SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

SCHOOL OF ELECTRICAL & ELECTRONICS ENGINEERING

DEPARTMENT OF ELECTRICAL & ELECTRONICS ENGINEERING

SEEA3016 - Embedded & IoT

COURSE OBJECTIVES

- To understand fundamentals of IoT and embedded system including essence, basic design strategy and process modeling.
- To develop comprehensive approach towards building small low cost embedded IoT system.
- To learn real world application scenarios of IoT along with its societal and economic impact using case studies

UNIT 1 INTRODUCTION TO EMBEDDED SYSTEM AND IoT 9 Hrs.

Introduction to embedded systems, Application Areas ,Categories of embedded systems, Overview of embedded system architecture, Specialties of embedded systems, recent trends in embedded systems, Introduction to ARM processor and its architecture. Internet Of Things Promises–Definition– Scope–Sensors, IoT Applications–Structure of IoT– IoT Map Device **;** IoT Sensors-Characteristics-types. IoT Issues and Challenges, Applications.

UNIT 2 EMBEDDED IoT PLATFORM DESIGN METHODOLOGY 9 Hrs.

Purpose and requirement specification, Process specification, Domain model specification, information model specification, Service specifications, IoT level specification, Functional view specification, Operational view specification, Device and component integration, Application development.

UNIT 3 PILLARS OF EMBEDDED IoT AND PHYSICAL DEVICES 9 Hrs.

The internet of devices, The internet of objects, The internet of transducer, o The internet of controllers, Device to Connect and Manage, talk, Connect. Network, Basic building blocks of and IoT device, Exemplary device: Raspberry Pi, Raspberry Pi interfaces, Programming Raspberry Pi with Python, ▪ Beagle board and other IoT Devices.

UNIT 4 WEB OF THINGS AND CLOUD OF THINGS 9 Hrs.

Web of Things versus Internet of Things, Two Pillars of the Web, Architecture Standardization for WoT, Cloud of Things: Grid/SOA and Cloud Computing, Cloud Middleware, Cloud Standards – Cloud Providers and Systems, Mobile Cloud Computing, The Cloud of Things Architecture.

UNIT 5 IoT CLOUD OFFERINGS AND IoT CASE STUDIES 9 Hrs.

Introduction to Cloud Storage Models, Communication API, Amazon Web Services for IoT, Skynet IoT Messaging Platform. Case Studies: Home Intrusion Detection, Weather Monitoring System, Air Pollution Monitoring, Smart Irrigation, Energy Harvesting.
Max. 45 Hrs.

COURSE OUTCOMES

On completion of the course, student will be able to
CO1 - Understand the basic concepts of embedded systems and IoT.
CO2 - Apply the design methodology for embedded IoT Platform.
CO3 - Develop programs using Python for Raspberry Pi.
CO4 - Comprehend web of things and cloud of things.
CO5 - Implement IoT Cloud Offerings for
CO6 - Solve the given societal challenge using IoT

TEXT / REFERENCE BOOKS

1.Adrian McEwen and Hakim Cassimally, ―Designing the Internet of Things‖, John Wiley and Sons Ltd, UK, 2014.
2.Vijay Madisetti, Arshdeep Bahga, ―Internet of Things (A Hands-on Approach), Universities Press, 2015.
3.Dieter Uckelmann, Mark Harrison, Florian Michahelles, ―Architecting the Internet of Things‖, Springer, New York, 2011.
4.John H. Davies, ―MSP430 Microcontroller Basics‖, First Edition, Newnes Publication. 2010.

UNIT I - Embedded & IoT - SEEA3016

I. INTRODUCTION TO EMBEDDED SYSTEM AND IOT

A **system** is a set of interrelated and interdependent components placed in an orderly manner that work together given an input to achieve some expected output.

An **Automated system** is the one that function without any manual intervention. It works without a human operator physically located at the site where the system is installed.

Embedded system :

- It is an integrated system with a combination of hardware and software which together form a component of a larger machine.
- An example of an embedded system is a microprocessor that controls an automobile engine.
- An embedded system is designed to run on its own without human intervention, and may be required to respond to events in real time.
- It is a dedicated computer system, designed to work for single or few specific functions often within a larger system.
- Embedded Systems, therefore, are Built to function with little or no human intervention.
- These systems are different from the general-purpose computers (desktops and laptops) – the general-purpose computer can handle a wide range of processing tasks unlike embedded systems.

Applications Areas of Embedded system :

- Television
- stereo
- remote control
- phone / mobile phone
- refrigerator
- microwave
- washing machine
- electric tooth brush
- oven / rice or bread cooker
- watch
- alarm clock
- electronic musical instruments
- electronic toys (stuffed animals,handheld toys, pinballs, etc.)
- medical home equipment (e.g. blood pressure, thermometer)

- **Medical Systems -** pace maker, patient monitoring systems, injection systems, intensive care units, …
- **Office Equipment -** printer, copier, fax, …
- **Tools -** multimeter, oscilloscope, line tester, GPS, …
- **Banking -**ATMs, statement printers, …
- **Transportation -** (Planes/Trains/[Automobiles] and Boats) -radar, traffic lights, signalling systems, …
- **Automobiles -**engine management, trip computer, cruise control, immobilizer, car alarm,airbag, ABS, ESP, …
- **Building Systems -**elevator, heater, air conditioning, lighting, key card entries, locks, alarm systems, …
- **Agriculture -**feeding systems, milking systems, …
- **Space -** satellite systems, …

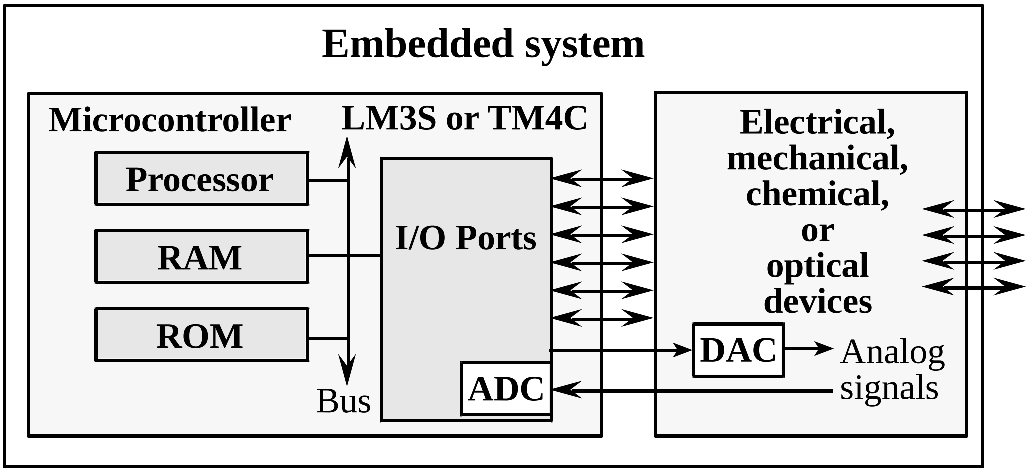


Fig. Block Diagram of Embedded System

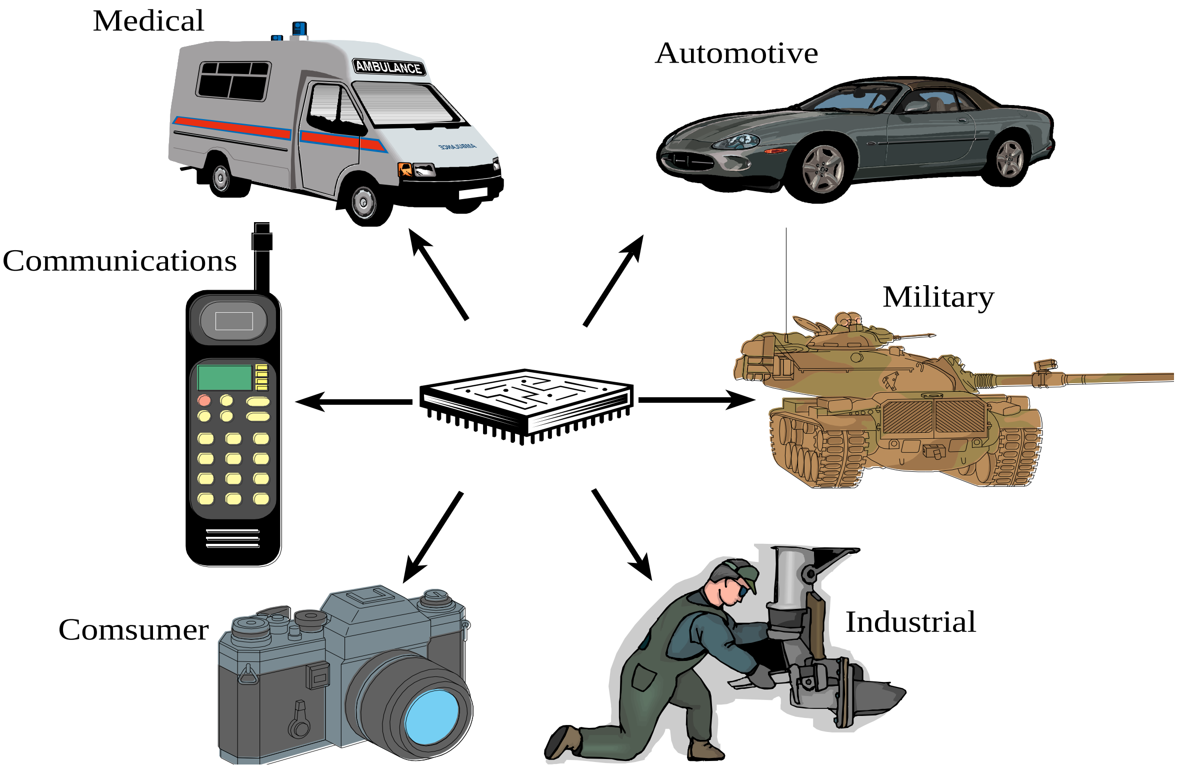


Fig. Application areas of Embedded System

Fig. Application areas of Embedded System

Categories of Embedded Systems (Classification) :

Embedded Systems are classified based on the two factors i.e.

I. Performance and Functional Requirements
II. Performance of Micro-controllers

I. Based on Performance and Functional Requirements it is divided into 4 types as follows :

1.**Real-Time Embedded Systems :**

A Real-Time Embedded System is strictly time specific which means these embedded systems provides output in a particular/defined time interval. These type of embedded systems provide quick response in critical situations which gives most priority to time based task performance and generation of output. That's why real time embedded systems are used in defense sector, medical and health care sector, and some other industrial applications where output in the right time is given more importance.Further this Real-Time Embedded System is divided into two types i.e.

- **Soft Real Time Embedded Systems –**

  In these types of embedded systems time/deadline is not so strictly followed. If deadline of the task is passed (means the system didn't give result in the defined time) still result or output is accepted.
- **Hard Real-Time Embedded Systems –**

  In these types of embedded systems time/deadline of task is strictly followed. Task must be completed in between time frame (defined time interval) otherwise result/output may not be accepted.

Examples :

- Traffic control system
- Military usage in defense sector
- Medical usage in health sector

2. **Stand Alone Embedded Systems :**

Stand Alone Embedded Systems are independent systems which can work by themselves they don't depend on a host system. It takes input in digital or analog form and provides the output.

Examples :

- MP3 players
- Microwave ovens
- Calculator

3. **Networked Embedded Systems :**

Networked Embedded Systems are connected to a network which may be wired or wireless to provide output to the attached device. They communicate with embedded web server through network.

Examples :

- Home security systems
- ATM machine
- Card swipe machine

4. **Mobile Embedded Systems :**

Mobile embedded systems are small and easy to use and requires less resources. They are the most preferred embedded systems. In portability point of view mobile embedded systems are also best.

Examples :

- MP3 player
- Mobile phones
- Digital Camera

II. Based on Performance and micro-controller it is divided into 3 types as follows :

1. **Small Scale Embedded Systems :**
Small Scale Embedded Systems are designed using an 8-bit or 16-bit micro-controller. They can be powered by a battery. The processor uses very less/limited resources of memory and processing speed. Mainly these systems does not act as an independent system they act as any component of computer system but they did not compute and dedicated for a specific task.

2. **Medium Scale Embedded Systems :**
Medium Scale Embedded Systems are designed using an 16-bit or 32-bit micro-controller. These medium Scale Embedded Systems are faster than that of small Scale Embedded Systems. Integration of hardware and software is complex in these systems. Java, C, C++ are the programming languages are used to develop medium scale embedded systems. Different type of software tools like compiler, debugger, simulator etc are used to develop these type of systems.

3. **Sophisticated or Complex Embedded Systems :**
Sophisticated or Complex Embedded Systems are designed using multiple 32-bit or 64-bit micro-controller. These systems are developed to perform large scale complex functions. These systems have high hardware and software complexities. We use both hardware and software components to design final systems or hardware products.

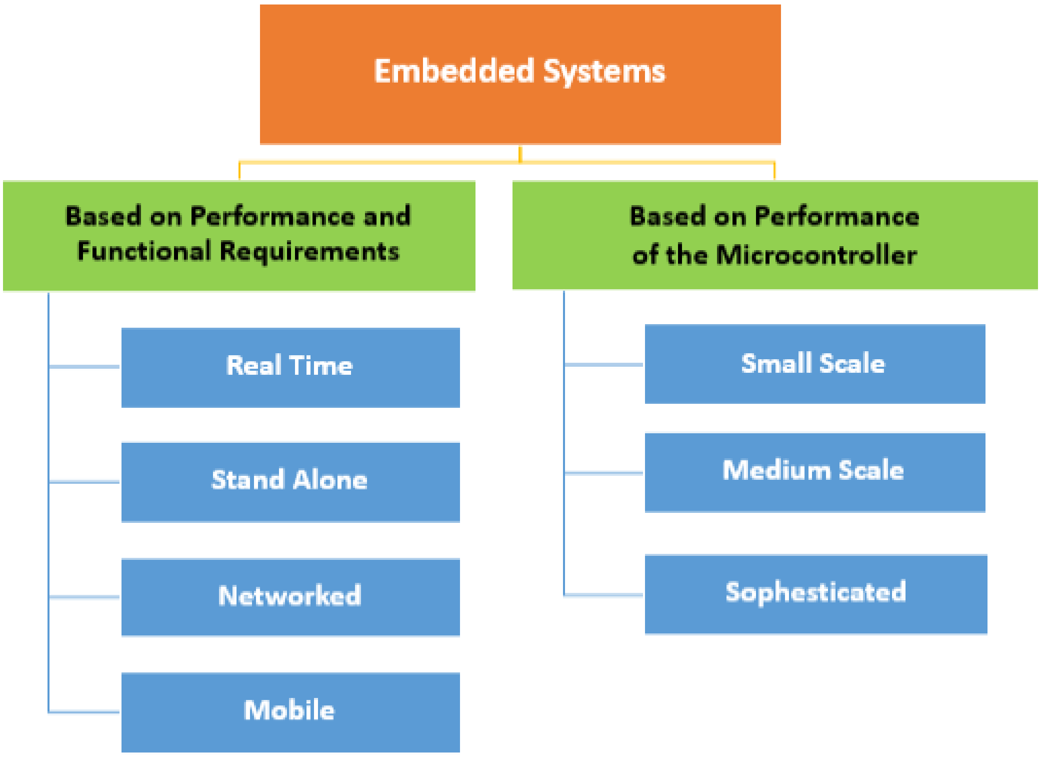

Embedded Systems
Based on Performance and Functional Requirements
Based on Performance of the Microcontroller
Real Time
Stand Alone
Networked
Mobile
Small Scale
Medium Scale
Sophesticated

Fig. Types of Embedded System

Difference Between Hard Real Time System And Soft Real Time System In

Basis Of Comparison	**Hard Real Time System**	**Soft Real Time System**
Data File Size	Size of the data file in soft real time systems is small or medium.	Size of the data file in soft real time systems is large.
Example	Examples of hard real time system are inkjet printer system, railway signaling system, air traffic control systems, nuclear reactor control systems, anti-missile system.	Examples of soft real time systems are DVD player, electronic games, multimedia system, web browsing, online transaction systems, telephone switches, virtual reality, weather station, mobile communication etc.
Restrictive Nature	A hard-real time system is very restrictive.	Soft real time system is less restrictive in nature.
Response Time	The response predefined time of hard real time systems is in order of milliseconds and therefore, missing of the deadline results in complete or massive system failure, and therefore this system should not miss the	The response predefined time of soft real time systems are not very stringent, therefore missing of the deadline only affects the performance and not the entire system. Soft real time systems misses deadline occasionally.

	deadline.	
Peak Load	Peak load performance should be predictable and should not violate the predefined deadlines.	In a soft real time system a degraded operation in a rarely occurring peak load can be tolerated.
Conditional Requirement	A hard real-time system must remain synchronous with the state of the environment at all time.	Soft real time system will slow down their response time if the load is very high.
Database Size & Integrity	Most of the hard real time systems have small databases and occasionally require short-term integrity of the system.	Most of the soft real time systems have larger databases and require long-term integrity of the system.
Error Handling	In case of an error in a hard real time system, the computation is rolled back or recovery is of limited use.	In case of an error in a soft real time system, the computation is rolled back to previously established checkpoint to initiate a recovery action.
Completion Of Task/Activity	Completion of the task or activity by hard real time systems is predefined or deterministic.	Completion of task or activity by soft real time system probabilistic.
Validation	The users of hard real time systems obtain validation when required.	Users of soft real time systems not always obtain the validation.

Characteristics / Features / Specialities of an Embedded Systems

Some of the key characteristics of Embedded Systems are as mentioned below.

- Application and Domain specific : All Embedded Systems are task specific. They do the same task repeatedly /continuously over their lifetime. An mp3 player will function only as an mp3 player. A washing machine can only wash, it cannot cook.
- Embedded systems are created to perform the task within a certain time frame. It must therefore perform fast enough. A car’s brake system, if exceeds the time limit, may cause accidents.
- They have minimal or no user interface (UI). A fully automatic washing machine works on its own after the programme is set and stops once the task is over.
- Certain Embedded systems are designed to react to the events that occur in the nearby environment and react accordingly. These events also occur real-time. Example:A thermometer, a GPS tracking device. An air conditioner adjusts its mechanical parts as soon as it gets a signal from its sensors to increase or decrease the temperature when the user operates it using a remote control.

- Embedded systems are built to achieve certain efficiency levels. They are small sized, can work with less power and are not too expensive.
- Embedded systems cannot be changed or upgraded by the users. Hence, they must rank high on reliability and stability. They are expected to function for long durations without the user experiencing any difficulties.
- Operation in harsh environment : Certain embedded systems are designed to operate in harsh environments like very high temperature of the deserts or very low temperature of the mountains or extreme rains.
- Distributed : Certain embedded systems are part of a larger system and thus form components of a distributed system.These components are independent of each other but have to work together for the larger system to function properly. Example: A car has many embedded systems controlled to its dash board. Each one is an independent embedded system yet the entire car can be said to function properly only if all the systems work together.
- Small size and weight : An embedded system that is compact in size and has light weight will be desirable or more popular than one that is bulky and heavy. Example: The heat seeking system used in some missiles needs to be small and lightweight so as to fit in well and allow high speed maneuvers.
- Power concerns : It is desirable that the power utilization and heat dissipation of any embedded system be low.If more heat is dissipated then additional units like heat sinks or cooling fans need to be added to the circuit.If more power is required then a battery of higher power or more batteries need to be accommodated in the embedded system.
- Microcontroller or microprocessors are used to design embedded systems.
- Embedded systems need connected peripherals to attach input & output devices.
- The hardware of an embedded-system is used for security and performance. The Software is used for features.
- Requires real time performance.
- It should have high availability and reliability.
- Developed around a real-time operating system
- Usually, have easy and a diskless operation, ROM boot
- It must be connected with peripherals to connect input and output devices.
- Offers high reliability and stability
- Limited memory, low cost, fewer power consumptions
- It does not need any secondary memory in computer.

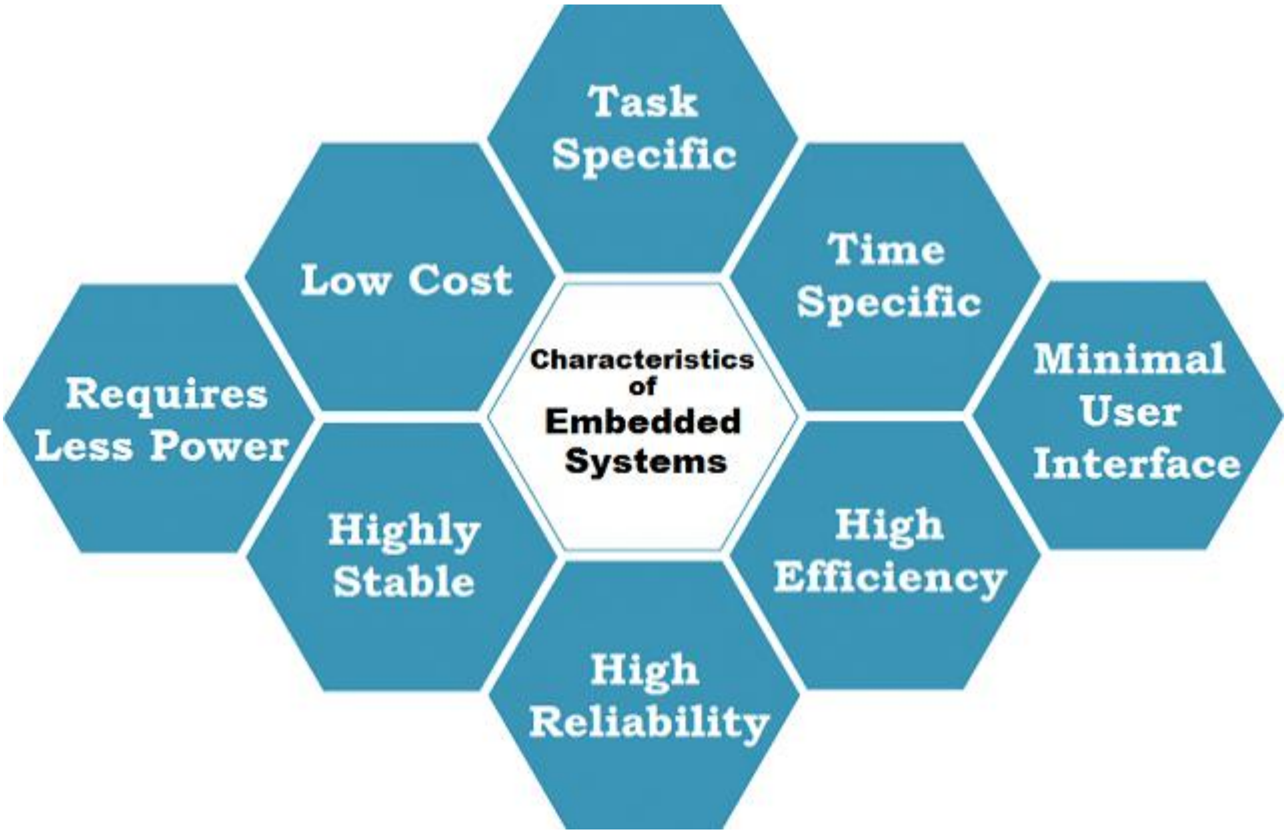


Fig. Characteristics of Embedded System

Advantages of Embedded System

The advantages of Embedded Systems are:

- They are convenient for mass production. This results in low price per piece.
- These systems are highly stable and reliable.
- Embedded systems are made for specific tasks.
- The embedded systems are very small in size, hence can be carried and loaded anywhere.(portability)
- These systems are fast.
- They also use less power - The power consumed by computer system is 60 W and 230 AC approximately and the power consumed by embedded system is less than 1 W and 3.3V.
- The embedded systems optimize the use or resources available.
- They improve the product quality.
- Accuracy

Disadvantages of Embedded System

The disadvantages of Embedded Systems are as follows:

- Once configured, these systems cannot be changed. Hence, no improvement or upgradation on the ones designed and created can be made.
- They are hard to maintain. It is also difficult to take a back-up of embedded files.
- Troubleshooting is difficult for embedded systems. Transferring data from one system to another is also quite problematic.
- Because these systems are made for specific tasks, hardware is limited.

Overview Of Embedded Systems Architecture :

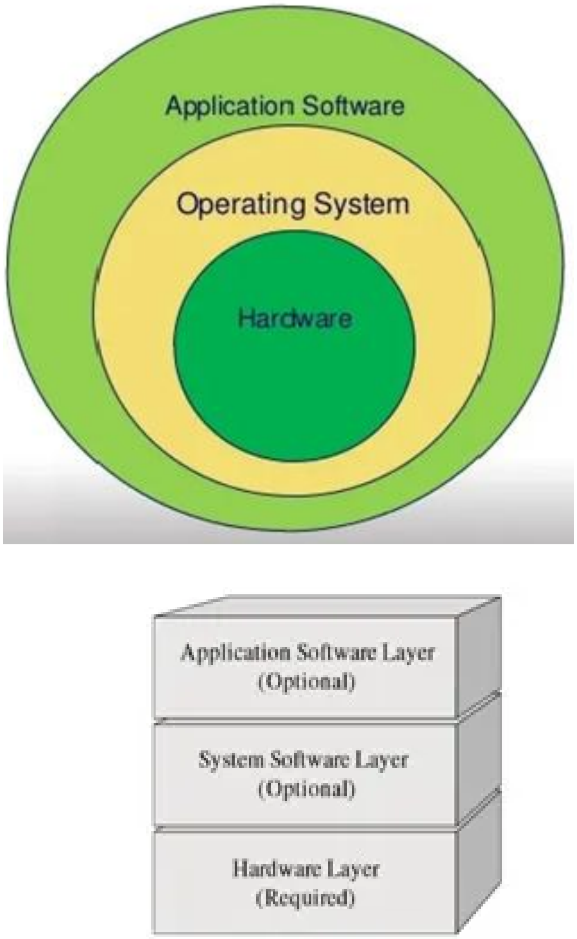

Application Software
Operating System
Hardware
Application Software Layer (Optional)
System Software Layer (Optional)
Hardware Layer (Required)

Fig. Architecture of Embedded System / Embedded system model

Components Of Embedded Systems Architecture :

An embedded system has 3 components:

- It has the hardware.
  - Processor : Microprocessor / Microcontroller / DSP processor / Controller
  - memory, bus, Input/Output

The size of components is kept small so that overall size of embedded system can be kept small.

- It has software program - embedded operating systems, different applications and device drivers.
  At a time one application can be reside in the system so that processing speed of task become fast and can get accurate results. The software used in the embedded system are specific to this type of system only and cannot be used as general software for other computational devices as the hardware components are different that can create problem of mismatch configuration.

- It has an actual real-time operating system (RTOS) that supervises the utility software and offer a mechanism to let the processor run a process as in step with scheduling by means of following a plan to manipulate the latencies. RTOS defines the manner the system works. It units the rules throughout the execution of application software. A small scale embedded device won't have RTOS.

Basic Structure Of An Embedded Systems Architecture

SENSOR:

It measures the quantities that are physical and converts it to an electrical signal which may be read by an observer or through any electronic tool like an A-D converter. A sensor shops the measured amount to the memory.

A-D CONVERTER:

An analog-to-digital converter that is used converts the analog signal sent by using the sensor right into a digital signal.

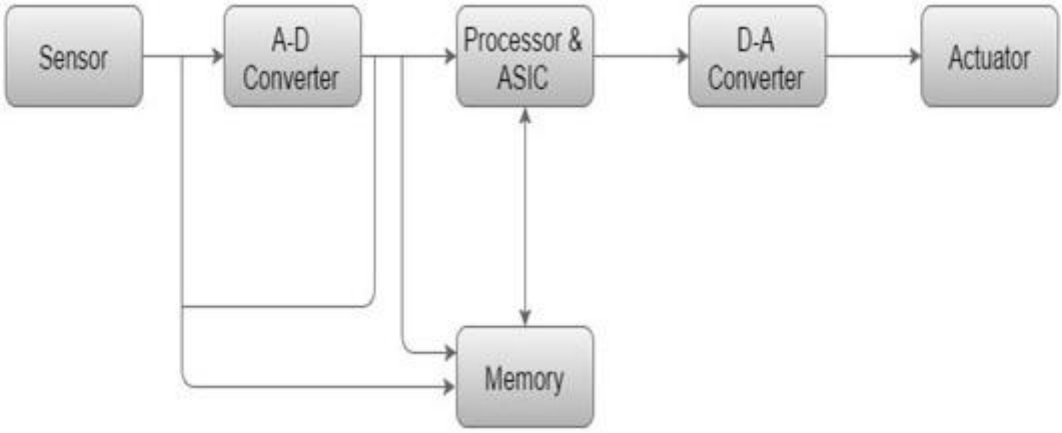


Fig. Basic structure of the embedded systems architecture.

PROCESSOR:
Processors process the records to degree the output and keep it to the memory.

D-A CONVERTER:
A virtual-to-analog converter converts the virtual records fed by using the processor to analog information.

ACTUATOR:
An actuator compares the output given by means of the D-A converter to the actual (anticipated) output saved in it and stores the authorized output.

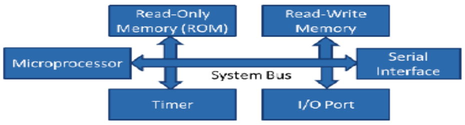
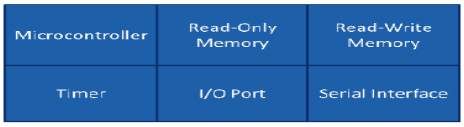
Embedded Product Development Life Cycle (EDLC) :
Developing an embedded system or product mainly goes through this three phases which are –

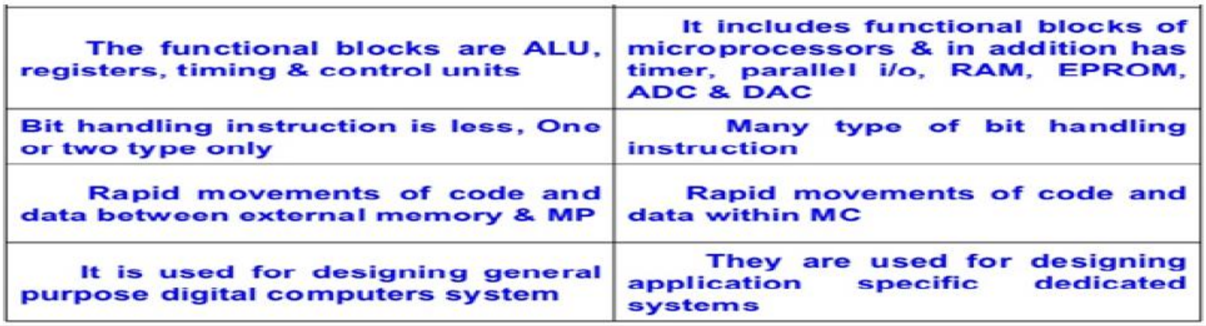
1. Analysis
2. Design
3. Implementation

If we will go a little bit deeper to the development steps it includes these 7 steps :

1. Requirement analysis
2. Examine
3. Design
4. Develop
5. Test
6. Deploy
7. Maintenance

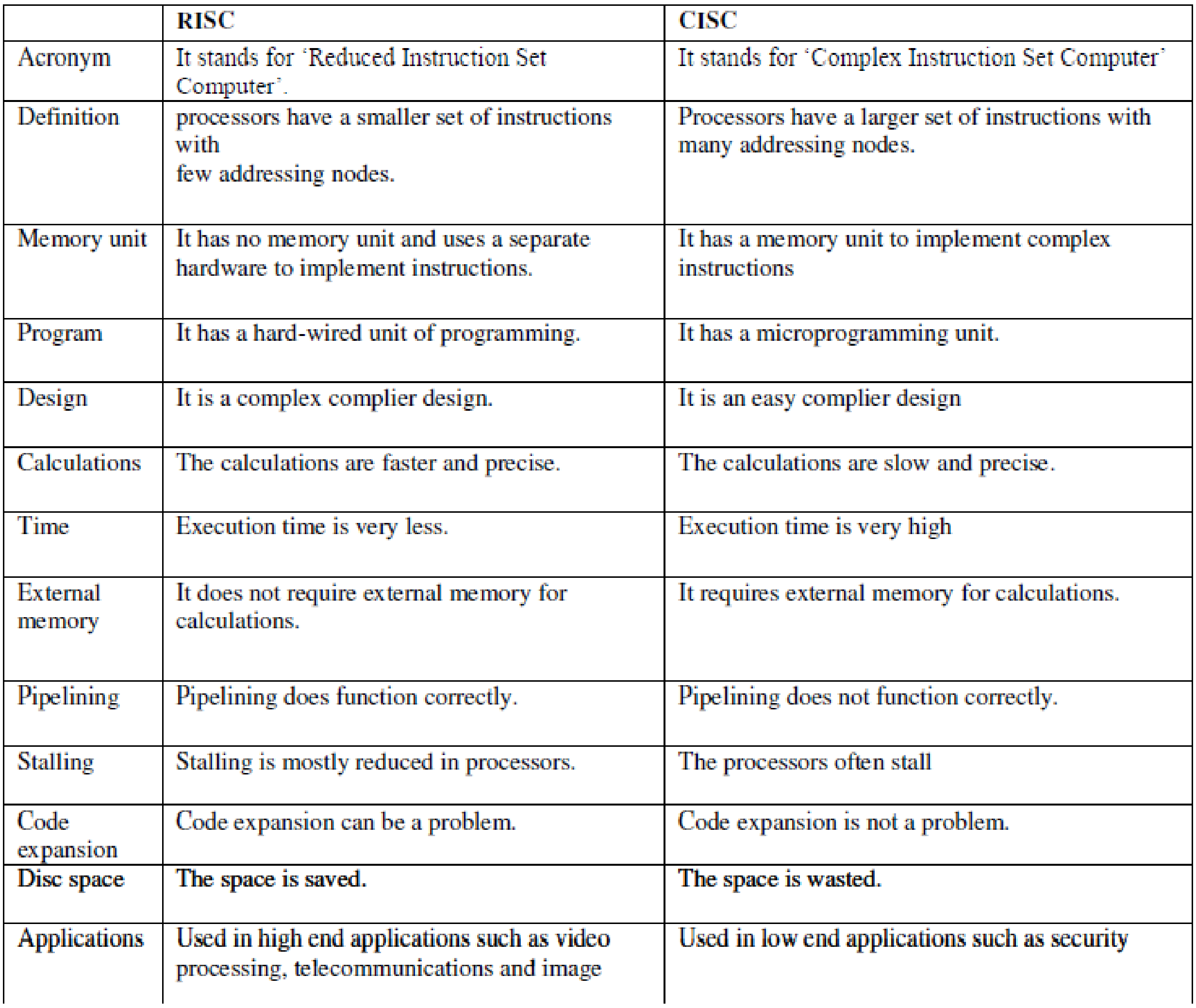
Difference between Microprocessor and microcontroller

Microprocessor	Micro Controller
 Read-Only Memory (ROM) Read-Write Memory Microprocessor System Bus Serial Interface Timer I/O Port	 Microcontroller Read-Only Memory Read-Write Memory Timer I/O Port Serial Interface
Microprocessor is heart of Computer system.	Micro Controller is a heart of embedded system.
It is just a processor. Memory and I/O components have to be connected externally	Micro controller has processor along with internal memory and i/O components
Since memory and I/O has to be connected externally, the circuit becomes large.	Since memory and I/O are present internally, the circuit is small.
Cannot be used in compact systems and hence inefficient	Can be used in compact systems and hence it is an efficient technique
Cost of the entire system increases	Cost of the entire system is low
Due to external components, the entire power consumption is high. Hence it is not suitable to used with devices running on stored power like batteries.	Since external components are low, total power consumption is less and can be used with devices running on stored power like batteries.
Most of the microprocessors do not have power saving features.	Most of the micro controllers have power saving modes like idle mode and power saving mode. This helps to reduce power consumption even further.
Since memory and I/O components are all external, each instruction will need external operation, hence it is relatively slower.	Since components are internal, most of the operations are internal instruction, hence speed is fast.
Microprocessor have less number of registers, hence more operations are memory based.	Micro controller have more number of registers, hence the programs are easier to write.
Microprocessors are based on von Neumann model/architecture where program and data are stored in same memory module	Micro controllers are based on Harvard architecture where program memory and Data memory are separate
Mainly used in personal computers	Used mainly in washing machine, MP3 players



The functional blocks are ALU, registers, timing & control units	It includes functional blocks of microprocessors & in addition has timer, parallel i/o, RAM, EPROM, ADC & DAC
Bit handling instruction is less, One or two type only	Many type of bit handling instruction
Rapid movements of code and data between external memory & MP	Rapid movements of code and data within MC
It is used for designing general purpose digital computers system	They are used for designing application specific dedicated systems

Comparison of RISC and CISC processors



	RISC	CISC
Acronym	It stands for 'Reduced Instruction Set Computer'.	It stands for 'Complex Instruction Set Computer'
Definition	processors have a smaller set of instructions with few addressing nodes.	Processors have a larger set of instructions with many addressing nodes.
Memory unit	It has no memory unit and uses a separate hardware to implement instructions.	It has a memory unit to implement complex instructions
Program	It has a hard-wired unit of programming.	It has a microprogramming unit.
Design	It is a complex complier design.	It is an easy complier design
Calculations	The calculations are faster and precise.	The calculations are slow and precise.
Time	Execution time is very less.	Execution time is very high
External memory	It does not require external memory for calculations.	It requires external memory for calculations.
Pipelining	Pipelining does function correctly.	Pipelining does not function correctly.
Stalling	Stalling is mostly reduced in processors.	The processors often stall
Code expansion	Code expansion can be a problem.	Code expansion is not a problem.
Disc space	The space is saved.	The space is wasted.
Applications	Used in high end applications such as video processing, telecommunications and image	Used in low end applications such as security

Recent trends in Embedded systems.

Embedded systems which are highly customizable are driving innovation and with a variety of programming strategies. The present and future industries that are aggressively growing include aerospace, healthcare, military, and consumer electronics applications, etc.

Python- In the last two years, it was found that the number of the projects were programmed in Python in the embedded space has increased double the times.

Wireless connections – By 2025 Internet of things is projected to have a tremendous growth of wireless connections and there will be a larger usage of IoT (Internet of things) globally.

Speed – 5g will enable transformation in embedded systems into different domains. Furthermore, 5G technology will have a greater impact on portable medical devices, the internet of things, telecom service providers, and automotive industries to just name a few. With 5G, the internet, processors, and systems will become superfast. Undoubtedly Quantum computing will play a huge role in increasing computing speed and for this we have to wait for just a few years.

Artificial intelligence – The new trend in the **embedded systems industry** is the increase in the usage of Artificial Intelligence and Machine Learning. AI is used almost in any segment such as e-commerce, manufacturing, supply chain, industrial control, and many more. AI takes the data and learns to make better decisions, while IoT and embedded systems which are physical devices help to generate data to achieve functionality.

Cyberattacks and Counterattacks – As cyberattacks are on the rise, enterprises are stepping up their efforts related to cybersecurity. It is projected that by 2025, cybersecurity revenues will reach US$254 billion. Embedded security hardware and software are growing aggressively, and new hard designs mostly have built-in security silicon.

Healthcare

With the recent changes in healthcare industries, there are developing applications in embedded systems. Some of the small embedded devices are already helpful in health care sectors like monitoring heart rate and used in intricate surgical procedures. Also, there is a trend to be faced like chips and processors are to be used in healthcare which has intelligence and functionality. These tiny, powerful devices will be able to monitor the condition of the patients and connect with a network-based diagnostic process.

Automobiles

There will be intelligent embedded devices in the automobile industry. Due to the impact of AI in this, the active area of research and technology has become an important component in transport for the future. With the advancements in AI, they tend to manually drive the vehicles and control transportation access while driving. There are also dozens of monitoring on-board sensors, AI and Robotic course can identify the situation and conflicts to be faced in driving. Also, giving alerts and detection of accident-prone areas will be avoided. In recent techs, modern embedded AI algorithms are used and hundreds of sensors are used to monitor the vehicle operation.

System-on-Chip solution

Embedded devices tend to reach globally by using the SOC market. Embedded world featured companies offering SoC solutions to the future. Thus, in 2021, Some of the SoC solutions like Ansem, Auriga partners get connected with medical and health care industries to create affordable and small size products which are highly beneficial for customers.

IoT Security

The Internet of Things remains a high priority in the industry of technology. The security issues are changing in the future. So, implementing IoT security solution along with embedded devices like AI and Robots helps for betterment. Also, these embedded devices tend to make decisions on data and their safety to avoid vulnerabilities. And, using IoT course related solutions with modernised techs, some of the protocols might get riddled. To avoid such mishandlings, we can use the embedded device with high-security options.

Automation everywhere

All the systems and products are getting automated nowadays. Mainly, due to the impacts of computer and robots and other machines, automation is found in every sector of development. Thus implementing Automation in everywhere is the trend to be faced in 2021. Due to pandemic restrictions till now, there is a need for remote access and covering all areas is necessary for life. With the physical co-location for remote areas is possible for some accommodation, it is proved to be autonomous everywhere. Also, with the help of embedded devices, there is an acceleration in connecting various storage elements. Also, they tend to monitor the surrounding for any threat. They get connected with cloud tech as a companion to drive the device for a rapid expansion of intelligent processing.

Virtual reality is becoming a reality

Since the early development of 3D virtual reality for gaming, VR and its offshoots have evolved into a serious, and seriously profitable, business. The initial VR concept has evolved to include augmented reality (AR), mixed or merged reality (MR), and extended Reality (XR), expanding the technology into a range of industrial and commercial applications.

AR is ideal for training. Users wear a see-through goggle with information superimposed on the screen. The information could be, for example, instructions for carrying out an equipment repair procedure. MR carries this one step further by allowing users to view both a real situation and other possible scenarios, mixing real and created images. Finally, XR is the ultimate application, projecting holographic-like images that look like the real, physical thing. For example, XR would lend a new dimension to remote conferencing; unlike traditional video conferencing, with XR an image of the person would appear in front of the user as if a face-to-face meeting were being held.

VR applications that can increase productivity span a broad range, including medicine, manufacturing, training, and entertainment. For example, a doctor-in-training requires many hands-on experiences, which may be costly to set up if not difficult to find. With VR, a simulation can be set up for the medical intern to practice without the risk of making a serious error on a real patient.

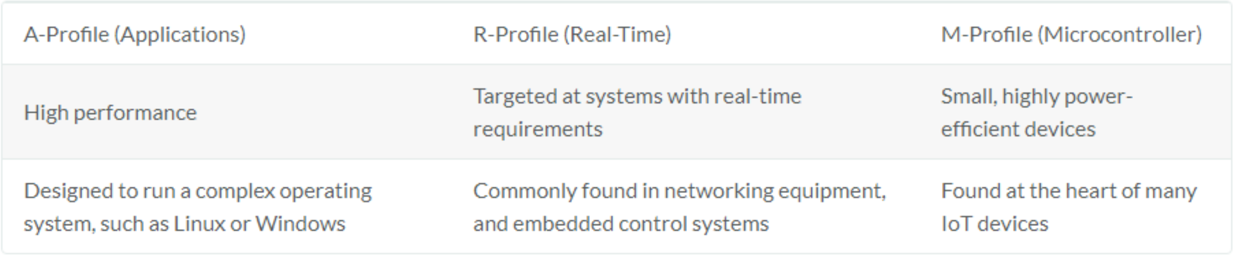
https://www.arcweb.com/blog/embedded-systems-trends-technologies-0

https://www.eetimes.eu/top-5-critical-trends-in-embedded-technology/

Introduction to ARM processor and its architecture

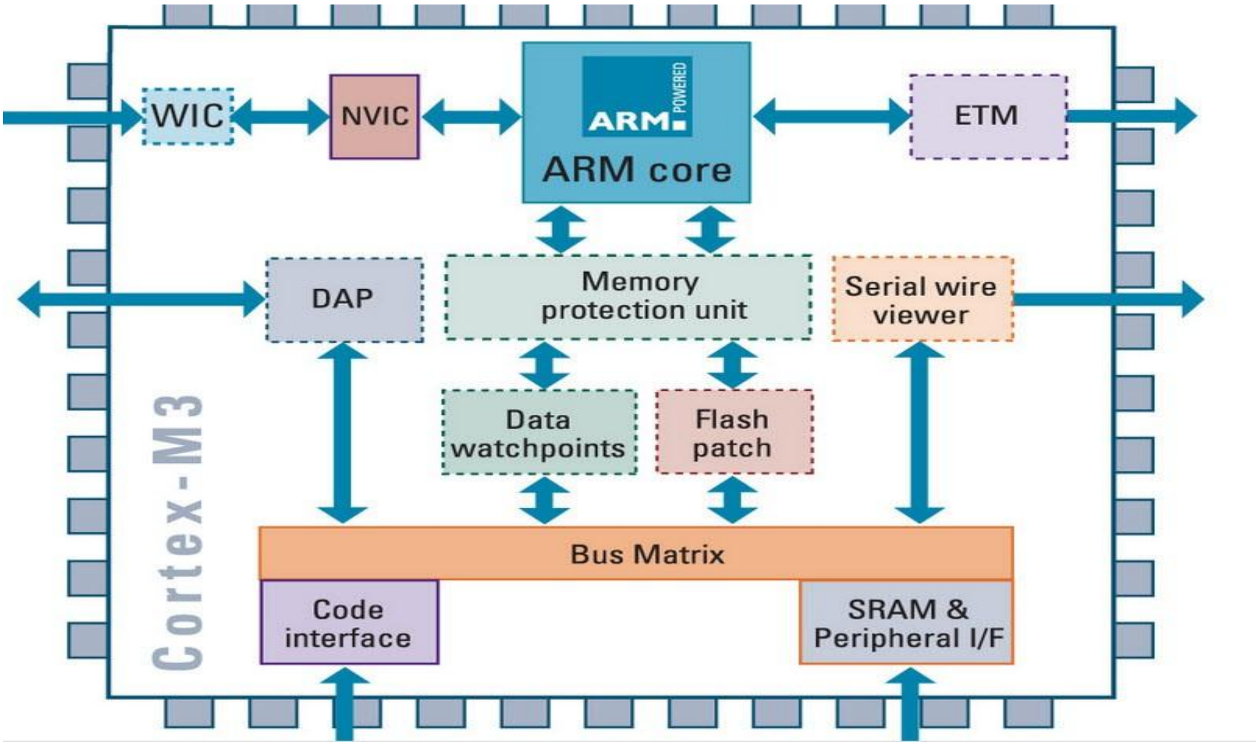
- **The Arm architecture provides the foundations for the design of a processor or core, things we refer to as a Processing Element (PE).**
- **Advanced RISC Machine**
- **The ARM architecture processor is an advanced reduced instruction set computing [RISC] machine and it’s a 32bit reduced instruction set computer (RISC).**
- **The ARM architecture comes with totally different versions like ARMv1, ARMv2, etc., and, each one has its own advantage and disadvantages.**

- **The ARM cortex is a complicated microcontroller within the ARM family that has ARMv7 design. There are 3 subfamilies within the ARM cortex family :**
  ARM Cortex Ax-series
  ARM-Cortex Rx-series
  ARM-Cortex Mx-series
- **Three architecture profiles of A, R, and M**



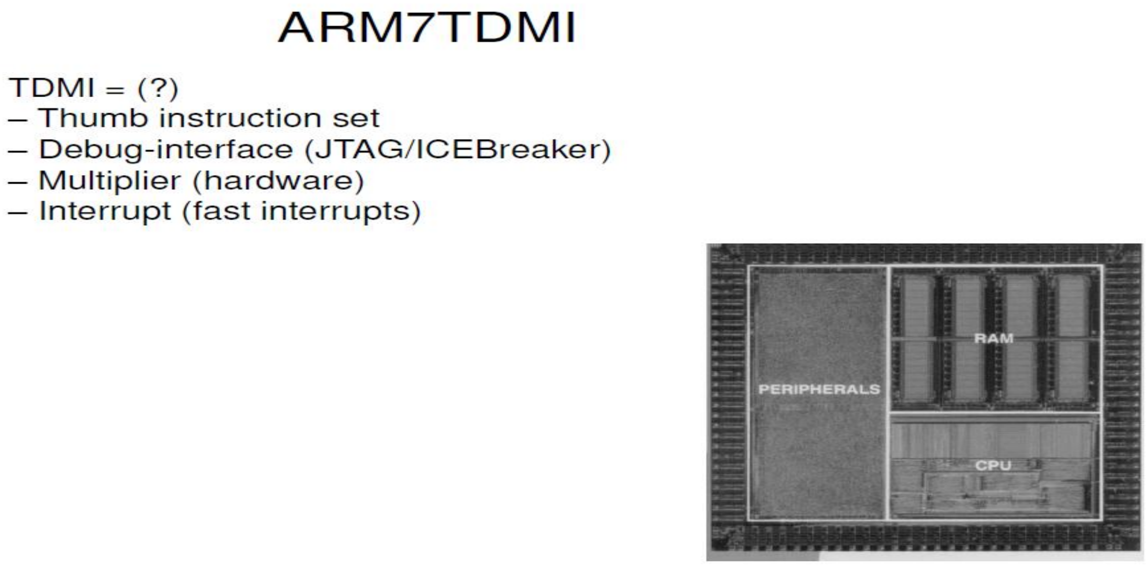
A-Profile (Applications)	R-Profile (Real-Time)	M-Profile (Microcontroller)
High performance	Targeted at systems with real-time requirements	Small, highly power-efficient devices
Designed to run a complex operating system, such as Linux or Windows	Commonly found in networking equipment, and embedded control systems	Found at the heart of many IoT devices

ARM Architecture :



- WIC – Wake up interrupt controller ; NVIC – Nested vectored interrupt controller
- ETM - The Embedded Trace Macrocell - interface enables you to connect an external ETM unit to the processor for real-time code and data tracing of the core in an embedded system. The ETM interface collects various processor signals and drives these signals from the processor.

- DAP - The Debug Access Port (DAP) is an implementation of an ARM Debug Interface version 5.1 (ADIv5. ... All the supplied components fit into the various architectural components for Debug Ports (DPs), which are used to access the DAP from an external debugger and Access Ports (APs), to access on-chip system resources.
- Booth multiplier ; Barrel shifter ; Control unit ; Register file



ARM7TDMI

TDMI = (?)
– Thumb instruction set
– Debug-interface (JTAG/ICEBreaker)
– Multiplier (hardware)
– Interrupt (fast interrupts)

RAM
PERIPHERALS
CPU

Requirement for Thumb Instruction set

https://www.embedded.com/introduction-to-arm-thumb/

JTAG - **Joint Test Access Group -** https://www.xjtag.com/about-jtag/what-is-jtag/

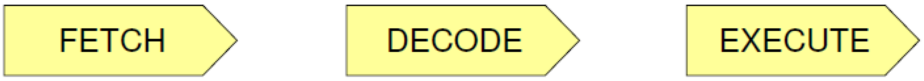
ARM7/ARM9 Architecture Feature Highlights

- 32/16-bit RISC architecture (ARM v4T)
- 32-bit ARM instruction set for maximum performance and flexibility
- 16-bit Thumb instruction set for increased code density
- Unified bus interface, 32-bit data bus carries both instructions and data
- 8-, 16-, and 32-bit Data Types
- Three-stage pipeline
- 4GBytes Linear Address Space
- 32-bit ALU and high-performance multiplier
- 37 piece of 32 bit register
- Very small die size and low power consumption
- Fully static operation
- Coprocessor interface
- Extensive debug facilities:
  - Embedded ICE-RT real-time debug unit.
  - On-chip JTAG interface unit.
- Interface for direct connection to Embedded Trace Macro cell (ETM).

- Pipelined (ARM7: 3 stages)
- Cached (depending on the implementation)
- Von Neuman-type bus structure (ARM7), Harvard (ARM9)
- 7 modes of operation (usr, fiq, irq, svc, abt, sys, und)
- Simple structure -> reasonably good speed / power consumption ratio
- Very Low Power Consumption: Industry-leader in MIPS/Watt.

ARM7 Pipeline Model

- **ARM7 → standard 3-stage pipelined architecture**



- **Fetch Instruction**
  - Select/Increment PC
  - Read next instruction
- **Related Blocks**
  - Address Selector
  - Address Incrementer
  - Address Register

- **Decode Instruction**
  - Generate Ctrl. signals
  - Generate immediate
  - Read from register file
- **Related Blocks**
  - Control Logic (Decoder)
  - Register File

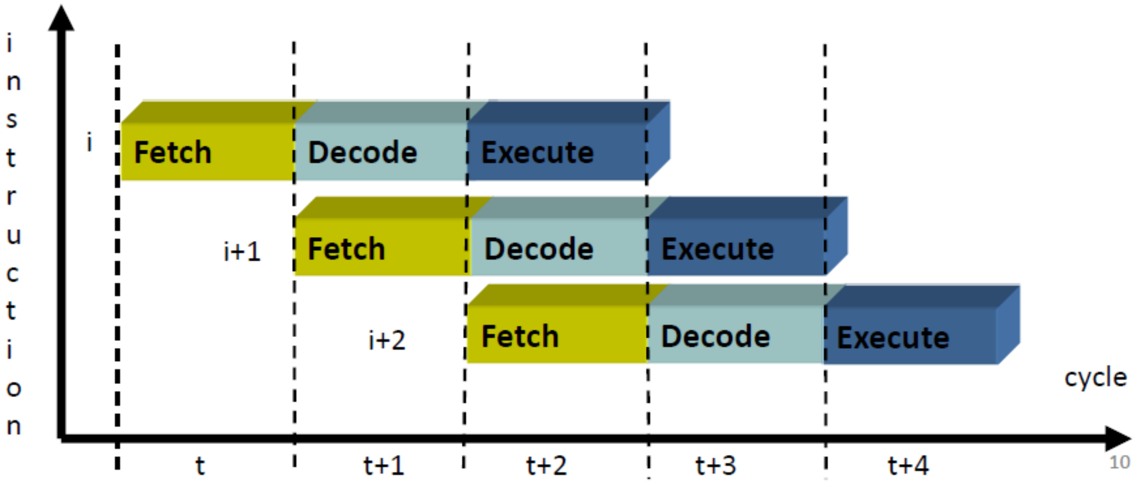
- **Execute Instruction**
  - Arithmetic / Logic
  - Calc. branch addr.
  - Load / Store
- **Related Blocks**
  - Shifter
  - Multiplier
  - ALU

*Register write back (WB) is hidden

Pipeline Organization

- 3-stage pipeline: Fetch – Decode - Execute
- Three-cycle latency,

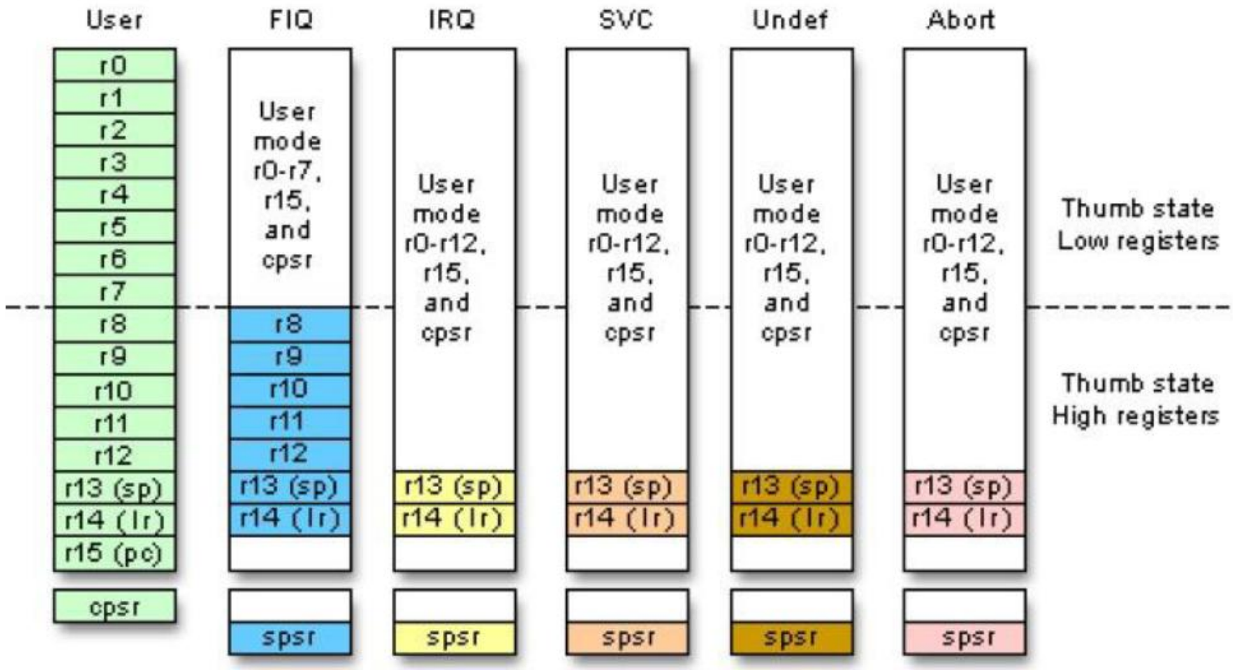
  one instruction per cycle throughput



ARM registers

- ARM cores use a 32-bit, Load-Store RISC architecture - That means that the core cannot directly manipulate the memory. All data manipulation must be done through registers loaded with data from memory, performing the data operation and then storing the value back to memory.
- There are 37 total registers in the processor – Registers are split among seven different processor modes.
- The seven processor modes are used to run user tasks, an operating system, and to efficiently handle exceptions such as interrupts.
- Some of the registers with in each mode are reserved for specific use by the core, while most are available for general use.
- The reserved registers that are used by the core for specific functions are

  r13 is commonly used as the stack pointer (SP),
  r14 as a link register (LR),
  r15as a program counter (PC),
  the Current Program Status Register (CPSR),
  and the Saved Program Status Register (SPSR).
- The SPSR and the CPSR contain the status and control bits specific to the operating mode, ALU status flags, interrupt disable/enable flags and whether the core is operating in 32-bit ARM or 16-bit Thumb state.
- The seven operating modes and the registers are shown under. (User, FIQ,IRQ-Interrupt modes, SVC-Supervisor, Undef – undefined, Abort – Abort mode)



User
FIQ
IRQ
SVC
Undef
Abort
r0
r1
r2
r3
r4
r5
r6
r7
r8
r9
r10
r11
r12
r13 (sp)
r14 (lr)
r15 (pc)
cpsr
User mode r0-r7, r15, and cpsr
r8
r9
r10
r11
r12
r13 (sp)
r14 (lr)
spsr
User mode r0-r12, r15, and cpsr
r13 (sp)
r14 (lr)
spsr
User mode r0-r12, r15, and cpsr
r13 (sp)
r14 (lr)
spsr
User mode r0-r12, r15, and cpsr
r13 (sp)
r14 (lr)
spsr
User mode r0-r12, r15, and cpsr
r13 (sp)
r14 (lr)
spsr
Thumb state Low registers
Thumb state High registers

The ARM7TDMI processor has seven modes of operation:

- User mode is the usual ARM program execution state, and is used for executing most application programs.
- Fast Interrupt (FIQ) mode supports a data transfer or channel process.

