INTERNET OF THINGS

SAMRAT KRISHNA GADDAM
NAVEEN VIJAYAKUMAR WATSON,ANGELIN GLADYS JESUDOSS

Made with ❤ on the Notion Press Platform
www.notionpress.com

TO MY PARENTS AND BROTHER

for raising me to believe that anything was possible.

Foreword

The book has been written in an easy-to-understand and student-friendly manner, and includes several illustrative figures and examples, sample codes and project case studies. The text explains architecture, design principles, hardware and software designs, while keeping multidisciplinary under-graduates and post-graduates in mind as primary readers. Each chapter begins by defining the learning objectives for each section, recall from previous chapters, introduction and the important terms covered in that chapter.The book is intended as a text for students of Computer Science and Engineering, Information Technology, Master of Computer Applications and M.Sc(Computer Science).

Preface

What if an umbrella could sense the local weather and remind the user if it is needed to be carried along that day; or if some kind of wearable device could monitor a patient’s health and when it is about to deteriorate, communicate the nature of emergency to the doctor directly and promptly; or if a car could have some computation and predictive analytics system which could apprise the user about the upcoming servicing schedules to avoid any sudden component failures beforehand? Internet of Things (IoT) and Internet-connected Cloud Platform-as-a-Service (PaaS) can make the above cases come true as real-life situations. In this book, the author takes the readers through all the important design and implementation details of various functions possible with IoT. For instance, the first chapter of the book introduces the concept of a smart umbrella which can, using specific sensors and computational devices and connectivity to the Internet, share weather updates with its users. A chapter on smart cars or connected cars elucidates how such cars can perform automatic detection of service requirements by direct transfer of service-relevant data to the car maintenance and service center or to the driver/car user server, and also send automatic reminders of servicing appointments, thereby lessening service visit.

Acknowledgements

First and foremost, we are grateful to all the co-authors Mr.Naveen Vijaya Kumar Watson and Mrs.Angelin Gladys Jesudoss for their time, effort, and understanding during the preparation of the book.It is an extraordinary delight to recognize the help and commitments of numerous people to this exertion. It is an enormous pleasure to acknowledge the support and contributions of many individuals to this effort. First, we would like to thank our editor, Notion Press Publishing for guidance, encouragement, and support. I also thank the management of P.B.Siddhartha College of Arts & Science, Vijayawada, Andhar Pradesh, India for their support.

Samrat Krishna Gaddam,
Assistant Professor,
P.B.Siddhartha College of Arts & Science,
Vijayawada, AP, India.
P.No-9177937461
Email-gsamratkrishna@pbsiddhartha.ac.in

Prologue

Chapter 1 gives an overview of 'Internet of Things'. It first gives vision and definitions of IoT, and meaning of smart hyperconnected devices which enable the IoT applications/ services. Next, the chapter describes IoT conceptual framework and architectural views, technology behind IoTs, communication modules and protocols such as MQTT. Then the chapter describes the sources of IoTs, such as RFIDs and wireless sensor networks. Machine-to-machine communication technologies now enhanced to IoTs. The chapter also introduces the concept of wearbale watches, smart home and smart cities.

Chapter 2 describes the design principles for connected devices. It describes IETF sixlayered design for IoT applications, ITU-T reference model and ETSI M2M domains and high-level capabilities. It then describes first architectural layer/device and gateway domain wireless and wired communication protocols and technologies. It then describes second architectural layer/device and gateway domain functionalities which are data enrichment, transcoding, consolidation, privacy issues, and device configuring, management, and ID management. The chapter also explains the need of ease of designing and affordability.

Chapter 3 describes the design principles for web connectivity, the data format standards JSON, TLV and MIME for communication, and devices CoAP, CoAP-SMS, CoAP-MQ, MQTT and XMPP protocols for the devices connectivity to the web. The chapter also describes SOAP, REST, HTTP RESTful and WebSockets methods, which the communication gateway deploys.

Chapter 4 describes Internet connectivity principles. Concepts such as IPv4, IPv6, 6LowPAN, HTTP, HTTPS, FTP, Telnet and other protocols used at IETF layer 6, application layer by IoT applications/services/processes have also been discussed.

Chapter 5 defines the methods of data acquiring, organising and analytics in IoT/M2M applications/services/business processes. The chapter covers data generation, acquisition, validation, data and events assembly and data store processes.

