Emerging issues in Computer science

Week 7: Internet of Things and smart systems

Lecturer: Ikwap Flavia Agatha

Lecture Learning Out come

At the End of this lesson, we will be able to;

- Understand IOT, Characteristics and IOT key Features
- Understand the Evolution of IOT
- Understand the Functional Blocks of IOT Systems
- Understand IOT Communication Model
- Understand IOT Enabling Technologies
- Understand IOT Levels and deployment
- Understand Domain Specific IOT
- Understand Smart systems, Sensors and Actuators

Introduction to the Internet of Things (IOT)

- The Internet of Things (IOT) refers to a rapidly growing network of physical objects—commonly referred to as "smart" devices—that are embedded with sensors, software, and other technologies to connect and exchange data over the Internet. These objects are uniquely identifiable and capable of sensing their environment, performing tasks, and communicating both with each other and with external systems.
- IOT goes far beyond simple connectivity; it enables intelligent interaction between devices, allowing them to share data and automatically respond to real-world events. The result is a more efficient and automated world, where processes can be triggered and services delivered with minimal human intervention.

Key Features of IOT

- **Device Interconnectivity**: Devices can interact and exchange information via the Internet.
- **Environmental Awareness**: Devices can sense their surroundings and take context-based actions.
- **Automation**: IOT enables processes to be triggered automatically, reducing the need for human input.
- **Real-Time Communication**: Devices communicate and respond in real time, enhancing operational efficiency.
- **Service Integration**: Systems deliver services based on real-world data and scenarios.

Origin and Evolution of IOT

- **Early Concepts and Innovations**
- The idea of connected devices dates back as early as **1832** with the **electromagnetic telegraph**, allowing the transmission of messages using electric signals.
- In the **1980s and 1990s**, experimental devices began to appear:
- **1980**: A Coca-Cola vending machine at **Carnegie Mellon University** was internet-enabled to report stock and temperature.
- **1990**: **John Romkey** created the first Internet-connected toaster using TCP/IP.
- **1991**: A **webcam** at the University of Cambridge monitored a coffee pot—streaming images to reduce trips for empty coffee.

Origin and Evolution of IOT

- **The Term "Internet of Things"**
- The term **"Internet of Things"** was coined in **1999** by **Kevin Ashton** during a presentation at Procter & Gamble. He proposed using **RFID technology** to improve supply chain management

(**Radio Frequency Identification (RFID)** is a technology that uses radio waves to automatically identify objects, animals, or people). Though his focus was on RFID, his terminology and vision laid the groundwork for today's IP-based IoT systems.

Origin and Evolution of IOT

- **Milestones in IOT Development**
- **2000**: LG Electronics unveiled a smart refrigerator that allowed online shopping and video calls.
- **2005**: A rabbit-shaped robot was introduced, capable of providing weather, news, and stock updates.
- **2008**: The first International IoT Conference in Switzerland brought together researchers from 23 countries.
- **2011**: Gartner’s Hype Cycle acknowledged IoT as an emerging technology, especially with the arrival of IPv6, essential for scalable connectivity

Origin and Evolution of IOT

- **Modern Adoption and Industry Impact**
- Since the 2010s, IOT has transitioned from concept to reality:
- Major companies like Google, Apple, Samsung, Cisco, and General Motors have invested in IoT products, ranging from smart glasses and thermostats to autonomous vehicles.
- As of 2018, IOT remained a dominant force on Gartner’s Hype Cycle, signaling its movement from emerging innovation to widespread, productive technology.

Characteristics of IOT

- IOT systems possess several key characteristics that define their behavior, performance, and integration abilities.
- **Dynamic and Self-Adapting:** IOT devices can automatically adjust their operations based on changes in the environment or user context. Example: A surveillance system that adapts its camera angles or motion sensitivity based on time of day or detected activity.
- **Self-Configuring:** Devices can automatically configure themselves to join the network and function in coordination with other devices. Simplifies deployment and supports scalability in large systems.
- **Interoperable Communication Protocols:** IOT systems support multiple communication standards and protocols, enabling seamless communication between diverse devices and infrastructure. Essential for building flexible, vendor-independent ecosystems.

Characteristics of IOT

- **Unique Identity:** Every IOT device has a unique identifier (often an IP address) to distinguish it on the network. This allows targeted control and monitoring of individual devices.
- **Integrated into Information Network:** Devices are part of larger networks that enable them to share data and interact with other systems.
- **Advantage**: Enhances coordination, decision-making, and data-driven automation.
- **Scalability**: IOT systems are designed to handle growth in the number of devices, users, and data volume. Crucial for large-scale applications like smart cities or industrial automation.