- Interrupt (IRQ) mode is used for general-purpose interrupt handling.
- Supervisor mode is a protected mode for the operating system.
- Abort mode is entered after a data or instruction Prefetch Abort.
- System mode is a privileged user mode for the operating system**.**
- Undefined mode.

INTRODUCTION OF IOT

The Internet of Things is a technology that has slowly gained momentum and is now silently shaping our future. IoT is the result of humankind's curiosity and intention to lead a convenient and connected lifestyle, reducing labor and eliminating the chances of human errors. That's why we decided to make devices smart and take care of things that will draw out efficiency. By making the devices connected to each other and the internet, we've let them collect and communicate data and make precise and informed decisions through Machine Learning and Neural Networks (complex mechanisms). This step has achieved outstanding outcomes.

Right now, there are billions of connected devices across the globe, collecting billions of petabytes of data every single day. These massive chunks of data are home to crucial pieces of information that can take care of home security, entertainment needs and go on to the extent of saving water and control fuel emissions. We have all had our experiences with IoT through smartphones, direct-to-home television services, smart televisions, and more.

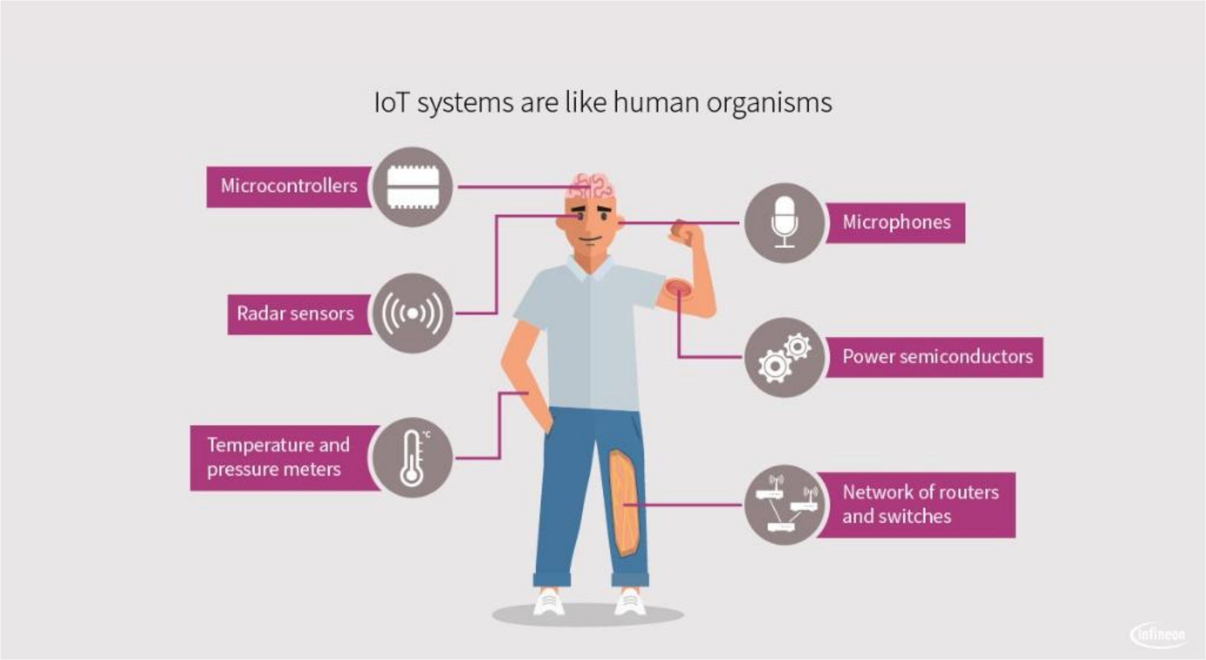
The Internet of Things (IoT) describes the network of physical objects—"things"—that are embedded with sensors, software, and other technologies for the purpose of connecting and exchanging data with other devices and systems over the internet. These devices range from ordinary household objects to sophisticated industrial tools. With more than 7 billion connected IoT devices today, experts are expecting this number to grow to 10 billion by 2020 and 22 billion by 2025. Oracle has a network of device partners.

https://www.rattletech.com/internet-of-things/ - PPT

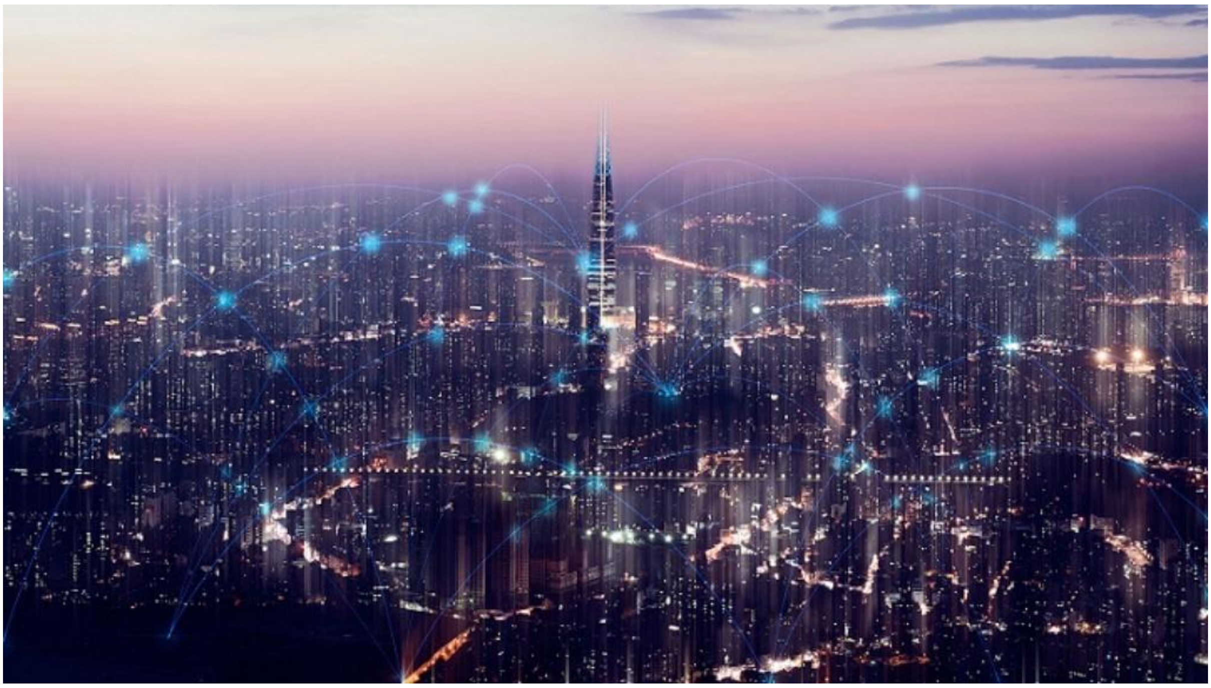
Definition: What is the Internet of Things?

The Internet of Things (IoT) unites physical objects with the virtual world. Intelligent devices and machines are connected to each other and the Internet. They capture relevant information about their direct environment, then analyze and link it. The devices perform specific tasks on that basis. A **sensor**, for example, measures the temperature outside and the smart device it is installed in responds by turning up the heating. All of that is done automatically, without users taking any actions themselves. Users can still control the IoT devices remotely if they wish, for example, using an app on their smartphone.

That is made possible by the interplay between connected components, such as **microcontrollers**, sensors and actuators, which convert electrical impulses into pressure, motion, temperature or other mechanical parameters. IoT systems are complex: They combine individual devices, databases and gateways linking multiple networks with each other. They are connected to the Internet, usually over a wireless interface, and send data or receive commands.


IoT systems are like human organisms
Microcontrollers
Radar sensors
Temperature and pressure meters
Microphones
Power semiconductors
Network of routers and switches
infineon

Every morning the shutters open at the appointed time, the heating in the bathroom is turned on automatically and the coffee machine starts brewing the first cup of coffee. When you set off for work, the garage opens by itself and the alarm system is activated. The connected car receives information about a traffic jam as you drive to work and suggests a new route on its own. And the production plant at the company communicates directly with the ordering system and logistics to ensure the relevant goods are made. All these connected, smart devices and machines are part of the Internet of Things.



How machines communicate with each other

Devices must be able to communicate with each other without any human intervention so that processes in the Internet of Things can be automated. A machine-to-machine (M2M) infrastructure enables information to be exchanged between vehicles, systems, machines, containers, electricity, gas and water meters, or robots – and also with a control center. M2M needs a data endpoint (DEP), in other words, the device or machine, a communications network and a data integration point (DIP), such as a server. One example: A plant produces goods and sends the quantity of raw materials to the server via WiFi. The server remotely monitors whether the machine requires servicing or new raw materials. To enable communication, the devices not only have sensors, but also a transmitter to send data over the communications network – via mobile communications, WiFi, a fixed-line network, Bluetooth, satellite or RFID. The receiver is a central system, such as a server. It gathers the information, processes it and initiates an action.

Why is Internet of Things (IoT) so important?

Over the past few years, IoT has become one of the most important technologies of the 21st century. Now that we can connect everyday objects—kitchen appliances, cars, thermostats, baby monitors—to the internet via embedded devices, seamless communication is possible between people, processes, and things.

By means of low-cost computing, the cloud, big data, analytics, and mobile technologies, physical things can share and collect data with minimal human intervention. In this hyperconnected world, digital systems can record, monitor, and adjust each interaction between connected things. The physical world meets the digital world—and they cooperate.

What technologies have made IoT possible?

While the idea of IoT has been in existence for a long time, a collection of recent advances in a number of different technologies has made it practical.

- **Cheap devices** – **Access to** simple sensors **low-cost, low-power sensor technology** to fully-certified network connectivity modules, the costs of IoT devices are low enough for both consumers & businesses.

  Connectivity. A host of network protocols for the internet has made it easy to connect sensors to the cloud and to other "things" for efficient data transfer. The presence of multiple wired & wireless networks in homes & offices, and across cities, enable these IoT devices to be cheaply & easily connected to the Internet.

  Cloud computing platforms. The increase in the availability of cloud platforms enables both businesses and consumers to access the infrastructure they need to scale up without actually having to manage it all. Moore's law has enabled computing in the cloud & at the edge to become cheaper & cheaper.

  Machine learning and analytics. With advances in machine learning and analytics, along with access to varied and vast amounts of data stored in the cloud, businesses can gather insights faster and more easily. The emergence of these allied technologies continues to push the boundaries of IoT and the data produced by IoT also feeds these technologies.

  Conversational artificial intelligence (AI). Advances in neural networks have brought natural-language processing (NLP) to IoT devices (such as digital personal assistants Alexa, Cortana, and Siri) and made them appealing, affordable, and viable for home use.

  IPv6 – Unlike the IPv4 protocol, which saw its last unassigned IP addresses allocated in 2011, the IPv6 protocol provides enough IP addresses to accommodate all the IoT devices that will ever be connected to the Internet.

Internet of Things Topology

The general topology of an end-to-end Internet of Things system consists of:

- IoT devices – Most IoT devices fall into one of these 2 categories (or are a combination of the two):
  1. Sensors – These devices are designed to monitor something specific in their surroundings & communicate the results to a computing device. Examples include sensors for motion, water/leaks, light, temperature, heat, etc.
  2. "Kinetic" devices – These devices are designed to perform an action when instructed to do so by a computing device. Examples include locks, actuators, alarms, etc.
- PAN/LAN – The IoT devices communicate with Edge devices over a Personal Area Network or a Local Area Network, over one or more of these common protocols – RFID, BLE, NFC, Zigbee, Z-Wave, Bluetooth, WiFi or Ethernet.(Near Field Communication (NFC) and Bluetooth Low Energy (BLE)

- Edge devices – If the IoT devices are not directly connected to the cloud, which is the case in most homes & offices, they communicate over the PAN or LAN with a smartphone or a gateway. A smartphone can be used to connect Bluetooth, BLE or NFC-based IoT devices on one side to the WAN on the other side. A gateway can be used to connect Zigbee, Z-Wave, Bluetooth, BLE, NFC & RFID-based IoT devices on one side to a wired or wireless LAN on the other side. These edge devices provide a bridge to the cloud, or in many cases, run software that directly analyzes the data from the IoT devices and controls them.

- WAN – The Wide Area Network connects the IoT devices to the cloud (through the edge devices), using cellular, cable, DSL (digital subscriber line) or fiber data connections.

- Cloud computing – The cloud is where all the complex analysis & control software resides. Due to the emergence of inexpensive cloud infrastructure that can be rented from the likes of Amazon, IBM, Microsoft & Google, complex home, office & city IoT solutions can be implemented at very low cost.

- Fog computing – In many cases, all the IoT analysis & control software cannot be implemented in the cloud, due to security, privacy, performance or other issues. In such cases, the software runs on the edge devices, bringing intelligence & processing closer to where the data is created.

Why are sensors that important for the Internet of Things?

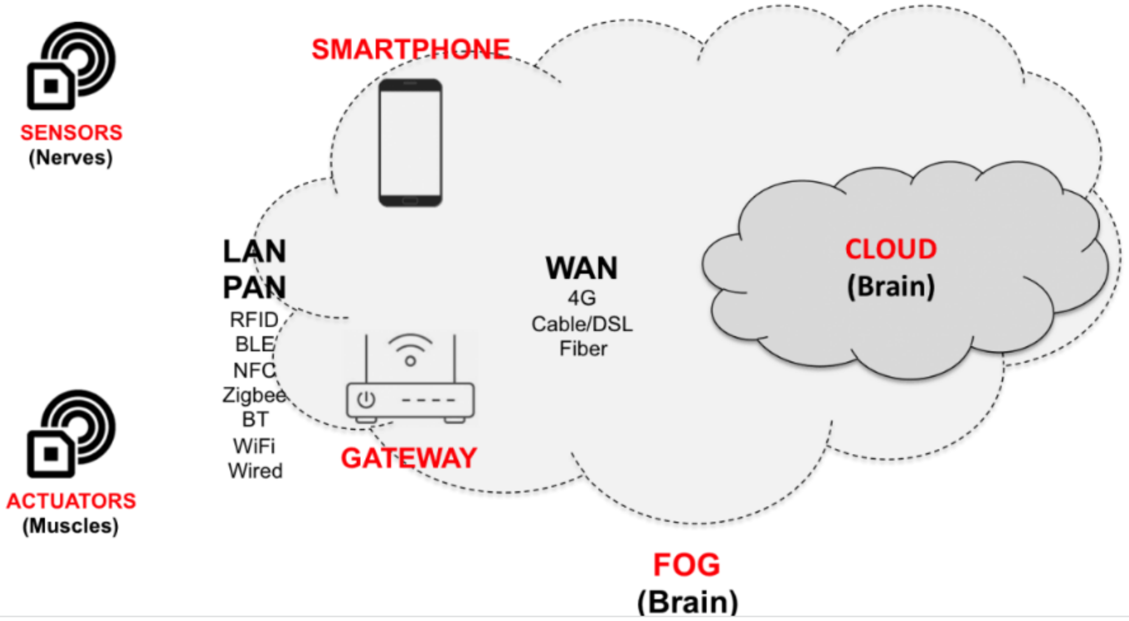
Whereas humans can perceive their environment with their senses, machines need sensors to do that, which is why sensors are the most important suppliers of data in the Internet of Things. They come in different types, for example, for determining the temperature, humidity, motion, light, mechanical pressure, CO_2 content, ultrasound or air pressure. The sensor on an object ascertains the status of its environment, and the microcontroller in the system processes the collected data. This data is then transmitted over the network to a software application. The sensor can be linked via Bluetooth to a smartphone app with which users can read the data. Or the information is sent over the Internet to a cloud platform on which the data is analyzed. If, for example, a sensor in the smart home indicates that it is getting dark, the shutters are lowered. Temperature sensors can be used in industry to precisely control heating or cooling valves, for instance. They measure how warm or cold a room or hall is and the valves react accordingly.

The rise of the IoT – it all began with a toaster-1991

1999- RFID (radio-frequency identification) is a technology where a device can read and store data from a tag in a non-contacting manner. Here, too, physical objects were therefore connected with the virtual world,

2000, the electronics company LG presented the idea of an Internet refrigerator

The network specialist Cisco calculated that by 2008 there were more devices connected to the Internet than people on Earth. That includes not only smartphones and computers, but all sorts of objects. More and more devices will be smart in the future: Around 75 billion devices worldwide will be connected to the Internet in 2025.



What does the Internet of Things enable?

The Internet of Things is transforming the world around us into a constantly-connected, intelligent, cohesive one. The 6 major areas undergoing transformation are:

1. Smart objects – By attaching low-cost RFID tags & Bluetooth beacons, "dumb" physical objects can be connected to the Internet, to help people nearby to learn about & interact with these objects
2. Smart homes – Will improve residents' safety & comfort, and lower their energy expenses
3. Smart buildings – Will lower operating costs (energy, water), implement predictive & preemptive maintenance, and improve workforce's productivity
4. Smart cars – Will improve riders' productivity & safety, and lower city congestion & pollution
5. Smart cities – Will lower operating costs (energy, water), improve workforce's productivity, improve air quality, and improve safety
6. Smart medicine – Will lower patients' costs (remote diagnosis & treatment), improve response to intermittent patient symptoms, improve medical staff's productivity

But can we call a system that remotely monitors & controls IoT devices "smart"? In a general sense, yes, but the tremendous advances in artificial intelligence & machine learning will make these systems truly smart. Sophisticated AI & machine-learning-based algorithms running both in the cloud & at the edge (fog computing) will be able to analyze and understand the situation, and then react appropriately, without any human intervention.

What are the challenges facing the Internet of Things?

The Internet of Things promises to change the way we live, but it does come with significant challenges that will need to be overcome to make sure that the benefits outweigh the risks. The main challenges are:

1. Security – Hackers have already managed to penetrate unsecured smart home devices to install malware that then attacks other Internet sites. The problem will get a lot worse if critical connected devices in offices, hospitals, cities, etc. get compromised. For the Internet of Things to succeed, a robust set of security schemes & protocols need to be enabled, and diligently implemented.
2. Data ownership – A battle has already started brewing on who owns the vast amount of data generated by connected cars – car manufacturers want to own the data, but the independent service & repair shops also want access, so they don't get locked out of future revenue. New partnerships & business models will need to be created to share the data without causing data security issues.
3. Powering billions of IoT devices – To be truly valuable, IoT devices will need to be continuously on & connected, so they are collecting data & acting on it throughout the day. Continuously powering these billions of devices in buildings & cities will need new power distribution & management schemes.
4. Handling all the resulting e-waste – Our e-waste from frequently-replaced computers, mobile phones, tablets & TVs is already polluting drinking water & causing harm to ecosystems around the world. The additional e-waste from billions of discarded IoT devices is going to make the problem exponentially worse. We need new strategies & methods to limit & dispose the resulting e-waste, if we are to benefit from the Internet of Things without harming life on earth.

Over the next few years, the Internet of Things will make our homes, offices, cars, cities & even our bodies better. But the true holy grail of IoT will be when all these individual smart areas start communicating & collaborating with each other to form a single cohesive system – a "smart world".

https://www.oracle.com/in/internet-of-things/what-is-iot/

What is industrial IoT?

Industrial IoT (IIoT) refers to the application of IoT technology in industrial settings, especially with respect to instrumentation and control of sensors and devices that engage cloud technologies. Refer to thisTitan use case PDF for a good example of IIoT. Recently, industries have used machine-to-machine communication (M2M) to achieve wireless automation and control. But with the emergence of cloud and allied technologies (such as analytics and machine learning), industries can achieve a new automation layer and with it create new revenue and business models. IIoT is sometimes called the fourth wave of the industrial revolution, or Industry 4.0. The following are some common uses for IIoT:

- Smart manufacturing
- Connected assets and preventive and predictive maintenance
- Smart power grids
- Smart cities
- Connected logistics
- Smart digital supply chains

What are IoT applications?

Business-ready, SaaS IoT Applications

IoT Intelligent Applications are prebuilt software-as-a-service (SaaS) applications that can analyze and present captured IoT sensor data to business users via dashboards. We have a full set of IoT Intelligent Applications.
IoT applications use machine learning algorithms to analyze massive amounts of connected sensor data in the cloud. Using real-time IoT dashboards and alerts, you gain visibility into key performance indicators, statistics for mean time between failures, and other information. Machine learning–based algorithms can identify equipment anomalies and send alerts to users and even trigger automated fixes or proactive counter measures.

With cloud-based IoT applications, business users can quickly enhance existing processes for supply chains, customer service, human resources, and financial services. There's no need to recreate entire business processes.

https://www.infineon.com/cms/en/discoveries/internet-of-things-basics/

LIST OF TOP 15 APPLICATIONS OF IOT

1. Smart Homes

One of the best and the most practical applications of IoT, smart homes really take both, convenience and home security, to the next level. Though there are different levels at which IoT is applied for smart homes, the best is the one that blends intelligent utility systems and entertainment together. For instance, your electricity meter with an IoT device giving you insights into your everyday water usage, your set-top box that allows you to record shows from remote, Automatic Illumination Systems, Advanced Locking Systems, Connected Surveillance Systems all fit into this concept of smart homes. As IoT evolves, we can be sure that most of the some vital concerns like Traffic Management, Waste Management, Water Distribution, Electricity Management, and devices will become smarter, enabling enhanced home security.

2. Smart City

Not just internet access to people in a city but to the devices in it as well – that's what smart cities are supposed to be made of. And we can proudly say that we're going towards realizing this dream. Efforts are being made to incorporate connected technology into infrastructural requirements and more. All these work towards eliminating some day-to-day challenges faced by people and bring in added convenience.

- Connected public transport (74%)
- Traffic monitoring and management (72%)
- Water level / Flood monitoring (72%)

- Video surveillance and analytics (72%)
- Connected streetlights (68%)
- Weather monitoring (68%)
- Air quality / Pollution monitoring (68%)
- Smart metering – water (66%)
- Fire / Smoke detection (66%)
- Water quality monitoring (64%)

3. Self-driven Cars

We’ve seen a lot about self-driven cars. Google tried it out, Tesla tested it, and even Uber came up with a version of self-driven cars that it later shelved. Since it’s human lives on the roads that we’re dealing with, we need to ensure the technology has all that it takes to ensure better safety for the passenger and those on the roads.

The cars use several sensors and embedded systems connected to the Cloud and the internet to keep generating data and sending them to the Cloud for informed decision-making through Machine Learning. Though it will take a few more years for the technology to evolve completely and for countries to amend laws and policies, what we’re witnessing right now is one of the best applications of IoT.

4. IoT Retail Shops

If you haven’t already seen the video of Amazon Go – the concept store from the eCommerce giant, you should check it out right away. Perhaps this is the best use of the technology in bridging the gap between an online store and a retail store. The retail store allows you to go cashless by deducting money from your Amazon wallet. It also adds items to your cart in real-time when you pick products from the shelves.

If you change your mind and pick up another article, the previous one gets deleted and replaces your cart with the new item. The best part of the concept store is that there is no cashier to bill your products. You don’t have to stand in line but just step out after you pick up your products from shelves. If this technology is effective enough to fetch more patronage, this is sure to become a norm in the coming years.

5. Farming

Farming is one sector that will benefit the most from the Internet of Things. With so many developments happening on tools farmers can use for agriculture, the future is sure promising. Tools are being developed for Drip Irrigation, understanding crop patterns, Water Distribution, drones for Farm Surveillance, and more. These will allow farmers to come up with a more productive yield and take care of the concerns better.

6. Wearables

Wearables remain a hot topic in the market, even today. These devices serve a wide range of purposes ranging from medical, wellness to fitness. Of all the IoT startups, Jawbone, a wearables maker, is second to none in terms of funding.

7. Smart Grids and energy saving.

One of the many useful IoT examples, a smart grid, is a holistic solution that applies an extensive range of Information Technology resources that enable existing and new gridlines to reduce electricity waste and cost. A future smart grid improves the efficiency, reliability, and economics of electricity.

The progressive use of intelligent energy meters, or meters equipped with sensors, and the installation of sensors in different strategic points that go from the production plants to the different distribution points, allows better monitoring and control of the electrical network.

By establishing a bidirectional communication between the service provider company and the end user, information of enormous value can be obtained for the detection of faults, decision making and repair thereof.

It also allows offering valuable information to the end user about their consumption patterns and about the best ways to reduce or adjust their energy expenditure.

8. Industrial Internet

The Industrial Internet of Things consists of interconnected sensors, instruments, and other devices connected with computers' industrial applications like manufacturing, energy management, etc. While still being unpopular in comparison to IoT wearables and other uses, market researches like Gartner, Cisco, etc., believe the in dustrial internet to have the highest overall potential.

9. Telehealth - (Digital health/Telehealth/Telemedicine)

Telehealth, or Telemedicine, hasn't completely flourished yet. Nonetheless, it has great future potential. IoT Examples of Telemedicine include the digital communication of Medical Imaging, Remote Medical Diagnosis & Evaluations, Video Consultations with Specialists, etc.

10. Smart Supply-chain Management

Supply-chains have stuck around in the market for a while now. A common example can be Solutions for tracking goods while they are on the road. Backed with IoT technology, they are sure to stay in the market for the long run.

11. Traffic monitoring.

The Internet of things can be very useful in the management of vehicular traffic in large cities, contributing to the concept of smart cities.

When we use our mobile phones as sensors, which collect and share data from our vehicles through applications such as Waze or Google Maps, we are using the Internet of Things to inform us and at the same time contribute to traffic monitoring, showing the conditions of the different routes, and feeding and improving the information on the different routes to the same destination, distance, estimated time of arrival.

12. Fleet management.

The installation of sensors in fleet vehicles helps to establish an effective interconnectivity between the vehicles and their managers as well as between the vehicles and their drivers. Both driver and manager/ owner can know all kinds of details about the status, operation and needs of the vehicle, just by accessing the *software in* charge of collecting, processing and organizing the data. Even, receive alarms in real time of maintenance incidents without having been detected by the driver.

The application of the Internet of Things to fleet management assists with geolocation (and with it the monitoring of routes and identification of the most efficient routes), performance analysis, telemetry control and fuel savings , the reduction of polluting emissions to the environment and can even provide valuable information to improve the driving of vehicles.

13. Hospitality.

The application of the IoT to the hotel industry brings with it interesting improvements in the quality of the service. With the implementation of electronic keys, which are sent directly to the mobile devices of each guest, it is possible to automate various interactions.

Thus, the location of the guests, the sending of offers or information on activities of interest, the realization of orders to the room or *room service* , the automatic charge of accounts to the room or the request of personal hygiene supplies, are activities that can be easily managed through integrated applications using the Internet of Things technology.

With the use of electronic keys, the *check-out* process is automated, disabling the operation of doors, offering information about the rooms immediately available, and even assigning housekeeping tasks to maintenance personnel.

14. Water supply.

A sensor, either incorporated or adjusted externally to water meters, connected to the Internet and accompanied by the necessary *software* , helps to collect, process and analyze data, which allows understanding the behavior of consumers, detecting faults in the supply service, report results and offer courses of action to the company that provides the service.

Likewise, it offers final consumers the possibility of tracking their own consumption information, through a web page and in real time, even receiving automatic alerts in case of detecting consumption out of range to their average consumption record, which could indicate the presence of a leak.

15. Maintenance management.

One of the areas where the application of IoT technology is most extensive is precisely maintenance management. Through the combination of sensors and ***software*** specialized in **CMMS/ EAM** maintenance management, a multifunctional tool is obtained whose use can be applied to a multiplicity of disciplines and practices, with the purpose of extending the useful life of physical assets, while guaranteeing asset reliability and availability.

When the characteristics of the *software in* charge of processing and arranging the data collected by the sensors are designed to specifically address the maintenance management needs of physical assets, their application is almost unlimited.

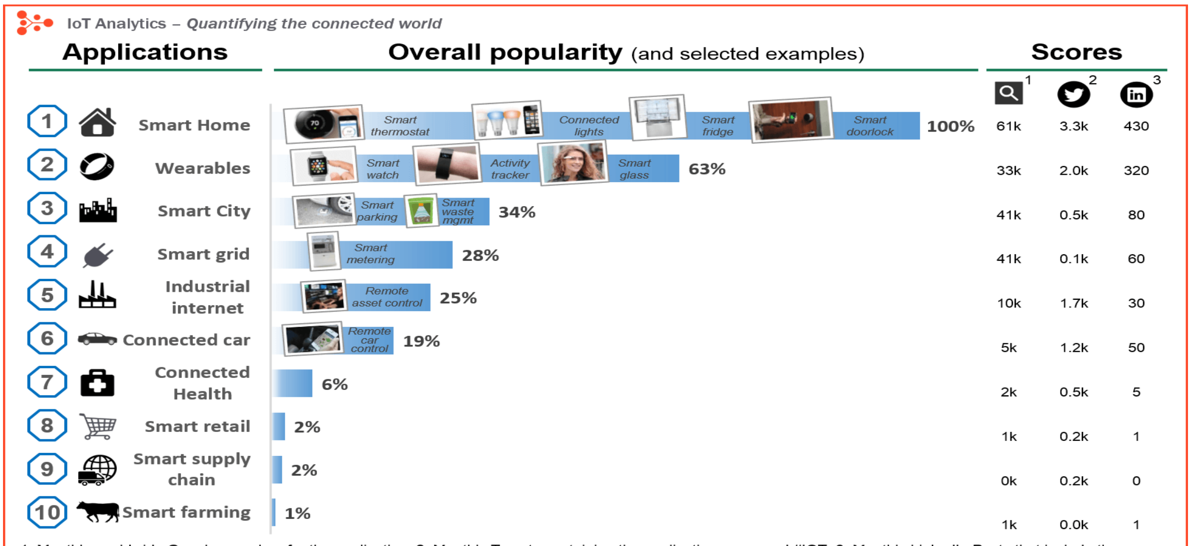
The real-time monitoring of physical assets allows determining when a measurement is out of range and it is necessary to perform condition-based maintenance (CBM), or even applying Artificial Intelligence (AI) algorithms such as *Machine Learning* or *Deep Learning* to predict the failure before it happens.

Scope & Future of IoT

As you know, IoT is evolving and is being experimented with and used in tons of different ways than we can ever imagine. Some IoT examples can be Smart Breweries, Smart Coffee Machines, Smart Parking Facilities, Smart Supply-chain Mechanisms, and more.

The Internet of Things will keep on growing. **Connected cars** will find the quickest and safest routes in the future. Connected streetlights will record data on traffic, safety, lighting and even the air quality. **Connected air taxis** will take people in cities from A to B. This development will be driven by expansion of the new high-speed mobile network **5G**, which enables faster and more stable data transmission. That will be a must, since the greater the number of devices that communicate with each other, the greater the volume of data there will be. The 5G mobile network standard can cope with that torrent of data and process it in the cloud.

Edge computing could also grow in importance moving ahead: IoT devices will then be able to process data directly on the device where it is generated. That means data can be analyzed faster, for example, in the connected car, than in the cloud. Artificial intelligence (AI) will also drive the Internet of Things. Computers and algorithms will then be able to handle problems on their own and become better and better ("machine learning"). Four-out-of-ten industrial companies already use data analytics and AI for digital product development. It is therefore likely that the Internet of Things will only be able to unfurl its full potential when it is combined with artificial intelligence. Companies already learn how to improve their products by analyzing data. However, they can boost that ability if machines and algorithms can detect patterns on their own.



1. Monthly worldwide Google searches for the application 2. Monthly Tweets containing the application name and #IOT 3. Monthly LinkedIn Posts that include the application name. All metrics valid for Q4/2014.
Sources: Google, Twitter, LinkedIn, IoT Analytics

IoT Sensors

Sensors are everywhere. They’re in our homes and workplaces, our shopping centers and hospitals. They’re embedded in smart phones and an integral part of the Internet of Things (IoT). Sensors have been around for a long time. The first thermostat was introduced in the late 1880s and infrared sensors have been around since the late 1940s. The IoT and its counterpart, the Industrial Internet of Things (IIoT), are bringing sensor usage to a new level.

Broadly speaking, sensors are devices that detect and respond to changes in an environment. Inputs can come from a variety of sources such as light, temperature, motion and pressure. Sensors output valuable information and if they are connected to a network, they can share data with other connected devices and management systems.

Sensors are crucial to the operation of many of today’s businesses. They can warn you of potential problems before they become big problems, allowing businesses to perform predictive maintenance and avoid costly downtime. The data from sensors can also be analyzed for trends allowing business owners to gain insight into crucial trends and make informed evidence-based decisions.

Sensors come in many shapes and sizes. Some are purpose-built containing many built-in individual sensors, allowing you to monitor and measure many sources of data. In brownfield environments, it’s key for sensors to include digital and analog inputs so that they can read data from legacy sensors.

How does an IoT sensor work?

An IoT system consists of sensors/devices which **“talk” to the cloud through some kind of connectivity**. Once the data gets to the cloud, software processes it and then might decide to perform an action, such as sending an alert or automatically adjusting the sensors/devices without the need for the user.

Sensors used within these devices are meant to detect, measure and report *one* real-world variable at a time. Their entire purpose is to help us understand the world around us in a measurable, universal way that "anybody" can understand.

There are many types of IoT sensors and an even greater number of applications and use cases. Here are 10 of the more popular types of IoT sensors and some of their use cases.

IoT sensors have become critical to improving operational efficiency, reducing costs and enhancing worker safety.

1. Temperature Sensors

Temperature sensors measure the amount of heat energy in a source, allowing them to detect temperature changes and convert these changes to data. Machinery used in manufacturing often requires environmental and device temperatures to be at specific levels. Similarly, within agriculture, soil temperature is a key factor for crop growth.

2. Humidity Sensors

These types of sensors measure the amount of water vapor in the atmosphere of air or other gases. Humidity sensors are commonly found in heating, vents and air conditioning (HVAC) systems in both industrial and residential domains. They can be found in many other areas including hospitals, and meteorology stations to report and predict weather.

Temperature & Humidity Sensors

The heavy-duty Laird industrial IoT temperature and humidity sensor
Temperature and humidity sensors are just as popular and widely used as our first two picks. They usually come bundled together in pre-made IoT modules.

On one hand, temperature sensors measure the amount of heat energy in a source and they are meant to measure temperature changes. On the other hand, humidity sensors measure the amount of water vapor in the atmosphere of various gases.

Temperature monitoring is a common use case in industrial settings where machines need to operate at a certain temperature for long periods of time.

Other use cases we typically see within our own customer base are:

1. Food Safety Compliance
2. Cold-chain Monitoring in Healthcare and Hospitality
3. Warehouse & Inventory Management
4. HVAC Systems Monitoring

Humidity is often tracked alongside temperature as well. Measuring the former is helpful with heating, air conditioning, weather stations, and even soil moisture.

Examples of IoT temperature and humidity sensors:

- The Sentrius™ RS1xx from Laird (Industrial)
- The no-brand DHT-22 2302 (DIY)

3. Pressure Sensors

A pressure sensor senses changes in gases and liquids. When the pressure changes, the sensor detects these changes, and communicates them to connected systems. Common use cases include leak testing which can be a result of decay. Pressure sensors are also useful in the manufacturing of water systems as it is easy to detect fluctuations or drops in pressure.

They are used to measure pressure (the force required to stop a fluid from expanding) in gases or liquids. These can come in all sizes and shapes, and they are one of the most popular examples of IoT sensors mainly due to industrial applications that are fully embracing this new connectivity effort.

Pressure sensors can be of different types:

- Barometric pressure sensors, contained in most weather stations. These are meant to measure changes in atmospheric pressure.
- Gas pressure sensors meant to monitor pressure changes in gases, especially in oil, energy, and utility applications.

Though not technically pressure transducers, load cells are also a variation of pressure sensors, and can be a preferred method when it comes to measuring weight (e.g. animal weight, or the level of a tank or silo).

These types of sensors form the backbone of our gas and energy infrastructure because, without them, we wouldn't be able to monitor system pressure. Hook an IoT module to one of these and your data is ready to go on your computer screen.

Examples of IoT pressure sensors:

- The Long Range Wireless Pressure Sensor from NCD (Industrial)
- The E8PC from Omron (Industrial)

4. Proximity Sensors

Proximity sensors are used for non-contact detection of objects near the sensor. These types of sensors often emit electromagnetic fields or beams of radiation such as infrared. Proximity sensors have some interesting use cases. In retail, a proximity sensor can detect the motion between a customer and a product in which he or she is interested. The user can be notified of any discounts or special offers of products located near the sensor. Proximity sensors are also used in the parking lots of malls, stadiums and airports to indicate parking availability. They can also be used on the assembly lines of chemical, food and many other types of industries.

Proximity & Motion Sensors

An industrial IoT proximity sensor from NCD
As a very common type of device, a proximity sensor is used in countless IoT applications. Proximity sensors are able to detect the presence of nearby objects without any physical contact by emitting an electromagnetic field or a beam of electromagnetic radiation (e.g. infrared) and looking for changes in the field.

Also known as range sensors, some of these devices use ultrasonic waves to measure the distance between themselves and the object detected. A few great examples of these are manufactured by Maxbotix.

Proximity sensors can be confused with motion sensors, but they are not the same thing. While a proximity sensor may be used to detect motion by measuring the distance from an object, PIR (Pyroelectric InfraRed) sensors are a good low-cost, specialized alternative for motion detection as their output is mostly binary: a 1 or a 0.

These are typically used in motion detectors which can be employed for anything from turning a light on to alerting the police in case of suspect motion.

Examples of IoT proximity and motion sensors:

- The Long Range Wireless Proximity & Light Sensor from NCD (Industrial)
- Various Proximity Sensors from Seeed (DIY & Modular)

5. Level Sensors

Level sensors are used to detect the level of substances including liquids, powders and granular materials. Many industries including oil manufacturing, water treatment and beverage and food manufacturing factories use level sensors. Waste management systems provide a common use case as level sensors can detect the level of waste in a garbage can or dumpster.

6. Accelerometers

Accelerometers detect an object’s acceleration i.e. the rate of change of the object’s velocity with respect to time. Accelerometers can also detect changes to gravity. Use cases for accelerometers include smart pedometers and monitoring driving fleets. They can also be used as anti-theft protection alerting the system if an object that should be stationary is moved.

7. Gyroscope

Gyroscope sensors measure the angular rate or velocity, often defined as a measurement of speed and rotation around an axis. Use cases include automotive, such as car navigation and electronic stability control (anti-skid) systems. Additional use cases include motion sensing for video games, and camera-shake detection systems.

Gyroscopic & Acceleration Sensors

The ADXL335 accelerometer from Analog

Many of you will already be acquainted with gyroscopes and accelerometers. They're the sensors that allow your smartphone to sense whether your phone is upright or in landscape mode, and they're widely used in mobile game design.

In terms of IoT applications, these little sensors are just as popular as the ones used in your phone. Even though they bear differences, most accelerometer chips also ship with a gyroscope, hence why we combined them into one section.

The difference between the two is the following:

A gyroscope is a device that makes use of Earth's gravity to help determine orientation. Its design consists of a freely-rotating disk called a rotor, mounted onto a spinning axis in the center of a larger and more stable wheel. As the axis turns, the rotor remains stationary to indicate the central gravitational pull, and thus which way is down.

On the other hand:

An accelerometer is a compact device designed to measure non-gravitational acceleration. When the object it's integrated into goes from a standstill to any velocity, the accelerometer is designed to respond to the vibrations associated with such movement.

In IoT, accelerometers are at the heart of vibration sensors which can turn acceleration data into vibration frequencies (that's Gs into Hz!), commonly used to detect subnormal industrial machine operations.

Gyroscopes can be used to determine the moving direction of a GPS-tracked object, as well as wind direction in weather and clean energy applications.

Examples of IoT gyroscopic and acceleration sensors:

- The KX Series of Accelerometers from Kionix (Industrial)
- The ADXL335 Accelerometer from Analog Devices (DIY)

8. Gas Sensors

These types of sensors monitor and detect changes in air quality, including the presence of toxic, combustible or hazardous gasses. Industries using gas sensors include mining, oil and gas, chemical research andmanufacturing. A common consumer use case is the familiar carbon dioxide detectors used in many homes.

Flow & Gas Sensors

Useful in many different cases, gas IoT sensors are on the rise (Source: Spec Sensors)

Flow sensors are devices used for measuring the flow rate or quantity of a moving liquid or gas and they're somewhat related to gas sensors which are electronic devices that detect and identify different types of gases.

Flow sensors encompass all types of devices used to measure liquid and gas flows. Use cases include industrial process monitoring, HVAC, as well as gas and water management applications. The same goes for gas sensors.

Smart metering is one of the areas affected most by flow sensors. Here, ultrasonic flow meters are paired with IoT modules to send data to a remote location.

With the advent of cellular IoT technologies like NB-IoT and LTE-M—and the obvious benefits for utility companies and end-users alike—flow, gas, and electric sensors are expected to grow significantly by 2025.

This clearly shows a strong trend in smart metering as well as an overall increase in usage among these types of sensors for industrial purposes.

Looking at gas sensors in specific, these are traditionally bulky devices that only recently turned into low-power adaptations to monitor things like air quality in local environments. The topic of air quality is further discussed at list item 11.

Examples of IoT gas and flow sensors:

- Various Flow Sensors from Sensirion (Industrial)
- Various Gas Sensors from Spec Sensors (DIY & Modular)

9. Infrared Sensors

These types of sensors sense characteristics in their surroundings by either emitting or detecting infrared radiation. They can also measure the heat emitted by objects. Infrared sensors are used in a variety of different IoT projects including healthcare as they simplify the monitoring of blood flow and blood pressure. Televisions use infrared sensors to interpret the signals sent from a remote control. Another interesting application is that of art historians using infrared sensors to see hidden layers in paintings to help determine whether a work of art is original or fake or has been altered by a restoration process.

10. Optical Sensors

Optical sensors convert rays of light into electrical signals. There are many applications and use cases for optical sensors. In the auto industry, vehicles use optical sensors to recognize signs, obstacles, and other things that a driver would notice when driving or parking. Optical sensors play a big role in the development of driverless cars. Optical sensors are very common in smart phones. For example, ambient light sensors can extend battery life. Optical sensors are also used in the biomedical field including breath analysis and heart-rate monitors.

Light Sensors

Smart light IoT sensors are designed for industrial and consumer use (Source: Enlighted)

Also known as Photoelectric Devices or Photo Sensors, light sensors are more common than you'd think. These are passive photoelectric devices that convert light energy (photons) into electrical energy (electrons).

However, there's more to these devices than their working principle. Just like pressure sensors, light sensors can serve all kinds of purposes and are used heavily in brightness control, security, and even agriculture.

Keeping track of changes in light is useful for weather monitoring as well as applications in agriculture where measuring the light absorbed by the soil is key.

These sensors can also be a simpler, cost-effective alternative to motion sensors, allowing for presence detection, say, in a hotel room, warehouse, or hallway.

Seeed Studio recently published an in-depth article on the topic of light sensors, with relevant hardware examples from their own product lineup.

Examples of IoT light sensors:

- The Smart Sensors from Enlighted (Consumer & Enterprise)
- The Long Range Ambient Light Sensor from NCD (Industrial)

11. Sound Sensors

The aptly-named SoundSensor from SensorTeam

Sound sensors aren't as used as other types of sensors in our list, but they're still worth a mention due to their unique properties and interesting use cases.

A sound sensor is defined as a module that detects sound waves through its intensity, converting it to electrical signals. When the device detects a change in intensity, it can send the data back to your dashboard.

Simple sound sensors like the ones offered from Seeed Studio are fairly inexpensive and they can help you figure out a few different use cases. For example, detecting the presence of sound in an otherwise quiet room.

There are also sound recorders available which not only detect sound but also record it as soon as the intensity changes. That's particularly useful in security.

Lastly, an advanced use case involves the so-called Sound/Noise Level Meters; devices that assess noise across a frequency range.

These allow for more complex ambient noise metering applications.

Examples of IoT sound sensors:

- The aptly-named SoundSensor from SensorTeam (Construction)
- The Sound & Noise IoT Sensor from IoTsens (Smart Cities)

12. Moisture Sensors

Moisture sensors are important for use cases in smart agriculture (Source: Sensoterra)
Moisture sensors are key to recent advancements in agriculture, allowing farmers to constantly monitor soil health. As AgriTech Tomorrow points out, soil conditions change constantly throughout the growing season.

Here's what they have to say about IoT soil sensors:

...because of recent developments in soil and water monitoring, the critical information is being received in real-time measurements from the field, helping farmers make faster, more accurate crop production decisions.

Most moisture sensors are single-point, meaning that they measure moisture in the soil in a singular, fixed location.

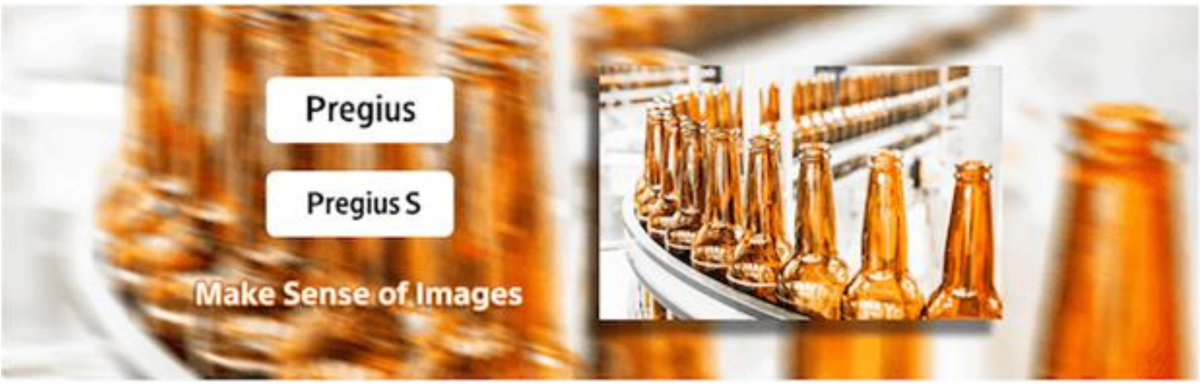
This type of sensor should be installed in multiple locations across a crop field to increase the accuracy of the measurements.

Depth also plays an important role here as the effects of irrigation can be stronger or weaker at different levels. Moisture devices are usually equipped with long probes that go deep into the soil and measure moisture at different depths.

Examples of IoT moisture sensors:

- The Wireless Soil Moisture Sensors from Sensoterra (Agriculture)
- The Long Range IoT Soil Moisture Sensor from NCD (Agriculture)

13. Image Sensors



The advanced Pregius industrial IoT image sensor technology from Sony
A sensor is a device gathering information about the physical world to provide a set of measurements over time. Under this same logic, we can think of a camera as a 2-dimensional optical/IR sensor providing a matrix of measurements over time.

Cameras are widely used as a sensing mechanism, from people counting applications to AI-powered pattern recognitions.

For example, a unique take on this is the PigVision technology from Asimetrix: an image-based sensor measuring pig weight in real-time based on trained AI models.

Here's a video from the explaining how the technology works:

PigVision is a great example of what image sensors can achieve
As shown in the screenshot above, Sony is also a big proponent of these technologies in industrial, utility, and aviation settings. You can check out their Pregius image technology from the resources down below.

Examples of IoT image sensors:

- The OS02F10 from OmniVision (Industrial)
- Sony's Pregius Line of Image Sensors (Industrial)

14. Magnetic Sensors

The CT8xx Series of magnetic IoT sensors from Crocus Technology
Magnetic sensors are used in both consumer and industrial applications to detect large bodies of metal such as cars, panels, housing, etc. These devices detect changes and disturbances in a magnetic field like flux, strength and direction.

There are three types of magnetic sensors typically used in IoT:

1. TMR (Tunneling Magnetoresistive) Sensors
2. Reed Switches
3. Hall Effect Sensors

Tunnel magnetoresistance is a fairly complex quantum mechanical phenomenon so we won't get into the scientific details of it in this article.

However, TMR sensors are becoming increasingly popular for measuring mechanical displacement and motion in industrial, automotive (think automated parking and self-driving cars), and consumer applications.

Our second entry—the reed switch—is an electrical switch operated by a magnetic field. These switches can be actuated by an electromagnetic coil and they are commonly used in security systems to control the flow of electricity and trigger an action in case the presence of an unwanted visitor is detected.

Lastly, we have the hall effect sensor, a device used to measure the magnitude of a magnetic field. Diodes has a great article explaining their role in IoT.

Also, a few years back we ran an interesting project using two different magnetic sensors characterized by low power consumption and high sensitivity:

1. The MMC5883MA from Memsic
2. The LSM303AGR from STMicroelectronics

Check out the series for some hands-on action with magnetic sensors!

Examples of IoT magnetic sensors:

- Custom Magnetic Sensors from HyperTech (Asset Tracking)
- The CT8xx Series from Crocus Technology (Energy Sector)

15. Air Quality Sensors

Air quality sensors can be used in industrial and consumer settings (Source: Ambient Weather)
Even though air and water quality are functions of several sensor types used in conjunction, we thought they both deserved a category of their own.

The first thing that comes to mind when we talk about air quality is pollution. But how do we measure it? **By using so-called Particulate Matter sensors.** These are comprised of mainly two types based on size of the particles found in the air:

1. PM10
2. PM2.5

Particulate matter sensors that can detect the former are more common as the particles are larger (generally 10 micrometers, hence the name). Detecting fine particles that are 2.5 micrometers and smaller becomes gradually harder.

Industrial particulate matter sensors from Sensirion and other manufacturers can detect these fine particles and help with implementing air quality systems.

As discussed in previous sections, gas sensors also play a huge role in recognizing changes in the air, and they're usually embedded together with PM sensors.

Examples of IoT air quality sensors:

- Various Air Quality Sensors from ams (Industrial)
- The Ambient Weather PM2.5 Particulate Monitor (Smart Home)

16. Water Quality Sensors

A variety of sensors can help with sensing water quality as well (Source: Georg Fischer) Water is something that most of us take for granted, but many populations in the world do not have clear access to this crucial resource. And if they do, its quality could be poor or outright toxic. That's where water quality sensors come in.

Like the previous section, there are several variables that contribute to water quality, some of which are subject to physical measurements via sensors like:

1. pH Sensors
2. Turbidity Sensors
3. ORP (Oxidation-Reduction Potential) Sensors

When used in conjunction, these sensors form a complete picture of the water's quality. For example, here's what the USGS has to say about pH:

pH is a measure of the relative amount of free hydrogen and hydroxyl ions in the water. Water that has more free hydrogen ions is acidic, whereas water that has more free hydroxyl ions is basic. pH can be affected by chemicals in the water and is an important indicator of water that is changing chemically.

Based on this, pH sensors play a huge role in monitoring water quality by looking at its alkalinity. Turbidity sensors play a similar role by measuring the amount of light scattered by suspended solids in water. Less light means lower quality.

Finally, Oxidation-Reduction Potential sensors measure the ability of a solution to act as an oxidizing or reducing agent. As pointed out by Vernier, these may be used in swimming pools to measure the oxidizing ability of chlorine.

Examples of IoT water quality sensors:

- The Signet 2724-2726 from Georg Fischer (Industrial)
- The Waspmote Water Quality Module from Libelium (Various)

These were the 12 types of sensors that are most relevant in the IoT world.

MYTHINGS IoT Sensor

The MYTHINGS Smart Sensor is a self-contained, battery-powered multi-purpose IoT sensor that allows you to capture critical data points like acceleration, temperature, humidity, pressure and GPS. The smart sensor is integrated with the MYTHINGS Library – a hardware independent, small-footprint and power-optimized library of code, featuring the MIOTY (TS-UNB) low-power wide area network protocol.

Fig. A pilot's cockpit filled with flight instruments such as altimeters and fuel levels

These dashboards are filled with flight instruments reporting variable data from all types of sensors. A few examples are fuel levels, system pressure, and so on.

The idea behind next-generation IoT sensors is that through the use of so-called IoT modules, these devices can aggregate and send their data directly to an IoT dashboard, making it much faster and easier to make important decisions.

Predictive maintenance is one of the key areas where IoT sensors are promising to change the way manufacturing and industry are operated. Other interesting use cases are rising in popularity as well.

Challenges in Internet of things (IoT)

The Internet of Things (IoT) has fast grown to be a large part of how human beings live, communicate and do business. All across the world, web-enabled devices are turning our global rights into a greater switched-on area to live in. There are various types of challenges in front of IoT.

Security challenges in IoT :

1. **Lack of encryption** – Although encryption is a great way to prevent hackers from accessing data, it is also one of the leading IoT security challenges. These drives like the storage and processing capabilities that would be found on a traditional computer. The result is an increase in attacks where hackers can easily manipulate the algorithms that were designed for protection.
2. **Insufficient testing and updating** – With the increase in the number of IoT(internet of things) devices, IoT manufacturers are more eager to produce and deliver their device as fast as they can without giving security too much of although. Most of these devices and IoT products do not get enough testing and updates and are prone to hackers and other security issues.
3. **Brute forcing and the risk of default passwords** – Weak credentials and login details leave nearly all IoT devices vulnerable to password hacking and brute force. Any company that uses factory default credentials on their devices is placing both their business and its assets and the customer and their valuable information at risk of being susceptible to a brute force attack.
4. **IoT Malware and ransomware** – Increases with increase in devices. Ransomware uses encryption to effectively lock out users from various devices and platforms and still use a user's valuable data and info. **Example** – A hacker can hijack a computer camera and take pictures. By using malware access points, the hackers can demand ransom to unlock the device and return the data.
5. **IoT botnet aiming at cryptocurrency** – IoT botnet workers can manipulate data privacy, which could be massive risks for an open Crypto market. The exact value and creation of cryptocurrencies code face danger from mal-intentioned hackers.

