UNIT-2:

* Machine-to-Machine Communications
- Difference b/w IoT & M2M
- Interoperability in IoT
- Introduction to Arduino Programming
- Integration of Sensors and Actuators with Arduino.

UNIT-2:

*M2M Communication

→ It refers to the direct exchange of data b/w devices (or) machines without human involvement.

→ It enables connected devices to share information & make decisions automatically.

→ It means devices talking to each other directly without human involvement.

→ they share data & make decisions automatically,

→ It is like machines chatting with each other

Ex: - Environment monitoring
- Smart Agriculture

Features

→ Automatic Data Exchange
→ Real-Time Monitoring
→ Remote control

Example:

→ A gas sensor measure air quality, sends data to Raspberry Pi.

→ A Smart meter sends electricity usage data directly to the utility company

Applications

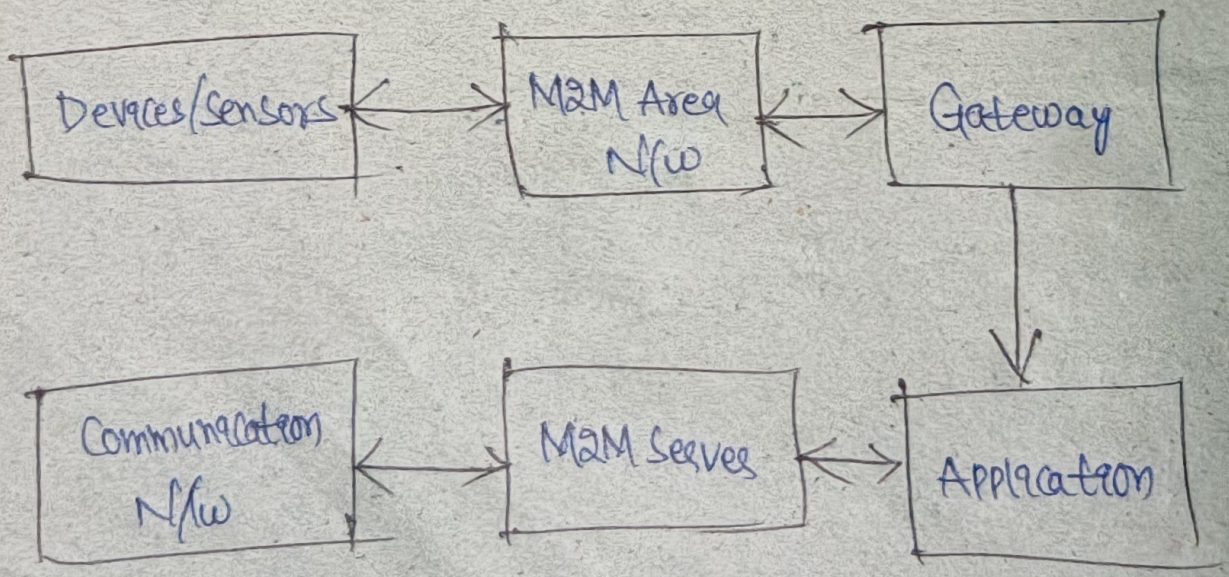
- Security
- Tracking & tracing
- Healthcare
- Payment
- Remote control
- Smart Cities
- Agriculture
- Manufacturing.

M2M architecture

→ M2M architecture refers to the framework.
→ It allows devices to communicate withe each other without human involvement.
→ This architecture is commonly used in IoT applications.
→ It is a framework that enables devices to exchange data.

