UNIT V

6 IoT Design and System Engineering

Syllabus

Discuss IOT Requirements; Hardware & Software; Study of IOT Sensors; Tagging and Tracking; Embedded Products; IOT Design; SIM Card Technology; IOT Connectivity and Management; IOT Security & IOT Communication.

Contents

6.1 IoT Requirement : Hardware and Software

6.1.1 Arduino

- Arduino is an open - source electronics platform based on easy - to - use hardware and software.
- Arduino boards are able to read inputs - light on a sensor, a finger on a button, or a Twitter message - and turn it into an output - activating a motor, turning on an LED, publishing something online.
- The Arduino Uno can be powered via the USB connection or with an external power supply. The power source is selected automatically.

Features :

- Support fast computations, ARM based MCU
- AVR micro - controller clock is ATSAMSX8I
- Operating input voltages is 3.3 Volt
- It uses EEPROM, SRAM and Flash memory
- It also support USB and UART
- Fig. 6.1.1 shows Arduino board.

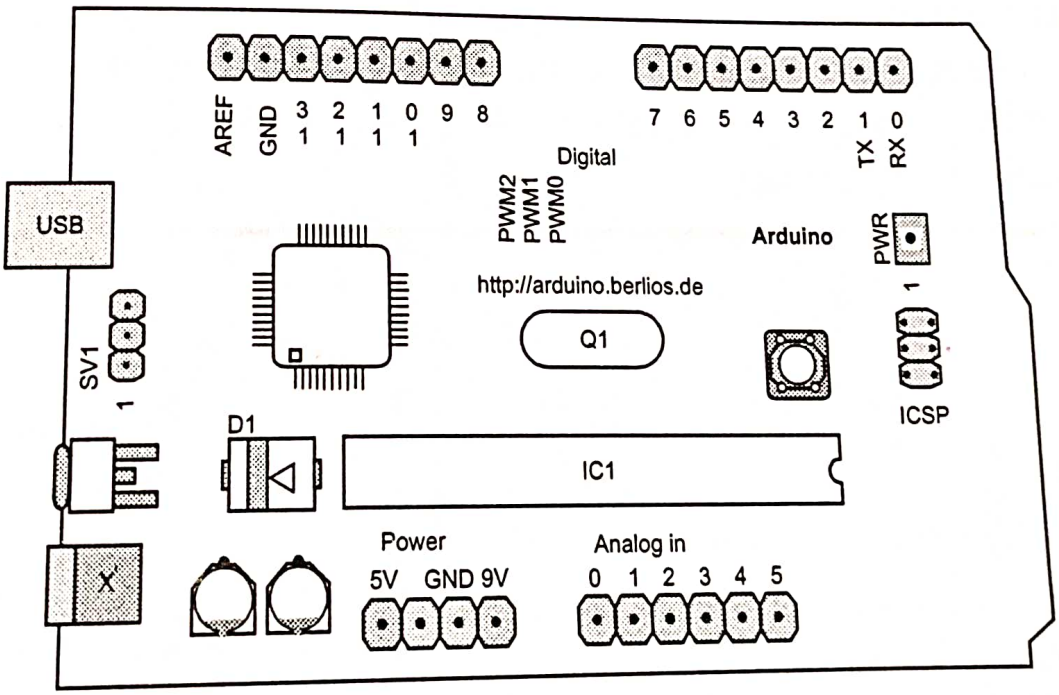

Fig. 6.1.1 Arduino board

Starting clockwise from the top center,

- Analog Reference pin (1st pin)
- Digital ground
- Digital pins 2 - 13 (green)

- Digital pins 0 - 1/Serial In/Out - TX/RX : These pins cannot be used for digital I/O (digital Read and digital Write) if you are also using serial communication
- Reset button - S1
- In-circuit serial programmer
- Analog in pins 0 - 5
- Power and ground pins
- External power supply in (9 - 12VDC) - X1
- Toggles external power and USB power (place jumper on two pins closest to desired supply) - SV1 (purple)
- USB

Digital pins

- In addition to the specific functions listed below, the digital pins on an Arduino board can be used for general purpose input and output via the pinMode( ), digitalRead( ), and digitalWrite( ) commands.
- Each pin has an internal pull-up resistor which can be turned on and off using digitalWrite( ), when the pin is configured as an input. The maximum current per pin is 40 mA.
- **Serial :** 0 (RX) and 1 (TX). Used to receive (RX) and transmit (TX) TTL serial data. On the Arduino Diecimila, these pins are connected to the corresponding pins of the FTDI USB-to-TTL Serial chip. On the Arduino BT, they are connected to the corresponding pins of the WT11 Bluetooth module. On the Arduino Mini and LilyPad Arduino, they are intended for use with an external TTL serial module (e.g. the Mini - USB Adapter).
- **External interrupts :** 2 and 3. These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attachInterrupt() function for details.
- **PWM :** 3, 5, 6, 9, 10, and 11. Provide 8-bit PWM output with the analogWrite() function. On boards with an ATmega8, PWM output is available only on pins 9, 10, and 11.
- **BT Reset :** 7. (Arduino BT-only) Connected to the reset line of the bluetooth module.
- **SPI :** 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). These pins support SPI communication, which, although provided by the underlying hardware, is not currently included in the Arduino language.

- **LED :** 13. On the Diecimila and LilyPad, there is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.

Analog pins

- In addition to the specific functions listed below, the analog input pins support 10 - bit analog - to - digital conversion (ADC) using the analogRead() function.
- Most of the analog inputs can also be used as digital pins: analog input 0 as digital pin 14 through analog input 5 as digital pin 19. Analog inputs 6 and 7 (present on the Mini and BT) cannot be used as digital pins.
- **I2C :** 4 (SDA) and 5 (SCL). Support I2C (TWI) communication.

Power pins

- VIN (sometimes labelled "9 V"). The input voltage to the Arduino board when it's using an external power source.
- You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin. Note that different boards accept different input voltages ranges, please see the documentation for your board. Also note that the LilyPad has no VIN pin and accepts only a regulated input.
- **5 V :** The regulated power supply used to power the microcontroller and other components on the board. This can come either from VIN via an on-board regulator, or be supplied by USB or another regulated 5 V supply.
- **3V3 :** (Diecimila-only) A 3.3 volt supply generated by the on-board FTDI chip.
- **GND :** Ground pins.

Other pins

- **AREF :** Reference voltage for the analog inputs. Not currently supported by the Arduino software.
- **Reset :** Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.
- It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button.
- The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7 V, however, the 5 V pin may supply less than five volts and the board may be unstable. If using more than 12 V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.

Arduino Uno R3 programming

- The programming of an Arduino Uno R3 can be done using IDE software. The microcontroller on the board will come with pre-burned by a boot loader that permits to upload fresh code without using an exterior hardware programmer.
- The communication of this can be done using a protocol like STK500.
- We can also upload the program in the microcontroller by avoiding the boot loader using the header like the In-Circuit Serial Programming.

6.1.2 Raspberry Pi

- A Raspberry Pi is a credit card-sized computer originally designed for education, inspired by the 1981 BBC Micro.
- Creator Eben Upton's goal was to create a low - cost device that would improve programming skills and hardware understanding at the pre - university level.
- The Raspberry Pi is slower than a modern laptop or desktop but is still a complete Linux computer and can provide all the expected abilities that implies, at a low-power consumption level.

Versions	Remarks
Raspberry Pi 1	• The original Raspberry Pi had 256 Mb of RAM, which increased to 512 MB in a later revision. • It has a 26-way GPIO connector
Pi Zero	• The Pi Zero includes the GPIO connector, but the header pins are not soldered
Raspberry Pi 2	• The Raspberry Pi 2 swapped the single-core processor for a much faster quad-core processor and increased the memory to 1 GB RAM
Raspberry Pi 3	• The Raspberry Pi 3 changes the processor to an even more powerful 64-bit processor. • It also adds Wi-Fi and bluetooth which previously needed to be added as a USB device. • The Raspberry Pi 3 Model B was launched in February 2016.

- To get the Raspberry Pi working an SD card needs to be prepared with the Linux operating system installed.
- Raspberry Pi users have made many creative and impressive projects using this device. It can also be programmed to assist in 'housekeeping' your network by functioning as NAS, LDAP server, web server, media server, DNS server etc.
- The Raspberry Pi Foundation recommends Python. Any language which will compile for ARMv6 can be used. Installed by default on the Raspberry Pi : C, C++, Java, Scratch and Ruby.

6.1.3 About the Board

- Fig 6.1.2 shows the Raspberry Pi board. The Raspberry Pi does not have a separate CPU, RAM or GPU. Instead they are all squeezed into one component called a system on Chip or SoC unit.
- Raspberry Pi is open hardware with the exception of its primary chip, the Broadcomm SoC which runs the main components of the board - CPU, graphics, memory, USB controller etc.

