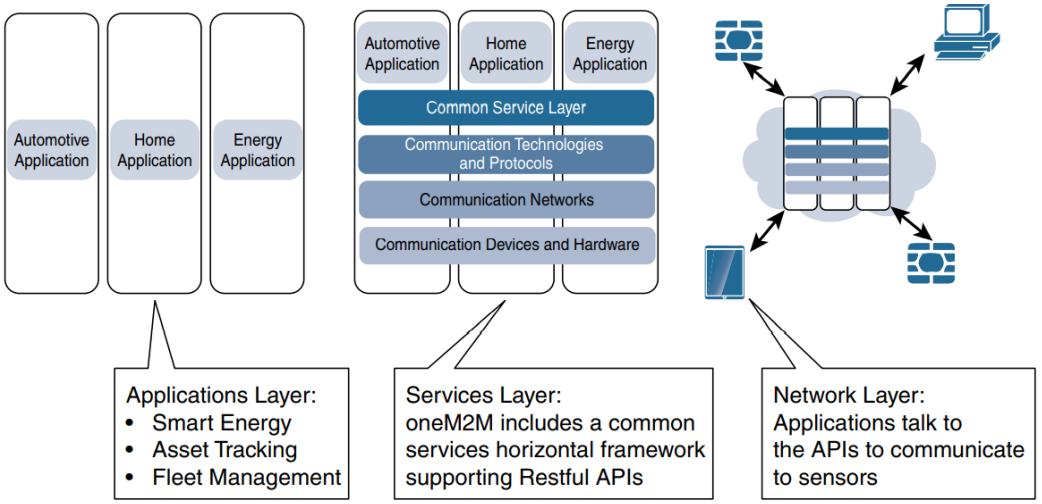
# MODULE 1

## EVOLUTIONARY PHASES OF THE INTERNET

- **CONNECTIVITY PHASE**
  - Digital access
  - Web Browser, Email, Search Engines
- **NETWORK ECONOMY PHASE**
  - Digital Business
  - E-Commerce, Supply Chain Management
- **IMMERSIVE EXPERIENCE PHASE**
  - Digital Interaction
  - Social Media, Video Streaming, Online Gaming
- **IoT PHASE**
  - Digital World
  - Connecting Objects, Infrastructure, People & Data

## ONEM2M ARCHITECTURE

- OneM2M architecture was developed by European Telecommunication Standards Institutes (ETSI) in order to standardise machine to machine communication. Later this model was adapted for IoT
- The main goal of OneM2M architecture is to create a common service layer which can be readily embedded in field device in order to enable communication with the server
- OneM2M architecture offers interoperability at all levels of the IoT functional stack
- It divides the IoT function stack into three groups namely **Application Layer, Service Layer** & **Network Layer**




- **APPLICATION LAYER**
  - This layer manages the communication between field devices and the servers. It contains various application layer protocols and standards for defining API's that can be used for communication

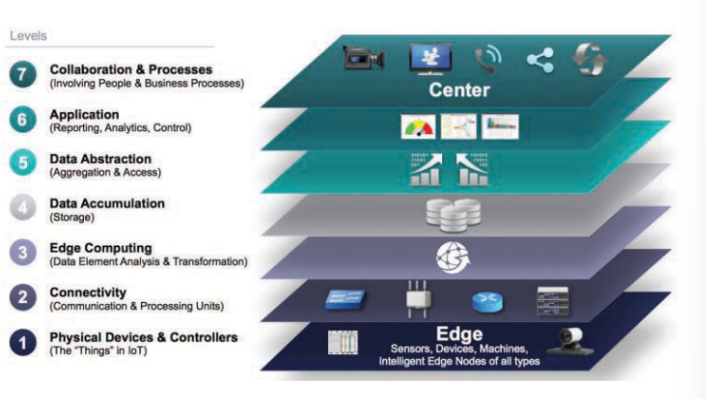
- **SERVICE LAYER**
    - This is a conceptual layer which adds support for API and middleware to access 3$^{RD}$ party service. OneM2M service ;ayer can be readily embedded in various software and hardware components to establish connectivity
- **NETWORK LAYER**
    - This layer manages the communication device and the communication network that connects the devices. It contains various protocols like IEEE 802.15.4, ZigBee etc. In cases where machine to machine communication is not needed a Field Area Network (FAN) is used for communication

