1 unit IOT and applications

Embedded Systems (Jawaharlal Nehru Technological University, Kakinada)

ADITYA COLLEGE OF ENGINEERING & TECHNOLOGY(A)
(An AUTONOMOUS Institution)
Approved by AICTE, New Delhi * Permanently Affiliated to JNTUK, Kakinada
Accredited by NBA* Accredited by NAAC A+ Grade with CGPA of 3.40
Recognized by UGC Under Sections 2(f) and 12(B) of the UGC Act, 1956
Aditya Nagar, ADB Road, Surampalem, Gandepalli Mandal, Kakinada District - 533437, A.P
Ph. 99591 76665, Email: office@acet.ac.in, www.acet.ac.in

IOT and Applications

Dr.R.V.S.Lalitha, Professor, Department of CSE, 8008379819

UNIT I:

Introduction to IoT: Introduction to IoT, Architectural Overview, Design principles and needed capabilities, Basics of Networking, M2M and IoT Technology Fundamentals- Devices and gateways, Data management, Business processes in IoT, Everything as a Service (XaaS), Role of Cloud in IoT, Security aspects in IoT.

UNIT II:

Elements of IoT: Hardware Components- Computing- Arduino, Raspberry Pi, ARM Cortex-A class processor, Embedded Devices – ARM Cortex-M class processor, Arm Cortex-M0 Processor Architecture, Block Diagram, Cortex-M0 Processor Instruction Set, ARM and Thumb Instruction Set.

UNIT III:

IoT Application Development: Communication, IoT Applications, Sensing, Actuation, I/O interfaces.

Software Components- Programming API's (using Python/Node.js/Arduino) for Communication Protocols-MQTT, ZigBee, CoAP, UDP, TCP, Bluetooth.

Bluetooth Smart Connectivity Bluetooth overview, Bluetooth Key Versions, Bluetooth Low Energy (BLE) Protocol, Bluetooth, Low Energy Architecture, PSoC4 BLE architecture and Component Overview.

UNIT IV:

Solution framework for IoT applications: Implementation of Device integration, Data acquisition and integration, Device data storage- Unstructured data storage on cloud/local server, Authentication, authorization of devices.

UNIT V:

IoT Case Studies: IoT case studies and mini projects based on Industrial automation, Transportation, Agriculture, Healthcare, Home Automation. Cloud Analytics for IoT Application :Introduction to cloud computing, Difference between Cloud Computing and Fog Computing: The Next Evolution of Cloud Computing, Role of Cloud Computing in IoT, Connecting IoT to cloud, Cloud Storage for IoT Challenge in integration of IoT with Cloud.

Text Books:

1. Raj Kamal, "Internet of Things: Architecture and Design Principles", 1st Edition, McGraw Hill Education,2017.
2. The Definitive Guide to the ARM Cortex-M0 by JosephYiu,2011
3. Vijay Madisetti, Arshdeep Bahga, Internet of Things, "A Hands on Approach", UniversityPress,2015

References:

1. CypressSemiconductor/PSoC4BLE(BluetoothLowEnergy)ProductTrainingModules.
2. PethuruRajandAnupamaC.Raman,"TheInternetofThings:EnablingTechnologies,Platforms,andUse Cases",CRCPress,2017.

UNIT I:

Introduction to IoT: Introduction to IoT, Architectural Overview, Design principles and needed capabilities, Basics of Networking, M2M and IoT Technology Fundamentals- Devices and gateways, Data management, Business processes in IoT, Everything as a Service (XaaS), Role of Cloud in IoT, Security aspects in IoT.

1. Introduction to IoT

Internet of Things

Internet of Things (IoT) is a concept which enables communication between internetworking devices and applications, whereby physical objects or 'things' communicate through the Internet.

The concept of IoT began with things classified as identity communication devices. Radio Frequency Identification Device (RFID) is an example of an identity communication device. Things are tagged to these devices for their identification in future and can be tracked, controlled and monitored using remote computers connected through the Internet.

The concept of IoT enables, for example, GPS-based tracking, controlling and monitoring of devices; machine-to-machine (M2M) communication; connected cars; communication between wearable and personal devices and Industry 4.0.

The IoT concept has made smart cities a reality and is also expected to make self-driving cars functional very soon.

IoT Definition

The Internet is a vast global network of connected servers, computers, tablets and mobiles that is governed by standard protocols for connected systems. It enables sending, receiving, or communication of information, connectivity with remote servers, cloud and analytics platforms.

Thing is a word used to refer to a physical object, an action or idea, a situation or activity, in case when one does not wish to be precise. Example of **reference to an object** is—an umbrella is a useful thing in rainy days.

Internet of Things means a network of physical things (objects) sending, receiving, or communicating information using the Internet or other communication technologies and network just as the computers, tablets and mobiles do, and thus enabling the monitoring, coordinating or controlling process across the Internet or another data network.

Internet of Things is the network of physical objects or 'things' embedded with electronics, software, sensors and connectivity to enable it to achieve greater value and service by exchanging data with the manufacturer, operator and/or other connected devices. Each thing is uniquely identifiable through its embedded computing system but is able to interoperate within the existing Internet **infrastructure.**

1.1.2 IoT Vision

Internet of Things is a vision where things (wearable watches, alarm clocks, home devices, surrounding objects) become 'smart' and function like living entities by sensing, computing

and communicating through embedded devices which interact with remote objects (servers, clouds, applications, services and processes) or persons through the Internet or Near-Field Communication (NFC) etc. The vision of IoT can be understood through Examples 1.1 and 1.2.

Example 1.1.

Through computing, an umbrella can be made to function like a living entity. By installing a tiny embedded device, which interacts with a web based weather service and the devices owner through the Internet the following communication can take place. The umbrella, embedded with a circuit for the purpose of computing and communication connects to the Internet. A website regularly publishes the weather report. The umbrella receives these reports each morning, analyses the data and issues reminders to the owner at intermittent intervals around his/her office-going time. The reminders can be distinguished using differently coloured LED flashes such as red LED flashes for hot and sunny days, yellow flashes for rainy days. A reminder can be sent to the owner's mobile at a pre-set time before leaving for office using NFC, Bluetooth or SMS technologies. The message can be—(i) Protect yourself from rain. It is going to rain. Don't forget to carry the umbrella; (ii) Protect yourself from the sun. It is going to be hot and sunny. Don't forget to carry the umbrella. The owner can decide to carry or not to carry the umbrella using the Internet connected umbrella.

The HAZ Umbrella frees users from cumbersome traditional umbrellas with a built-in high precision motor, microcontroller and a high capacity Li-ion battery. This setup allows it to open, extend and close with the push a single button. Ready for use or storage in under two seconds, the umbrella is fully automated and convenient for users who are carrying multiple objects.