The blockchain companies are trying to boost security. Blockchain technology itself is not particularly vulnerable, but the app development process is.

Design challenge in IoT :

1. **Battery life is a limitation** –
   Issues in packaging and integration of small-sized chip with low weight and less power consumption. If you've been following the mobile space, you've likely see how every yr it looks like there's no restriction in terms of display screen size. Take the upward thrust of 'phablets', for instance, which can be telephones nearly as huge as tablets. Although helpful, the bigger monitors aren't always only for convenience, rather, instead, display screen sizes are growing to accommodate larger batteries. Computers have getting slimmer, but battery energy stays the same.
2. **Increased cost and time to market** –
   Embedded systems are lightly constrained by cost.
   The need originates to drive better approaches when designing the IoT devices in order to handle the cost modelling or cost optimally with digital electronic components. Designers also need to solve the design time problem and bring the embedded device at the right time to the market.
3. **Security of the system** –
   Systems have to be designed and implemented to be robust and reliable and have to be secure with cryptographic algorithms and security procedures.
   It involves different approaches to secure all the components of embedded systems from prototype to deployment.

Deployment challenges in IoT :

1. **Connectivity** –
   It is the foremost concern while connecting devices, applications and cloud platforms. Connected devices that provide useful front and information are extremely valuable. But poor connectivity becomes a challenge where IoT sensors are required to monitor process data and supply information.
2. **Cross platform capability** –
   IoT applications must be developed, keeping in mind the technological changes of the future.
   Its development requires a balance of hardware and software functions.
   It is a challenge for IoT application developers to ensure that the device and IoT platform drivers the best performance despite heavy device rates and fixings.
3. **Data collection and processing** –
   In IoT development, data plays an important role. What is more critical here is the processing or usefulness of stored data.
   Along with security and privacy, development teams need to ensure that they plan well for the way data is collected, stored or processed within an environment.
4. **Lack of skill set** –
   All of the development challenges above can only be handled if there is a proper skilled resource working on the IoT application development.
   The right talent will always get you past the major challenges and will be an important IoT application development asset.

Challenges Faced By IoT in Agricultural Sector

- Last Updated : 11 Dec, 2019

By adopting IoT in the agricultural sector we get numerous benefits, but still, there are challenges faced by IoT in agricultural sectors. The biggest challenges faced by IoT in the agricultural sector are lack of information, high adoption costs, and security concerns, etc. Most of the farmers are not aware of the implementation of IoT in agriculture. Major problem is that some of them are opposed to new ideas and they do not want to adopt even if it provides numerous benefits. The best thing that can be done to raise awareness of IoT's impact is to demonstrate farmers the use of IoT devices like drones, sensors and other technologies and they could provide them ease at work and accompanied by real-world examples.

Challenges Faced by Farmers in adopting IoT for Agriculture

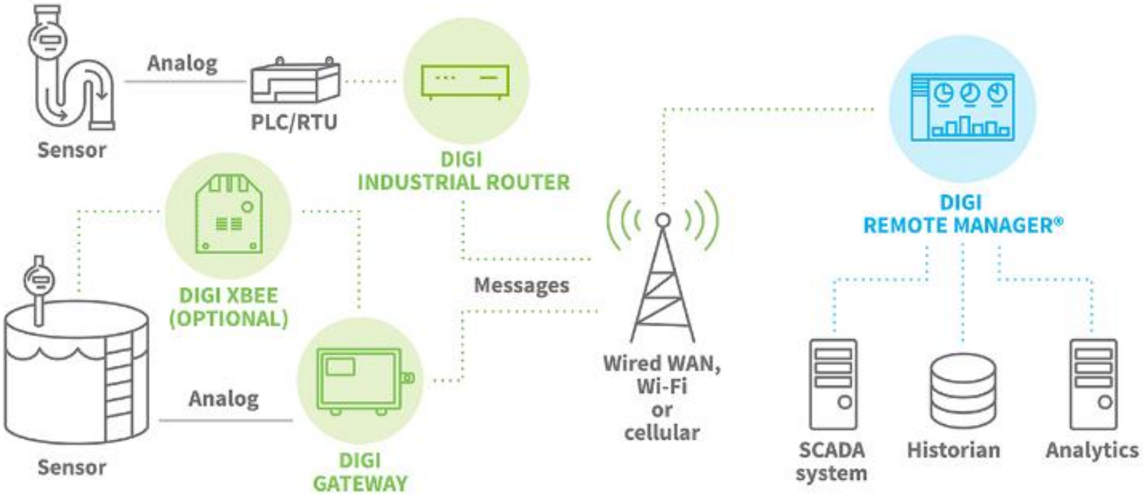
1. Lack of Infrastructure: Even if the farmers adopt IoT technology they won't be able to take benefit of this technology due to poor communication infrastructure. Farms are located in remote areas and are far from access to the internet. A farmer needs to have access to crop data reliably at any time from any location, so connection issues would cause an advanced monitoring system to be useless.

2. High Cost: Equipment needed to implement IoT in agriculture is expensive. However sensors are the least expensive component, yet outfitting all of the farmers' fields to be with them would cost more than a thousand dollars. Automated machinery cost more than manually operated machinery as they include cost for farm management software and cloud access to record data. To earn higher profits, it is significant for farmers to invest in these technologies however it would be difficult for them to make the initial investment to set up IoT technology at their farms.

3. Lack of Security: Since IoT devices interact with older equipment they have access to the internet connection, there is no guarantee that they would be able to access drone mapping data or sensor readouts by taking benefit of public connection. An enormous amount of data is collected by IoT agricultural systems which is difficult to protect. Someone can have unauthorized access IoT providers database and could steal and manipulate the data

Structure of IoT

What makes an automated device like a motion-activated light switch different from an IoT-connected device that does the same thing? In a word: data. With an IoT-connected device, when a sensor detects motion and an actuator turns on a light, those actions are captured as data and sent to the cloud or a data center for recording and analysis. And where there's data, there needs to be an IoT architecture that tells the data where to go, what format to use, how to get there and what actions to take based upon this data.



IoT system architecture is often described as a four-stage process in which data flows from sensors attached to "things" through a network and eventually on to a corporate data center or the cloud for processing, analysis and storage.

In the Internet of Things, a "thing" could be a machine, a building or even a person. Processes in the IoT architecture also send data in the other direction in the form of instructions or commands that tell an actuator or other physically connected device to take some action to control a physical process. An actuator could do something as simple as turning on a light or as consequential as shutting down an assembly line if impending failure is detected.

STAGE 1: Sensors and Actuators

The process starts with sensors and actuators, the connected devices that monitor (in the case of sensors) or control (in the case of actuators) some "thing" or physical process. Sensors capture data regarding the status of a process or an environmental condition, such as temperature, humidity, chemical composition, fluid levels in a tank, fluid flow in a pipe, or the speed of an assembly line as well as much more.

In some cases, a sensor might detect a condition or event that requires an almost immediate response so that an actuator can perform remediation actions in real time, for example, adjusting the flow rate of a fluid, or the movements of an industrial robot. In these situations, very low latency between the sensor and analysis/triggered actuator is required. To avoid the delay of a round-trip of data to the server, analysis of data to determine failure and sending of control to the "thing", this critical processing is performed in close proximity to the process being monitored or controlled. This "edge" processing can be performed by a system on module (SOM) device

STAGE 2: Internet Gateways and Data Acquisition Systems

A data acquisition system (DAS) collects raw data from the sensors and converts it from analog into digital format. The DAS then aggregates and formats the data before sending it through an Internet gateway via wireless WANs (such as Wi-Fi or Cellular) or wired WANs for the next stage of processing.

At this point, the volume of data is at its maximum. The quantities can be huge, especially, for example, in a factory setting where hundreds of sensors may be gathering data simultaneously. For that reason, the data is also filtered and compressed to an optimum size for transmission.

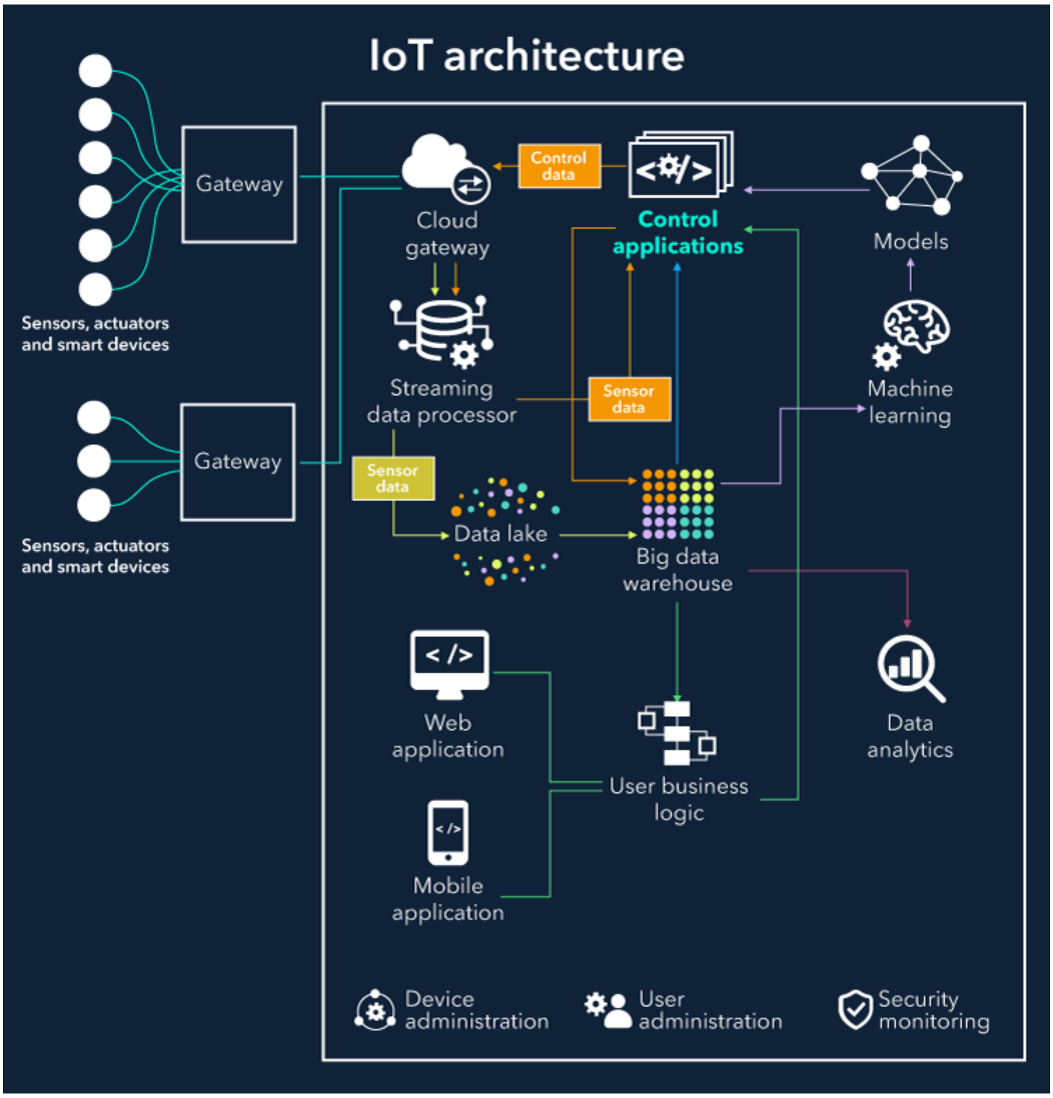
STAGE 3: Pre-processing: Analytics at the Edge

Once the IoT data has been digitized and aggregated, it will need processing to further reduce the data volume before it goes to the data center or cloud. The edge device may perform some analytics as part of the pre-processing. Machine learning can be very helpful at this stage to provide feedback into the system and improve the process on an ongoing basis, without waiting for instructions to come back from the corporate data center or cloud. Processing of this type will generally take place on a device in a location close to where the sensors reside, such as in an on-site wiring closet.

STAGE 4: In-depth Analysis in the Cloud or Data Center

At Stage 4 in the process, powerful IT systems can be brought to bear to analyze, manage, and securely store the data. This usually takes place in the corporate data center or in the cloud, where data from multiple field sites/sensors can be combined to provide a broader picture of the overall IoT system and deliver actionable insights to both IT and business managers. A company may have operations in different geographies and IoT data can be analyzed to identify key trends and patterns, or to spot anomalies.

At this level, industry-specific and/or company-specific applications can be used to perform in-depth analysis and apply business rules to determine whether action needs to be taken. The incoming data may indicate desirable changes to device settings or other ways to optimize the process, forming a loop that facilitates constant improvement. Stage 4 also includes storage in a data warehouse, both for record keeping and for further analysis.



IoT Map Device

A vast amount of data is created every day from sensors and devices: GPS devices on vehicles, objects, and people; sensors monitoring the environment; live video feeds; speed sensors in roadways; social media feeds; and more, all connected through the Internet. What this Internet of Things means is that we have an emerging source of valuable data. It's called "real-time" data. Only recently has the technology emerged to enable this real-time data to be incorporated into GIS applications.

The real-time GIS capabilities of the ArcGIS platform have transformed how information is utilized during any given situation. Real-time dashboards fed by the IOT provide actionable views into the daily operations of organizations, empowering decision-makers and stakeholders with the latest information they need to drive current and future ideas and strategies. Dashboards answer questions such as: What's happening right now? Where is it happening? Who is affected? What assets are available? Where are my people?

Some applications of real-time dashboards

- Local governments use real-time information to manage operations such as tracking and monitoring snowplows and trash trucks.
- Utilities monitor public services including water, wastewater, and electricity for consumers.
- Transportation departments track buses and trains and monitor traffic flows, road conditions, and incidents.
- Airport authorities and aviation agencies track and monitor air traffic worldwide.
- Oil and gas companies monitor equipment in the field, tanker cars, and field crews.
- Law enforcement agencies monitor crime as it happens, as well as incoming 911 calls.
- Companies use real-time social media feeds such as Twitter to gauge feedback and monitor social sentiment about particular issues.
- To issue early warnings and reports, federal agencies such as the Federal Emergency Management Agency (FEMA), US Geological Survey (USGS), National Oceanographic and Atmospheric Administration (NOAA), and Environmental Protection Agency (EPA) gather vast amounts of information about the environment. They monitor weather, air and water quality, floods, earthquakes, and wildfires.
- Individuals use elements of the IOT—smartphones, smartwatches, smart sensors, radio-frequency identifications (RFIDs), beacons, fitness bands, and so on—to capture and visualize information about every type of activity.

Emergency management agencies monitor public safety during large events, such as marathons and the Olympics.

Real-time data is as current as the data source that is updating it, whether that data is being updated every second, minute, hour, or daily. What is real time to one organization might not be real time to another, depending on the type of scenario being monitored.

Real time is a concept that typically refers to the awareness of events at the same rate or at the same time as they unfold (without significant delay). It's often confused with frequency, or the intervals between events, which is essentially how often the event is updated. The update interval, or frequency, relates to the term "temporal resolution," which can vary from one application to another.

For example, most aircraft monitoring systems provide two updates every second, whereas it may take every hour to provide a weather update. For monitoring their networks, energy utilities use systems, also known as SCADA (Supervisory Control and Data Acquisition), that sample data about voltage, flow, pressure, and more from analog devices at very high frequencies (e.g., 50 hertz). This can result in high resource requirements for network bandwidth, system memory, and storage volume.

The data that fueled geographic applications in the past was created to represent the state of something at a specific point in time: data captured for what has happened, or what is happening, or what will happen. Although this GIS data is valuable for countless GIS applications and analyses, today the current snapshot of what is happening now falls out of sync very quickly with the real world, in many cases becoming outdated almost as soon as it is created.

What is real-time GIS?

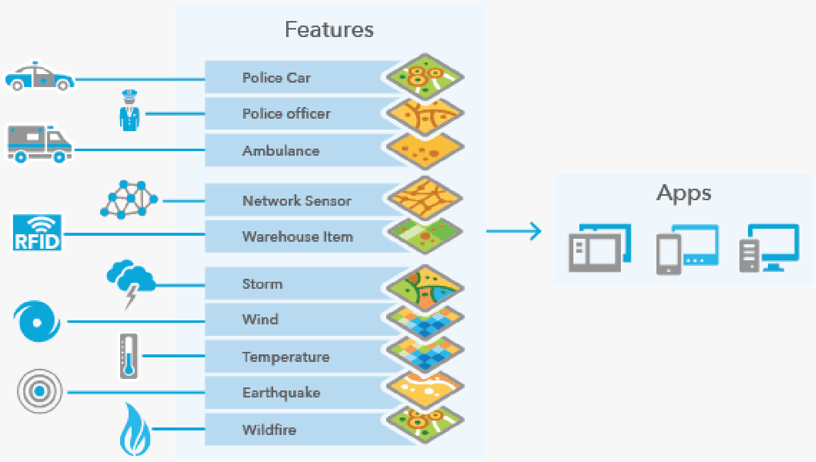
Real-time GIS can be characterized as a continuous stream of events flowing from IoT sensors or data feeds. Each event represents the latest measured state, including position, temperature, concentration, pressure, voltage, water level, altitude, speed, distance, and directional information flowing from a sensor.

Maps provide the most basic frameworks for viewing, monitoring, and responding to real-time data feeds.

Acquire real-time data

A utility organization may want to visually represent the live status of its network with information that is captured by sensors in the field. Although the sensors on the network are not physically moving, their status and the information they send changes rapidly. Radio-frequency identification (RFID) is being used in a wide variety of environments to keep track of items of interest. Warehouses and logistics companies use RFID to track and monitor inventory levels. Hospitals use it to track equipment to make sure it has gone through proper cleansing procedures before being used.

A wide range of real-time data is accessible today. Connectors exist for many common devices and sensors enabling easy integration between the IoT and your GIS.



Part-A

Q.No	Questions	Competence	BT Level
1.	What are real-time embedded systems?	Analyse	BTL 4
2.	What are hard and soft embedded systems?	Analyse	BTL 4
3.	What are the various classifications in embedded systems?	Understand	BTL 2
4.	What is an embedded system? What are the components of embedded system?	Understand	BTL 2
5.	What are the applications of an embedded system?	Analyse	BTL 4
6.	Give some examples for small scale embedded systems.	Analyse	BTL 4
7.	Give some examples for medium scale embedded systems	Analyse	BTL 4
8.	Give some examples for sophisticated embedded systems	Analyse	BTL 4
9.	What are the requirements of embedded system?	Understand	BTL 2
10.	Give the steps in embedded system design?	Understand	BTL 2
11.	What are the challenges of embedded systems?	Analyse	BTL 4
12.	What are the three processor series in ARM cortex family?	Remember	BTL 1
13.	What is the nomenclature for ARMTDMI processor?	Understand	BTL 2
14.	What is Thumb instructioin set and why is it needed in ARM?	Analyse	BTL 4
15.	What are the seven modes of operation in ARM processor?	Remember	BTL 1
16.	Narrate the 3 stage pipeline architecture in ARM processor?	Understand	BTL 2

Part-B

Q.No	Questions	Competence	BT Level
1.	Analyze the various application areas of IoT.	Analysis	BTL4
2.	Categorise the various types of sensors used in IoT.	Analysis	BTL4
3.	Discuss the challenges faced in the implementation of Internet of Things?	Analysis	BTL4

4.	Discuss on the recent trends in Embedded system.	Analysis	BTL4
5.	Categorise the Characteristics of Embedded system ?	Analysis	BTL4

UNIT II - EMBEDDED IoT PLATFORM DESIGN METHODOLOGY

IoT Design Methodology

Figure 5.1 shows the steps involved in the IoT system design methodology. Each of these steps is explained in the sections that follow. To explain these steps, we use the example of a smart IoT-based home automation system.

Step 1: Purpose & Requirements Specification

The first step in IoT system design methodology is to define the purpose and requirements of the system. In this step, the system purpose, behavior and requirements (such as data collection requirements, data analysis requirements, system management requirements, data privacy and security requirements, user interface requirements, ...) are captured.

Applying this to our example of a smart home automation system, the purpose and requirements for the system may be described as follows:

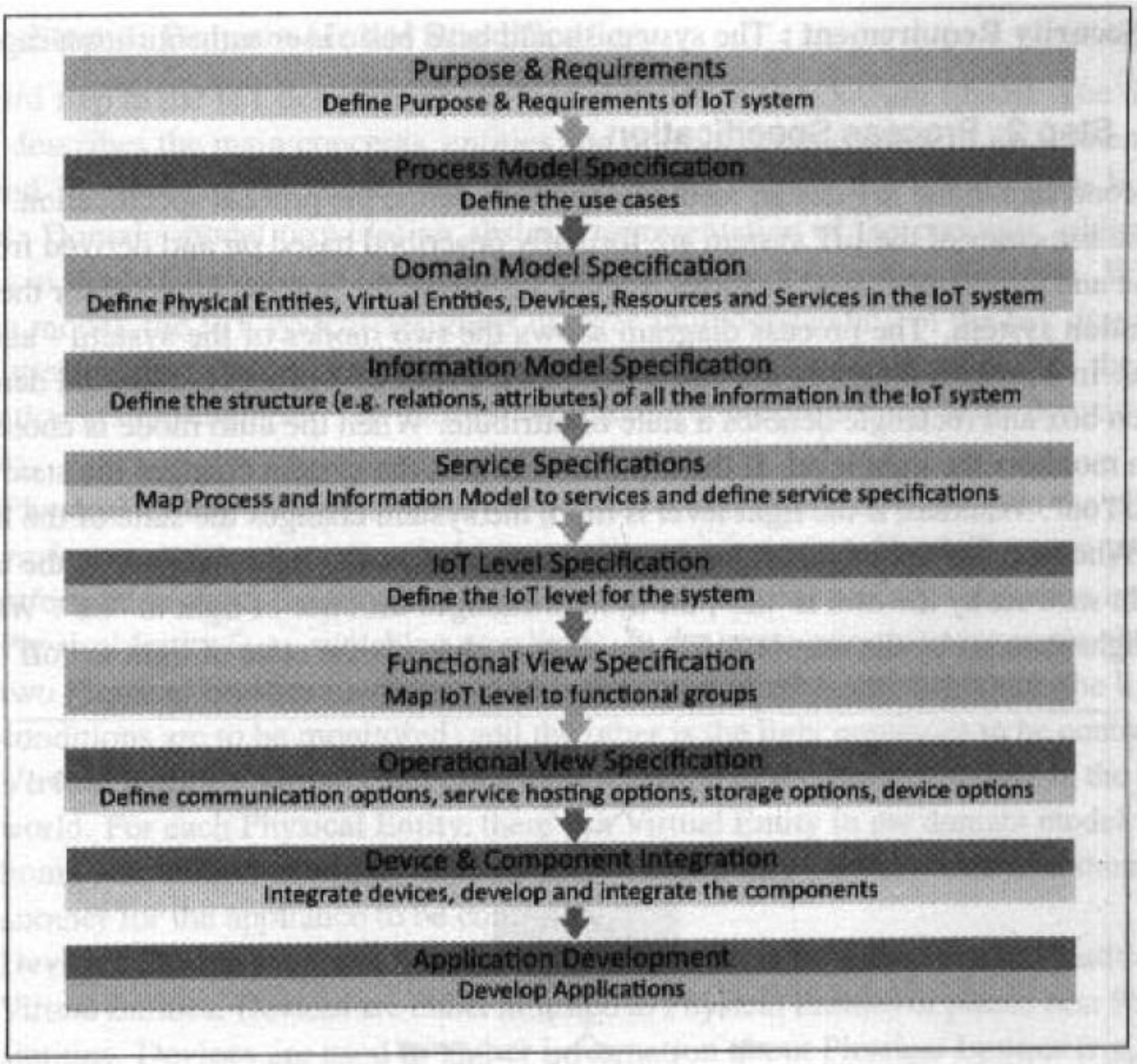

Purpose & Requirements
Define Purpose & Requirements of IoT system
Process Model Specification
Define the use cases
Domain Model Specification
Define Physical Entities, Virtual Entities, Devices, Resources and Services in the IoT system
Information Model Specification
Define the structure (e.g. relations, attributes) of all the information in the IoT system
Service Specifications
Map Process and Information Model to services and define service specifications
IoT Level Specification
Define the IoT level for the system
Functional View Specification
Map IoT Level to functional groups
Operational View Specification
Define communication options, service hosting options, storage options, device options
Device & Component Integration
Integrate devices, develop and integrate the components
Application Development
Develop Applications

Fig. Steps involved in IoT system design methodology

- **Purpose :** A home automation system that allows controlling of the lights in a home remotely using a web application.
- **Behavior :** The home automation system should have auto and manual modes. In auto mode, the system measures the light level in the room and switches on the light when it gets dark. In manual mode, the system provides the option of manually and remotely switching on/off the light.
- **System Management Requirement :** The system should provide remote monitoring and control functions.
- **Data Analysis Requirement :** The system should perform local analysis of the data.
- **Application Deployment Requirement :** The application should be deployed locally on the device, but should be accessible remotely.
- **Security Requirement :** The system should have basic user authentication capability.

Step 2: Process Specification

The second step in the IoT design methodology is to define the process specification. In this step, the use cases of the IoT system are formally described based on and derived from the purpose and requirement specifications. Figure 5.2 shows the process diagram for the home automation system. The process diagram shows the two modes of the system - auto and manual. In a process diagram, the circle denotes the start of a process, diamond denotes a decision box and rectangle denotes a state or attribute. When the auto mode is chosen, the system monitors the light level. If the light level is low, the system changes the state of the light to "on". Whereas, if the light level is high, the system changes the state of the light to "off". When the manual mode is chosen, the system checks the light state set by the user. If the light state set by the user is "on", the system changes the state of light to "on". Whereas, if the light state set by the user is "off", the system changes the state of light to "off".

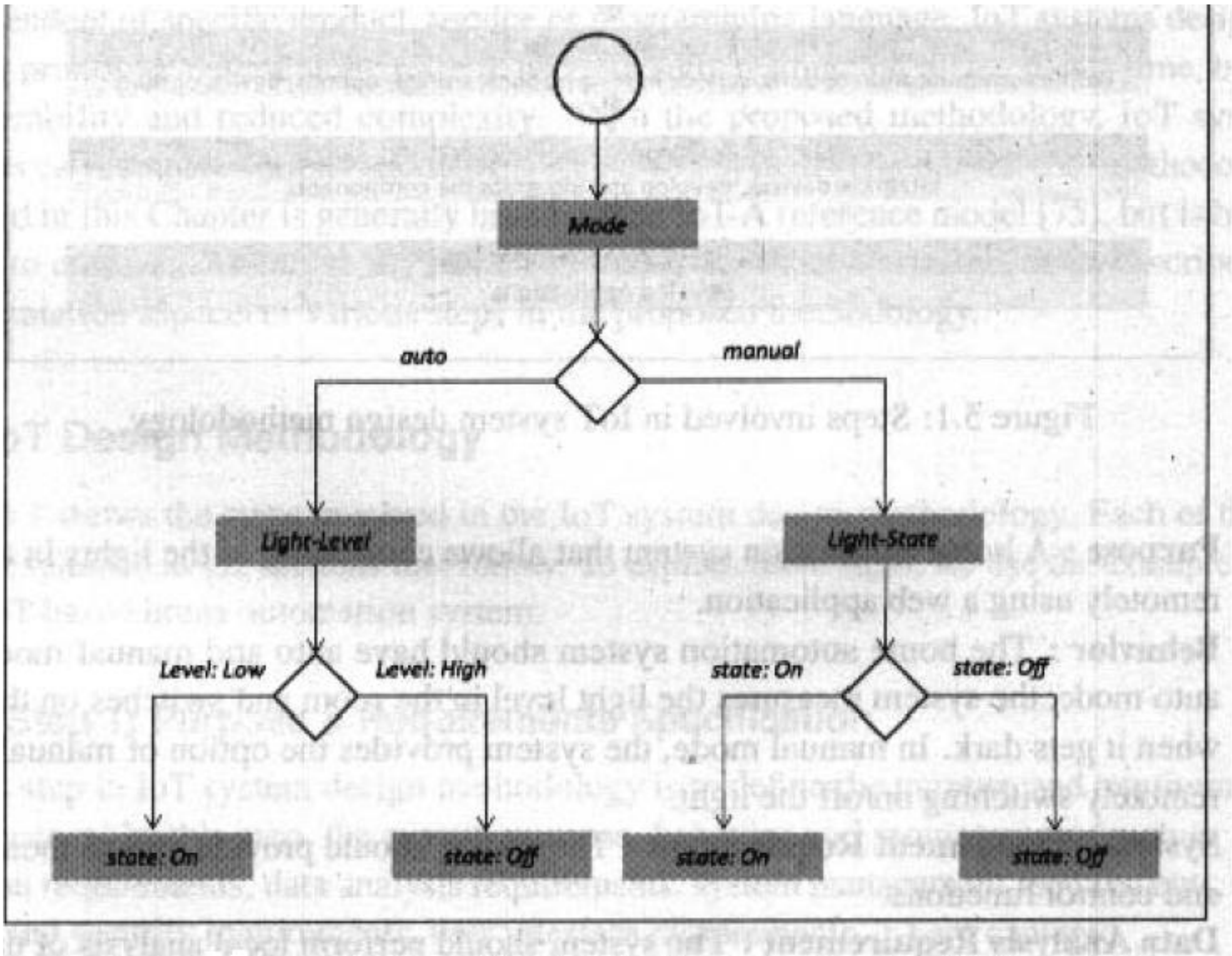


Figure Process specification for home automation IoT system

Step 3: Domain Model Specification

The third step in the IoT design methodology is to define the Domain Model. The domain model describes the main concepts, entities and objects in the domain of IoT system to be designed. Domain model defines the attributes of the objects and relationships between objects. Domain model provides an abstract representation of the concepts, objects and entities in the IoT domain, independent of any specific technology or platform. With the domain model, the IoT system designers can get an understanding of the IoT domain for which the system is to be designed. Figure 5.3 shows the domain model for the home automation system example. The entities, objects and concepts defined in the domain model include:

- **Physical Entity :** Physical Entity is a discrete and identifiable entity in the physical environment (e.g. a room, a light, an appliance, a car, etc.). The IoT system provides information about the Physical Entity (using sensors) or performs actuation upon the Physical Entity (e.g., switching on a light). In the home automation example, there are two Physical Entities involved - one is the room in the home (of which the lighting conditions are to be monitored) and the other is the light appliance to be controlled.
- **Virtual Entity :** Virtual Entity is a representation of the Physical Entity in the digital world. For each Physical Entity, there is a Virtual Entity in the domain model. In the home automation example, there is one Virtual Entity for the room to be monitored, another for the appliance to be controlled.
- **Device :** Device provides a medium for interactions between Physical Entities and Virtual Entities. Devices are either attached to Physical Entities or placed near Physical Entities. Devices are used to gather information about Physical Entities (e.g., from sensors), perform actuation upon Physical Entities (e.g. using actuators) or used to identify Physical Entities (e.g., using tags). In the home automation example, the device is a single-board mini computer which has light sensor and actuator (relay switch) attached to it.
- **Resource :** Resources are software components which can be either "on-device" or "network-resources". On-device resources are hosted on the device and include software components that either provide information on or enable actuation upon the Physical Entity to which the device is attached. Network resources include the software components that are available in network (such as a database). In the home automation example, the on-device resource is the operating system that runs on the single-board mini computer.
- **Service :** Services provide an interface for interacting with the Physical Entity. Services access the resources hosted on the device or the network resources to obtain information about the Physical Entity or perform actuation upon the Physical Entity.

In the home automation example, there are three services: (1) a service that sets mode to auto or manual, or retrieves the current mode; (2) a service that sets the light appliance state to on/off, or retrieves the current light state; and (3) a controller service that runs as a native service on the device. When in auto mode, the controller service monitors the light level and switches the light on/off and updates the status in the status database. When in manual mode, the controller service retrieves the current state from the database and switches the light on/off. The process of deriving the services from the process specification and information model is described in the later sections.

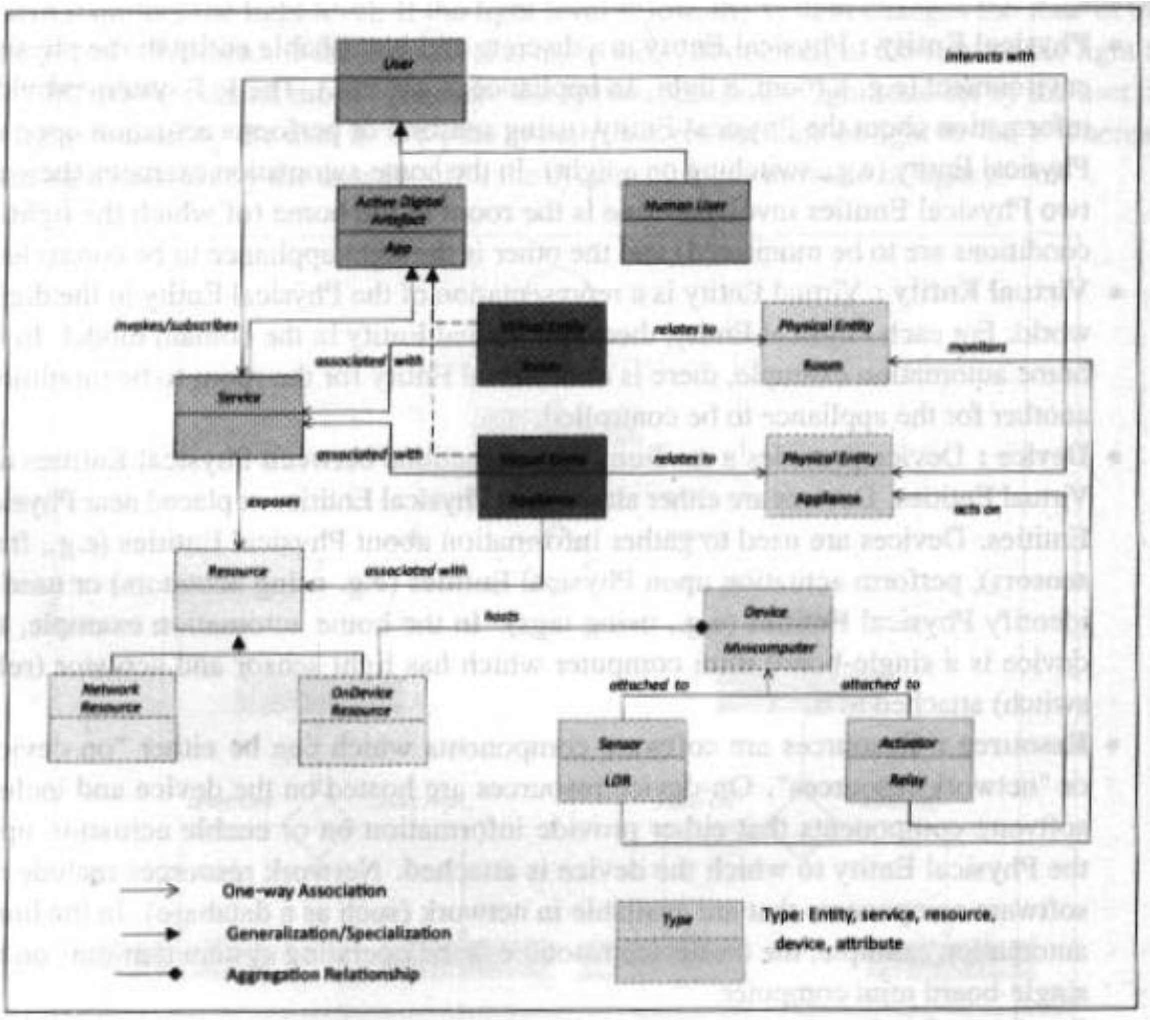


Figure Domain model of the home automation IoT system

Step 4: Information Model Specification

The fourth step in the IoT design methodology is to define the Information Model. Information Model defines the structure of all the information in the IoT system, for example, attributes of Virtual Entities, relations, etc. Information model does not describe the specifics of how the information is represented or stored. To define the information model, we first list the Virtual Entities defined in the Domain Model. Information model adds more details to the Virtual Entities by defining their attributes and relations. In the home automation example, there are two Virtual Entities - a Virtual Entity for the light appliance (with attribute - light state) and a Virtual Entity for the room (with attribute - light level). Figure 5.4 shows the Information Model for the home automation system example.

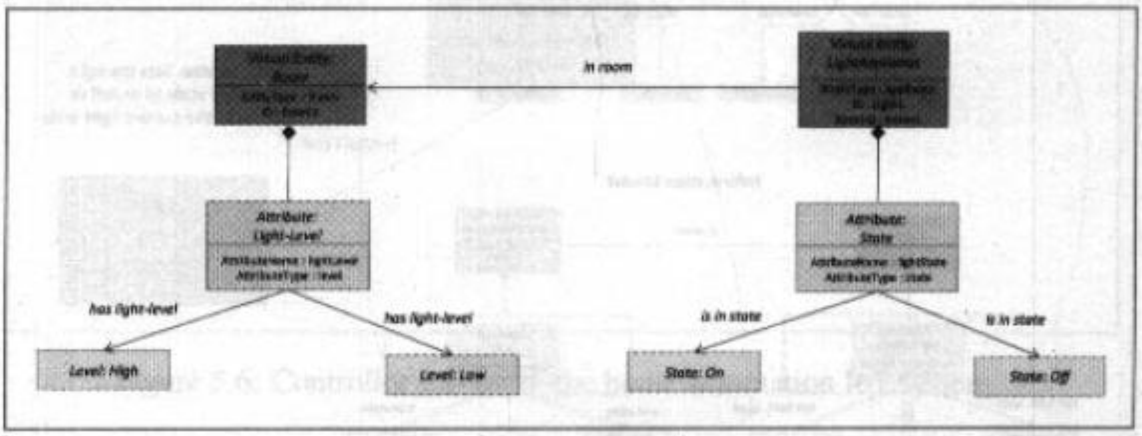
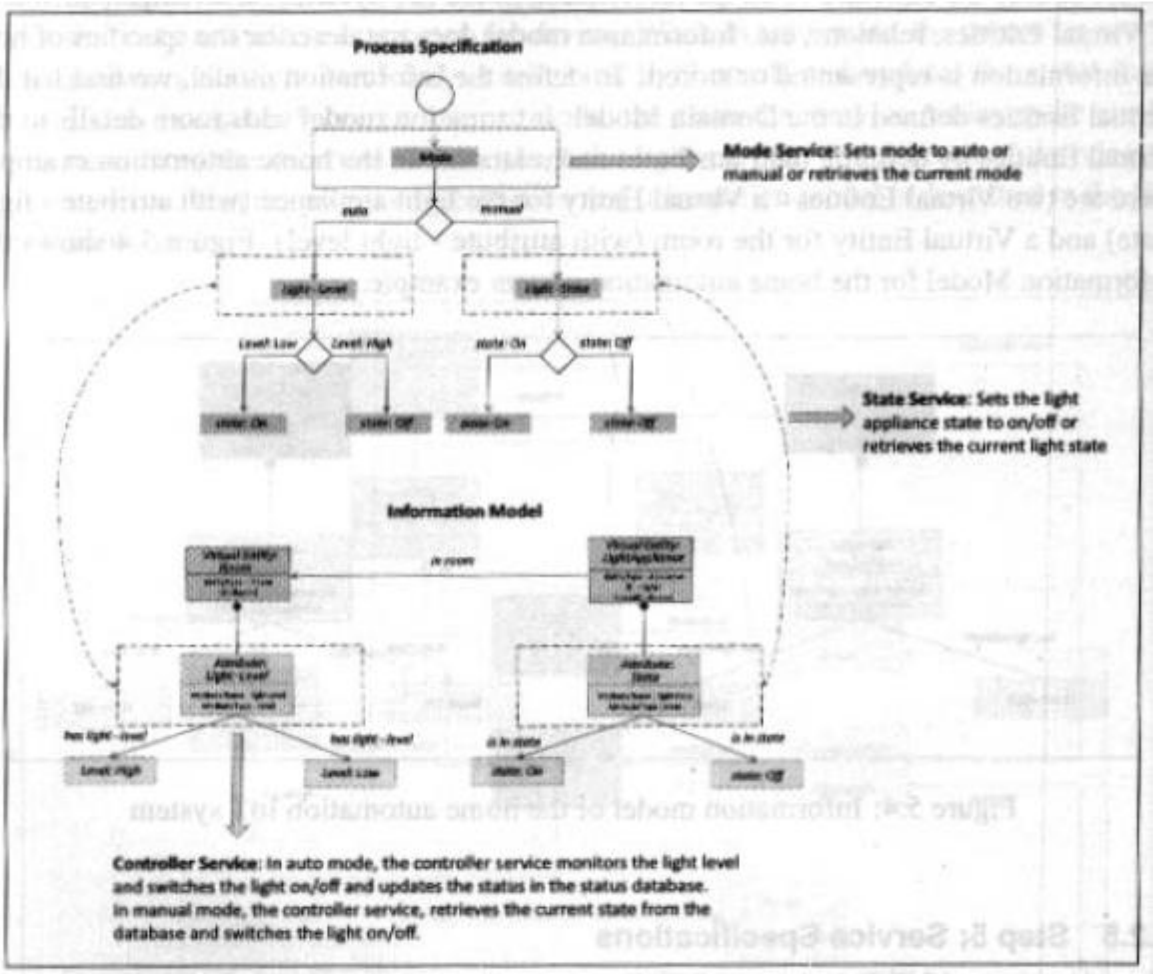


Figure Information model of the home automation IoT system

Step 5: Service Specifications

The fifth step in the IoT design methodology is to define the service specifications. Service specifications define the services in the IoT system, service types, service inputs/output, service endpoints, service schedules, service preconditions and service effects.

You learned about the Process Specification and Information Model in the previous sections. Figure 5.5 shows an example of deriving the services from the process specification and information model for the home automation IoT system. From the process specification and information model, we identify the states and attributes. For each state and attribute we define a service. These services either change the state or attribute values or retrieve the current values. For example, the Mode service sets mode to auto or manual or retrieves the current mode. The State service sets the light appliance state to on/off or retrieves the current light state. The Controller service monitors the light level in auto mode and switches the light on/off and updates the status in the status database. In manual mode, the controller service, retrieves the current state from the database and switches the light on/off.



Mode Service: Sets mode to auto or manual or retrieves the current mode

State Service: Sets the light appliance state to on/off or retrieves the current light state

Controller Service: In auto mode, the controller service monitors the light level and switches the light on/off and updates the status in the status database. In manual mode, the controller service, retrieves the current state from the database and switches the light on/off.

Figure Deriving services from process specification and information model for home automation IoT system

Figures 5.6, 5.7 and 5.8 show specifications of the controller, mode and state services of the home automation system. The Mode service is a RESTful web service that sets mode to auto or manual (PUT request), or retrieves the current mode (GET request) . The mode is updated to/retrieved from the database. The State service is a RESTful web service that sets the light appliance state to on/off (PUT request), or retrieves the current light state (GET request). The state is updated to/retrieved from the status database. The Controller service runs as a native service on the device. When in auto mode, the controller service monitors the light level and switches the light on/off and updates the status in the status database. When in manual mode, the controller service, retrieves the current state from the database and switches the light on/off.

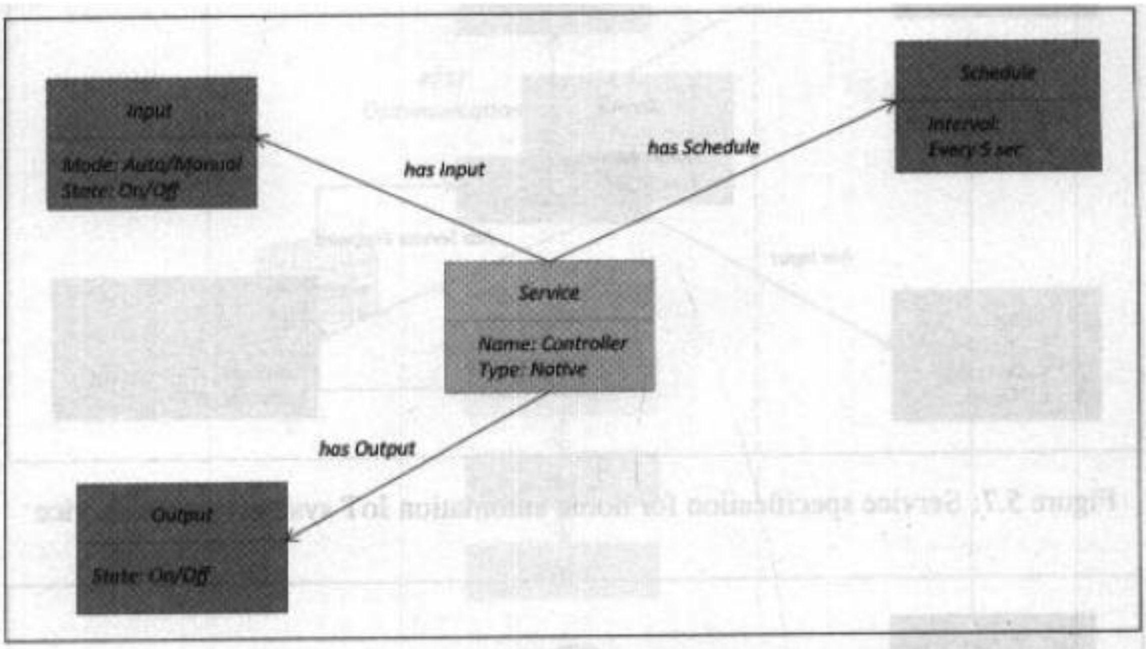


Figure : Controller service of the home automation IoT system

Step 6: IoT Level Specification

The sixth step in the IoT design methodology is to define the IoT level for the system. In Chapter-1, we defined five IoT deployment levels. Figure 5.9 shows the deployment level of the home automation IoT system, which is level-1.

Step 7: Functional View Specification

The seventh step in the IoT design methodology is to define the Functional View. The Functional View (FV) defines the functions of the IoT systems grouped into various Functional Groups (FGs). Each Functional Group either provides functionalities for interacting with instances of concepts defined in the Domain Model or provides information related to these concepts.

The Functional Groups (FG) included in a Functional View include:

- **Device :** The device FG contains devices for monitoring and control. In the home automation example, the device FG includes a single board mini-computer, a light sensor and a relay switch (actuator).

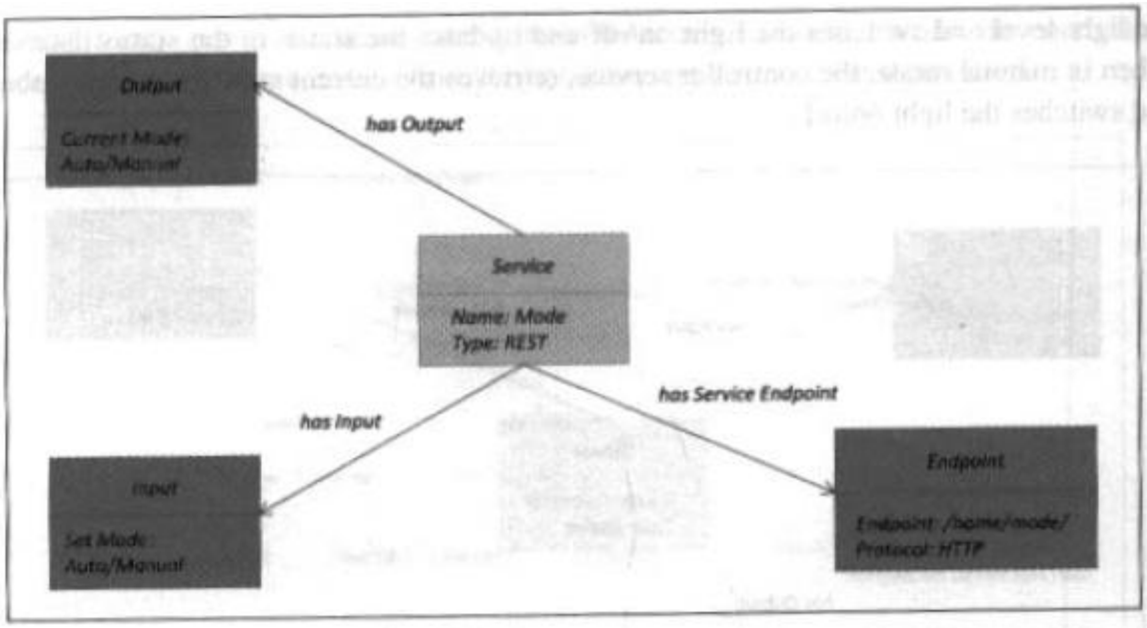


Figure 5.7: Service specification for home automation IoT system - mode service

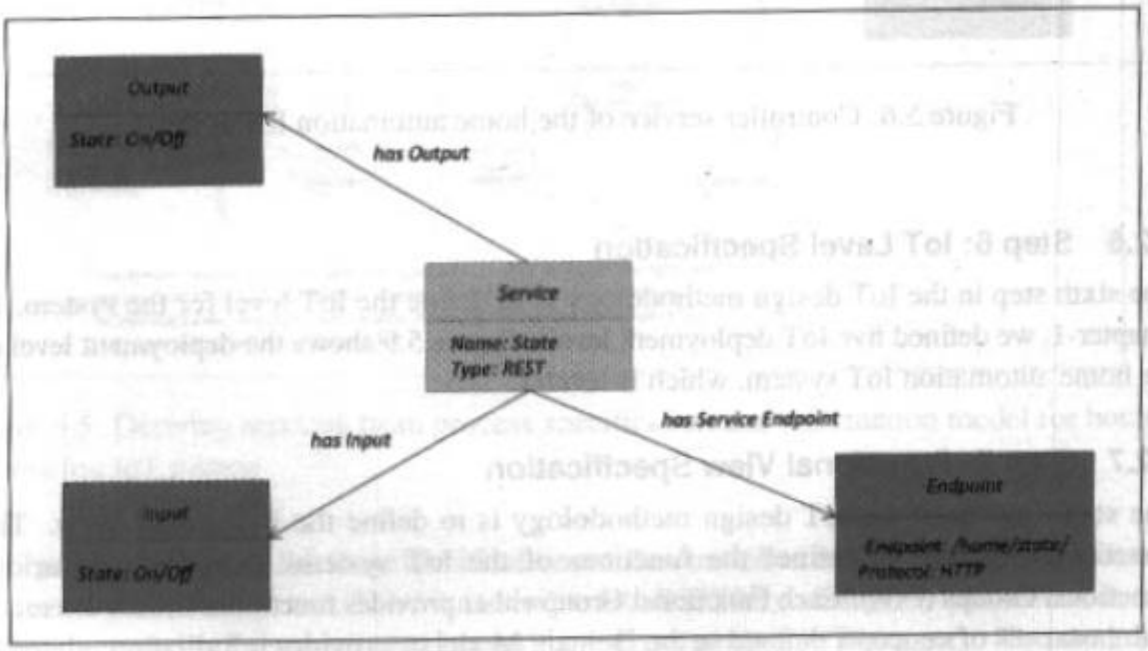


Figure 5.8: Service specification for home automation IoT system - state service

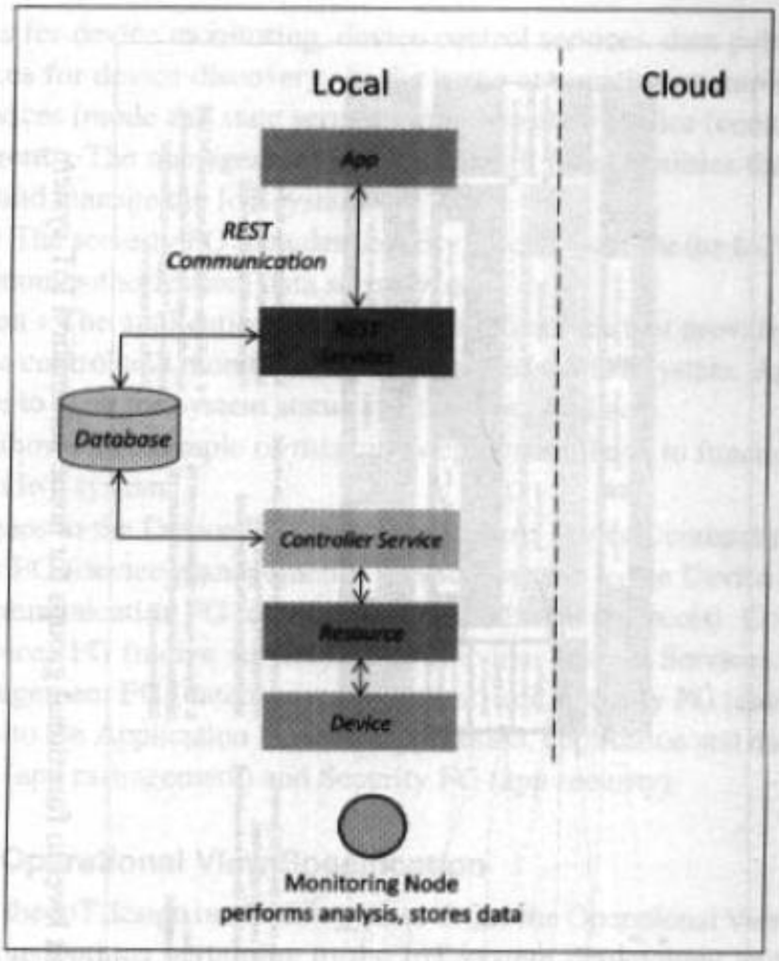
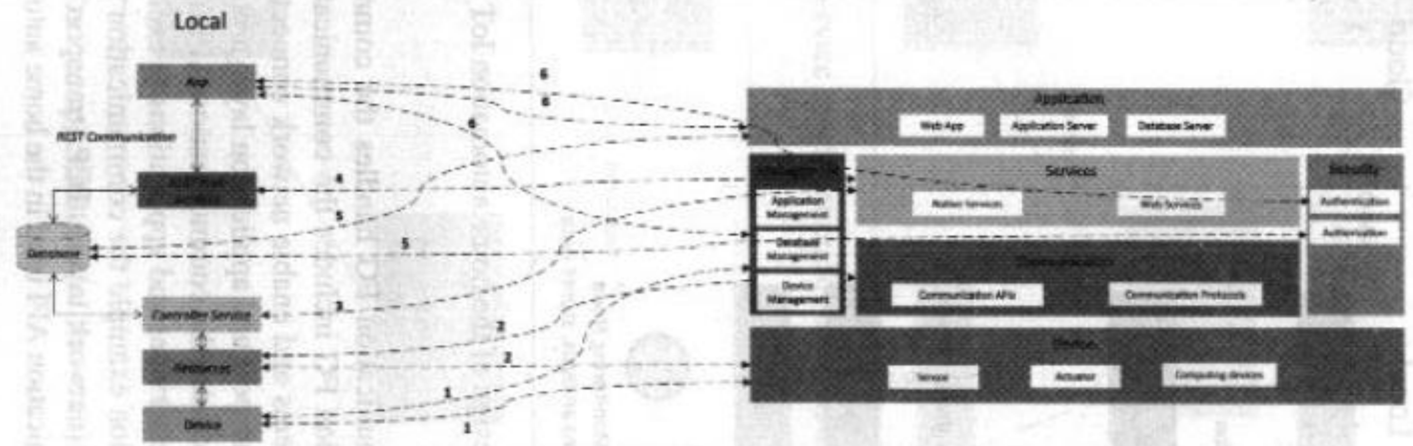


Figure 5.9: Deployment design of the home automation IoT system

- **Communication :** The communication FG handles the communication for the IoT system. The communication FG includes the communication protocols that form the backbone of IoT systems and enable network connectivity. You learned about various link, network, transport and application layer protocols in Chapter-1. The communication FG also includes the communication APIs (such as REST and WebSocket) that are used by the services and applications to exchange data over the network. In the home automation example the communication protocols include - 802.11 (link layer), IPv4/IPv6 (network layer), TCP (transport layer), and HTTP (application layer). The communication API used in the home automation examples is a REST-based API.
- **Services :** The service FG includes various services involved in the IoT system such



1. IoT device maps to the Device FG (sensors, actuators devices, computing devices) and the Management FG (device management)

2. Resources map to the Device FG (on-device resource) and Communication FG (communication APIs and protocols)

3. Controller service maps to the Services FG (native service). Web Services map to Services FG (web services)

4. Web Services map to Services FG (web services)

5. Database maps to the Management FG (database management) and Security FG (database security)

6. Application maps to the Application FG (web application, application and database servers), Management FG (app management) and Security FG (app security)

Figure 5.10: Mapping deployment level to functional groups for home automation IoT system

as services for device monitoring, device control services, data publishing services and services for device discovery. In the home automation example, there are two REST services (mode and state service) and one native service (controller service).

