Module 1

Introduction to Internet of Things: Introduction, Physical design of IOT, Logical Design of IOT, IOT enabling technologies, IOT Levels & Deployment Templates.

Introduction to Internet of Things

1.1 Introduction

Internet of Things (IoT) comprises things that have unique identities and are connected to the internet.

- Existing devises, such as networked computers or 4G enabled mobile phones already have some form of unique identities and are also connected to the internet, the focus on IoT in the configuration, control and networking via the internet of devices or things, that are traditionally not associated with the Internet. These include devices such as thermostats, utility meters, a blue tooth- connected headset, irrigation pumps and sensor or control circuits for an electric car's engine
- Experts forecast that by the year 2020 there will be a total of 50 billion devices/ things connected to the internet.
- The scope of IoT is not limited to just connected things (Devices, appliance, machines) to the Internet.
- Applications on IoT networks extract and create information from lower-level data by filtering, processing, categorizing, condensing and contextualizing the data.
- The information obtained is then organized and structured to infer knowledge about the system and or its user, its environment and its operations and progress towards its objectives, allowing a smarter performance.

Definition and characteristics of IoT:

Definition

A dynamic global network infrastructure with self – configuring based on standard and interoperable communication protocols where physical and virtual "things" have identified, physical attributes, and virtual personalities and use intelligent interfaces, often communicate data associated with users and their environment.

Characteristics

- **Dynamic and self-Adapting:**
  IoT devices and systems may have the capability to dynamically adapt with the changing contexts and take actions based on their operating condition.Ex: Surveillance cameras can adapt their modes based on whether it is day or night.
- **Self – Configuring:**
  IoT devices may have self-Configuring capability allowing a large number of devices to work together to provide certain functionality.
- **Interoperable communication protocols:**
  IoT Devices may support a number of interoperable communication protocols and can communicate with other devices and also with the infrastructure.
- **Unique Identity:**
  Each IoT devices has a unique identity and a unique identifier. IP address, URI). IoT systems may have intelligent interfaces which adapt based on the context, allow communication with users and the environment contexts.
- **Integrated into information network:**
  IoT devices are usually integrated into the information network that allows them to communicate and exchange data with other devices and systems.

1.2 Physical Design of IoT

1.2.1 Things of IoT

The “Things” in IoT usually refers to IoT devices which have unique identities and can perform remote sensing, Actuating and monitoring capabilities. IoT devices can exchange data with other connected devices and applications (directly or indirectly), or collect data from other devices and process the data locally or send the data to Centralized servers or cloud-based applications back ends for processing the data or from some tasks locally and other task within the IoT infrastructure, based on temporal and space constraints (ie: Memory, processing calibrators, communication latencies and speed and deadlines).

Figure 1.1 shows a block diagram of a typical IoT device, An IoT device may consist of several interfaces connections to other devices, both wired and wireless. These include I) IoT interfaces for sensors II) interfaces for internet connectivity III) memory and storage interfaces IV) audio video interfaces. An IoT Device can collect various types of data from the the onboard or attached sensors, such as temperature, humidity, light intensity.
IoT devices can also be varied types, for instance, wearable sensors, smart watches, LED light automobiles and industrial machines. Almost all IoT devices generate data in Some form or the other which when processed by Data Analytics systems leads to Useful information to guide further actions locally or remotely. Figure 1.2 shows different types of IoT devices.

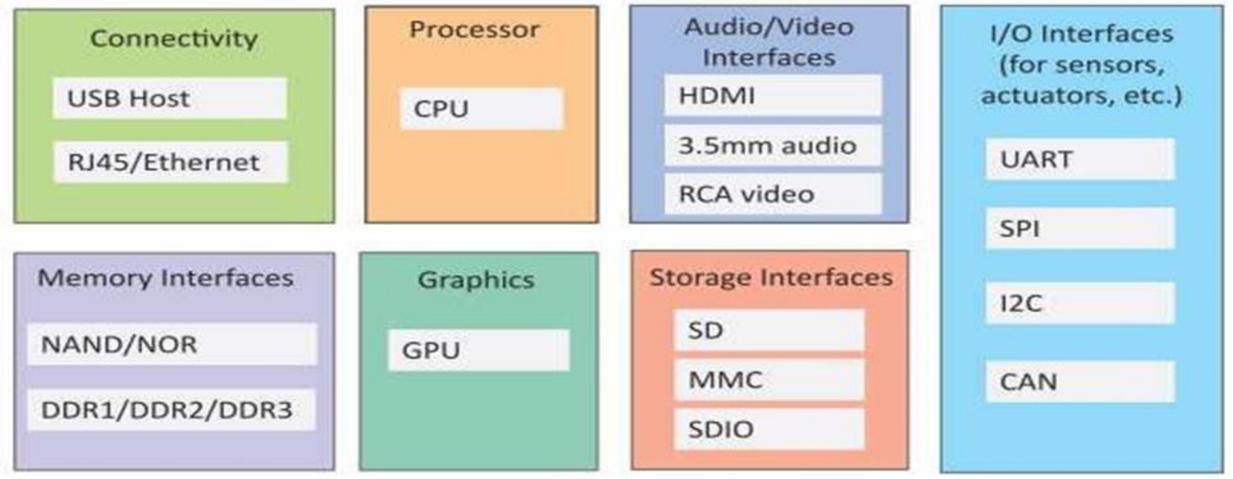


Figure 1.1 Generic block diagram of an IoT Device

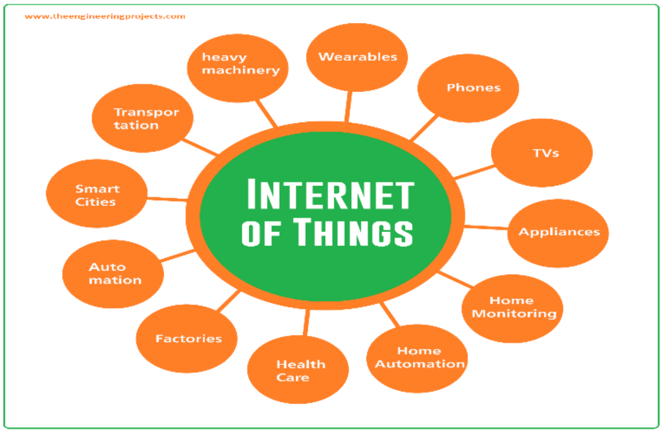


Figure 1.2 IoT devices

1.2.2 IoT Protocol

Link Layer:

Link Layer protocols determine how the data is physically sent over the networks physical layer or medium (example copper wire, electrical cable, or radio wave). The Scope of The Link Layer is the Last Local Network connections to which host is attached. Host on the same link exchange data packets over the link layer using the link layer protocol. Link layer determines how the packets are coded and signaled by the hardware device over the medium to which the host is attached.

802.3 Ethernet:

802.3 is a collection of wired Ethernet standards for the link layer. For example, 802.3 10BASE5 Ethernet that uses coaxial cable as a shared medium, 802.3.i is standard for 10 BASET Ethernet over copper twisted pair connection, Standards provide data rates from 10 Mb/s to 40 gigabits per second and the higher. The shared medium in Ethernet can be a coaxial cable, twisted pair wire or and Optical fiber. Shared medium carries the communication for all the devices on the network.

