IOT Theory Assignment

Khush Purohit
22UCSE4013
CSE VIII Sem

Q1 Explain Open IOT Architecture.

Open IOT Architecture is a modular and scalable framework that facilitates the integration and communication of diverse IOT systems and devices. It enhances interoperability, flexibility and ease of development across multiple platforms

Layers of Open IOT Architecture:

1) Perception Layer (Device Layer):
- Acts as the sensory part of the system
- Includes sensors and actuators to gather physical parameters.
- Converts physical signals into digital data.

2) Network Layer:
- Responsible for transmitting data from perception layer to middleware / cloud.
- Uses technologies like WiFi, Zigbee, 6lo, LTE, etc
- Ensures secure & effective data transmission

3) Middleware Layer
- Acts as a bridge between hardware & applications
- Handles device management, data filtering, aggregation and service orchestration.
- Provides APIs for developers

4) Application Layer
- Offers user specific services such as smart homes

smart cities, industrial automation.
- Interfaces with user through dashboards apps or websites

5) Business layer:
- Analyze data and operations to derive business models and strategies.
- Manages overall system performance, policies & business goals.

Q2 Write short note on various sensors & actuators used in IOT:

In an IOT system, sensors & actuators are the crucial components interact directly with the physical environment. The act as systems's 'senses' & 'hands'.

Sensors: Sensors detect & measure physical phenomena like temperature, light, motion, pressure etc) from the surrounding environment.

They convert these physical inputs into electrical signals (data) that can be processed by IOT systems. They provide raw information about the state of physical world.

- Temperature & Humidity Sensor: Monitor environmental conditions. Used in weather stations, smart homes, HVAC systems, cold chain monitoring.
- Motion Sensor: (PIR) Detect movement. Used in security systems, automatic lighting.

- Light Sensors (LDR/ Photodiodes): Measure ambient light levels. Used in automatic lighting.

- Proximity Sensors: Detect the presence of nearby objects without physical contact. Used in Parking assistance.

- Accelerometers & Gyroscopes: Measures acceleration & orientation. Used in wearables, smartphones & drones.

- Gas & Air Quality Sensors: Detect specific gas (CO, CO_2) or particulate matter. Used for safety & environmental monitoring

- GPS Modules: Determine demographic location.

II. Actuators: these recieve commands/ electric signals from IOT systems & puform a physical action in the environment.

- They convert electrical energy back into physical energy, allowing IOT system to control or manipulate it's surrounding based on the sensor data and Programmed logic.

- Relays & Switches: Turn electrical circuits on or off. Used to control lights, appliances, motors.

- Motors (DC, Servo): Create rotational or linear motion. Used in robotics, automated valves, smart locks, robotic arms.

- LEDs & Displays: Provide visual feedback or

information. Used as status indicators in smart light information panels.

- Buzzers & Speakers: Produce sound for alerts or communication. Used in alarm systems, notification devices.

Q3 Compare HTTP with MQTT for IoT applications.

HTTP (Hypertext transfer Protocol and MQTT (Message Queing Telemetry Transport) are two common protocols used in IoT, but they serve different Purposes & have distinct characteristics.

Feature	HTTP	MQTT	IoT Suitability
Communication model	Request / Response	Publish / Subscribe (Pub / Sub)	MQTT's Pub/Sub is better for pushing data from sensors or commands to devices. HTTP requires the device to initiate the request.
Overhead	High (Text-based large headers)	Very low (Binary, minimal 2 byte header)	MQTT's low overhead is crucial for constrained devices (low power, bandwidth, memory).
Transport layer	Typically TCP	Typically TCP (can run over UDP also)	Both offer TCP for reliability, but MQTT's design minimizes TCP overhead impact.
State	Stateless (each req. independent)	Stateful (persistent session possible)	MQTT's statefulness allows broker to know device status & manage session efficiently.

Reliability (QoS)	Relies on TCP, no built in app level QoS	Built in QoS levels (0,1,2)	MQTT provides application level guarantees for message delivery vital for critical data.
Data Format	Any (JSON, text, binary, XML)	Agnostic (Payload is binary data)	Both are flexible but MQTT's focus is on efficient transport, no D.S.
Scalability	Scales well for web less for IoT	Excellent broker handles many connections)	MQTT brokers are designed to handle thousands of concurrent device connections.
Network Use	Assumes reliable n/w, less tolerant	Designed for unreliable, high latency n/w.	MQTT excels in typical IoT network conditions. Handles disconnections better.
Directionality	Client → server	Bidirectional (via broker)	MQTT easily supports server to device messages without complex workarounds like HTTP long polling.

Qy Write short note on AMQP & CoAP.

AMQP (Advanced Messaging Queuing Protocol)

AMQP is an Open standard, application layer Protocol for asynchronous, reliable messaging using message oriented middleware. It originated in the financial industry, emphasizing interoperability & robustness.