- **Management :** The management FG includes all functionalities that are needed to configure and manage the IoT system.
- **Security :** The security FG includes security mechanisms for the IoT system such as authentication, authorization, data security, etc.
- **Application :** The application FG includes applications that provide an interface to the users to control and monitor various aspects of the IoT system. Applications also allow users to view the system status and the processed data.

Figure 5.10 shows an example of mapping deployment level to functional groups for home automation IoT system.

IoT device maps to the Device FG (sensors, actuators devices, computing devices) and the Management FG (device management). Resources map to the Device FG (on-device resource) and Communication FG (communication APIs and protocols). Controller service maps to the Services FG (native service). Web Services map to Services FG . Database maps to the Management FG (database management) and Security FG (database security). Application maps to the Application FG (web application, application and database servers), Management FG (app management) and Security FG (app security).

Step 8: Operational View Specification

The eighth step in the IoT design methodology is to define the Operational View Specifications. In this step, various options pertaining to the IoT system deployment and operation are defined, such as, service hosting options, storage options, device options, application hosting options, etc.

Figure 5.11 shows an example of mapping functional groups to operational view specifications for home automation IoT system.

Operational View specifications for the home automation example are as follows:

- Devices: Computing device (Raspberry Pi), light dependent resistor (sensor), relay switch (actuator).
- Communication APIs: REST APIs
- Communication Protocols: Link Layer - 802.11, Network Layer - IPv4/IPv6, Transport - TCP, Application - HTTP.
- Services:
  1. Controller Service - Hosted on device, implemented in Python and run as a native service.
  2. Mode service - REST-ful web service, hosted on device, implemented with

Django-REST Framework.

3. State service - REST-ful web service, hosted on device, implemented with Django-REST Framework.

- Application:
  Web Application - Django Web Application,
  Application Server - Django App Server,
  Database Server - MySQL.
- Security:
  Authentication: Web App, Database
  Authorization: Web App, Database
- Management:
  Application Management - Django App Management
  Database Management - MySQL DB Management,
  Device Management - Raspberry Pi device Management.

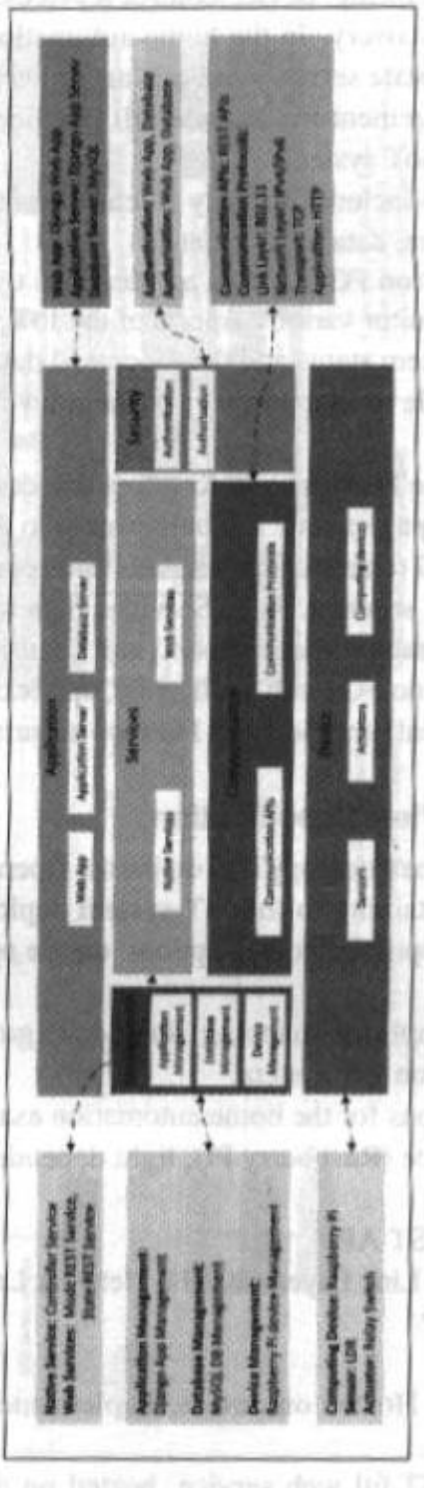


Figure 5.11: Mapping functional groups to operational view for home automation IoT system

Step 9: Device & Component Integration

The ninth step in the IoT design methodology is the integration of the devices and components. Figure 5.12 shows a schematic diagram of the home automation IoT system. The devices and components used in this example are Raspberry Pi mini computer, LDR sensor and relay switch actuator. A detailed description of Raspberry Pi board and how to interface sensors and actuators with the board is provided in later chapters.

Step 10: Application Development

The final step in the IoT design methodology is to develop the IoT application. Figure 5.13 shows a screenshot of the home automation web application. The application has controls for the mode (auto on or auto off) and the light (on or off). In the auto mode, the IoT system controls the light appliance automatically based on the lighting conditions in the room. When auto mode is enabled the light control in the application is disabled and it reflects the current state of the light. When the auto mode is disabled, the light control is enabled and it is used for manually controlling the light.

Case Study on IoT System for Weather Monitoring

In this section we present a case study on design of an IoT system for weather monitoring using the IoT design methodology. The purpose of the weather monitoring system is to collect data on environmental conditions such as temperature, pressure, humidity and light in an area using multiple end nodes. The end nodes send the data to the cloud where the data is aggregated and analyzed.

Figure 5.14 shows the process specification for the weather monitoring system. The process specification shows that the sensors are read after fixed intervals and the sensor measurements are stored.

Figure 5.15 shows the domain model for the weather monitoring system. In this domain model the physical entity is the environment which is being monitored. There is a virtual entity for the environment. Devices include temperature sensor, pressure sensor, humidity sensor, light sensor and single-board mini computer. Resources are software components which can be either on-device or network-resources. Services include the controller service that monitors the temperature, pressure, humidity and light and sends the readings to the cloud.

Figure 5.16 shows the information model for the weather monitoring system. In this example, there is one virtual entity for the environment being sensed. The virtual entity has attributes - temperature, pressure, humidity and light. Figure 5.17 shows an example of deriving the services from the process specification and information model for the weather monitoring system.

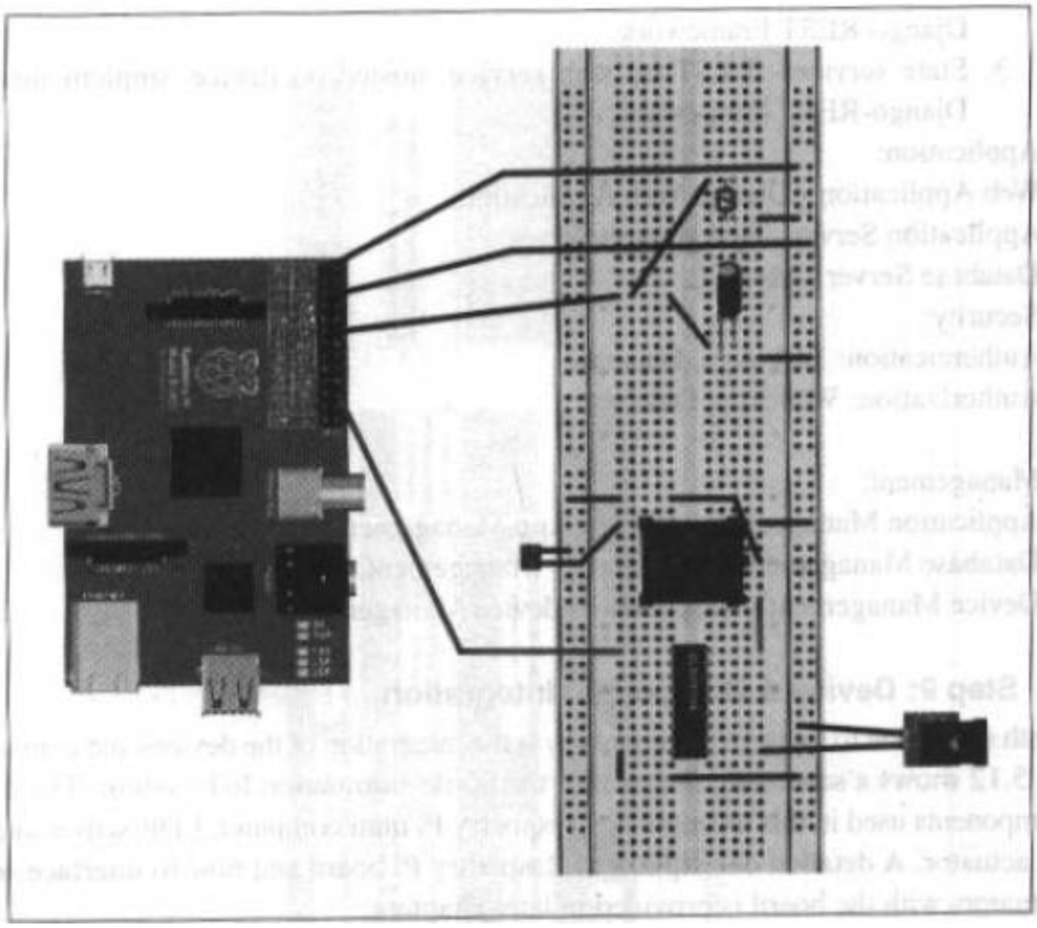


Figure 5.12: Schematic diagram of the home automation IoT system showing the device, sensor and actuator integrated

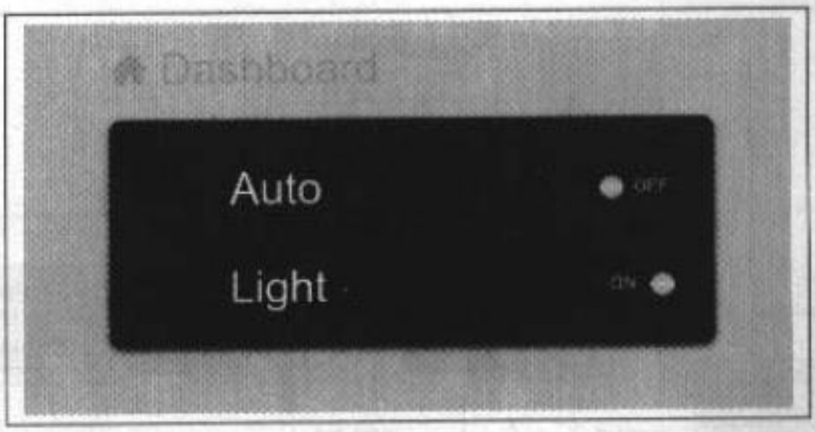


Figure 5.13: Home automation web application screenshot

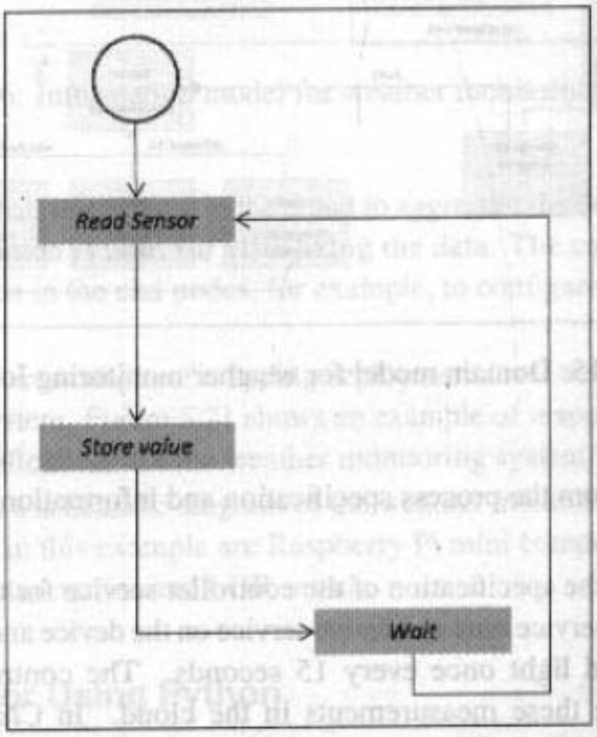


Figure 5.14: Process specification for weather monitoring IoT system

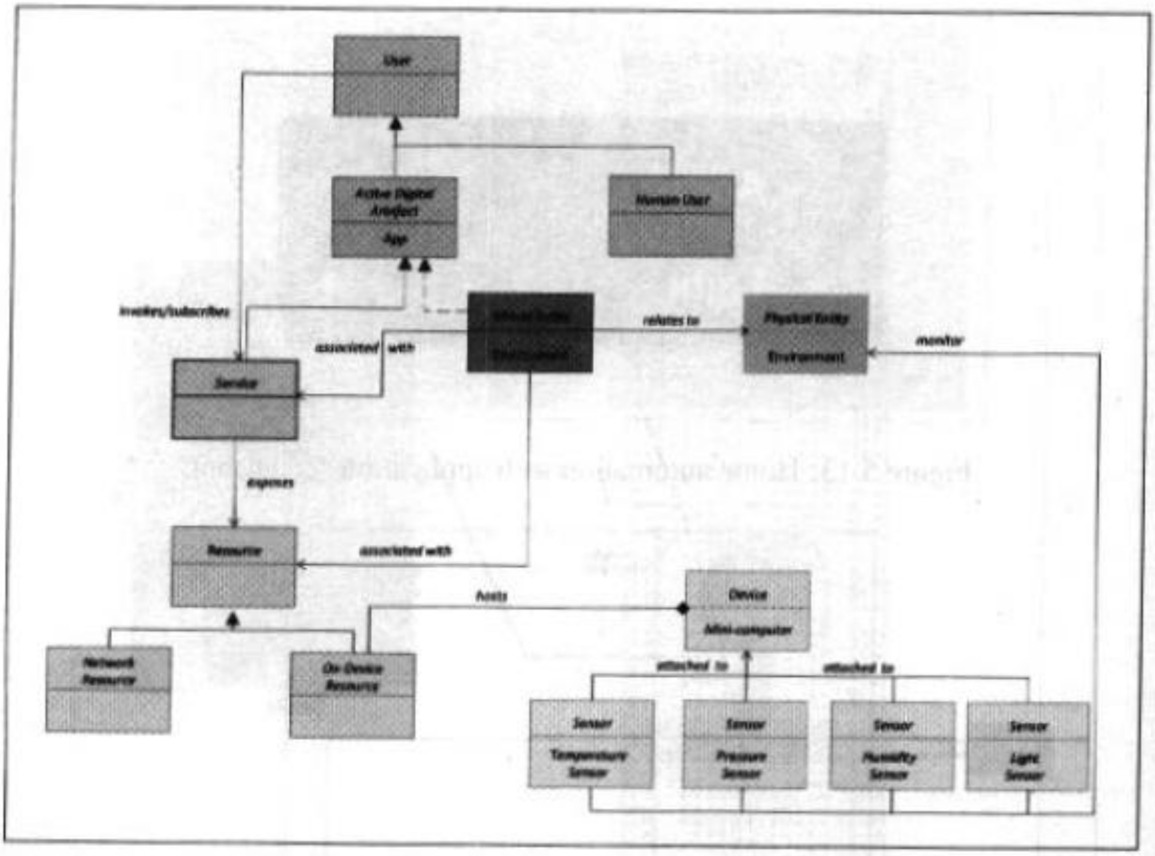


Figure 5.15: Domain model for weather monitoring IoT system

Figure 5.18 shows the specification of the controller service for the weather monitoring system. The controller service runs as a native service on the device and monitors temperature, pressure, humidity and light once every 15 seconds. The controller service calls the REST service to store these measurements in the cloud. In Chapter-8 we describe a Platform-as-a-Service called Xively that can be used for creating solutions for Internet of Things. An implementation of a controller service that calls the Xively REST API to store data in Xively cloud is described in Chapter-9.

Figure 5.19 shows the deployment design for the system. The system consists of multiple nodes placed in different locations for monitoring temperature, humidity and pressure in an area. The end nodes are equipped with various sensors (such as temperature, pressure, humidity and light). The end nodes send the data to the cloud and the data is stored in a cloud database. The analysis of data is done in the cloud to aggregate the data and make predictions. A cloud-based application is used for visualizing the data. The centralized controller can send control commands to the end nodes, for example, to configure the monitoring interval on the end nodes.

Figure 5.20 shows an example of mapping deployment level to functional groups for the weather monitoring system. Figure 5.21 shows an example of mapping functional groups to operational view specifications for the weather monitoring system.

Figure 5.22 shows a schematic diagram of the weather monitoring system. The devices and components used in this example are Raspberry Pi mini computer, temperature sensor, humidity sensor, pressure sensor and LDR sensor.

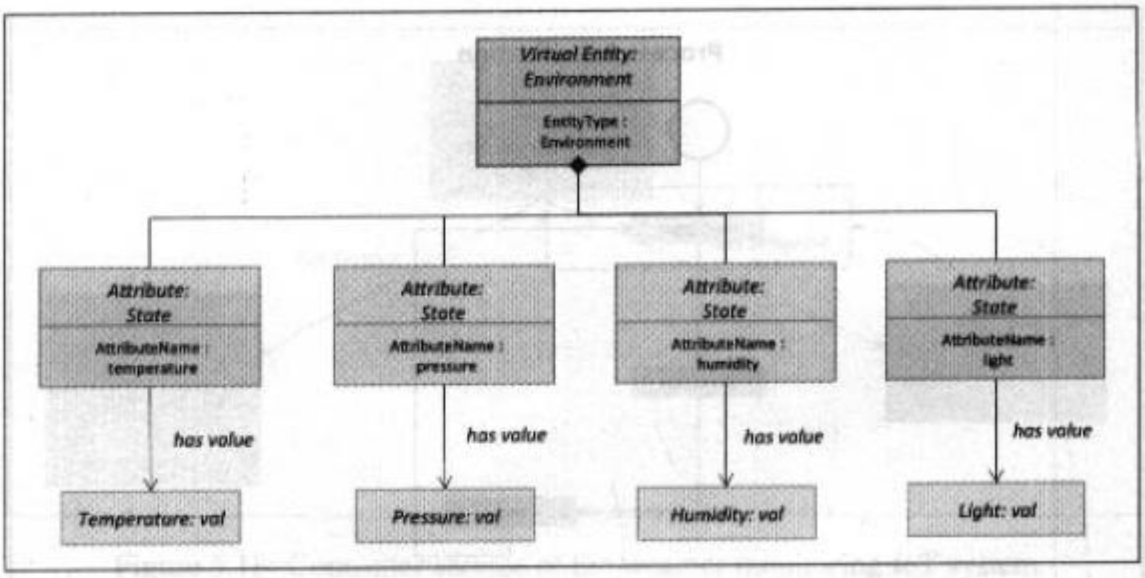
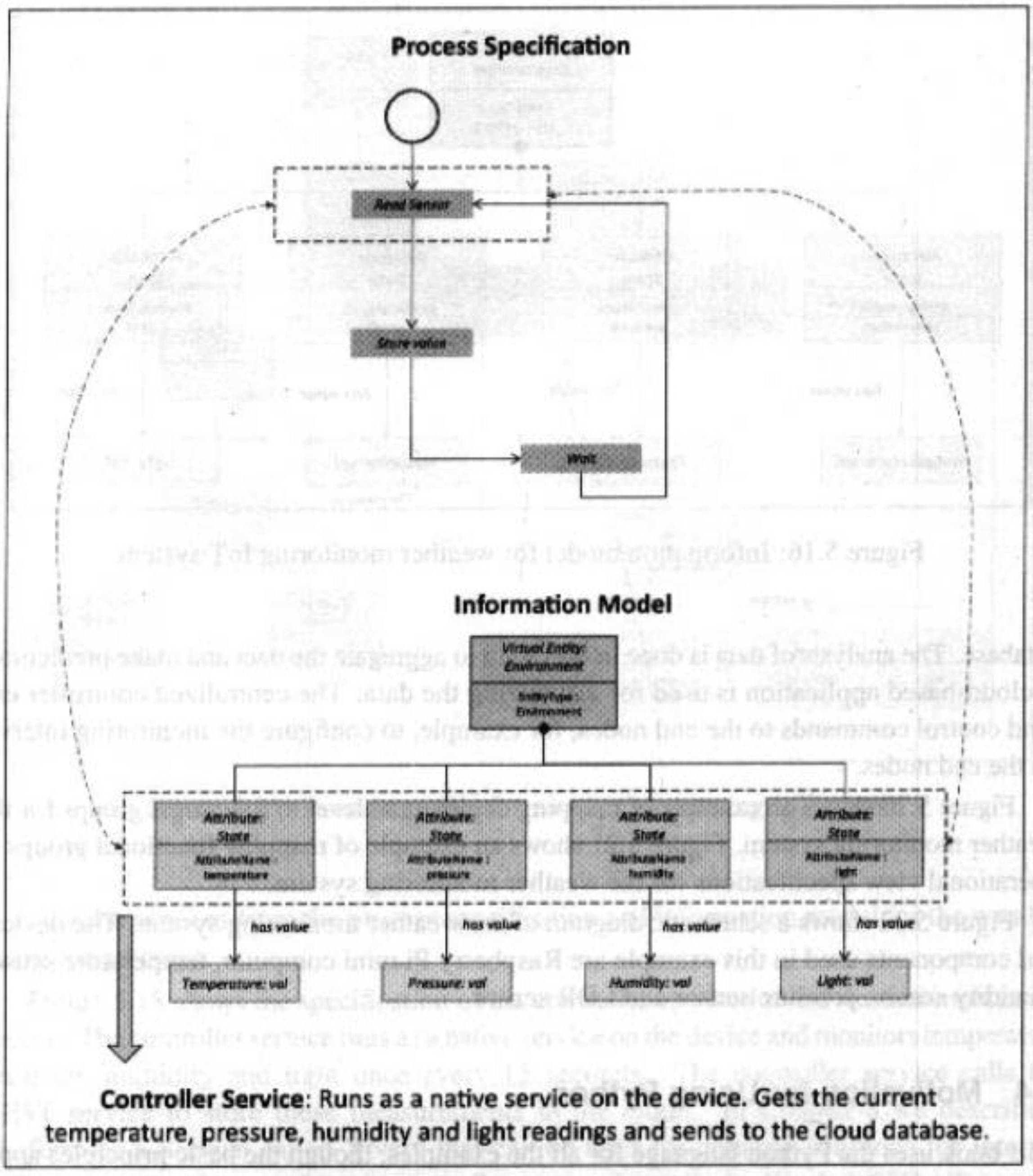


Figure 5.16: Information model for weather monitoring IoT system



Controller Service: Runs as a native service on the device. Gets the current temperature, pressure, humidity and light readings and sends to the cloud database.

Figure 5.17: Deriving services from process specification and information model for weather monitoring IoT system

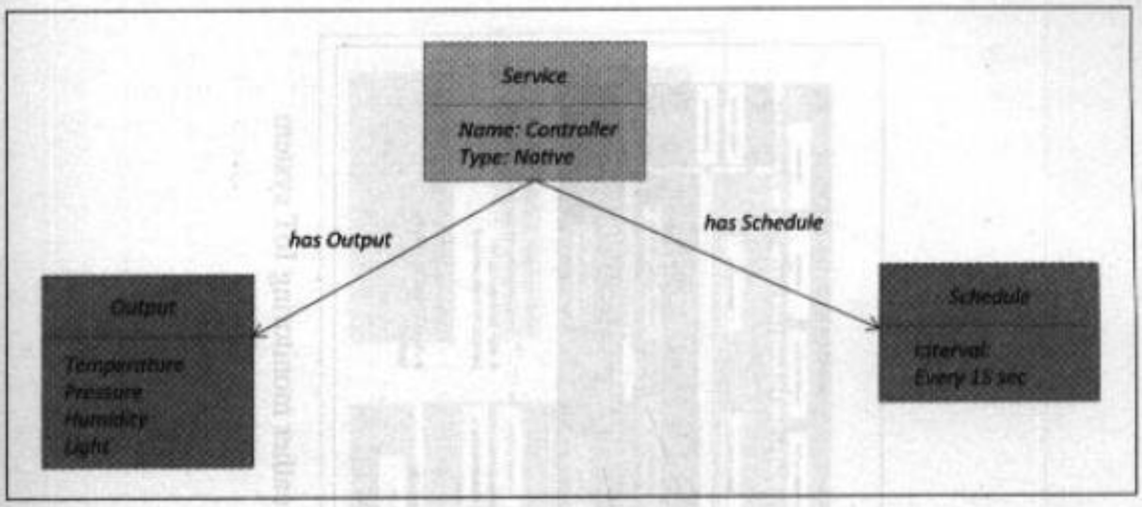


Figure 5.18: Controller service of the weather monitoring IoT system

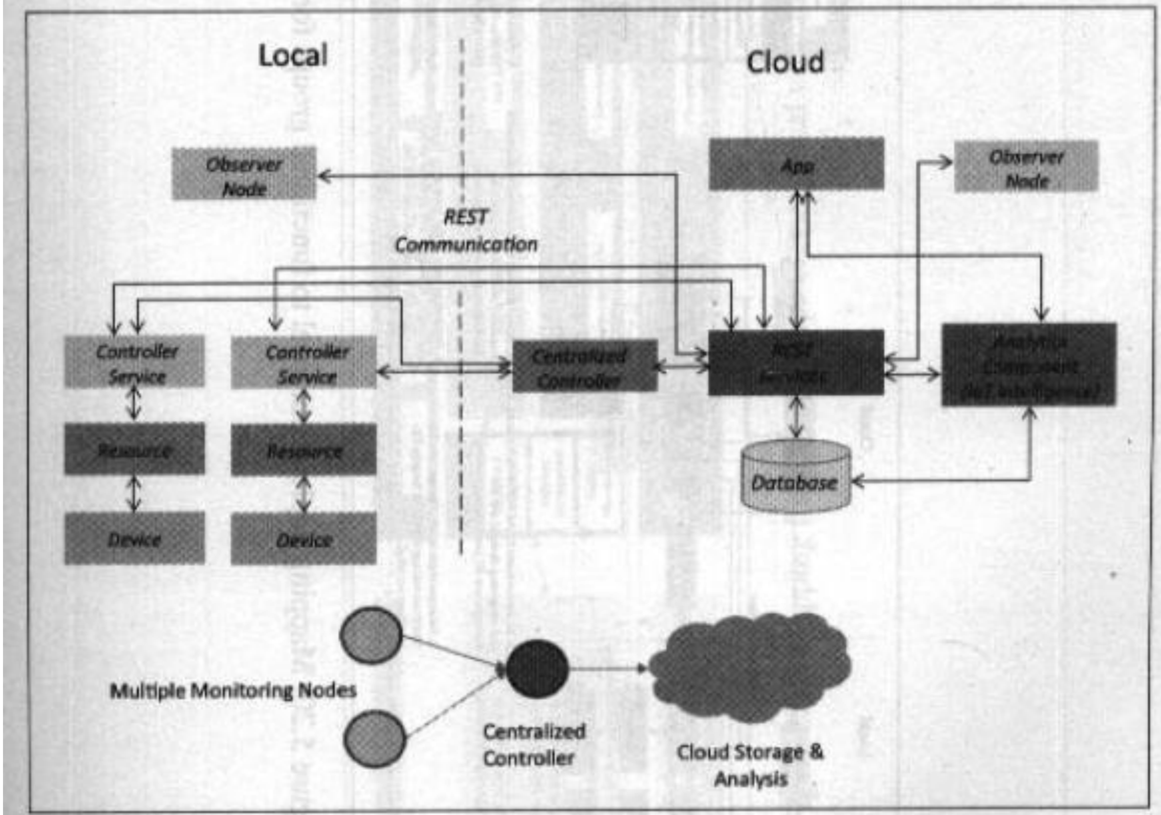
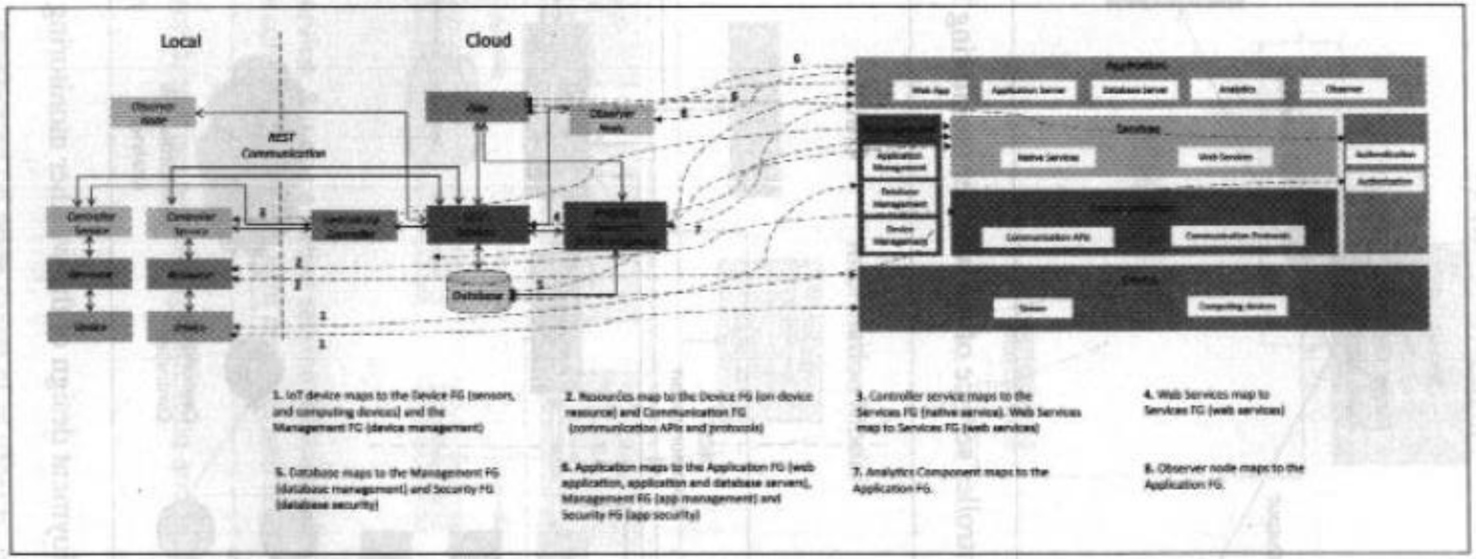


Figure 5.19: Deployment design of the weather monitoring IoT system



1. IoT device maps to the Device FG (sensors, and computing devices) and the Management FG (device management)

2. Resources map to the Device FG (on device resource) and Communication FG (communication APIs and protocols)

3. Controller service maps to the Services FG (native service). Web Services map to Services FG (web services)

4. Web Services map to Services FG (web services)

5. Database maps to the Management FG (database management) and Security FG (database security)

6. Application maps to the Application FG (web application, application and database servers), Management FG (app management) and Security FG (app security)

7. Analytics Component maps to the Application FG.

8. Observer node maps to the Application FG.

Figure 5.20: Mapping deployment level to functional groups for the weather monitoring IoT system

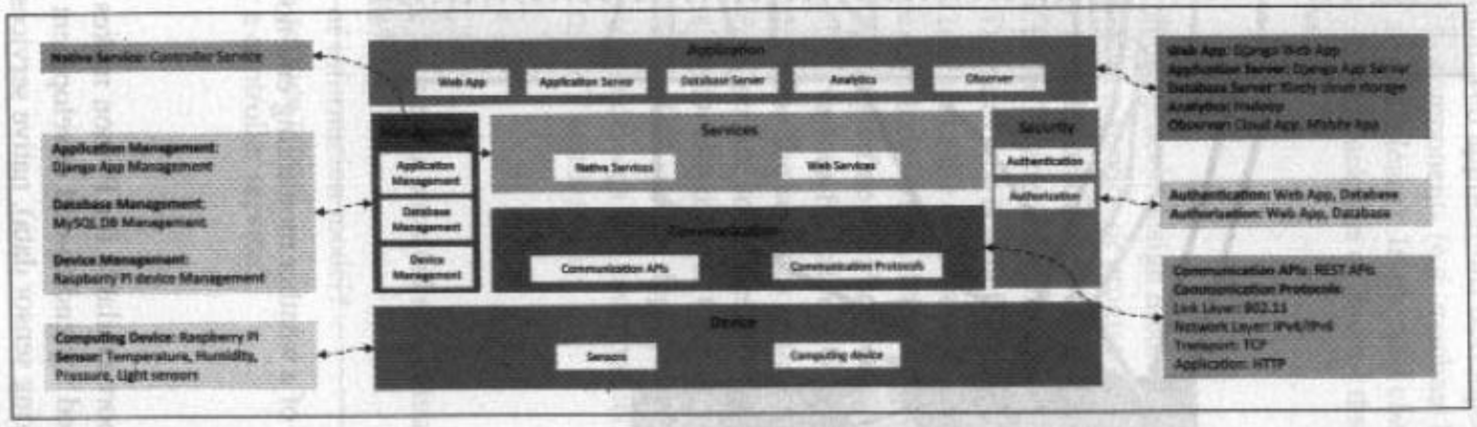


Figure 5.21: Mapping functional groups to operational view for the weather monitoring IoT system

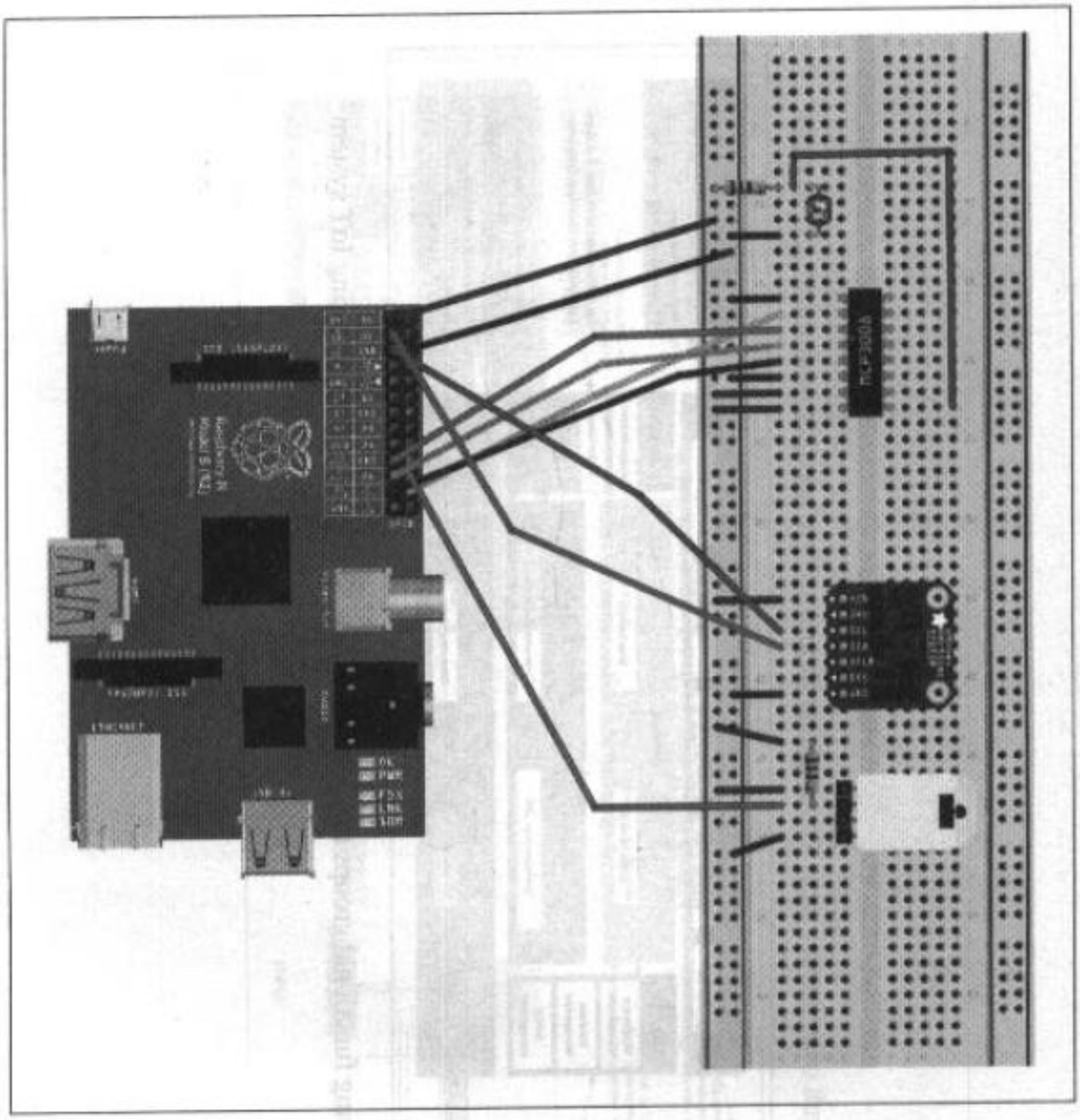


Figure 5.22: Schematic diagram of a weather monitoring end-node showing the device and sensors

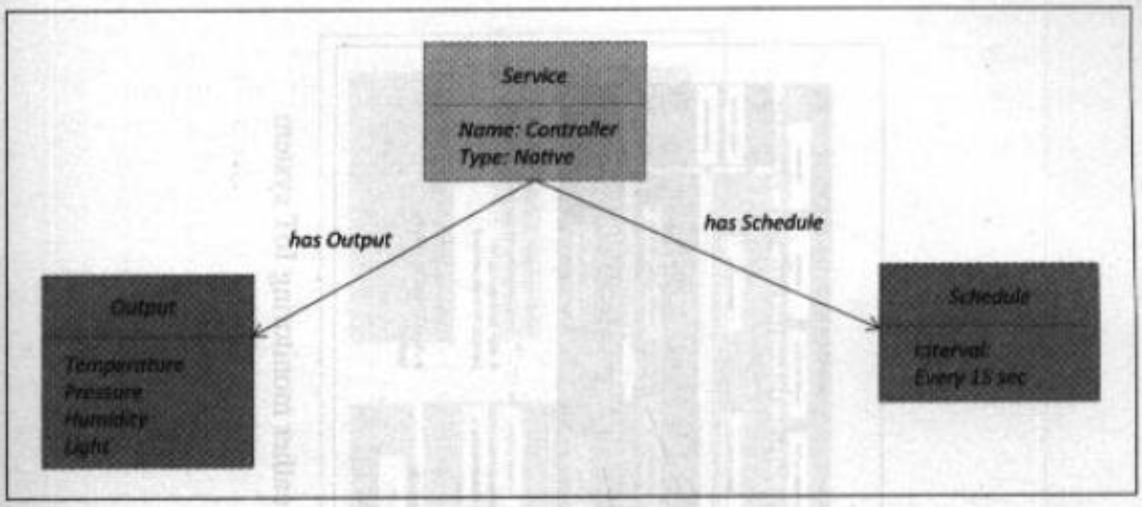


Figure 5.18: Controller service of the weather monitoring IoT system

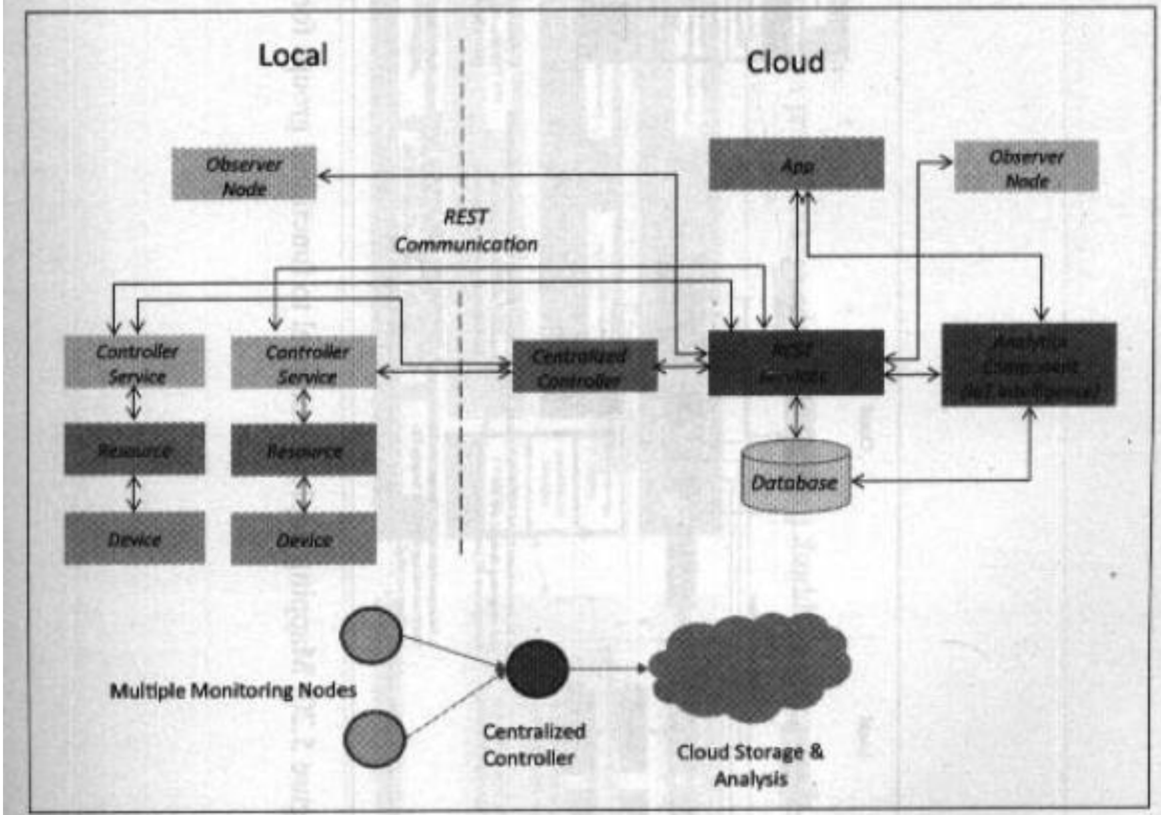


Figure 5.19: Deployment design of the weather monitoring IoT system

Part-A			
Q.No	Questions	Competence	BT Level
1.	What is domain model specification in IoT design	Understand	BTL 2

	methodology?		
2.	What is information model in IoT design methodology?	Understand	BTL 2
3.	What is purpose specification in IoT design methodology?	Understand	BTL 2
4.	What is requirement specification in IoT design methodology?	Understand	BTL 2
5.	What is service specification in IoT design methodology?	Understand	BTL 2
6.	What is level specification in IoT design methodology?	Understand	BTL 2
7.	What is functional view specification in IoT design methodology?	Understand	BTL 2
8.	What is operational view specification in IoT design methodology?	Understand	BTL 2
9.	What is device and component integration in IoT design methodology?	Understand	BTL 2
Part-B			
Q.No	Questions	Competence	BT Level
1.	Elaborate on the various steps involved in the IoT design methodology.	Analysis	BTL4
2.	Investigate on the domain model specification in the IoT Design?	Analysis	BTL4
3.	Analyze the method of deriving services from process and information model in a home automation system.	Analysis	BTL4
4.	Discuss on the various functional groups of a home automation.	Analysis	BTL4
5.	Illustrate the process specification, domain model and information model of a weather monitoring system.	Analysis	BTL4

UNIT III - PILLARS OF EMBEDDED IoT AND PHYSICAL DEVICES

Pillars of Embedded IoT

Outline

•Horizontal and Vertical Applications of IoT
•Four Pillars of IoT
•M2M : Internet of Devices
•RFID : Internet of Objects
•WSN : Internet of transducer
•SCADA : Internet of Controllers
•DCM :

–Device : Things that talk
–Connect : Pervasive Network
–Manage : Create Business Values

IoT Pysical Devices and Endpoints - –Basic building blocks of IoT device

•Exemplary device: Raspberry Pi

•Raspberry Pi interfaces

•Programming Raspberry Pi with Python

•Beagle board and other IoT Devices

Horizontal Applications

•Provide solution to common problems

•These are not business specific

•Can be Used monitored and controlled by multiple companies

•Allows multiple providers to work together on a single platform

Advantages

•Robust

•Developed fast and less costly

Vertical Applications

•These are business specific

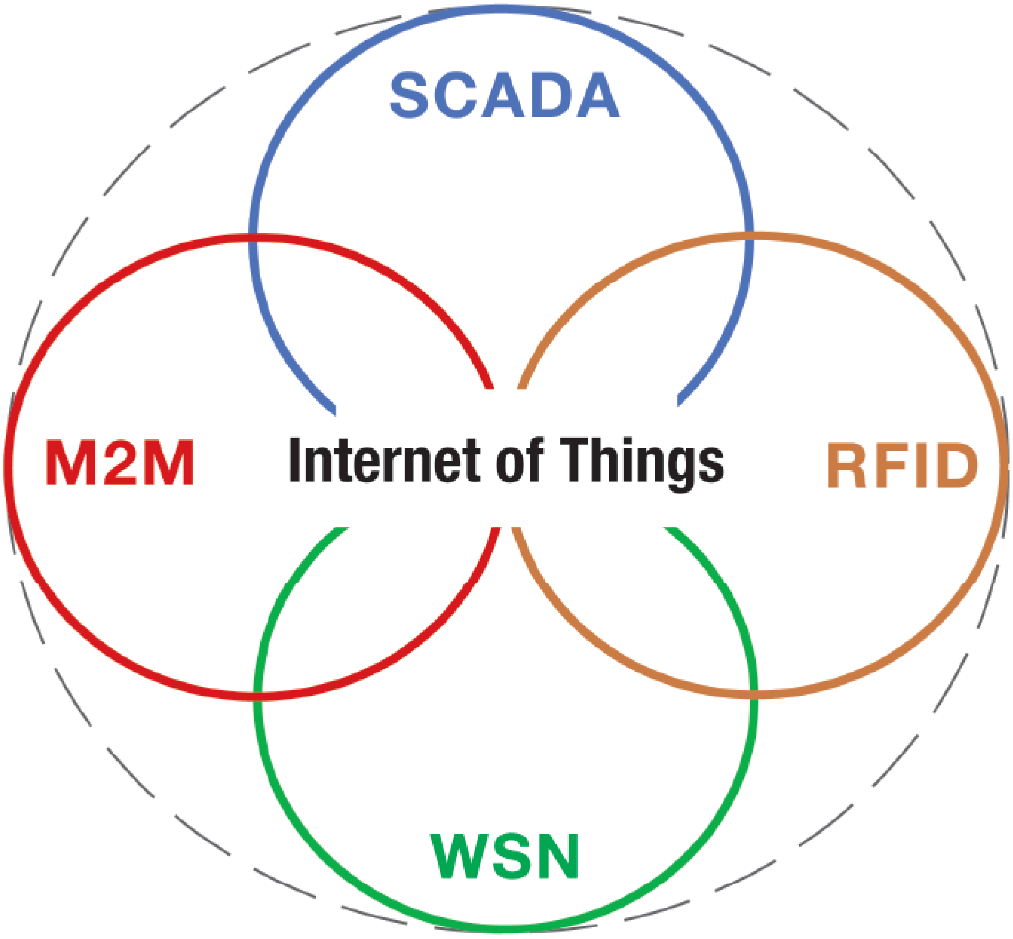
•Can be Used monitored and controlled by only single company

•Does not allow multiple providers to work together on a single platform

Advantages
•No compatibility issue as no other companies are involved

Disadvantages
•Depended entirely on a single vendor for modifications or upgrades

Four Pillars of IoT



M2M - •Machine to Machine

•Enables **flow of data between machines** which **monitors data by** means of **sensors** and at other end **extracts the information on gathered data and processes it**.

•Subset of IoT

•It uses WAN , GPRS, Cellular and Fixed N/w’s

M2M Architecture

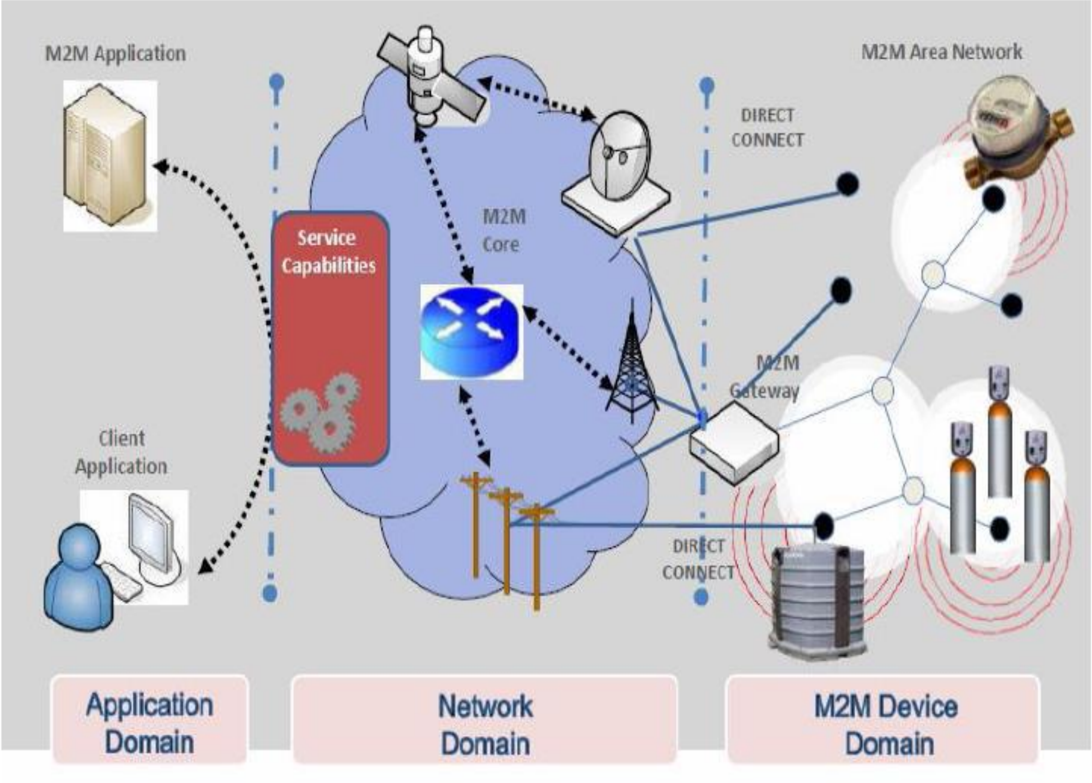


Figure 1: Architecture of M2M system

Components of M2M architecture are :

1)M2M Devices

2)M2M Area Network i.e Device Domain

3)M2M Gateway

4)M2M Communcation N/w’s : Network Domain

5)M2M Applications i.e Application Domain

M2M Devices:

Device that are capable of replying to request for data contained within those devices or capable of transmitting data autonomously are M2M Devices.

•**Sensors and communication devices** are the endpoints of M2M applications.

M2M Area Network

•Provide connectivity between M2M Devices and M2M Gateways.

•E.g. Personal Area Network

M2M Network Domain

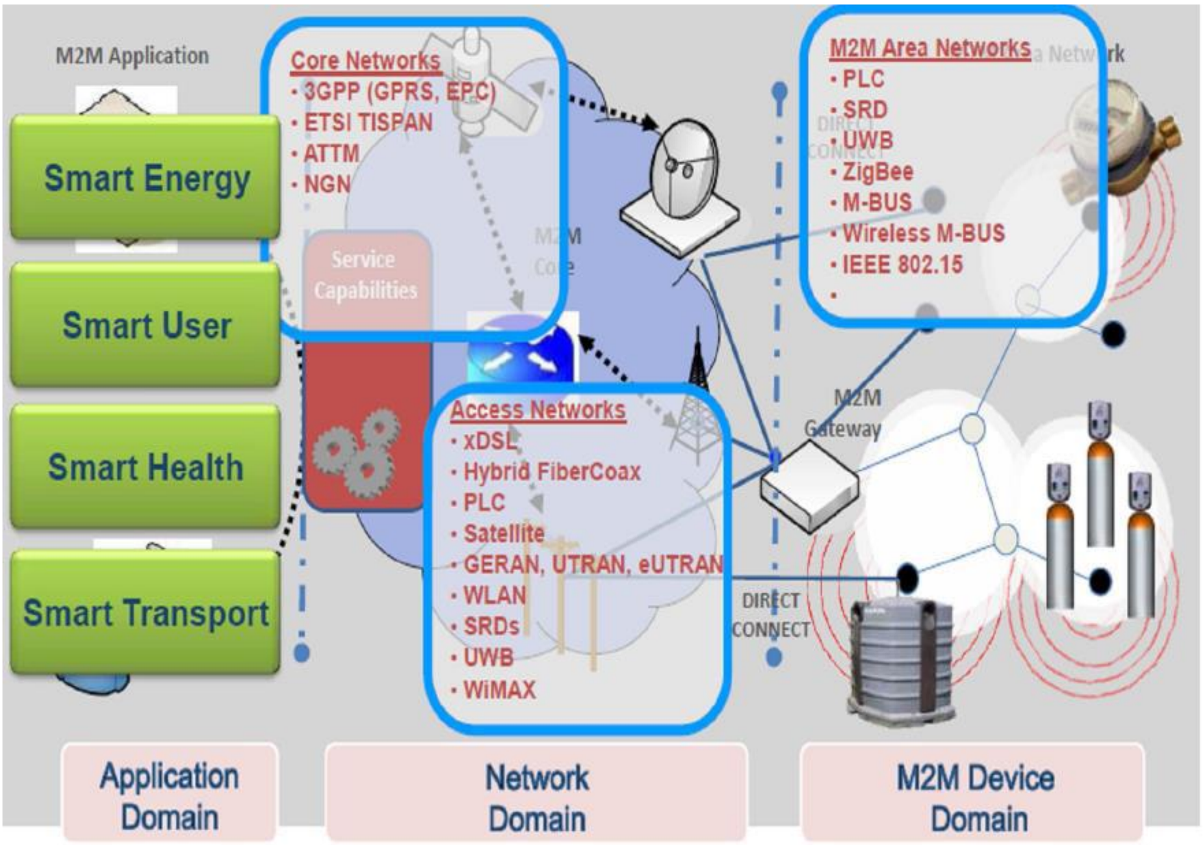
•Communication between M2M Gateways and M2M Applications.

•E.g. WiMax, WLAN, LTE

M2M Application Domain

•It contains the middleware layer where data goes through various application services and is used by the specific business-processing engines.

Examples of M2M Components

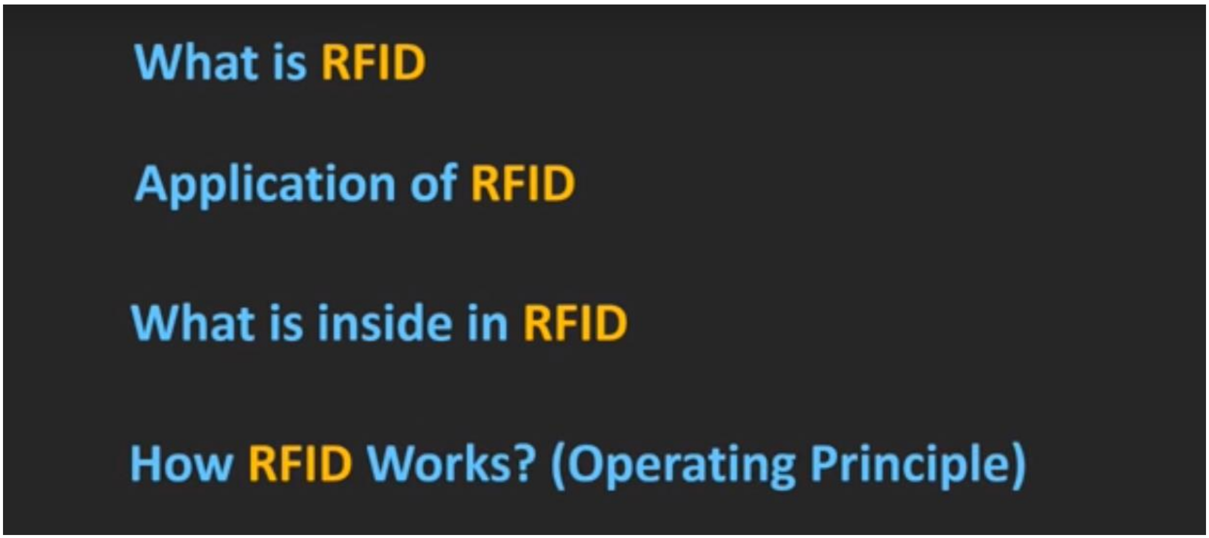


RFID -•Radio Frequency Identification

•Uses **radio frequency** to **read** and capture information stored on a **tag** attached to an object.