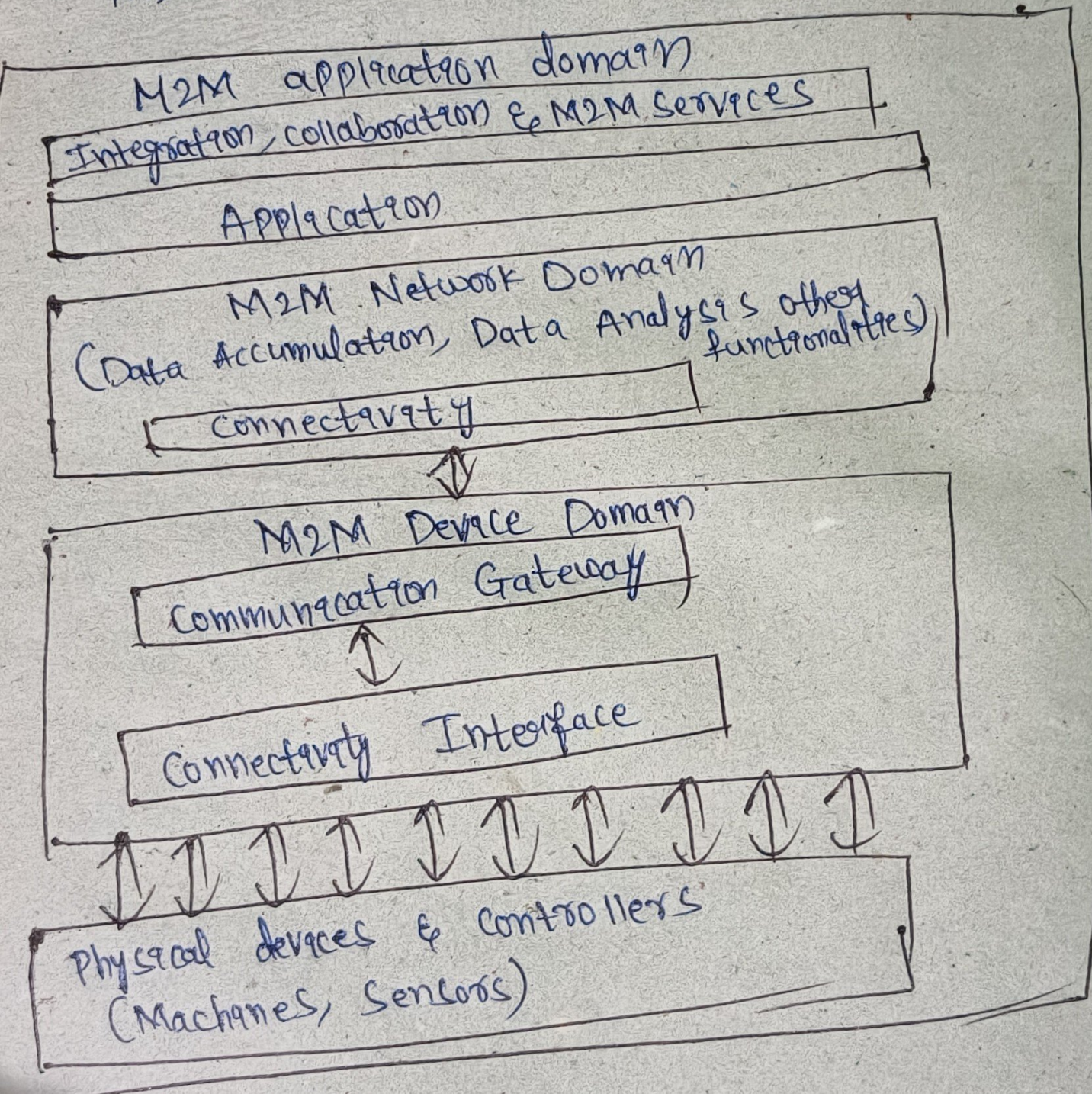
Components of M2M components

1. Devices/Sensors: these are physical devices or sensors that collect data.
   Ex: Temperature sensor, GPS device
2. M2M Area N/w: The N/w that connects devices.
   Ex: Bluetooth, Zigbee, Wi-Fi
3. Gateway: Acts as bridge b/w M2M area n/w & the communication n/w.
4. Communication N/w: the medium for data transmission
   Ex: Ethernet, Satellite
5. M2M Servers: The central system that processes & manages the data.
6. Application: The end-user interface or system



→ M2M Architecture consists of 3 domains:

- M2M device domain
- M2M Network domain
- M2M application domain



* Difference b/w IoT and M2M

IoT	M2M
1). IoT stands for "Internet of things"	1). M2M stands for "Machine to Machine".
2). It is highly scalable.	2). It is less scalable.
3). It uses IP protocols.	3). It uses non-IP protocols.
4). It is s/w-based.	4). It is h/w-based.
5). High Interoperability.	5). Limited Interoperability
6). used cloud platforms.	6). Doesn't neccesarly use the cloud.
7). It uses internet and cellular networks.	7). It uses either an internet or non-internet connection.
8). used for automation.	8). Mainly used for automation.
9). Two-way communication.	9). one-way communication.
10). It requires internet connection.	10). doesn't require internet.
11). Structured & Unstructured data.	11). Structured data.
12). It applies to sensors.	12). It applies to machines
13). Cloud communication	13). Point-to-Point communication
14). Scope is large	14). Scope is limited.
15). Network Connection	15). Point-to-Point Connection.
16). Protocols: HTTP, FTP	16). Protocols: MQTT, XMPP

Interoperability

Inter + Operational ability

⇒ the ability of IoT System & components
- to interact with each other
- & shares data is known as "Interoperability"

⇒ Ability to interact with each other
- & share data in IoT System & components is "Interoperability".

→ It helps in exchange of data among devices.

→ It makes the devices work collaboratively.

→ It is the key feature of IoT.

→ If it is not there, then many benefits of IoT cannot be used.

Example: Bus application, that can find a route (short route)

Inorder to find the route, it takes help from Interoperation

↓

Interoperation with "Traffic Monitoring Service" of the city try to find an optimal route
↳ short

↓

It ~~try~~ to obtain less conjugate route (less traffic route)

Types

1. Syntatic Interoperability: Without knowing anything it will perform operation.

2. Semantic Interoperability: It will perform operation after understanding the meaning.

3. Technical Interoperability:

→ It does not require knowledge of data format.

→ But it need knowledge to establish the communication.

4. IoTivity: IoT + activity

→ It is offered by open connectivity foundation.

→ It is open-source.