**IoT WORLD FORUM ARCHITECTURE**

- The IoTWF architecture divides the IoT function stack into 7 layers. It Provides clean and simplified view of IoT
- Each of the seven layers have a specific functionality and the entire model is enriched with security measures
- The data flows from layer 1 (Physical & controller layer) to layer 7 (Collaboration & Process Layer) in an upward fashion. The seven layers of the IoTWF model are explain below
    - **Physical Device & Controller Layer**
      This layer includes all the things on the IoT which include various endpoint devices and sensors. This layer is the point of origin of data in IoT
    - **Connectivity Layer**
      This layer includes all the networking devices. The main function of this layer is to provides timely and reliable transmission of data from layer 1 to layer 3
    - **Edge Computing Layer**
      This layer processes the data obtained from layer 1 to make it ready for storage and processing by higher layers. Various process like data evaluation, filtering, aggregation reformatting, decoding are carried out and if any invalid data is received suitable actions are taken
    - **Data Accumulation Layer**
      This layer obtains the processed data and stores it in formats which can be queried
    - **Data Abstraction Layer**
      This layer reconciles multiple formats of data, ensuring consistent semantics from various data sources., ensures that the data obtained is complete and consolidates the data.
    - **Application Layer**

This layer includes the application which consumes the data. The applications might monitor, control or provoked reports based in the analysis on the data

- **Collaboration & Process Layer**
  This layer delivers the insights and information obtained from the applications to consumers thus making IoT useful




**BENEFITS OF IOTWF ARCHITECTURE**

- Decomposes the IoT problem into smaller parts
- Explains how layers are related to each other
- Different parts can be provided by different vendors
- Have a process defined interface which improves interoperability
- Enriched with security measures

**SOME OTHER COMMON IoT ARCHITECTURES**

- Purdue Model for Control Horechey
- Industrial Internet Reference Architecture (IIRA)
- Internet of Things - Architecture (IoT-A)

**SIMPLIFIED IOT ARCHITECTURE AND CORE IOT FUNCTIONAL STACK**

- Write content based on IoTWF

| PARAMETER | CLOUD COMPUTING | FOG COMPUTING | EDGE COMPUTING |
|---|---|---|---|
| **LATENCY** | High | Medium | Low |
| **SECURITY** | Less Secure | Highly Secure | Highly Secure |
| **RESPONSE TIME** | Minutes, Days | Minutes, Seconds | Seconds, Milliseconds |
| **SCALABILITY** | High & Easy to Scale | Scales Within Network | Less & Hard to Scale |
| **LOCATION OF PROCESSING** | At Server | At Intermediate Location | At Edge Device |

| | | | |
|---|---|---|---|
| | | (Like Gateway) | |
| **INTEROPERABILITY** | High | High | Low |
| **COMPUTING POWER** | High | Limited | Limited |
| **GEO DISTRIBUTION** | Centralised | Distributed | Distributed |

# MODULE 2

### SENSORS

- A sensor is a physical device that measures some physical quality and converts it into electrical / digital signals. This digital signal can later be consumed by another device
- Sensors can be classified based on various constraints, some of them are
    - **Active & Passive -** Active sensors require an external power supply to operate whereas Passive sensors do not require an external power supply.
    - **Invasive & Non-Invasive -** Invasive sensors are part of the environment in which they are measuring where as Non-Invasive sensors are not part of the environment
    - **Contact & Non-Contact -** Contact sensors require physical connection to what they are measuring where as Non-Contact sensors do not need a physical connection
    - **Absolute & Relative -** Absolute sensors measures values on an absolute scale where as Relative sensors measures values with reference to a fixed value
    - **Area of Application -** Sensors can be categories based on the industries / fields in which they are being used
    - **How Sensor Measures -** Sensors can be categories based on they type of physical mechanism they use to measure a value
    - **What Sensor Measures -** Sensors can be categories based on the actual physical value which they measure.

### ACTUATORS

- Actuators are a natural complement to sensors. They receive electrical / digital signal and perform some physical action relevant to the signal
- Actuators can be classified based on various constraints, some of them are
    - **Type of Motion -** Actuators can be categorised based on the type of motion they perform like linear, rotation, oscillation etc.
    - **Power Consumption -** Actuators can be classified based on their power consumption like micro. low, medium high etc.
    - **Binary & Continuous -** Actuators can be classified based on the number of states and type of their outputs
    - **Area of Application -** Actuators can be categories based on the industries / fields in which they are being used
    - **Type of Energy -** Actuators can be classified based on their energy type like mechanical, electrical, hydroelectric, thermal etc.

**SMART OBJECTS**

- A smart object is the simple and fundamental building block of IoT. A smart object can be describe based on four fundamental characteristics
    - **Processing Unit**
    - **Sensors & Actuators**
    - **Communication Device**
    - **Power Source**
- As the evolution of IoT development proceeds we are able to the following trends in characteristic of smart objects
    - **Reduction in Size**
    - **Reduction in Power Consumption**
    - **Increased Processing Power**
    - **Improved Communication Capabilities**
    - **Increased Communication Standards**