Reliability: Offers strong delivery guarantees & transaction supp.

Queuing: Based on message queues, exchanges & bindings.

Flexibility: Supports various messaging ~~Platform~~ patterns

like Publish subscribe, Point to Point request reply

Interoperability: Designed to work between different vendor broker & client libraries.

Security: Integrates well with SASL & TLS for authentication & encryption.

While more feature rich & potentially heavier than MQTT, AMQP is often used in backend of large scale IoT platforms. It excels at reliably routing messages between different cloud services, handling data processing pipelines, and integrating IoT data with enterprise systems. It's less common directly on highly constrained devices due to its complexity compared to MQTT / CoAP.

CoAP (Constrained Application Protocol)

CoAP is a specialized web transfer protocol designed explicitly for resource constrained devices & networks. It aims to provide restful (HTTPs) interaction capabilities in these environments.

Light weight: Very small 4 byte binary header & simple message format

UDP-based: Typically runs over UDP, reducing connection Overhead compared to TCP. Includes mechanism for optional reliability over UDP.

RESTful Model: Uses methods like GET, POST, PUT, DELETE, similar to HTTP, making it easy to

integrate with webservices via proxies

Asynchronous : Supports asynchronous message exchanges

- Built in support for subscribing to resource changes
- Resource Discovery : Built in mechanisms for devices to discover available resources on a server.

CoAP is a direct fit for communication involving constrained nodes, especially over UDP-based networks. It acts as a light weight alternative to HTTP for device-to-server interactions, resource monitoring, and simple command execution in environments where MQTT might also be considered, but where a RESTful, resource-oriented model is preferred.

Q5 Explain Q.S, Q.E and XaaS.

Q.S (Quality of Service)

This focuses on the technical performance of service or network.

A set of technologies & measurement parameters used to manage network resources & guarantee a certain level of performance for data transmission. It deals with objectives, measurable characteristics.

The purpose of this is to prioritize traffic, ensure reliability for critical applications, and meet Service Level Agreement (SLAs). In essence it's about how well the network delivers the data packets.

QoE (Quality of Experience)
This focuses on the user's subjective perception & satisfaction with a service or application.

It is a measure of the overall delight or annoyance a user experiences when using a technology or service. It's user centric & subjective.

While influenced by technical QoS metrics, It also includes factors like ease of use, Content quality, device capabilities, user expectations & context.

The purpose of this is to understand & optimise the end-user's overall satisfaction which is the ultimate goal of most services

XAAS : (Anything / Everything as a service)
This is an umbrella term representing the broad category of services delivered over the internet rather than being provided locally or on premises. It signifies making virtually any IT function, resource or product available on demand typically with a pay as you go model.

It provides scalability, flexibility potentially lower costs & reduce management overhead by abstracting underlying ~~benefits~~ infrastructure or platforms.

This includes:

- SAAS : (Software as a Service) On demand software
  PAAS : (Platform as a Service) On demand platform for developing ~~Computing infrastructure~~
  IAAS : (Infrastructure as a Service) On demand computing infrastructure.

Q6 Design an IoT system to send temperature, Humidity & moisture data values to a server.

1 Sensor Selection :- Use temperature sensors like DHT11 or DHT22 to measure temperature and humidity.
- Incorporate a soil moisture sensor to measure moisture levels

2 Microcontroller or IoT Devices : Employ a microcontroller such as an Arduino or an ESP 32 to read data from the sensors. These devices are capable of handling multiple sensors and provide connectivity options.

3 Communication Protocol : Choose a light weight communication protocol such as MQTT or HTTP to send data to the server. MQTT is often preferred for IoT due to its efficiency in transmitting small data packets.

4 Connectivity : Establish a connection to the internet using WiFi or any other connectivity modules.

5 Server Integration : Set up a server to receive, store & analyze the transmitted data. This could be a cloud based IoT platform like Thingspeak.

6 Data transmission : Program the microcontroller to read data from the sensors at regular intervals & send it to the server over the chosen protocols.

7 Power Management : Ensure an appropriate power supply for the entire setup, considering battery or solar powered solutions for remote deployments.

Q1 Design an IOT system to connect home automation devices with mobile phone.

1 Home Automation Devices: These could include smart lights, thermostats, security cameras, smart plugs etc. equipped with sensors & actuators to execute tasks.

2 Communication Protocols: Implement communication protocols like MQTT or CoAP to enable efficient and reliable data exchanges between devices & mobile phones. MQTT is widely used due to its lightweight nature.

3 IoT Gateway: Use an IoT Gateway to connect multiple home devices & manage communication with the mobile phone. The gateway acts as a bridge between local devices & internet.

4 Connectivity Options: Utilize wifi or bluetooth connectivity to establish communication between home devices & the gateway. For remote access, devices and the gateway.