•A tag can be read from up to several feet away and does not need to be within direct line-of-sight of the reader to be tracked.

•Uses NFC (Next Field Communication protocol), IC (Integrated Circuit) Cards, Radio Waves



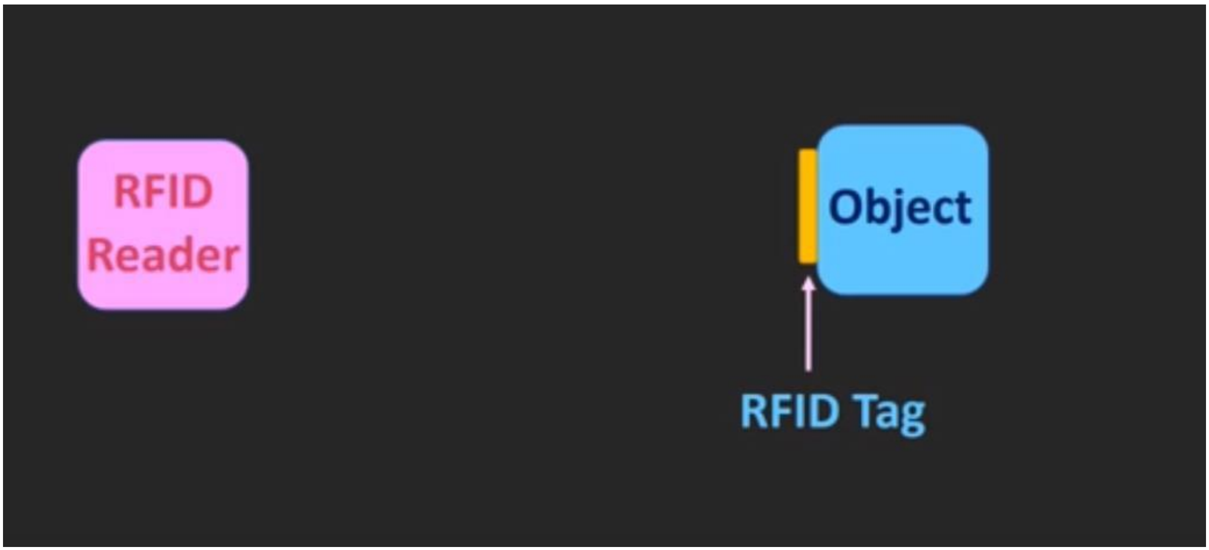
What is RFID

Application of RFID

What is inside in RFID

How RFID Works? (Operating Principle)

What is RFID?



Objects can be books in library

Objects can be items in shopping mall

Objects can be inventory in the warehouse

Objects can be a car

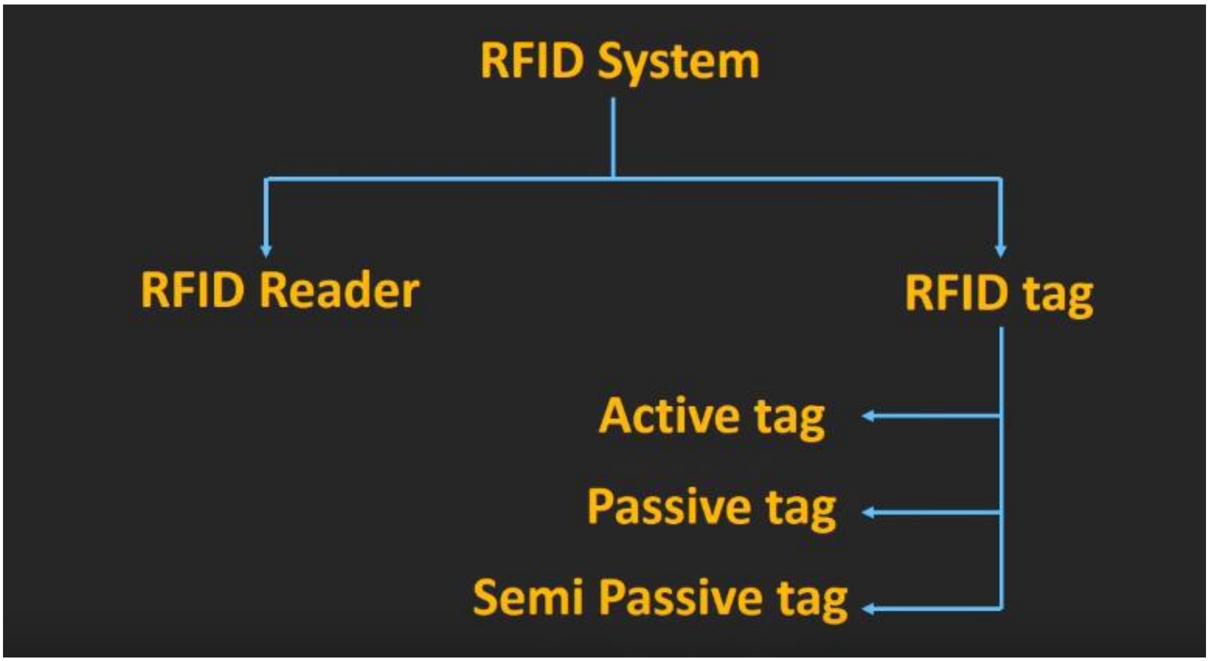
RFID Vs Bar Code

In Bar code the scanner must be in line of sight…and this is not mandatory in RFID.

•RFID can track multiple objects while Bar Code cannot .

RFID Vs Bar Code Vs QR Code

RFID System


RFID System

RFID Reader

RFID tag

Active tag

Passive tag

Semi Passive tag

RFID Tags

•

Passive Tags do not have their own power supply, hence rely on radiowaves for source of energy

•SemiPassive Tag have their own power supply, but for transmitting back they rely on signals coming from RFID Reader

•Active Tag uses their own power supply for both transmitting and receiving

•Range of Passive Tags is less than that of Semi and Active Tags

Passive Tags are cheaper

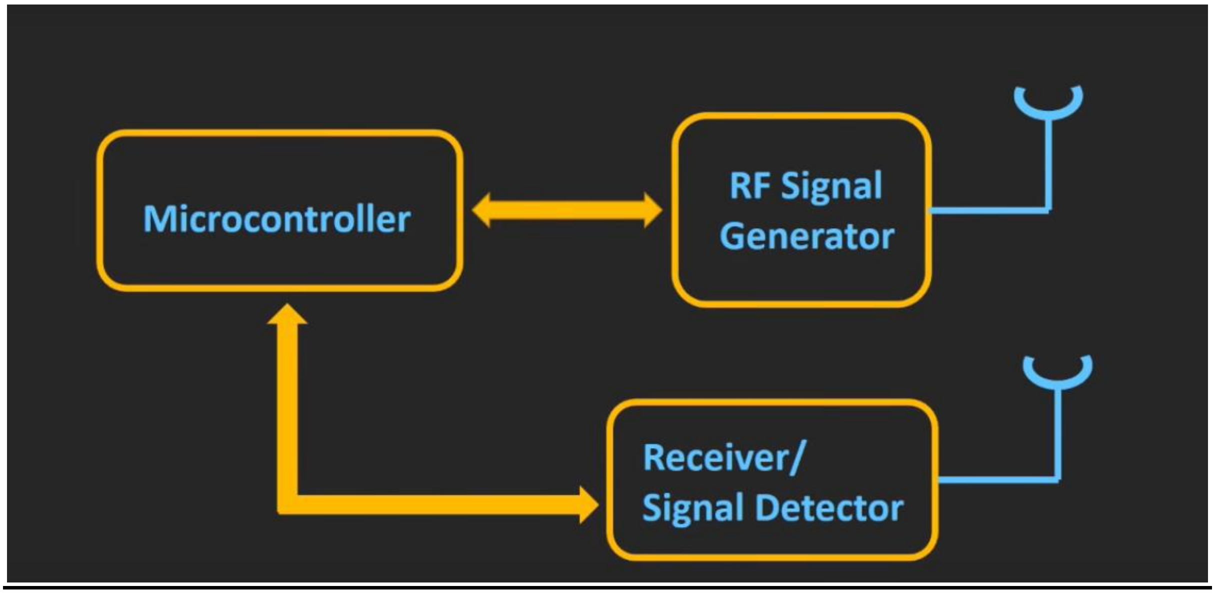
•Passive tags do not use any power source hence are compact

RFID Reader

•Come in many size and shapes

HandHeld Reader

RFID Reader

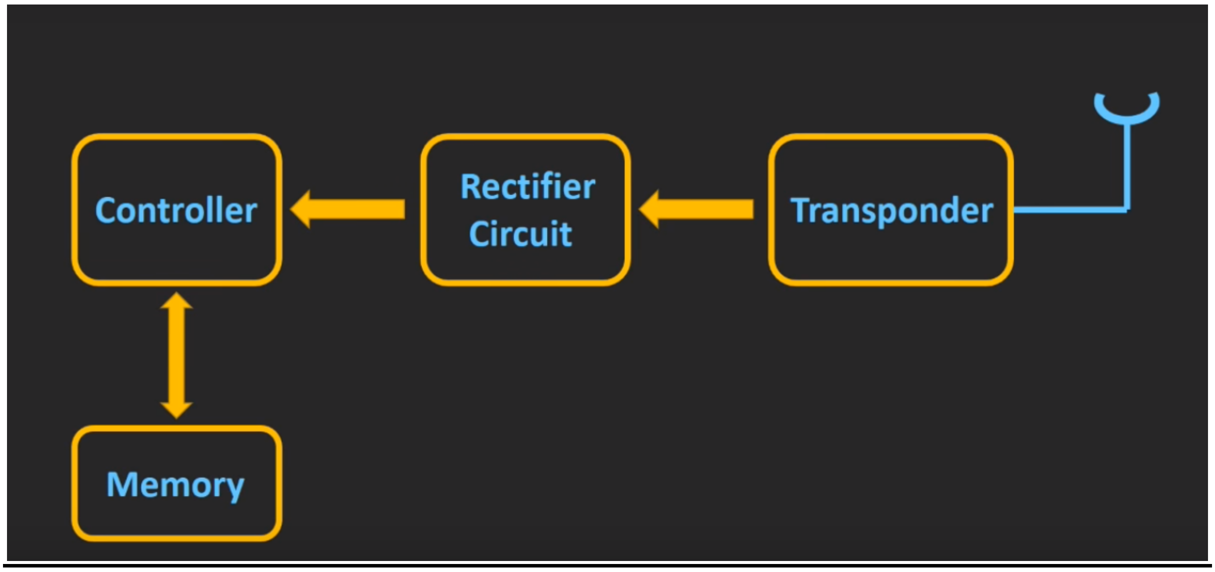


RF Signal Generator Generates radio waves which are transmitted through the antenna

•Receiver or signal detector receives the signals coming from the object

•And to process these signals microcontroller is used

RFID Tag

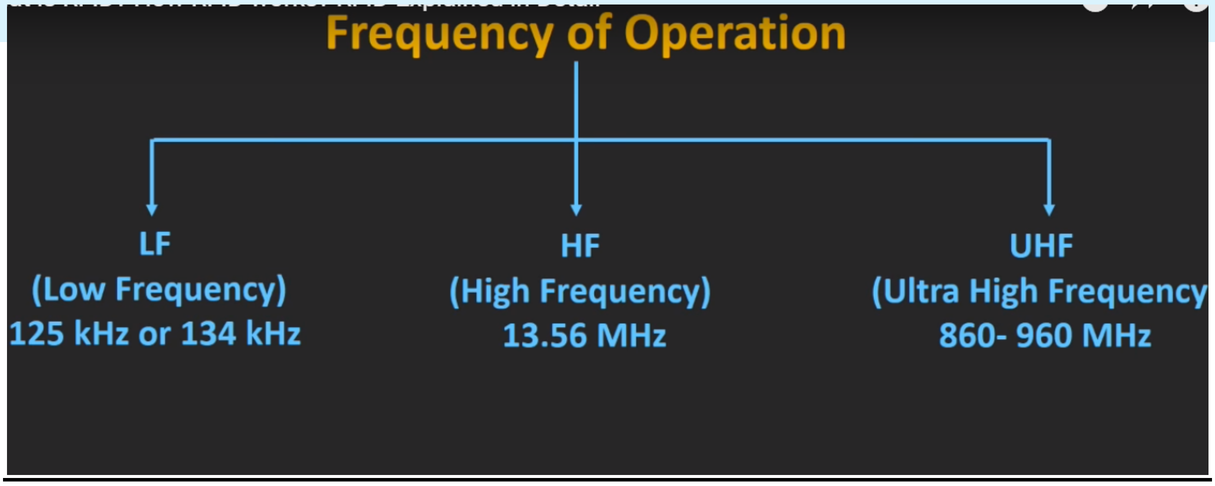


Transponder receives signals from reader and sends back feedback to the reader

•The Passive Tags use the rectifier circuit to store the energy coming from the radio waves.

•This energy is used as the supply for the controller and the memory element

RFID Frequency Operation



Frequency of Operation

LF
(Low Frequency)
125 kHz or 134 kHz

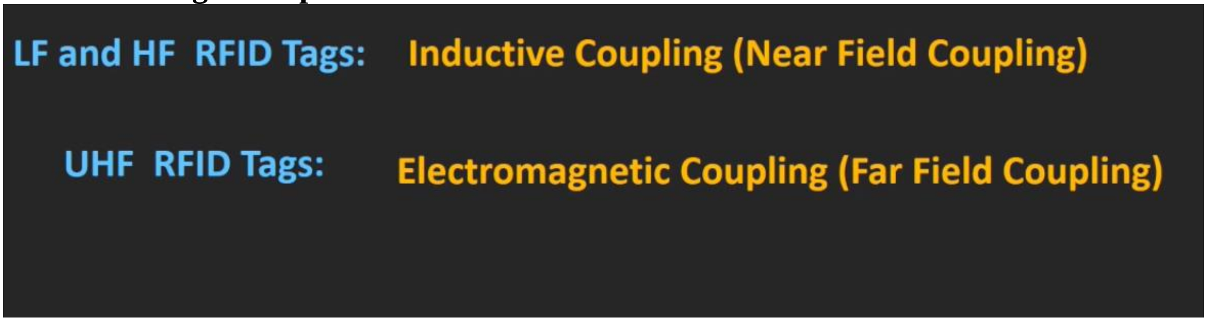
HF
(High Frequency)
13.56 MHz

UHF
(Ultra High Frequency
860- 960 MHz

Person Identification
Clothes at Shopping Mall
30 mm
90mm
RFID
Hang Tag
SIZE L
€50
Vehicle Identification at Toll Plaza
tag reader
traffic monitoring camera
E-ZPass tag
5 MPH
traffic gate

RFID Frequency Operation
Frequency Operation varies from country to country.

RFID Working Principle



LF and HF RFID Tags: Inductive Coupling (Near Field Coupling)

UHF RFID Tags: Electromagnetic Coupling (Far Field Coupling)

SCADA

- **Supervisory Control and DataAcquisition**
- These connect , monitor and control equipment's **using short range n/w inside a building or an industrial plant**
- Uses **BacNet** (communication protocol) , **CanBus** (Controller Ara N/w) and

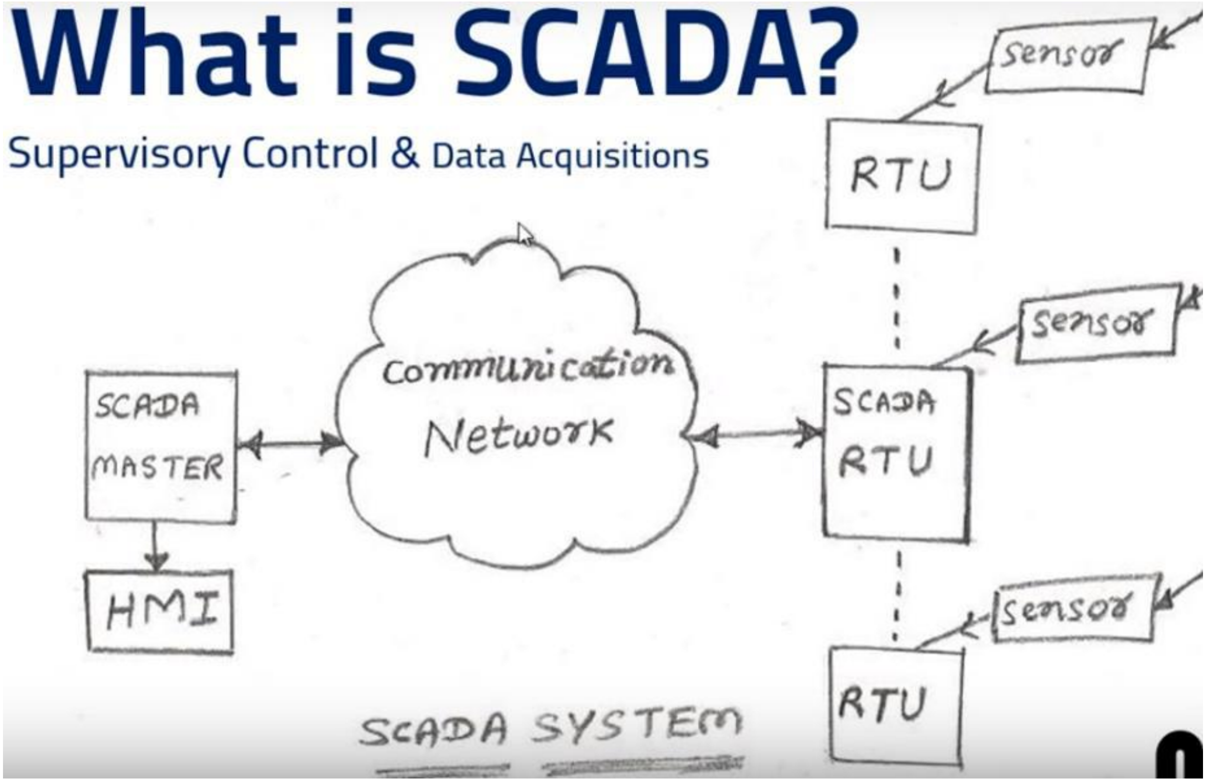
Wired FieldBuses(Industrial Computer NetworkProtocols)

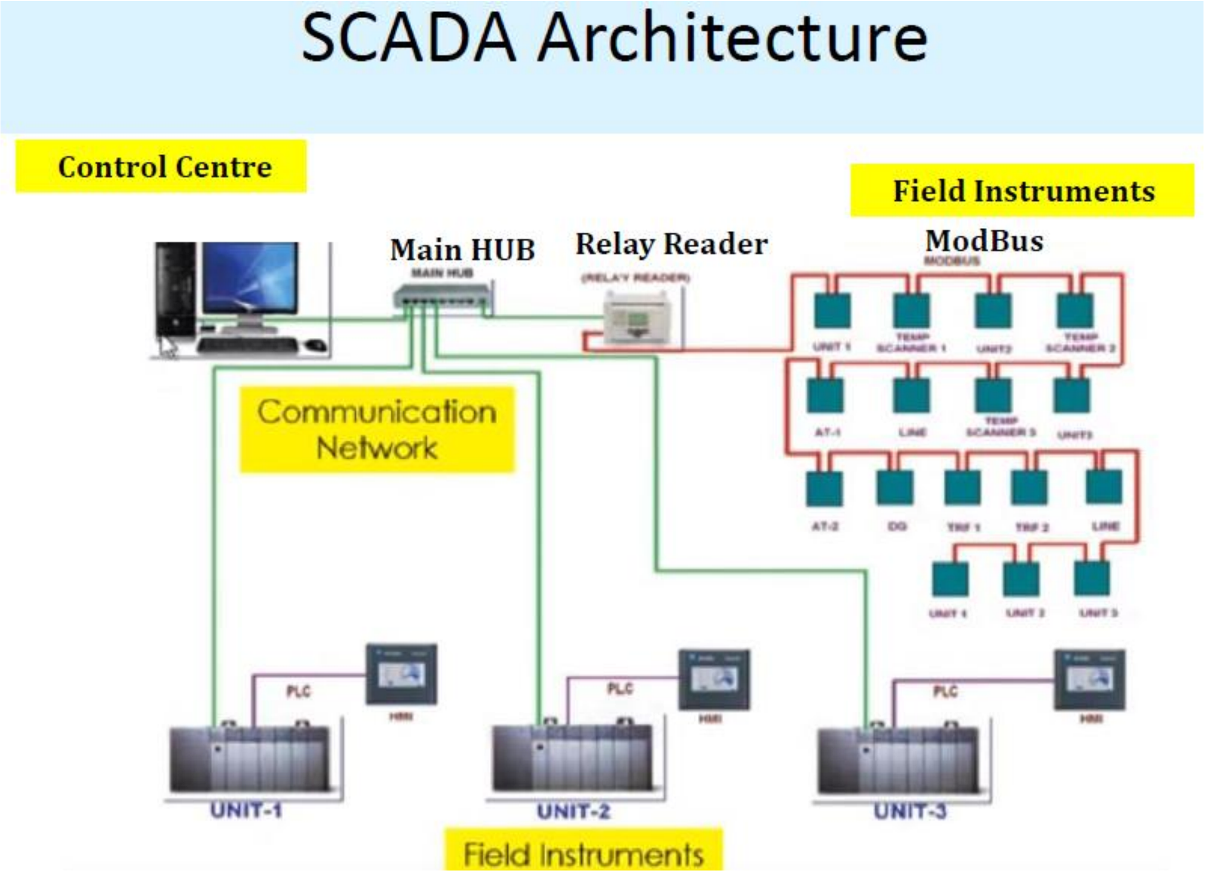
- **Supervisory** means top level

- **Control** means controlling things

- **Data Acquisition** means acquiring the data / reading the data

- **SCADA ia a s/w** used to **control** the **hardware** i.e PLC, drives , servers , sensors and also **acquire the data** which is **stored on** the **personal computer** or **Human Machine Interface(HMI)**

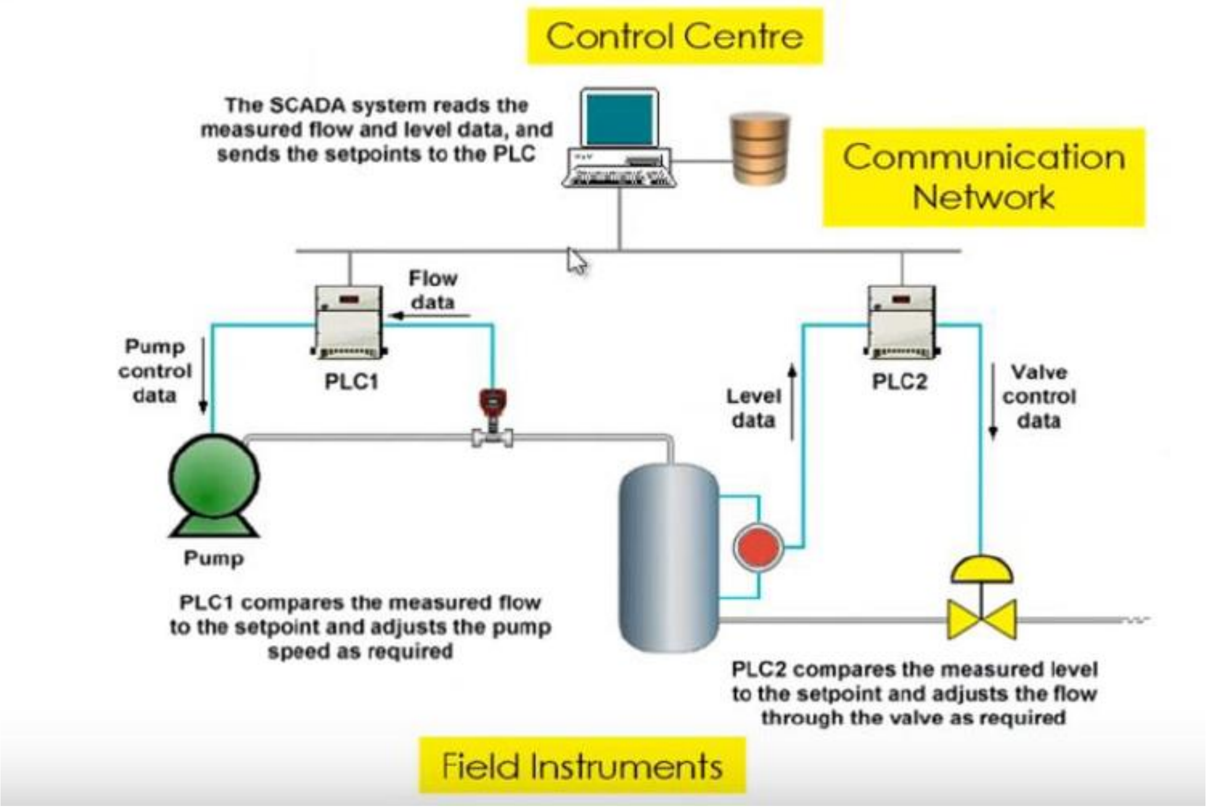




- SCADA Architecture has a **control centre** connected to the main hub(i.e the ethernet port)
- **The PLC**(Relay Reader is **connected to** the **ethernet board** which is overall **connected tothe CPU.**
- **PLC** on other hand is **connected to various field instruments** which can be the temperature sensors or actuators that can be analog or digital
- The PC has a **SCADA s/w** which can **interact with** the **field instruments**
- PC is also connected to various other PLC’s UnitsI , Unit II, as shown in Diagram.
- There is an Human Machine Interface(**HMI**) which is **individually**

  connected to the PLC
- The **HMI** individually **monitors and controls** the **PLC** TO read information from all the units we need aSCADA system.
- **PLC (Programmable Logic Unit) monitor and control the data**
- **RTU(Remote Terminal Units)** receive data from sensors and convert this data to digital data and sendto SCADA system

Example of SCADA Architecture

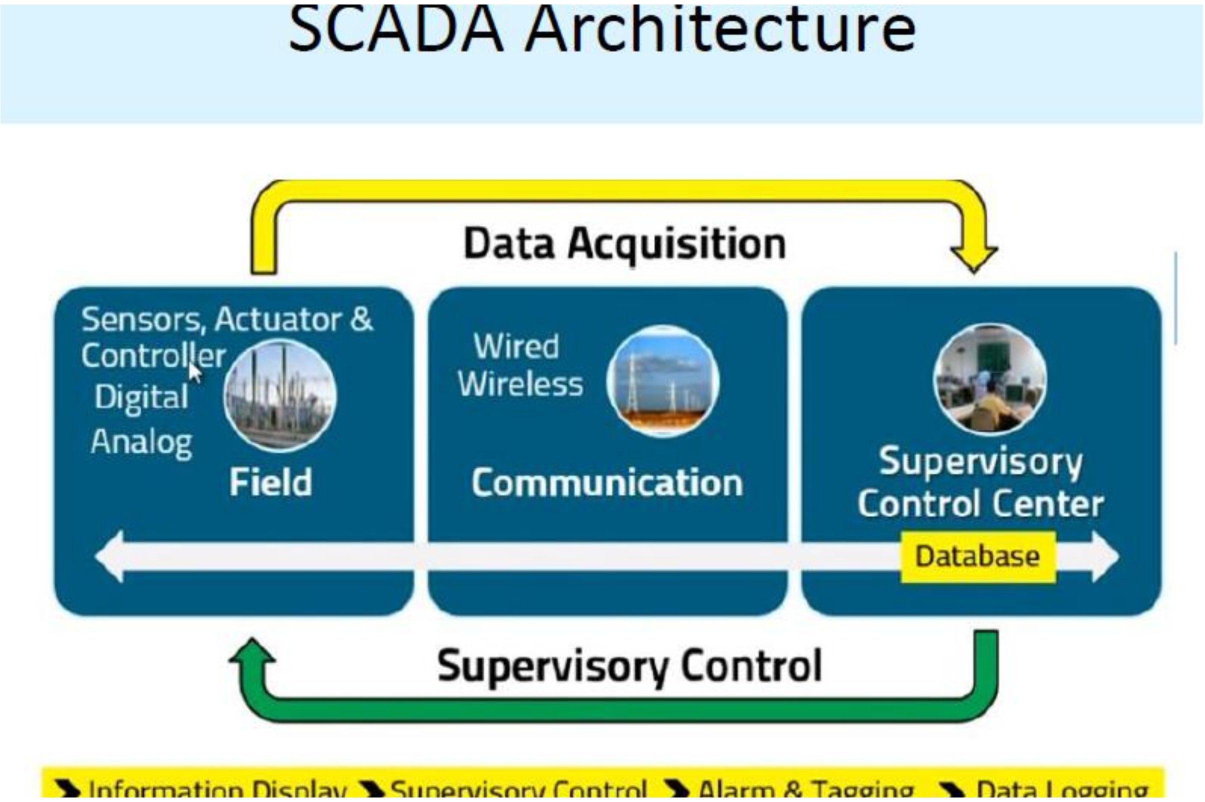

Control Centre
The SCADA system reads the measured flow and level data, and sends the setpoints to the PLC
Communication Network
Flow data
Pump control data
PLC1
Level data
PLC2
Valve control data
Pump
PLC1 compares the measured flow to the setpoint and adjusts the pump speed as required
PLC2 compares the measured level to the setpoint and adjusts the flow through the valve as required
Field Instruments

As shown in the figure the Pump is controlled by the PLC1, which controls speed of the pump.

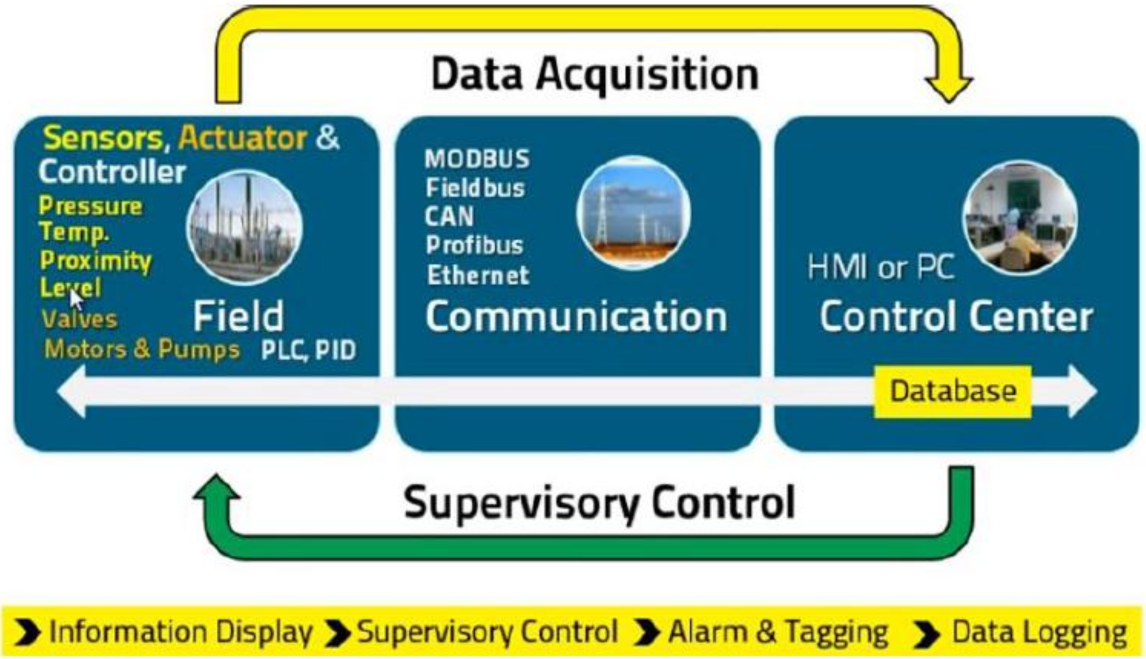
•There is a water level sensor that senses the level of water in water tank and gives the information to PLC2 that monitors the level of the water.

•The PLC1 and PLC2 are connected to the SCADA system.

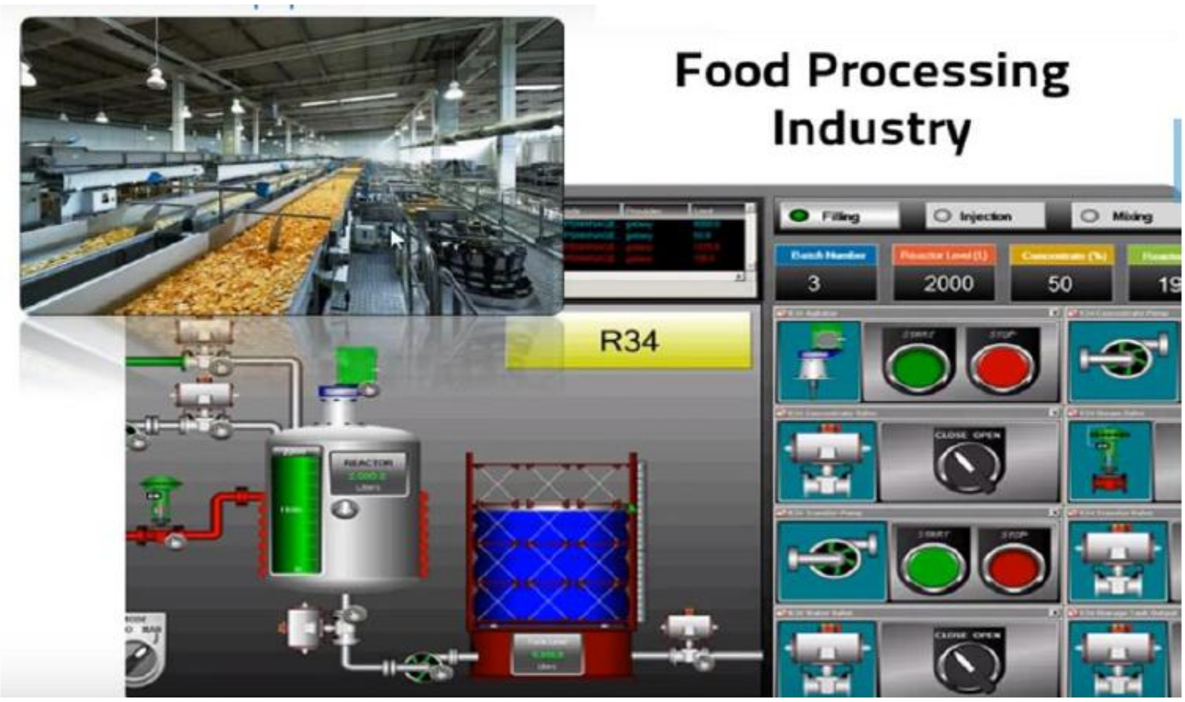
The system contains the database which can store the data of speed of the pump and water level.



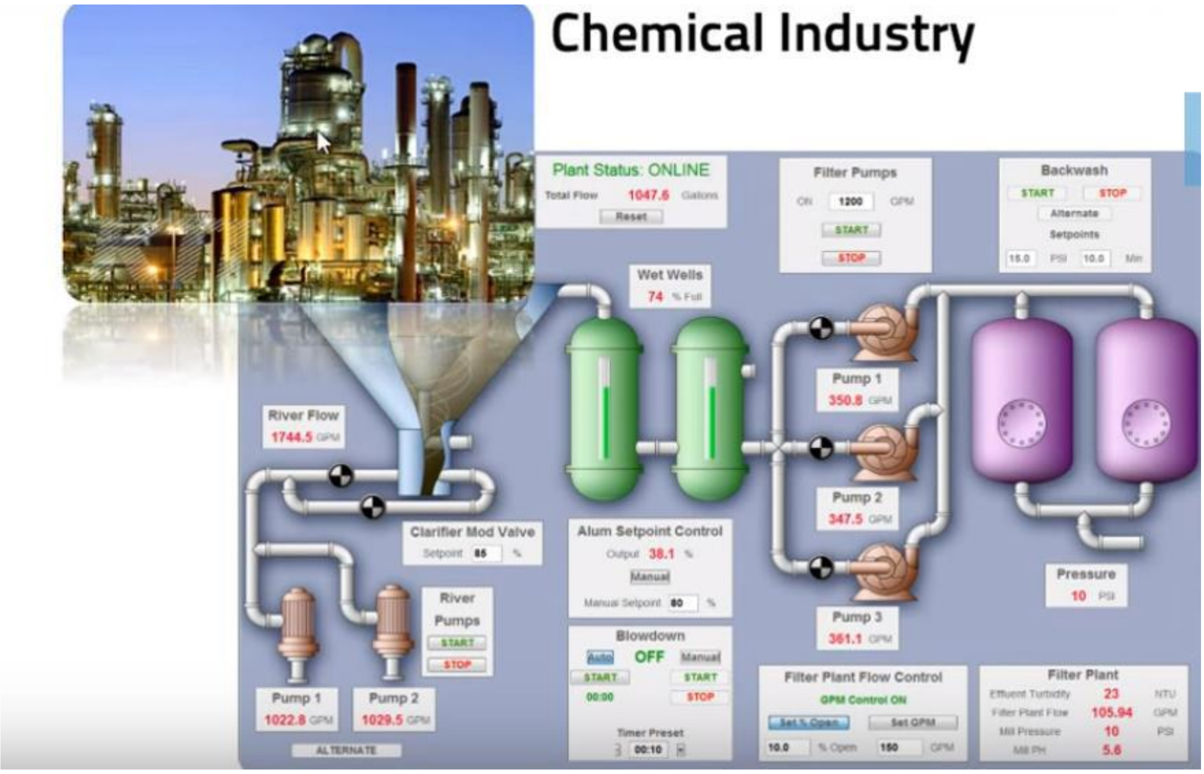
SCADA Architecture



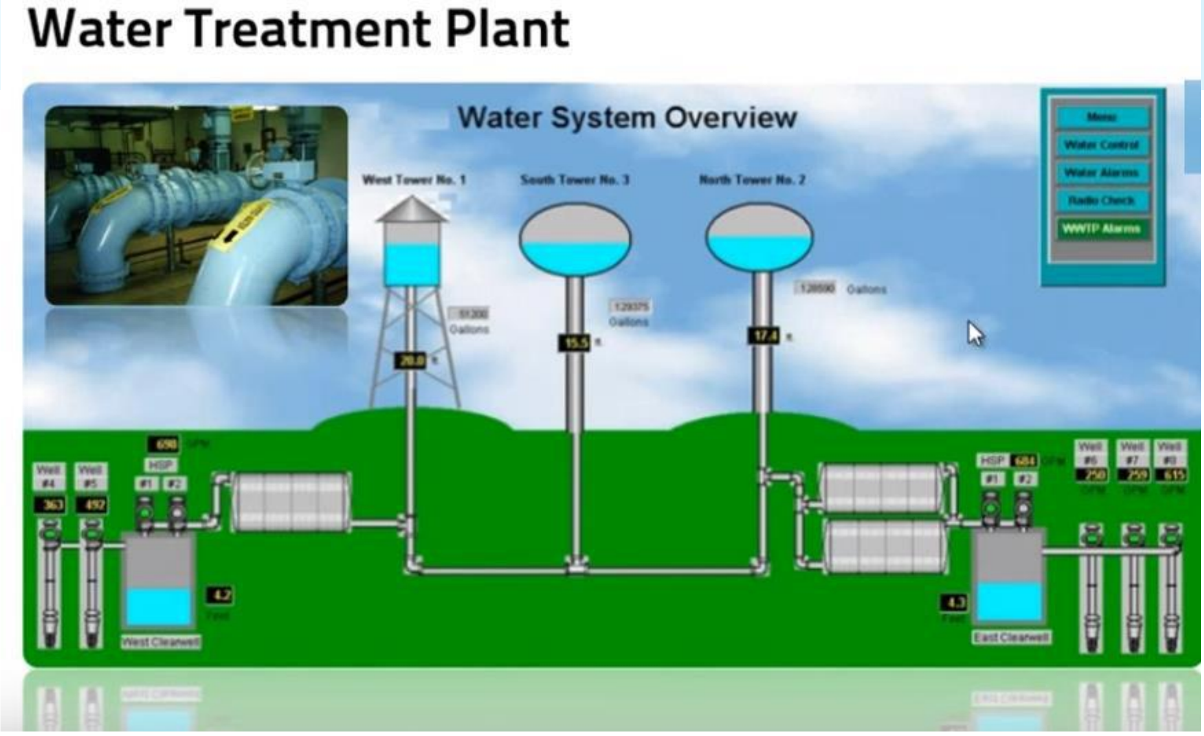
SCADA Applications



Food Processing Industry



Chemical Industry



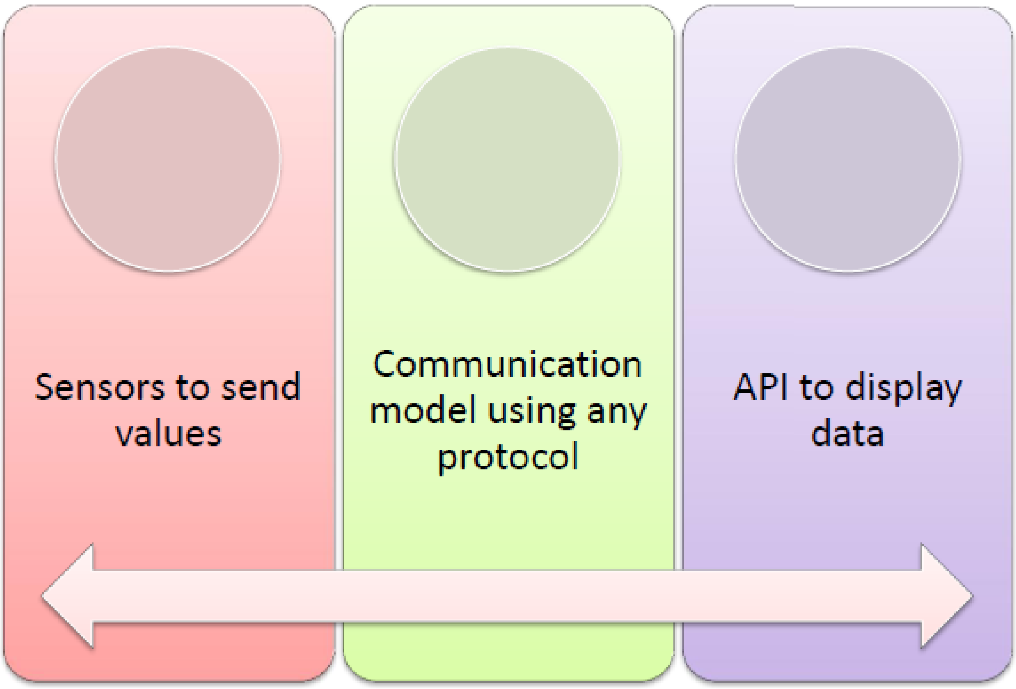
Water Treatment Plant

WSN

- **Wireless Sensor N/w**
- It senses and gathers data using sensors which are spatially distributed
- It collects this data into a centralized location with the help of wired / wireless connection

*It is a deployment of several devices equipped with sensors that perform a collaborative measurement process

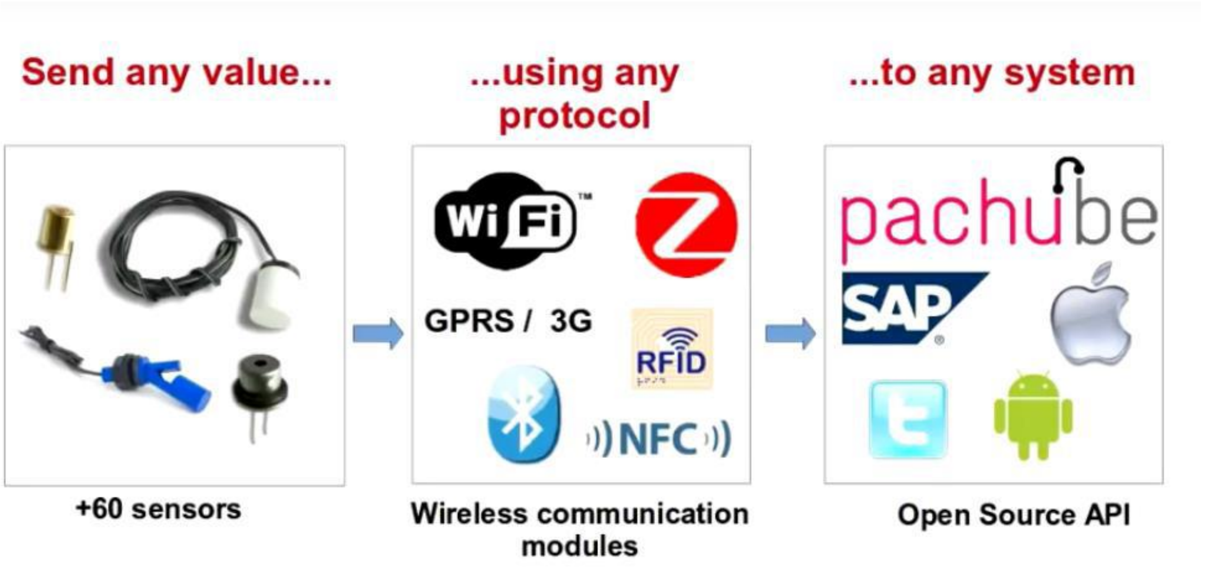
- **Consists of three basic things**



Sensors to send values

Communication model using any protocol

API to display data


Send any value...

...using any protocol

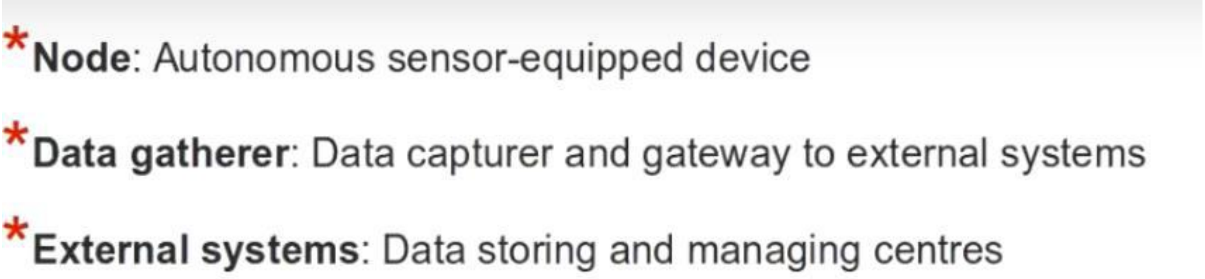
...to any system

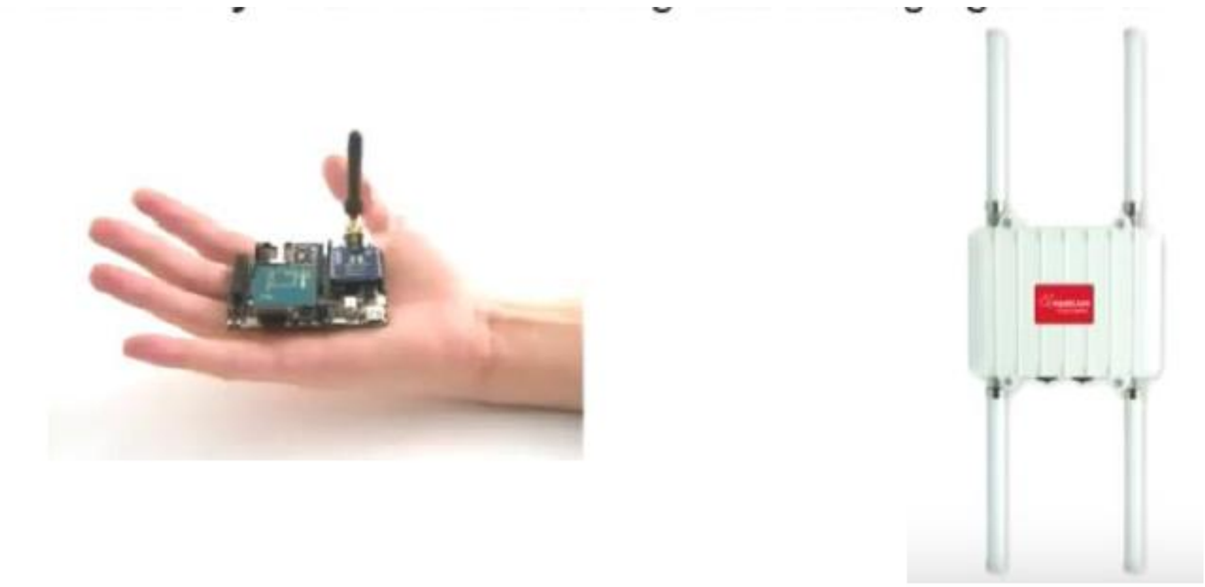
+60 sensors

Wireless communication modules

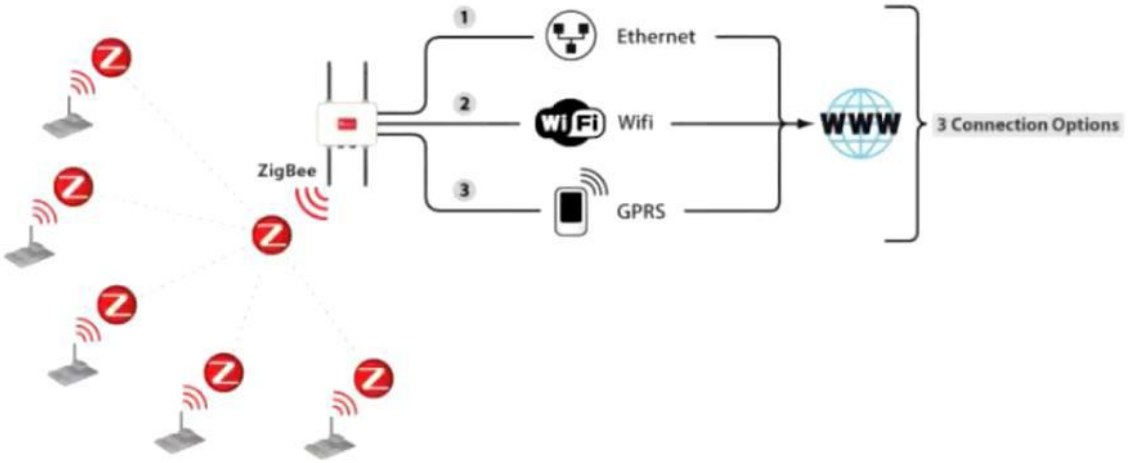
Open Source API

WSN Elements


- **Node**: Autonomous sensor-equipped device
- **Data gatherer**: Data capturer and gateway to external systems
- **External systems**: Data storing and managing centres



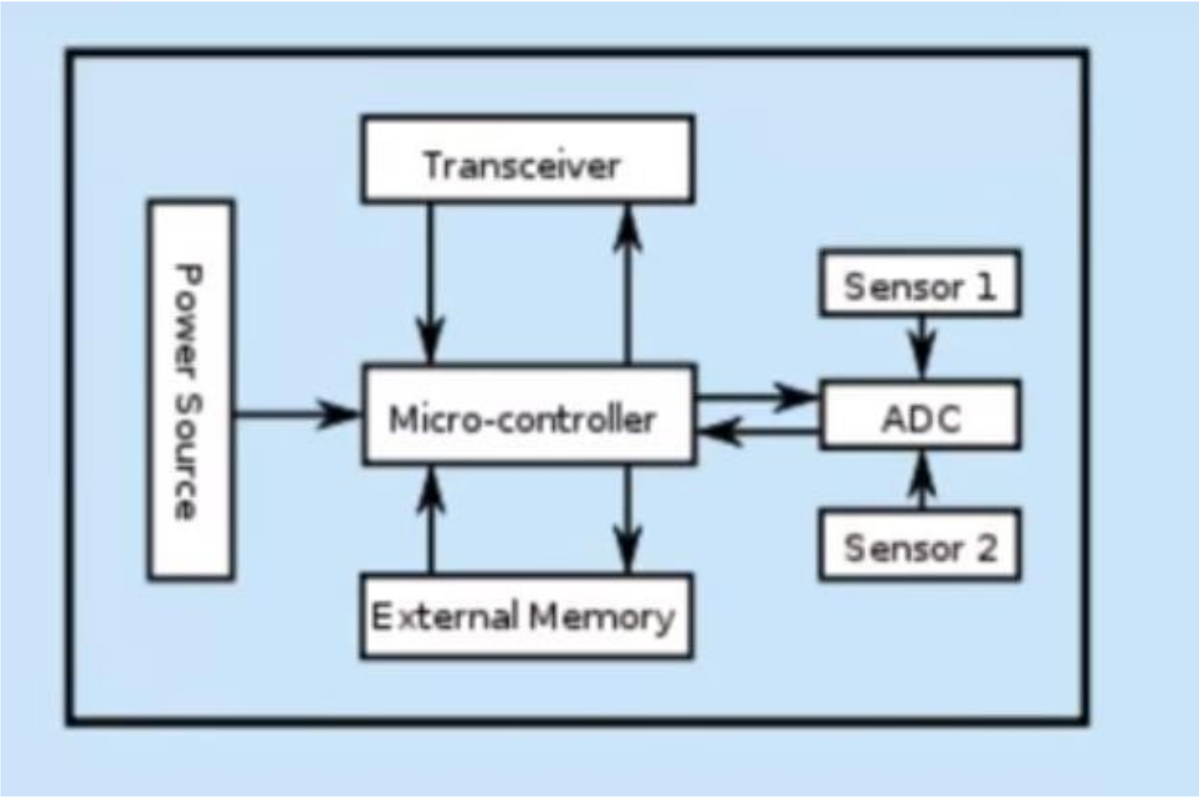
Working of WSN

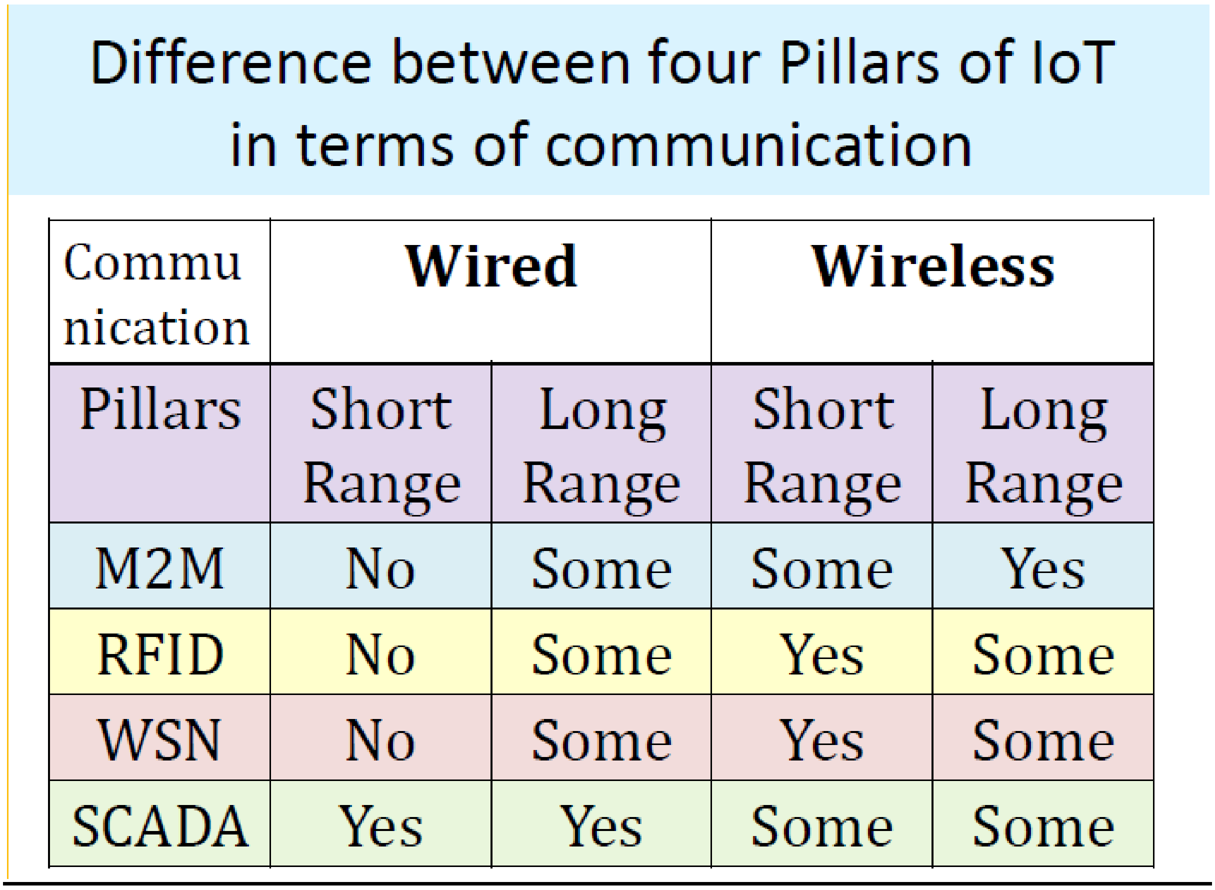


In the above diagram we can see the **sensors (nodes)** are **sensing** the **device values**

•These **transmit** the **information to** the **measuring device (data gatherer)** which **transmits** the **values to external systems using Ethernet, Wifi, or GPRS**.

