communication

working



CoAP uses HTTP methods like GET, POST, PUT, DELETE to interact with resources.

CoAP resources are identified by URL.

CoAP message contains header (message ID, type), payload.

→ CoAP defines 4 messages

1) CON : Confirmable ; Requires acknowledgment from receiver.
   Message will be retransmitted to ensure reliable delivery
2) NON : Non confirmable ; No need of ack
3) ACK : acknowledgment ; The message sent by receiver. Contains message ID
4) RST : Reset ; If a message cant be processed then it is sent.

Cloud9

Hybrid cloud : Combination of public, private cloud. Organizations use this.

Public cloud offers scalability & cost effective
private cloud offers security → It is for you only

Federated cloud : Extension of hybrid cloud.
Multiple private & public clouds are connected.
If one cloud is not working, you can work in other because data is made redundant.
Ideal for IOT applications

Edge / Fog Cloud:

Edge / Fog computing is like having mini computers closer to sensors. Instead of sending data to clouds, it is processed where generated

→ Low latency, Bandwidth usage less, Data stays closer to source so more secured.

→ Prefered for IoT apps which need fast response.

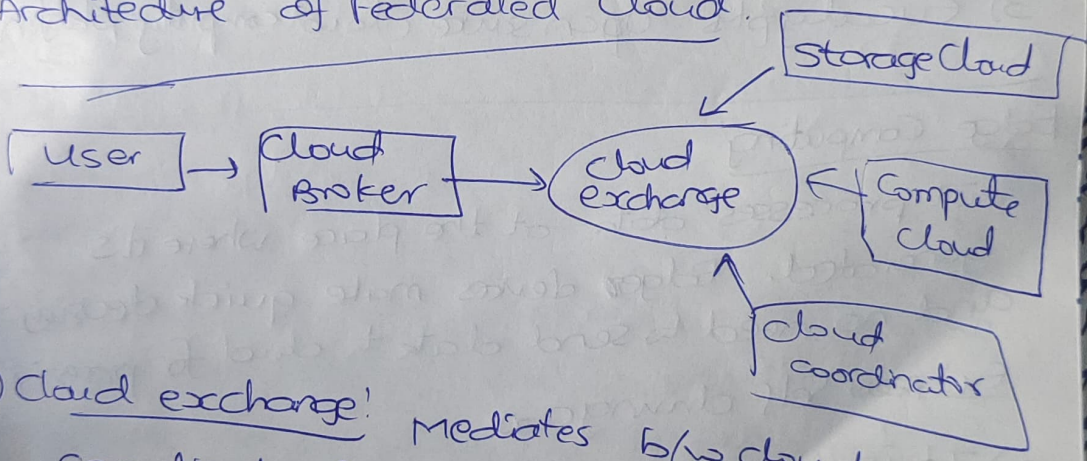
Software defined cloud:

---

1) Federated Cloud:

Cloud Federation: Integration / Collaboration of multiple clouds to meet busines requirements

Architecture of Federated Cloud:



i) Cloud exchange: Mediates b/w cloud coordinator & broker.
Provides services asked by broker.
Maintains info about cloud provider

ii) Cloud coordinator: Assigns resources based on user credits & QOS.
Manages cloud enterprises

iii) Cloud broker: Interacts with cloud coordinator.
Finalizes suitable deals for clients based on available resources

Properties: 1) Users can interact via broker or directly

2) Cloud can serve Commercial & non commercial sector

3) Allows clouds to share resources

4) Allows connectivity around the globe

5) Gives best services.

Benefits: 1) Less energy 2) Reliability

3) Cost, Time saving 4) Global connectivity

Technologies:

1) Open Nebula: Cloud management platform

2) Aneka : Middleware

3) Eucalyptus: open source cloud computing platform

Edge Computing:

It processes data at the place where it's generated. Edge devices make quick decisions and no need to send data to cloud for processing

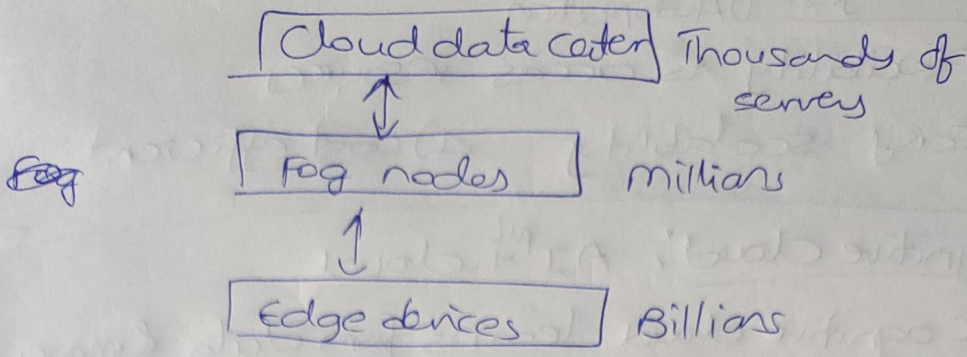
Ex: Self driving car.

→ It is less dependent on cloud.

Fog Computing:

It acts as middle layer between edge devices and cloud. It sends only important info to cloud. This saves a lot of space in cloud. It can handle millions of nodes. It requires high Bandwidth

Fog nodes analyses info & send only imp info to cloud



Edge computing	Fog computing
1) Less scable	1) More scalable
2) Billions of nodes presed	2) Millions
3) It is a sub division of fog computing	3) It is sub divisian of cloud computing
4) Less Bandwith req	4) More bandwidth
5) High privacy	5) Less privacy
6) Less power needed	6) More power need.
7) Operational cost is high	7) Low

(SDC)

Software Defined Cloud: A cloud computing model where resources are managed using softwar. No hardware used for configuration.

They use software applications to manage network, computing power, storage.

→ SDC makes cloud efficient.

→ we can scale up/down without configuring hardwar

working:

1) Software layer; It works on top of hardware & Controls cloud resources

2) SDC can also be used as hybrid cloud to meet our needs

Advantages: 1) Automation
2) Flexibility to access resources
3) Easy scalability 4) Cost efficient

Cognitive Cloud: AI^{ML} + cloud
Cognitive cloud learns from data & improves used for predictive analysis.

working:
1) Using AIML, firstly patterns are identified
2) They detect future needs, current trends & fix problems before they occur.
3) They help organization save time, reduce errors & improve performance

---

AWS for IOT: It provides scalable, flexible solutions to connect, manage, analyze IOT devices & data. Provides scalability.

AWS IOT Core: Connects devices to the cloud

AWS IOT greengrass: Allow IOT to edge devices & allows local processing

AWS IOT analytics: Allows processing, analysis of IOT data

AWS IOT device management: Simplifies management of IOT devices

AWS allows us to integrate to lambda, dynamo DB, etc easily

Things Speak: Provided by Mathworks. collects, visualize, analyze data. Provides API's. Integrated with MATLAB. Easy to integrate with REST APIs.