**HARDWARE PLATFORM**

- **CONTENT FOR ARDUINO & RASPBERRY PI**
- **CODE FOR ARDUINO**

# MODULE 3

# MODULE 4

## TYPES OF OPTIMIZATION

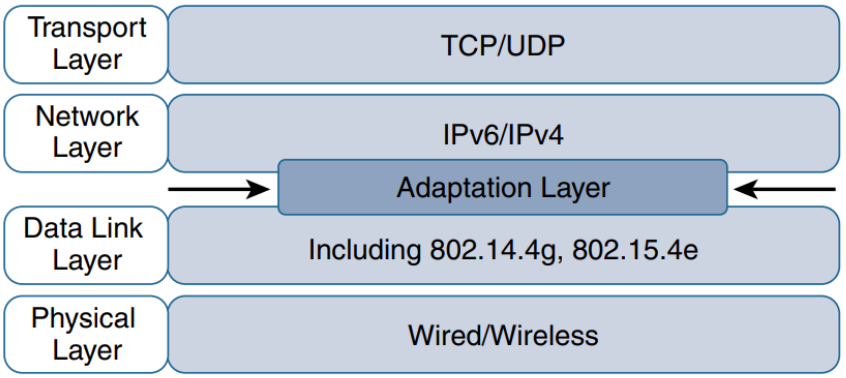
1. **ADAPTATION -** Application Layer Gateways are used for translating between Non-IP and IP stack
2. **ADOPTION -** Underlying non-IP stack is modified into IP stack

For example, SCADA supports both adaptation & adoption. ZigBee supports adaptation through ZigBee Gateway

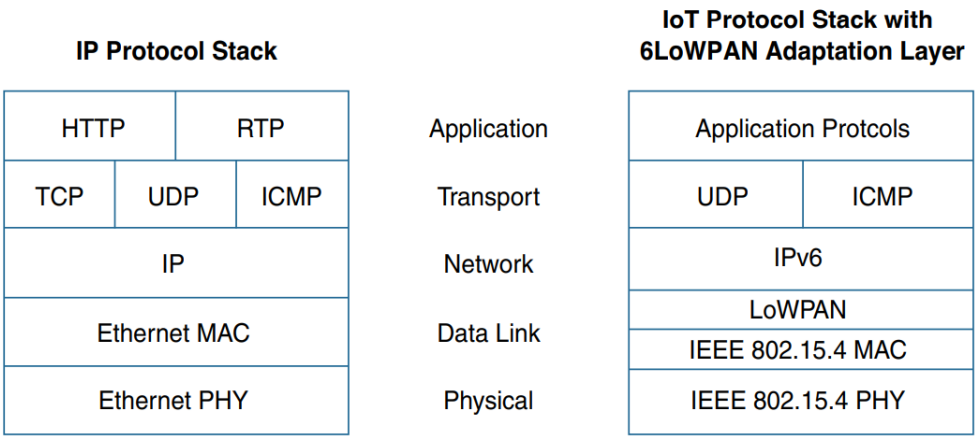
## NEED FOR OPTIMIZATION

1. **CONSTRAINED NODES**
   - IoT devices that have low processing capabilities and low power consumptions are called as constrained nodes
   - Due to this nature communication in these device has to be optimised
2. **CONSTRAINED NETWORKS**
   - Communication network with low bandwidth, high latency, high packet last and limited power supply is called as constrained network

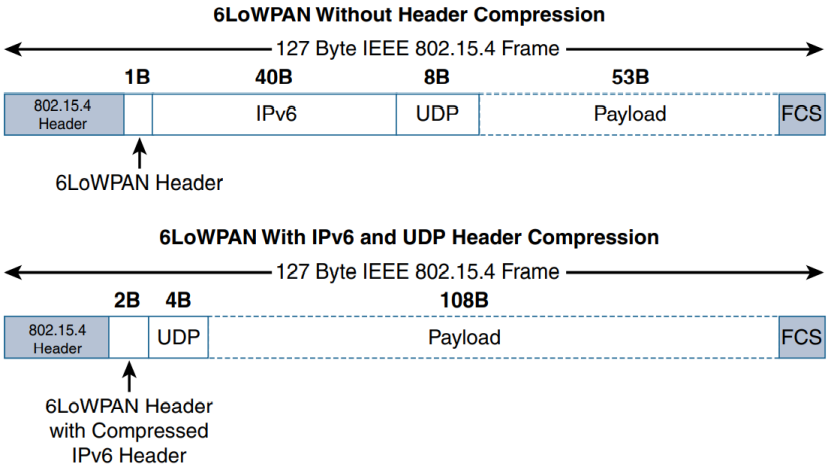
## 6LoWPAN TO 6Lo OPTIMISATION




- IoT implements optimization at the **Adaptation Layer** layer of the IP stack. An Adaptation Layer is a model for packaging IP into lower layer protocols. It is defined by IETF working group and published as RFC
- 6LoWPAN to 6Lo optimization is carried out through **Header Compression**, **Fragmentation** and **Mesh Addressing**. At the adaptation layer respective frame headers can be stacked depending on the need for optimization
- These optimization are based on IPv6 and IEEE 802.15.4 protocols