2. Architectural view

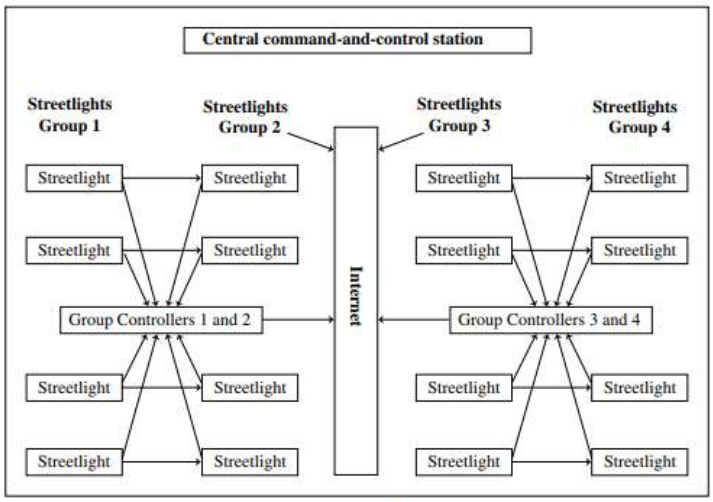


Figure 1.1 Use of Internet of Things concept for streetlights in a city

- As per Collins Dictionary, hyperconnectivity means use of multiple systems and devices to remain constantly connected to social networks and streams of information.
- Smart devices are devices with computing and communication capabilities that can constantly connect to networks.
- For example, a city network of streetlights which constantly connects to the controlling station as shown in Figure 1.1 for its services.
- Another example is hyperconnected RFIDs. An RFID or a smart label is tagged to all consignments. This way many consignments sent from a place can be constantly tracked. Their movement through remote places, inventories at remote locations, sales and supply chain are controlled using a hyper-connected framework for Internet of RFIDs.

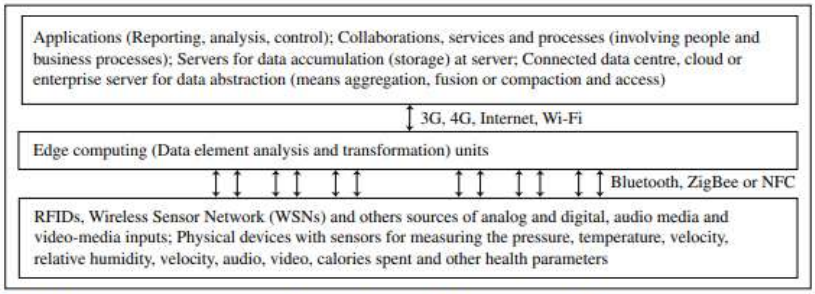


Figure 1.2 A general framework for IoT using smart and hyperconnected devices, edge computing and applications

- Figure 1.2 shows a general framework for IoT using smart and hyperconnected devices, edge computing and applications.
- A device is considered at the edge of Internet infrastructure. Edge computing implies computations at the device level before the computed data communicates over the internet.

1.2 IoT CONCEPTUAL FRAMEWORK

- Example 1.1 showed a single object (umbrella) communicating with a central server for acquiring data.
  The following equation describes a simple conceptual framework of IoT2 :
  Physical Object + Controller, Sensor and Actuators + Internet = Internet of Things
- Equation 1.1 conceptually describes the Internet of umbrellas as consisting of an umbrella, a controller, sensor and actuators, and the Internet for connectivity to a web service and a mobile service provider.
- An **actuator** is a part of a device or machine that helps it to achieve physical movements by converting energy, often electrical, air, or hydraulic, into mechanical force.
- Generally, IoT consists of an internetwork of devices and physical objects wherein a number of objects can gather the data at remote locations and communicate to units managing, acquiring, organising and analysing the data in the processes and services.
- Example 1.2 showed the number of streetlights communicating data to the group controller which connects to the central server using the Internet.
- A general framework consists of the number of devices communicating data to a data centre or an enterprise or a cloud server.
- The IoT framework of IoT used in number of applications as well as in enterprise and business processes is therefore, in general, more complex than the one represented by Equation 1.1. The equation below conceptually represents the actions and communication of data at successive levels in IoT consisting of internetworked devices and objects.
  Gather + Enrich + Stream + Manage + Acquire + Organise and Analyse

Equation 1.2 is an IoT conceptual framework for the enterprise processes and services, based on a suggested IoT architecture given by Oracle.

The steps are as follows:

1. At level 1 data of the devices (things) using sensors or the things gather the pre data from the internet.
2. A sensor connected to a gateway, functions as a smart sensor (smart sensor refers to a sensor with computing and communication capacity). The data then enriches at level 2, for example, by transcoding at the gateway. Transcoding means coding or decoding before data transfer between two entities.
3. A communication management subsystem sends or receives data streams at level 3.
4. Device management, identity management and access management subsystems receive the device's data at level 4.
5. A data store or database acquires the data at level 5.
6. Data routed from the devices and things organises and analyses at level 6. For example, data is analysed for collecting business intelligence in business processes.

The equation below is an alternative conceptual representation for a complex system.

- It is based on IBM IoT conceptual framework. The equation shows the actions and communication of data at successive levels in IoT. The framework manages the IoT services using data from internetwork of the devices and objects, internet and cloud services, and represents the flow of data from the IoT devices for managing the IoT services using the cloud server.

Gather + Consolidate + Connect + Collect + Assemble + Manage and Analyse

Equation 1.3 represents a complex conceptual framework for IoT using cloud-platform based processes and services.

The steps are as follows:

1. Levels 1 and 2 consist of a sensor network to gather and consolidate the data. First level gathers the data of the things (devices) using sensors circuits. The sensor connects to a gateway. Data then consolidates at the second level, for example, transformation at the gateway at level 2.
2. The gateway at level 2 communicates the data streams between levels 2 and 3. The system uses a communication-management subsystem at level 3.
3. An information service consists of connect, collect, assemble and manage subsystems at levels 3 and 4. The services render from level 4.
4. Real time series analysis, data analytics and intelligence subsystems are also at levels 4 and 5. A cloud infrastructure, a data store or database acquires the data at level 5.

- Figure 1.3 shows blocks and subsystems for IoT in the IBM conceptual framework.