Characteristics of IOT

- **Connectivity:** Reliable and persistent connection is a core characteristic, allowing devices to transmit and receive data. Connectivity technologies include: Wi-Fi, Bluetooth, Zigbee, LTE, 5G, etc.
- **Architecture**: IOT architecture defines how devices, networks, cloud services, and applications interact and exchange information. Architectures common layers include: Perception layer (sensors), Network layer (communication), and Application layer (services).
- **Intelligence and Identity:** Devices can process information locally or through cloud services, enabling intelligent responses. Example: Smart thermostats learning user preferences to optimize heating schedules.

Objectives of IOT

- **Connecting Things**: Linking devices like appliances, vehicles, and machinery to the internet.
- **Exchange of Data and Information**: Facilitating real-time sharing of sensor data, device status, and user inputs.
- **Sensing, Processing, Control, Actuation, and Monitoring**: Enabling devices to interact with the physical world and make autonomous decisions.
- **Providing Services**: Delivering value-added services such as automation, reporting, analytics, and user interaction.
  - Example: A smart irrigation system that adjusts water usage based on soil moisture data.

Functional Blocks of IOT Systems

An IOT system consists of several core components that enable it to perform its functions effectively:

- **Device Block:** Includes hardware for identification, sensing, actuation, control, and monitoring.

  Examples: Sensors, RFID tags, cameras, actuators.

- **Communication Block**: Manages data transmission between devices, networks, and cloud systems.

  Technologies: Wi-Fi, Zigbee, LoRa, 5G, etc.

- **Controller Services Block:** Offers services like: Device monitoring, Device control, Data publishing, Device discovery

- **Management Block**: Handles system governance including: Resource allocation, Configuration management**,** Security and privacy**,** Performance monitoring

IOT Functional Blocks

These blocks define the core components of an IOT system and how they interact to collect, transmit, store, analyze, and present data.

- **Security Block**: Ensures the security and integrity of the IOT system.
- Functions Include:
  - Authentication – Verifies the identity of users or devices.
  - Authorization – Determines who can access what.
  - Message & Content Integrity – Protects data from being altered during transmission.
- Data Security – Encrypts and secures stored and transmitted data.

IOT Functional Blocks

- **Application Block**: Serves as the user interface for interacting with the IOT system.

  Features: Control and monitor IOT devices, Visualize system status, Access and analyze processed data Example: A smartphone app to control a smart thermostat.

- **Resource Block**: Software components that interact with hardware (sensors and actuators).
- Functions: Access sensor data, Process and store local data, Control actuators

  Example: Embedded software on a smart irrigation controller.

- **Database Block**: Stores data generated by IOT devices. These include: *Local* – Data stored on the device or nearby gateway. *Cloud* – Data stored remotely for scalability and easy access.

  Role: Supports data history, trend analysis, and long-term insights.

IOT Functional Blocks

- **Web Services Block**: Facilitates interaction between devices, applications, and databases. Acts as middleware that connects different system components.
- **Protocols**: **HTTP/REST** – Traditional web protocols for simple communication.
  - **WebSockets** – Real-time communication protocol for interactive applications.
- **Analysis Component**: Processes and analyzes IOT data by Turning raw sensor data into meaningful insights, Generating charts, reports, alerts, or AI-driven action. Example: Predictive maintenance based on vibration analysis of machinery.

IOT Functional Blocks

- **Interaction & Communication Block**: Enables interaction between IoT devices and external systems. These include: Network interfaces (Wi-Fi, Zigbee, etc.), APIs and communication protocols
- **Internet Interaction Block**: Handles connectivity and data transfer over the Internet. For example; Network management**,** Cloud service integration
- **User Interface Block**: Allows users to access and control IoT services these include: Mobile apps, Web dashboards, Voice assistants

IOT Communication Models

Push-Pull Model: mostly used for Batch processing, load balancing.

How It Works:

- **Producers** push data into a **queue**.
- **Consumers** pull data from the queue when ready.

Exclusive Pair Model: used to secure point-to-point communication, such as a smart watch syncing to a specific phone. A **dedicated two-way communication** link between a specific client and server. Often used in device pairing (like Bluetooth).

IOT Communication Models

IOT communication models: define how data is exchanged between devices and systems. Here are the two main models:

- **Request-Response** Model commonly used in Web-based interfaces, configuration requests. **How It Works**:
  - A **client** sends a request to a **server**.
  - The server processes the request and sends back a response.
- **Stateless**: Each request is independent of others e.g : A web browser (client) requests a webpage from a server using HTTP.

Publish-Subscribe Model: used in event-driven systems, real-time alerts: Key Players: **Publisher** – Sends data to a “topic”. **Broker** – Manages topics and routes data. **Subscriber** – Listens for updates to a topic.