- **HEADER COMPRESSION**
    - The Header Compression method can reduce the 40 Bytes of IPv6 Header & 8 Bytes of UDP Header to 6 Bytes combined
    - This is achieved by using shared information available among the nodes in the network and assuming values for common header fields



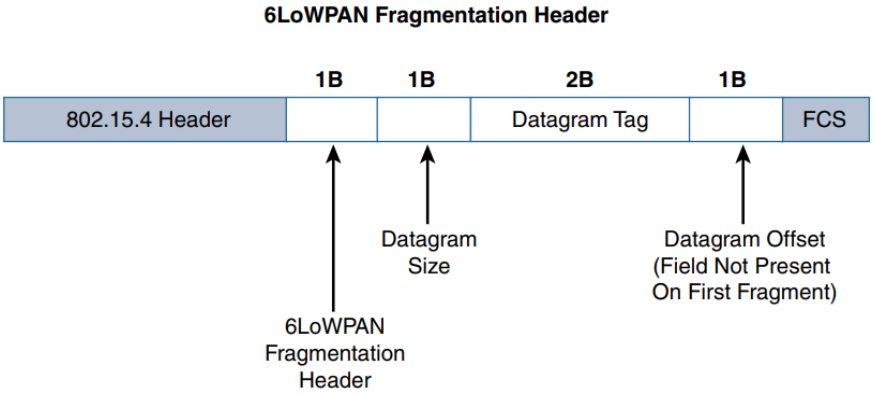

- **FRAGMENTATION**
    - Fragmentation is needed due to the conflict in Maximum Transmission Unit (MTU) between IPv6 packet and IEEE 802.15.4 frame
    - MTU of IPv6 is 1280 Bytes where as MTU of IEEE 802.15.4 is 127 Bytes, In this case a larger packet has to be storge inside a smaller frame which is not possible
    - Hence a IPv6 packet are fragmented and sent across multiple IEEE 802.15.4 frames
    - Fragmentation header contains the following header fields :

➢ **6LoWPAN Fragmentation Header -** Size of 1 Byte and it indicated weather the upcoming headers are used for fragmentation

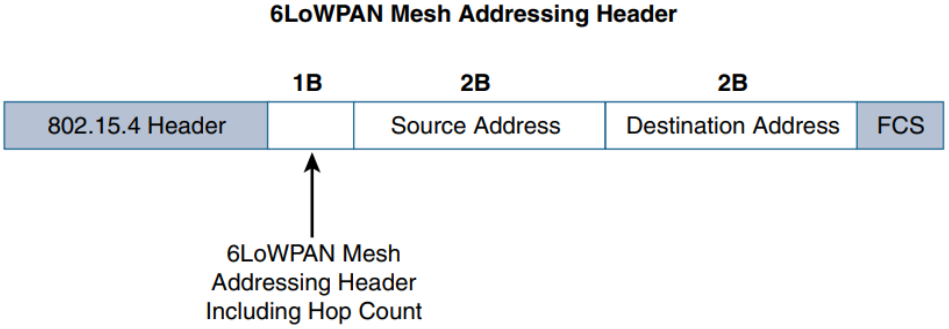
➢ **Datagram Size -** Size of 1 Byte and it specifies the total size of unfragmented payload

➢ **Datagram Tag -** Size of 2 Bytes and it specifies the set of fragments of a payload

➢ **Datagram Offset -** Size of 1 Byte and it specifies the position of the fragment in the payload. This field will be set 0 for the 1$^{ST}$ fragment

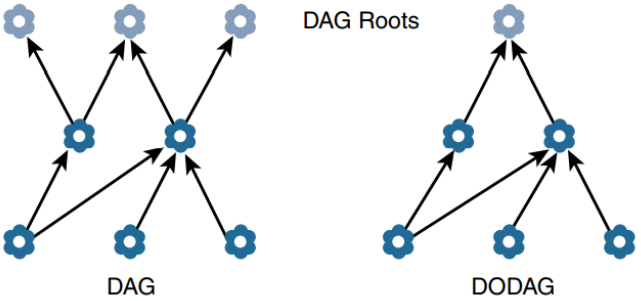



- **MESH ADDRESSING**
  - Mesh Addressing is used for forwarding frames over multiple hopes
  - Mesh Addressing header contains the following header fields :
    - ➢ **Hop Limit -** Size of 1 Byte and it is specific how many times a frame can be forwarded. On each hop the value is decreased by 1 and when it reaches 0 the frame is dropped
    - ➢ **Source Address & Destination Address -** Size of 2 Bytes each and it specifies the IEEE 802.15.4 address of the endpoints



