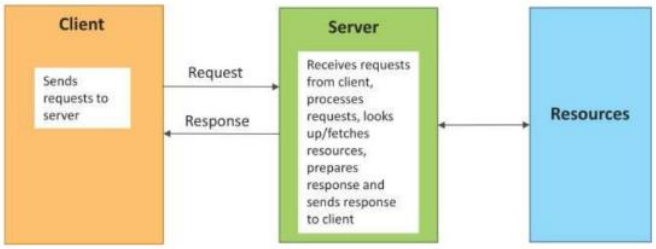
Explain IOT communication models?

The **Internet of Things (IoT)** refers to a network of physical devices, vehicles, appliances, and other objects embedded with sensors, software, and connectivity, allowing them to collect and exchange data. Communication is a fundamental component of the IoT, as it enables devices to share information with each other and with centralized systems (such as cloud servers or data centers).

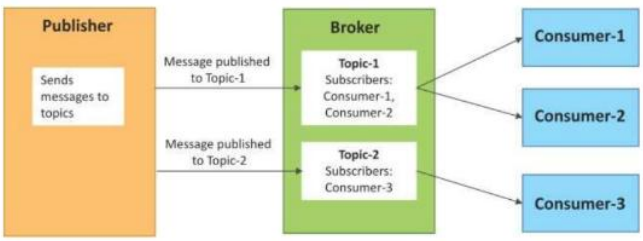
Request–Response Communication Model

- Request–Response is a 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 representations, prepares the response and then sends the response to the client.
- Stateless communication model



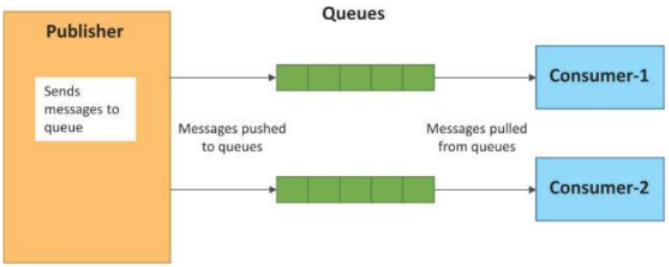
Publish–Subscribe Communication Model

- 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 receives data for a topic from the publisher, it sends the data to all the subscribed consumers.



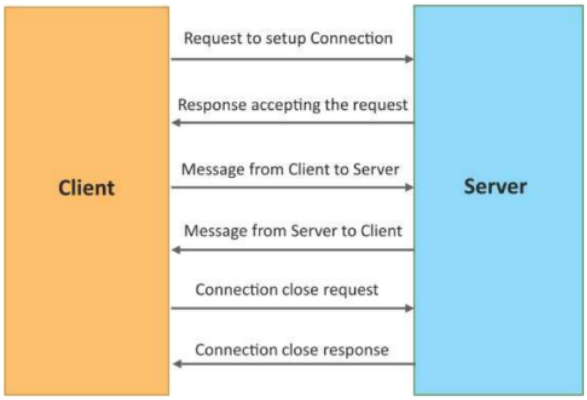
Push–Pull Communication Model

- 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 at which the consumers pull data.



Exclusive Pair Communication Model

- Exclusive Pair is a bidirectional, fully duplex communication model that uses a persistent connection between the client and the server.
- Once the connection is set up 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.



Explain ETSI M2M High Level architecture.

ETSI M2M (Machine-to-Machine) interfaces are part of the ETSI M2M architecture, which was developed by the

European Telecommunications Standards Institute (ETSI) to standardize communication between machines in IoT

(Internet of Things) environments. The ETSI M2M architecture laid the foundation for what later evolved into oneM2M, a

global IoT standard.

ETSI M2M Interfaces

ETSI M2M defines several key interfaces to enable communication between different system components:

1. mIa (M2M Service Capabilities - Application Interface)

○ This interface enables communication between M2M applications and the M2M service capabilities layer.

○ It provides functions like data management, security, and device management.

○ Example: An IoT application accessing an M2M platform to retrieve sensor data.

2. dIa (Device Application Interface)

○ This interface connects applications running on an M2M device with the M2M service capabilities within the same

device.

○ It is used for local communication between embedded applications and the device's M2M platform.

○ Example: A temperature sensor application communicating with the local M2M software.