IOT Enabling Technologies

- The Internet of Things (IoT) relies on a combination of technologies that allow devices to connect, communicate, process, and analyze data intelligently and securely. Here’s a breakdown:
- **Wireless Sensor Networks (WSNs)** is a group of spatially distributed sensors that monitor and record physical or environmental conditions (like temperature, humidity, or motion) and transmit the data wirelessly.
- **Role in IOT**:
  - Act as the “eyes and ears” of IOT systems.
  - Enable real-time monitoring of environments.
- **Examples**: Smart agriculture (soil moisture sensors), industrial automation, smart homes.

IOT Enabling Technologies

- **Cloud Computing,** the role it plays: Stores large volumes of IOT data.
  - Hosts IOT platforms and applications.
  - Offers scalability and accessibility from anywhere.
- **Examples**: AWS IOT Core, Microsoft Azure IOT Hub, Google Cloud IOT.
- **Big Data Analytics**: The process of analyzing huge datasets to uncover patterns, trends, and insights. Key role:
  - Helps make sense of data generated by billions of connected devices.
  - Supports decision-making, prediction, and automation.
- **Examples**: Predictive maintenance in factories, traffic analysis in smart cities, Embedded systems

IOT Enabling Technologies

- **Security Protocols and Architectures**: Tools and frameworks designed to protect ata, devices, and communication in IOT systems. Main function: Ensures data privacy and integrity.
- Protects devices from hacking or unauthorized access.
- **Communication Protocols**: Rules and standards for data exchange between IoT devices and systems. Main purpose: Enable interoperability between different devices. Define how data is transmitted, received, and processed.
- **Common Protocols**:
  - **MQTT** (lightweight messaging protocol)
  - **CoAP** (for constrained devices)
  - **HTTP** and **WebSocket** (web-based communication)
  - **Zigbee**, **Bluetooth**, **LoRa**, **NB-IOT** (wireless protocols)

IOT Enabling Technologies

- **Web Services**: Software systems designed to support interoperable machine-to-machine interaction over a network.
- Allow devices to interact with applications and platforms via APIs.
  - Support remote control, monitoring, and data retrieval.
- **Examples**: RESTful APIs for device management, JSON/XML data exchange.
- **Mobile Internet and Semantic Search Engines**
- **Mobile Internet**:
  - Enables remote access and control of IoT devices through smartphones or tablets.
  - Crucial for applications like smart home apps or wearable health devices.
- **Semantic Search Engines**:
  - Use AI to understand the meaning of data rather than just keywords.
  - Help in making IOT data more accessible, searchable, and useful for applications and users.

IOT Levels And Deployment Templates

System IOT Level 1: Single Node – Standalone System: Suitable for low-cost, low-complexity scenarios. The system processes small data volumes, and analysis is not computationally intensive

- **Architecture**: A single node/device handles everything:
  - Sensing/Actuation
  - Data storage
  - Analysis
  - Application hosting
- **Deployment Template**: All components are embedded in a single device. **Example**: Home Automation System
  - E.g., a smart light system that senses motion and turns lights on/off without any cloud dependency.

IOT Levels And Deployment Templates

IOT Level 2: Single Node with Cloud Storage and Local Analysis: Suitable for scenarios involving large data volumes, but local processing is sufficient for most tasks.

System Architecture:

- A single device handles sensing/actuation and local analysis.
- Data is sent to the cloud for storage.
- Application is cloud-based.

Deployment Template:

- Lightweight edge device + cloud backend for storage and application.

Example: **Smart Irrigation System**

- Soil moisture sensor processes basic logic locally (e.g., if dry → activate water pump), but logs data in the cloud.

IOT Levels And Deployment Templates

IOT Level 3: Single Node with Cloud Analysis: Suitable for solutions with large data volumes and heavy computational analysis (done in the cloud).

System Architecture:

- One device performs sensing/actuation.
- Cloud handles data storage, analysis, and application hosting.

- **Deployment Template**: Sensor → Cloud pipeline with minimal local processing. Example: Package Handling Tracker
- Uses an accelerometer to detect mishandling during shipping; raw sensor data is uploaded to the cloud and analyzed for impact patterns.

IOT Levels And Deployment Templates

IOT Level 4: Multiple Nodes with Local & Cloud Processing

System Architecture:

- Multiple nodes, each performing local analysis.
- Data from nodes is sent to the cloud for aggregation.
- Applications and visualization are cloud-based.
- May include observer nodes to subscribe to cloud updates.

Deployment Template:

- Distributed nodes (edge devices) + cloud hub + optional observers.
- Best for systems that need scalability with multiple sources of input.
- Noise Monitoring System
- Nodes with sound sensors locally measure decibel levels, send summaries to the cloud for mapping noise pollution.

IOT Levels And Deployment Templates

IoT Level 5: Multiple End Nodes with Coordinator Node: Fits Wireless Sensor Network (WSN) models, with heavy data loads and complex analytics.

System Architecture:

- Multiple end nodes handle sensing/actuation.
- A coordinator node collects data from these and uploads to the cloud.
- Cloud handles storage, analysis, and application.