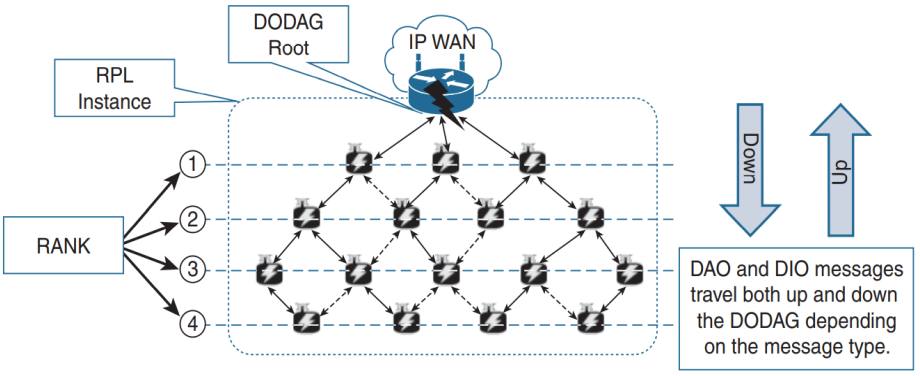
## RPL (ROUTING PROTOCOL FOR LOW POWER & LOSSY NETWORK)

- RPL is a distance-vector based **Routing Protocol for Low Power & Lossy Networks (LLNs)**
- Each node will act as  router in a RPL network, each node will examine the incoming packets and determine the next hop based in the informations available in the packet headers
- Due to the constrained nature of nodes RPL protocol can be operated under two modes
    - **Storing Mode -** Every node will contain a complete routing table of the RPL network, that is every node will know how to reach every other node in the network
    - **Non Storing Mode -** Only the border nodes (border routers) will contain a complete routing table of the RPL network. All other nodes will contain only the list of parents. All the packets are forwarded to the border node (border router) through the parents
- RPL is based on the concept of **Directed Acyclic Graph (DAG)** (A directed graph without any cycles). It builds **Destination Oriented Directed Acyclic Graph (DODAG)**. A DODAG is  a DAG which is routed to one destination. This destination will be a border router and is called as DODAG Root




- DODAG will contain **Upward Routes** (Routes towards the DODAG root) and **Downward Routes** (Routes towards the destinations)
- Upward Routes are discovered and configured using **DAG Information Object (DIO) Messages**. Nodes use DIO Message to determine their parents and finding best routes towards the DODAG root
- Downward Routes are discovered and configured using **DAG Advertisement Object (DAO) Messages**. Nodes use DAO messages to inform parents about their presence and to advertise about their descendants

- In Storing Mode, all the nodes will record the information propagated through the DAO Message, whereas in Non Storing Mode only the DODAG root will record the  information propagated through the DAO Message




- **Objective Function** is used for determining the rank of nodes using a set of defined metrics. **Rank** of a node tells us how close the node is to the DODAG root. Some of the common metrics used are
    - *Expected Transmission Count*
    - *Hop Count*
    - *Latency*
    - *Node State*
    - *Node Attribute*
    - *Node Energy*
    - *Throughput*
    - *Link Quality Level*
    - *Link Colour*

# MODULE 5

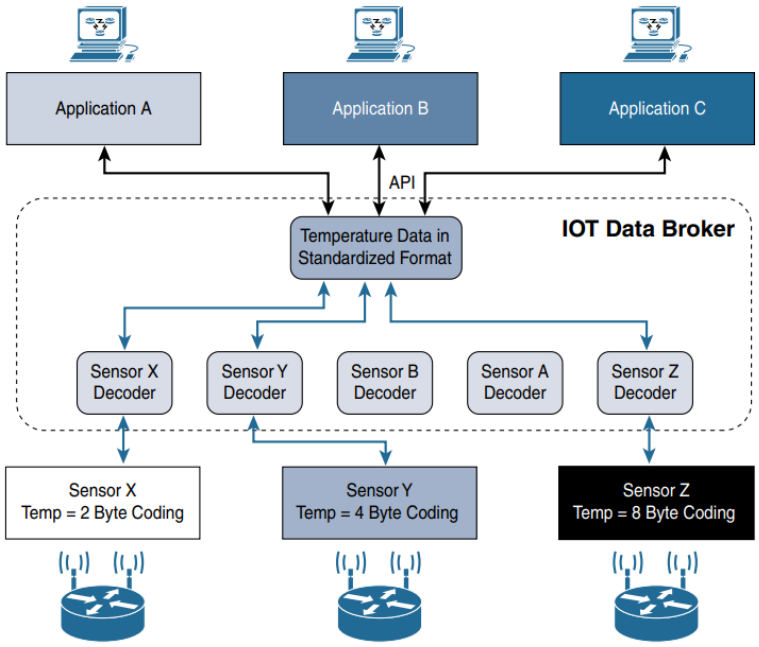
**Transmission Control Protocol (TCP)** or **User Datagram Protocol (UDP)** as 2 main transport layer protocols. The performance and scalability of IoT devices changes depending on which one of these is selected.