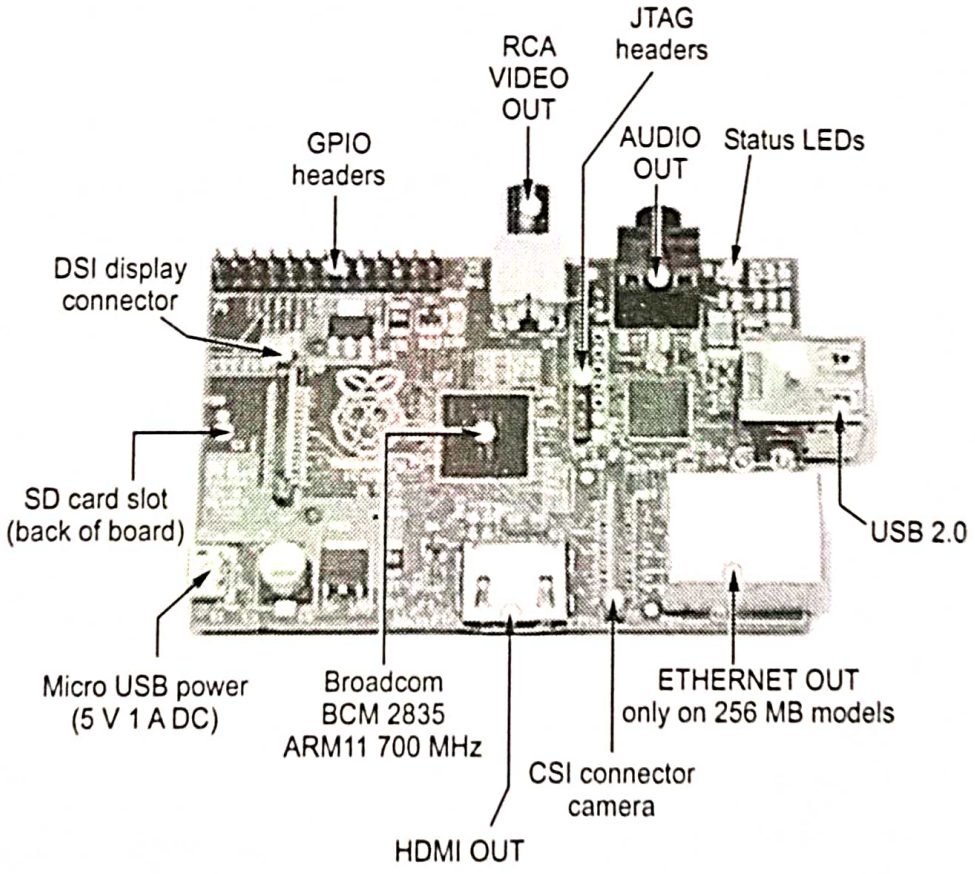

Fig 6.1.2 (a) Raspberry Pi circuit board

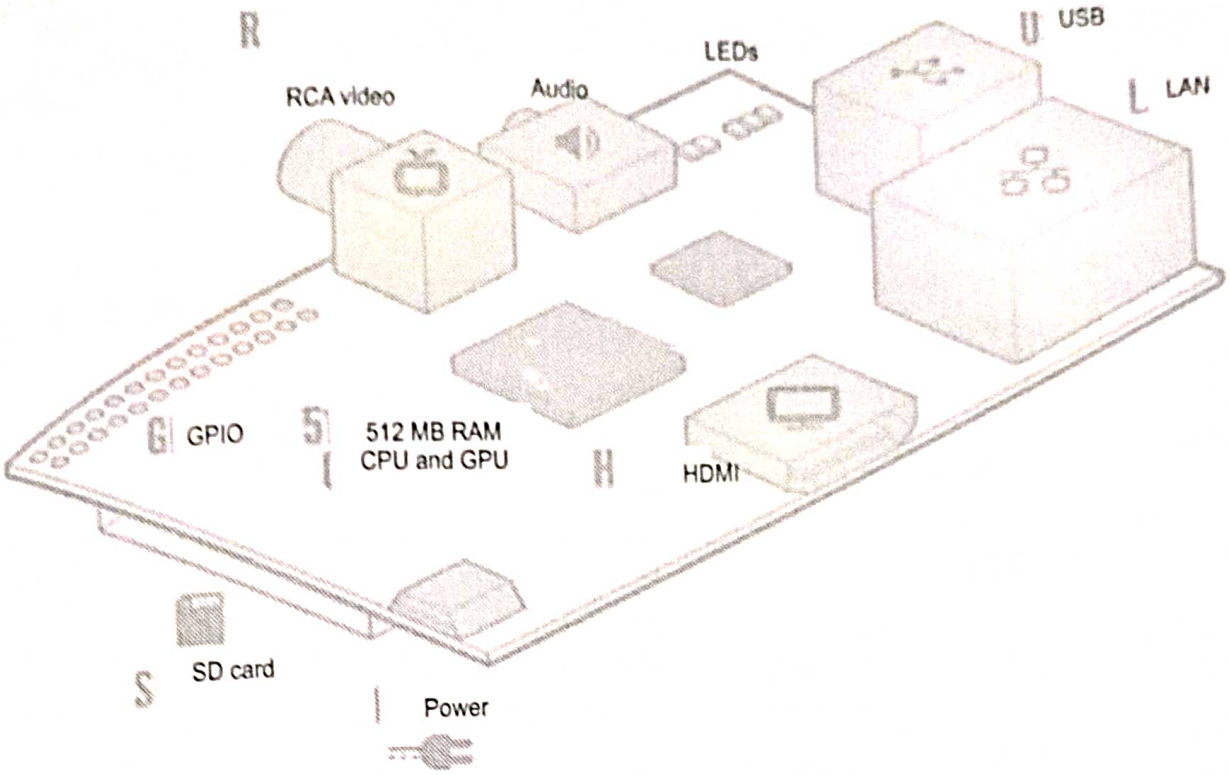

Fig 6.1.2 (b) Block diagram

- All of these Raspberry Pi Models share the following features :
  1. Operating systems : Raspbian RaspBMC, Arch Linux, Rise OS, OpenELEC Pidora
  2. Video output : HDMI Composite RCA
  3. Supported resolutions : 640x350 to 1920x1200, including 1080p, PAL and NTSC standards
  4. Power source : Micro USB

Components	Description
Processor	• Raspberry Pi uses an ARM processor which is also installed in a wide variety of mobile phones. • This CPU is single core, however it does have a co-processor to perform floating point calculations
Memory	• Model B Raspberry Pi has 512 MB SDRAM (Synchronous Dynamic RAM). • It store programs that are currently being run in the CPU
USB ports	• Board has two USB ports. USB port can provide a current upto 100 mA • Using powered hub, it is possible to connect more devices.

HDMI output	• High Definition Multimedia Interface (HDMI) supports high-quality digital video and audio through a single cable. • It is also possible to connect a computer monitor with a DVI connection to HDMI using a converter.
Composite video output	• It support composite video output with RCA jack and also support PAL and NTSC. • The TVDAC pin can be used to output composite video.
Audio output	• Audio output jack is 3.5 mm. • This jack is used for providing audio output to old television along with the RCA jack for video
GPIO pins	• Both models have a total of 26 GPIO pins, organized into one pin header, named the P1 header • The newer Raspberry Pi (model B revision 2) adds 8 more GPIO pins in a new pin header called P5
	• Not all the GPIO pins are programmable. Some of them are 5.0 VDC or 3.3 VDC positive power pins, some of them are negative ground pins and a few of them are marked DNC (do not connect). • The P1 header has 17 programmable pins and the P5 header adds 4 more. • Fig 6.1.3 shows GPIO pin header. • Reading from various environmental sensors. Writing output to dc motors, LEDs for status.
Power input	• Micro - USB connector is used for power input
Status LED	• It has five status LED.
CSI	• Camera Serial Interface (CSI) can be used to connect a camera module to Raspberry Pi
SD card slot	• This card is used for loading operating system

- The Raspberry Pi comes with a set of 26 exposed vertical pins on the board. These pins are a General Purpose Input /Output interface that is purposely not linked to any specific native function on the Raspberry Pi board.

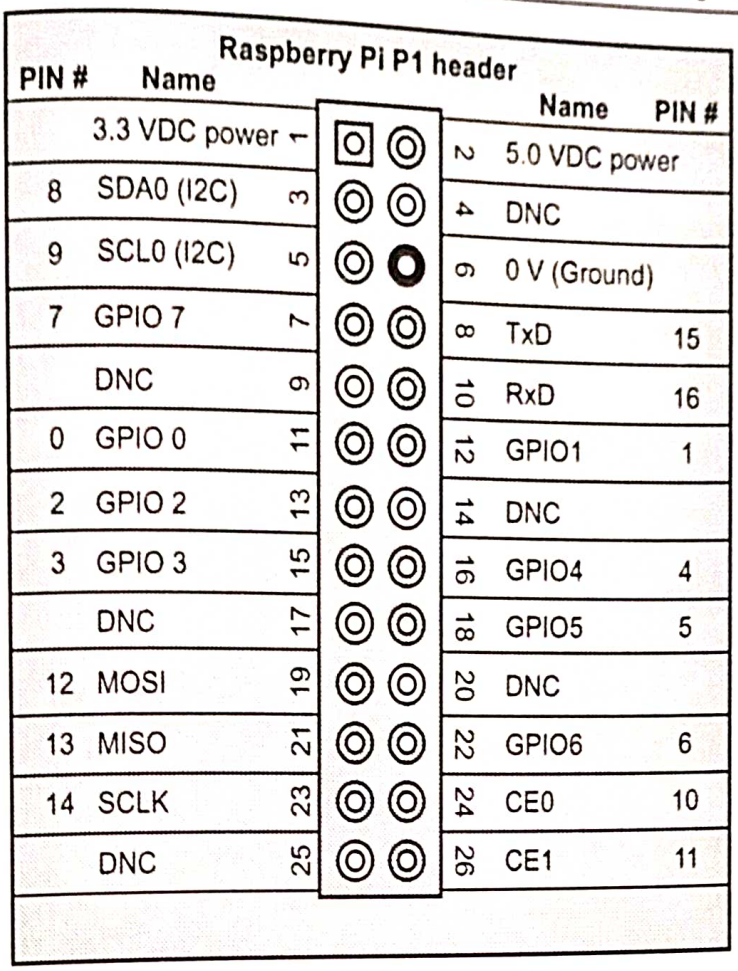

Fig. 6.1.3 GPIO pin header

- Instead, the GPIO pins are there explicitly for the end user to have low-level hardware access directly to the board for the purposes of attaching other hardware boards, peripherals, LCD display screens and other hardware devices to the Pi.

The status LEDs

Status LED	Color	Functions
ACT	Green	Lights when the SD card is accessed (marked OK on earlier boards)
PWR	Red	Hooked up to 3.3 V power
FDX	Green	On if network adapter is full duplex
LNK	Green	Network activity light
100	Yellow	On if the network connection is 100 Mbps

- The Raspberry Pi draws its power from a microUSB port and requires a microUSB-to-AC adapter. Because the Pi is a micro computer and not simply a cell phone getting a battery topped off, you need to use a high quality charger with

stable power delivery that provides a consistent 5 V with at least 700 mA minimum output for older model units and 2.5 A for the Pi 3.

6.1.4 Linux on Raspberry Pi

- There are several unix like operating systems for the RPI and there is an operating system called RISC OS that has its origin at the developers of the first ARM chips.
- The Raspberry Pi Foundation recommends the use of the following Linux Distributions :
  1. Debian 7
  2. Raspbian
  3. Arch Linux ARM
  4. QtonPi
- Raspbian is a free operating system based on Debian optimized for the Raspberry Pi (RPI) hardware.
- The default command prompt on the Pi consists of four components shown in Fig. 6.1.4.

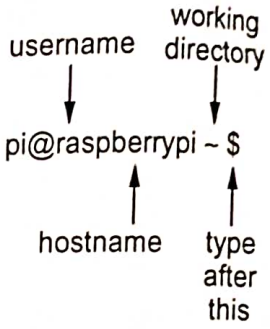

Fig. 6.1.4 Command prompt

- Raspbian is the desired operating system for the Raspberry Pi. In order to download and install the operating system onto our Raspberry Pi; you will need Raspbian, Win32DiskImager and USB memory card reader.
  1. Download both Raspbian and Win32DiskImager and save somewhere easily accessible
  2. Plug the USB memory card reader into computer
  3. Open Win32DiskImager
  4. Find the location of the image file and the memory card
  5. Click "Write"

Logging In

- Now it is time to turn on our Raspberry Pi. When the memory card, HDMI lead, Ethernet cable, mouse and keyboard are plugged in, plug in the power lead.
- As soon as you do this. You screen should be black and filled with white text. This will be visible every time you turn on your raspberry pi.

- Wait until your screen reads "raspberrypi login :"

Username = pi [ENTER]

Password = raspberry [ENTER]

```
Debian GNU/Linux wheezy/sid raspberrypi tty1

raspberrypi login: pi
Password:
Last login: Tue Aug 21 21:24:50 EDT 2012 on tty1
Linux raspberrypi 3.1.9+ #168 PREEMPT Sat Jul 14 18:56:31 BST 2012 armv6l

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.

Type 'startx' to launch a graphical session

pi@raspberrypi ~ $
```

Starting the Raspbian GUI

- GUI stands for Graphical User Interface and is a type of operating system. It is the most common type of user interface as it is a very 'friendly' way for people to interact with the computer. It makes use of pictures, graphics, icons and pointers, hence the name 'Graphical' User Interface. Fig. 6.1.5 shows Rasbian Linux desktop