802.11- WI-FI:

IEEE 802.11 is a collection of wireless Local area network. (WLAN) communication standards, including extensive descriptions of the link layer. For example, 802.11a operate in the 5 GHz band, 802.11b and 802.11g operate in the 2.4 GHz band. 802.11ac operates in the 5G hertz band.

802.16 WiMAX:

IEEE 802.16 is a collection of wireless broadband and Standards, including extensive descriptions for the link layer also called WiMAX wimax standard provides a data rates from from 1.5 Mb/s to 1Gb/s the recent update provides data rates of hundred megabits per second for mobile station.

802.15.4 LR-WPAN:

IEEE 802.15.4 is a collection of standards for low-rate wireless personal area network (LR- WPAN). These standard forms the basis of specifications for high level communication Zigbee. LR-WPAN standards provide data rates from 40 k b/ s. These standards provide low cost and low speed Communications for power constrained devices.

2G / 3G / 4G mobile communications:

These are the different generations of mobile communication standards including second generation (2G including GSM and CDMA). 3rd Generation (3G including UMTS and CDMA2000) and 4th generation 4G including LTE.

Network / internet layer:

The network layer is responsible for sending of IP datagrams from the source network to the destination network. This layer Performs the host addressing and packet routing. The datagrams contain a source and destination address which are used to route them from the source to the destination across multiple networks. Host Identification is done using the hierarchy IP addressing schemes such as ipv4 or IPv6.

IPV4: Internet protocol versions for open parents close (IPV4) is there most deployed internet protocol that is used to identify the device is on a network using a hierarchy latest scheme. It uses 32-bit addresses scheme that allows total of 2^{32} address. As more and more devices got connected to the internet. The Ipv4 has succeeded by IPv6.

IPv6: It is the newest versions of internet protocol and successor to IPv4. IPv6 uses 128-bit address schemes that are lost total of 2^{128} are $3.4* 10^{38}$ address.

6LoWPAN:

IPv6 over low power wireless personal area networks brings IP protocol to the low power device which have limited processing capability it operates in the 2.4 GHz frequency range and provide the data transfer rate off to 50 kb/s.

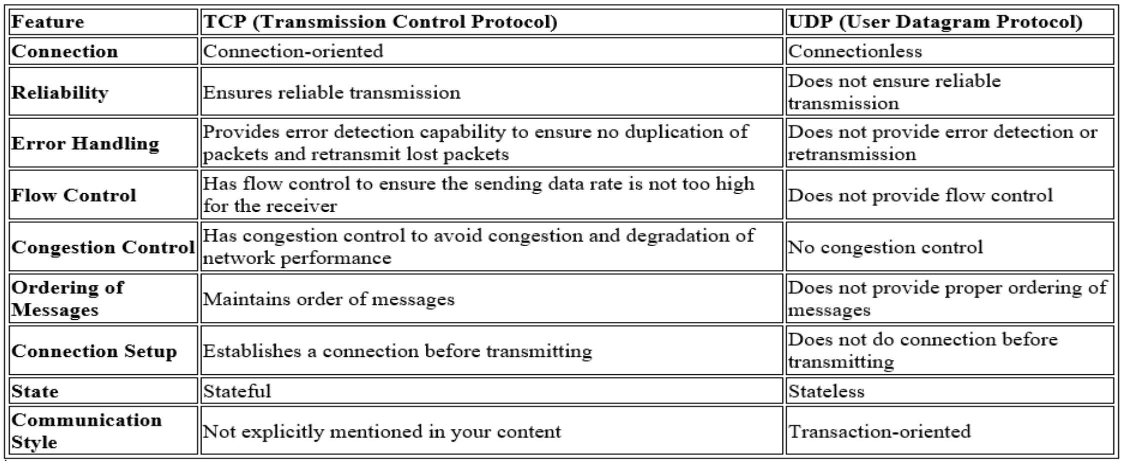
Transport layer:

The Transport layer protocols provide end-to-end message transfer capability independent of the underlying network. The message transfer capability can be set up on connections, either using handshake or without handshake acknowledgements. Provides functions such as error control, segmentation, flow control and congestion control.

TCP: Transmission control protocol is the most widely used to transport layer protocol that is used by the web browsers along with HTTP, HTTPS application layer protocols email program (SMTP application layer protocol) and file transfer protocol. TCP is a connection Oriented and stateful protocol while IP protocol deals with sending packets, TCP ensures reliable transmissions of packets in order. TCP also provide error deduction capability so that duplicate packets can be discarded and low packets are retransmitted.

The flow control capability ensures that the rate at which the sender since the data is not too high for the receiver to process.

UDP: unlike TCP, which requires carrying out an initial setup procedure, UDP is a connection less protocol. UDP is useful for time sensitive application they have very small data units to exchange and do not want the overhead of connection setup. UDP is a transactions oriented and stateless protocol. UDP does not provide guaranteed delivery, ordering of messages and duplicate eliminations.



Feature	TCP (Transmission Control Protocol)	UDP (User Datagram Protocol)
Connection	Connection-oriented	Connectionless
Reliability	Ensures reliable transmission	Does not ensure reliable transmission
Error Handling	Provides error detection capability to ensure no duplication of packets and retransmit lost packets	Does not provide error detection or retransmission
Flow Control	Has flow control to ensure the sending data rate is not too high for the receiver	Does not provide flow control
Congestion Control	Has congestion control to avoid congestion and degradation of network performance	No congestion control
Ordering of Messages	Maintains order of messages	Does not provide proper ordering of messages
Connection Setup	Establishes a connection before transmitting	Does not do connection before transmitting
State	Stateful	Stateless
Communication Style	Not explicitly mentioned in your content	Transaction-oriented

Application layer:

Application layer protocol defines how the application interfaces with the lower layer protocols to send the data over the network. Data are typically in files, is encoded by the application layer protocol and encapsulated in the transport layer protocol. Application layer protocol enables process-to-process connection using ports.

Http: Hypertext transfer protocol is the application layer protocol that forms the foundations of world wide web http includes commands such as GET, PUT, POST, DELETE, HEAD, TRACE, OPTIONS etc. The protocol follows a request-response model where are client sends request to server using the http, commands. Http is a stateless protocol and each http request is independent father request and http client can be a browser or an application running on the client example and application running on an IoT device, mobile applications or other software.

CoAP: Constrained application protocol is an application layer protocol for machine-to-machine application M2M meant for constrained environment with constrained devices and constrained networks. Like http CoAP is a web transfer protocol and uses a request- response model, however it runs on the top of the UDP instead of TC CoAP uses a client –server architecture where client communicate with server using connectionless datagrams. It is designed to easily interface with http like http, CoAP supports method such as GET, PUT, DELETE.

WebSocket: WebSocket protocol allows full duplex communication over a single socket connection for sending message between client and server. WebSocket is based on TCP and Allows streams of messages to be sent back and forth between the client and server while keeping the TCP connection open. The client can be a browser, a mobile application and IoT device

MQTT: Message Queue Telemetry Transport it is a lightweight message protocol based on public -subscribe model MQTT uses a client server Architecture by the clients such as an IoT device connect to the server also called the MQTT broker and publishers message to topic on the server. The broker forwards the message to the clients subscribed to topic MQTT is well suited for constrained and environments.

XMPP: Extensible Messaging and Presence Protocol it is a protocol for real-time communication and streaming XML data between network entities XMPP powers wide range of applications including messaging, presence, data syndication, gaming multiparty chat and voice / voice calls. XMPP Allows sending small chunks of XML data from one network entity to another in real time. XMPP supports both client to server and server –client communication path.