- *Adrian McEwen* and *Hakim Cassimally* equation is a simple conceptualisation of a framework for IoT with connectivity to a web service:

  Physical Object + Controller, Sensor and Actuators + Internet = Internet of Things

- An equation to conceptualise a general framework for IoT with connectivity to a data centre, application or enterprise server for data storage, services and business processes is:

  Gather + Enrich + Stream + Manage + Acquire + Organise and Analyse = Internet of Things

  Orcale suggested IoT architecture is the basis for this equation.

- Another equation which conceptualises the general framework for IoT using the cloud based services is:

  Gather + Consolidate + Connect + Collect + Assemble + Manage and Analyse = Internet of Things

-

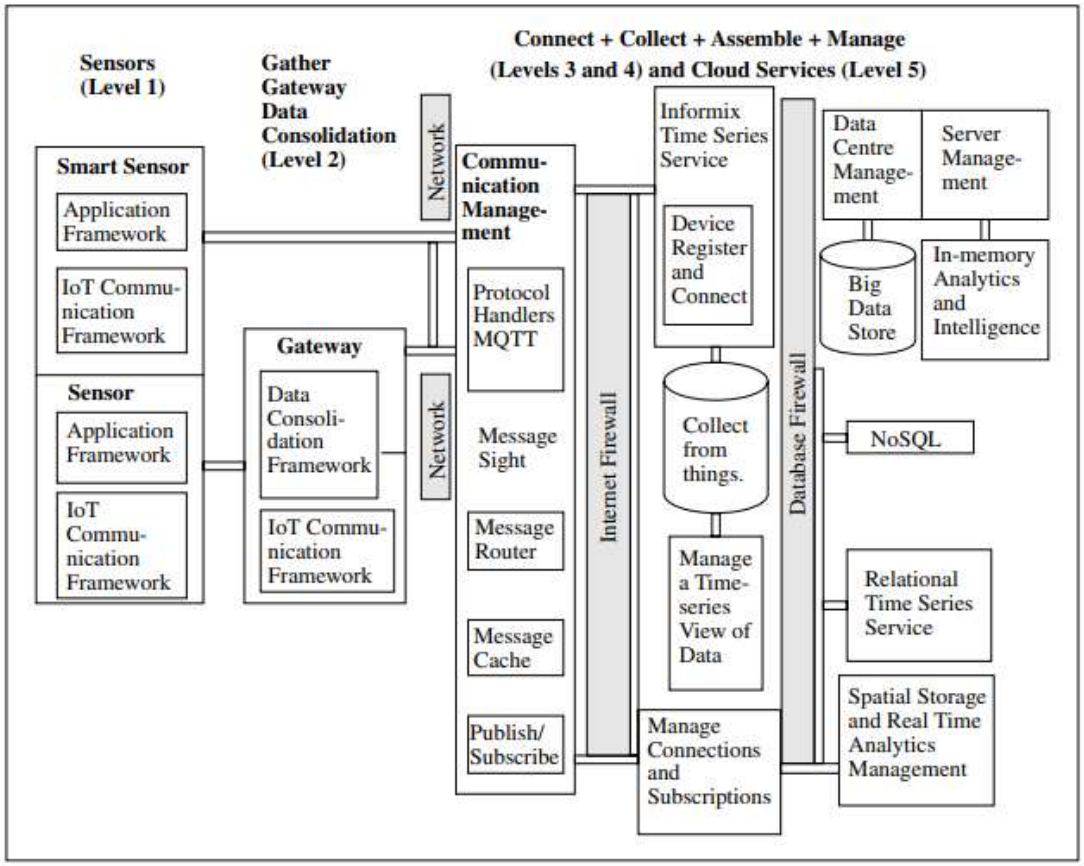


Figure 1.3 IBM IoT conceptual framework

IBM IoT conceptual framework blocks and components are the basis of this equation.

- In general, things refer to an internetwork of devices and physical objects. This framework consists of a number of subsystems.
- The data is acquired at remote locations in a database or data store. The services and processes need data managing, acquiring, organising and analysing.

MQTT (Message Queuing Telemetry Transport) is a lightweight, publish-subscribe based messaging protocol designed for resource-constrained devices and low-bandwidth, high-latency, or unreliable networks. It is widely used in Internet of Things (IoT) applications, providing efficient communication between sensors, actuators, and other devices.

1.3 IoT ARCHITECTURAL VIEW

An IoT system has multiple levels (Equations 1.1 to 1.3). These levels are also known as tiers. A model enables conceptualisation of a framework. A reference model can be used to depict building blocks, successive interactions and integration. An example is CISCO's presentation of a reference model comprising seven levels (Figure 1.4).

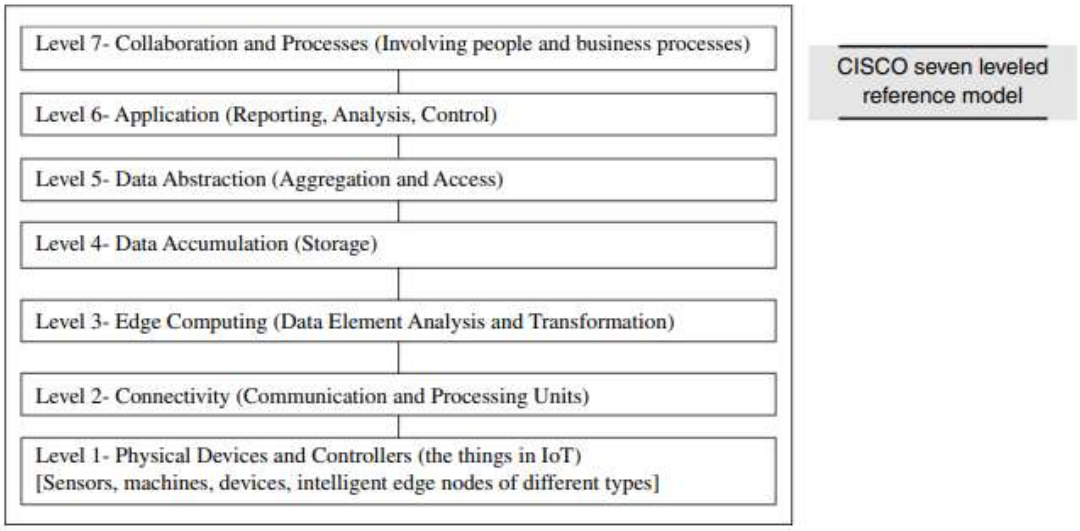


Figure 1.4 An IoT reference model suggested by CISCO that gives a conceptual framework for a general IoT system

Figure 1.5 shows an Oracle suggested IoT architecture.