1. Type the line : "startx"

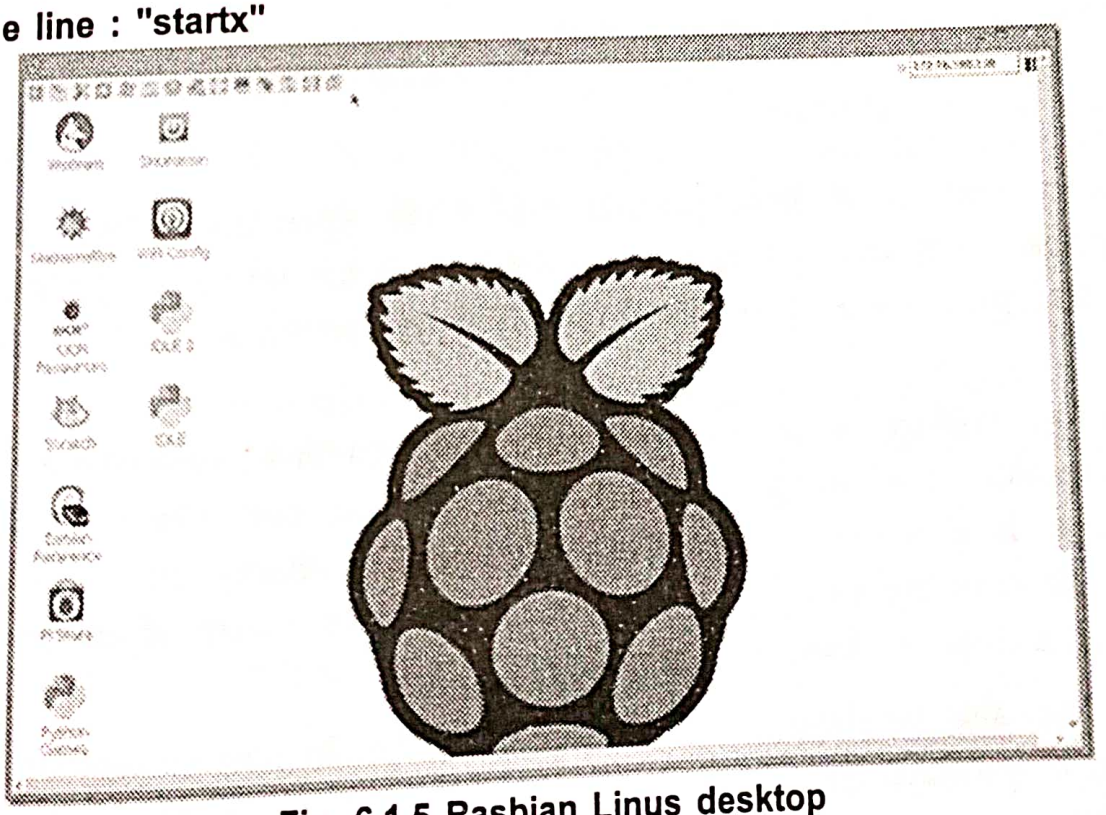

Fig. 6.1.5 Rasbian Linus desktop

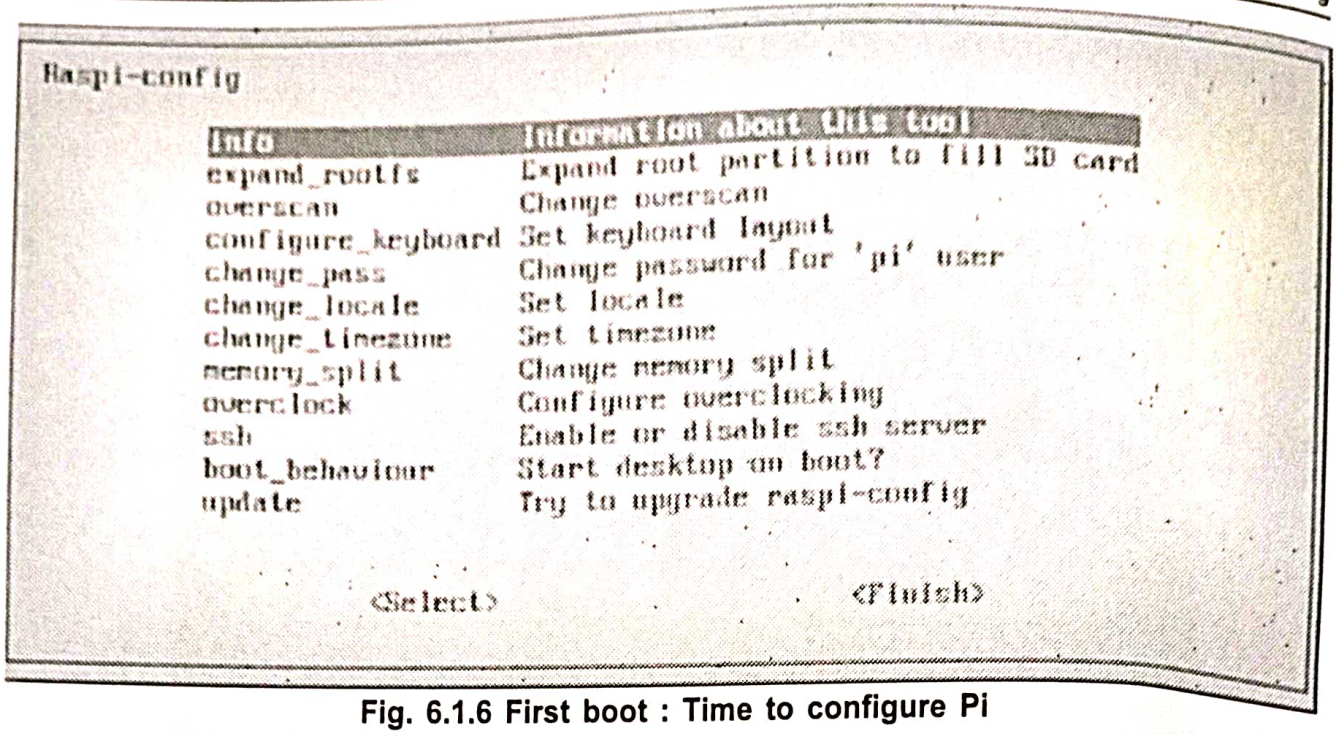

Fig. 6.1.6 First boot : Time to configure Pi

6.1.5 Difference between Raspberry Pi and Desktop Computers

- In Raspberry Pi, operating system is installed on SD card whereas in desktop computer, operating system is installed in hard disk.
- Raspberry Pi does not have their own CPU and RAM.
- Processing power of Raspberry Pi is less as compared to desktop computers.
- Raspberry Pi uses less power than desktop computers.

6.2 Study of IoT Sensor

- Sensor converts a physical quantity into a corresponding voltage. Sensor is a device that when exposed to a physical phenomenon (temperature, displacement, force, etc.) produces a proportional output signal (electrical, mechanical, magnetic, etc.).
- The term transducer is often used synonymously with sensors. Sensor is a device that responds to a change in the physical phenomenon. On the other hand, a transducer is a device that converts one form of energy into another form of energy. Sensors are transducers when they sense one form of energy input and output in a different form of energy.
- Sensors can also be classified as passive or active. In passive sensors, the power required to produce the output is provided by the sensed physical phenomenon itself whereas the active sensors require external power source.
- In embedded system, sensor and actuators are used for controlling the system. Sensors are connected to input port. Actuators are connected to output port.

- Sensor captures the changes in the environmental variable. Middle system process the information. Actuators are changed according to the input variable. It displays the output.
- Example of control is air conditioner system. It controls the room temperature to a specified limit.
- Deflection : The signal produces some physical (deflection) effect closely related to the measured quantity and transuded to be observable.
- Null : The signal produced by the sensor is counteracted to minimize the deflection. That opposing effect necessary to maintain a zero deflection should be proportional to the signal of the measurand.
- Here, the output is usually an 'electrical quantity' and measurand is a 'physical quantity, property or condition which is to be measured'.
- A sensor is a 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.
- A sensor can be very small and itself can be a trackable devices. The sensor itself, if not connected is not part of the IoT or WSN value chain.

Specifications of sensor :

1. **Accuracy :** Error between the result of a measurement and the true value being measured.
2. **Resolution :** The smallest increment of measure that a device can make.
3. **Sensitivity :** The ratio between the change in the output signal to a small change in input physical signal. Slope of the input - output fit line.
4. **Repeatability / Precision :** The ability of the sensor to output the same value for the same input over a number of trials.
5. **Bandwidth :** The frequency range between the lower and upper cut-off frequencies, within which the sensor transfer function is constant gain or linear.

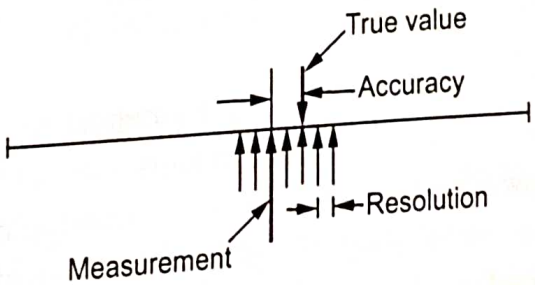

Fig. 6.2.1 Accuracy vs Resolution

6.2.1 Sensor Component

- Fig. 6.2.2 shows sensor node. A basic sensor node comprises five main components.

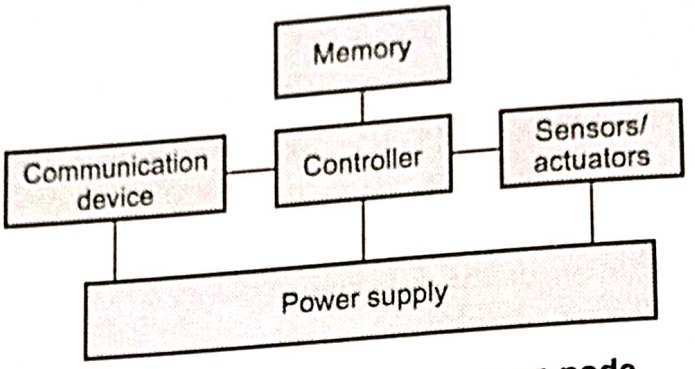

Fig. 6.2.2 Components of sensor node

1. **Controller :** A controller to process all the relevant data, capable of executing arbitrary code.
2. **Memory :** Some memory to store programs and intermediate data; usually, different types of memory are used for programs and data.
3. **Sensors and actuators :** The actual interface to the physical world : devices that can observe or control physical parameters of the environment.
4. **Communication :** Turning nodes into a network requires a device for sending and receiving information over a wireless channel.
5. **Power supply :** As usually no tethered power supply is available, some form of batteries is necessary to provide energy. Sometimes, some form of recharging by obtaining energy from the environment is available as well.

- For actual communication, both a transmitter and a receiver are required in a sensor node. The essential task is to convert a bit stream coming from a microcontroller and convert them to and from radio waves. For practical purposes, it is usually convenient to use a device that combines these two tasks in a single entity. Such combined devices are called **transceivers.**

6.2.2 Sensor Types

- Sensors can be roughly categorized into three categories :

1. **Passive, omnidirectional sensors :** These sensors can measure a physical quantity at the point of the sensor node without actually manipulating the environment by active probing. In this sense, they are passive. Moreover, some of these sensors actually are self-powered in the sense that they obtain the energy they need from the environment.

- Typical examples for such sensors include thermometer, light sensors, vibration, microphones, humidity, mechanical stress or tension in materials, chemical sensors sensitive for given substances, smoke detectors, air pressure and so on.

2. **Passive, narrow - beam sensors :** These sensors are passive as well, but have a well-defined notion of direction of measurement. A typical example is a camera, which can "take measurements" in a given direction, but has to be rotated if need be.

3. **Active sensors :** This last group of sensors actively probes the environment, for example, a sonar or radar sensor or some types of seismic sensors, which generate shock waves by small explosions. These are quite specific, triggering an explosion is certainly not a lightly undertaken action and require quite special attention.

- **Active sensors :** Require an external source of power (excitation voltage) that provides the majority of the output power of the signal
- **Passive sensors :** The output power is almost entirely provided by the measured signal without an excitation voltage.
- **Digital sensors :** The signal produced or reflected by the sensor is binary.
- Analogue sensors produce a continuous output signal or voltage which is generally proportional to the quantity being measured.

6.2.3 Difference between Actuator and Sensor

Actuators	Sensor
It is output device.	It is input device.
Converts an electrical signal to a physical output.	Converts a physical parameter to an electrical output.
A component of a machine that is responsible for moving and controlling mechanism.	A device that detects events or changes in the environment and send that information to another electronic device.
It helps to control the environment or physical changes.	It help to monitor the changes in the environment.

6.3 IoT Design

- The rise of the Internet of Things has led to an explosion of new sensor computing plat - forms. The complexity and application domains of IoT devices range from simple self - monitoring devices in vending machines to complex interactive devices with artificial intelligence in smart vehicles and drones.

- As IoT developers wish to meet more aggressive platform objectives and protect market share through feature differentiation, they must choose between low-cost, and low-performance CPU-based Commercial - Off - The - Shelf (COTS) systems, and high-performance custom platforms with hardware accelerators such as GPU and FPGA.
- An IoT platform facilitates communication, data flow, device management, and the functionality of applications. The goal is to build IoT applications within an IoT platform framework. The IoT platform allows applications to connect machines, devices, applications, and people to data and control centers.
- Home automation can be described as introduction of technology within the home environment to provide convenience, comfort, security and energy efficiency to its occupants.
- A home automation system can involve switching off electrical appliances like air-conditioners or refrigerators when a desired temperature has been reached, then switching on again when the temperature has crossed a certain value
- Fig 6.3.1 shows design methodology steps.
- Sensors : Sensors are the eyes of a home automation system. They "see" the environment and convert what they find into an electrical quantity that can be measured by a microcontroller or system processor.