DDS: Data distribution service is the date centric middleware standard for device-to-device machine to machine communication DDS uses a publish subscribe model where publisher example device that generate data create topics to which subscribers per can subscribe publisher is an object responsible for data distributions and the subscriber responsible for receiving published data. DDS provide quality of service (QoS) control and configurable reliability

AMQP: Advanced Message Queuing protocols. it is an open application layer protocol for business messaging. AMQP support point to point and publish - subscribe model routing and queuing. AMQP broker receive message from publisher's example devices or applications that generate data and about them over connections to consumers publishers publish the message to exchange which then distribute message copies to queues.

1.3 Logical design of IoT

Logical design of an IoT system refers to an abstract representation of the entities and process without going into low level specification of the implementations.

1.3.1 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 as shown in the figure 1.3.

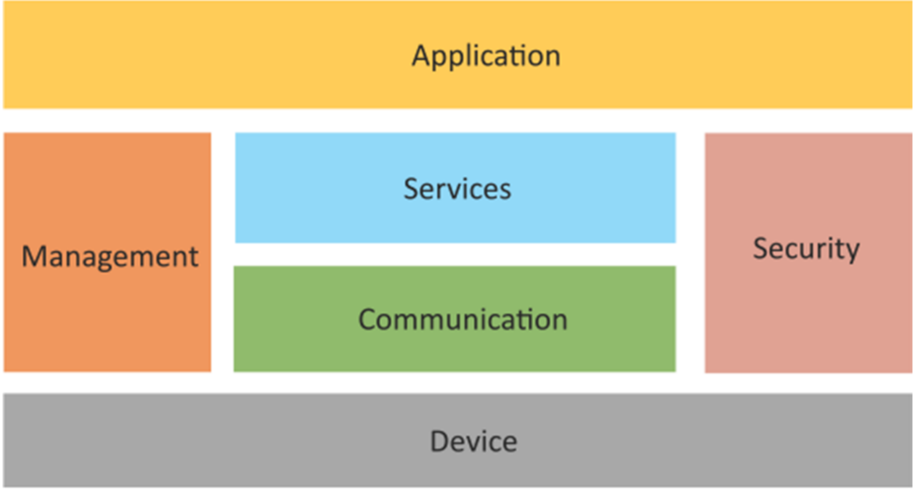


Figure 1.3: Functional Blocks of IoT

The function blocks are described as follows.

Devices: An IoT system comprises of the devices that provide sensing, actuation, monitoring and control function

Communication: communication block handles the communication systems

Services: An IoT system uses various types of IoT services such as services for device monitoring, device control services, data publishing services and services for device Discovery.

Management: Functional blocks provide various functions to govern the IoT system

Security: Security functional block security IoT system and by providing functions such as application authorization message and content integrity and data security.

Application: IoT application provides and interface that the user can used to control and monitor various aspects of the IoT system. Application also allows users to view the system status and view or analyze the processed to data.

1.3.2 IoT communication model

- **Request response:** Request-response is a Communications model in which the client sends request to the server and the server responds to the requests. when the server receives a request, it decides how to respond, if it shows the data retrieved resources definitions for the response, and then send the response to the client. Access to response model is a stateless communication model and each request response per is independent of others. Figure 1.5 shows the block diagram of Request-Response Communication Model.

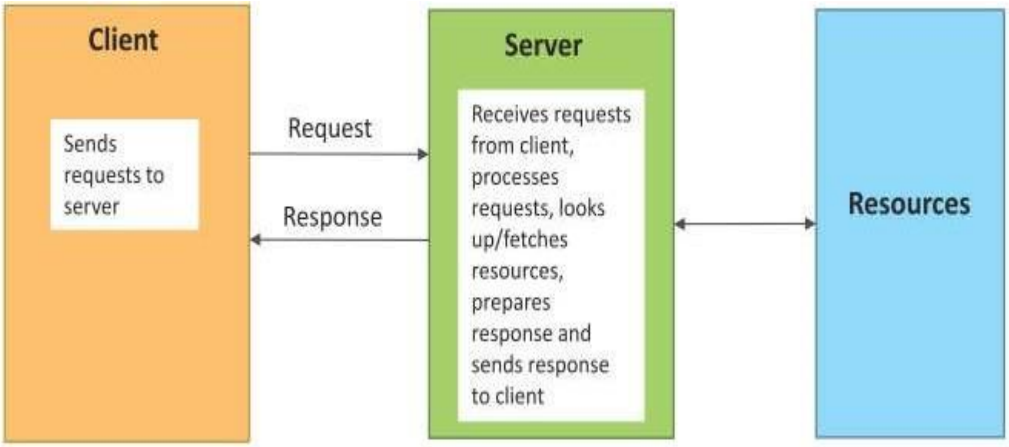

Client
Sends requests to server
Request
Response
Server
Receives requests from client, processes requests, looks up/fetches resources, prepares response and sends response to client
Resources

Figure 1.5: block diagram of Request-Response Communication Model

- **Publish - Subscribe:** Respect is a communication model that involve Publishers brokers and consumers. Publishers are the source of data. Publishers send the data to the topics which is managed by the broker. Publishers are not aware of the consumer. Consumers Subscribe to the topic which are managed by the broker. When the broker receives the data for a topic from the publisher, it sends the data to all the subscribed consumers. Figure 1.6 shows the block diagram of Publish-Subscribe Communication Model.

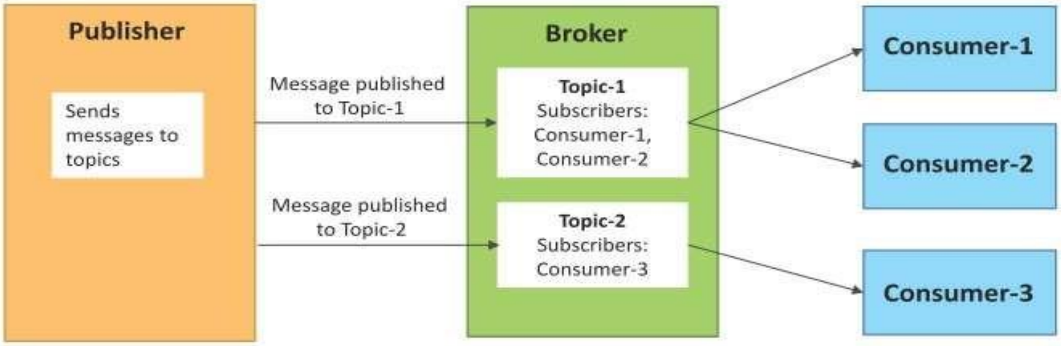


Figure 1.6: block diagram of Publish-Subscribe Communication Model.

- **Push pull:** Push pull is 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 consumer. Queues help in decoupling the messaging between the Producers and Consumers. It also acts as a buffer which helps in situations when there is a mismatch between the rate at which the produces push data and the rate at which the consumers full the data, Figure 1.7 shows the block diagram of Push-Pull Communication Model.