**TRANSMISSION CONTROL PROTOCOL vs USER DATAGRAM PROTOCOL**

- https://www.geeksforgeeks.org/differences-between-tcp-and-udp/

## APPLICATION LAYER NOT PRESENT

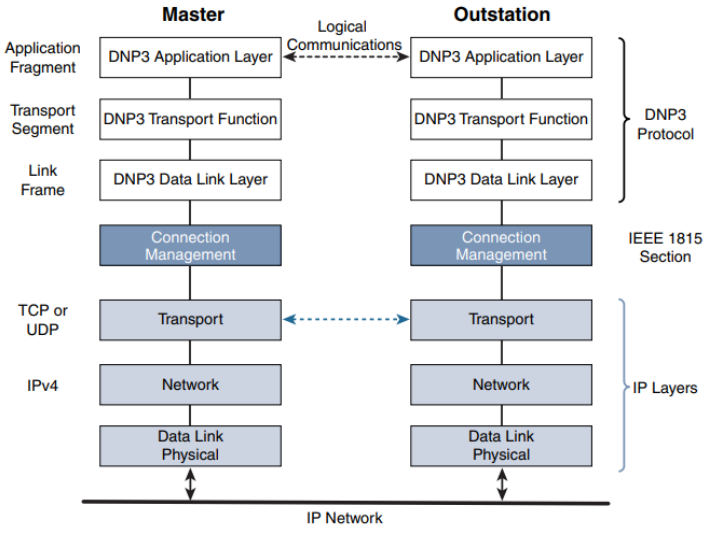
- Devices that fall under class 0 send and receive only a few bytes of data and they are severely constrained (low power consumption & low processing capabilities). Hence Implementing a complete network protocol stack is not feasible
- Due to this nature constrained devices such as sensors and actuators are being deployed without an application layer.
- Currently there is no standards for such deployment due to which each vendor will have their own implementation which gives rise to interoperability issues
- **IoT Data Brokers** are used to overcome this issue. **IoT Data Brokers** are middleware which standardises the outputs of sensors in order for applications to consume the data easily and effectively. *Ex. Temperature Sensor*
- Commercial companies also uses IoT Data Brokers as a source of revenue by supplying IoT related data through them




## SUPERVISORY CONTROL AND DATA ACQUISITION (SCADA)

- SCADA was initially implemented over serial links and later it was adapted to work over IP stack
- SCADA data collection and control of sensor in a remote manner

- SCADA  is a collection of protocols which are used for control and management of devices. Some those protocols are **DNP3, Modbus & IEC**
- **ADOPTION OF SCADA FOR IP (BASED ON DNP3)**
    - DNP3 is a SCADA protocols which works based on master-slave architecture
    - A master is a powerful computer which is present in a central facility. A salve is a remote device. DNP3 refers to slaves as outstations
    - An outstation will monitor and collect data about the environment it is present in. It transmits the data (when requested) and events to the master in an asynchronous manner
    - A master can control outstations by sending commands to them
    - **IEEE 1815-2012** specifies the standards for adopting DNP3 over the IP. It involves addition of a connection management layer as an intermediate between DNP3 protocol layers and IP protocol layers.
    - The DNP3 protocol layers are unaware of the presence of IP protocol layers in this configuration
    - IEEE 1815-2012 implementation will provide a native support of IP on DNP3

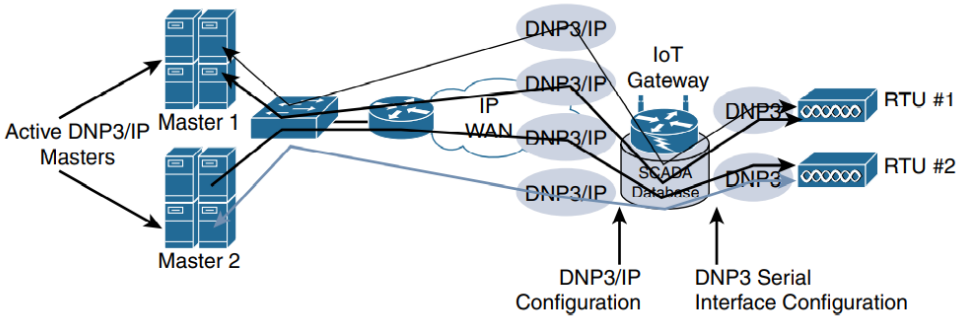



- **TUNNELING LEGACY SCADA OVER IP NETWORK**
    - Tunnelling involve transportation of legacy serial DNP3 over IP networks this can be achieve in the following ways