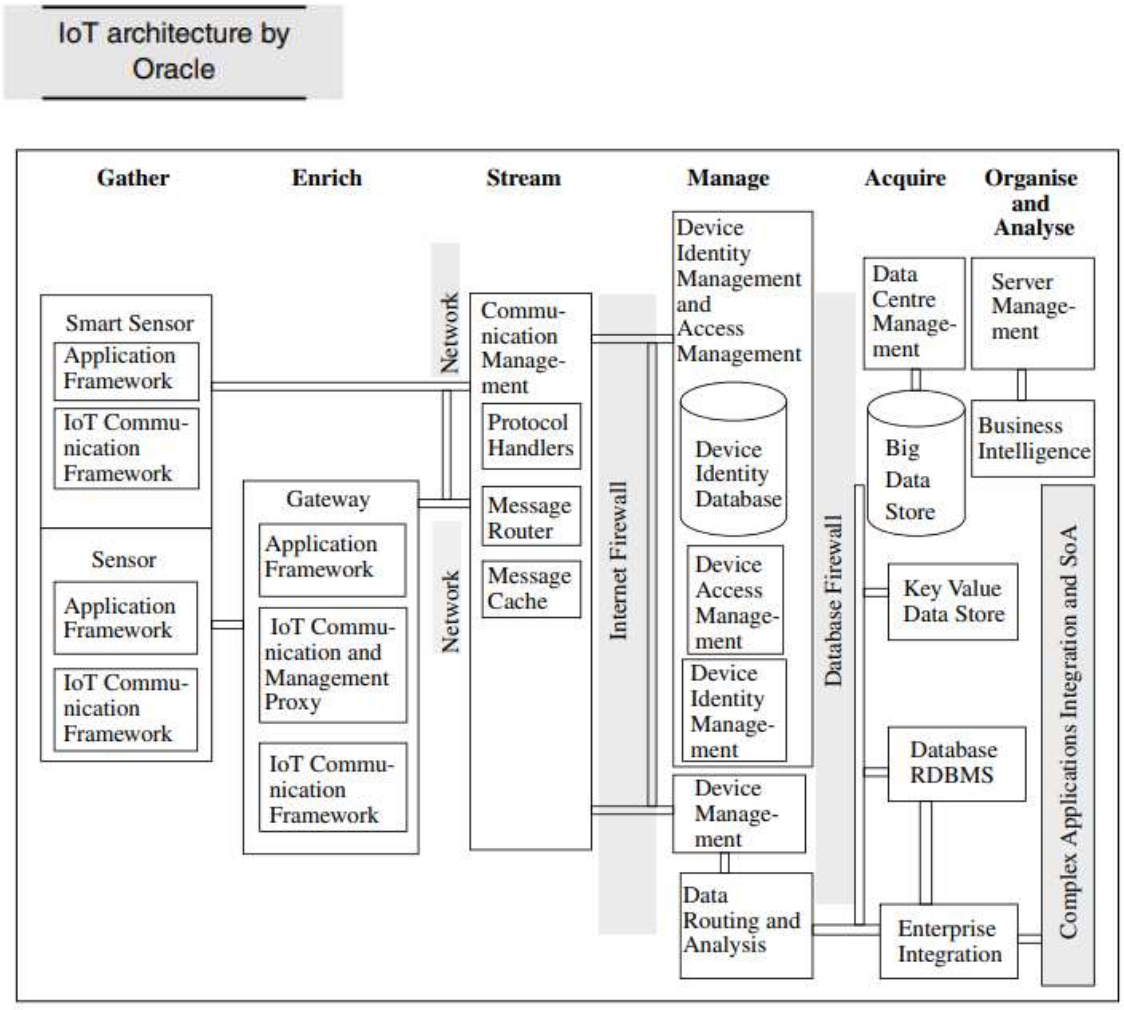


Figure 1.5 Oracle's IoT architecture (Device identity management means identifying a device, registering a device for actions after identifying, de-registering the device, assigning unique identity to the device. Device access management means enabling, disabling the device access, authenticating a device for access, authorizing a device for access to a subsystem.
An architecture has the following features:

● The architecture serves as a reference in applications of IoT in services and business processes.

● A set of sensors which are smart, capture the data, perform necessary data element analysis and transformation as per device application framework and connect directly to a communication manager.

- A set of sensor circuits is connected to a gateway possessing separate data capturing, gathering, computing and communication capabilities. The gateway receives the data in one form at one end and sends it in another form to the other end.
- The communication-management subsystem consists of protocol handlers, message routers and message cache.
- This management subsystem has functionalities for device identity database, device identity management and access management.
- Data routes from the gateway through the Internet and data centre to the application server or enterprise server which acquires that data.
- Organisation and analysis subsystems enable the services, business processes, enterprise integration and complex processes

1.4 TECHNOLOGY BEHIND IoT

Hardware (Arduino Raspberry Pi, Intel Galileo, Intel Edison, ARM mBed, Bosch XDK110, Beagle Bone Black and Wireless SoC)

● Integrated Development Environment (IDE) for developing device software, firmware and APIs
● Protocols [RPL, CoAP, RESTful HTTP, MQTT, XMPP (Extensible Messaging and Presence Protocol)]
● Communication (Powerline Ethernet, RFID, NFC, 6LowPAN, UWB, ZigBee, Bluetooth, WiFi, WiMax, 2G/3G/4G)
● Network backbone (IPv4, IPv6, UDP and 6LowPAN)
● Software (RIOT OS, Contiki OS, Thingsquare Mist firmware, Eclipse IoT)
● Internetwork Cloud Platforms/Data Centre (Sense, ThingWorx, Nimbits, Xively, openHAB, AWS IoT, IBM BlueMix, CISCO IoT, IOx and Fog, EvryThng, Azure, TCS CUP)
● Machine learning algorithms and software

The following five entities can be considered for the five levels behind an IoT system (Figure 1.3):
1. Device platform consisting of device hardware and software using a microcontroller (or SoC or custom chip), and software for the device APIs and web applications

2. Connecting and networking (connectivity protocols and circuits) enabling internetworking of devices and physical objects called things and enabling the internet connectivity to remote servers
3. Server and web programming enabling web applications and web services
4. Cloud platform enabling storage, computing prototype and product development platforms
5. Online transactions processing, online analytics processing, data analytics, predictive analytics and knowledge discovery enabling wider applications of an IoT system

1.4.1 Server-end Technology

IoT servers are application servers, enterprise servers, cloud servers, data centres and databases. Servers offer the following software components:

- Online platforms
- Devices identification, identity management and their access management
- Data accruing, aggregation, integration, organising and analysing
- Use of web applications, services and business processes

1.4.1 Server-end Technology

IoT servers are application servers, enterprise servers, cloud servers, data centres and databases. Servers offer the following software components:

- Online platforms
- Devices identification, identity management and their access management
- Data accruing, aggregation, integration, organising and analysing
- Use of web applications, services and business processes

1.4.1 Server-end Technology

IoT servers are application servers, enterprise servers, cloud servers, data centres and databases. Servers offer the following software components:

- Online platforms
- Devices identification, identity management and their access management
- Data accruing, aggregation, integration, organising and analysing
- Use of web applications, services and business processes

Control Units

Most commonly used control unit in IoT consists of a Microcontroller Unit (MCU) or a custom chip. A microcontroller is an integrated chip or core in a VLSI or SoC. Popular microcontrollers are ATmega 328, ATMega 32u4, ARM Cortex and ARM LPC.