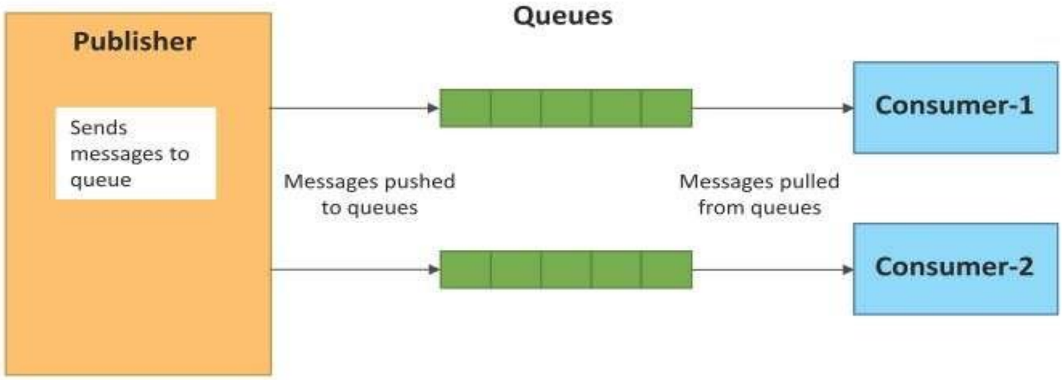


Figure 1.7: block diagram of Push-Pull Communication Model.

- **Exclusive pair:** Exclusive pair is a bi directional, fully duplex communication model that uses a persistent connection between the client and the server. once the condition 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 a stateful Communications model and the server is aware of all the open connections. Figure 1.8 shows the block diagram of Exclusive Pair Communication Model

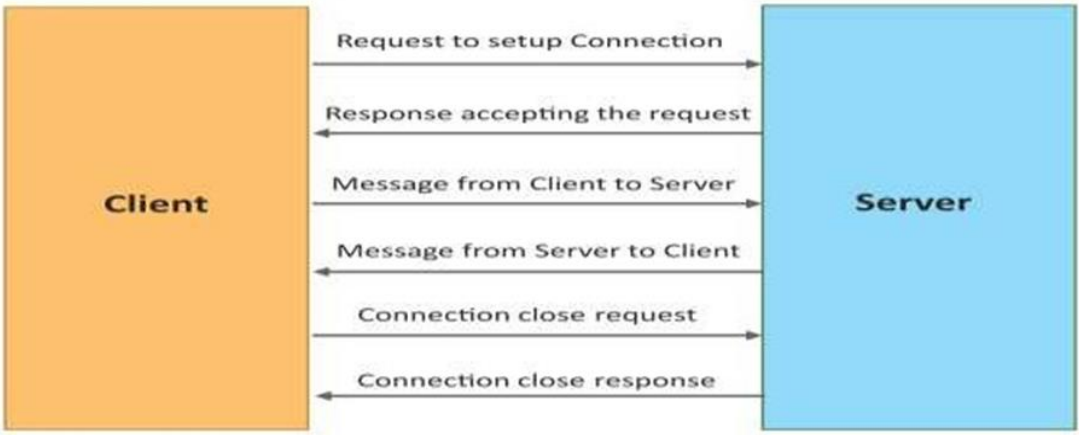

Client
Server
Request to setup Connection
Response accepting the request
Message from Client to Server
Message from Server to Client
Connection close request
Connection close response

Figure 1.8: block diagram of Exclusive Pair Communication Model

1.3.3 IoT communication APIs

REST- based communication API:

Representational state transfer is a set of architectural principles by which you can design web services and Web API that focus on a system resource and how resources states are addressed and transferred. REST API follow the request- response communication model.

The REST architectural constraints apply to the components, connectors, and data elements, within a distributed hypermedia system. Figure 1.9 shows the communication between with REST APIs

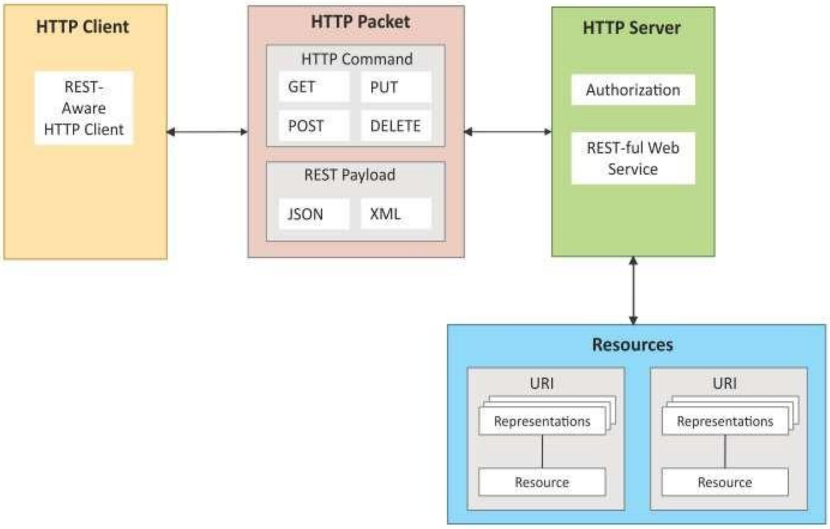


Figure 1.9: Communication with REST APIs

The REST architectural constraints are as follows:

- **Client server:** The principle behind the client-server conference separations of concerns for example client should not be concerned with the storage of data which is their concern of the server. Similarly, the server should not be concerned about the user interface which is a concern of the client. separation allows client and server to be independently deployed and updated.

- **Stateless:** Each request from client to server must contain all the information necessary to understand the request, and cannot take advantage of any stored context on the server.

- **Catchable:** Catch constrain requires that the data within the response to a request be implicitly or explicitly labeled as catchable or non-catchable. Then a client cache is given the right to reuse that response data for later, equivalent requests. completely eliminate some attractions and improve efficiency and scalability.

- **Layered system:** System constraint come off constraints, constrains the behavior of components such that each component cannot see beyond the immediate layer with which they are interacting. Example client cannot tell whether it is connected directly to the end server or to an intermediary along the way system scalability can be improved allowing intermediaries to respond to request instead of tender server.

- **Uniform interface:** Uniform interface constraints requires that the method of communication between client and server must be uniform. Resources are identified in the request and separate from the representation of the resource that are returned to the client. When climbing holds a representation of your resource it has all the information required to update or delete the resource

- **Code on demand:** Service can provide executable code script for clients to execute in their context.

A RESTful web service is a 'web API" implemented using HTTP and REST principles. Figure 1.10 shows the interactions in the request-response model used by REST.

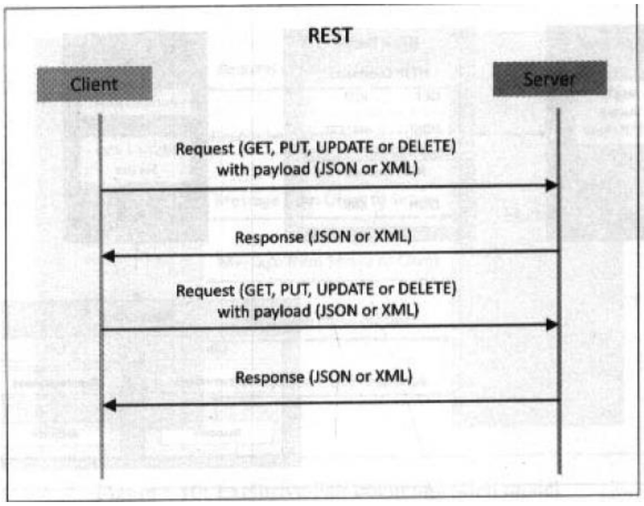

REST
Client
Server
Request (GET, PUT, UPDATE or DELETE) with payload (JSON or XML)
Response (JSON or XML)
Request (GET, PUT, UPDATE or DELETE) with payload (JSON or XML)
Response (JSON or XML)

Figure 1.10 Request-response model used by REST.