Deployment Template:

- Hierarchical architecture: End nodes → Coordinator → Cloud. Example: Forest Fire Detection System

- Temperature, smoke, or flame sensors send data to a central node, which uploads it for AI-based fire prediction.

IOT Levels And Deployment Templates

IOT Level 6: Multiple Independent Nodes with Centralized Control:

Best for real-time monitoring and control across distributed environments.

System Architecture: Multiple independent nodes send data directly to the cloud.

- A centralized controller:
  - Monitors node status.
  - Sends control commands.
- Cloud handles: Data storage, Advanced analytics, Application visualization

Deployment Template:

- Full cloud-centric system with decentralized sensing and centralized control.

- Example: Weather Monitoring System
  - Independent nodes measure temperature, humidity, pressure, etc., send data to the cloud; a central system triggers alerts or controls connected systems (e.g., turning on irrigation in dry areas).

Domain Specific IOTS

1. Home Automation: IOT in home automation improves convenience, energy efficiency, and safety.

Smart Lighting: Adapts to ambient conditions, Automatically switches on/off or dims lights.

- **Smart Appliances**: Allows remote monitoring and control. Provides status updates (e.g., smart fridge, washing machine).

Intrusion Detection: Uses PIR (Passive Infrared) and door sensors. Integrated with security cameras.

Smoke/Gas Detectors: Detects smoke (early fire warning), Alerts can trigger fire alarm systems, Gas detectors identify harmful gases like **CO**.

e) Surveillance: Cloud-based storage for video feeds, Scalable and accessible for public safety.

f) Emergency Response: Fire, gas, and water leak detectors, Generates alerts to minimize damage to infrastructure.

Domain Specific IOTS

2. Smart Cities: IOT in cities enhances public services, safety, and infrastructure management.

a) Smart Parking: Detects empty slots, Sends data to a smart app for drivers to find spaces easily.

b) Smart Lighting (Public Infrastructure): Applied to roads, parks, and buildings. Automatically adjusts brightness and saves energy.

c) Smart Roads: Equipped with sensors and Provide real-time info on traffic, road conditions, and accidents.

d) Structural Health Monitoring: Uses vibration sensors, Monitors buildings, bridges for integrity and early fault detection.

e) Surveillance: Cloud-based storage for video feeds, Scalable and accessible for public safety.

f) Emergency Response: Fire, gas, and water leak detectors, Generates alerts to minimize damage to infrastructure.

Domain Specific IOTS

Environment: IOT helps monitor and manage environmental parameters effectively.

a) Weather Monitoring: Collects data from multiple sensors (temperature, pressure, etc.), Sends to cloud for analysis and visualization.

b) Air Pollution Monitoring: Detects gases: CO, CO_2, NO, NO_2, etc. Helps in pollution control decision-making.

Retail: IoT boosts operational efficiency and customer experience in retail.

a) Inventory Management: Uses RFID for real-time inventory tracking.

b) Smart Payments: Contactless transactions using NFC, Bluetooth.

c) Smart Vending Machines: Monitors internal status (stock, temperature, faults).

Sends data to cloud for predictive maintenance.

Domain Specific IOTS

c) Noise Pollution Monitoring: Monitors noise levels across a city. Aggregated in the cloud to create noise maps.

d) Forest Fire Detection: Detects early signs of wildfire: Prevents large-scale damage to property and natural resources.

e) River Flood Detection: Sensors monitor water levels and flow rates.

Provides early warnings to minimize flood damage.

Energy: IOT is revolutionizing how we manage and distribute energy.

a) Smart Grids: Real-time monitoring of power generation and distribution. Helps utilities manage energy more efficiently. Detects faults and predicts failures.

b) Renewable Energy Systems: IoT at interconnection points tracks energy output. Controls for wind turbines regulate voltage and manage power feeds.

Domain Specific IOTS

Logistics: IOT enables real-time tracking and optimization in logistics.

a) Route Generation & Scheduling: Cloud-based systems generate optimal delivery routes.

b) Fleet Tracking: GPS-enabled systems for real-time vehicle location monitoring.

c) Shipment Monitoring: Monitors conditions like temperature and humidity. Ensures perishable goods are safely transported.

Agriculture: IoT enhances productivity and resource management in farming.

a) Smart Irrigation: Measures soil moisture levels. Activates irrigation only when needed.

b) Greenhouse Control Automates environment (temperature, humidity, light) for crop growth.

Domain Specific IOTS

Industry: IOT improves operational efficiency and safety in industrial environments.

a) Machine Diagnosis & Prognosis: Predicts faults and schedules maintenance.

b) Indoor Air Quality Monitoring: Ensures air quality meets safety standards.

Health and Lifestyle: IOT empowers individuals and healthcare providers with real-time data.

a) Health & Fitness Monitoring: Tracks vitals (heart rate, steps, sleep). Data stored and analyzed in cloud apps.

b) Wearable Electronics: Smart watches, fitness bands, health trackers. Often sync with smartphones and cloud for analysis.