3. mId (M2M Service Capabilities - Network Interface)

○ This interface connects the M2M Service Capabilities Layer with core network services (e.g., operator networks).

○ It provides access to network functions such as authentication, billing, and location services.

○ Example: An M2M service interacting with a mobile network to authenticate an IoT device.

ETSI M2M resource management

Resources:

● Represented as data structures like devices, sensors, or applications.

● Identified using Uniform Resource Identifiers (URIs).

Resource Tree:

● Organized hierarchically like a file system.

● Parent-child relationships between resources.

Resource Types:

● Containers: Store data.

● Content Instances: Data entries in containers.

● Subscriptions: Notify users or applications of resource changes.

● Access Control Policies (ACP): Define access rights.

Resource CRUD Operations:

● Create: Add a new resource.

● Retrieve: Fetch resource data.

● Update: Modify resource attributes.

● Delete: Remove resources.

Discovery:

● Mechanism to search for available resources.

Security:

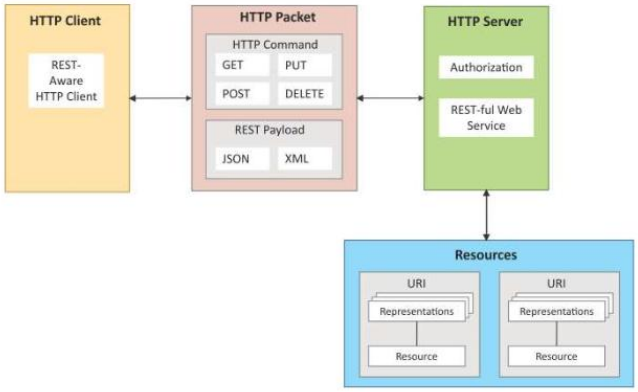
● Access control to ensure only authorized users or applications manage resources.

What is IOT? Describe communication API's in detail.

The **Internet of Things (IoT)** refers to the network of physical devices, vehicles, appliances, and other objects that are embedded with sensors, software, and connectivity, allowing them to collect and exchange data over the internet. These "smart" devices communicate with each other and with centralized systems, enabling automation, real-time monitoring, and decision-making without requiring constant human intervention. The IoT has a wide range of applications across industries, from smart homes and healthcare to transportation, agriculture, and manufacturing.

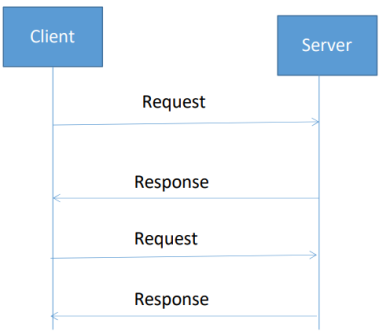
REST-based Communication APIs

- Representational State Transfer (REST) is a set of architectural principles by which you can **design web services and web APIs** that focus on a system's resources and how resource states are addressed and transferred.
- REST APIs **follow the request–response communication model.**
- REST architectural constraints apply to the components, connectors and data elements within a distributed hypermedia system.



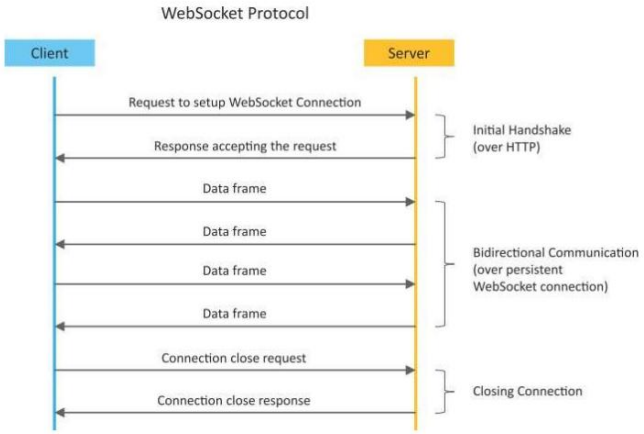
REST-based Communication APIs Constraints

- **Client – Server**
- **Stateless**
- **Cacheable**
- **Layered System**
- **Uniform Interface**
- **Code on demand**



WebSocket-based Communication APIs

- WebSocket APIs **allow bi-directional, full duplex communication** between clients and servers.
- WebSocket APIs follow the **exclusive pair communication model**.