An MCU comprises a processor, memory and several other hardware units which are interfaced together. It also has firmware, timers, interrupt controllers and functional IO units. Additionally, an MCU has application-specific functional circuits designed as per the specific version of a given microcontroller family. For example, it may possess Analog to Digital Converters (ADC) and Pulse Width Modulators (PWM).

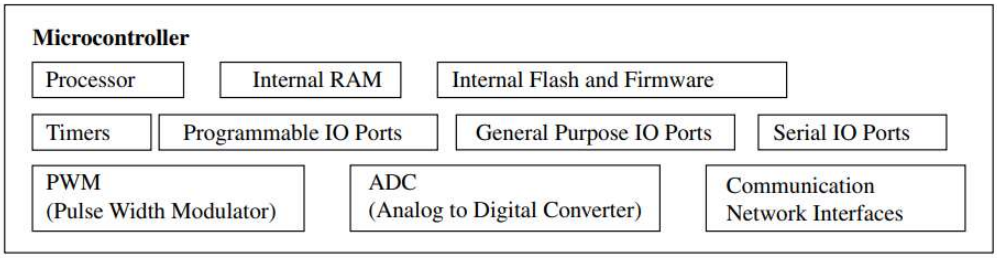


Figure 1.6 Various functional units in an MCU that are embedded in an IoT device or a physical object

Communication Module

A communication module consists of protocol handlers, message queue and message cache. A device message-queue inserts the messages in the queue and deletes the messages from the queue in a first-in first-out manner. A device message-cache stores the received messages.

Representational State Transfer (REST) architectural style can be used for HTTP access by GET, POST, PUT and DELETE methods for resources and building web services. Software IoT software consists of two components—software at the IoT device and software at the IoT server.

Middleware

OpenIoT is an open source middleware. It enables communication with sensor clouds as well as cloud-based 'sensing as a service'. IoTSyS is a middleware which enables provisioning of communication stack for smart devices using IPv6, oBIX, 6LoWPAN, CoAP and multiple standards and protocols. The oBIX is standard XML and web services protocol oBIX (Open Building Information Xchange)

Operating Systems (OS)

Examples of OSs are RIOT, Raspbian, AllJoyn, Spark and Contiki. RIOT is an operating system for IoT devices. RIOT supports both developer and multiple architectures, including ARM7, Cortex-M0, Cortex-M3, Cortex-M4, standard x86 PCs and TI MSP430.

Raspbian is a popular Raspberry Pi operating system that is based on the Debian distribution of Linux.

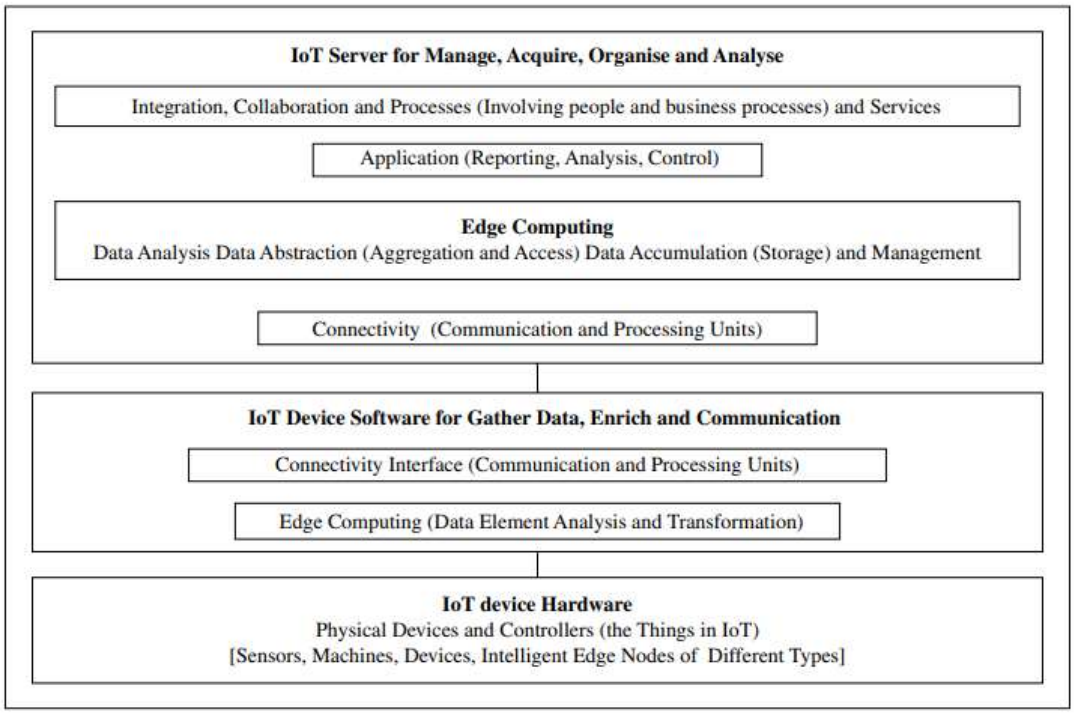


Figure 1.7 IoT software components for device hardware

Firmware

Thingsquare Mist is an open-source firmware (software embedded in hardware) for true Internet-connectivity to the IoT. It enables resilient wireless mesh networking. Several microcontrollers with a range of wireless radios support Things MIST.

1.4.3 Development Tools and Open-source Framework for IoT Implementation

Eclipse IoT (www.iot.eclipse.org) provides open-source implementation of standards such as MQTT CoAP, OMA-DM and OMA LWM2M, and tools for working with Lua, services and frameworks that enable an Open Internet of Things.

Arduino development tools provide a set of software that includes an IDE and the Arduino programming language for a hardware specification for interactive electronics that can sense and control more of the physical world.

Kinoma Software platform: Arduino development tools provide a set of software that includes an IDE and the Arduino programming language for a hardware specification for interactive electronics that can sense and control more of the physical world.