Parts of WSN

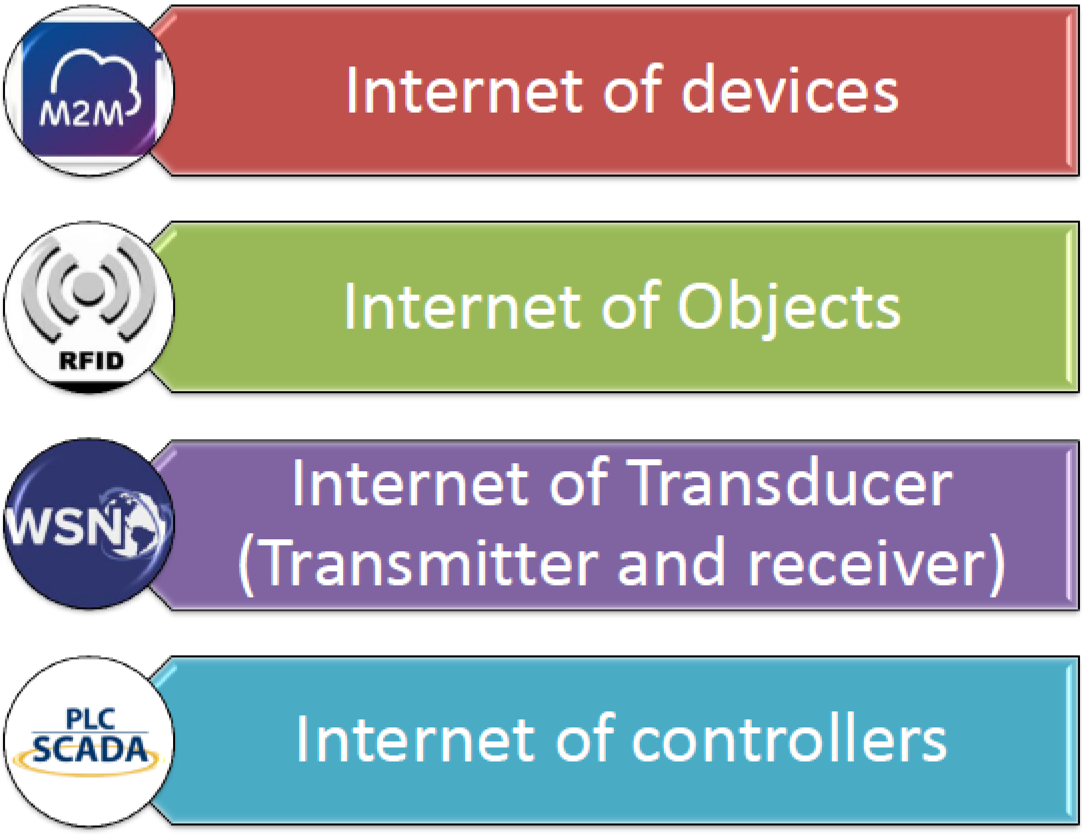




Difference between four Pillars of IoT in terms of communication

Communication	Wired		Wireless	
Pillars	Short Range	Long Range	Short Range	Long Range
M2M	No	Some	Some	Yes
RFID	No	Some	Yes	Some
WSN	No	Some	Yes	Some
SCADA	Yes	Yes	Some	Some

So.....



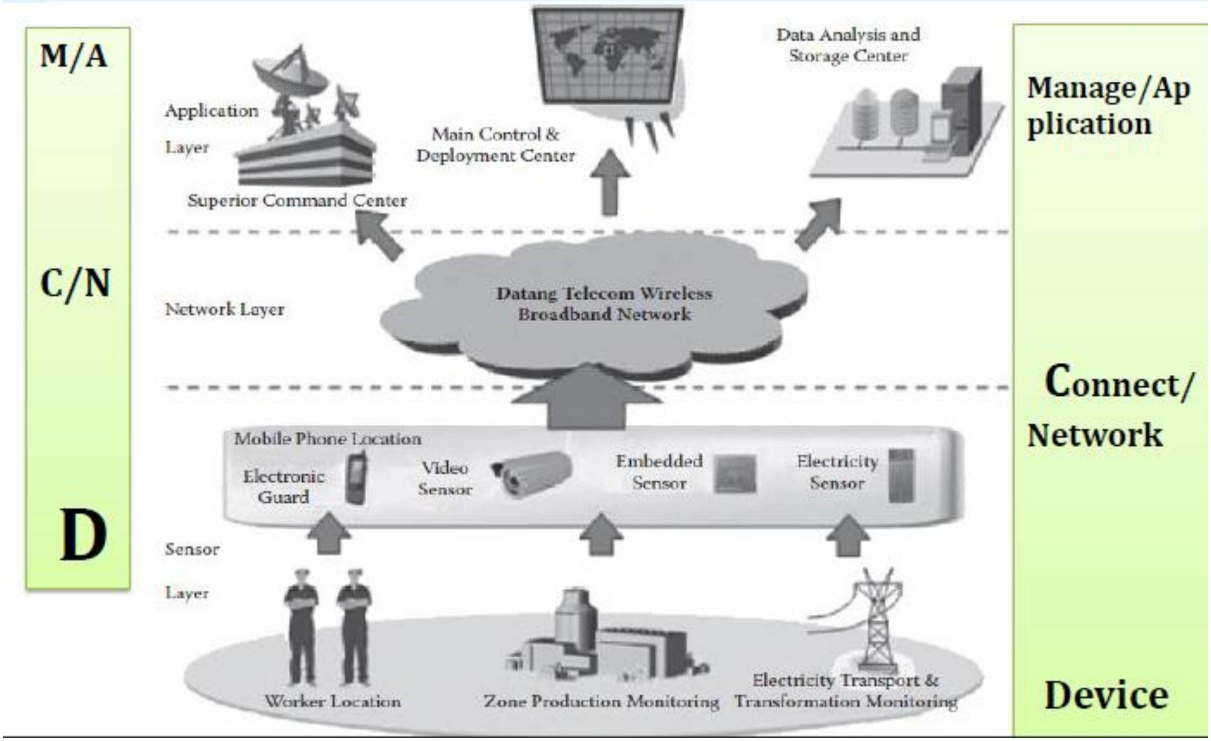
Internet of devices

Internet of Objects

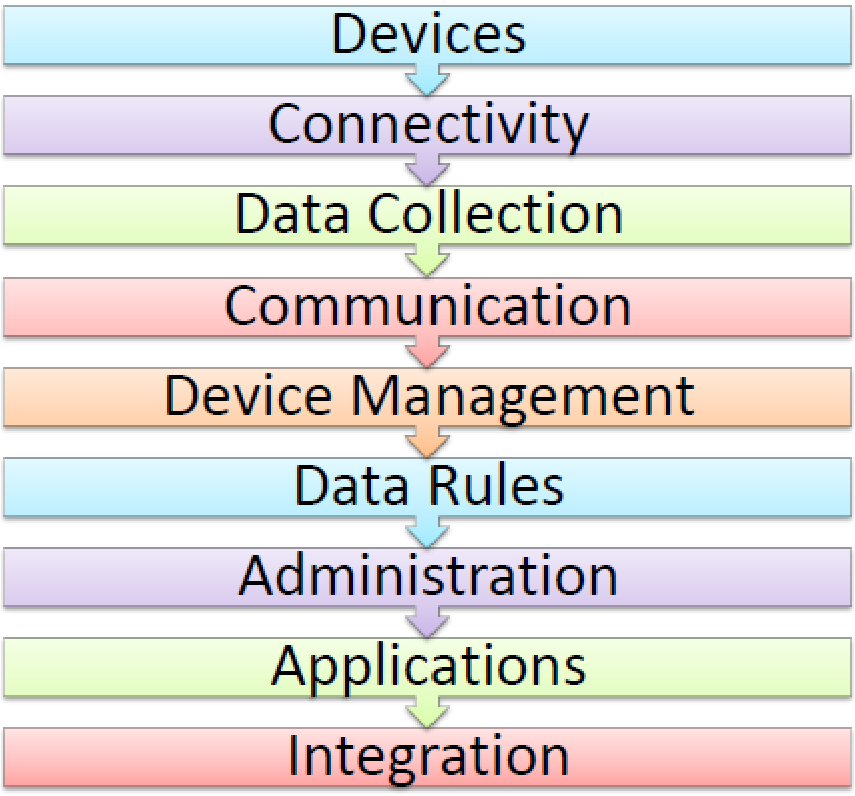
Internet of Transducer (Transmitter and receiver)

Internet of controllers

Three Layer architecture of IoT



Nine Layer architecture of IoT



Devices

Connectivity

Data Collection

Communication

Device Management

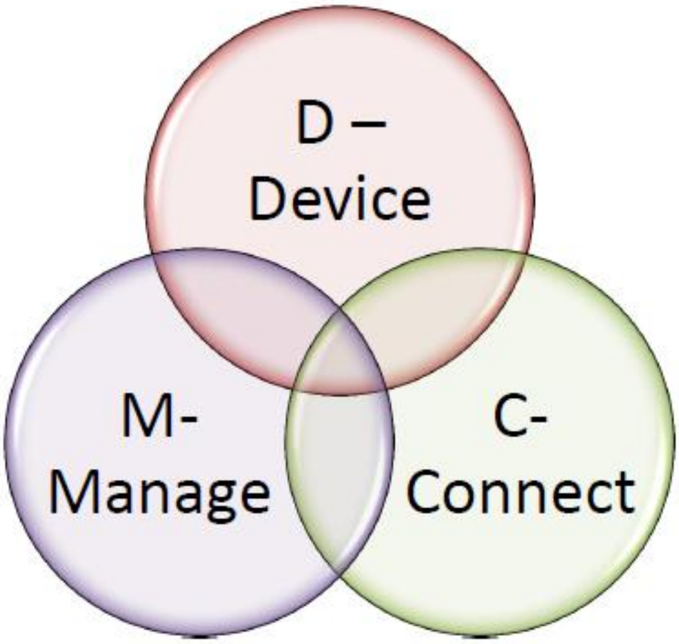
Data Rules

Administration

Applications

Integration

DCM



Devices : Things that Talk

Inherent Intelligent •Means Inbuilt Intelligent
•Eg. Washing Machines, ventilation, and air-conditioning, (HVAC) controllers

Enabled Intelligent
•Means need to be made intelligent

•RFID tagged devices

Sensors

•Perform **Input function**

•Device that **responds to a physical stimulus**, **measures** the physical stimulus quantity, and **converts it into** a signal, usually **electrical,** which can be read by an observer or by an instrument

•Also called **detector**

•A sensor is basically an **electrical device**. It could be an **M2M terminal**, an **RFID reader**, or a **SCADA meter**.

•A sensor can be very small and itself can be a trackable device

Perform Input function

•Device that responds to a physical stimulus, measures the physical stimulus quantity, and converts it into a signal, usually electrical, which can be read by an observer or by an instrument

Actuators

•Performs Output function

•RFID tagged devices

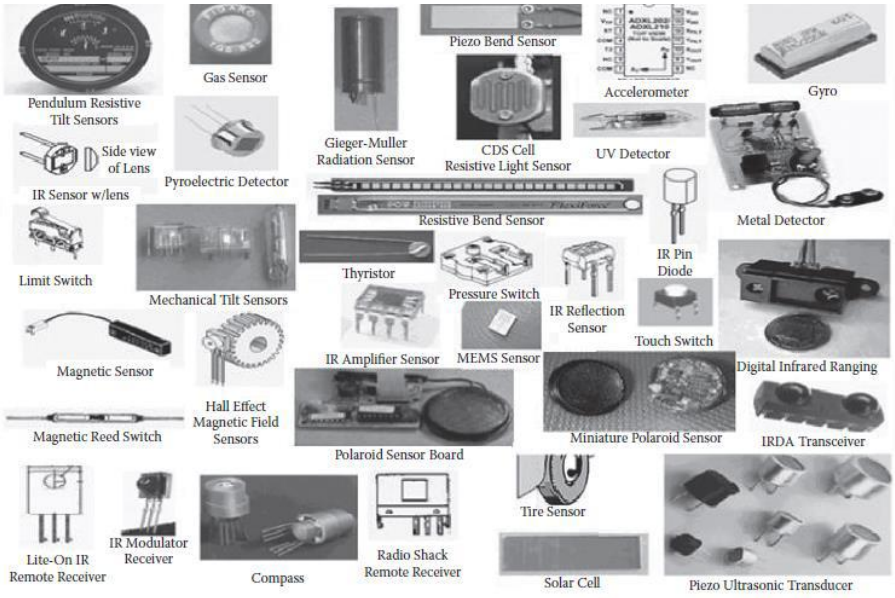
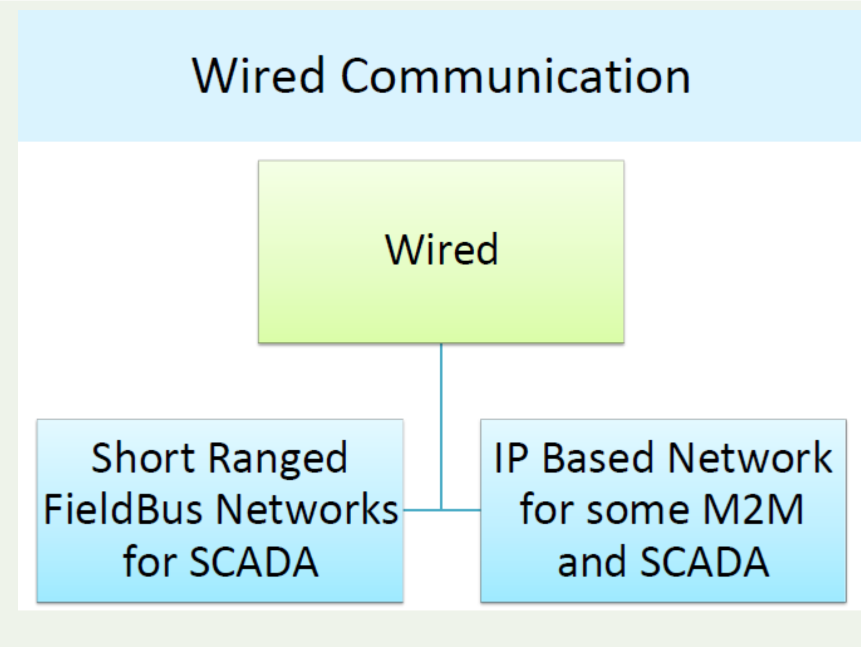


Figure 4.3 Examples of sensors.

Sensor Type and Example

Sensor Type	Example
Position, angle, displacement, distance, speed, acceleration	position sensor, Accelerometer, capacitive displacement sensor ….
Pressure	barometer, boost gauge, pressure sensor

Acoustic, sound,vibration	Geophone, hydrophone, lace sensor, microphone
Automotive, transportation	Air-fuel ratio meter, engine coolant temperature (ECT) sensor, parking sensors, ……..
Environment, weather, moisture, Humidity	leaf sensor, rain sensor, soil moisture sensor
Flow, fluid velocity	Gas meter, water meter
Optical, light, imaging, photon	Contact image sensor, infra-red sensor……..
Connect : Pervasive Network •The communications layer is the foundational infrastructure of IoT. •There are two major communication technologies: **wireless and wired** (or wireline). •When talking about **IoT, wireless communications** is the topic most of the times, because three (M2M, RFID, and WSN) of the four IoT pillars are based on wireless	



Wired Communication

Wired

Short Ranged FieldBus Networks for SCADA

IP Based Network for some M2M and SCADA

Wired Communication : FieldBus

•Field bus is the name of a family of **industrial computer network protocols** used **for real-time distributed** control of nodes.

•Standardized as **IEC 61158**

•**Field** refers to **geographical area** or contextual area in industry where IOT devices are distributed

•**Bus** refers to the **electrical medium** that carries **data** through it.

Field bus has Eight different protocol sets
–Type 1: Foundation field bus H1

–Type 2 ControlNet

–Type 3 PROFIBUS

–Type 4 P-Net

–Type 5 FOUNDATION field bus HSE (high-speed Ethernet)

–Type 6 SwiftNet (a protocol developed for Boeing, since withdrawn)

–Type 7 WorldFIP

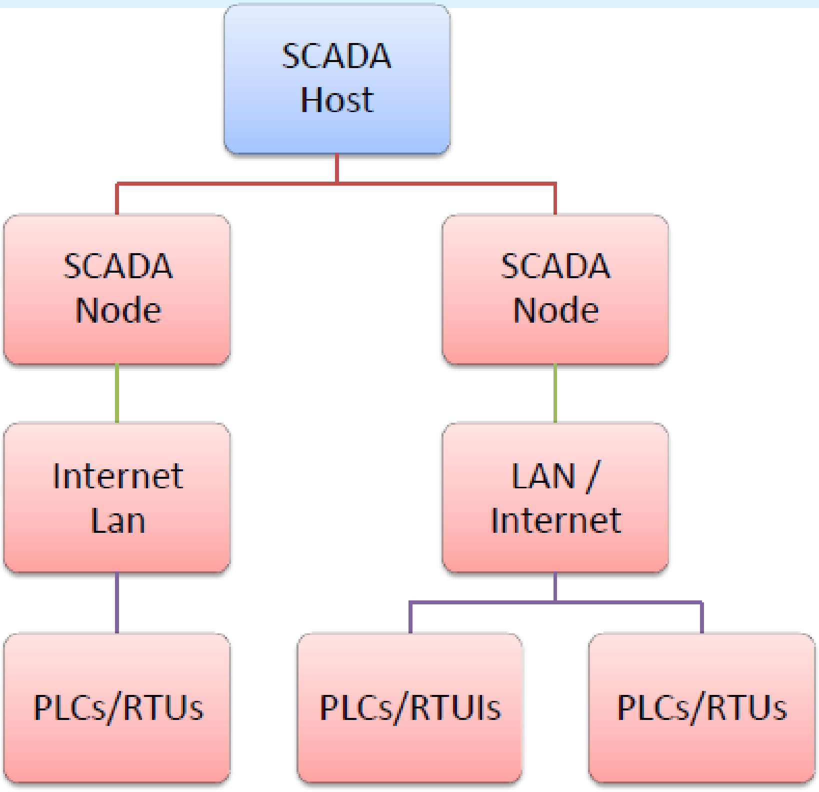
–Type 8 Interbus

Most use **Twisted pair or optical fiber** as a physical communication medium

•**Uses bidirectional communication** between the field instruments and HMI..

•Mainly **used for SCADA Based Applications**

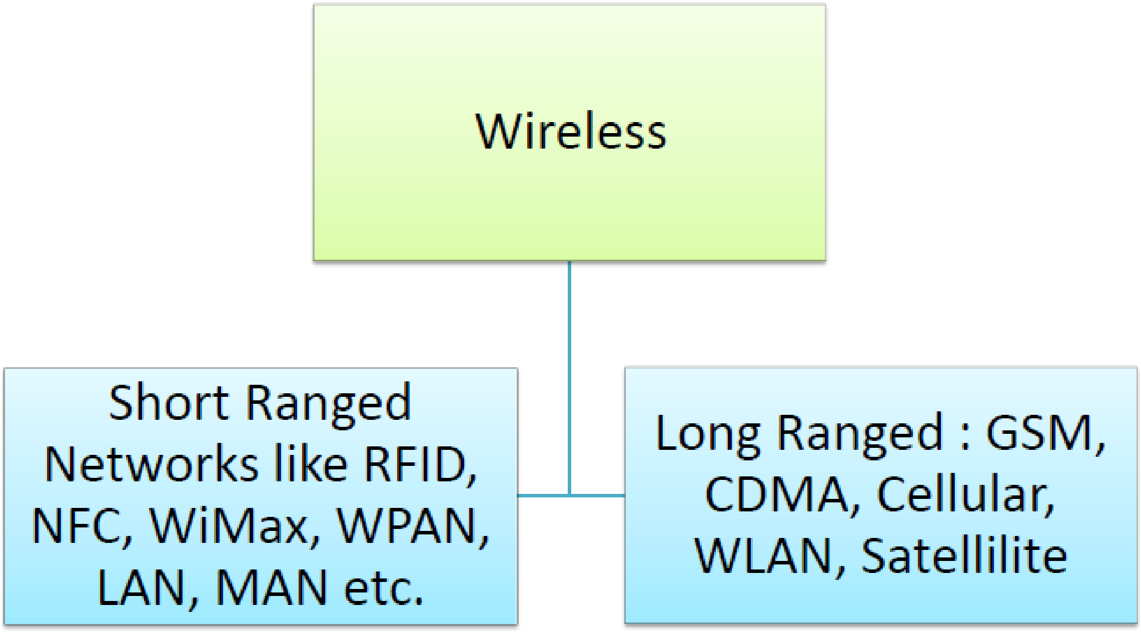
Wired Communication : FieldBus Communication



Examples for FieldBus

- Automatic meter Reading : ModBus
- Home Automation : Bacnet
- Power Automation : Profibus

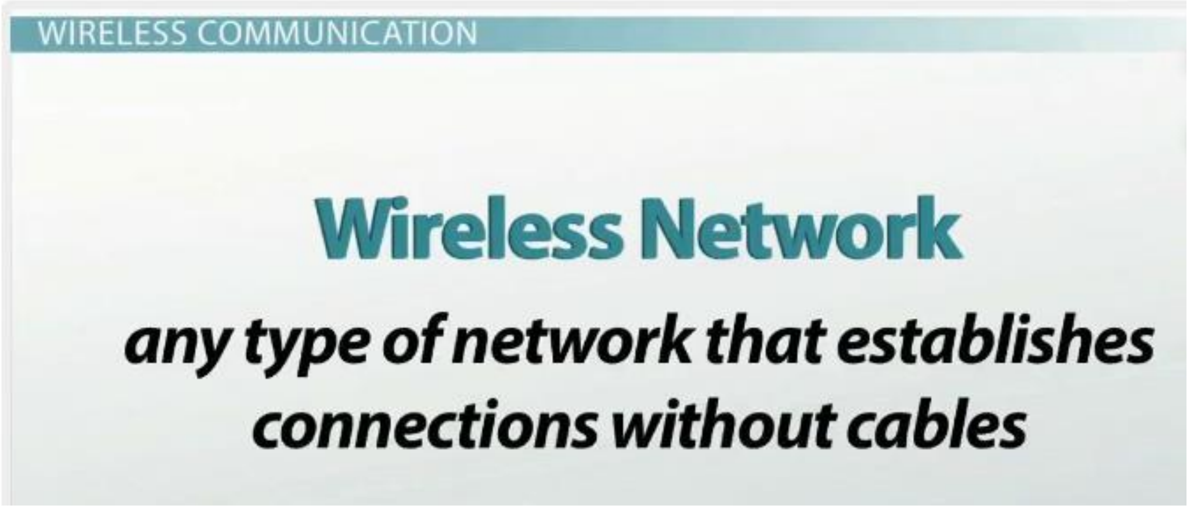
Wireless Communication



Wireless

Short Ranged Networks like RFID, NFC, WiMax, WPAN, LAN, MAN etc.

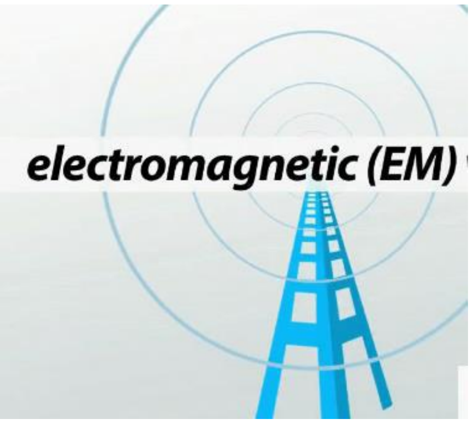
Long Ranged : GSM, CDMA, Cellular, WLAN, Satellilite



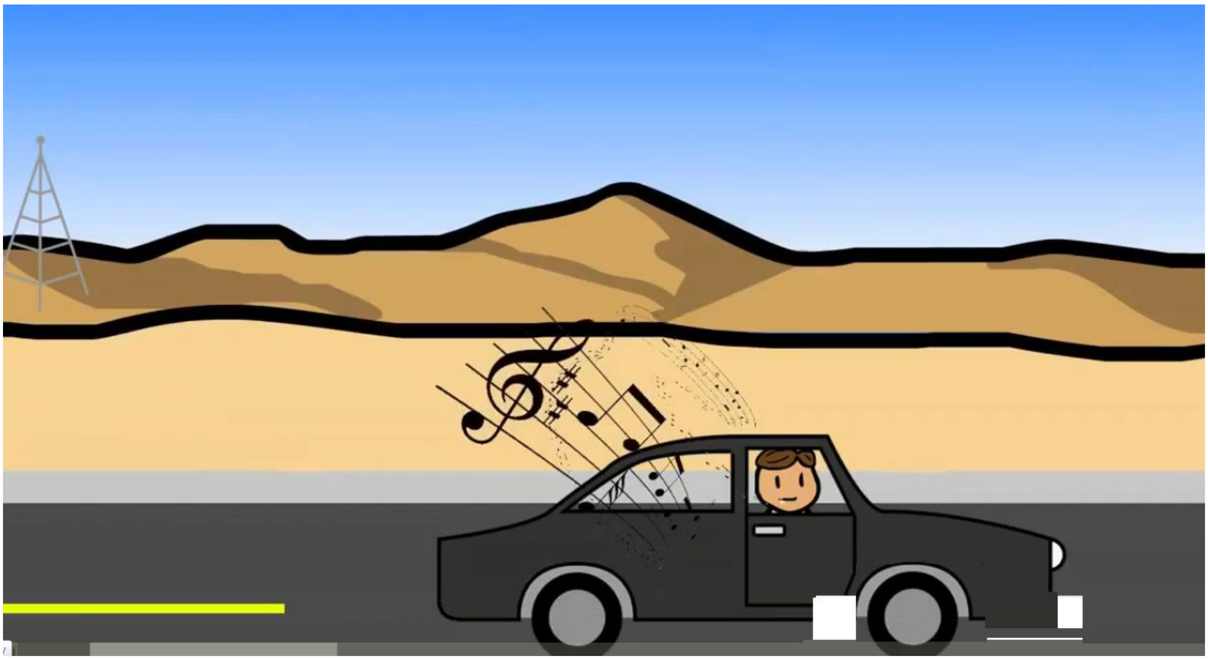
WIRELESS COMMUNICATION

Wireless Network

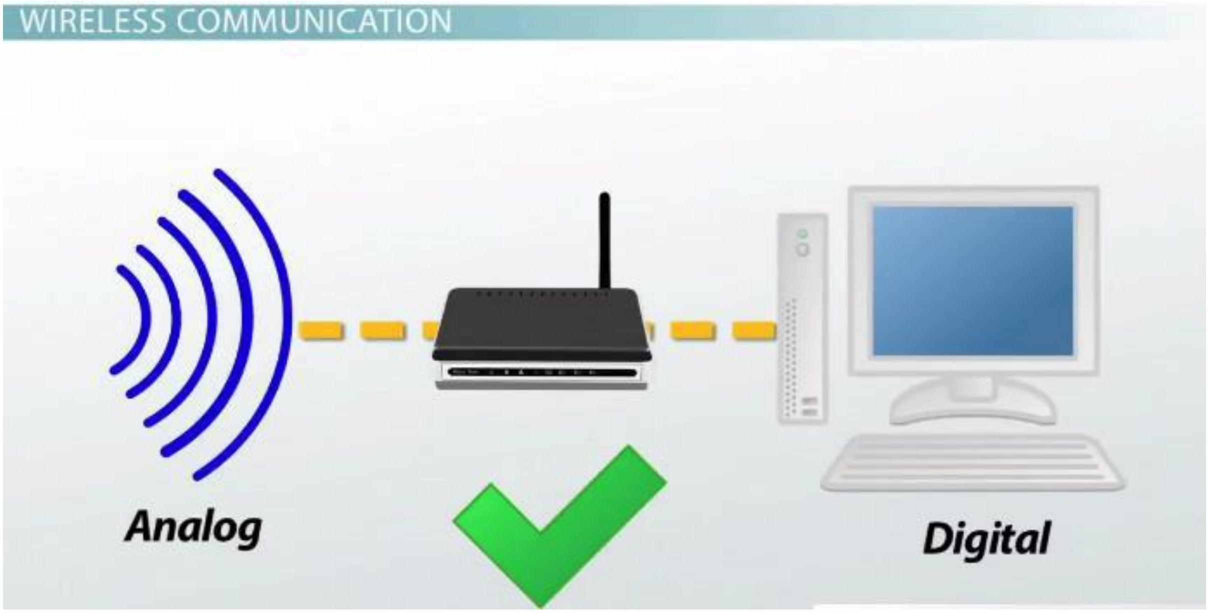
any type of network that establishes connections without cables



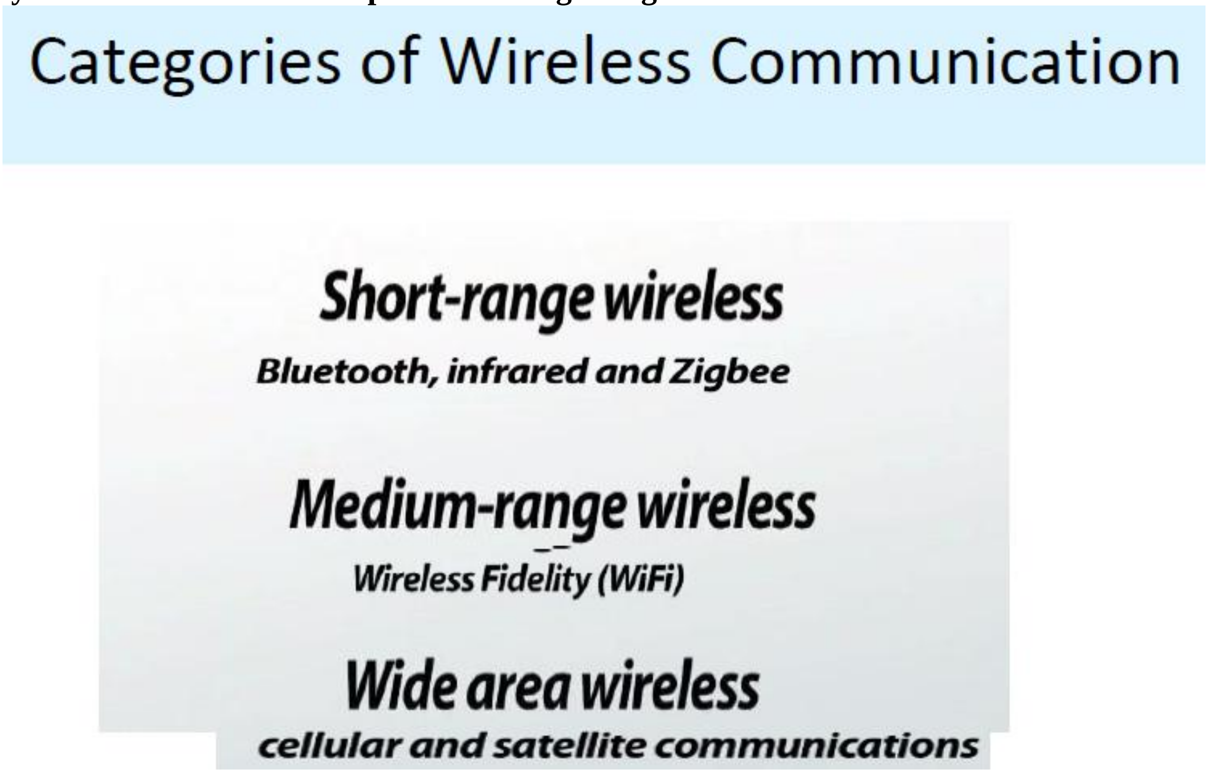
Wireless Communication **use electromagnetic waves** that travel through the air
•Simplest example of this is : **Radio**



When we listen radio in a moving car we are actually receiving radio waves which is one type of electromagnetic wave.



Electromagnetic waves are analog whereas information in computer is digital; **wireless systems therefore need adapter for analog to digital conversion.**



Categories of Wireless Communication

Short-range wireless

Bluetooth, infrared and Zigbee

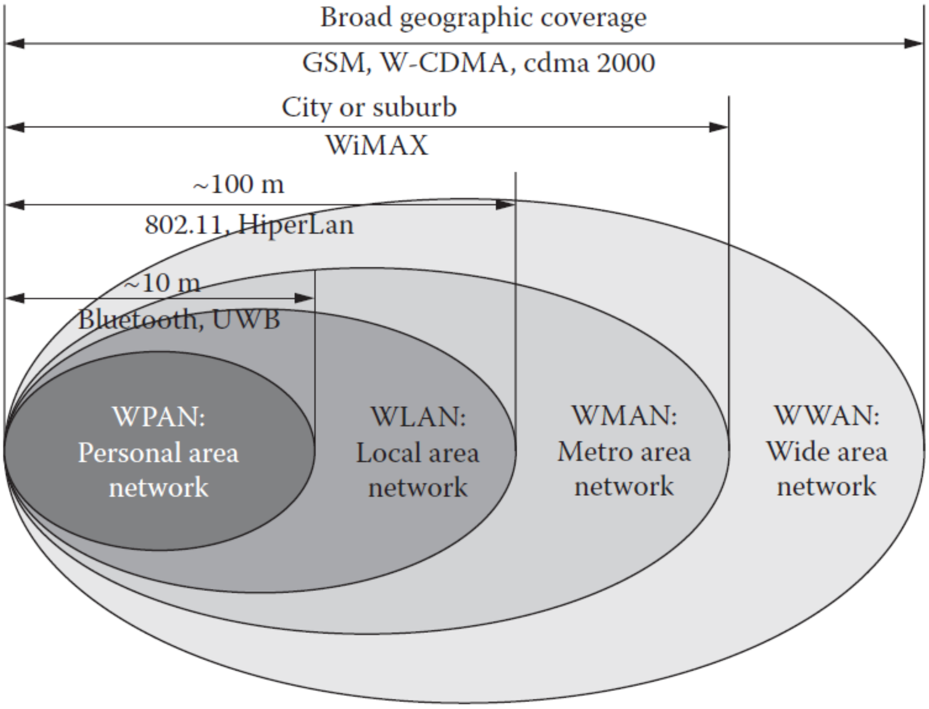
Medium-range wireless

Wireless Fidelity (WiFi)

Wide area wireless

cellular and satellite communications

Short and long range wireless n/w's



Standards of Wireless n/w

RFID and NFC are parts of WPAN.

•6LowPAN (IPv6 over low power wireless personal area networks)

•BSN (Body Sensor n/w)

•HomeIR: wireless IR home networking

•HomeRF: wireless RF home networking

Communication Mechanisms of Wireless n/w

Orthogonal frequency division multiplexing (OFDM) and orthogonal frequency division multiple access(OFDMA)

•Ad hoc sensor network

•Software defined radio (SDR)

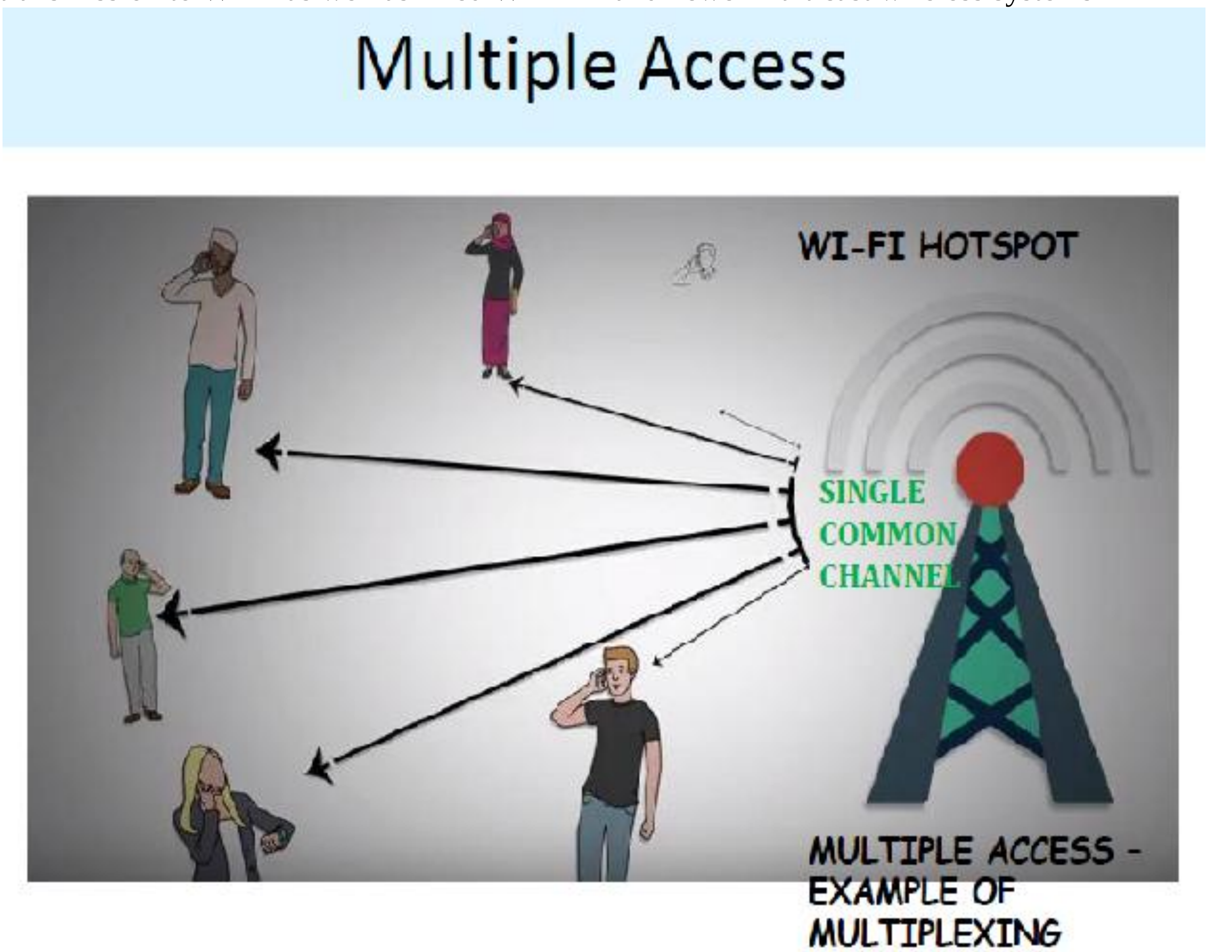
•Cognitive radio (CR)

OFDM/OFDMA

These are two different variants of the same **broadband wireless air interface**.

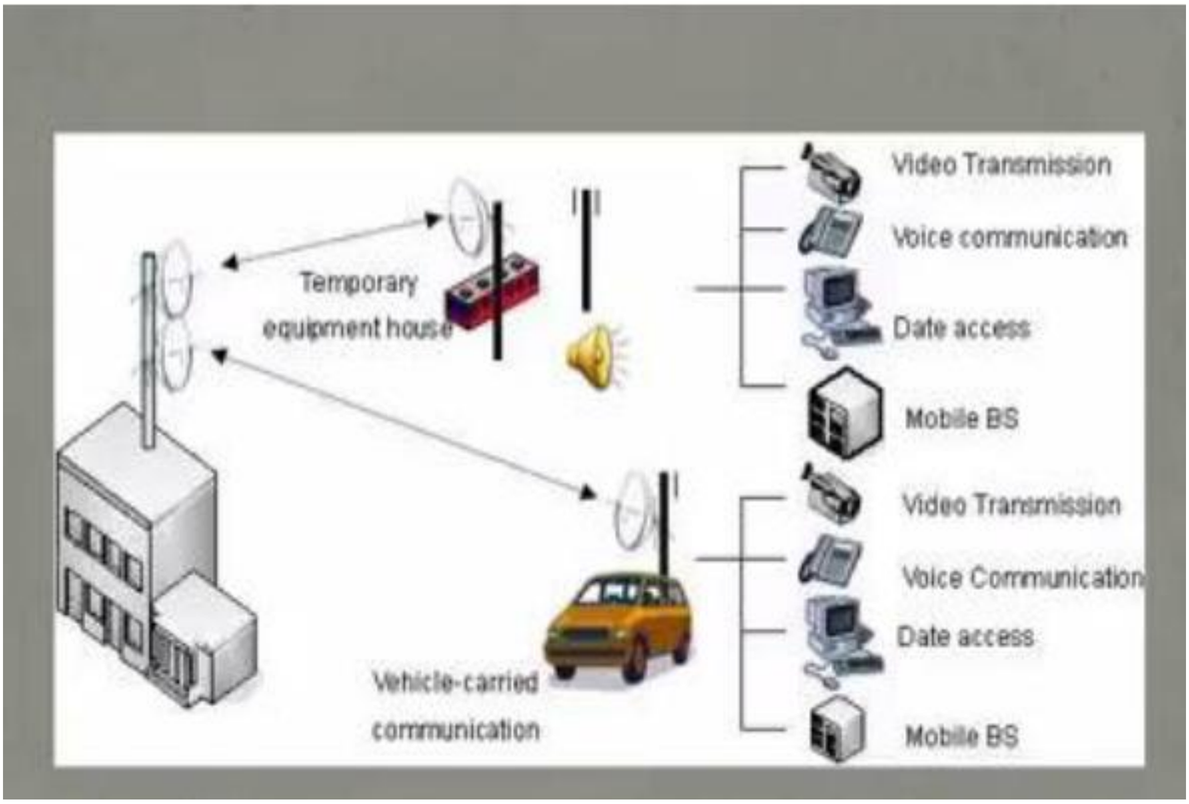
•**Long-Term Evolution (*LTE*)** the standard for **high-speed wireless communication** used for mobile devices and data terminal is an **OFDMA-based** technology **standardized in** *3rd Generation Partnership Project* (**3GPP**).

•Typically occupy roaming, fixed, and one-way transmission standards, ranging from TV transmission to Wi-Fi as well as fixed WiMAX and newer multicast wireless systems



Multiple Access

Adhoc Sensor n/w



Using Software to replace Hardware to modulate the signals.

•**Hardware is Still needed for RF** front end and **ADC** (Analog to Digital Converter)/ DAC (Digital to Analog Converter)

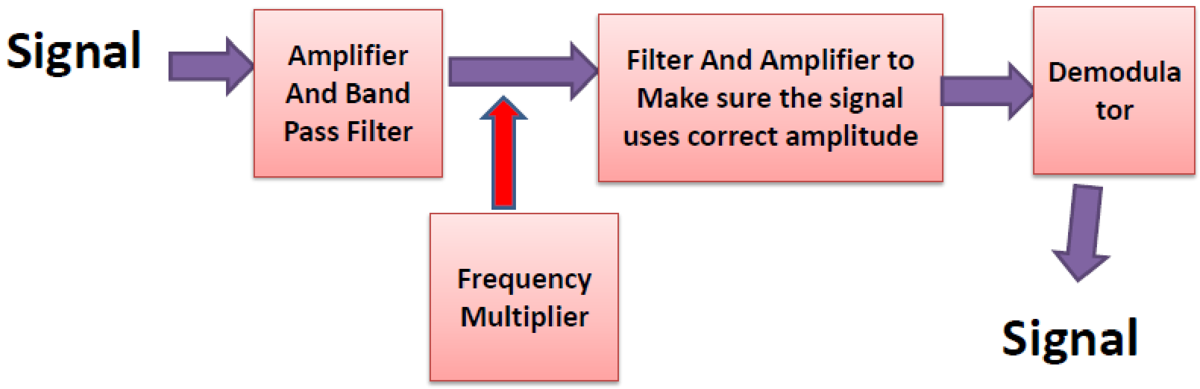
•Advantage : We can receive many signals with single piece of hardware Software Defined Radio

Using Software to replace Hardware to modulate the signals.

•**Hardware is Still needed for RF** front end and **ADC** (Analog to Digital Converter)/ DAC (Digital to Analog Converter)

•Advantage : We can receive many signals with single piece of hardware

H/w Radio Receiver



Signal

Amplifier And Band Pass Filter

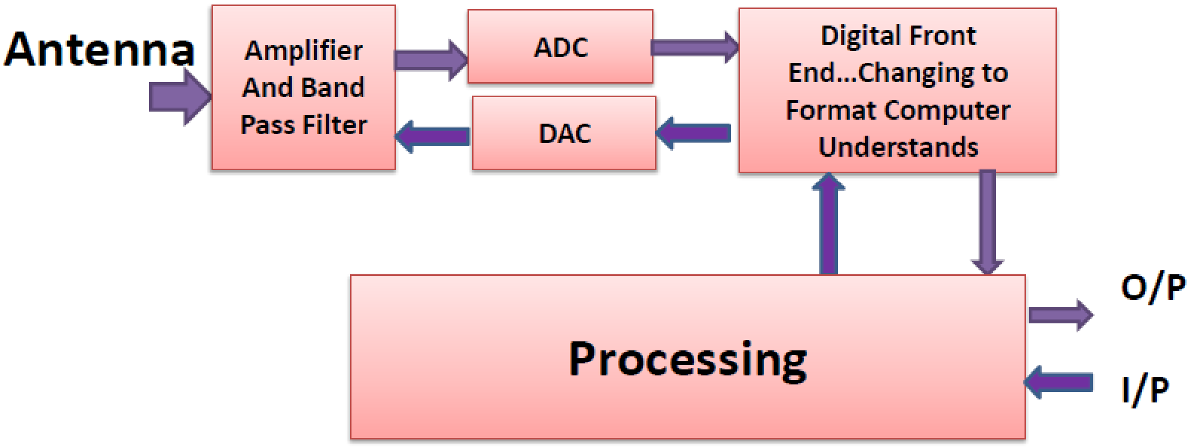
Filter And Amplifier to Make sure the signal uses correct amplitude

Demodulator

Frequency Multiplier

Signal

Software Defined Radio



Antenna

Amplifier And Band Pass Filter

ADC

DAC

Digital Front End...Changing to Format Computer Understands

Processing

O/P

I/P

Software Defined Radio :

SDR is the result of an evolutionary process from purely hardware-based equipment to fully software-based equipment.

•All functions, modes, and applications, such as transmit frequencies, modulation type, and other RF parameters, can be configured and reconfigured by software

•Software-defined refers to the use of software processing within the radio system or device to implement operating (but not control) functions

Cognitive Radio

CR is a form of wireless communication in which a transceiver can intelligently detect which communication channels are in use and which are not, and instantly move into vacant channels while avoiding occupied ones.

•This optimizes the use of available RF spectrum while minimizing interference to other users

•SDR is a required basic platform on which to build a CR.

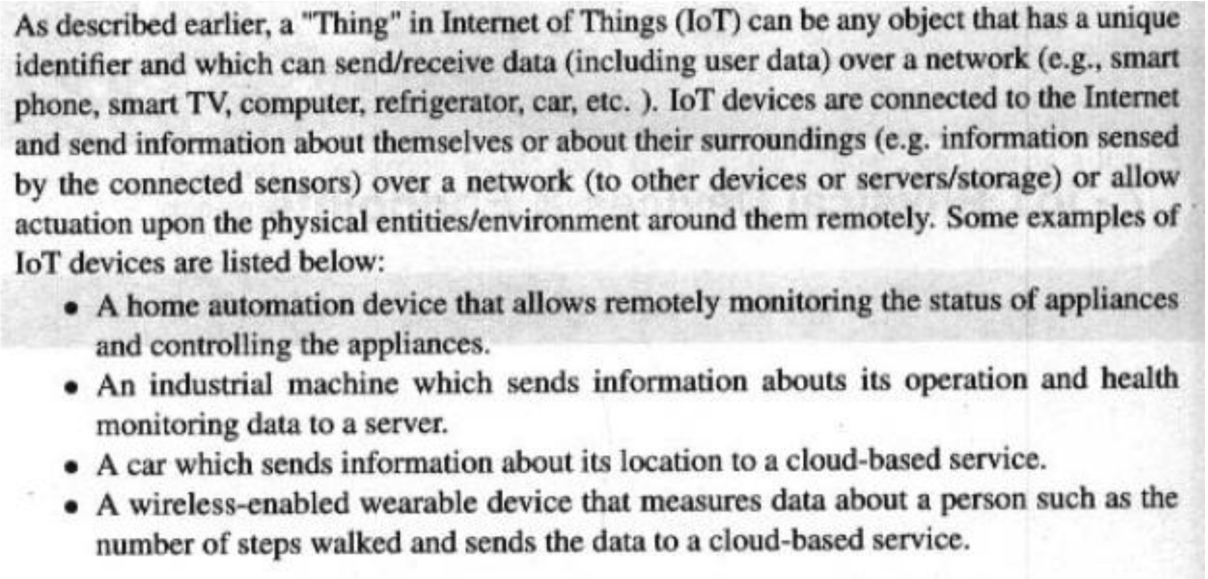
•Cognitive radios are radios that are aware of their environment and internal state and can make decisions about their radio operating behavior based on that information and predefined objective.

Satellite IoT

https://www.youtube.com/watch?v=hXa3bTcIGPU

Physical Devices

What is an IoT Device



As described earlier, a "Thing" in Internet of Things (IoT) can be any object that has a unique identifier and which can send/receive data (including user data) over a network (e.g., smart phone, smart TV, computer, refrigerator, car, etc.). IoT devices are connected to the Internet and send information about themselves or about their surroundings (e.g. information sensed by the connected sensors) over a network (to other devices or servers/storage) or allow actuation upon the physical entities/environment around them remotely. Some examples of IoT devices are listed below:

- A home automation device that allows remotely monitoring the status of appliances and controlling the appliances.
- An industrial machine which sends information abouts its operation and health monitoring data to a server.
- A car which sends information about its location to a cloud-based service.
- A wireless-enabled wearable device that measures data about a person such as the number of steps walked and sends the data to a cloud-based service.

Basic building blocks of an IoT Device

An IoT device can consist of a number of modules based on functional attributes, such as:

- Sensing: Sensors can be either on-board the IoT device or attached to the device. IoT device can collect various types of information from the on-board or attached sensors such as temperature, humidity, light intensity, etc. The sensed information can be communicated either to other devices or cloud-based servers/storage.
- Actuation: IoT devices can have various types of actuators attached that allow taking actions upon the physical entities in the vicinity of the device. For example, a relay switch connected to an IoT device can turn an appliance on/off based on the commands sent to the device.
- Communication: Communication modules are responsible for sending collected data to other devices or cloud-based servers/storage and receiving data from other devices and commands from remote applications.
- Analysis & Processing: Analysis and processing modules are responsible for making sense of the collected data.

The representative IoT device used for the examples in this book is the widely used single-board mini computer called Raspberry Pi (explained in later sections). The use of Raspberry Pi is intentional since these devices are widely accessible, inexpensive, and available from multiple vendors. Furthermore, extensive information is available on their programming and use both on the Internet and in other textbooks. The principles we teach in this book are just as applicable to other (including proprietary) IoT endpoints, in addition to Raspberry Pi. Before we look at the specifics of Raspberry Pi, let us first look at the building blocks of a generic single-board computer (SBC) based IoT device.

Figure 7.1 shows a generic block diagram of a single-board computer (SBC) based IoT device that includes CPU, GPU, RAM, storage and various types of interfaces and peripherals.

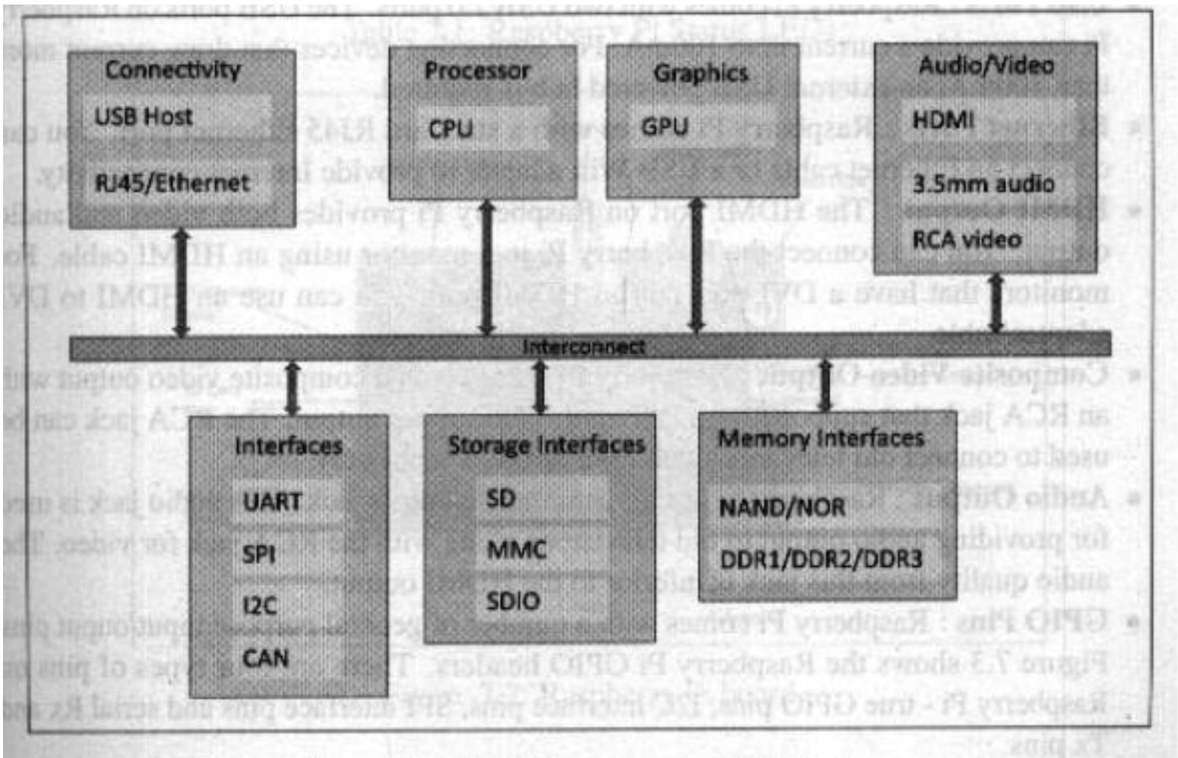


Figure 7.1: Block diagram of an IoT Device

Exemplary Device: Raspberry Pi

Raspberry Pi [104] is a low-cost mini-computer with the physical size of a credit card. Raspberry Pi runs various flavors of Linux and can perform almost all tasks that a normal desktop computer can do. In addition to this, Raspberry Pi also allows interfacing sensors and actuators through the general purpose I/O pins. Since Raspberry Pi runs Linux operating system, it supports Python "out of the box".

About the Board

Figure 7.2 shows the Raspberry Pi board with the various components/peripherals labeled.