RESTful web service is a collection of resources which are represented by URLs. The client sends requests to these URLs using the methods defined by the HTTP protocol (e.g., GET, PUT, POST, or DELETE), as shown in Table 1.1.

HTTP Method	Resource Type	Action	Example
GET	Collection URI	List all the resources in a collection	http://example.com/api/tasks/(list all tasks)
GET	Element URI	Get information about a resource	http://example.com/api/tasks/1/(get information on task-1)
POST	Collection URI	Create a new resource	http://example.com/api/tasks/(create a new task from data provided in the request)
POST	Element URI	Generally not used	
PUT	Collection URI	Replace the entire collection with another collection	http://example.com/api/tasks/(replace entire collection with data provided in request)
PUT	Element URI	Update a resource	http://example.com/api/tasks/1/(update task-1 with data provided in the request)
DELETE	Collection URI	Delete the entire collection	http://example.com/api/tasks/(delete all tasks)
DELETE	Element URI	Delete a resource	http://example.com/api/tasks/1/(delete task-1)

Table 1.1: HTTP request methods and actions

WebSocket based communication API:

WebSocket API allow bi directional, full duplex communication between client and server. Unlike request-response API allow full duplex communication and do not require new connection to be set up for each message to be sent. WebSocket communication begins with connection setup request send by the client to the server. The request is sent over http and the server interprets it as an upgrade request. If the server support protocol response to the website handshake response after the connection setup the client and the server can send data or messages to each other in full duplex model. WebSocket API reduce network traffic and latency as there is no overhead for connection setup and determination records to each message. Figure 1.11 shows the exclusive pair model used by WebSocket APIs.

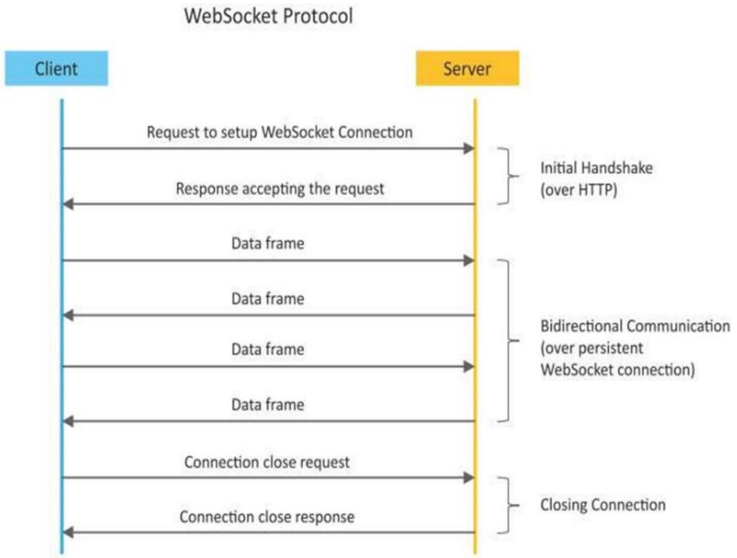

WebSocket Protocol
Client
Server
Request to setup WebSocket Connection
Response accepting the request
Initial Handshake (over HTTP)
Data frame
Data frame
Data frame
Data frame
Bidirectional Communication (over persistent WebSocket connection)
Connection close request
Connection close response
Closing Connection

Figure 1.11 shows the exclusive pair model used by WebSocket APIs.

1.4 IoT enabling Technologies

It is enabled by several Technologies including wireless sensor networks, cloud computing big Data Analytics, embedded system, security protocols and architectures, communication protocols, web service, mobile internet and semantic search engine.

1.4.1 wireless sensor network

Wireless sensor network (wsn) comprises of distributed devices with the sensor which are used to monitor the environmental and physical conditions. A WSN consists of a number of end nodes and routers and a coordinator. End nodes have several sensors attached to them. End node can also act as a router. Routers are responsible for routing the data packet from end nodes to the coordinator. The coordinator node collects the data from all the notes coordinator also act as a Gateway that connects the WSN to the internet. IoT systems are described as follows

- Weather monitoring system using WSN in which the nodes collect temperature, humidity and other data which is aggregated and analyzed.
- Indoor air quality monitoring system using WSN to collect data on the indoor air quality and connections of various gases.
- Soil moisture monitoring system using WSN to monitor soil moisture at various location.
- Surveillance systems use WSN for collecting surveillance data (motion detection data)
- Smart grid uses wireless sensor network for monitoring the grid at various point.
- Structural health monitoring systems use WSN to monitor the health of structure by writing vibration data from sensor nodes deployed at various points in the structure.

1.4.2 Cloud computing:

Cloud Computing is a transformative computing paradigm that involves delivering applications and services over the internet. Cloud Computing involves provisioning of computing networking and storage resources on demand and providing these resources as metered services to the users, in a “pay as you go” model. cloud Computing resources can be provisioned on demand by the user without requiring interactions with the Cloud Service Provider. The process of provisioning resources used automatic Cloud Computing resources can be accessed then it worked using standard access mechanism that provide platform-independent access through the use of heterogeneous client platforms such as workstations laptops tablets and Smartphones the computing and storage resources provided by Cloud Service Provider our food to serve multiple users using multi-Tenancy. Multi-tenant aspects on the multiple users to be served by the same physical hardware.
Cloud Computing services are offered to user in different forms

Infrastructure as a service (IAAS):
IaaS provides the user the ability provision computing and storage resources. These

resources are provided to the users as virtual machine instances and virtual storage. Users can start, stop configure and manage the virtual machines instance on the virtual storage using can deploy operating systems and applications on their choice on the actual resources provisions in the cloud. Cloud Service Provider manages the underlying infrastructure.

Platform as a service (PaaS):
platform as a service provides the user the ability to develop and deploy application in the cloud using the deployment tool application programming interfaces API, software libraries and services provided by the Cloud Service Provider. The Cloud Service Provider manages the underlying cloud infrastructure including servers, network, operating systems and storage.

Software as a service (SaaS):
Provide the user a complete software application of the user interface to the application itself. The Cloud Service Provider manage the underlying cloud infrastructure including server, network storage and application software, and the user is unaware of the underlying architecture of the cloud. Applications are provided to the user through a thin client interface example Browser application. SaaS applications are accessed from various client smartphones running different operating system.

1.4.3 Big Data Analytics

Big data is defined as collections of data set whose volume, velocity in terms of its temporal variations) or variety, is so large that it is difficult to store, manage, process and analyze the data using traditional database and data processing tools. Big Data Analytics involving several steps starting from Data cleaning data munging data processing and visualization.

Some examples of big data generated by IoT systems are described as follows

1. Sensor data generated by IoT system such as weather monitoring stations
2. Machine sensor data collected from sensor embedded in Industrial and energy system for monitoring their files and protecting failure
3. Health and fitness data generated by IoT devices such as wearable fitness band.
4. Data generated by IoT system for Location tracking of vehicle.
5. Data generated by retail inventory monitoring system.