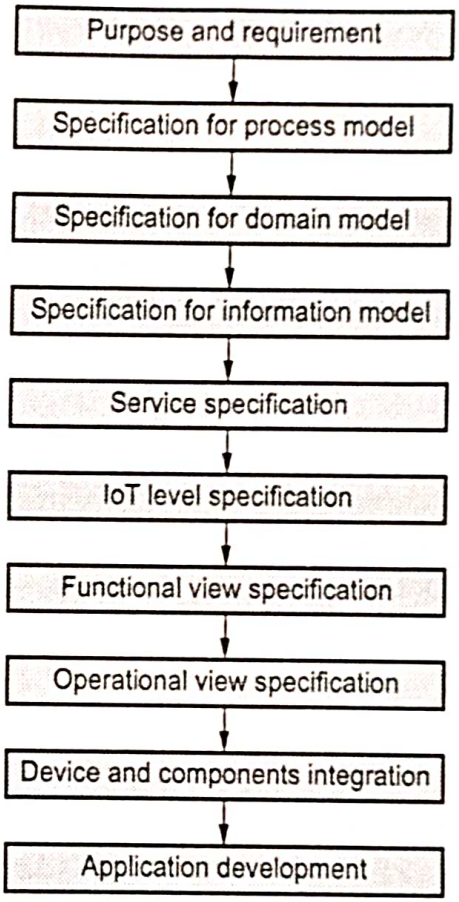

Fig. 6.3.1 Design methodology steps

- Remote Connectivity : Depending on need and various design considerations, users may need to be able to control the system and appliances remotely.

6.3.1 Purpose and Requirement Specification

- Purpose : Using web application, user can remotely control the home electronic devices (light and air conditioner)
- Behavior : Home automation system will works in two modes : manual and automatic mode. In manual mode, web system provides options of manually and

remotely switching ON/OFF the light. In automatic mode, sensor is provided in the room. System measures the darkness of the room and light is ON when darkness increases. Room temperature is measured by another sensor, air conditioner is ON when room temperature is increases.

- System Management Requirement : Remote monitoring and control function is provided by system
- Data analysis requirement : Data analysis is based on local data.
- Application deployment Requirement : Application is installed on local device but it support remote operation.
- Security requirement : Basic authentication mechanism must be provided.

6.3.2 Process Specification

- Process specification is defined after requirement step. Draw the use case based upon first stage. Fig. 6.3.2 shows process specification diagram.

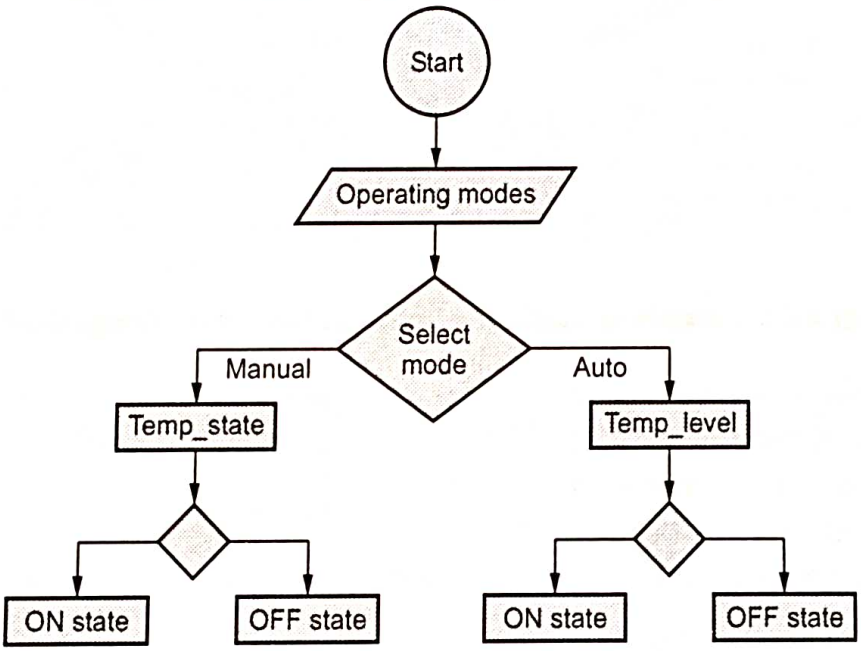

Fig. 6.3.2 Process specification diagram

- In auto mode, system monitors the room temperature and light level and takes the decision for switching ON / OFF.
- In manual mode, user performs the operation.

6.3.3 Domain Model Specification

- A domain model defines the main concepts of a specific area of interest.
- The IoT is a support infrastructure for enabling objects and places in the physical world to have a corresponding representation in the digital world. Fig 6.3.3 shows mapping concept of physical world to virtual world.

- As interaction with the physical world is the key for the IoT; it needs to be captured in the domain model (DM). The DM defines the main concepts of the Internet of Things and the relations between these concepts.
- User and a physical entity are two concepts that belong to the domain model. A User can be a human user, and the interaction can be physical.

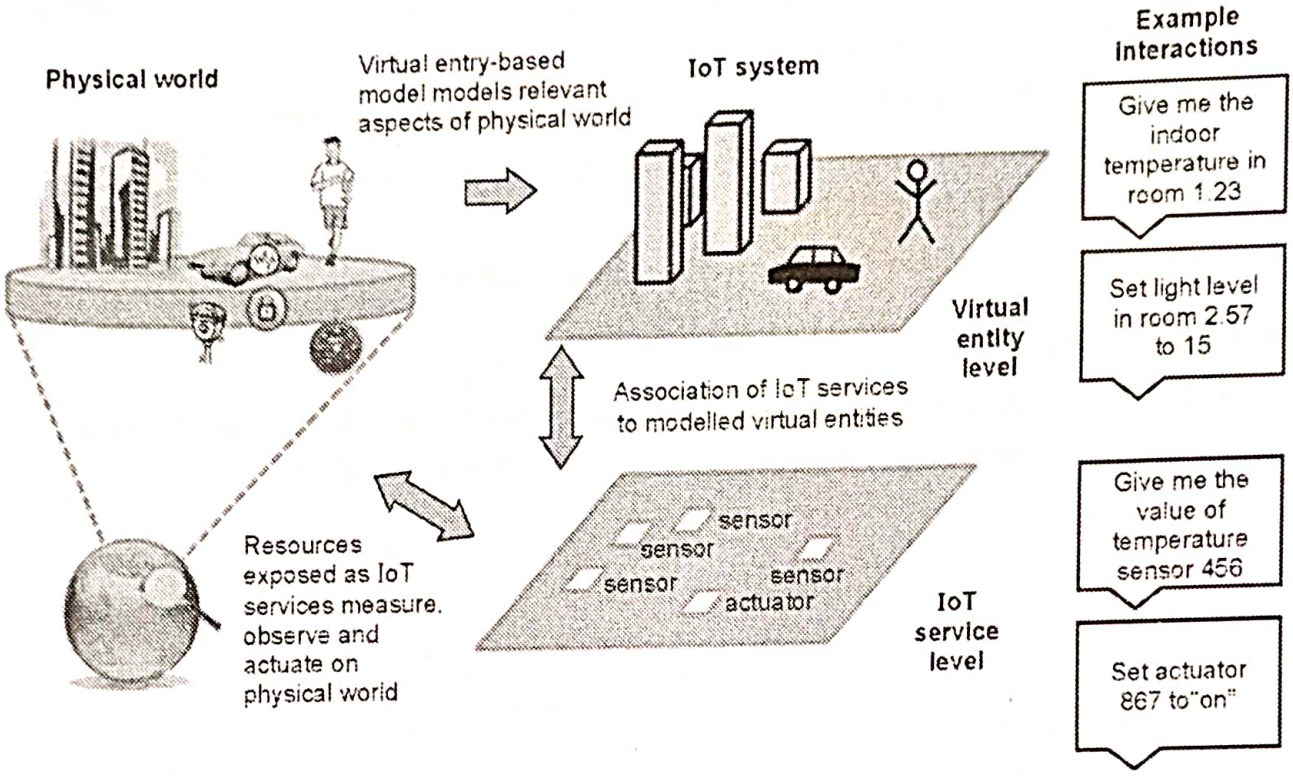

Fig 6.3.3 : Mapping concept of physical world to virtual world

- The physical interaction is the result of the intention of the human to achieve a certain goal. Fig 6.3.4 shows IoT domain model.
- A Physical entity, as the model shows, can potentially contain other physical entities; for example, a building is made up of several floors, and each floor has several rooms.
- A Physical entity is represented in the digital world as a Virtual entity. A

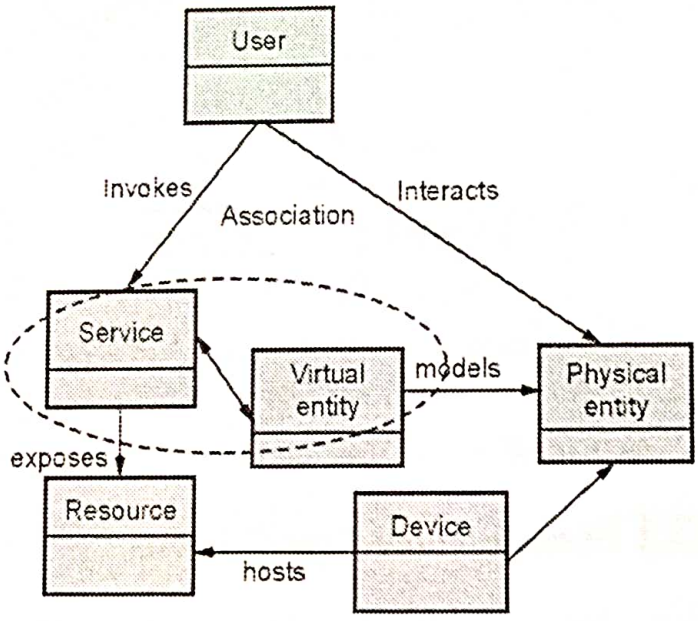

Fig. 6.3.4 : IoT Domain model

Virtual entity can be a database entry, a geographical model, an image or avatar, or any other Digital Artifact.

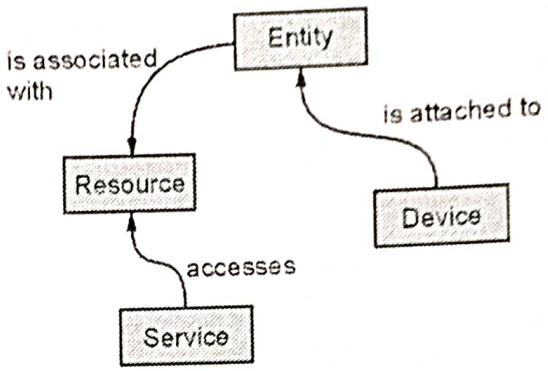

Fig. 6.3.5 : Key concepts and interaction in IoT model

- The relations between services and entities are modeled as associations. These associations could be static, e.g. in case the device is embedded into the entity; they could also be dynamic, e.g., if a device from the environment is monitoring a mobile entity. These identified concepts of the IoT domain and the relations between them are depicted in Fig 6.3.5.
- One physical entity can be represented by multiple virtual entities, each serving a different purpose. For the IoT domain Model, three kinds of device types are the most important :

1. **Sensors :** These are simple or complex devices and contain a transducer that converts physical properties such as temperature into electrical signals. These devices include the necessary conversion of analog electrical signals into digital signals.
2. **Actuators :** These devices that involve a transducer that converts electrical signals to a change in a physical property.
3. **Tags :** Tags in general identify the Physical entity that they are attached to.

- Home Automation System Example :
  1. **Physical entity :** Room in the home and room temperature
  2. **Device :** Single board mini computer with sensor and relay switch
  3. **Resources :** Operating system which runs on mini computer
  4. **Services :** Mode selection, controller service which runs services on the device, retrieve the room temp.