5 Mobile Application: Develop a mobile app which provides user with an intuitive interface to monitor & control devices. The app can display real time data, send commands, and receive alerts.

6 Cloud Integration: Integrate a cloud platform to store data, process commands & enable remote access to home automation devices via the mobile phone.

Q8 For a low cost IoT device develop a "Day time Client" Program.

```
# include <NTPClient.h>
# include <WifiUdp.h>
# include <ESP8266Wifi.h>

const char * ssid = " WiFi ID" ;
const char * password = " Wifi Password" ;

WiFiUDP ntpUDP ;
NTPClient timeClient (ntpUDP, "pool.ntp.org", 19800, 60000 );

void setup () {
    Serial.begin (115200 );
    Wifi.begin (ssid, password);
    while (Wifi.status () != WL_CONNECTED ) {
        delay (1000);
        Serial.println (" Connecting to wifi ... " );
    }
    Serial.println (" Connected to Wifi " );
    timeClient.begin ();
}

void loop () {
    timeClient.update ();
    Serial.println (timeClient.getformattedTime ());
    delay (1000);
}
```

Q9 Write short note on IoT Analytics server (Thingspeak).

An IoT Analytics server like Thingspeak is a platform designed to aggregate, visualize & analyze live data stream from IoT devices.

1 Data Collection: Thingspeak allows IoT devices to send sensor data to the cloud using protocols like MQTT.

2 Visualization: The platform provides tools to create real time visualization of the collected data, enabling users to monitor trends and patterns effectively.

3 Analytics: Thingspeak integrates MATLAB for advanced data analytics, allowing users to process and analyze data directly on the platform.

4 Applications:
It is widely used for ~~the~~ environmental monitoring, energy management, smart farming and other IoT applications.

5 Ease of Use: Thingspeak simplifies IoT system development by eliminating the need for setting up server or developing custom web software.

Q10 Difference between HTTP & MQTT for IoT applications.

HTTP (Hyper Text Transfer Protocol): This is a request response protocol ~~widely~~ used widely on the web. It consumes more bandwidth and resources since it involves headers and a relatively larger overhead. This makes it less ideal for constrained devices or

networks. It is based on client-server model, where the client actively requests data from the server. This is suitable for web applications where real time communication is not critical.

This relies on TCP for reliability but lacks built in features for message delivery guarantees.

MQTT (Message Queuing Telemetry Transport):

This is a lightweight, publish-subscribe protocol designed specifically for IoT and low bandwidth scenarios. This is optimized for a minimal bandwidth usage, making it suitable for low powered IoT devices. This uses a publish-subscribe model where devices communicate through a broker enabling better scalability and decoupling.

- This is ideal for IoT applications requiring real time data transfer and low latency communication.
- This provides Quality of Service (QoS) levels to ensure reliable message delivery according to application needs.

Q How can you make IoT client devices much safer.

1 Encryption:

Use strong encryption protocols like AES (Advanced Encryption Standard) to protect data in transit and at rest.

2 Authentication: Implement strong authentication mechanisms, such as two-factor authentication (2FA) or device specific credentials to prevent unauthorized access.

3) Secure Firmware Updates: Ensure firmware updates are transmitted over secure channels and verify updates with digital signatures to avoid tampering.

4) Firewall Implementation: Equip IoT devices with firewalls to monitor and filter incoming and outgoing traffic based on security rules.

5) Access Control: Limit access to IoT devices & their data to authorized users only: Role based access control can be helpful (RBAC).

6) Secure communication Protocols: Adopt secure communication Protocols like TLS (Transport Layer Security) or HTTPS for safe data exchange.

7) Physical Security: Protect IoT devices from physical access or tampering, espacially in sensitive environments.

Q12 What are the different types of sensors in IoT?

(1) Temperature Sensors: Measures environmental or object temperatures. Commonly used in weather monitoring systems and smart thermostats.
Exp DHT 11, LM 35.

(2) Humidity Sensors: Detect moisture levels in the air. Frequently applied in environmental & HVAC (Heating, Ventilation & Air Conditioning) systems Exp DHT 22

(3) Light Sensors: Detect light intensity. Widely used in smart lightning systems & camera modules.
Exp LDR

④ Proximity Sensor Measures the distance of objects without physical contact. Used in parking systems & industrial robots. Exp: Ultrasonic Sensor.

⑤ Gas Sensor:
Monitor levels of gases like CO_2 or methane Applied in air quality monitoring & industrial safety systems.

⑥ Sound Sensor:
Captures sound waves. Used in voice controlled applications and noise monitoring.
Exp Microphones.

⑦ Location Sensor:
Provide location data using GPS (Global Positioning Systems). Key for logistics, navigation and asset tracking
Exp GPS module.

⑧ Moisture Sensor:
Detect soil moisture levels. Essential for smart farming & irrigation management.
Exp: soil moisture sensor.