Difference between REST and WebSocket-based Communication APIs

Comparison Based on	REST	Websocket
State	Stateless	Stateful
Directional	Unidirectional	Bidirectional
Req-Res/Full Duplex	Follow Request Response Model	Exclusive Pair Model
TCP Connections	Each HTTP request involves setting up a new TCP Connection	Involves a single TCP Connection for all requests
Header Overhead	Each request carries HTTP Headers, hence not suitable for real-time	Does not involve overhead of headers.
Scalability	Both horizontal and vertical are easier	Only Vertical is easier

List and explain IoT enabling technologies.

IoT Enabling Technologies

The **Internet of Things (IoT)** involves a wide range of technologies that enable devices to collect, exchange, and process data. These enabling technologies are essential for building and scaling IoT systems. Below are some of the key enabling technologies for IoT:

---

1. Wireless Sensor Networks (WSN)

Definition:

A **Wireless Sensor Network (WSN)** is a network of spatially distributed sensors that collect data from the environment and communicate wirelessly. These sensors can monitor parameters like temperature, humidity, pressure, light, motion, and other physical attributes. The data collected by these sensors is typically sent to a central system or cloud for processing and analysis.

Key Features:

- **Sensors**: Devices that detect and measure physical parameters (temperature, motion, pressure, etc.).
- **Wireless Communication**: Sensors in the WSN communicate with each other and with the central system using wireless protocols like Wi-Fi, Zigbee, Bluetooth, or LoRa.
- **Low Power Consumption**: Many WSN devices are designed to work with low power to ensure long operational lifetimes.
- **Distributed Network**: The sensors work autonomously and are distributed in an area, providing real-time data collection without the need for manual intervention.

Role in IoT:

WSNs form the core infrastructure for many IoT systems. They are crucial for applications where real-time monitoring is needed, such as environmental monitoring, smart cities, industrial automation, and healthcare (e.g., wearable health sensors).

Use Case Example:

In **smart agriculture**, WSNs are deployed in farms to monitor soil moisture levels, weather conditions, and crop health, providing farmers with real-time data to optimize irrigation and farming practices.

---

2. Cloud Computing

Definition:

Cloud computing refers to the delivery of computing services like storage, processing power, and software over the internet, rather than on local servers or personal devices. IoT devices often require vast amounts of storage and computing resources, which cloud computing provides.

Key Features:

- **Scalability**: Cloud platforms allow IoT systems to scale easily by providing on-demand resources based on data requirements.
- **Remote Access**: Cloud systems can be accessed from anywhere in the world, enabling remote monitoring and control of IoT devices.
- **Data Storage and Processing**: Cloud computing allows for the storage and processing of large amounts of data generated by IoT devices. It provides infrastructure for data analytics, machine learning, and AI processing.
- **Cost-Effectiveness**: Cloud computing reduces the need for on-premise infrastructure, reducing costs and complexity for IoT applications.

Role in IoT:

Cloud computing enables the collection, processing, and analysis of data from IoT devices. It provides a platform to store large volumes of data and supports advanced analytics and machine learning models to derive insights from the data.

Use Case Example:

In **smart homes**, cloud platforms collect data from various devices (e.g., thermostats, security cameras, smart lighting) and analyze usage patterns to automate home settings and provide remote access through mobile apps.

---

3. Big Data Analytics

Definition:

Big Data Analytics refers to the use of advanced analytics techniques to analyze large, complex datasets that are too big to be processed by traditional data-processing tools. IoT systems generate vast amounts of data, and big data analytics helps to process and extract meaningful insights from this data.

Key Features:

- **Data Volume**: IoT systems generate massive amounts of data in real-time, which big data analytics can handle.
- **Data Variety**: IoT data comes from various sources and in various formats, including text, images, video, and sensor data. Big data analytics tools can process these diverse types of data.
- **Real-Time Analysis**: Big data analytics can process data as it is generated, allowing for real-time decision-making and automation.
- **Predictive Analytics**: Big data analytics can use historical data to predict trends, behavior, and future events.

Role in IoT:

Big data analytics plays a crucial role in IoT by processing the enormous volumes of data generated by IoT devices. It enables actionable insights, predictions, and optimizations across industries.

Use Case Example:

In **smart cities**, sensors and devices collect data on traffic flow, pollution, and energy usage. Big data analytics processes this data to optimize traffic signals, reduce energy consumption, and predict future infrastructure needs.