What is a Smart Object?

- A **Smart Object** is any physical object embedded with **technology** (sensors, actuators, communication, processing) that enables it to:
- **Sense and interact** with its environment.
- **Communicate** with other devices or external systems.
- **Process data** and make decisions.
- These objects form the foundation of the Internet of Things (IoT) by transforming ordinary items into intelligent, interconnected systems.

Core Building Blocks of Smart Objects (IOT Enablers)

- Smart objects typically consist of the following four components:

1. Sensors and Actuators

- **Sensors** collect data from the environment (e.g., temperature, light, motion).
- **Actuators** perform actions based on commands (e.g., turn on a motor, unlock a door).
- A smart object can contain **multiple sensors and/or actuators**, depending on the application.

2. Processing Unit

- This is the **“brain”** of the smart object.

Core Building Blocks of Smart Objects (IOT Enablers)

It:

- Acquires and processes sensor data.
- Makes decisions (e.g., using logic or AI algorithms).
- Sends control signals to actuators.
- Manages communication and power systems.

- Usually implemented via **microcontrollers** or **embedded systems**.

3. Communication Module

- Enables the smart object to:
  - Exchange data with other devices.

Core Building Blocks of Smart Objects (IOT Enablers)

- Connect to the internet or local networks.
- Can be wired (Ethernet, USB) or wireless (Wi-Fi, Bluetooth, Zigbee, LoRa, NB-IoT).
- Communication is crucial for interconnectivity in IOT systems.

4. Power Source

- Provides energy to run sensors, processors, and communication modules.
- Can be:
  - Battery-powered
  - Solar-powered
  - Powered via mains

Note: The communication module typically consumes the most power.

Why Smart Objects Matter in IoT

- Smart objects are the **core components** of IOT systems. They:
- Enable **real-time interaction** with the physical world.
- Provide the **data** needed for automation, analytics, and decision-making.
- Support **autonomous systems** by acting on insights locally or via cloud processing.

Sensor Network

- A **sensor** is a device that collects data from a physical environment and converts it into electrical signals for processing, measurement, and analysis.
- The physical phenomena being measured could include temperature, pressure, light, sound, motion, position, flow, humidity, radiation, and more.
- A **sensor network** is a system composed of multiple sensors, computational components, and communication units, all working together to record, monitor, and respond to events or conditions.
- Sensors work by detecting physical changes and converting those into digital signals.

Sensor Network

- These digital outputs are then sent to another system or device to be transformed into meaningful information for use by machines or people.
- Sensor data isn’t limited to what humans can perceive — in fact, sensors offer much broader and more precise measurements than human senses can provide.
- This gives sensors the ability to offer enhanced or "superhuman" sensory capabilities.
- Sensors respond to environmental changes or system states and then process or transmit that data accordingly.

Sensor Network

- They act as input devices by detecting and reacting to changes in physical characteristics caused by external stimuli.
- For example, a temperature sensor transforms heat into an electrical signal, while a barometer converts atmospheric pressure into a signal.
- These events can occur in diverse environments, including the physical world, industrial settings, or biological systems.
- The systems that use or manage this sensor data might be consumer applications, government agencies, military operations, or industrial organizations.
- Sensor networks are widely used for applications like remote sensing, medical telemetry, surveillance, monitoring, and data gathering.

What is a Sensor Network?

A sensor network consists of:

- Multiple sensors,
- Processing units,
- Communication modules,...working together to record, observe, and respond to specific events or conditions. These networks are often used in applications like:
- Remote monitoring
- Medical telemetry
- Environmental sensing
- Surveillance
- Data collection

Examples of sensors

- **Touch Sensors**: Detect physical contact (e.g., microswitches).
- **Stress/Force Sensors**: Measure the force exerted; often use strain gauges.
- **Ultrasonic Sensors**: Detect objects without contact by emitting sound waves and measuring the echo.
- **Proximity Sensors**: Detect when an object is near another.
- **Accelerometers**: Measure acceleration using microstructures (MEMS) that bend in response to motion or gravity.
- **Pressure Sensor**: Measures the force from gases or liquids and converts it to electrical signals. Often uses strain gauges or piezoelectric materials. Applications: Barometers, tire pressure monitors, altimeters, hydraulic systems.

Examples of sensors

- **Infrared (IR) Sensor**: Detects infrared radiation or uses it for communication. Applications: Motion detection, remote controls, night vision, robotics.
- **Humidity Sensor**: Measures air moisture by Detecting changes in resistance or capacitance due to humidity. Applications: Weather monitoring, greenhouse control, HVAC systems, food storage.
- **Tilt Sensor**: Detects tilt or orientation. Uses MEMS or a moving part that alters electrical output when tilted. Applications: Anti-theft systems, gaming devices, vehicle stability, drones.
- **Vision and Voice Sensors**: Captures and processes visual data (e.g., cameras for object recognition).
- **Voice Sensor (Microphone)**: Converts sound to electrical signals; supports voice recognition. Applications: Facial/voice recognition, smart assistants, surveillance, robotics.