- **Processor & RAM** : Raspberry Pi is based on an ARM processor. The latest version of Raspberry Pi (Model B, Revision 2) comes with 700 MHz Low Power ARM1176JZ-F processor and 512 MB SDRAM.
- **USB Ports** : Raspberry Pi comes with two USB 2.0 ports. The USB ports on Raspberry Pi can provide a current upto 100mA. For connecting devices that draw current more than 100mA, an external USB powered hub is required.
- **Ethernet Ports** : Raspberry Pi comes with a standard RJ45 Ethernet port. You can connect an Ethernet cable or a USB Wifi adapter to provide Internet connectivity.
- **HDMI Output** : The HDMI port on Raspberry Pi provides both video and audio output. You can connect the Raspberry Pi to a monitor using an HDMI cable. For monitors that have a DVI port but no HDMI port, you can use an HDMI to DVI adapter/cable.
- **Composite Video Output** : Raspberry Pi comes with a composite video output with an RCA jack that supports both PAL and NTSC video output. The RCA jack can be used to connect old televisions that have an RCA input only.
- **Audio Output** : Raspberry Pi has a 3.5mm audio output jack. This audio jack is used for providing audio output to old televisions along with the RCA jack for video. The audio quality from this jack is inferior to the HDMI output.
- **GPIO Pins** : Raspberry Pi comes with a number of general purpose input/ouput pins. Figure 7.3 shows the Raspberry Pi GPIO headers. There are four types of pins on Raspberry Pi - true GPIO pins, I2C interface pins, SPI interface pins and serial Rx and Tx pins.
- **Display Serial Interface (DSI)** : The DSI interface can be used to connect an LCD panel to Raspberry Pi.
- **Camera Serial Interface (CSI)** : The CSI interface can be used to connect a camera module to Raspberry Pi.
- **Status LEDs** : Raspberry Pi has five status LEDs. Table 7.1 lists Raspberry Pi status LEDs and their functions.
- **SD Card Slot** : Raspberry Pi does not have a built in operating system and storage. You can plug-in an SD card loaded with a Linux image to the SD card slot. Appendix-A provides instructions on setting up New Out-of-the-Box Software (NOOBS) on Raspberry Pi. You will require atleast an 8GB SD card for setting up NOOBS.
- **Power Input** : Raspberry Pi has a micro-USB connector for power input.

Status LED	Function
ACT	SD card access
PWR	3.3V Power is present
FDX	Full duplex LAN connected
LNK	Link/Network activity
100	100 Mbit LAN connected

Table 7.1: Raspberry Pi Status LEDs

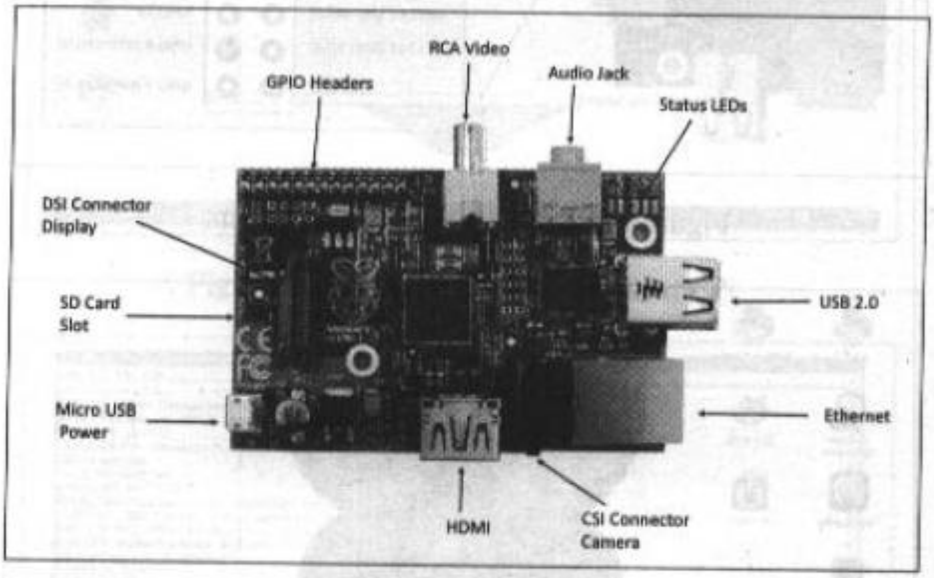


Figure 7.2: Raspberry Pi board

Raspberry Pi Interfaces

Raspberry Pi has serial, SPI and I2C interfaces for data transfer as shown in Figure 7.3.

Serial

The serial interface on Raspberry Pi has receive (Rx) and transmit (Tx) pins for communication with serial peripherals.

SPI

Serial Peripheral Interface (SPI) is a synchronous serial data protocol used for communicating with one or more peripheral devices. In an SPI connection, there is one master device and one or more peripheral devices. There are five pins on Raspberry Pi for SPI interface:

- **MISO (Master In Slave Out)** : Master line for sending data to the peripherals.
- **MOSI (Master Out Slave In)** : Slave line for sending data to the master.
- **SCK (Serial Clock)** : Clock generated by master to synchronize data transmission
- **CE0 (Chip Enable 0)** : To enable or disable devices.
- **CE0 (Chip Enable 1)** : To enable or disable devices.

I2C

The I2C interface pins on Raspberry Pi allow you to connect hardware modules. I2C interface allows synchronous data transfer with just two pins - SDA (data line) and SCL (clock line).

Programming Raspberry Pi with Python

In this section you will learn how to get started with developing Python programs on Raspberry Pi. Raspberry Pi runs Linux and supports Python out of the box. Therefore, you can run any Python program that runs on a normal computer. However, it is the general purpose input/output capability provided by the GPIO pins on Raspberry Pi that makes it useful device for Internet of Things. You can interface a wide variety of sensor and actuators with Raspberry Pi using the GPIO pins and the SPI, I2C and serial interfaces. Input from the sensors connected to Raspberry Pi can be processed and various actions can be taken, for instance, sending data to a server, sending an email, triggering a relay switch.

Controlling LED with Raspberry Pi

Let us start with a basic example of controlling an LED from Raspberry Pi. Figure 7.9 shows the schematic diagram of connecting an LED to Raspberry Pi. Box 7.1 shows how to turn the LED on/off from command line. In this example the LED is connected to GPIO pin 18. You can connect the LED to any other GPIO pin as well.

Box 7.2 shows a Python program for blinking an LED connected to Raspberry Pi every second. The program uses the RPi. GPIO module to control the GPIO on Raspberry Pi. In this program we set pin 18 direction to output and then write *True/False* alternatively after a delay of one second.

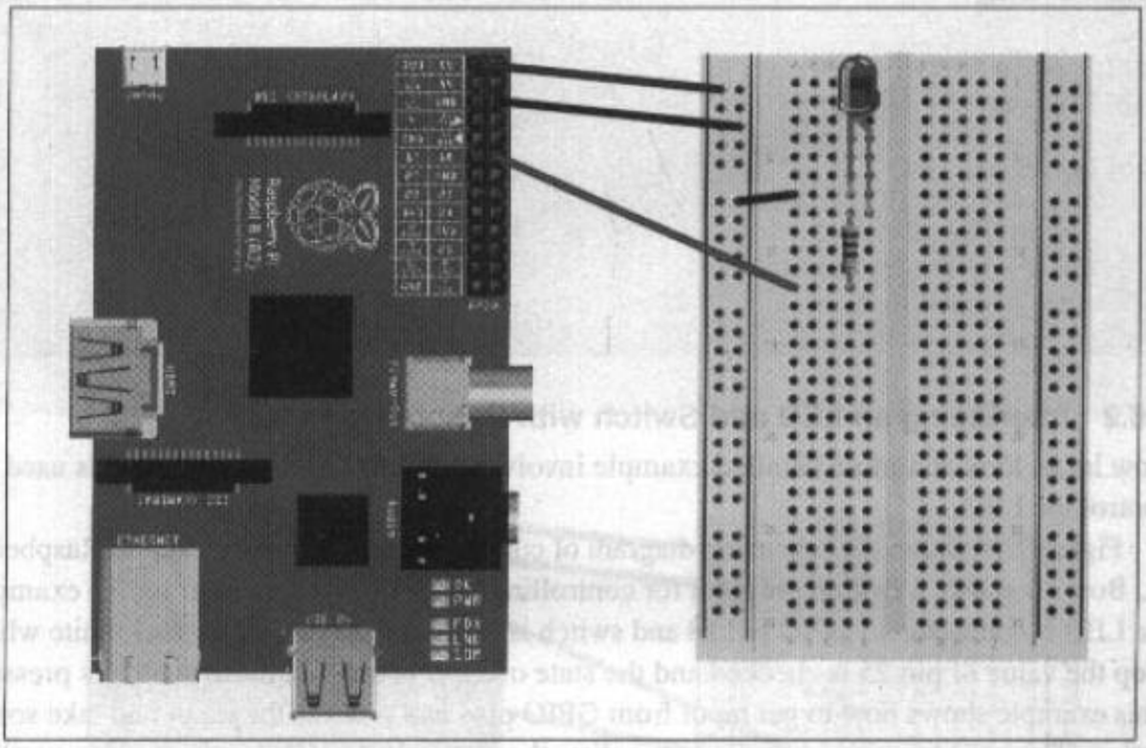


Figure 7.9: Controlling LED with Raspberry Pi

■ Box 7.1: Switching LED on/off from Raspberry Pi console

```
$echo 18 > /sys/class/gpio/export
$cd /sys/class/gpio/gpio18

#Set pin 18 direction to out
$echo out > direction

#Turn LED on
$echo 1 > value
```

```
#Turn LED off
$echc 0 > value
```

■ Box 7.2: Python program for blinking LED

```
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)
GPIO.setup(18, GPIO.OUT)

while True:
   GPIO.output(18, True)
   time.sleep(1)
   GPIO.output(18, False)
   time.sleep(1)
```

Interfacing an LED and Switch with Raspberry Pi

Now let us look at a more detailed example involving an LED and a switch that is used to control the LED.

Figure 7.10 shows the schematic diagram of connecting an LED and switch to Raspberry Pi. Box 7.3 shows a Python program for controlling an LED with a switch. In this example the LED is connected to GPIO pin 18 and switch is connected to pin 25. In the infinite while loop the value of pin 25 is checked and the state of LED is toggled if the switch is pressed. This example shows how to get input from GPIO pins and process the input and take some action. The action in this example is toggling the state of an LED. Let us look at another example, in which the action is an email alert. Box 7.4 shows a Python program for sending an email on switch press. Note that the structure of this program is similar to the program in Box 7.3. This program uses the Python SMTP library for sending an email when the switch connected to Raspberry Pi is pressed.

■ Box 7.3: Python program for controlling an LED with a switch

```
from time import sleep
import RPi.GPIO as GPIO

GPIO.setmode(GPIO.BCM)
```

```
#Switch Pin
GPIO.setup(25, GPIO.IN)

#LED Pin
GPIO.setup(18, GPIO.OUT)

state=false

def toggleLED(pin):
   state = not state
   GPIO.output(pin, state)

while True:
   try:
      if (GPIO.input(25) == True):
         toggleLED(pin)
      sleep(.01)
   except KeyboardInterrupt:
      exit()
```

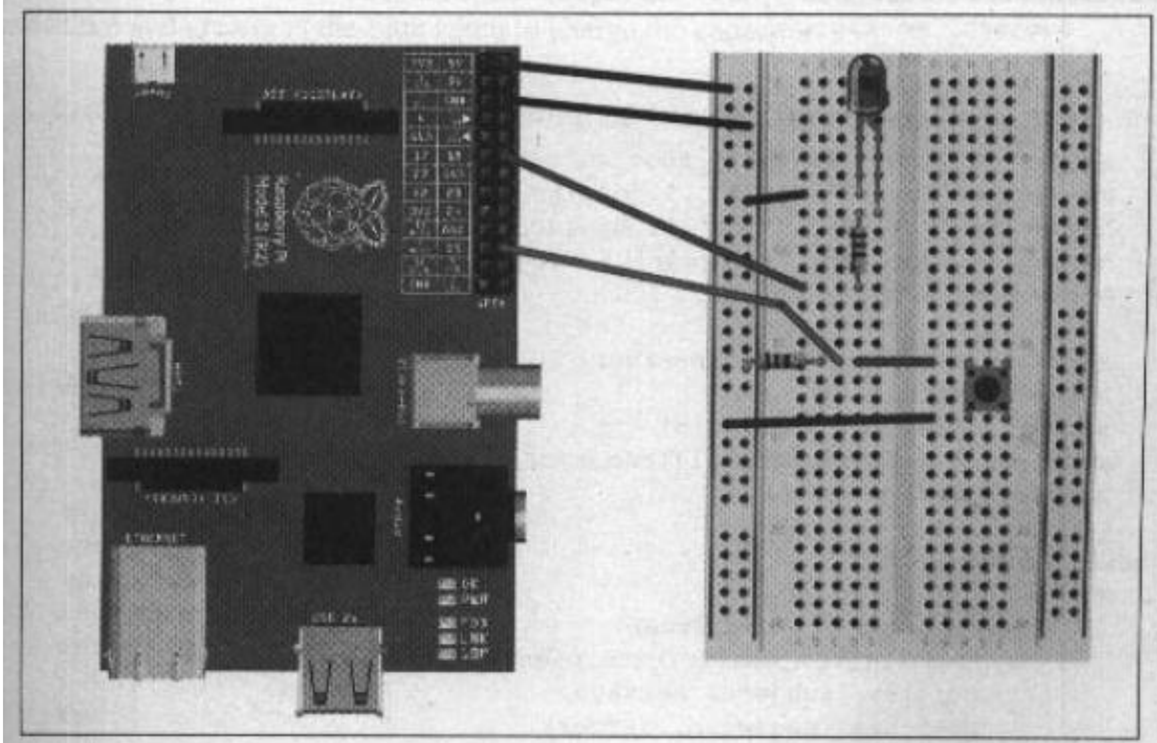


Figure 7.10: Interfacing LED and switch with Raspberry Pi

Interfacing a Light Sensor (LDR) with Raspberry Pi

So far you have learned how to interface LED and switch with Raspberry Pi. Now let us look at an example of interfacing a Light Dependent Resistor (LDR) with Raspberry Pi and turning an LED on/off based on the light-level sensed.

Figure 7.11 shows the schematic diagram of connecting an LDR to Raspberry Pi. Connect one side of LDR to 3.3V and other side to a 1μF capacitor and also to a GPIO pin (pin 18 in this example). An LED is connected to pin 18 which is controlled based on the light-level sensed.

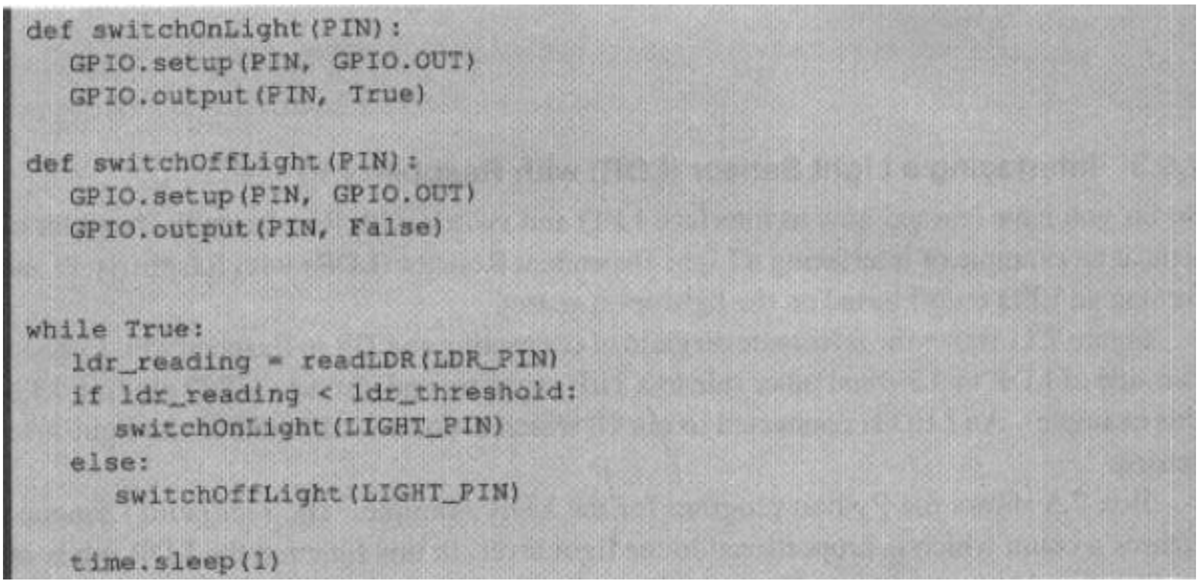
Box 7.5 shows the Python program for the LDR example. The *readLDR*() function returns a count which is proportional to the light level. In this function the LDR pin is set to output and low and then to input. At this point the capacitor starts charging through the resistor (and a counter is started) until the input pin reads high (this happens when capacitor voltage becomes greater than 1.4V). The counter is stopped when the input reads high. The final count is proportional to the light level as greater the amount of light, smaller is the LDR resistance and greater is the time taken to charge the capacitor.

■ Box 7.5: Python program for switching LED/Light based on reading LDR reading

```
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)
ldr_threshold = 1000
LDR_PIN = 18
LIGHT_PIN = 25

def readLDR(PIN):
   reading=0
   GPIO.setup(LIGHT_PIN, GPIO.OUT)
   GPIO.output(PIN, False)
   time.sleep(0.1)
   GPIO.setup(PIN, GPIO.IN)
   while (GPIO.input(PIN)==False):
      reading=reading+1
   return reading
```



```
def switchOnLight(PIN):
   GPIO.setup(PIN, GPIO.OUT)
   GPIO.output(PIN, True)

def switchOffLight(PIN):
   GPIO.setup(PIN, GPIO.OUT)
   GPIO.output(PIN, False)


while True:
   ldr_reading = readLDR(LDR_PIN)
   if ldr_reading < ldr_threshold:
      switchOnLight(LIGHT_PIN)
   else:
      switchOffLight(LIGHT_PIN)

   time.sleep(1)
```

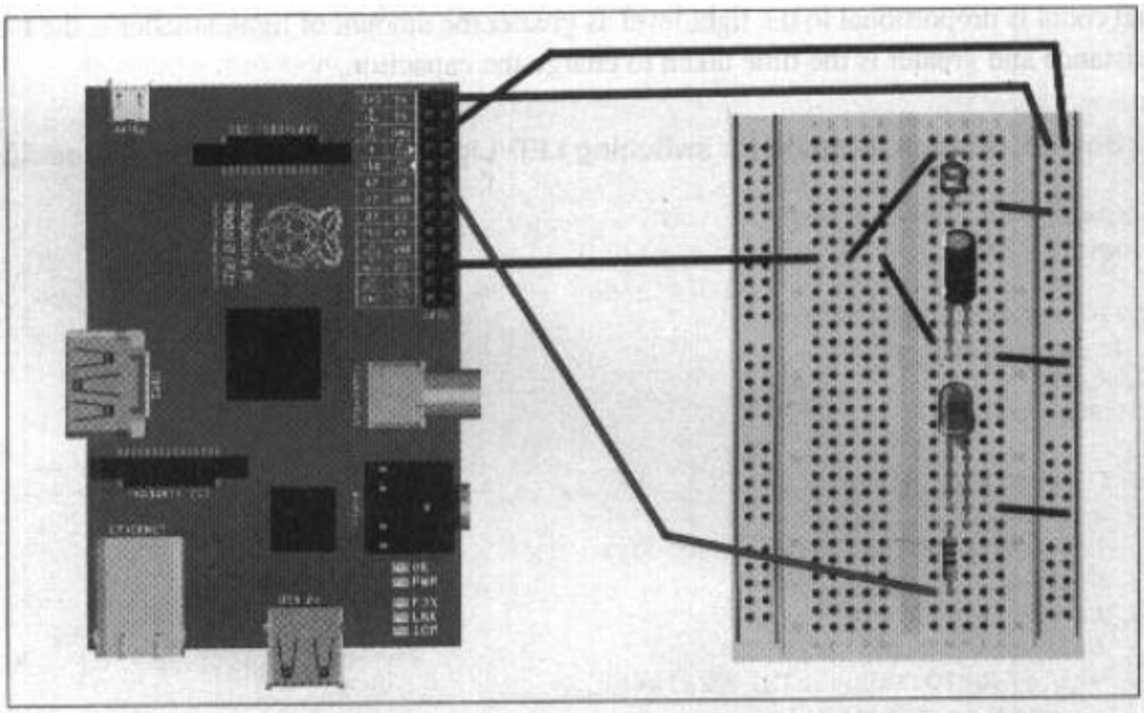


Figure 7.11: Interfacing LDR with Raspberry Pi

Other IoT Devices

Let us look at single-board mini-computers which are alternatives to Raspberry Pi. Table 7.3 provides a comparison of some single-board mini-computers that can be used for IoT.

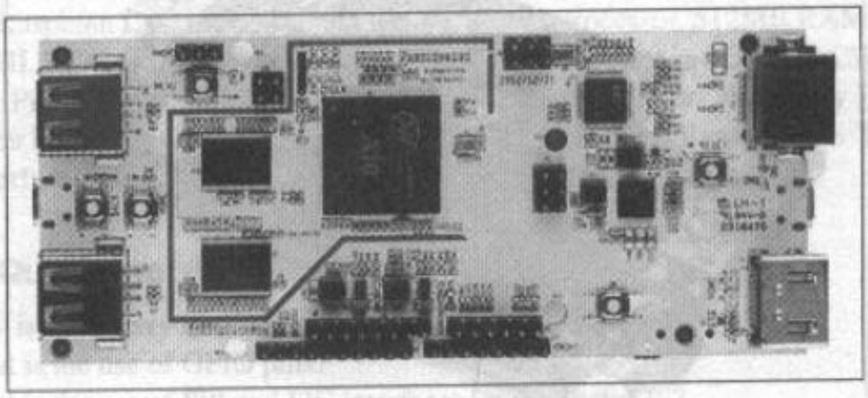


Figure 7.12: pcDuino

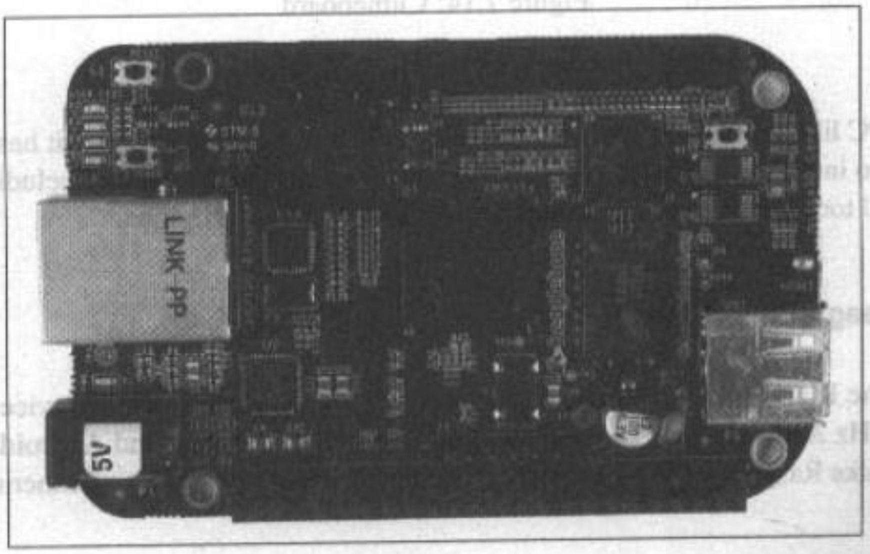


Figure 7.13: Beaglebone Black

BeagleBone Black

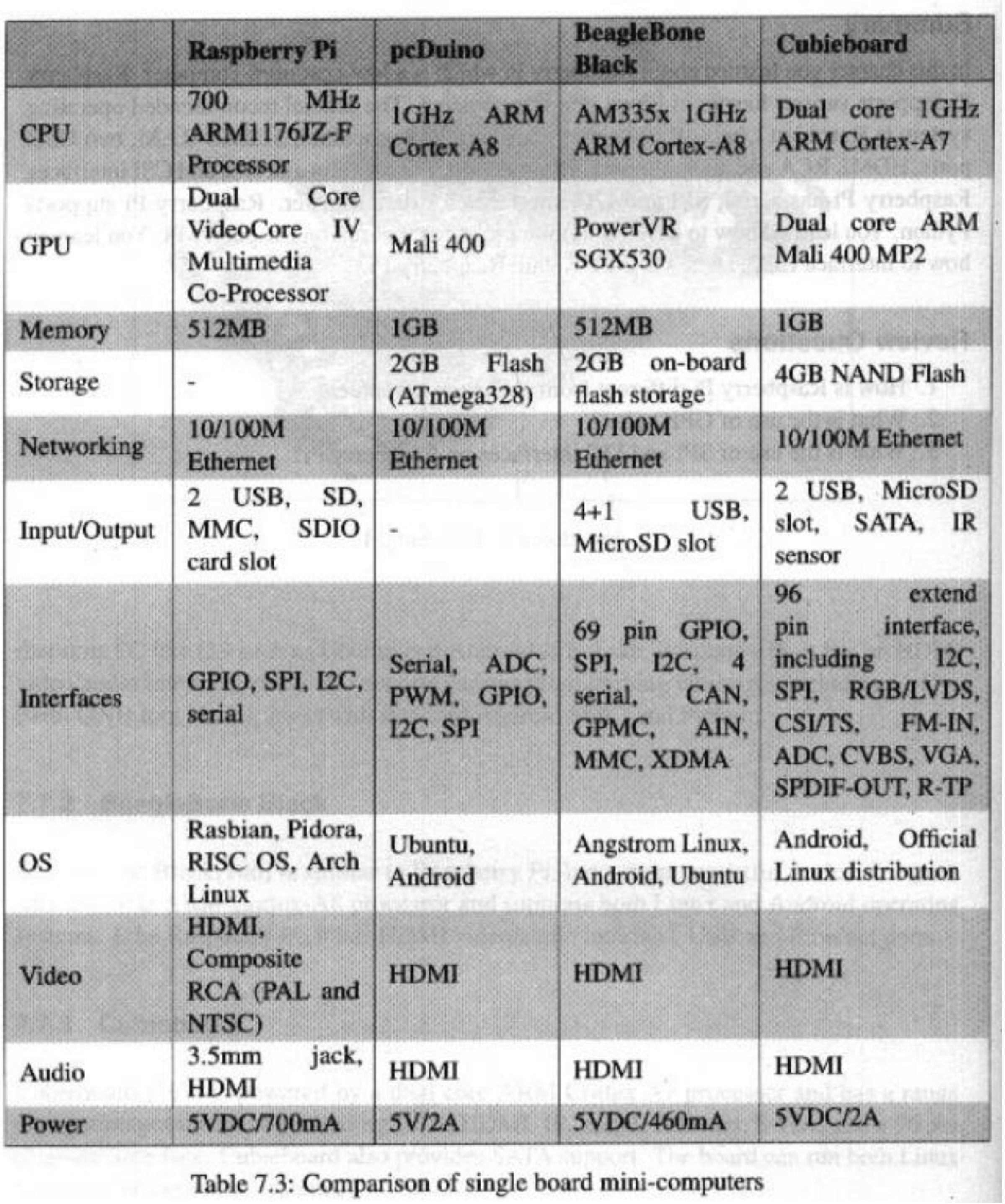
BeagleBone Black[106] is similar to Raspberry Pi, but a more powerful device. It comes with a 1 GHz ARM Cortex-A8 processor and supports both Linux and Android operating systems. Like Raspberry Pi, it has HDMI video/audio interface, USB and Ethernet ports.

pcDuino

pcDuino [105] is an Arduino-pin compatible single board mini-computer that comes with a 1 GHz ARM Cortex-A8 processor. pcDuino is a high performance and cost effective device that runs PC like OS such as Ubuntu and Android ICS. Like, Raspberry Pi, it has an HDMI video/audio interface. pcDuino supports various programming languages including C, C++ (with GNU tool chain), Java (with standard Android SDK) and Python.

Cubieboard

Cubieboard [107] is powered by a dual core ARM Cortex A7 processor and has a range of input/output interfaces including USB, HDMI, IR, serial, Ethernet, SATA, and a 96 pin extended interface. Cubieboard also provides SATA support. The board can run both Linux and Android operating systems.



	Raspberry Pi	pcDuino	BeagleBone Black	Cubieboard
CPU	700 MHz ARM1176JZ-F Processor	1GHz ARM Cortex A8	AM335x 1GHz ARM Cortex-A8	Dual core 1GHz ARM Cortex-A7
GPU	Dual Core VideoCore IV Multimedia Co-Processor	Mali 400	PowerVR SGX530	Dual core ARM Mali 400 MP2
Memory	512MB	1GB	512MB	1GB
Storage	-	2GB Flash (ATmega328)	2GB on-board flash storage	4GB NAND Flash
Networking	10/100M Ethernet	10/100M Ethernet	10/100M Ethernet	10/100M Ethernet
Input/Output	2 USB, SD, MMC, SDIO card slot	-	4+1 USB, MicroSD slot	2 USB, MicroSD slot, SATA, IR sensor
Interfaces	GPIO, SPI, I2C, serial	Serial, ADC, PWM, GPIO, I2C, SPI	69 pin GPIO, SPI, I2C, 4 serial, CAN, GPMC, AIN, MMC, XDMA	96 extend pin interface, including I2C, SPI, RGB/LVDS, CSI/TS, FM-IN, ADC, CVBS, VGA, SPDIF-OUT, R-TP
OS	Rasbian, Pidora, RISC OS, Arch Linux	Ubuntu, Android	Angstrom Linux, Android, Ubuntu	Android, Official Linux distribution
Video	HDMI, Composite RCA (PAL and NTSC)	HDMI	HDMI	HDMI
Audio	3.5mm jack, HDMI	HDMI	HDMI	HDMI
Power	5VDC/700mA	5V/2A	5VDC/460mA	5VDC/2A

Table 7.3: Comparison of single board mini-computers

Part-A			
Q.No		Competence	BT Level
1.	What are the four pillars of IoT	Analyse	BTL 4
2.	What are the benefits of Horizontal standard based platforms	Analyse	BTL 4
3.	Give examples of vertical standard based platforms.	Understand	BTL 2
4.	Name some application areas for Cellular M2M.	Understand	BTL 2
5.	What are the barriers to the global M2M market?	Analyse	BTL 4
6.	What is an internet of objects?	Analyse	BTL 4
7.	What is an internet of transducers?	Analyse	BTL 4
8.	What is an internet of controllers?	Analyse	BTL 4
9.	What are the three layer of IoT architecture?	Understand	BTL 2
10.	What are the two major communication technologies?	Understand	BTL 2
11.	What are the basic Building blocks of an IoT Device?	Analyse	BTL 4
Part-B			
Q.No	Questions	Competence	BT Level
1.	With the neat illustration, describe the 4 pillars of IoT.	Analysis	BTL4
2.	Discuss in detail about the Machine to Machine architectire.	Analysis	BTL4
3.	Discuss in detail the operation of RFID.	Analysis	BTL4
4.	Discuss in detail about SCADA operation	Analysis	BTL4
5.	Discuss in detail about Wireless sensor network.	Analysis	BTL4

UNIT IV - WEB OF THINGS AND CLOUD OF THINGS

Web of Things versus Internet of Things: What is the difference?

Internet: Interconnected networks

- They are interconnected via IP(Internet Protocol)
- There are IP addresses in the internet, no domain names such as Wikipedia.org
- Started around 1950 in a effort to make two computers talk to each other.

Web: Linked documents and resources

- Uses HTTP
- The web needs the Internet Underneath to function
- Started around 1980 in an effort to help people share data over the Internet.

Two pillars of Web

First pillar consists :

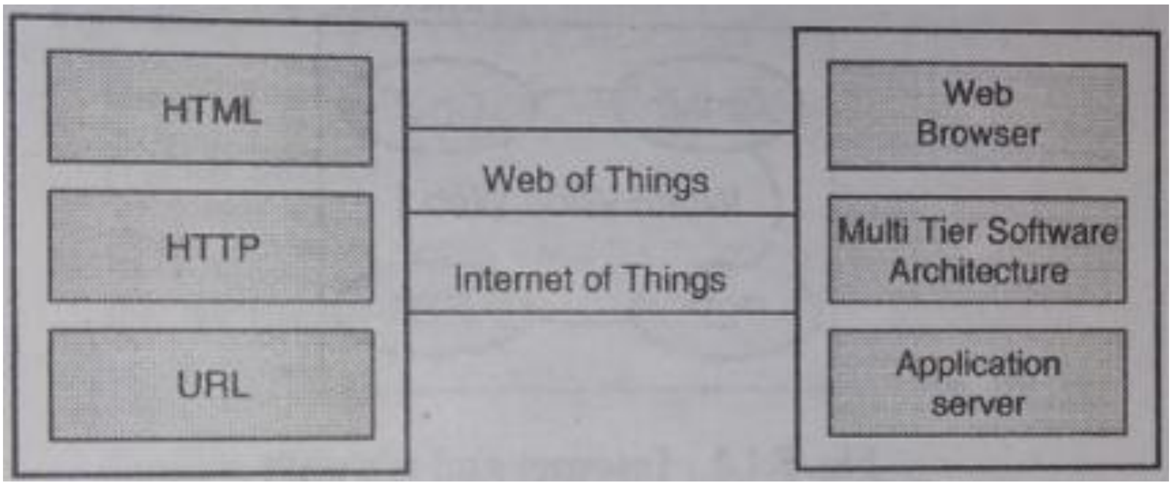
1. Hyper Text Markup Language(HTML)
2. Hyper Text Transfer Protocol(HTTP)
3. Uniform Resource Locator(URL)

First pillar technique provides a way to describe a resource or service , how to locate it and how to distribute it over the Web.

Second pillar consists:

1. Web Browser
2. Multi – Tiered Software Architectures
3. Application Servers

Second pillar techniques provide a way to access the resource or service via URLs, how to design the web Application and how to run and maintain the web application.



Architecture Standardization for WoT

Platform Middleware for WoT :

Middleware of WoT application and IoT focuses upon communication interface and application’s business logic in depth.

- WoT tries to connect everyday objects like domestic appliances, sensors, actuators and some other embedded system and provide a unified application.
- To realize this Application sense of IoT , WoT uses object Oriented approach to view everything as object.
- Many techniques like Object Oriented Design, JavaScript Object Notation(JSON),Document Object Model(DOM) etc are famous in WoT.

M2M Middleware Standards

The key elements of M2M standardization are as follows:

1. M2M Device
2. M2M Area Network
3. M2M Gateway
4. M2M Communication Network
5. M2M Application Server

WSN Middleware Standards

Wireless Sensor Network(WSN) Standardization approach is done by Ubiquitous Sensor Network(USN) Standardization which includes following components :

1. USN Application Platform

2. USN Middleware
3. Network Infrastructures
4. USN Gateways
5. Sensor Networks

SCADA Middleware Standards

- SCADA Standardization provides a framework for framework for information exchange model, device interfaces, services, protocols and software object modeling.
- SCADA middleware platform focuses on following points in industry automation:

1. Interaction
2. Communication Mechanisms
3. Context
4. Secure Distributed Storage
5. Additional Tools

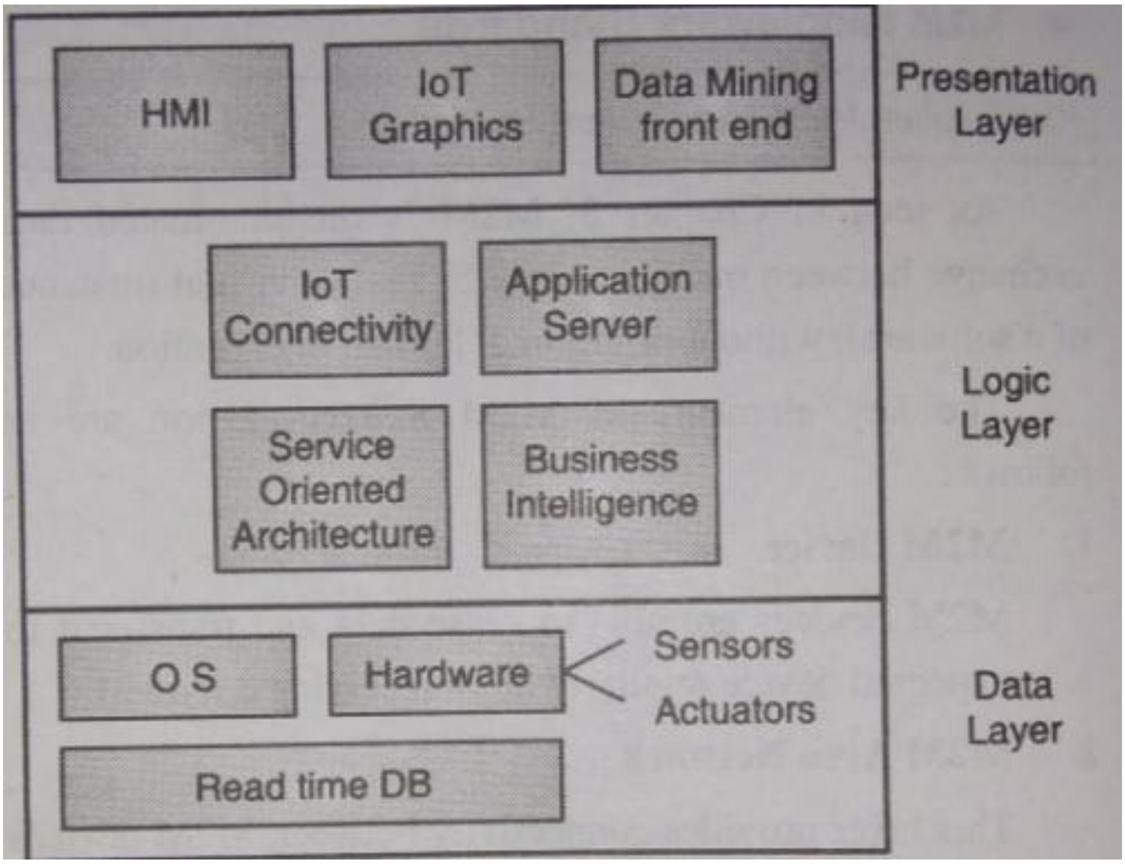
RFID Middleware Standards

- RFID middleware specifies the business logic that is used to filter the data and extract the useful information from the huge data.
- RFID is the most complete and standardized approach in all of the four pillars of IoT.

1. Manage Device
2. Collect and Integrate Data
3. Structure and Filter Data
4. Tag ID Association

- RFID tag information is recorded by mobile terminals with RFID readers and it also stores the location based data.

Unified Multitier WoT Architecture



- Multitier IoT Middleware adds the IoT based connectivity and service oriented layer in the existing three tiered architecture.

- The basic three tiered architecture consists of the data layer, logic layer, and presentation layer.

- **Data Layer:**

1. The data tier handles all the data generation and storage in WoT.
2. This layer handles data gathered from numerous distributed sensors and provides a mechanism to store it.

- **Logic Layer:**

1. Logic layer defines middleware and the application logic of the whole IoT and WoT system.
2. The basic middleware techniques like application server utility using IIS,.NET framework is also implemented in logic layer.

Unified Multitier WoT Architecture

- **Presentation Layer:**

1. Presentation layer defines the User Interface of the WoT to the outside world.

2. This layer provides a mechanism to access the information and services provided by the bottom two layers.
3. IoT based system and WoT has IoT based graphical representation that provide s functionalities like report generation, data mining, HMI tools in SCADA, DSS, etc.

WoT Portals and Business Intelligence

- The web portal provides a point of access to the WWW using URLs and addresses.
- Web portals have portlets , an application in web portals which receives request and return appropriate information.
- Some WoT portals are:

1. Pachub
2. Microsoft Hohm/Google Power Meter
3. Sun SPOT(Small Program Object Technology)
4. Sensor Map Microsoft

- The huge data collected by large number of sensors, is available in raw format. To make sense of this data the WoT middleware defines a business logic which converts this data into meaningful information . Such logic is also known as Business Intelligence (BI)of the WoT.
- Data mining and Decision Support System are well known mechanisms to implement BI.
- Common uses of BI:

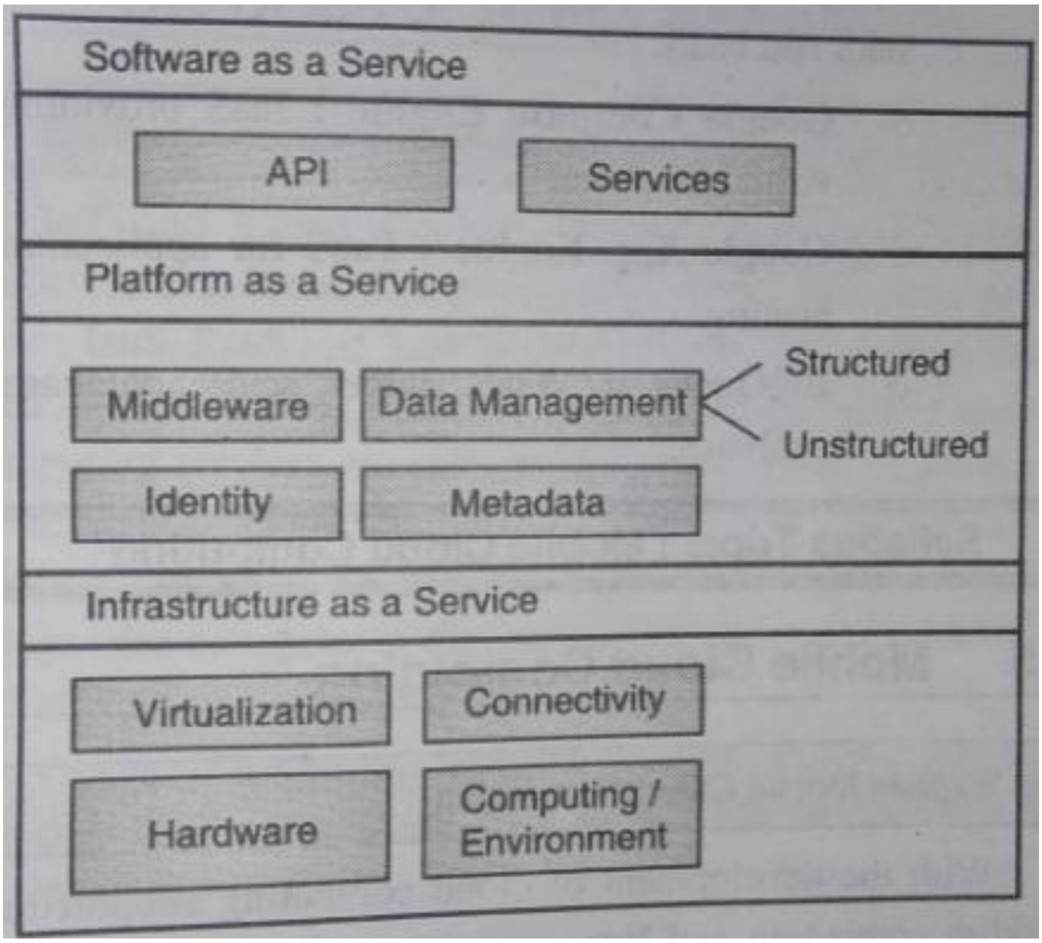
1. Data report generation
2. Data analysis
3. Data mining
4. Benchmarking

Cloud of Things –Grid/SOA and Cloud Computing

- Cloud Computing is derived from grid computing , where both technologies tries to use computing resources from multiple locations.
- Parallel Virtual Machine (PVM),Message Passing Interface(MPI),play important role in building the middleware for grid computing and cloud computing.
- Cloud Computing also supports virtualization creates virtual versions of the resources like servers, storage , OS, etc.

- Virtualization is done into types , Single System Virtualization (SSV) and Multi System Virtualization(MSV).
- SSV technique provides one to many virtualization. MSV provides many to one virtualization.
- SSV can be used with lower level nodes of MSV but is not compulsory.
- Service Oriented Architecture(SOA)provides a software design that helps communicate between application components.

Cloud Middleware



- Cloud provides its services in three categories IaaS, PaaS and SaaS.
- SaaS utilizes the functionalities of IaaS Middleware and PaaS Middleware to provide Software as a service .
- IaaS middleware include components like network management, system management , configuration of system , etc.
- PaaS middleware handles identity and access management of computing devices , data management which include managing metadata and categorizing structured and unstructured data on the cloud.

- PaaS includes all business level solution and services that are required for SaaS.
- SaaS use IaaS and PaaS middleware functionalities to provide its services and doesn't have SaaS middleware separately.

Cloud Standards

- Cloud standards work on these features and create a standard way to provide above services.
- Following list describes some of cloud standardization organizations :

1. National Institute of Standards and Technology(NIST)
2. Distributed Management Task Force(DMTF)
3. European Telecommunications Standards Institute
4. Cloud Management Working Group(CMWG)
5. Standards Acceleration to Jumpstart Adoption of Cloud Computing(SAJACC)
6. Open Cloud Consortium

Cloud Providers and Systems

- Many cloud service providers are available and many of them are working for cloud development providing Open Source application support.
- Some of the service providers are listed below.
- **Open Source IaaS and PaaS projects :**

1. OpenStack
2. OpenNebula
3. RedHat Cloud
4. Other products are

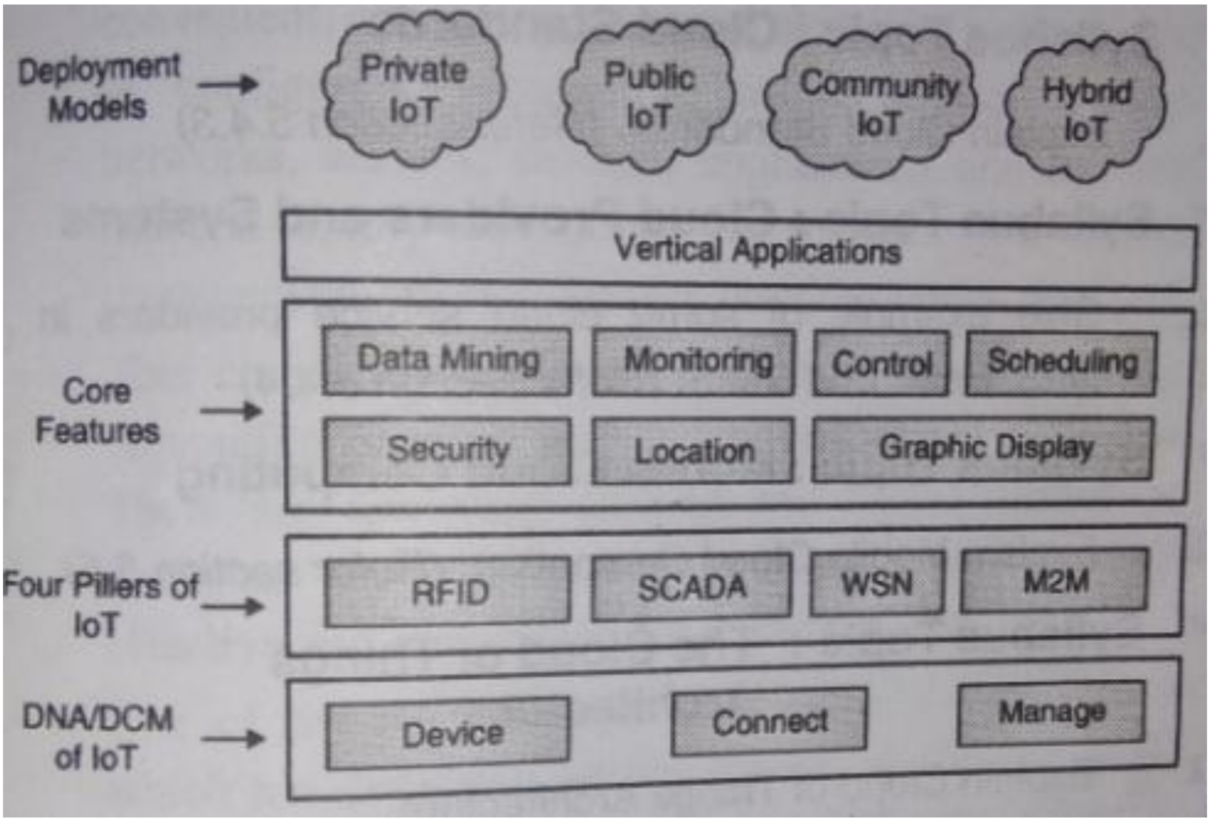
- **Open Source SaaS Projects:**

1. Zoho
2. Pentaho
3. SourceTap
4. Amazon Web Services

Mobile Cloud Computing

- Mobile Cloud Computing(MCC) provides the functionality of a standard cloud in the small handheld devices like smartphones.
- Machine to Machine communication model and sensor capabilities are two main characteristics of mobile cloud computing.
- Mobile cloud collects data in the surrounding and location based data and uses the cloud services for storage , processing and display.
- Mobile Computing , cloud computing and IoT are all part of a big picture providing IoT support to the user's smartphone via cloud services.
- Apple's iCloud services allows user to store data like images , music files , documents etc . on a remote server which can be downloaded by same user's device running Mac or Window based OS.
- Window Live Mesh allows files and folders synchronization on two machines running Windows and Mac OS computers.

The Cloud of Things Architecture



The Cloud of Things Architecture

- Cloud of Things refers to providing IoT based services over the cloud with the help of IaaS , PaaS , SaaS.

- The base layer defines the DNA of IoT which forms the foundation of Cloud of Things.

  1. Device: Devices are smart things like WSN sensors , SCADA Actuators , etc.

  2. Connect: Connect phase provide communication mechanisms for cloud of things.

  3. Manage: Manages the application middleware and cloud computing that forms the backend of the Cloud of Thing.

  4. M2M: M2M is the area of interest for telecom companies.

  5. RFID: RFID gives dumb things identity in IoT based system making them traceable.

  6. WSN:WSN provide the much needed network for sensors and actuators deployed in IoT system.

  7.SCADA : SCADA provides IT controlled smart systems for industrial automation , smart grid , building , etc.

  8.Data Mining : All the data gathered need to be refined into useful information which is done using data mining.

  9.Private IoT : It is managed by the organization .

  10.Public IoT : Public IoT service provider sells these services for a some cost.

  11.Community IoT : Community IoT is an integrated system shared by multiple organizations supporting common community concerns.

  12.Hybrid IoT : This IoT system is integration of two or more of IoT system.

  All these applications and services are powered by the web technologies and cloud technologies that gives the sense of Cloud of Things of IoT.

Part-A			
Q.No	Questions	Competence	BT Level
1.	Compare web of things and internet of things.	Analyse	BTL 4
2.	What are the functions of first pillar of web of things?	Analyse	BTL 4
3.	What are the functions of second pillar of web of things?	Understand	BTL 2
4.	List out the layers in the first pillar of web of things.	Understand	BTL 2
5.	List out the layers in the second pillar of web of things.	Understand	BTL 2
6.	Compare grid and cloud computing	Analyse	BTL 4
7.	List out some of the IoT web portals.	Understand	BTL 2
8.	Compare SSV and MSV techniques.	Analyse	BTL 4
9.	List out some of the cloud standardization organizations.	Understand	BTL 2
10.	List out some of the cloud service providers.	Understand	BTL 2
11.	Compare public and private IoT.	Analyse	BTL 4
12.	What is community IoT?	Analyse	BTL 2
13.	What id hybrid IoT?	Analyse	BTL 2
14.	List out some of the open source SaaS providers.	Remember	BTL 1
15.	List out some of the open source IaaS providers.	Remember	BTL 1
16.	List out some of the open source PaaS providers.	Remember	BTL 1
Part-B			
Q.No	Questions	Competence	BT Level
1.	Describe the key elements of M2M middleware standards.	Analysis	BTL4
2.	Describe the key elements of WSM middleware standards.	Analysis	BTL4
3.	Describe the key elements of SCADA middleware standards.	Analysis	BTL4

4.	Describe the key elements of RFID middleware standards.	Analysis	BTL4
5.	Explain the functional layers of unified multitier WoT architecture.	Analysis	BTL4
6.	Explain the implementation of Service oriented architecture in cloud computing.	Analysis	BTL4
7.	Explain the concept of mobile cloud computing.	Analysis	BTL4
8.	Explain the cloud of things architecture.	Analysis	BTL4

UNIT V - IoT CLOUD OFFERINGS AND IoT CASE STUDIES

Introduction to Cloud Storage Models

Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources that can rapidly provisioned and released with minimal management effort or service provider interaction.

Communication API

- The cloud models for IoT applications are heavily relied on communication APIs.
- Communication APIs facilitate data transfer, control information transfer from application to cloud, one service to another.
- The communication APIs specify the service invocation structure, URL patterns, response formats etc.

WAMP

WAMP stands Web Application Messaging Protocol. It is a communication protocol which can be used in cloud storage models for IoT. WAMP is sub-protocol of web-socket.

There are key concepts that must be understood while discussing WAMP:

- Transport
- Session
- Clients
- Application Code

Xively Cloud for Iot

- **Xively** is an IoT platform owned by Google. Xively offers product companies a way to connect products, manage connected devices and the data they produce, and integrate that data into other systems.
- The backend structure provided by Xively is mainly the data collection, management and distribution infrastructure.
- There are multiple language and support from Xively. It supports and implements in its library the standard HTTP API, Sockets and MQTT.

Python Web Application Framework :

Django

- Django is a free and open source web application framework, written in Python. A web framework is a set of components that helps you to develop websites faster and easier.

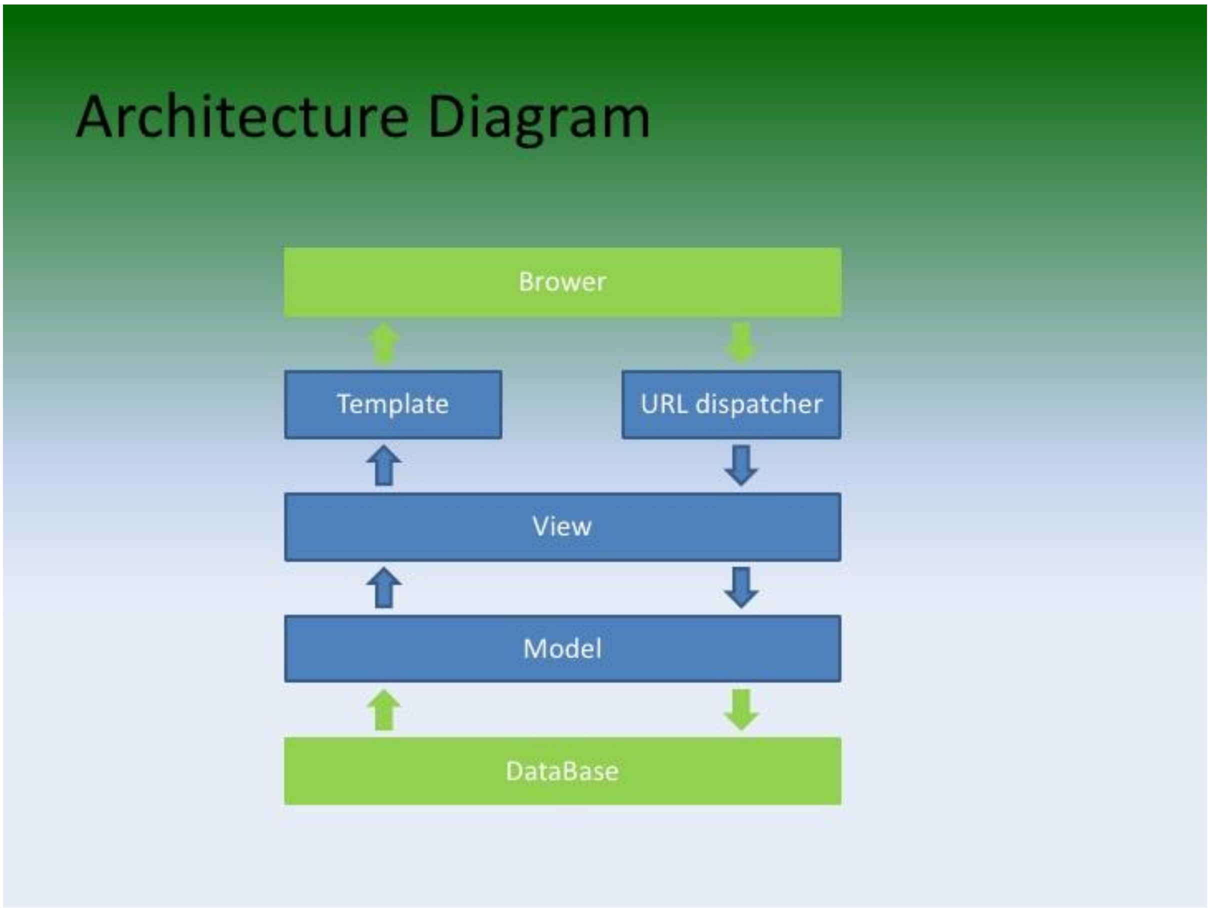
- Frameworks exist to save you from having to reinvent the wheel and to help alleviate some of the overhead when you're building a new site.
- Imagine a mailbox (port) which is monitored for incoming letters (requests). This is done by a web server. The web server reads the letter and then sends a response with a webpage. But when you want to send something, you need to have some content. And Django is something that helps you create the content.