Characteristics of big data include:

Volume: Through there is no fixed threshold for volume of data to be considered as big data, however the term big data is used for massive scale data that is difficult to store, manage and process using traditional data bases and data processing architecture. The volume of data generated by modern IT, industrial and Healthcare systems for example is a growing exponentially driven by the lowering cost of data storage and processing architectures and the need to extract valuable insights from the data to improve business processes, efficiency and services to consumer.

Velocity: Velocity is another important characteristic of big data and the primary reasons for exponential growth of data velocity of the data of a store how fast the data is generated and how frequently it varies. Modern IT Industrial and other systems are generating data at increasing the highest speeds.

Variety: Variety refers to the forms of the data. Big data comes in for different forms such as structured or unstructured data including text data, audio, video and sensor data.

1.4.4 Communications protocol:

Communications protocols form the backbone of IoT system and enable network connectivity and coupling to applications. Communications protocols allow device to exchange data over the network. These protocols define the data exchange formats and data encoding schemes for devices and routing of packets from source to destination. Other function of the protocol includes sequence control flow control and transmissions of Lost packet.

1.4.5 Embedded systems

An Embedded system is computer system that has computer hardware and software embedded perform specific task. In contrast to general purpose computers or personal computers which can perform various types of tasks, embedded systems are designed to perform a specific set of tasks. Embedded system includes Microprocessor and Microcontroller memory Ram ROM cache networking units (Ethernet WI-FI adaptor) input/output unit display keyboard, display and storage such as Flash Memory some embedded system have specialist processes such as digital signal processor DSP graphic processor and application.

1.5 IoT levels and Deployment Templates

In this section we define various levels of IoT systems with increasing completely. IoT system comprises of the following components:

1. **Device:** An IoT device allow identification, remote sensing, actuating and remote monitoring capabilities.
2. **Resources:** Resources are software components on the device for accessing and storing

information for controlling actuator connected to the device also include software components that enable network access for the device .

3. **Controller service:** Controller Service is a native service that runs on the device and interact with the web services. Controller service sends data from the device to the web service receive command from the application from controlling the device.
4. **Database:** Database can be either local or in the cloud and stores the data generated by the IoT device.
5. **Web service:** Serve as a link between the device, application database and analysis components. Web Services can be implemented using HTTP and REST principles or using website protocol.

A comparison of REST and WebSocket is provided below:

Stateless/stateful: Rest services stateless in nature. Each request contains all the information needed to process it. Request are independent of each other. Website on the other hand is stateful in nature where the server maintains the state and is aware of all the open connections.

Directional / Bi- directional: REST service operates over http and unidirectional. Request is always sent by a client and the server response to the request. And other hand website is a bi directional product server to send message to each other

Request response / full duplex: REST service follower request response Communications model where the client sends request and the server response to the request. Website and the other hand Allow full-duplex Communications between the client and server, it means both client and server can send messages to can independently.

TCP connections: For REST Service each http request involves setting up in a new TCP connections WebSocket on the other hand involves a single TCP connection over which the client and server communicate in a full duplex mode.

Headache Overhead: REST service operates over http, and each request is independent of others. Thus, each request carries http header which is an overhead. Due to the overhead of http headers, REST is not suitable for real time applications left hand does not involve overhead of headers. After the initial handshake the client and server exchange messages with minimal frame information.

Scalability: Scalability is easier in this case of the REST services of request are independent and no state information needs to be maintained by the server. Thus, both horizontal out and vertical scaling solutions are possible for REST services. For webSockets horizontal scaling can be cumbersome due to stateful nature of the communication. Since the server maintains the state of our connection, vertical scaling is easier for WebSocket than horizontal scaling.

Analysis component: The analysis component is responsible for analyzing the IoT data and generate results in the form which are easy for the user to understand. Analysis of IoT data can be performed either locally or in the cloud. Analyzed results are stored in the local or cloud database.

Application: IoT applications provide an interface that the user can use to control and monitor various aspects of the IoT system. Applications also allow user to view the system status and view the processed data.

1.5.1 IoT level 1:

Level One IoT system has a single node / device that performs sensing and/or actuation, stores data, performs analysis and the host to the application as shown in the figure 1.12. Level 1 IoT systems are suitable for modeling low cost and low complexity solutions where the data involving is not big and the analysis requirements are not computationally intensive.

Let us now consider an **example** of Level 1 IoT system for **home automation**. This system consists of the single node that allows controlling the lights and appliances in your home remotely. The device used in this system interface with their lights and appliances using electronic relay switches.

The status information of each light or appliance is maintained in a local database. REST service deployed locally Allow retrieving and updating the state of each light or appliances in the status database.

The controller service continuously monitors the state of each light or appliance and triggers the relay switches accordingly. The applications which is deployed locally has a user interface for controlling the lights or appliances. since the device is connected to the internet, the application can be accessed remotely as well.

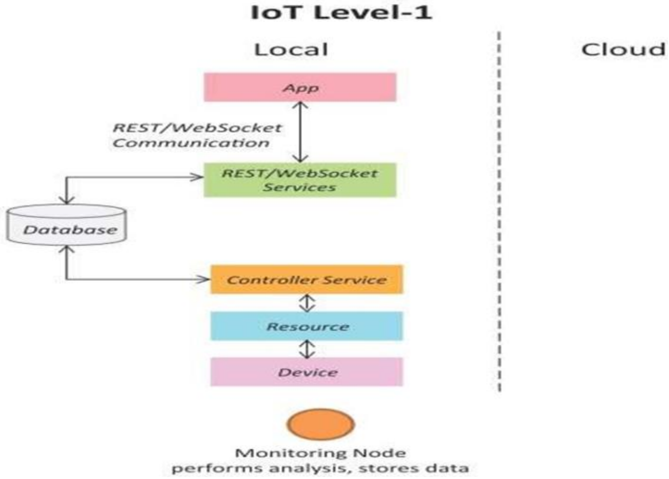


Figure 1.12: IoT Level-1

1.5.2 IoT level 2:

Level 2 IoT system has a single node that performs sensing and/or actuation and local analysis as shown in figure 1.13. Data is stored in the cloud and application is usually cloud based systems are suitable for solutions where the data in world is big, however the primary analysis requirement is not computationally intensive and can be done local itself.

Construct an **example** of Level 2 IoT system for **smart irrigation**.

The system consists of the single node that monitor the soil moisture level and control segregation system. The device used in this system collect soil moisture data from sensor the controller service continuously monitors the moisture level. If the monster level drops below a threshold, the irrigation system is turned on. For controlling the irrigation system actuators such as solenoid valve can be used. Rest Web Services is used for storing and retrieving data which is stored in the cloud database. A cloud-based application is used for visualizing the moisture level over a period of time, which can help in making decisions about irrigation schedules.