1.4.4 APIs and Device Interfacing Components

Connectivity interface consists of communication APIs, device interfaces and processing units. Figure 1.8 shows the mbed™ API and device interfacing components.

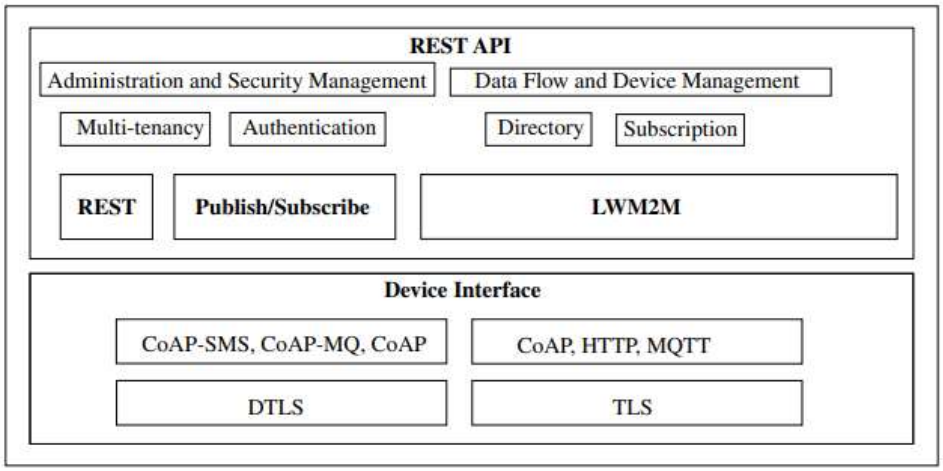


Figure 1.8 mBed™ API and device interfacing components

3. Design principles and needed capabilities

IEEE P80	Guide for Safety in AC Substation Grounding
IEEE 255	Standard Letter Symbols for Semiconductor Devices, IEEE-255-1963
IEEE 260	Standard Letter Symbols for Units of Measurement, IEEE-260-1978 (now 260.1-2004)
IEEE 488	Standard Digital Interface for Programmable Instrumentation, IEEE-488-1978 (now 488.1)
IEEE 519	Recommended Practice and Requirements for Harmonic Control in Electric Power Systems
IEEE 603	Standard Criteria for Safety Systems for Nuclear Power Generating Stations

IEEE 610	Standard Glossary of Software Engineering Terminology
IEEE 754	Floating point arithmetic specifications
IEEE 802	LAN/MAN
IEEE 802.1	Standards for LAN/MAN bridging and management and remote media access control (MAC) bridging
IEEE 802.2	Standards for Logical Link Control (LLC) standards for connectivity
IEEE 802.3	Ethernet Standards for Carrier Sense Multiple Access with Collision Detection (CSMA/CD)
IEEE 802.4	Standards for token passing bus access
IEEE 802.5	Standards for token ring access and for communications between LANs and MANs
IEEE 802.6	Standards for information exchange between systems
IEEE 802.7	Standards for broadband LAN cable
IEEE 802.8	Fiber-optic connection
IEEE 802.9	Standards for integrated services, like voice.
IEEE 802.10	Standards for LAN/MAN security implementations
IEEE 802.11	Wireless Networking – "WiFi"
IEEE 802.12	Standards for demand priority access method

IEEE 802.14	Standards for cable television broadband communications
IEEE 802.15.2	Bluetooth and Wi-Fi coexistence mechanism
IEEE 802.15.4	Wireless Sensor/Control Networks "Zigbee"
IEEE 802.15.6	Wireless Body Area Network[17] (BAN)
IEEE 802.16	Wireless Networking – "WiMAX"
IEEE 802.24	Standards for Logical Link Control (LLC) standards for connectivity
IEEE 828	Configuration Management in Systems and Software Engineering
IEEE 829	Software Test Documentation
IEEE 830	Software Requirements Specifications
IEEE 854	Standard for Radix-Independent Floating-Point Arithmetic, IEEE-854-1987 (replaced by IEEE-754-2008 and newer)
IEEE 896	Futurebus
IEEE P1003.1	Portable Operating System Interface – – POSIX
IEEE 1016	Software Design Description
IEEE 1028	Standard for Software Reviews and Audits
IEEE 1044.1	Standard Classification for Software Anomalies
IEEE 1059	Software Verification And Validation Plan

IEEE 1073	Point of Care Medical Device Communication Standards
IEEE 1074	Software Development Life Cycle
IEEE 1076	VHDL – VHSIC Hardware Description Language
IEEE 1149.1	JTAG
IEEE 1149.6	AC-JTAG
IEEE 1180	Discrete cosine transform accuracy
IEEE 1196	NuBus
IEEE 1233	System Requirements Specification
IEEE 1275	Open Firmware
IEEE 1284	Parallel port
IEEE P1363	Public key cryptography
IEEE 1364	Verilog
IEEE 1394	Serial bus – "FireWire", "i.Link"
IEEE 1471	software architecture / system architecture
IEEE 1541	Prefixes for Binary Multiples
IEEE 1547	Standard for Interconnection and Interoperability of Distributed Energy Resources with Associated Electric Power Systems Interfaces

IEEE 1584	Guide for Performing Arc Flash Hazard Calculations
IEEE 1588	Precision Time Protocol
IEEE 1609	Wireless Access in Vehicular Environments (WAVE)
IEEE P1619	Security in Storage Working Group (SISWG)
IEEE 1625	Standard for Rechargeable Batteries for Multi-Cell Mobile Computing Devices
IEEE 1666	IEEE Standard for Standard SystemC Language Reference Manual
IEEE 1667	Standard Protocol for Authentication in Host Attachments of Transient Storage Devices
IEEE 1701	Optical Port Communication Protocol to Complement the Utility Industry End Device Data Tables
IEEE 1800	SystemVerilog
IEEE 1801	Unified Power Format
IEEE 1849	IEEE Standard for eXtensible Event Stream (XES) for Achieving Interoperability in Event Logs and Event Streams
IEEE 1855	IEEE Standard for Fuzzy Markup Language
IEEE 1901	Broadband over Power Line Networks
IEEE 1906.1	Recommended Practice for Nanoscale and Molecular Communication Framework
IEEE 1914	Next Generation Fronthaul Interface Working Group