Sensor Categories

- **Active vs Passive**:
  - *Active sensors* emit energy and measure the response (e.g., ultrasonic, laser).
  - *Passive sensors* detect energy already present in the environment (e.g., cameras, microphones).
- **Contact vs Non-contact**: Based on whether the sensor needs physical contact to function.
- **Invasive vs Non-invasive**: Invasive sensors are embedded in the system being monitored; non-invasive ones remain external.
- **By Application**: Sensors are also classified based on the industries they serve (e.g., medical, industrial, consumer electronics).

Sensor Categories

- **By Measurement Method**: Categories include thermoelectric, piezoresistive, electrochemical, optical, and more.
- **By Measured Quantity**:
  - *Digital Sensors*: Provide discrete (binary) outputs representing ON/OFF states.
  - *Scalar Sensors*: Measure quantities that only need magnitude (e.g., temperature, pressure).
  - *Vector Sensors*: Capture direction and magnitude (e.g., acceleration, velocity).

Key Sensor Features

- **Sensitivity**: Measures how responsive the sensor is to a particular input; ratio of output signal to input signal.
- **Selectivity**: A good sensor is sensitive only to the desired property and ignores unrelated factors (e.g., a temperature sensor ignores light or sound).
- **Non-intrusiveness**: Does not alter or affect the property it measures.

Key Sensor Features

- **Resolution**: Smallest detectable change in input. Higher resolution implies better detection of fine variations, but not necessarily better accuracy.
- **Accuracy**: How close the sensor's output is to the actual value.
- **Precision**: Consistency of the sensor's output over time under the same conditions.
- **Response Time**: How quickly a sensor reacts to changes in the measured input.
- **Reliability**: Long-term performance without frequent failures.
- **Operating Range**: Span between the minimum and maximum values the sensor can accurately measure.
- **Transfer Function**: Describes the relationship between input and output, typically represented mathematically, in tables, or via graphs. Real-world sensors often deviate slightly from ideal behavior and must be calibrated.

sensor measurement errors

- Sensors may not always produce perfect readings due to several types of errors. These errors affect the accuracy, precision, and reliability of the measurements.
- **Offset Error**:
  The sensor responds with the correct sensitivity, but the output signal is shifted from the expected value. This means the sensor's response curve is parallel to the ideal but offset vertically.
- **Nonlinearity Error**:
  Occurs when the sensor output doesn't increase proportionally with the input. The response curve deviates from a straight line, showing inconsistency in sensitivity across the range.
- **Deadband**:
  A range of input values where the sensor fails to respond. This insensitivity usually happens around zero or other specific values.

Sensor Measurement Errors

- **Hysteresis**:
  The sensor output varies depending on whether the input is increasing or decreasing, even if the final value is the same. This causes lag in response.
- **Random Noise**:
  Small, rapid fluctuations in the output signal despite a constant input. Often caused by electrical noise, temperature, or interference within the circuit or sensing element.
- **Drift**:
  Over time, the sensor may slowly produce incorrect readings for a fixed input. This gradual change, without any change in the actual input, is known as drift.

Actuator

- An **actuator** is a device that carries out physical actions in response to instructions from a control system. It serves as the **"muscle"** of the system, converting commands into motion or force. While **sensors act as the "senses"**, gathering information about the environment, **actuators are the "hands" or "legs"**, acting to change the environment.
- In a control system:
- **Sensors** detect conditions (like temperature, position, or pressure) and send this data to a controller.
- The **controller** processes this data and decides what action is needed.
- **Actuators** then carry out that action—such as opening a valve, pushing a piston, or rotating a gear.

How an Actuator Works

- **Receives Input Signal**:
  The actuator gets a control signal, which could be:
  - **Electrical** (from a microcontroller or computer)
  - **Hydraulic** (fluid under pressure)
  - **Pneumatic** (compressed air)
- **Converts Energy to Motion**:
  This input signal is translated into mechanical movement:
  - **Linear motion** (moving straight forward/backward)
  - **Rotational motion** (spinning or turning)
- **Performs Mechanical Work**:
  Depending on its application, the actuator might:
  - Open or close a valve, Move a robotic arm, Turn a wheel or shaft, Push, pull, lift, or adjust a mechanism