---

4. Embedded Systems

Definition:

An **embedded system** is a specialized computer designed to perform a specific task within a larger system, often with real-time constraints. Embedded systems are typically integrated into IoT devices to handle tasks such as data collection, processing, and communication.

Key Features:

- **Real-Time Operation**: Embedded systems are designed for real-time processing, meaning they can respond to events or inputs within a specific time frame.
- **Dedicated Purpose**: Unlike general-purpose computers, embedded systems are designed to perform a specific function, such as controlling a thermostat or monitoring a sensor.
- **Low Power Consumption**: Many embedded systems are designed to be energy-efficient, which is essential for battery-powered IoT devices.
- **Connectivity**: Embedded systems are typically equipped with communication interfaces like Wi-Fi, Bluetooth, Zigbee, or cellular networks to enable data exchange.

Role in IoT:

Embedded systems are at the heart of most IoT devices. They manage the sensors, control device operations, and facilitate communication with other devices or cloud platforms. These systems enable the functionality of IoT devices by integrating hardware and software into compact, efficient units.

Use Case Example:

In **wearable health devices**, embedded systems manage the collection of data from sensors that monitor heart rate, steps, and other vital signs. The embedded system processes the data locally and communicates with a smartphone app for analysis.

Summary Table of IoT Enabling Technologies

Technology	Definition	Key Features	Role in IoT	Use Case Example
Wireless Sensor	A network of spatially distributed sensors for	Wireless communication, low	Collects real-time data from the environment,	Smart agriculture: Monitoring soil

Technology	Definition	Key Features	Role in IoT	Use Case Example
Network (WSN)	monitoring environments.	power, distributed network.	crucial for monitoring.	and crop conditions.
Cloud Computing	Delivery of computing services over the internet.	Scalability, remote access, data storage, cost-effectiveness.	Processes, stores, and analyzes large amounts of IoT data.	Smart homes: Remote control of devices through the cloud.
Big Data Analytics	Analysis of large, complex datasets to extract meaningful insights.	Data volume, real-time processing, predictive analytics.	Analyzes IoT data to derive insights and make predictions.	Smart cities: Traffic optimization and energy consumption.
Embedded Systems	Specialized computer systems for specific tasks.	Real-time operation, low power, connectivity, dedicated purpose.	Drives the functionality of IoT devices through hardware and software.	Wearables: Health monitoring devices like fitness trackers.

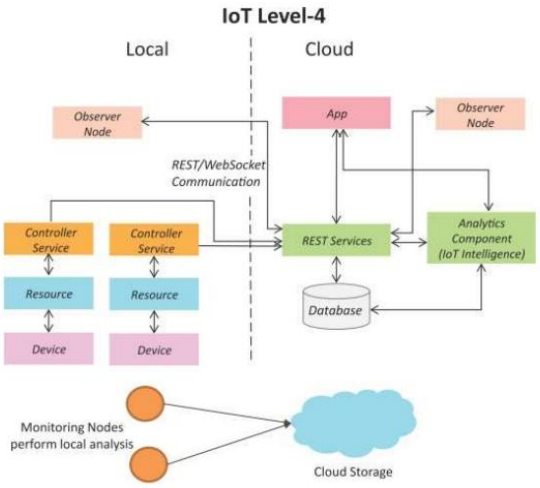
Conclusion

The IoT ecosystem relies heavily on enabling technologies like **Wireless Sensor Networks**, **Cloud Computing**, **Big Data Analytics**, and **Embedded Systems** to collect, store, analyze, and act upon data. These technologies make it possible to deploy scalable, efficient, and intelligent IoT systems across a wide range of industries, from healthcare and agriculture to smart homes and cities.

Explain IoT deployment levels 4 to 6 in detail.