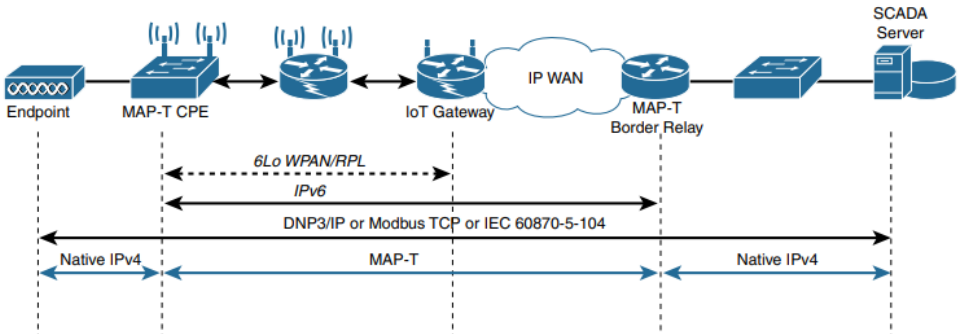
- ➢ **Using Raw Sockets**
- ➢ **Using Protocol Translation**

- **RAW SOCKET**
  - ➢

- **PROTOCOL TRANSLATION**
  - ➢ Protocol translation involves usage of intermediate devices for translating DNP3 into IP. These devices are called as **IoT Gateways**
  - ➢ IoT gateways are generally present closer to the edge of the network




- **SCADA OVER LOW POWER LOSSY NETWORK (LLN)**
  - LLNs are highly constrained. Due to this nature they use optimised communication through 6LoWPAN
  - 6LoWPAN supports only IPv6 networks where SCADA protocol works only IPv4. Due to this protocols must be translated when SCADA is used with LLN
  - This is implemented using MAP-T (Mapping of Address and Port Translation). This involves usage of two main component - **MAP-T CPE** (Customer Premise Equipment) and **MAP-T Border Relay**
  - MAP-T CPE is placed near the outstations (slave) and MAP-T Border Relay is place near the SCADA servers (master)
  - The network between the two components will be LLNs




# MODULE 6

**CoAP ()**

**MQTT (MESSAGE QUEUING TELEMETRY TRANSPORT)**

- Message Queuing Telemetry Transport is a li

# MODULE 7

Big Data is characterised by 6 V's - **Volume, Velocity, Veracity, Variety, Value, Variability**

### TYPES OF DATA ANALYTICS

1. **DESCRIPTIVE -** What is happening?
2. **DIAGNOSTIC -** Why did it happen?
3. **PRESCRIPTIVE -** What should you do about it?
4. **PREDICTIVE -** What will likely happen in the future?

**Perspective & Predictive** data analytics is resource intensive and complex but they are more valuable. **Descriptive & Diagnostic** data analytics are less compress but they are less valuable

### IoT DATA ANALYTICS CHALLENGES IN RDBMS

1. **SCALING PROBLEM -** When a large number of sensors and smart objects are used the amount of data produced and the rate at which the data is produced will be large. Traditional RDBMS systems are not efficient enough in these situation and hence results in performance issues
2. **VOLATILE DATA** - Data generated by sensors and smart objects have a dynamic structure. RDBMS can only handle data with fixed schema modifying the schema everything the structure changes is tedious

To address these challenges **NoSQL Database** is use for handling data in IoT

**CLASSIFICATION OF DATA**

| STRUCTURED DATA | UNSTRUCTURED DATA |
|---|---|
| Data will have a well defined schema | Data lacks a scheme or it is loosely define |
| Suitable for storing and processing using SQL DB | Suitable for storing and processing using NoSQL DB |
| Data can be understand and analysed easily | Data cannot be understand and analysed easily |
| E.g Data generated by application | E.g. Data generated by sensors |

| DATA IN MOTION | DATA IN REST |
|---|---|
| Data is in transit | Data is stationary |
| Cannot be stored and processed | Can be stored and processed |
| Real Time and Online | Stored and Offline |
| Traditional data analytics is not suitable. Stream analytics should be used | Traditional data analytics can be used |
| E.g | E.g |

**MACHINE LEARNING**

| SUPERVISED LEARNING | UNSUPERVISED LEARNING |
|---|---|
| Dataset is known and labelled | Dataset is unknown and not labelled |
| Number of target classes is known | Number of target class in unknown |
| Algorithm is complex | Algorithms is simple |
| Predictions are more reliable and accurate | Predication are less reliable and accurate |
| Suitable for offline and stored data | Suitable for online and realtime data |
| Ex. Regression | Ex. Clustering |

| LOCAL LEARNING | REMOTE LEARNING |
|---|---|
| Data is collected and process in the node itself (edge) or in the gateway (fog) | Data is collected and sent to a central computing system (cloud) |

**APPLICATION OF MACHINE LEARNING IN IoT**

- **Monitoring**
- **Behaviour Control**
- **Operations Optimization**
- **Self-Optimization & Self-Healing**