IEEE 1914.1	Standard for Packet-based Fronthaul Transport Networks
IEEE 1914.3	Standard for Radio Over Ethernet Encapsulations and Mappings
IEEE 2030	Guide for Smart Grid Interoperability of Energy Technology and Information Technology Operation with the Electric Power System (EPS), End-Use Applications, and Loads
IEEE 2030.5	Standard for Smart Energy Profile Application Protocol
IEEE 2050	RTOS for embedded systems standard
IEEE 2143.1	Standard for General Process of Cryptocurrency Payment
IEEE 2413	Standard for an Architectural Framework for the Internet of Things (IoT)
IEEE 2418.2	Approved Draft Standard Data Format for Blockchain Systems
IEEE 2600	Hardcopy Device and System Security (and related ISO/IEC 15408 Protection Profiles)
IEEE 3001.4	Recommended Practice for Estimating the Costs of Industrial and Commercial Power Systems
IEEE 7010	Recommended Practice for Assessing the Impact of Autonomous and Intelligent Systems on Human Well-Being
IEEE 12207	Information Technology – Software life-cycle processes
IEEE C37.2040	Standard Cybersecurity Requirements for Substation Automation, Protection, and Control Systems
IEEE Switchgear Committee	C37 series of standards for Low and High voltage equipment

IEEE Transformers Committee	C57 series of standards for the design, testing, repair, installation and operation and maintenance of transformers

-

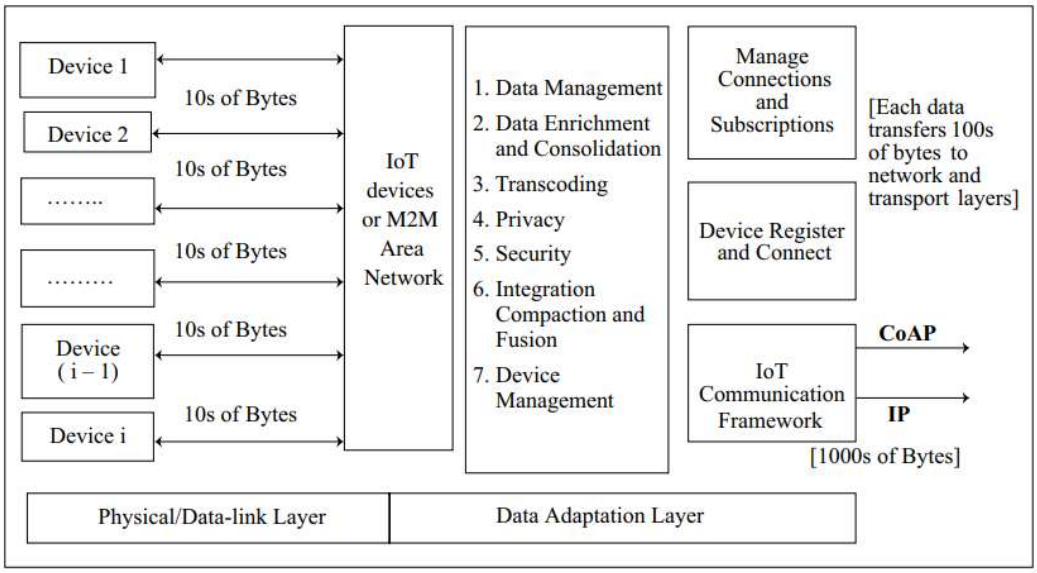


Figure 2.7 IoT or M2M gateway consisting of data enrichment and consolidation, device management and communication frameworks at the adaptation layer

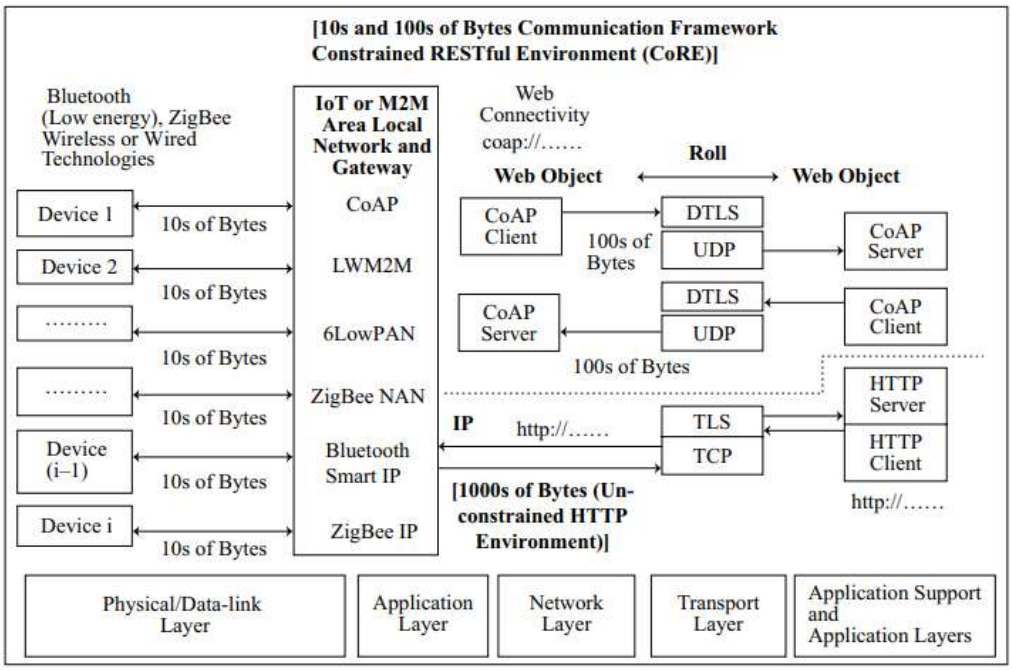


Figure 3.1 IoT or M2M devices local network connectivity and web connectivity in constrained (above thick dotted line) and unconstrained RESTful HTTP (below thick dotted line) environments using communication protocols

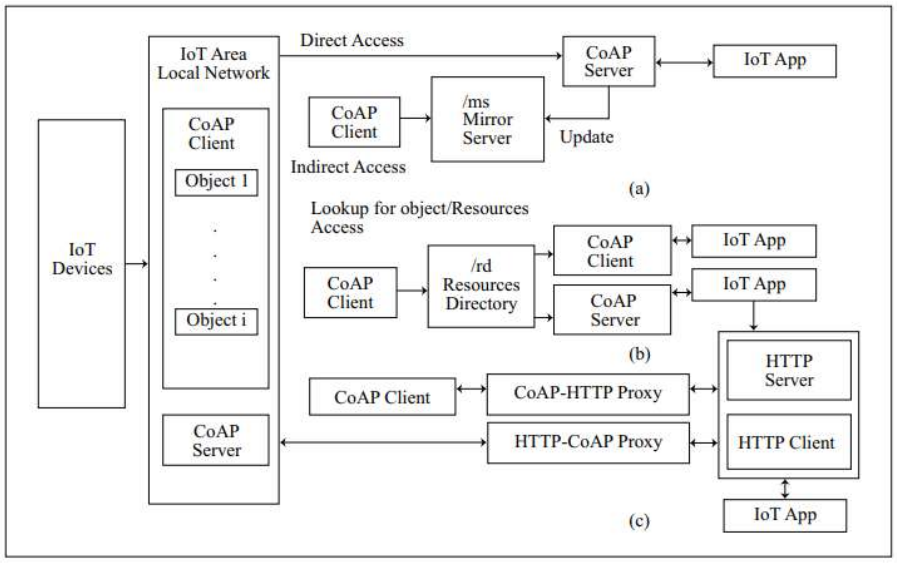
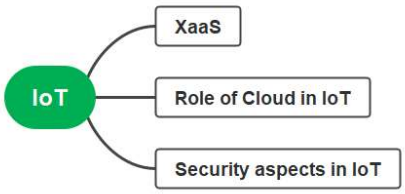