Types Of Actuators

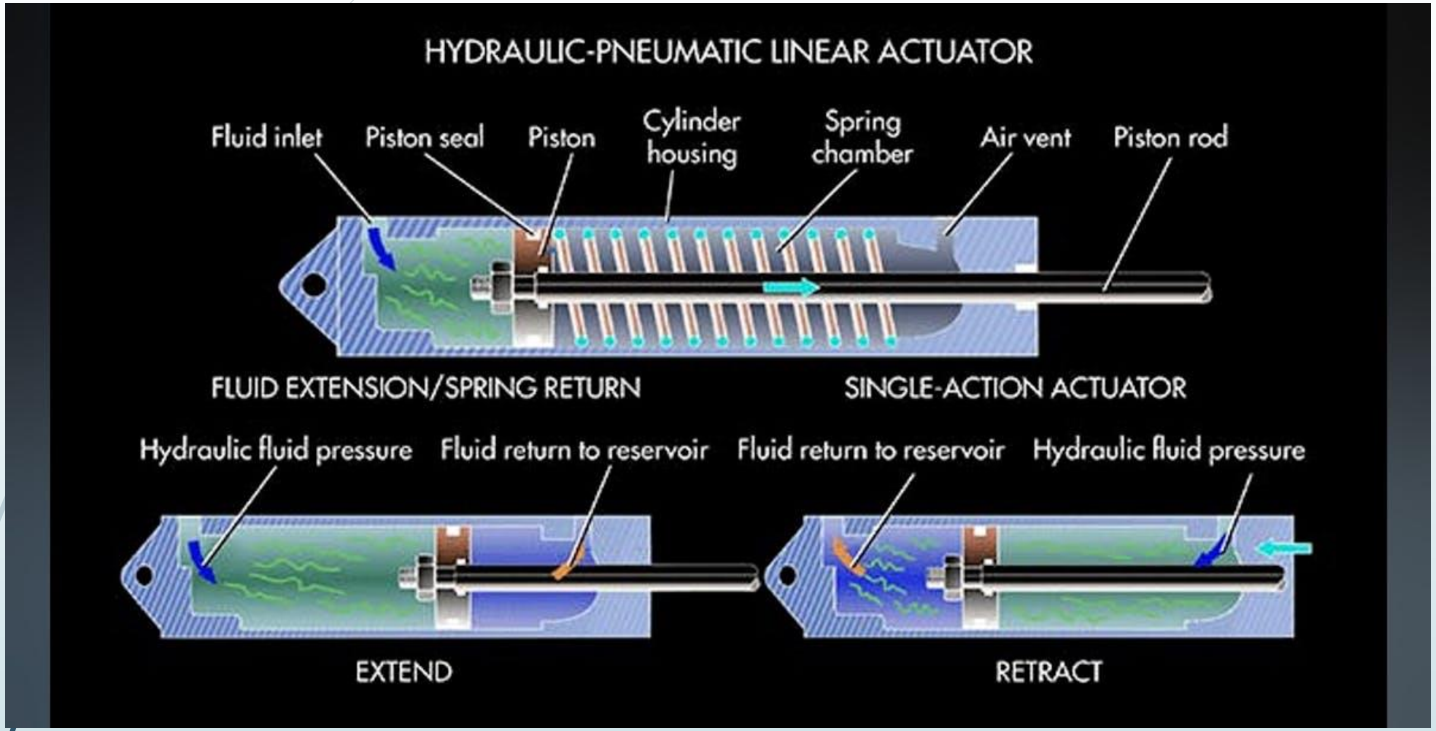
1. **Hydraulic Actuator:**

- Operates using fluid power—typically a cylinder or motor that uses pressurized hydraulic fluid to produce mechanical motion.
- Can generate linear, rotary, or oscillating movement. Because liquids can't be compressed, it can generate high force. However, its usage is limited due to slower response times (low acceleration).

2. Pneumatic Actuator

- Converts energy from compressed air or vacuum pressure into motion—either linear or rotary.
- Commonly used in systems like water valve control (e.g., rack and pinion mechanisms).
- Offers fast response to start/stop commands.
- Doesn't need energy to be stored before use, making it ideal for quick, reactive applications.
- Can produce large forces from minor pressure changes—similar to how air brakes respond to light pedal pressure in vehicles.