6.3.4 Information Model Specification

- An abstract description (UML diagram or ontology) for explaining information about elements or concepts defined in the IoT Domain Model.
- The information model models domain model concepts that are to be explicitly represented and manipulated in the digital world. In addition the information model explicitly models relations between these concepts.

- Fig 6.3.6 shows information model. The information model is a meta model that provides a structure for the information. This structure provides the basis for defining the functional interfaces.

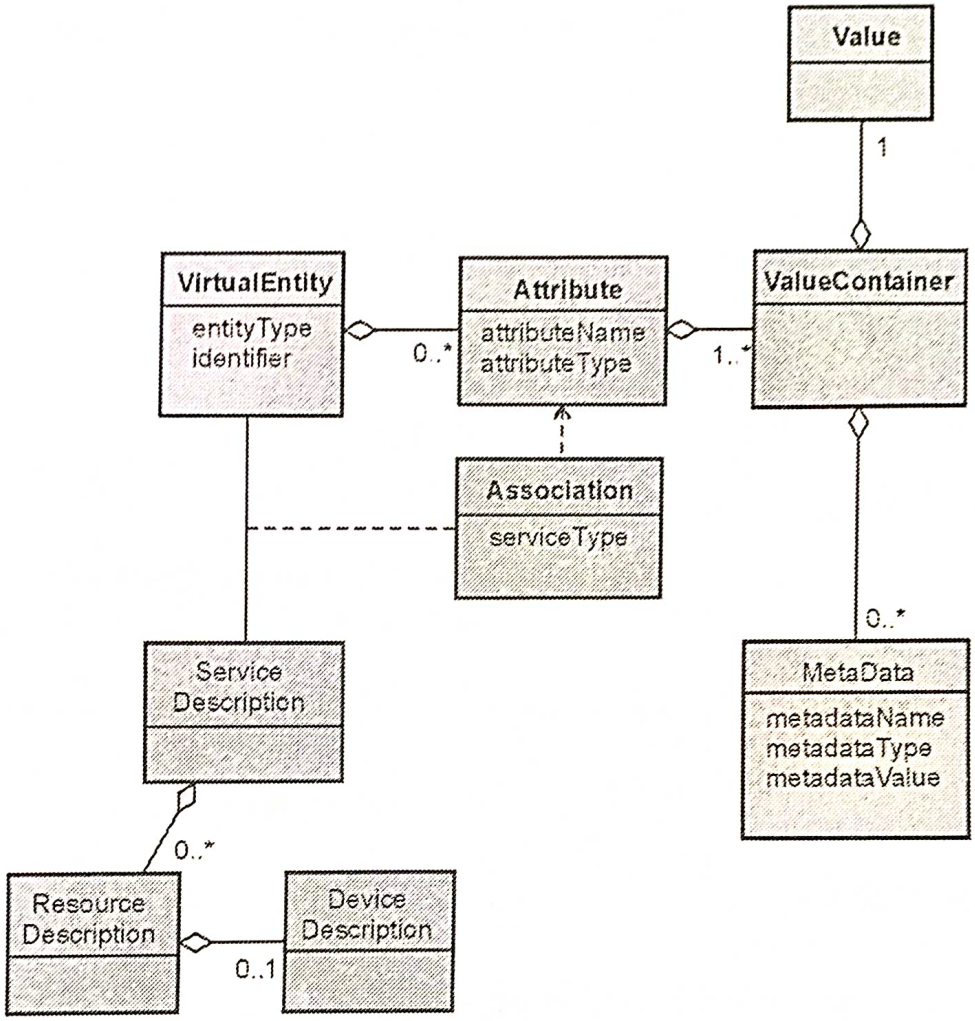

Fig 6.3.6 : IoT information model

- IoT Information Model is represented using Unified Modeling Language (UML) diagram. The IoT Information Model maintains the necessary information about Virtual Entities and their properties or attributes.
- The information model for an object can contain information about the objects structure and resource types. This can enable APIs to automatically be composed by middleware and automatically consumed by application software.
- Additional metadata can indicate context, such as geographical location, and bindings, such as message protocols and event handlers, as well as access control information.
- The IoT Information Model describes Virtual Entities and their attributes that have one or more values annotated with meta-information or metadata. The attribute values are updated as a result of the associated services to a Virtual Entity.

6.3.5 Service Specification

- Service specification in IoT defines list of services provided by the system. It includes service type, input/output, service endpoints, schedulers, service effects and service preconditions.
- State and attributes are defined in process specification and information model. For example in home automation system, mode setting is an example of mode service. User can set the auto or manual mode.
- Mode services sets mode to auto or manual or retrieves the current mode.
- In manual mode, the controller service, retrieves the current state from the database and switches the air conditioner ON/OFF.

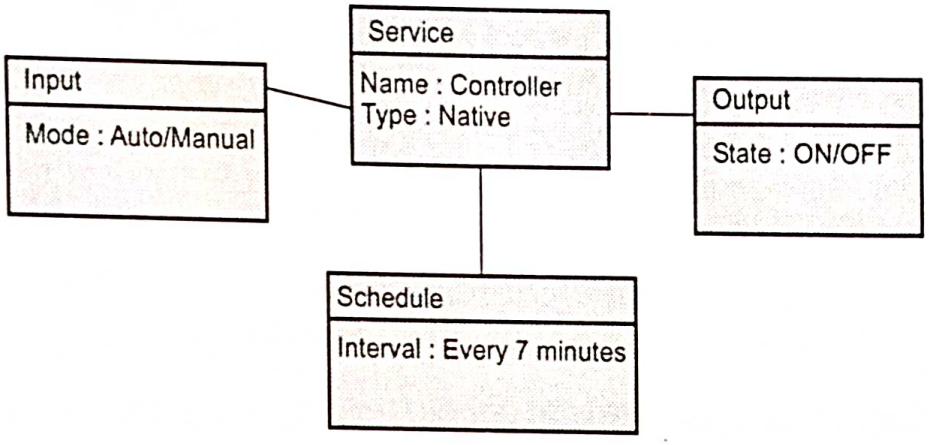

Fig 6.3.7 : Home automation Controller service

6.3.6 IoT Level Specification

- IoT development level are six types.

IoT Level 1	Single node, perform sensing, perform analysis and hosts the application
IoT Level 2	Single node, perform sensing, perform local analysis
IoT Level 3	Single node, data is analyzed in cloud and application is cloud based
IoT Level 4	Multiple Node, perform local analysis, application is cloud based
IoT Level 5	Multiple end node and coordinator node,
IoT Level 6	Multiple independent node, perform sensing, send data to cloud

6.3.7 Functional View Specification

- Functional View describes the system's runtime Functional Components, their responsibilities, default functions, interfaces and primary interactions. The Functional View derives from the Functional Model and reflects the developer's perspectives on the system.

- It will need to be extended with all identified (and recommended) new profile-specific Functional Components including their interfaces and a list of Sequence Charts illustrating recommended usage of those components.
- Fig 6.3.8 shows functional view.

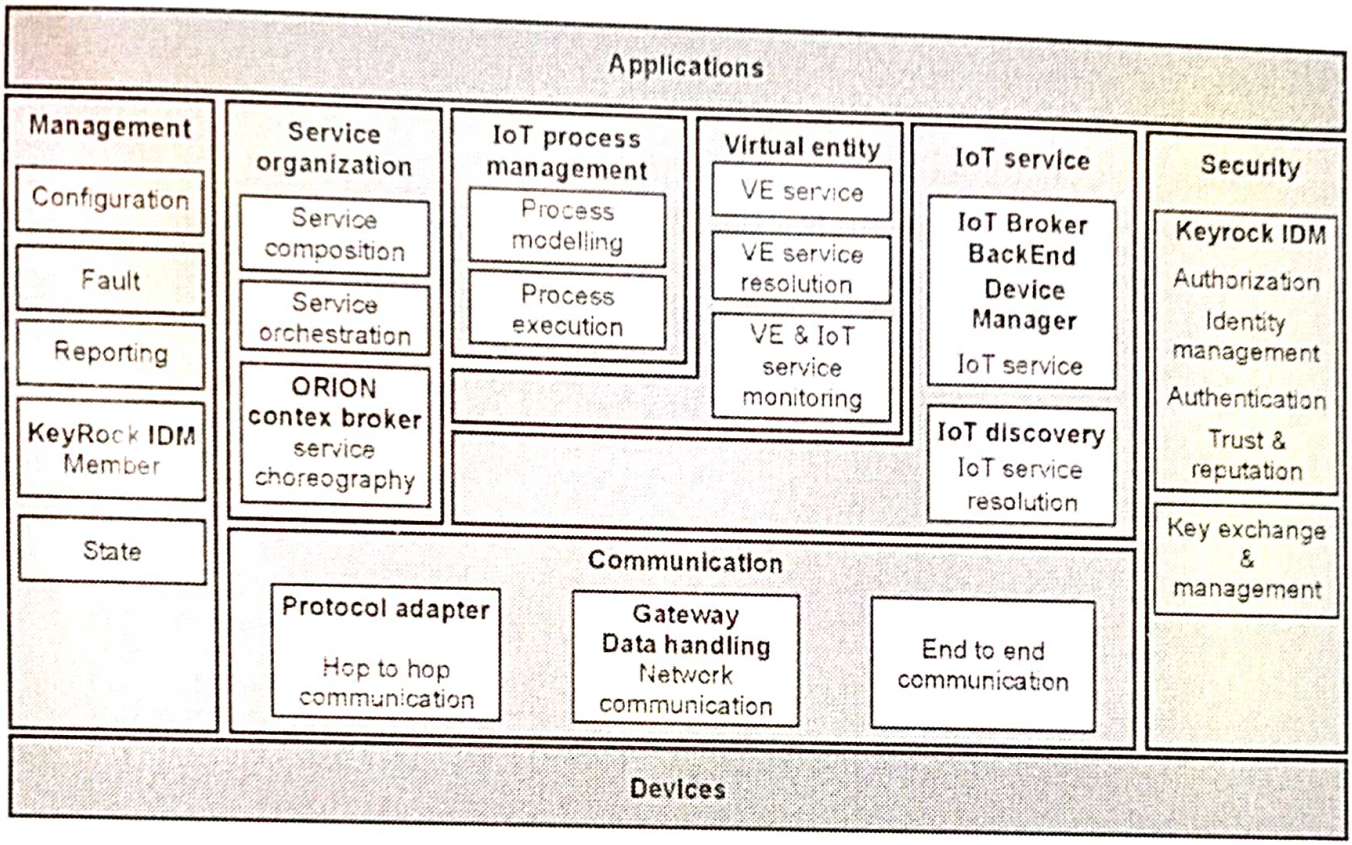

Fig 6.3.8 : Functional view

- The viewpoints used for constructing the IoT Functional View are hence :
  1) The Unified Requirements;
  2) The IoT Functional Model
- Once all Functional Components are defined, the default function set, system use cases, sequence charts and interface definitions are made.
- **Device and application functional group :** Device functional components contains the sensing, actuation tag, processing and storage components. Application functional group contains standalone application.
- **Communication functional group :** It contains the components for end-to-end communication, network communication, and Hop - by - Hop communication.

Communication type	Description
End-to-End Communication FC	1) Responsible for end-to-end transport of application layer messages through diverse network and MAC/physical layers
	2) Used with mesh radio networking technologies such as IEEE 802.15.4
	3) The End-to-End FC interfaces the Network FC on the "southbound" direction.
Hop-by-hop FC	1) Responsible for transmission and reception of physical and MAC layer frames to/from other devices.
	2) Two interfaces used : one "southbound" to/from the actual radio on the device, and another for "northbound" to/from the Network FC in the Communication FG.
Network FC	1) Responsible for message routing and forwarding and the necessary translations of various identifiers and addresses.
	2) Network FC interfaces the End-to-End Communication FC on the "northbound" direction and the Hop-by-Hop Communication FC on the "southbound" direction.