Figure 3.2 (a) Direct and indirect accesses of CoAP client objects to a CoAP server, (b) CoAP client access for lookup of object or resource using a resource directory, and (c) CoAP client and server access using proxies

4. **Basics of Networking**
5. **M2M and IoT Technology Fundamentals**
6. **Devices and gateways**
7. **Data Management**

8. **Business process in IoT**
9. **Everything as a service(XaaS)**
10. **Role of Cloud in IoT**
11. **Security aspects in IoT**



6.1 Introduction

A few conventional methods for data collection and storage are as follows:

● Saving devices' data at a local server for the device nodes

● Communicating and saving the devices' data in the files locally on removable media, such as micro SD cards and computer hard disks

● Communicating and saving the data and results of computations in a dedicated data store or coordinating node locally

● Communicating and saving data at a local node, which is a part of a distributed DBMS

● Communicating and saving at a remote node in the distributed DBMS

● Communicating on the Internet and saving at a data store in a web or enterprise

6.2 CLOUD COMPUTING PARADIGM FOR DATA COLLECTION, STORAGE AND COMPUTING

XAAS Everything-as-a-Service

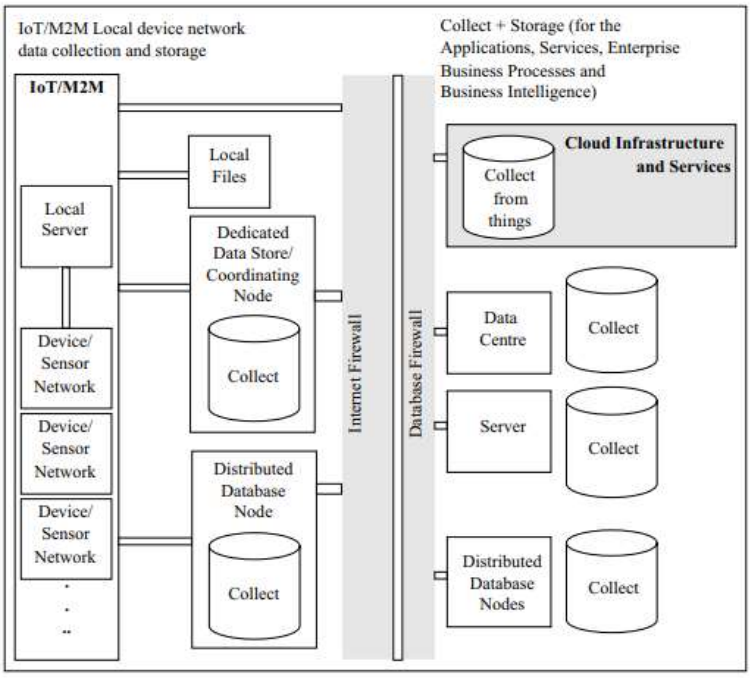


Figure 6.1 Devices or sensors network data collection at a device local-server, local files, dedicated data store, at a coordinating node, a local node of a distributed DBMS, Internet-connected server of data centre, server or distributed database nodes or a cloud infrastructure

(i) Devices or sensor networks data collection at the device web server, (ii) Local files, (iii) Dedicated data store at coordinating node, (iii) Local node in a distributed DBMS, (iv) Internet-connected data centre, (v) Internet-connected server, (vi) Internet-connected distributed DBMS nodes, and (vii) Cloud infrastructure and services.

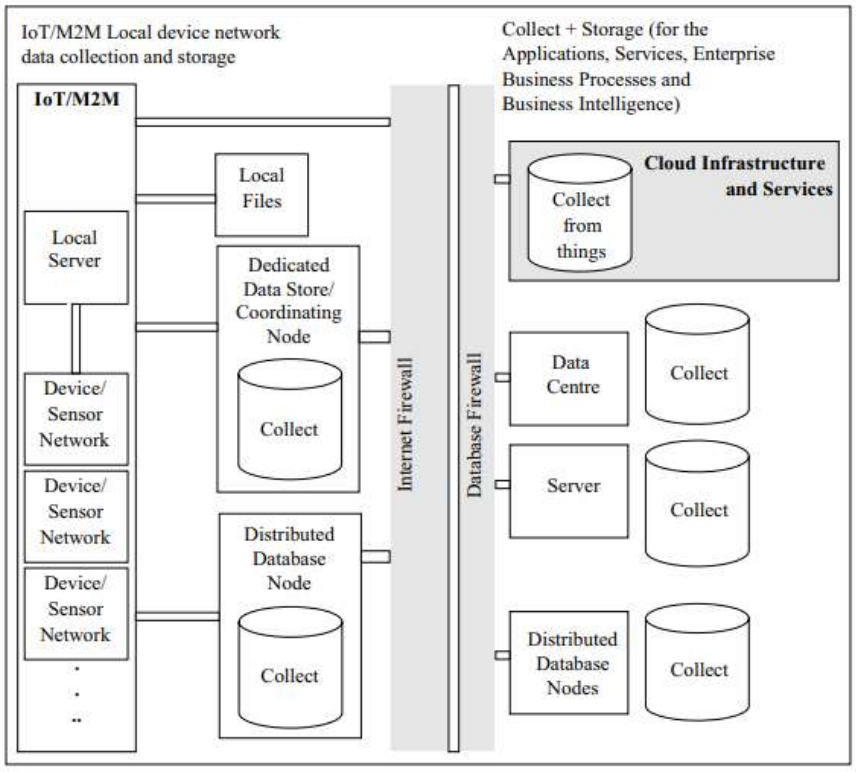


Figure 6.1 Devices or sensors network data collection at a device local-server, local files, dedicated data store, at a coordinating node, a local node of a distributed DBMS, Internet-connected server of data centre, server or distributed database nodes or a cloud infrastructure