Hydraulic Actuator



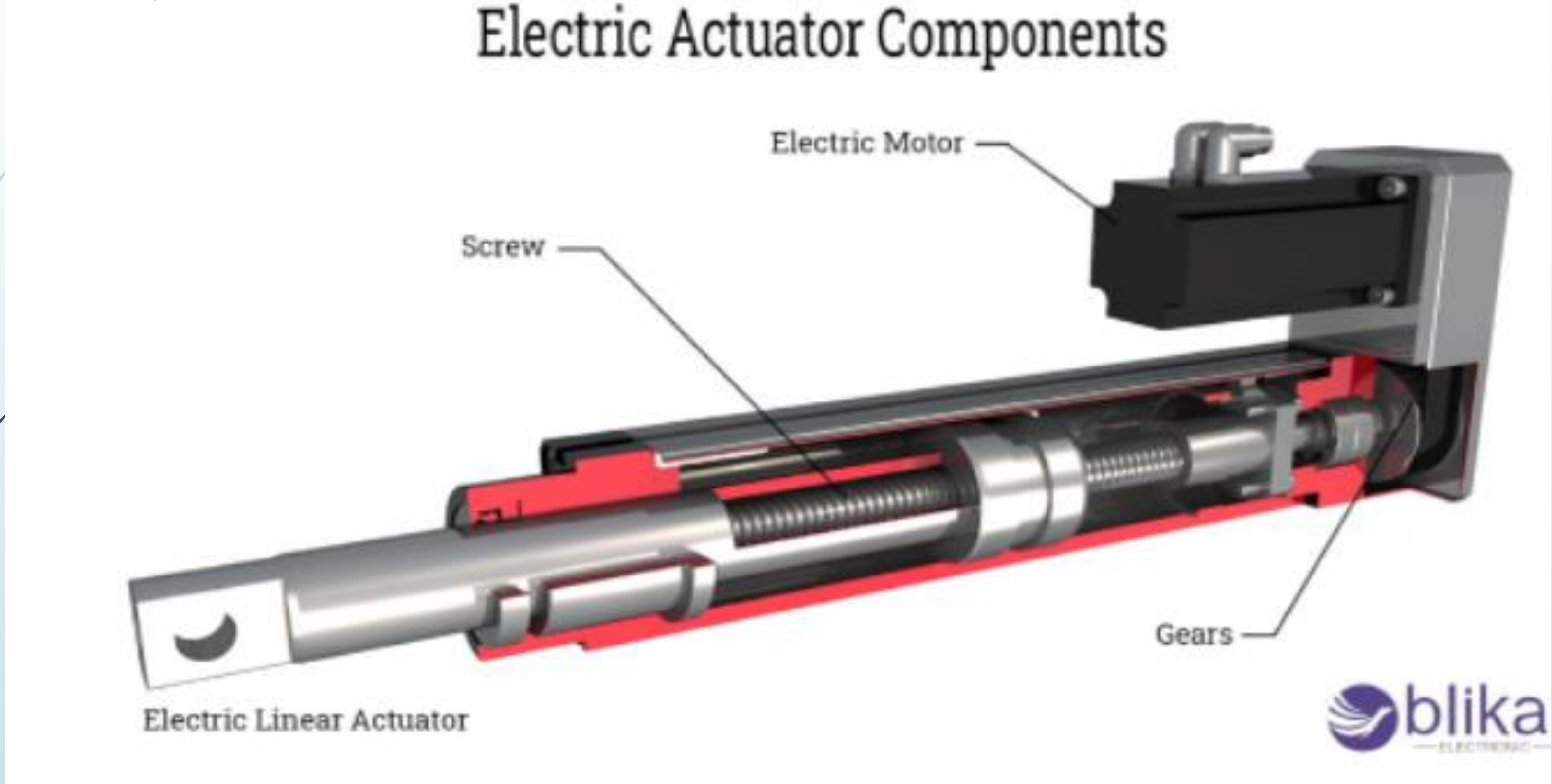
- https://img.machinedesign.com/files/base/ebm/machinedesign/image/2023/03/promo.6414bb2aafdaa.png?auto=format,compress&fit=fill&fill=blur&w=1200&h=630

Types of Actuators

Electric Actuator

- Powered by electric motors that convert electrical energy into mechanical movement (torque).
- Frequently used in devices like solenoid valves to regulate water flow in response to signals.
- Known for being affordable, clean (no fluid leaks), and fast-acting.
- Thermal/Magnetic Actuator
- Activated through heat or magnetic fields.
- Lightweight, compact, cost-effective, and high in power output relative to size.
- Often uses Shape Memory Alloys (SMA) or Magnetic Shape Memory Alloys, which return to a predetermined shape when heated or exposed to a magnetic field.

Electric Actuator



- [Electric Actuator: Types, Applications & Advantages]

Types Of Actuators

Thermal/Magnetic Actuator

- Activated through heat or magnetic fields.
- Lightweight, compact, cost-effective, and high in power output relative to size.
- Often uses **Shape Memory Alloys (SMA)** or **Magnetic Shape Memory Alloys**, which return to a predetermined shape when heated or exposed to a magnetic field.

Mechanical Actuator

- Converts rotary motion into linear motion.
- Uses components like gears, chains, pulleys, and rails.
- A typical example is the **rack and pinion** mechanism found in steering systems.

References

1. Banafa, A. (2023). *Introduction to Internet of things.* Oxon: River Publishers.

2. Greengard, S. (2015). *The internet of things.* Massachusetts Institute of Technology.

3. Masode, F. S. (2025). *Internet of things Applications and Technology.* Taylor & Francis Group, LLC.

Next Lecture

Emerging Trends In Software Engineering