IoT Level-4

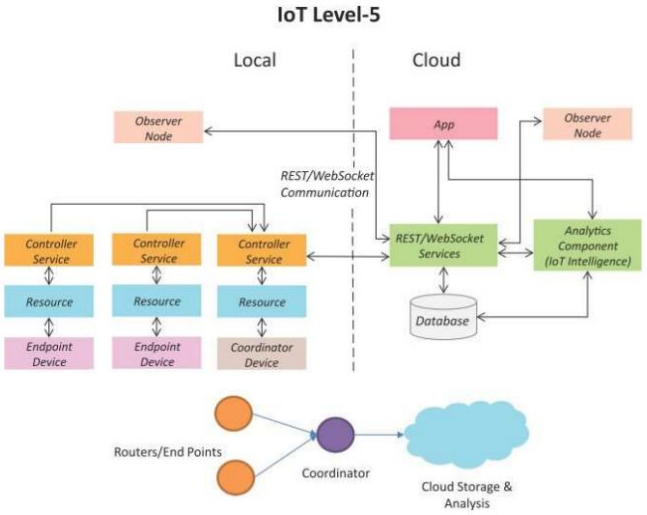
- A level-4 IoT system has multiple nodes that perform local analysis. Data is stored in the cloud and the application is cloud-based.
- Level-4 contains local and cloud-based observer nodes which can subscribe to and receive information collected in the cloud from IoT devices.
- Level-4 IoT systems are suitable for solutions where multiple nodes are required, the data involved is big and the analysis requirements are computationally intensive.



IoT – Level 4 Example …Tracking Package Handling

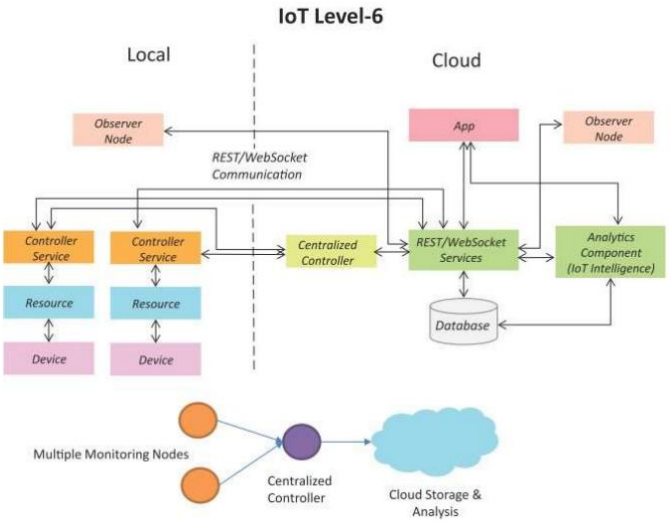
IoT Level-5

- A level-5 IoT system has multiple end nodes and one coordinator node.
- The end nodes perform sensing and/or actuation.
- The coordinator node collects data from the end nodes and sends it to the cloud.
- Data is stored and analyzed in the cloud and the application is cloud-based.
- Level-5 IoT systems are suitable for solutions based on wireless sensor networks, in which the data involved is big and the analysis requirements are computationally intensive.



IoT Level-6

- A level-6 IoT system has multiple independent end nodes that perform sensing and/or actuation and send data to the cloud.
- Data is stored in the cloud and the application is cloud-based.
- The analytics component analyzes the data and stores the results in the cloud database.
- The results are visualized with the cloud-based application.
- The centralized controller is aware of the status of all the end nodes and sends control commands to the nodes.



Illustrate unified data standards in IoT.

Unified Data Standards in IoT for data Exchange

Common set of standards for internet of things.

A message translation to be replaced with a unified message structure to communicate among embedded, gateway, and cloud machines.

IoT landscape is very vast in nature. It comprises industrial, energy. agriculture, healthcare, supply chain, automotive, smart city and many more.

So, it would be difficult to create a unified standard for the industry as a whole.

Solutions for unified data

1. The Resource Description Framework (RDF)

as a general method for conceptual description or modeling of information that is implemented in web resources, using a variety of syntax formats.

It could be a metadata model for WoT applications.

2. SOAP and REST frameworks can be used to provide data exchange protocols for Iot applications.

SOAP stands for Simple Object Access Protocol....a messaging protocol for data transfer between clients and servers.

Solutions for unified data.....

3. M2M/IoT protocol stack, a unified loT data format and protocol standards proposed for e-commerce or e-business.

4. EDI (Electronic data interchange) describes the rigorously standardized format of electronic documents.

The EDI standards were designed to be independent of communication and software technologies.

EDI can be transmitted using any methodology agreed to by the sender and recipient.

Solutions for unified data.....

5. ebXML:

some borrowed ideas from both EDI and XML.

to build an interoperable e-commerce infrastructure.

In a computer system, ebXML specifies the business rules for how two different systems talk to each other.

XML or ebXML coexists with the popular web formatting language HTML.

(HTML tells us how the data should look, but XML tells us what it means.)