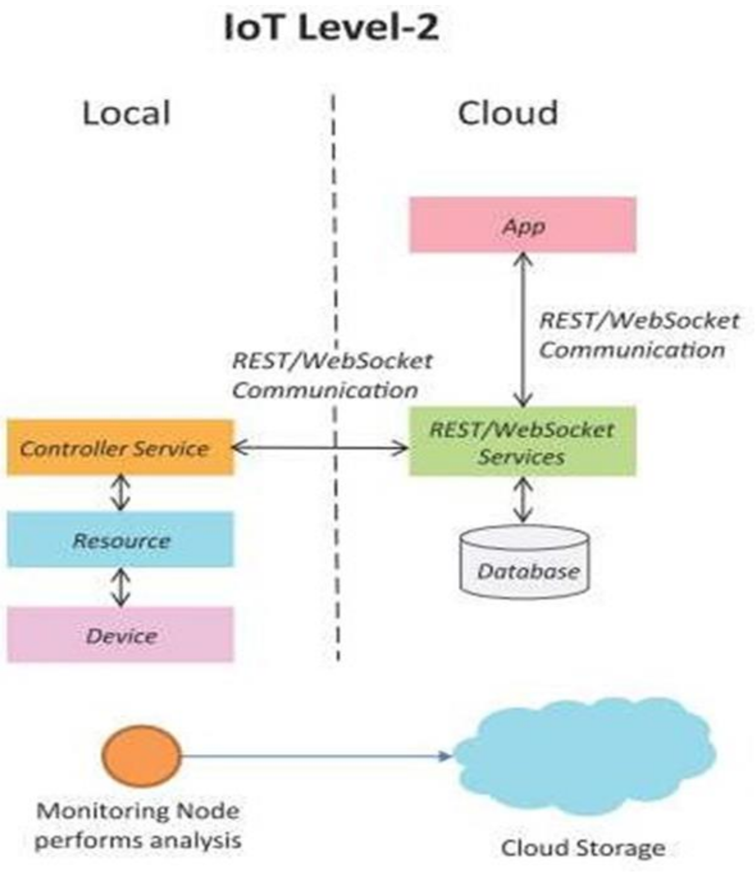


Figure 1.13: IoT Level-2

1.5.3 IoT Level 3:

Level 3 system has a single node. Data is stored and analyzed in the cloud application is cloud-based as shown in figure 1.14. Level 3 IoT system suitable for solutions where the data involved is big and analysis requirements computationally intensive.
Let us considered an **example** of Level 3 IoT system **tracking package handling**. The system consists of a single node that monitors the vibration level for package being shipped.

The device in the system uses accelerometer and gyroscope sensor for monitoring vibration levels. The controller service sends sensor data to the cloud in real time using a website service. The data is stored in the cloud and also visualized using a cloud-based application.

The analysis component in the cloud can Trigger alert the vibration level becomes greater than threshold. The benefit of using WebSocket service instead of the REST service this example the sensor data can be sent in real-time to the cloud. Cloud based application can subscribe to the sensor data feeds for you in the real-time data.

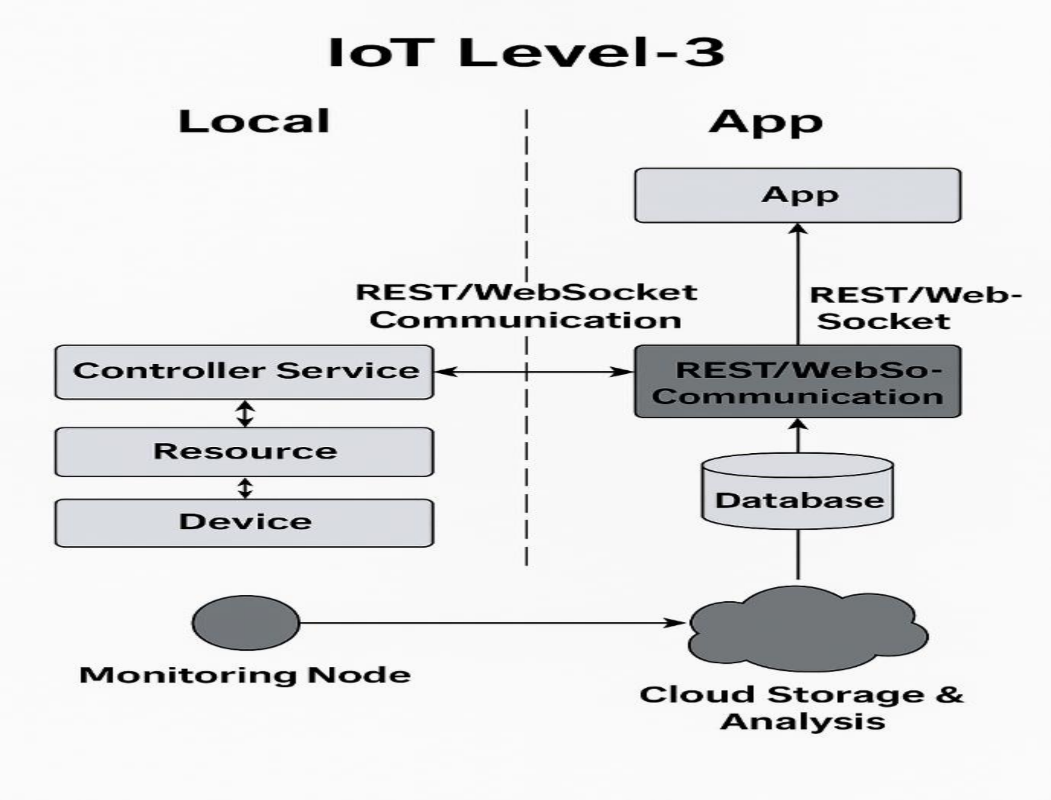


Figure 1.14: IoT Level-3

1.5.4 IoT level 4

A level 4 IoT system has multiple notes that perform local analysis. Data is stored in the cloud and application is cloud based as shown in the figure 1.15. level 4 contains local and cloud-based observer notes which can subscribe to and receive information collected in the cloud from IoT devices. Observer node can process information and use it for various applications, however observer notes do not perform any control function. 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.

let us consider an example of level four IoT system for **noise monitoring**. The system consists of multiple notes placed in different locations for monitoring noise level in an area. In this example with sound sensor. Nodes are independent of each other each node runs in one controller service that sends the data to the cloud. The data is stored in a cloud database the analysis of the data collected from a number of notes is done in the cloud