**NoSQL DATABASE**

- Generally schemaless or have a dynamic schema
- Highly scalable (Both vertical & horizontal scalability)
- Supports multiple data models
- Handles realtime and volatile data efficiently
- Supports distributed data storage
- **Types of NoSQL Database**
    - Key-Value Storage
    - Document Storage
    - Graph Storage
    - Wide Column Storage

# MODULE 8

**HADOOP**

- Hadoop is an open source framework which is used for storing and processing big data efficiently in a distributed fashion. The three important components of Hadoop are
    - **Hadoop Distributed File System (HDFS)**
        - ➢ Name Node & Data Node (Diagram)
        - ➢ Block & Rack Awareness Algorithm (Diagram)
        - ➢ Replication Factor
    - **MapReduce**
        - ➢ Mapping Phase
        - ➢ Reducing Phase
        - ➢ Shuffle & Sort Phase
    - **Yet Another Resource Negotiator (YARN)**
        - ➢ Definition

**HADOOP ECOSYSTEM**

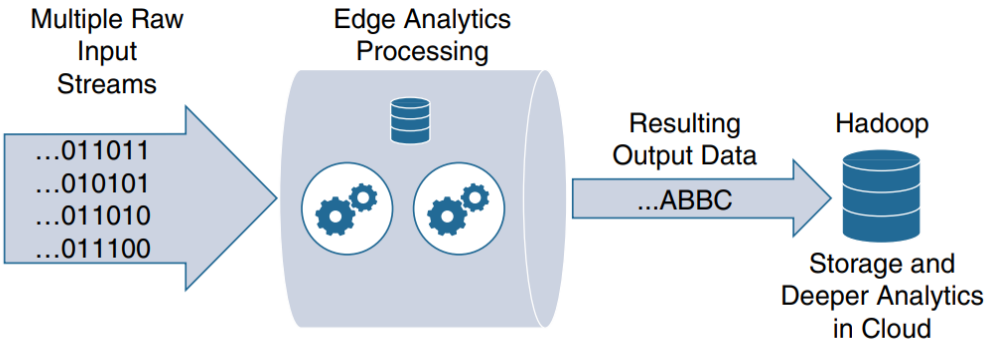
- Hadoop Ecosystem is a collection of more than 100+ software modules which are used for managing various data lifecycles Data Collection, Data Preprocessing, Data Visualization and Data Storage
- **APACHE KAFKA**
    - Apache Kafka is an open source distributed publisher-subscriber based message handling system. It is fast and highly scalable
    - The main role of message handling systems is to receive data or messages from various sources and deliver them to the streaming system. EX Apache Spark Streaming
    - Apache Kafka prepares the data received for processing and analysis but streaming systems
    - The main components of apache kafka are Topic & Data Broken. A produce will writer the data to a topic and consumer will read data from a topic
    - Due to the distributed nature of apache kafka it can handle multiple produces and consume concurrently
- **APACHE SPARK**
    - Apache Spark is an open source distributed in-memory data analysis platform. It helps in accelerating the performance of process in hadoop ecosystem
    - In MapReduce the data is read from the disk and processed then it is written back to the disk. Accessing the disk frequently will increase the latency of the application
    - To overcome the problem of latency spark uses high speed memory for storing and processing data
    - Real Time streaming data can be process using apache sparks software module called as apache spark streaming
    - Apache spark streaming can utilise real time data from apache kafka. It partitions the data into micro batches called and DStreams
    - These DStreams will then be processed using the spark processing engine

**EDGE STREAMING ANALYTICS**

- Edge devices contain a lot of sensors and they produce large volumes of data at a high rate. Due to this nature the data cannot be propagated to cloud for analysis everytime because it will consume a lot of bandwidth and it is inefficient. Hence analytic must be performed directly on the edge
- The edge is highly distributed due to which any analytics performed must be highly coordinated and structured. The important stages of edge analytics are
    - **Raw Input Stream**

The Raw Input Stream is data generated by sensors which is subjected to analytics at the APU

- **Analytical Processing Unit (APU)**
  The APU applies various filters and functions to the raw data to extract useful information. Some common operation performed on APU are
  1. **Filter -** Extract important information from the raw data
  2. **Transform -** The extracted data is converted to suitable format
  3. **Time -** Flowing data must be ordered based on time to process
  4. **Correlate -** Multiple streams and historical data can be combined to perform analysis
  5. **Match Pattern** - Pattern matching can be used to get deeper insights of data. Machine learning can be used to make it more efficient
  6. **Business Insights -** From the output obtained suitable business choice can be made
- **Output Stream**
  The Output Stream is the result generated by APU. It is utilised to make business choices and then it is stored in the cloud for further analysis




**NETWORK STREAMING ANALYTICS**

# MODULE 9

# MODULE 10