- **IoT service functional group :** It consists of IoT Service FC and the IoT Service Resolution FC. Various service implementations are covered in service FC and service resolution FC contains the necessary functions to realize a directory of IoT Services that allows dynamic management of IoT service descriptions.
- **Virtual entity functional group :** The Virtual Entity FG contains functions that support the interactions between Users and Physical Things through Virtual Entity services.
- **Process management FG :** Provides the functional concepts necessary to conceptually integrate the IoT world into traditional (business) processes.
- The process modeling FC which provides the tools required for modeling IoT-aware business processes that will be serialized and executed in the Process Execution FC, which is responsible for deploying process models to the execution environments.
- Service organization FG : Acts as a communication hub between several other Functional Groups by composing and orchestrating Services of different levels of abstraction.
- The service orchestration FC resolves the IoT Services that are suitable to fulfill service requests coming from the Process Execution FC or from Users while the

Service composition FC is responsible for creating services with extended functionality by composing IoT services with other services.

- Service choreography FC offers a broker that handles Publish/Subscribe communication between services.
- Virtual Entity FG : Provides functionality for the interaction of VEs with the IoT system, for VE look-up and discovery and for providing information concerning VEs. The VE Resolution FC provides discovery services for associations between VEs and IoT.
- VE and IoT Service Monitoring FC is responsible for automatically finding new associations based on service descriptions and information about VE's. the VE Service FC handles entity services.
- Service FG : Provides IoT services as well as functionalities for discovery, look-up, and name resolution of IoT Services.
- **Security FG :** It is responsible for security and privacy matters in IoT-A-compliant IoT systems.
  1) The Authorization FC is used to apply access control and access policy management while, the Authentication FC is used for user and service authentication.
  2) Key Exchange and Management (KEM) FC enables secure communications ensuring integrity and confidentiality by distributing keys upon request in a secure way.
- **Management FG :** It is responsible for the composition and tracking of actions that involve the other FGs.
  1) Configuration FC is responsible for initializing the system's configuration.
  2) The Fault FC is used to identify, isolate, correct and log faults that occur in the IoT system.
  3) The Member FC is responsible for the management of the membership of any relevant entity
- The reporting FC generates reports about the system and, finally, the State FC can change or enforce a particular state on the system by issuing a sequence of commands to the other FCs.

6.3.8 Operational View Specification

- Deployment and operational view depends on the specific actual use case and requirements. Smart object in the IoT uses different methods for communication using different technology.

- Hence the deployment and operation view is very important to address how actual system can be realized by selecting technologies and making them communicate and operate in a comprehensive way.
- It provides an IoT Reference Model with a set of guidelines to application users. The different design choices that they have to face while designing the actual implementation of their services.
- The viewpoints used in the deployment and operation view are the following :
  1) The IoT domain model diagram is used as a guideline to describe the specific application domain.
  2) The functional model is used as a reference to the system definition.
  3) Network connectivity diagrams can be used to plan the connectivity topology to enable the desired networking capability of the target application; at the deployment level, the connectivity diagram will be used to define the hierarchies and the type of the sub-networks composing the complete system network;
  4) Device descriptions can be used to map actual hardware on the service and resource requirements of the target system.

6.3.9 Device and Component Integration

- In this step we have to integrate the devices and components. The devices and components used in home automation examples are raspberry Pi, sensor, laser pointer, light dependent resistor etc.

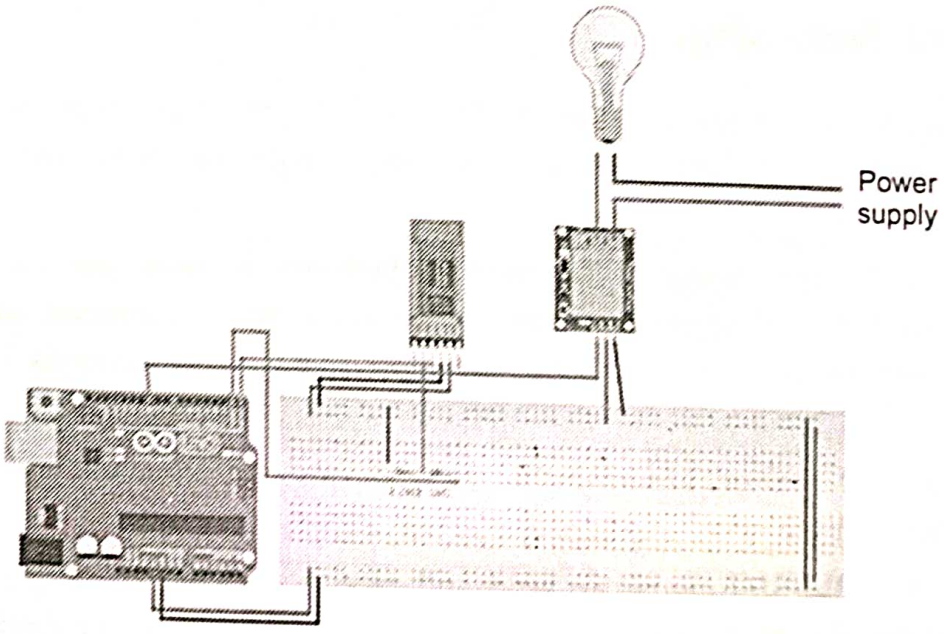

Fig 6.3.9 Schematic diagram for home automation

- Now is a good time to become acquainted with Raspberry Pi and its potential for use in IoT projects. When developing an innovative use for this device, you will need to come up with a software design solution that addresses your end-user requirements. You can accomplish this more easily when you have a better sense of the Raspberry Pi capabilities being demonstrated out in the field now.
- From controlling the room lights with your smartphone to scheduling events to occur automatically, home automation has taken convenience to a whole new level. Instead of using mechanical switches, you can now conveniently control all the devices in your home from your fingertips.

6.3.10 Application Development

- Application development is final stage in developing IoT application. In web application, it display mode control : Auto and Manual
- Two modes are provided : auto mode and manual mode.
- If auto mode is enabled the light control in the web application is disabled and it reflects the current state of light.
- If auto mode disabled, the light control is enabled and it is used for manually controlling light and air conditioner.

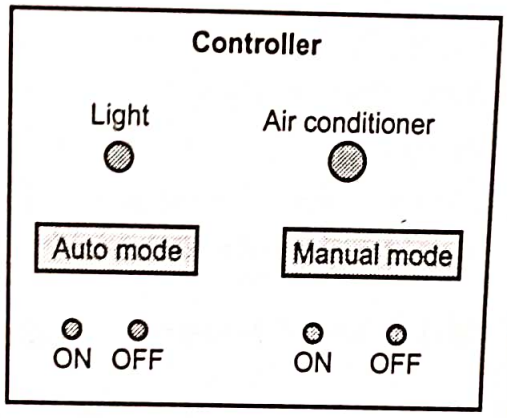

Fig 6.3.10

6.4 SIM Card Technology

- IoT sim cards are designed for smart devices. IoT sim cards allow devices to be connected and able to communicate with each other. IoT SIM card also called M2M card.
- IoT SIM cards are designed to connect between devices and between two machines' northbound apps. As a result, the automotive, industrial, logistics and telematics and medical industries are the most common applications for IoT SIM cards.
- Traditional SIM cards are mostly utilized in consumer devices such as smartphones and tablets.
- Devices with M2M SIM cards can send and receive data across cellular networks. In IoT devices, the M2M SIM may share data directly with other devices and/or the software that manages the platform.

- IoT SIM cards are available with 2G, 3G, 4G/LTE, 5G, NB-IoT, LTE-M (Cat M1) network options and are compatible with thousands of wireless IoT devices.
- **Features :**
  1. M2M SIMs are more durable.
  2. M2M SIMs are remotely manageable.
  3. M2M SIMs support data plan aggregations.
  4. M2M SIMs may have fixed IP addresses
- Types of IoT SIM card :
- Fig. 6.4.1 shows IoT SIM card.

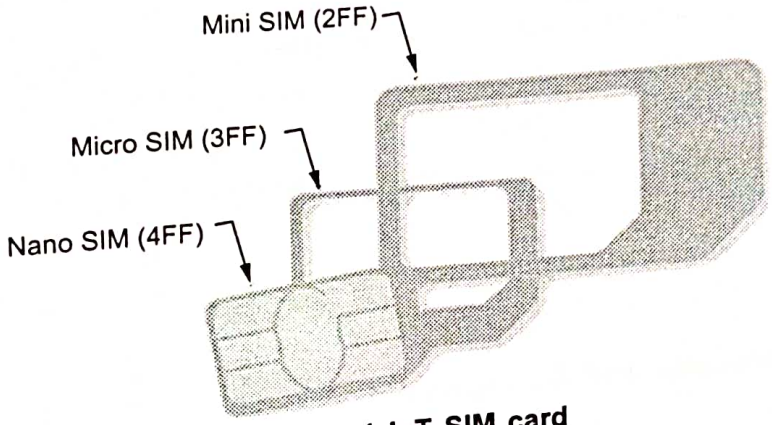

Fig. 6.4.1 IoT SIM card

1. Full - Size (1FF) : This is the largest M2M SIM card, about the size of a credit card. For the most part, it has been phased out by smaller, modern SIMs.
2. Mini - SIM (2FF) : This is the industry standard SIM card size, measuring 25 mm x 15 mm x 0.76 mm. It's typically used in devices like vehicles, vending machines and payment points.
3. Micro - SIM (3FF) : The micro-SIM is half the size of the mini and is used in portable devices like tablets, GPS, mHealth and other mobile IoT devices.
4. Nano - SIM (4FF) : The nano - SIM is 40 % smaller than the micro variation, making it great for small IoT devices. These SIMs have relatively little protection so they're not recommended for harsh environments.
5. eSIM (MFF2) : Embedded SIMs, also known as eSIMs, measure only 6 mm x 5 mm x 1 mm. These are the most popular IoT SIM because of their convenient size and durability. The card is not removable or interchangeable.

- SIM cards typically store a set of authentication credentials which help keep their data secure.

- Integrated Circuit Card Identifier (ICCID) is the identifier of the actual SIM itself. Authentication key (ki) is a 128 - bit value used as part of the authentication process for SIMs on the GSM mobile network.
- Local Area Identity (LAI) describes the distinct location area of an operator network. This data helps the network pinpoint the exact location of a SIM.
- **Applications :**
  1. Wearables - Smart watches and fitness trackers are already popular and smart glasses are beginning.
  2. Home automation devices - From smart lighting, to thermostat control and even windows, fridges and alarms.
  3. Agricultural sensors - Have revolutionized the way that farms are managed, with livestock tracking and weather and soil monitoring.
  4. Healthcare monitors - Provide doctors with a patients' physical data - like blood pressure and heart rate - So they can make more informed recommendations remotely.
  5. Logistics and fleet management sensors - Allow businesses to track locations and progress in real time.

6.4.1 IoT Connectivity and Management

- An IoT connectivity management platform connects and streamlines the management of IoT devices, such as M2M SIM cards. Sometimes referred to as a "connected devices platform," the software promotes the scalability of IoT device controls and reduces time - consuming onboarding processes.
- The software allows businesses to reduce downtime from device repairs by proactively checking for errors and predicting failures before they happen. It also streamlines connectivity management activities, bolstering productivity and reducing the time spent monitoring.
- Timely and controlled access to the data insights provided by connected devices and sensors enables the creation of new processes, products and business models. This leads to informed business decisions.
- Connectivity Management Platform (CMP) is an application that allows user to monitor, analyze, provision and modify our cellular Internet of Things and Machine - to - Machine (M2M) deployments.

6.5 IoT Security

- In a network, endpoints are the devices that are connected to the internet, and this includes IoT devices. Every endpoint provides a potential point of entry for a bad actor to expose the network to outside risks. So, as with other endpoints, IoT devices are vulnerable to intense weaponization.
- IoT security covers both physical device security and network security, and impacts the processes, technologies, and measures necessary to protect IoT devices and networks.
- It spans industrial machines, smart energy grids, building automation systems, entertainment devices, and more, including devices that often aren't designed for network security.
- IoT device security must protect systems, networks, and data from a broad spectrum of IoT security attacks, which target various types of vulnerabilities.
- Communication attacks on the data transmitted between IoT devices and servers. Lifecycle attacks on the IoT device as it changes hands from user to maintenance. Attacks on the device software. Physical attacks, which directly target the chip in the device.