Analyze the efforts for protocol standardization in IoT.

Explain the architecture of OGC in detail with respect to IoT.

Open Geospatial Consortium architecture

The Open Geospatial Consortium (OGC) is an organization made up of companies, governments, and universities that create standards for geospatial data and services. This means they develop guidelines to help make geographic information available and usable on the web, mobile apps, and other location-based services.

OGC helps systems and devices that collect geospatial data (like sensors) communicate and share information.

For example, if a sensor is monitoring weather conditions or traffic, the OGC helps ensure that the data from those sensors can be easily discovered, understood, and used by other systems or applications.

Key points of OGC's work:

- Sensor Web Enablement (SWE): This is a part of OGC focused on making sensors more accessible and usable. It helps with tasks like:
  - Finding sensors that meet certain needs or criteria.
  - Understanding the sensor's capabilities and the quality of its measurements.
  - Getting real-time or historical data from the sensors in standard formats.
  - Asking sensors to take new measurements.
  - Subscribing to alerts from sensors when specific conditions are met (like a weather sensor alerting when it rains).

OGC SWE includes the following key standards:

- SensorML and Transducer Model Language (TML): These are descriptions (in XML format) that explain how sensors work and what they measure.
- Observations and Measurements (O&M): This standard describes how data collected from sensors should be organized and shared.
- SWE Common Data Model: This standard outlines how sensor data is structured at a low level.

● Sensor Observation Service (SOS): A service for retrieving sensor data based on certain requests or filters.

● Sensor Planning Service (SPS): A service that allows users to request sensors to take specific measurements.

● PUCK: A protocol that defines how metadata (like information about a sensor's capabilities) is retrieved.

How the system works:

● OGC uses a system architecture based on Service-Oriented Architecture (SOA).

● There is a Catalog (CAT) where descriptions of available OGC services, including sensors and sensor-related services, are stored.

● When a sensor system is installed, it retrieves a description of its capabilities (via SensorML) and registers itself in the catalog to make it discoverable.

● Sensor Observation Service (SOS) is used to manage requests and responses for sensor data.

● Applications can search the catalog to find the sensor they need, and then request data from that sensor through the SOS.

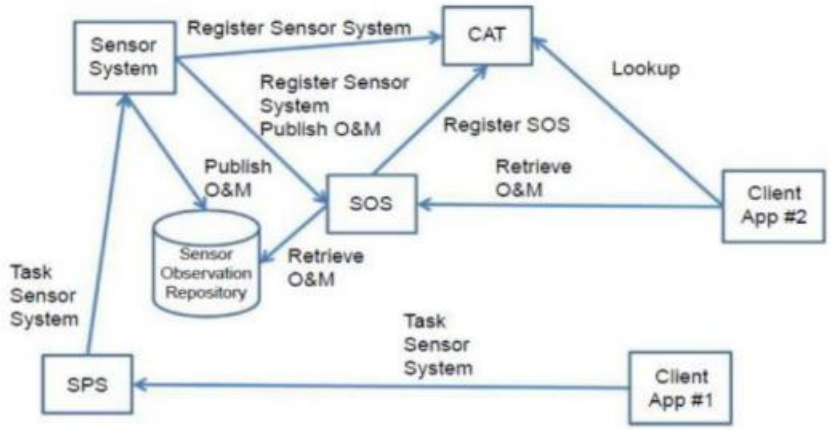
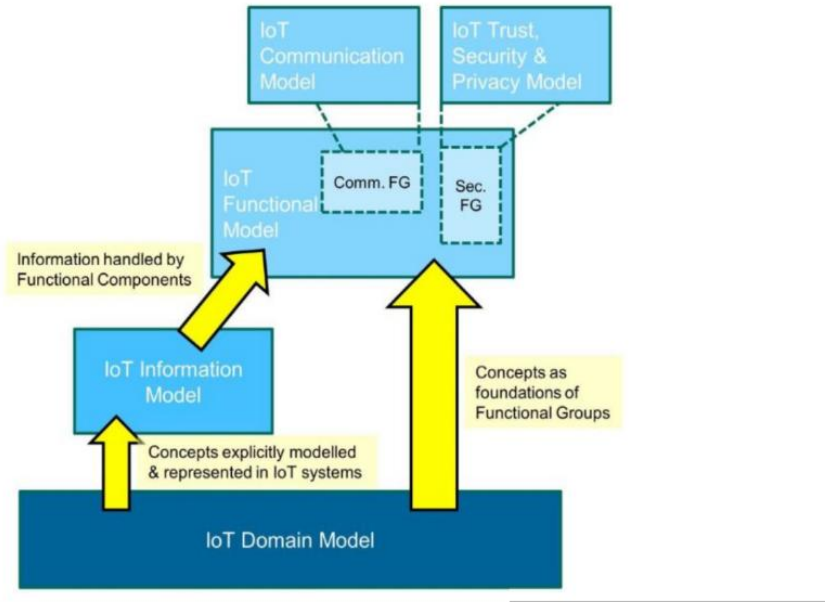