- It allows complete connection b/w devices

* Introduction to Arduino Programming

→ Arduino Programming plays a crucial role.
→ It helps in building IoT Projects.
  - by interfacing sensors, actuators
  → We can develop Projects
→ Arduino is an Open-Source Platform.
→ It consists both hardware & S/w to write & upload code.
→ It is a simple environment to write, compile & upload code to Arduino boards.
→ It uses language similar to C/C++
→ It supports libraries.
  → It has microcontroller.

Basic Structure of Arduino code:

- setup function( ) → runs once when the board is reset Powered on
- loop( ) → repeats continuously.
  → contains the main logic

Example: LED Blinking

```
void setup()
{
  pinMode(13, OUTPUT);
}
void loop()
{
  digitalWrite(13, HIGH); → LED turns on
  delay(1000); → Pauses for 1 sec
  digitalWrite(13, LOW); → turns off
  delay(1000);
}
```

→ It is a small hardware device..
- Open Source
- Easy to use
- Versatile
- Large Community

Applications

→ Environmental Monitoring
→ Home Automation
→ Smart Agriculture
→ Wearable Devices

Arduino Boards

→ there are various types of Arduino boards like:

- Arduino Uno
- Arduino Mega
- Arduino nano etc..

→ Most Popular is **Arduino Uno**

→ It has 14 digital i/p & o/p pins
- 6 analog inputs
- a USB connection
- a Power jack
- & a reset button.

Shields: Shields are boards that are connected on top of arduino for improving its features

→ An Arduino program begins with **setup()**
- to initialize settings.

→ After the setup, loop() function is used.
- to run infinitely.

example

```
void setup()
{
  Serial.begin(9600);
  Serial.println("Hello world");
}
void loop()
{
}
```

Installation

1. Go to website (https://docs.arduino.cc/software/ide-v2)
2. Download it.
3. Run the download file.
4. Agree the Lacences
5. Click on next.
6. Installation process goes on
7. Next → Accept → Next → Finish

* Integration of Sensors and Actuators with Arduino

→ It refers to the process of connecting input devices (sensors) & output devices (Actuators) to the arduino board.

→ It means connecting sensors & actuators to an Arduino board.

- So they can interact with the real world.

Sensor: Input device (gas sensor, temperature sensor --)
- Collect data

Actuator: Output device (motor)
- Perform actions

→ After connecting to Arduino it can:
- Read input
- Process it
- Perform action

Steps to integrate

Step-1: Connect Sensor to Arduino
→ Sensors are input devices
ex: LDR → A0

Step-2: Connect Actuator to Arduino
→ Actuators are output devices.
ex: LED → D9

Step-3: Initialize Pins in Arduino
→ In setup(), define sensor pin as Input; actuator pin as output.

```
pinMode(sensorPin, INPUT);
pinMode(actuatorPin, OUTPUT);
```

Step-4: Read Sensor data
→ use functions like analogRead() or digitalRead()
- to get data from sensor

```
int value = analogRead(A0);
```

Step-5: Apply Logic in Code
→ Based on sensor reading, apply if-else logic

```
if(value<500){
  digitalWrite(ledPin, HIGH);
}
```

Step-6: Control Actuator
→ Use digitalWrite() or analogWrite()
- to control actuators
(like motors, LED)

Step-7: Upload Code and Power Arduino

→ Use USB to upload code

→ Provide Power via USB or ~~battery~~ external adapter.

Sensor
↓
Arduino Input Pin
↓
Process in Code
↓
Arduino Output Pin
↓
Actuator

Flow Diagram of Process

5) Explain about integration of sensors and actuators with examples?

→ Sensors and actuators are essential components in IoT systems.
→ Sensors collect data from the environment.
→ Actuators perform actions based on that data.
→ Integration involves connecting these components to a microcontroller or processing unit (Arduino or Raspberry Pi) to work together

Working

→ Sensors collect data (eg: temperature, humidity)
→ the data is sent to a micro controller.
→ the unit processes the data and makes a decision.
→ Actuators perform actions based on the processed data.

6

Examples of Sensor & actuator Integration:

1. Automatic irrigation System.

- Sensor: Soil Moisture Sensor.
- Actuator: Water Pump
- Working:
  → the soil moisture sensor measures the moisture level in the soil.
  → If the moisture level is low, the microcontroller activates the water pump to irrigate the plants.
  → once the desired moisture level is reached, the pump stops.

2. Automatic Hand Sanitizer Dispenser:

- Sensor: Ultrasonic Sensor (to detect hand presence)
- Actuator: Pump motor
- Working:
  → the sensor detects a hand near the nozzle.
  → the microcontroller activates the pump to dispense sanitizer
  → the pump stops once the hand is removed.

3. Automatic Door Opening System:

- Sensor: Motion Sensor (PIR)
- Actuator: Door Motor
- Working:
  → the PIR sensor detects the presence of a person.
  → the microcontroller sends a signal to open the door.
  → After a few seconds, the door closes automatically.

4. Automatic Water Level Controller:

- Sensor: Water Level Sensor
- Actuator: Water Pump
- Working: → the sensor measures the water level in a tank
  → If the water level is low, the pump turns on to fill the tank.
  → once full, the pump automatically turns off.