Privacy - preserving Sharing of IoT Data

- The increasing development of IoT is dramatically changing the way people share information and communicate with their surrounding environment, enabling a constant, invisible and sometimes unintended data exchange, between things and people.
- The main objective of privacy preservation is ensuring that private data remains protected, while processing or releasing sensitive information.
- Privacy in the Internet of Things is the threefold guarantee to the subject for :
  a. Awareness of privacy risks imposed by smart things and services surrounding the data subject
  b. Individual control over the collection and processing of personal information by the surrounding smart things
  c. Awareness and control of subsequent use and dissemination of personal information by those entities to any entity outside the subject's personal control sphere.
- Fig. 6.5.1 shows an IoT reference model with relevant entities and data flows in a typical IoT application.
- The fundamental privacy mechanisms lie in the intelligent data management so that only the required data is collected. Detecting the redundancy, data is

anonymized at the earliest possible stage and then deleted at the earliest convenience.

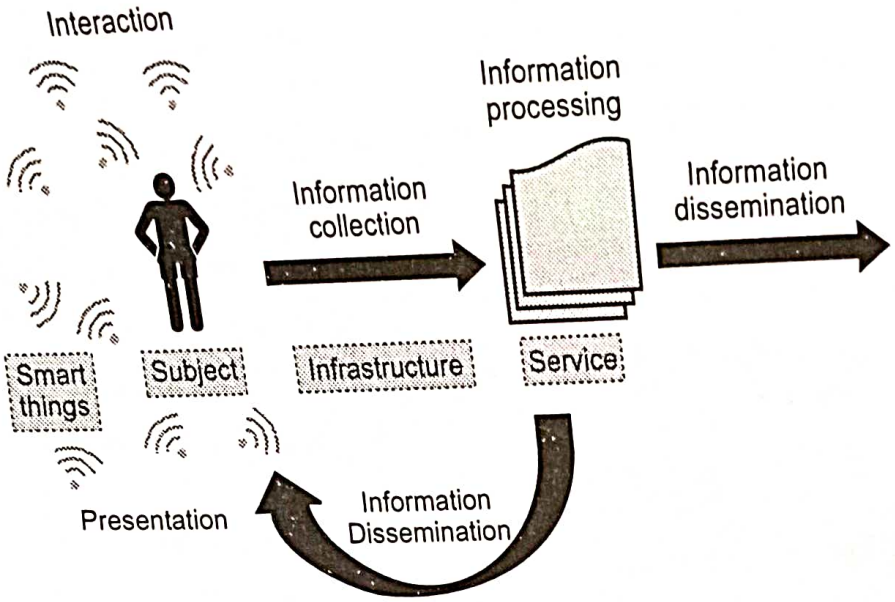

Fig. 6.5.1

6.5.1 Vulnerabilities of IoT

- IoT devices are vulnerable largely because these devices lack the necessary built-in security to counter threats. Aside from the technical aspects, users also contribute to the devices' vulnerability to threats. Here are some of the reasons these smart devices remain vulnerable :
  1. Limited computational abilities and hardware limitations.
  2. Heterogeneous transmission technology.
  3. Components of the device are vulnerable.
  4. Users lacking security awareness. Lack of user security awareness could expose smart devices to vulnerabilities and attack openings.
- An IoT device can have one or multiple vulnerabilities that make it an easy target for hackers to gain access to a network and move laterally to more critical devices or systems.
- IoT vulnerabilities due to participation of the number of layers, hardware sublayers and software in applications and services.
- The nature of IoT Vulnerabilities varies, for example, sensors, machines, automobiles, wearables, and so on. Each faces different kind of vulnerabilities and has complex security and privacy issues.

6.5.2 Misuse Cases in IoT Security

- Firstly, identify the actors and collaborating actors and develop misuse cases for each of them.
- Purpose : Creation of specifications for communicating the potential risks an rationale.
- Security-relevant decisions to the device platform, application or service.
- Security issues for which misuse cases required in IoT.
- Identity misuse cases : Non-repudiation.
- Crypto of floads : The transfer of cryptographic keys and messages need to be protected from eavesdropper or hackers.
- Eavesdropping
- Fake server
- Fake device platform
- Flooding of TCP/SYN
- Unauthorised access to a data store
- Audit and accountability : Need each transaction to be securely stored to later on auditing and fixing accountabilities.

6.5.3 Attacks

- Various possible attacks in different layers of IoT.
- There are various attack surfaces available for attackers, protection needs to be considered at three different layers :
  1) **Edge protection :** Ensures device, mobile app and web app integrity to prevent devices from becoming attack entry points.
  2) **Network protection :** Secures communication channels to prevent man-in-the-middle attacks.
  3) **Cloud protection :** Assures data privacy and prevents data leakage.

6.5.4 IoT Security Issue and Need

- The Internet of Things (IoT) refers to a concept of connected objects and devices of all types over the Internet wired or wireless. The popularity of IoT or the Internet of Things has increased rapidly, as these technologies are used for various purposes, including communication, transportation, education, and business development.

- The unconscious use, not changing passwords and the lack of device updates have increased cybersecurity risks and access to malicious applications to the IoT systems sensitive data.
- Most of the security professionals consider IoT as the vulnerable point for cyber-attacks due to weak security protocols and policies. Even though several security mechanisms were developed to protect IoT devices from cyber-attacks, security guidelines are not appropriately documented.
- IoT enabled devices have been used in industrial applications and for multiple business purposes. The apps help these businesses to attain a competitive edge over their competitors.
- However, due to the excessive adoption of various smart devices with data sharing and integration, the privacy and data breach becomes a significant concern to most businesses, as it interrupts the flow of work, activities, and network services.
- IoT system functionalities :
  1. Security patch must be upload time to time in microprocessor firmware.
  2. Monitor the access and usage of public network.
  3. User authentication is necessary.
  4. Only after authentication can the controller direct commands for things control that are present in the system.
- The Internet of Things (IoT) has become a ubiquitous term to describe the tens of billions of devices that have sensing or actuation capabilities and are connected to each other via the Internet.

Risks :

- The IoT includes everything from wearable fitness bands and smart home appliances to factory control devices, medical devices and even automobiles. Security has not been a high priority for these devices until now.
- The security of the Internet of Things, the following principles can be established.
  a) **Identity :** Trust is always tied to an identity. Therefore every device needs a unique identity that can't be changed. The device must also be able to prove its identity at all times.
  b) **Positive intention :** The device and linked service have positive intentions.

c) **Predictability and transparency :** The functional scope of the service provided by devices is known to its full extent. There are no undocumented (secret) functions. The behaviour of the system can be checked at any time by independent third parties.

d) **Reputation :** An increasing number of positive interactions between the things gradually form a reputation based intelligent network.

6.5.5 Security Architecture

- Fig. 6.5.2 shows IoT security architecture.

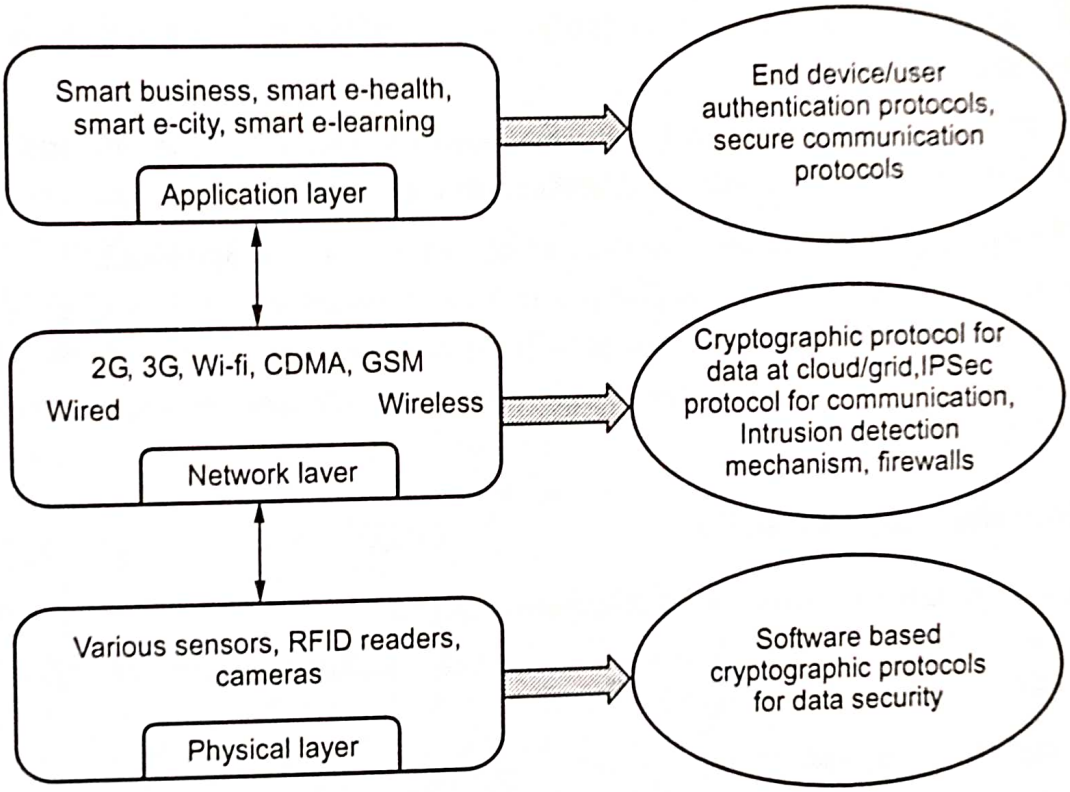

Fig. 6.5.2 IoT security architecture

- IoT systems are often highly complex, requiring end-to-end security solutions that span cloud and connectivity layers, and support resource-constrained IoT devices that often aren't powerful enough to support traditional security solutions.
- Application layer support user services. This layer helps users access IoT through the interface using PC, mobile equipment etc. This layer also support secure communication protocol and authentication protocols.
- Network layer support wired and wireless communication protocol and technology. This layer is responsible for dependable broadcast of data and information from the below layer.

- Sensors are the monitors that pick up data and relay it for further analysis. Actuators are devices that act as robotic controls. Many IoT attacks have used actuators, such as printers, as launch points into a business's network.
- An IoT security architecture is a blueprint that illustrates all components of the IoT infrastructure for all IoT projects and details how to secure each component.
- In both cases, it is imperative to ensure device access is controlled via settable passwords, encrypt any data stored locally and monitor and contain any executable code run by the device.
- Physical layer gathers all types of information with the help of physical equipment. IoT devices face many threats, including malicious data that can be sent over authenticated connections, exploiting vulnerabilities and/or misconfigurations.
- Such attacks frequently exploit many weaknesses, including but not limited to
  a) Failure to use code signature verification and secure boot,
  b) Poorly implemented verification models which can be bypassed.
- Attackers often use those weaknesses to install backdoors, sniffers, data collection software, file transfer capabilities to extract sensitive information from the system, and sometimes even Command & Control (C & C) infrastructure to manipulate system behaviour.

6.5.6 Security Requirement

- The key requirements for any IoT security solution are :
  1. Device and data security, including authentication of devices and confidentiality and integrity of data.
  2. Implementing and running security operations at IoT scale.
  3. Meeting compliance requirements and requests.
  4. Meeting performance requirements as per the use case.
- Application layer : Verification and user's confidentiality
- Support layer : Various encryption algorithms
- Network layer : Distributed denial of service attack
- Physical layer : Authentication.

6.5.7 Key Elements of IoT Security

- Fig. 6.5.3 shows key elements of IoT security.