Django Architecture

Django framework is based on the model template view architecture. These are the three logical component that provides high level of abstraction of each other.

- ❖ Model
- ❖ Template
- ❖ View

Django Architecture Diagram



Amazon Web Services for IoT

A subsidiary of Aamzon.com offers several cloud based on demand platform which can be used for IoT application development.

- Amazon EC2
- Amazon AutoScalling
- Amazon Simple Storage Service (S3)
- Amazon RDS
- Amazon DynamoDB

Case Study: Home Intrusion Detection

- The primary objective of home intrusion detection system is to develop an IoT system which can detect the intrusion with help of sensors.
- These sensors needs to be properly placed so that they detect intrusion effectively.
- UI devices should be connected to the network.

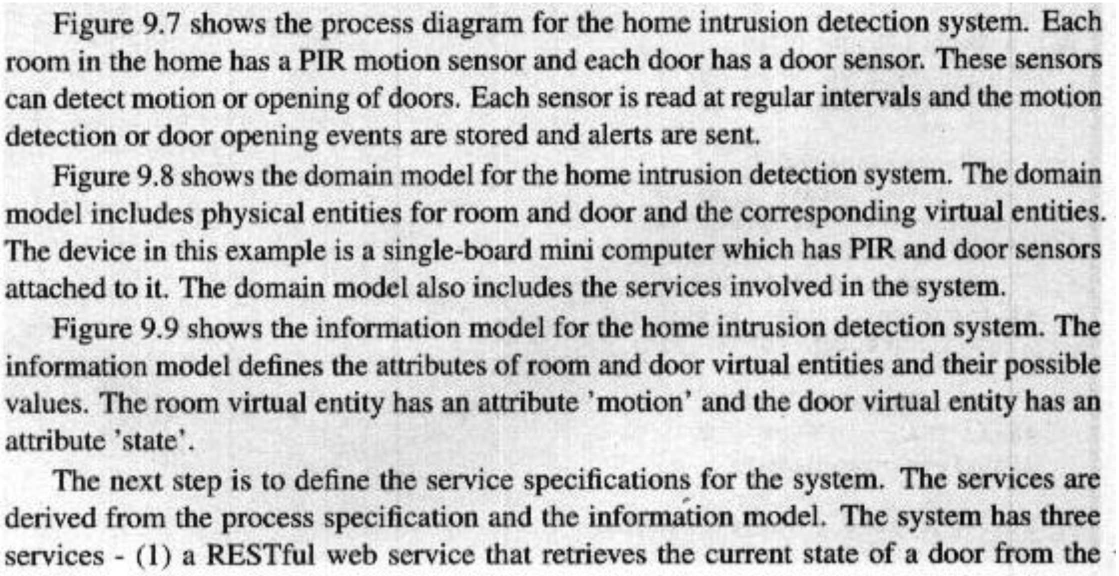
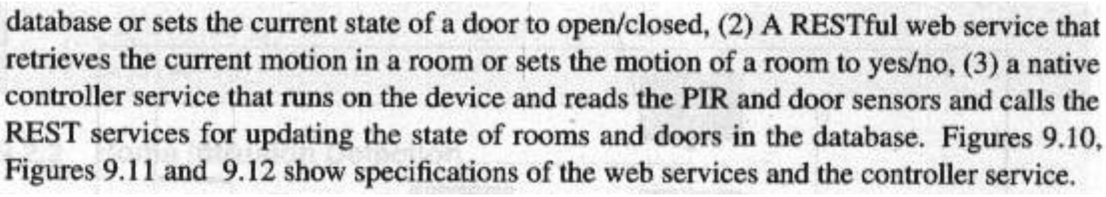


Figure 9.7 shows the process diagram for the home intrusion detection system. Each room in the home has a PIR motion sensor and each door has a door sensor. These sensors can detect motion or opening of doors. Each sensor is read at regular intervals and the motion detection or door opening events are stored and alerts are sent.

Figure 9.8 shows the domain model for the home intrusion detection system. The domain model includes physical entities for room and door and the corresponding virtual entities. The device in this example is a single-board mini computer which has PIR and door sensors attached to it. The domain model also includes the services involved in the system.

Figure 9.9 shows the information model for the home intrusion detection system. The information model defines the attributes of room and door virtual entities and their possible values. The room virtual entity has an attribute 'motion' and the door virtual entity has an attribute 'state'.

The next step is to define the service specifications for the system. The services are derived from the process specification and the information model. The system has three services - (1) a RESTful web service that retrieves the current state of a door from the



database or sets the current state of a door to open/closed, (2) A RESTful web service that retrieves the current motion in a room or sets the motion of a room to yes/no, (3) a native controller service that runs on the device and reads the PIR and door sensors and calls the REST services for updating the state of rooms and doors in the database. Figures 9.10, Figures 9.11 and 9.12 show specifications of the web services and the controller service.

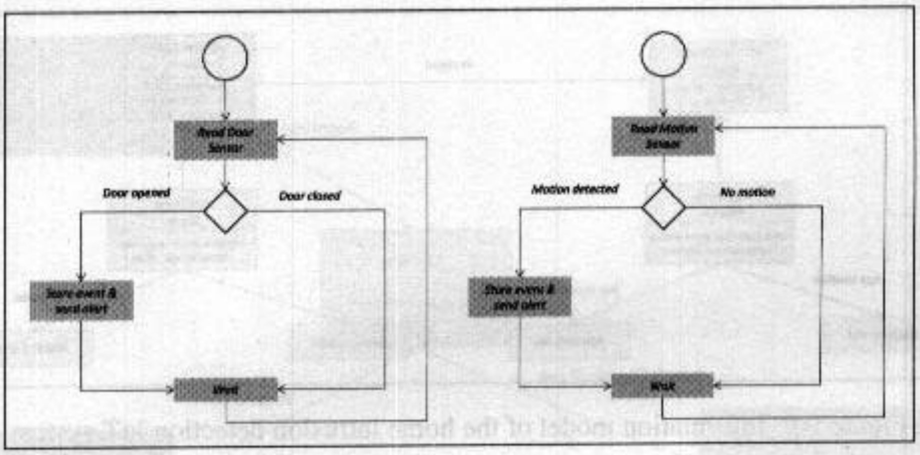


Figure 9.7: Process specification of the home intrusion detection IoT system

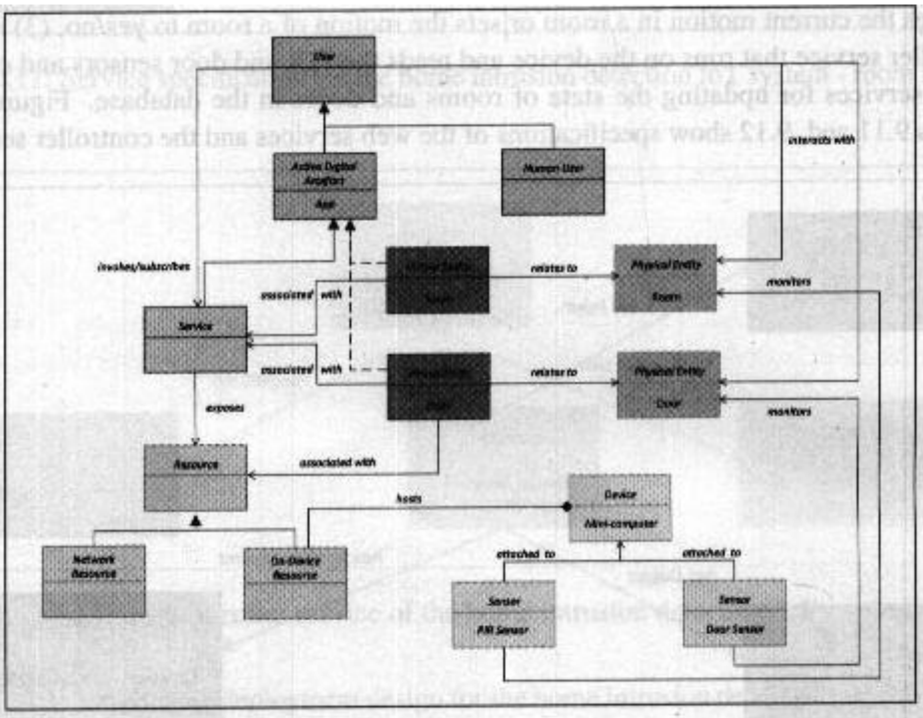


Figure 9.8: Domain model of the home intrusion detection IoT system

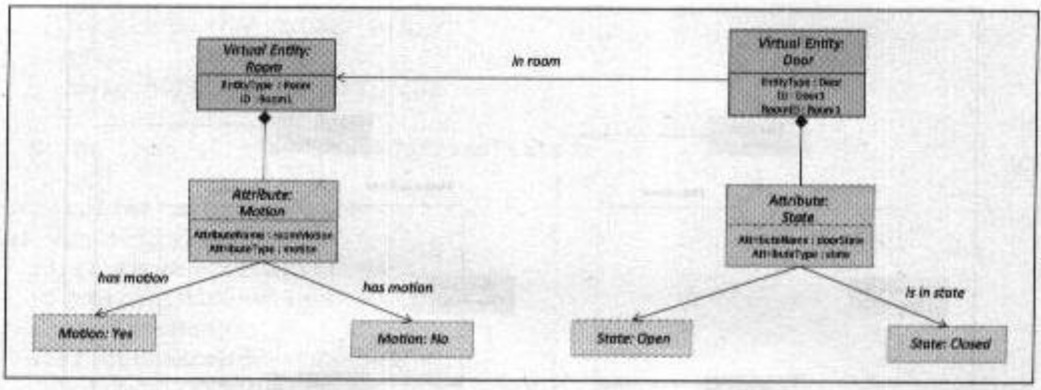


Figure 9.9: Information model of the home intrusion detection IoT system

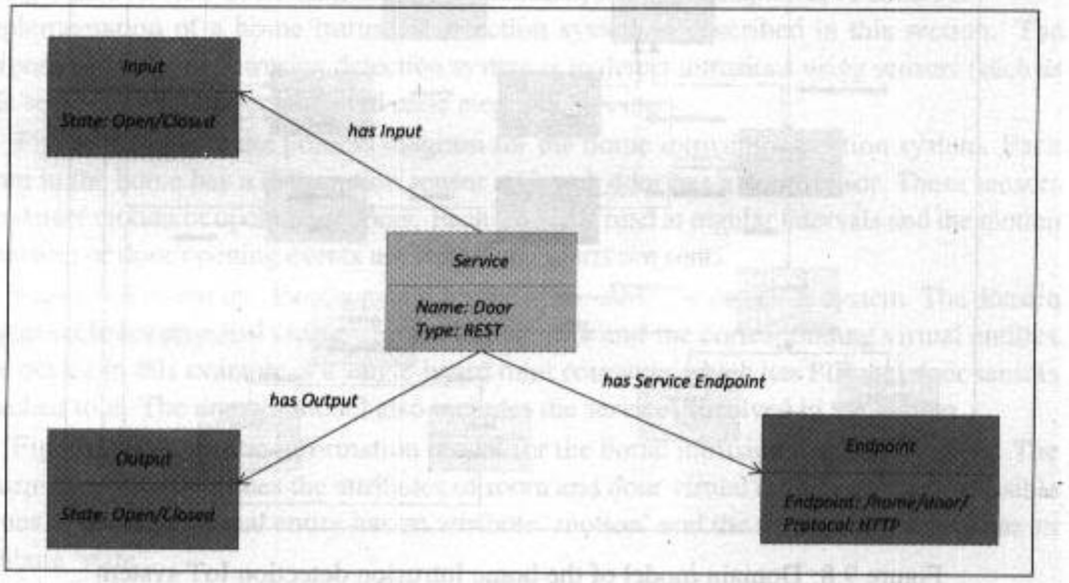


Figure 9.10: Service specification for the home intrusion detection IoT system - door service

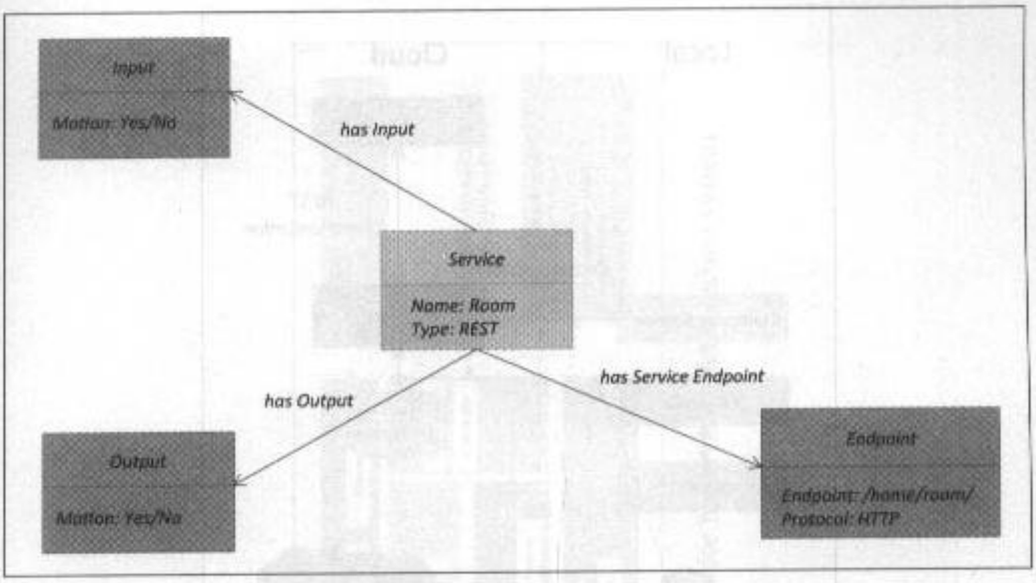


Figure 9.11: Service specification for the home intrusion detection IoT system - room service

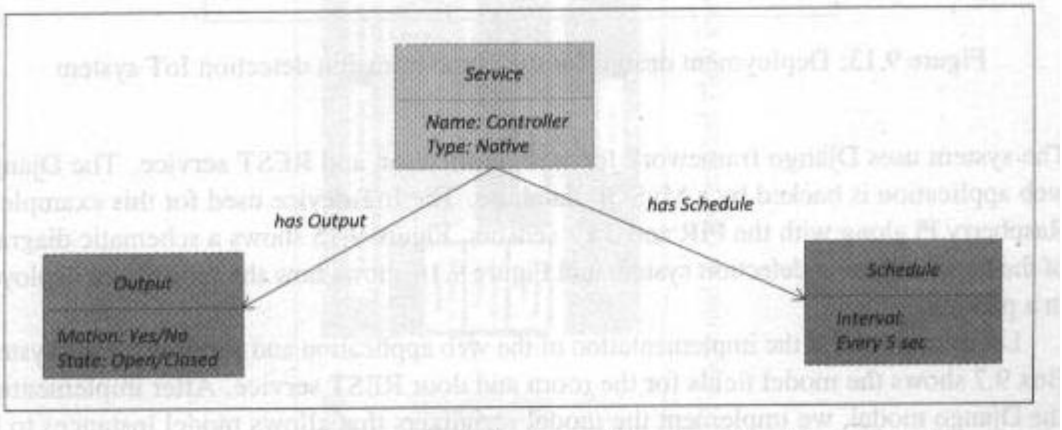


Figure 9.12: Controller service of the home intrusion detection IoT system

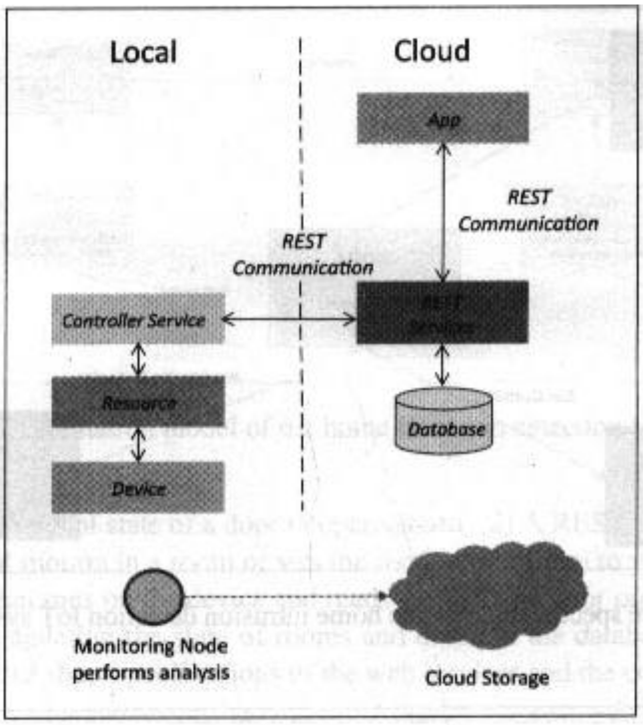


Figure 9.13: Deployment design for the home intrusion detection IoT system

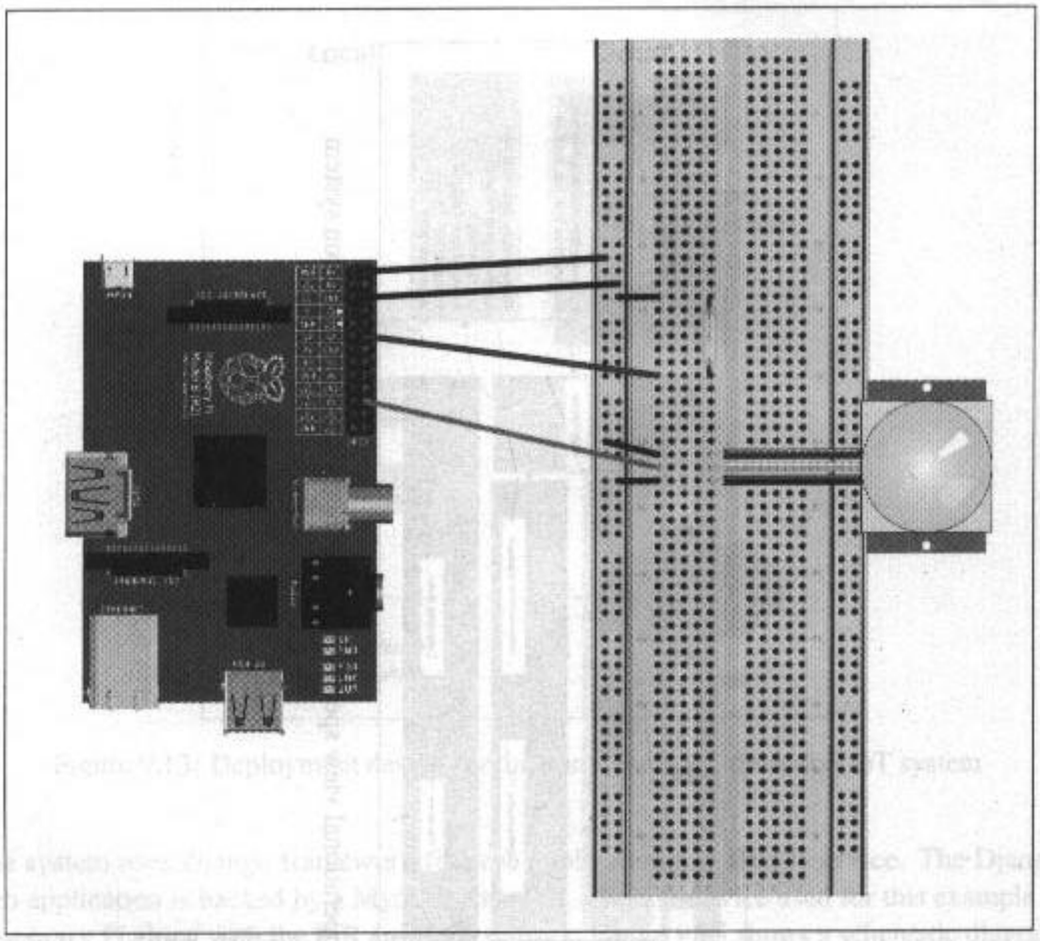


Figure 9.15: Schematic diagram of the home intrusion detection IoT system prototype, showing the device and ultrasonic sensor

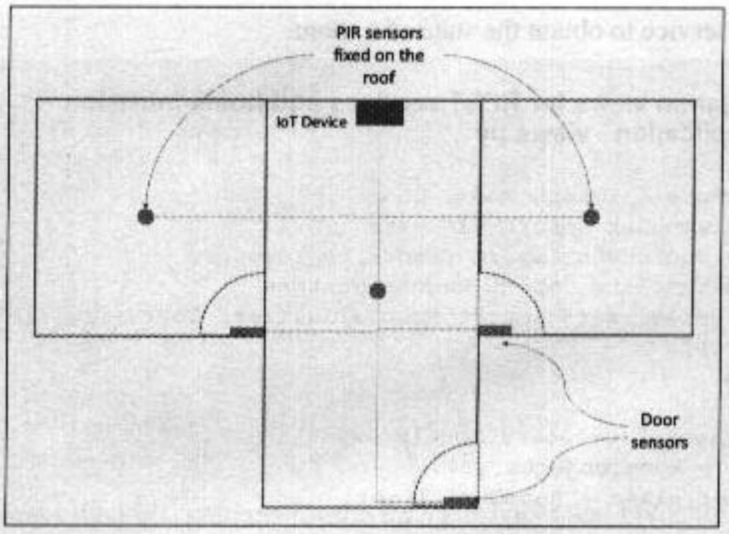
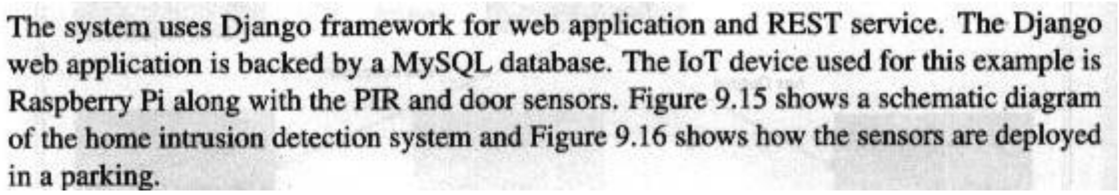
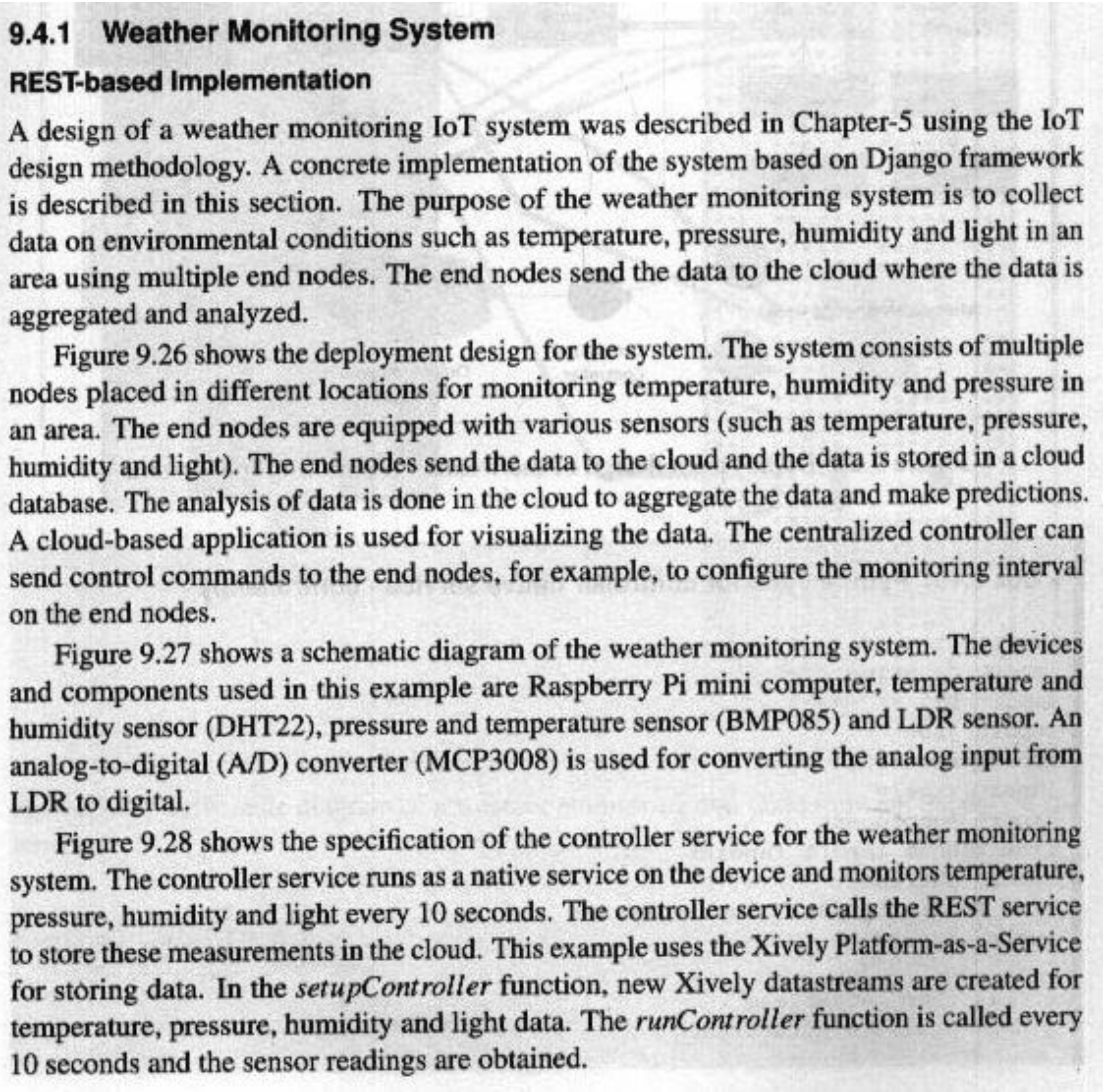


Figure 9.16: Deployment of sensors for home intrusion detection system



The system uses Django framework for web application and REST service. The Django web application is backed by a MySQL database. The IoT device used for this example is Raspberry Pi along with the PIR and door sensors. Figure 9.15 shows a schematic diagram of the home intrusion detection system and Figure 9.16 shows how the sensors are deployed in a parking.



9.4.1 Weather Monitoring System

REST-based Implementation

A design of a weather monitoring IoT system was described in Chapter-5 using the IoT design methodology. A concrete implementation of the system based on Django framework is described in this section. The purpose of the weather monitoring system is to collect data on environmental conditions such as temperature, pressure, humidity and light in an area using multiple end nodes. The end nodes send the data to the cloud where the data is aggregated and analyzed.

Figure 9.26 shows the deployment design for the system. The system consists of multiple nodes placed in different locations for monitoring temperature, humidity and pressure in an area. The end nodes are equipped with various sensors (such as temperature, pressure, humidity and light). The end nodes send the data to the cloud and the data is stored in a cloud database. The analysis of data is done in the cloud to aggregate the data and make predictions. A cloud-based application is used for visualizing the data. The centralized controller can send control commands to the end nodes, for example, to configure the monitoring interval on the end nodes.

Figure 9.27 shows a schematic diagram of the weather monitoring system. The devices and components used in this example are Raspberry Pi mini computer, temperature and humidity sensor (DHT22), pressure and temperature sensor (BMP085) and LDR sensor. An analog-to-digital (A/D) converter (MCP3008) is used for converting the analog input from LDR to digital.

Figure 9.28 shows the specification of the controller service for the weather monitoring system. The controller service runs as a native service on the device and monitors temperature, pressure, humidity and light every 10 seconds. The controller service calls the REST service to store these measurements in the cloud. This example uses the Xively Platform-as-a-Service for storing data. In the *setupController* function, new Xively datastreams are created for temperature, pressure, humidity and light data. The *runController* function is called every 10 seconds and the sensor readings are obtained.

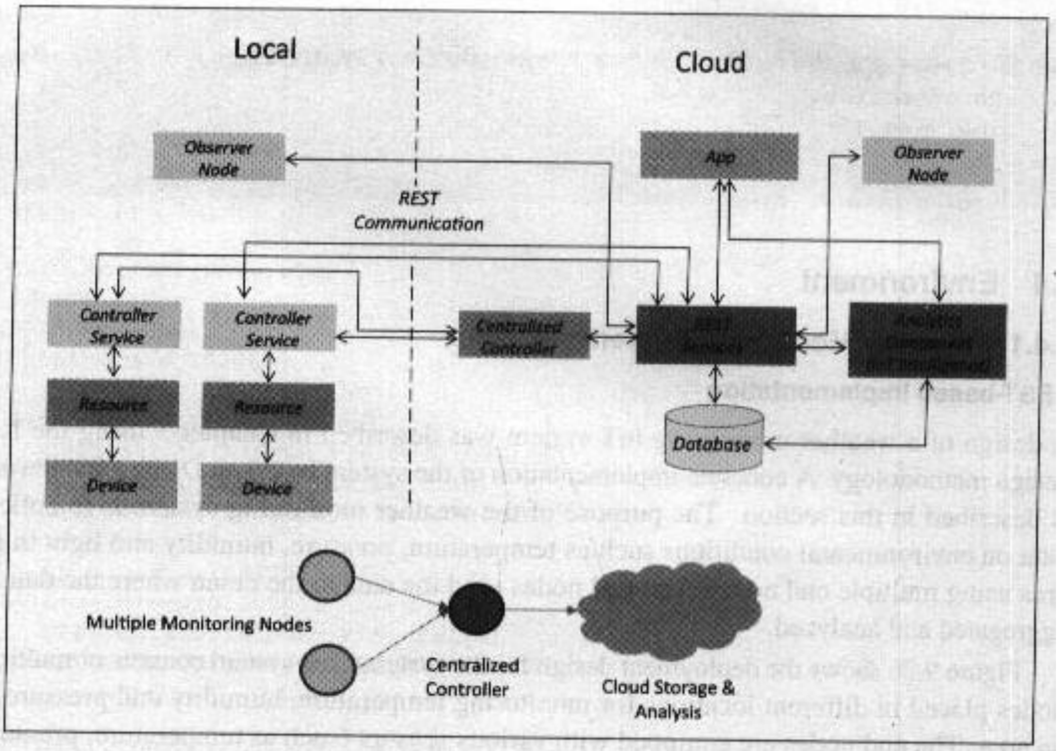


Figure 9.26: Deployment design of the weather monitoring IoT system

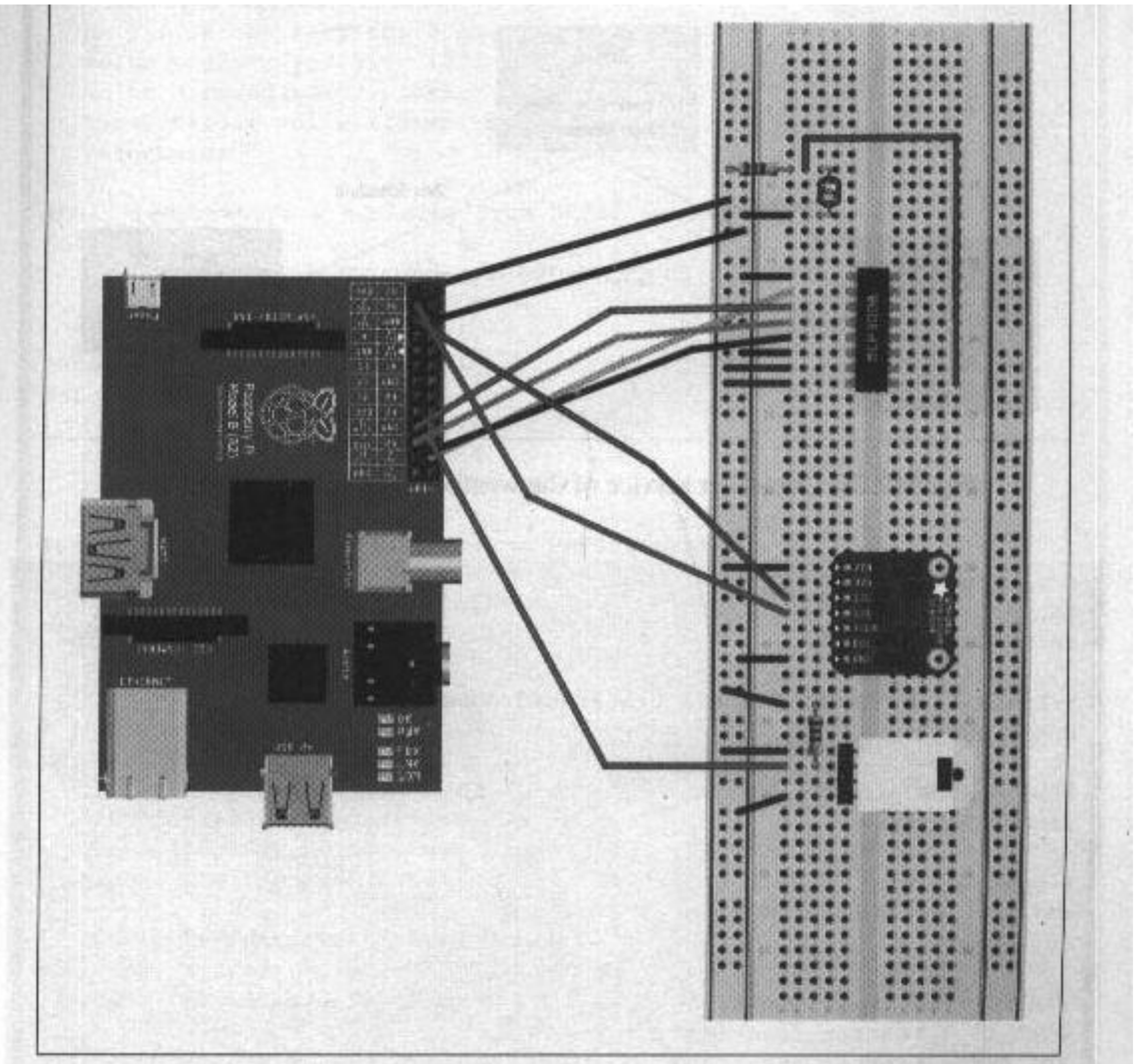


Figure 9.27: Schematic diagram of a weather monitoring end-node showing the device and sensors

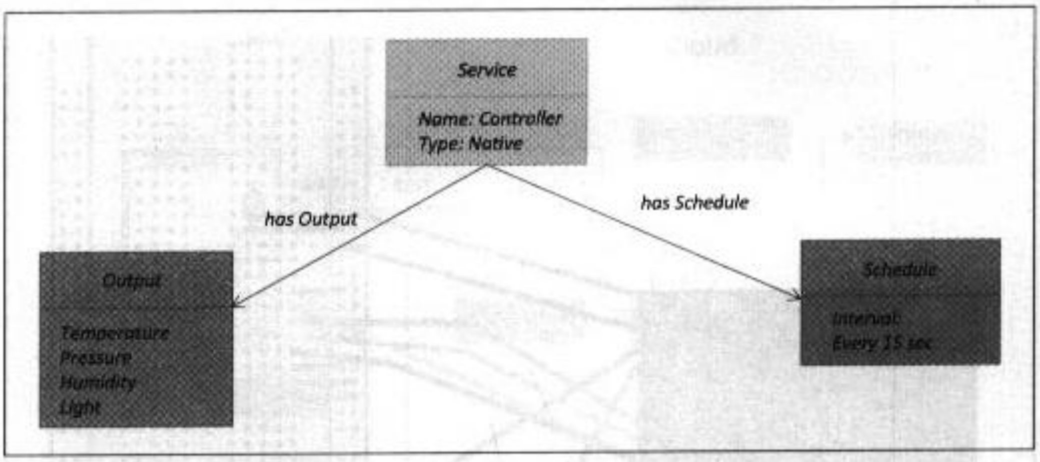
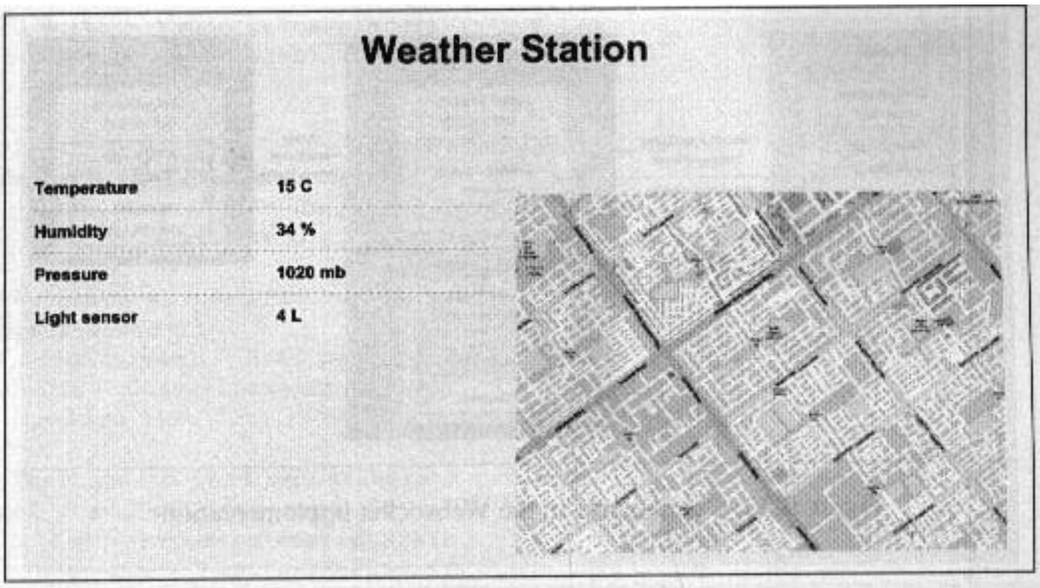


Figure 9.28: Controller service of the weather monitoring IoT system



Weather Station

Temperature	15 C
Humidity	34 %
Pressure	1020 mb
Light sensor	4 L

Figure 9.29: Screenshot of weather monitoring web application

9.4.3 Air Pollution Monitoring

IoT based air pollution monitoring systems can monitor emission of harmful gases by factories and automobiles using gaseous and meteorological sensors. This section provides an implementation of an air pollution monitoring IoT system. The deployment design for the system is similar to the deployment shown in Figure 9.26. The system design steps are similar to the weather monitoring system described in the previous section. Therefore only the schematic design and the controller implementation is provided.

The system consists of multiple nodes placed in different locations for monitoring air pollution in an area. The end nodes are equipped with CO and NO_2 sensors. The end nodes send the data to the cloud and the data is stored in a cloud database. A cloud-based application is used for visualizing the data.

Figure 9.33 shows a schematic diagram of air pollution monitoring end-node. The end node includes a Raspberry Pi mini-computer, MICS-2710 NO_2 sensor and MICS-5525 CO sensor. An A/D converter (MCP3008) is used for converting the analog inputs from the sensors to digital.

Box 9.27 shows the implementation of the native controller service for air pollution monitoring system. The controller service runs as a native service on the device and obtains the sensor readings every 10 seconds. The controller service calls the Xively REST service to store these measurements in the cloud.

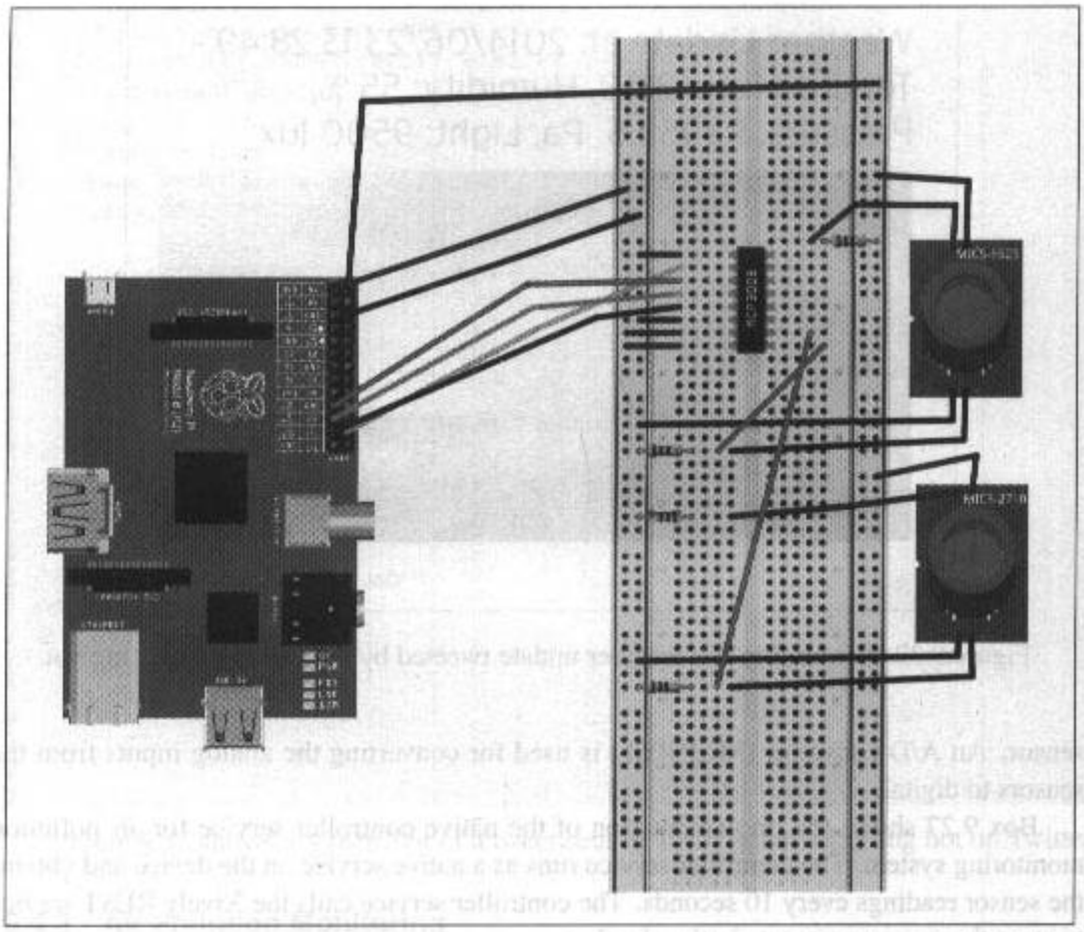


Figure 9.33: Schematic diagram of air pollution monitoring end-node showing the device and sensors

9.5.1 Smart Irrigation

Smart irrigation systems use IoT devices and soil moisture sensors to determine the amount of moisture in the soil and release the flow of water through the irrigation pipes only when the moisture levels go below a predefined threshold. Data on the moisture levels is also collected in the cloud where it is analyzed to plan watering schedules.

An implementation of a smart irrigation system is described in this section. The deployment design for the system is similar to the deployment shown in Figure 9.26. The system consists of multiple nodes placed in different locations for monitoring soil moisture in a field. The end nodes send the data to the cloud and the data is stored in a cloud database. A cloud-based application is used for visualizing the data. Figure 9.36 shows a schematic diagram of smart irrigation system end-node. The end node includes a Raspberry Pi mini-computer and soil moisture sensor. A solenoid valve is used to control the flow of water through the irrigation pipe. When the moisture level goes below a threshold, the valve is opened to release water. Box 9.30 shows the Python code for the controller native service for the smart irrigation system.

Smart irrigation systems can improve crop yields while saving water. Smart irrigation systems use IoT devices with soil moisture sensors to determine the amount of moisture in the soil and release the flow of water through the irrigation pipes only when the moisture levels go below a predefined threshold. Smart irrigation systems also collect moisture level measurements on a server or in the cloud where the collected data can be analyzed to plan watering schedules. Cultivar's RainCloud [56] is a device for smart irrigation that uses water valves, soil sensors and a WiFi enabled programmable computer.

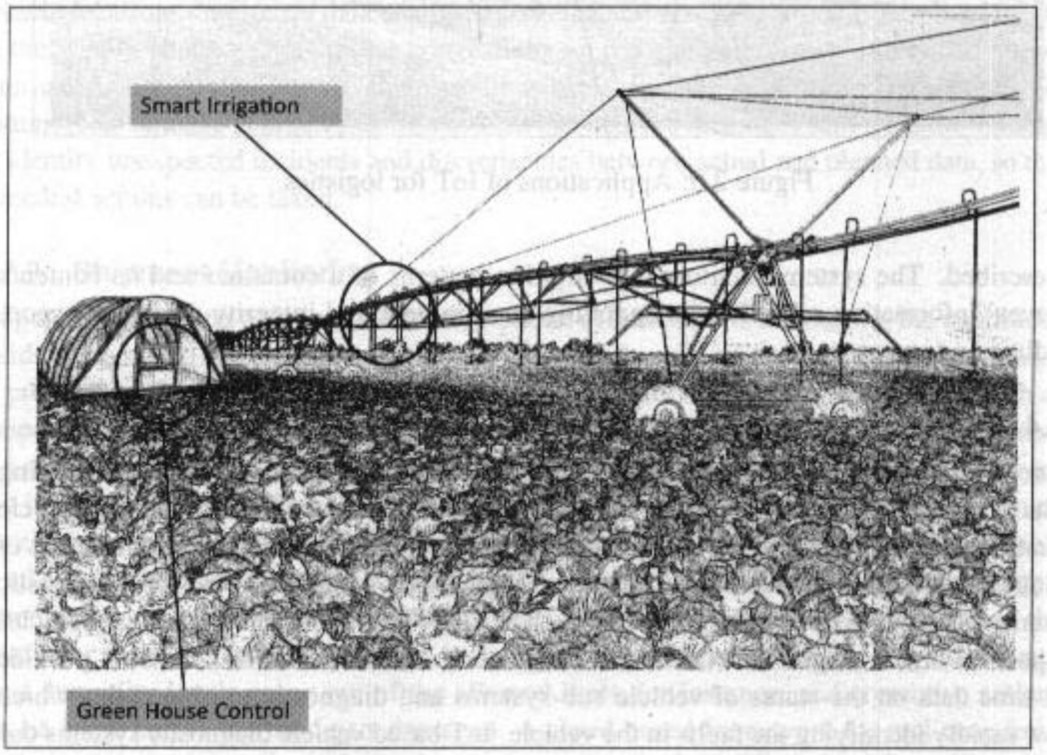


Figure 2.7: Applications of IoT for agriculture

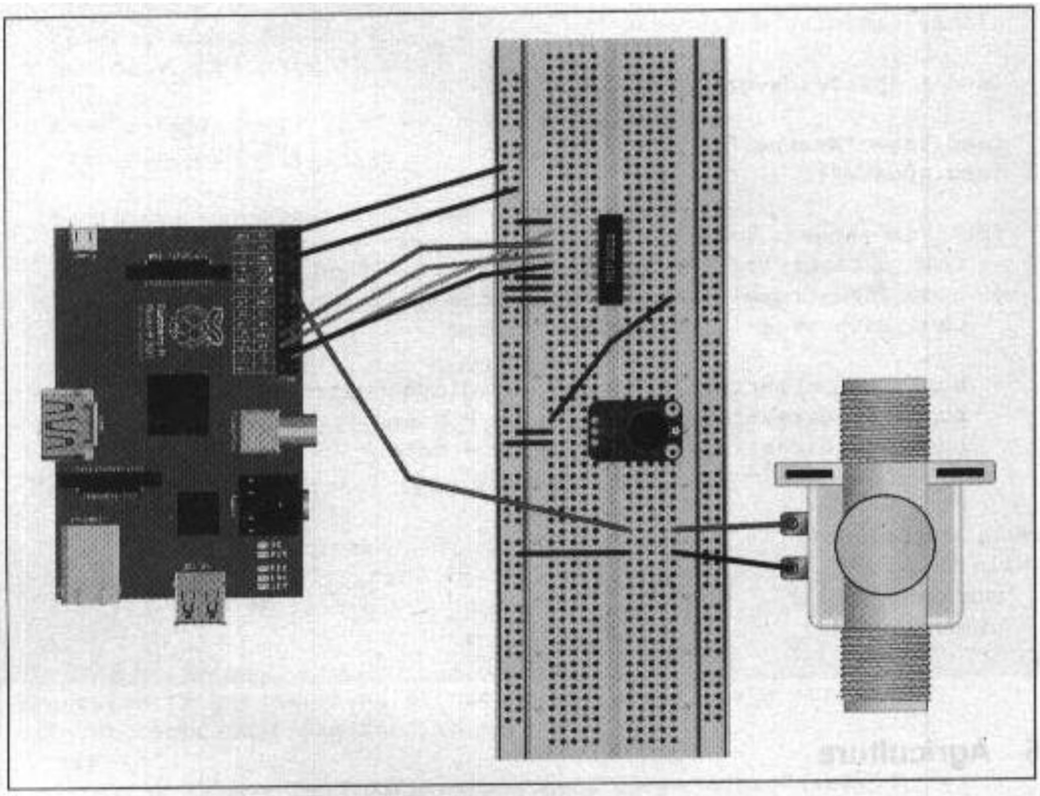


Figure 9.36: Schematic diagram of a smart irrigation system end-node showing the device and sensor

2.5 Energy

2.5.1 Smart Grids

Smart Grid is a data communications network integrated with the electrical grid that collects and analyzes data captured in near-real-time about power transmission, distribution, and consumption. Smart Grid technology provides predictive information and recommendations to utilities, their suppliers, and their customers on how best to manage power. Smart

Grids collect data regarding electricity generation (centralized or distributed), consumption (instantaneous or predictive), storage (or conversion of energy into other forms), distribution and equipment health data. Smart grids use high-speed, fully integrated, two-way communication technologies for real-time information and power exchange. By using IoT based sensing and measurement technologies, the health of equipment and the integrity of the grid can be evaluated. Smart meters can capture almost real-time consumption, remotely control the consumption of electricity and remotely switch off supply when required. Power thefts can be prevented using smart metering. By analyzing the data on power generation, transmission and consumption smart girds can improve efficiency throughout the electric system. Storage collection and analysis of smarts grids data in the cloud can help in dynamic optimization of system operations, maintenance, and planning. Cloud-based monitoring of smart grids data can improve energy usage levels via energy feedback to users coupled with real-time pricing information. Real-time demand response and management strategies can be used for lowering peak demand and overall load via appliance control and energy storage mechanisms. Condition monitoring data collected from power generation and transmission systems can help in detecting faults and predicting outages. In [49], application of IoT in smart grid power transmission is described.

2.5.2 Renewable Energy Systems

Due to the variability in the output from renewable energy sources (such as solar and wind), integrating them into the grid can cause grid stability and reliability problems. Variable output produces local voltage swings that can impact power quality. Existing grids were designed to handle power flows from centralized generation sources to the loads through transmission and distribution lines. When distributed renewable energy sources are integrated into the grid, they create power bi-directional power flows for which the grids were not originally designed. IoT based systems integrated with the transformers at the point of interconnection measure the electrical variables and how much power is fed into the grid. To ensure the grid stability, one solution is to simply cut off the overproduction. For wind energy systems, closed-loop controls can be used to regulate the voltage at point of interconnection which coordinate wind turbine outputs and provides reactive power support [52].

2.5.3 Prognostics

Energy systems (smart grids, power plants, wind turbine farms, for instance) have a large number of critical components that must function correctly so that the systems can perform their operations correctly. For example, a wind turbine has a number of critical components, e.g., bearings, turning gears, for instance, that must be monitored carefully as wear and tear in such critical components or sudden change in operating conditions of the machines can result in failures. In systems such as power grids, real-time information is collected using specialized electrical sensors called Phasor Measurement Units (PMU) at the substations. The information received from PMUs must be monitored in real-time for estimating the state of the system and for predicting failures. Energy systems have thousands of sensors that gather real-time maintenance data continuously for condition monitoring and failure prediction purposes. IoT based prognostic real-time health management systems can predict performance of machines or energy systems by analyzing the extent of deviation of a system from its normal operating profiles. Analyzing massive amounts of maintenance data collected from sensors in energy systems and equipment can provide predictions for the impending failures (potentially in real-time) so that their reliability and availability can be improved. Prognostic health management systems have been developed for different energy systems. OpenPDC [50] is a set of applications for processing of streaming time-series data collected from Phasor Measurement Units (PMUs) in real-time. A generic framework for storage, processing and analysis of massive machine maintenance data, collected from a large number of sensors embedded in industrial machines, in a cloud computing environment was proposed in [51].

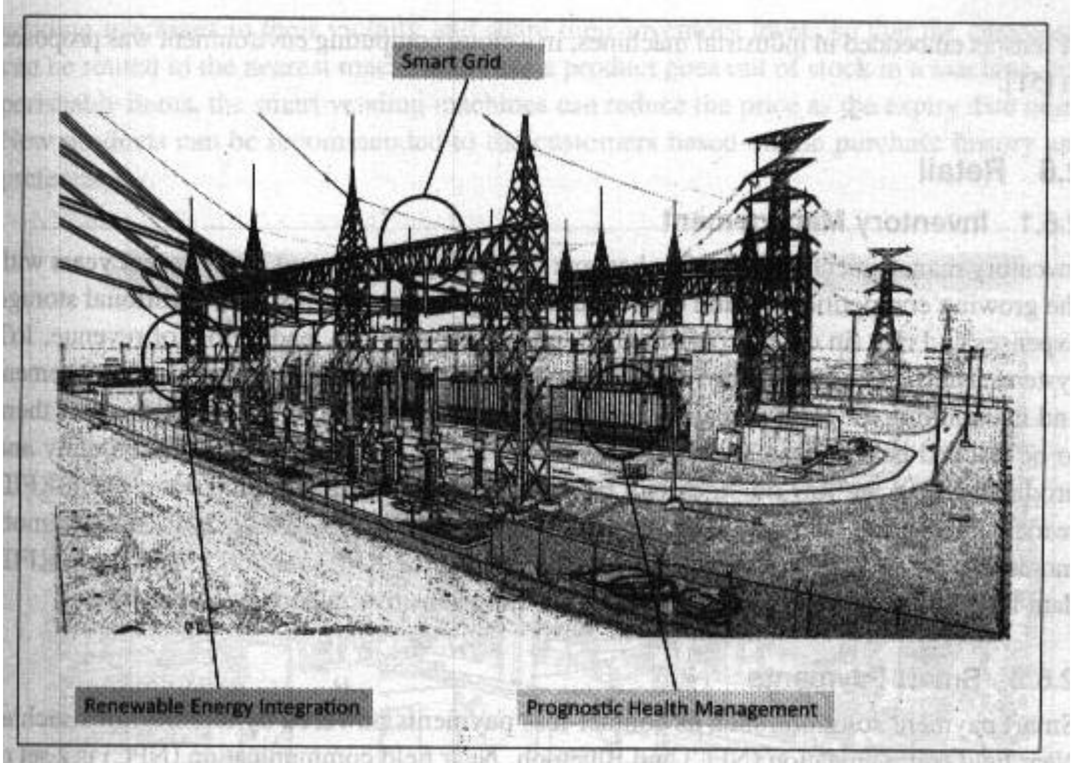


Figure 2.4: Applications of IoT for energy systems

Part-A			
Q.No	Questions	Competence	BT Level
1.	What is cloud computing ?	Analyse	BTL 4
2.	What is application programming interface in communication ?	Analyse	BTL 4
3.	What is autobahn framework?	Understand	BTL 2
4.	What are the commands for the installation of autobahn framework?	Understand	BTL 2
5.	What are the advantages of cloud computing ?	Analyse	BTL 4

Part-B			
Q.No	Questions	Competence	BT Level
1.	Explain web application messaging protocol.	Analysis	BTL4
2.	Draw and explain WAMP communication model and its protocol.	Analysis	BTL4
3.	Explain the features of Xively cloud for IoT and the way in which devices send data to Xively.	Analysis	BTL4
4.	Explain Python web application framework and Django architecture. How does Django processes request?	Analysis	BTL4
5.	Explain the cloud based amazon web services for IoT application development.	Analysis	BTL4
6.	Explain the Skynet – IoT messaging platform.	Analysis	BTL4
7.	Explain briefly the IoT based home intrusion detection system	Analysis	BTL4
8.	Provide an IoT solution for weathering monitoring system	Analysis	BTL4
9.	Explain briefly the IoT based air pollution monitoring system	Analysis	BTL4
10.	Provide an IoT solution for a smart irrigation system	Analysis	BTL4