FIGURE 6.10

OGC functional architecture and interactions.

Explain the IoT Reference Model in detail.



IOT domain model

→ It captures the basic attributes of the main concepts and the relationship between these Concepts

→ Abstraction level of the IoT Domain model have been chosen in such a way that its concepts are independent of specific technologies and use cases

→The idea is that these concepts are not expected to change much over the next decades or longer

Three Kinds of Device types for the IoT Domain Model

1. Sensors

2. Actuators

3. Tags. - In general, identify the Physical Entity that they are attached to. It can be both devices or physical entities but not both, as the domain model shows.

Example: Tag as a device - Radio Frequency ID. Tag as a P.E-Paper printed immutable. barcode or Quick Response (OR) code..

IOT Information Model

Virtual entity in IoT Domain Model is in the “thing” IoT, the IoT information model captures the details of a Virtual entity centric model. Similar to the IOT domain model, the IoT information Model is presented using Unified Modelling Language (UML) diagrams.

Functional Model

It aims at describing mainly the F.G and Their interaction with the ARM, while the Functional View of a Reference Architecture describes the functional components of FG interfaces, and interactions between components. The the functional View is typically derived from the functional Model is in Conjunction with high level requirements.

Device functional Group

→The Device FG contains all the possible functionality hosted by the physical Devices that are used for increment the Physical Entities.

→ The Device functionality includes sensing actuation, processing, storage, and identification components, the sophistication of which depends on the Device capabilities.

Communication functional group

→ Comm. FG. consists abstracts all the possible Communication mechanisms used by relevant Devices in an actual syst order to transfer information The System in to the digital world components or other Devices.

Virtual Entity functional group

→It corresponds to the virtual entity class in the IOT Domain model.

It contains the necessary, manage functionality to associations between virtual Entities with themselves as well as as between VE and related IoT Services.

IoT Service Organization functional group

→Its purpose is to host all functional Components that support The composition of IoT and Virtual Entity services.

IoT Business Process Management functional group

It is a collection of functionalities that allows smooth Integration of IoT related Services with the business process.

Management functional group

It includes the necessary functions for enabling fault and performance monitoring enabling fault and performance monitor of the system, configuration for enabling The system demands; to be flexible to changing user demands.

IoT system contain communicating entities, and therefore the corresponding communication model needs to capture the communication interactions of these entities.

Security functional group:-

The Security FG contains the functional components that ensure the secure operation of the system as well as the management of privacy. The Security FG contains components for Authentication of Users (Applications, Humans), Authorisation of access to Services by Users, secure communication (ensuring integrity and confidentiality of messages) between entities of the system such as Devices, Services, Applications, and last but not least, assurance of privacy of sensitive information relating to Human Users.

Application functional group:-

The Application FG is just a placeholder that represents all the needed logic for creating an IoT application. The applications typically contain custom logic tailored to a specific domain such as a Smart Grid

Communication model

Safety

the IoT Reference Model can only provide IoT-related guidelines for ensuring a safe system to the extent possible and controllable by

a system designer. Eg: smart grid.

Privacy

Because interactions with the physical world may often include humans, protecting the User privacy is of utmost importance for an

IoT system. The IoT-A Privacy Model depends on the following functional components: Identity Management, Authentication,

Authorisation, and Trust & Reputation

Trust

Generally, an entity is said to ‘trust’ a second entity when the first entity makes the assumption that the second entity will behave

exactly as the first entity expects.”

Security

The Security Model for IoT consists of communication security that focuses mostly on the confidentiality and integrity protection of

interacting entities and functional components such as Identity Management, Authentication, Authorisation, and Trust & Reputation.