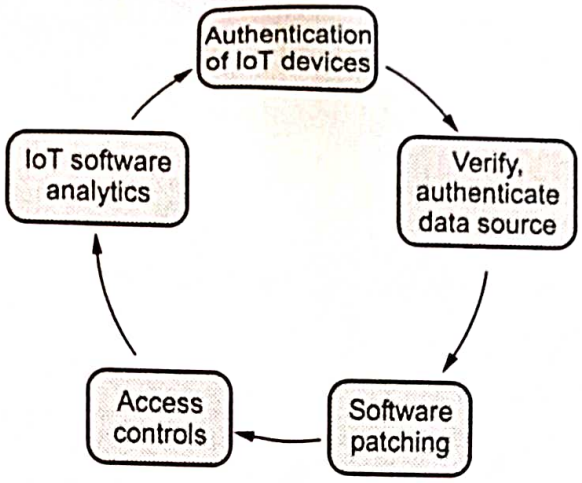

Fig. 6.5.3 Elements of IoT security

- **Authenticate IoT devices** before they integrate with the centralized network. This step avoids the risk of spoofing attacks on the IoT device which appears as a legitimate device on your network. Attacked devices collect data from other networked device or transmit malicious data to remaining devices on the network.
- It is essential to authenticate and verify software source running on your IoT device. Un Software not authenticated faces the risk of being compromised, and the device cannot detect such tampering issues unless software is supported with a digital signature of the vendor.
- **Software patching** avoids the IoT device from being compromised. However, software updates from authenticated sources must be accepted only. Software patches minimize risks of data loss or interference with business operations. For instance, a device is put for updating; all local data is written on a central network, other devices know when the updating device goes offline, and the update is performed, verified before returning back to the operational mode.
- **Access controls secure** IoT devices as well as the enterprise. Users are assigned roles to perform specific operations. Roles include querying current state of the IoT devices, software updates on these devices, and change device configurations. Other systems present on the network work on the principle of least privilege that allows users only a minimal set of privileges to perform business functions and technical processes. This method restricts damage caused due to a security breach of user's personal data.
- **IoT software** analytics must be designed on the basis of anomaly detection. Basic guidelines may be pre-defined, and variation from these guidelines should alert the user. For instance, higher-than-expected traffic from some devices would alert users that the devices have been compromised and are in use even after a

malicious attack. Considering the response to such anomalous behavior, you can perhaps switch off the affected devices or remove them from the connected network.

Identity establishment Issue :

- The identity management in IoT is performed by interchanging finding information between the things for first time connection.
- This process is affected to overhear which can lead to man-in-the-middle attack, and thus can threaten the whole IoT structure.
- Hence, there needs to be some pre-defined identity management entity or hub which can observes the relation process of devices by applying cryptography and other techniques to prevent identity theft.
- IoT requires security computes at all three layers; at physical layer for data collection, at network layer for overpower and dispatch, and at application layer to maintain confidentiality, authentication, and integrity.

6.5.8 Security Model for IoT

- Fig. 6.5.4 shows security model for IoT.

Application Layer	IoT Application
	Application support layer (Middleware technology security)
Transportation Layer	Local Area Network
	Core Network (Internet Security)
	Access Network (Ad hoc security, GPRS security, WiFi security)
Perception Layer	Perception Network (RFID security) (WSN Security)
	Perception Node (RSN Security)

Fig. 6.5.4 Security model for IoT

- Data and privacy protection is one of the application challenges of IoT. In IoT, RFID systems, WSNs sensors perceive for the end of the information technology, which protect the integrity and confidentiality of information by the password encryption technology.
- There are many ways to encrypt data and information, such as random hash lock protocol (hash function), hash chain protocol, extract key from an infinite channel, Encrypted identifier and so on.
- Identity authentication and access control can determine the communication between both sides and confirm each other's true identity, prevent disguised attacks to ensure the authenticity, validity of the information and so on.

- IoT must ensure the security of all layers. In addition, IoT security should also include the security of whole system crossing the perception layer, transportation layer and application layer.
- Perception layer includes RFID security, WSNs security, RSN security and any others.
- Transportation layer includes access network security, core network security and local network security. There are 3G access network security, Ad-Hoc network security, WiFi security and so on for these sub layers.
- Different network transmission has different technology. Application layer includes application support layer and specific IoT applications. The security in support layer includes middleware technology security, cloud computing platform security and so on.
- IoT applications in different industries have different application requirements. Thus, networking security is a large multi-layered security system, in addition, considering security of each layers also consider cross-layer integration of heterogeneous network security issues.
- Perception layer is mainly about information collection, object perception and object control. Perception layer can be divided into two parts : perception node (sensors or controllers, etc.), perception network that communicates with transportation network.
- Transportation layer mainly provides ubiquitous access environment for perception layer, perception of information transmission and storage, and application layer load other related businesses. Transportation layer can be divided into three layers by function : the access network, the core network and local area. It is a combination of a variety of heterogeneous networks.
- The application support layer, an advanced layer above the transportation layer, supports all sorts of business services and realizes intelligent computation and resources allocation in screening, selecting, producing and processing data. During the whole process, the application support layer can recognize valid data, spam data and even malicious data and filter them in time.
- Application support layer can be organized in different ways according to different services. Usually it includes middleware, M2M, cloud computing platform and service support platform.

6.5.9 Challenges in Designing IoT

- IoT enables a constant transfer and sharing of data among things and users. In such a sharing environment, authentication, authorization, access control and non-repudiation are important to ensure secure communication.
- The high level of heterogeneity, coupled to the wide scale of IoT systems, is expected to magnify security threats of the current Internet. The high number of inter-connected devices arises scalability issues.
- IoT systems integrate in a seamless way physical objects, data, and computing devices into a global network of information about "smart things".
- Fig. 6.5.5 shows high level security challenges of IoT.

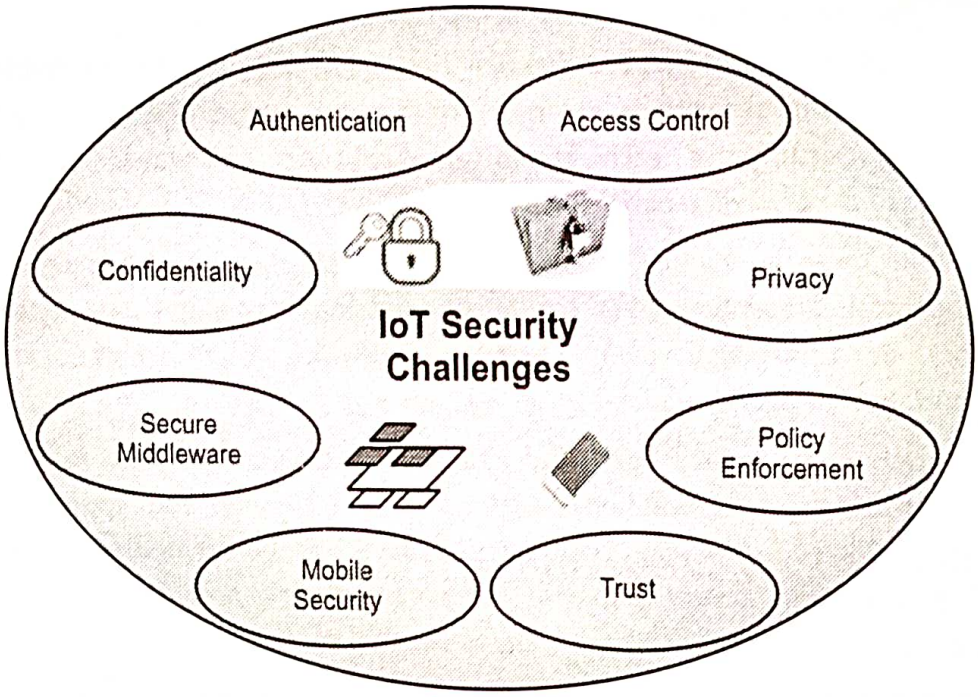

Fig. 6.5.5 IoT security challenges

- Access control refers to the permissions in the usage of resources, assigned to different actors of a wide IoT network.
- System safety is highly application - or application domain-specific. Trust Model that provides data integrity and confidentiality, and endpoint authentication and non-repudiation between any two system-entities that interact with each other.
- The trust requirements in IoT are related to identify management and access control issues. The IoT-A Privacy Model depends on the following functional components : Identity Management, Authentication, Authorization, and Trust & Reputation.

- Communication security : IoT systems are heterogeneous. Not only because of the variety of the entities involved, but also because they include Devices with various capabilities in terms of communication and processing.
- Communication security model must not only consider the hetereogenity of the system, but it also should guarantee a balance between security features, bandwidth, power supply and processing capabilities.
- IoT devices face many threats, including malicious data that can be sent over authenticated connections, exploiting vulnerabilities. Such attacks frequently exploit many weaknesses, including but not limited to :
  a) Failure to use code signature verification and secure boot, and
  b) Poorly implemented verification models which can be bypassed.
- Attackers often use those weaknesses to install backdoors, sniffers, data collection software, file transfer capabilities to extract sensitive information from the system, and sometimes even Command & Control (C&C) infrastructure to manipulate system behavior.
- The security challenges are as follows :
  a. Devices are not reachable : Most of the time a device is not connected.
  b. Devices can be lost and stolen : Makes security difficult when the device is not connected.
  c. Devices are not crypto-engines : Strong security difficult without processing power.
  d. Devices have finite life : Credentials need to be tied to lifetime.
  e. Devices are transportable : Will cross borders.
- IoT system has a cloud database that is connected to all your devices. These devices are connected to the internet and it could be accessed by the cybercriminals and hackers. As the number of connected devices increases, chances for hackers to breach the security system gets increased.

6.6 IoT Communication APIs

1. REST Based Communication API :

- A large part of the interoperability, scale, and control for IoT can be achieved through API management. Standards-based design patterns for Web APIs, API management, and a RESTful architecture provide tremendous value in simplifying the task of interoperability across heterogeneous systems handling vast amounts of data.
- Representational State Transfer (REST) APIs follow the request-response communication model.

- Applications conforming to the REST constraints can be called RESTful. RESTful systems typically communicate over HTTP with the same Methods (GET, POST, PUT, DELETE etc) that browsers use to retrieve web pages and to send data to remote servers.
  1. **Client - Server :** Requires that a service offer one or more operations and that services wait for clients to request these operations.
  2. **Stateless :** Requires communication between service consumer (client) and service provider (server) to be stateless.
  3. **Cache :** Requires responses to be clearly labeled as cacheable or non - cacheable.
  4. **Uniform interface :** Requires all service providers and consumers within a REST - compliant architecture to share a single common interface for all operations.
  5. **Layered system :** Requires the ability to add or remove intermediaries at runtime without disrupting the system.
  6. **Code - on - Demand :** Allows logic within clients (such as Web browsers) to be updated independently from server-side logic using executable code shipped from service providers to consumers.
- Each client request and server response is a message and REST-compliant applications expect each message to be self-descriptive. That means each message must contain all the information necessary to complete the task. Other ways to describe this kind of message are "stateless" or "context-free." Each message passed between client and server can have a body and metadata.
- **HTTP request methods and actions :**
  1. **GET :** Return whatever information is identified by the Request - URI.
  2. **HEAD :** Identical to GET except that the server must not return a message-body in the response, only the metadata.
  3. **OPTIONS :** Return information about the communication options available on the request/response chain identified by the Request - URI.
  4. **PUT :** Requests that the enclosed entity be stored under the supplied Request - URI.
  5. **POST :** Requests that the origin server accept the entity enclosed in the request as a new subordinate of the resource identified by the Request - URI.

6. **DELETE :** Requests that the origin server delete the resource identified by the Request - URI.

- The first three are read-only operations, while the last three are write operations.

2. WebSocket based communication API

- WebSocket support full-duplex, two-way communication between client and server.
- WebSocket APIs reduce the network traffic and latency as there is no overhead for connection setup and termination requests for each message.
- WebSocket uses a standard HTTP request-response sequence to establish a connection. When the connection is established, the WebSocket API provides a read and write interface for reading and writing data over the established connection in an asynchronous full duplex manner.
- WebSocket also provides an interface for asynchronously closing the connection from either side.