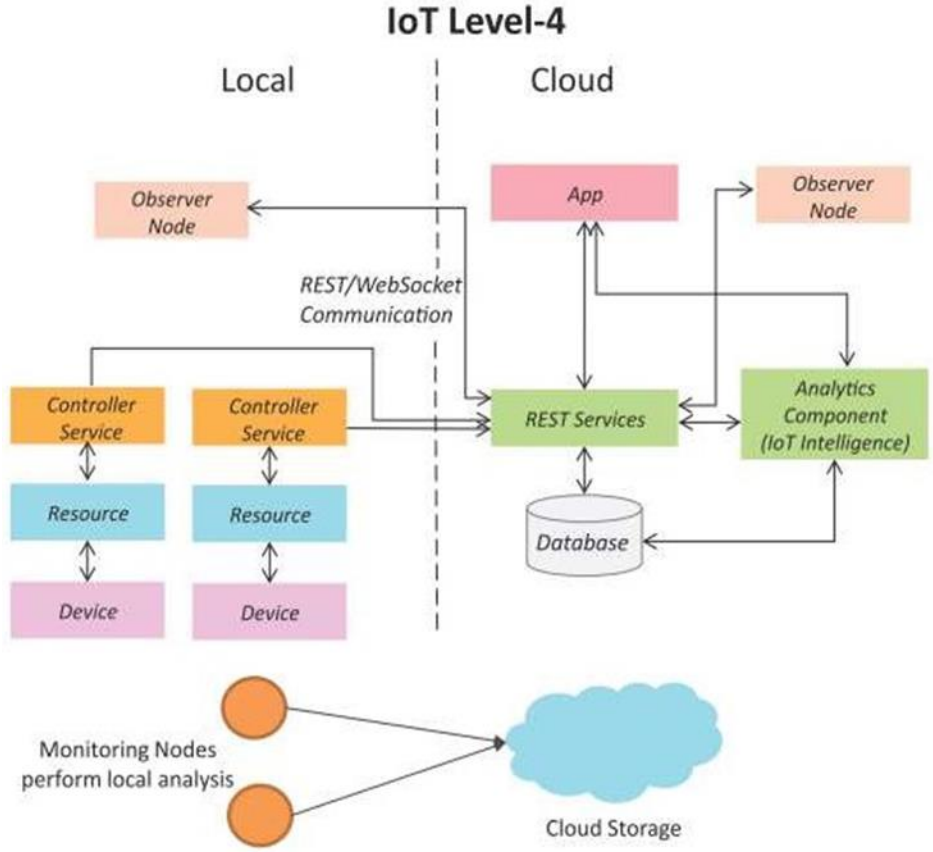


Figure 1.15: IoT Level-4

1.5.5 IoT Level 5:

IoT system has multiple end nodes and one coordinator node as shown in figure 1.16 the end nodes that perform sensing and / or actuation. Coordinator node collects data from the entry and send to the cloud. Data is stored and analyzed in the cloud and applications is cloud based. Level 5 IoT system are suitable **example** for **forest fire detection**. The system consists of multiple nodes placed in different locations for monitoring temperature, humidity and carbon dioxide levels in a forest. The endnotes in this example are equipped with various sensors such as temperature humidity and to CO2.

The coordinator node collects the data from the end nodes and act as a Gateway that provides internet connectivity to the IoT system. The controller service on the coordinator device sends the collected data to the cloud. The data is stored in the cloud database. The analysis of the data is done in the computing cloud to aggregate the data and make prediction.

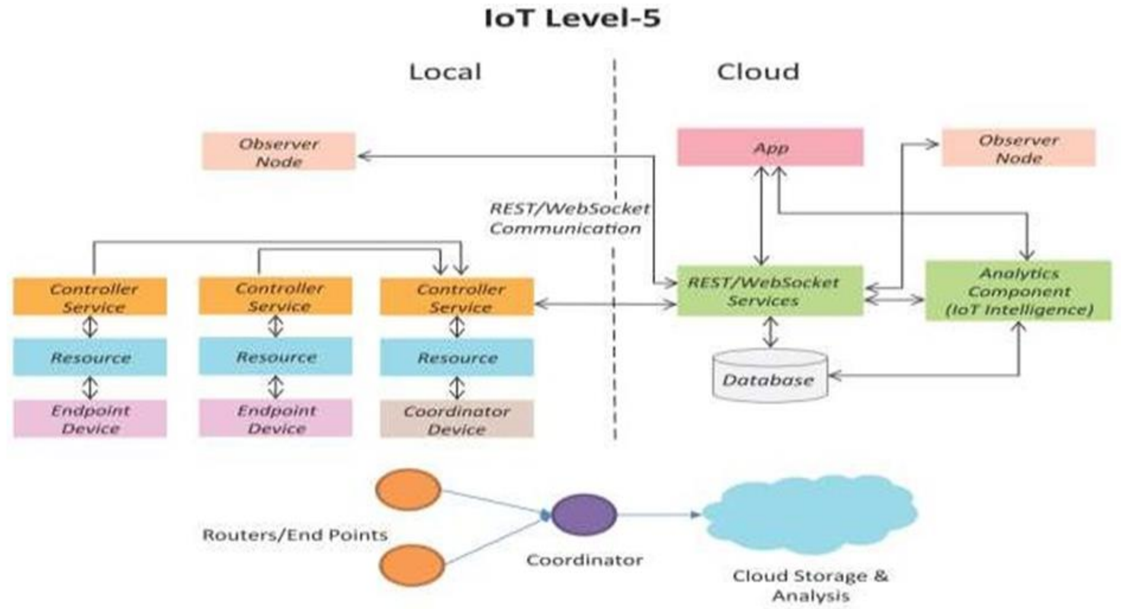


Figure 1.16: IoT Level-5

1.5.6 IoT Level 6:

IoT Level 6 system has multiple Independent and nodes that perform sensing and / or actuations and send data to the cloud. Data is stored in the cloud and applications is cloud based as shown in figure 1.17.

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 notes and send control commands to the notes.

Let us consider an **example** of the level 6 IoT system for **weather monitoring**. The system consists of multiple nodes placed in different location for monitoring temperature, humidity and pressure in an area. The end nodes are equipped with various sensors such as temperature, pressure and humidity. The end nodes send the data to the cloud in real time using a WebSocket service. The data is stored in a cloud database. The analysis of the data is done in the cloud to aggregate the data and make predictions. A cloud-based applications is used for visualizing the data.

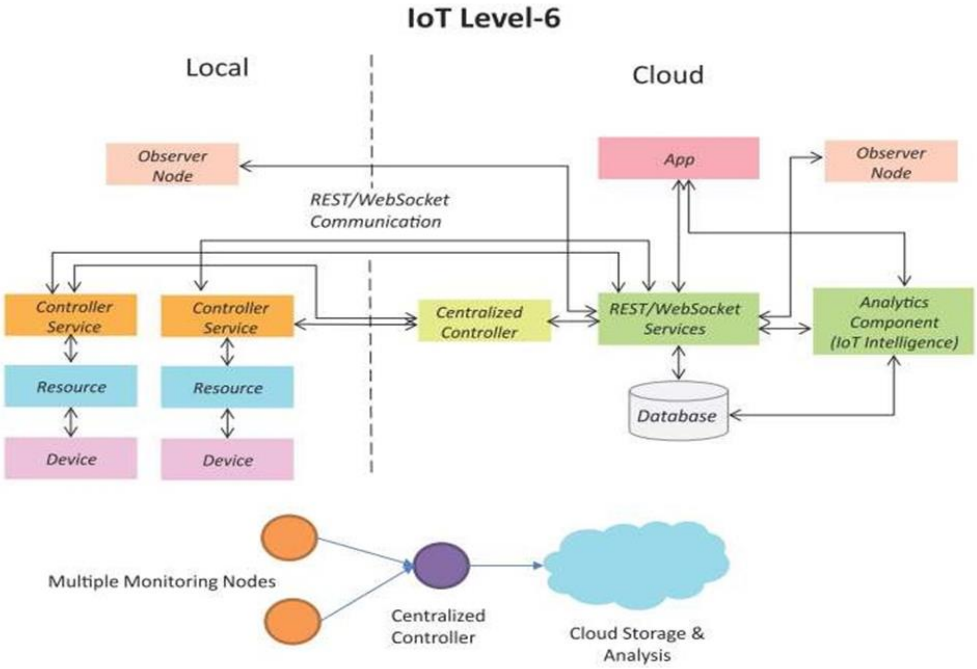


Figure 1.17: IoT Level-6

Module Question Bank

1. Describe an example of an IoT system in which information and knowledge are inferred from data.
2. Why do IoT systems have to be self-adapting and self-configuring?
3. What is the role of things and the Internet in IoT?
4. What is the function of the communication functional block in an IoT system?
5. Describe an example of an IoT service that uses the publish-subscribe communication model.
6. Describe an example of an IoT service that uses WebSocket-based communication.
7. What are the architectural constraints of REST?
8. What is the role of a coordinator in a wireless sensor network?
9. What is the role of a controller service in an IoT system?