CHAPTER I
Internet of Things: An Overview

1.1 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. The following subsections describe IoT basics, definition, vision, conceptual frameworks and architectures.

1.1.1 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 in English has number of uses and meanings. In a dictionary, 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. Streetlight is also referred to as a thing. Example of reference to an action is— such a thing was not expected from him. Example of reference to a situation is—such things were in plenty in that regime. Thus, combining both the terms, the definition of IoT can be explained as follows: 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. Another source, defines the term IoT as follows: 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 linked image cannot be displayed. The file may have been moved, renamed, or deleted. Verify that the link points to the correct file and location.

Use of Internet of Things concept for streetlights in a city

1.1.3 Smart and Hyperconnected Devices

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. 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. 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. Several new terms have been used in the figure. Chapters ahead will define and explain these terms in more detail.

The linked image cannot be displayed. The file may have been moved, renamed, or deleted. Verify that the link points to the correct file and location.

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

1.2 IoT CONCEPTUAL FRAMEWORK

The following equation describes a simple conceptual framework of IoT2 : Physical Object + Controller, Sensor and Actuators + Internet = Internet of Things … 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. 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. 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. 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 IoT conceptual framework for the enterprise processes and services, based on a suggested IoT architecture given by Oracle (Figure 1.5 in Section 1.3). The steps are as 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 …

The steps are as follows:

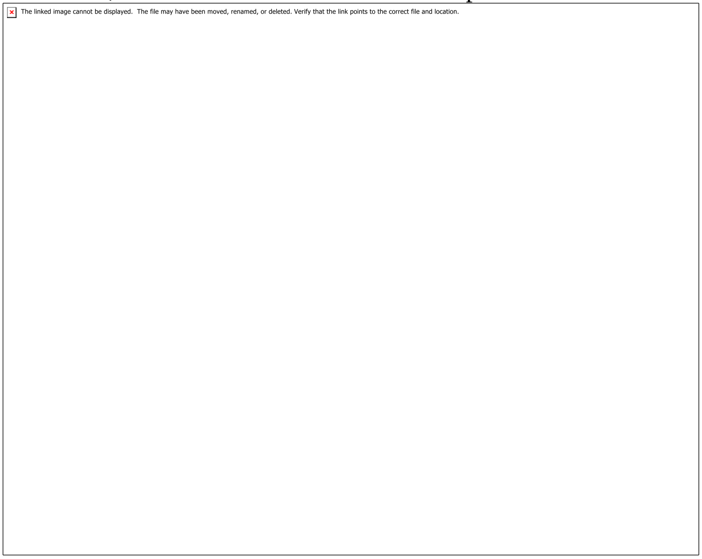
1. Levels 1 and 2 consist of a sensor network to gather and consolidate the data. First level gathers the data of the things (devices) using sensors circuits. The sensor connects to a gateway. Data then consolidates at the second level, for example, transformation at the gateway at level

2. The gateway at level 2 communicates the data streams between levels 2 and 3. The system uses a communication-management subsystem at level

3. An information service consists of connect, collect, assemble and manage subsystems at levels 3 and 4. The services render from level

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.

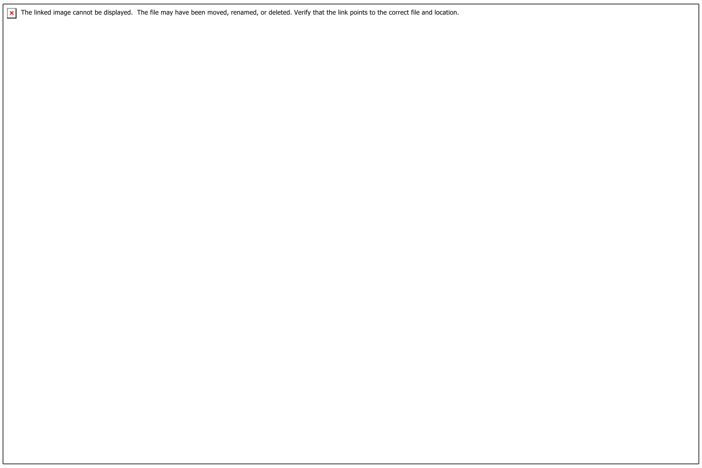
Various conceptual frameworks of IoT find number of applications including the ones in M2M communication networks, wearable devices, city lighting, security and surveillance and home automation. Smart systems use the things (nodes) which consist of smart devices, smart objects and smart services. Smart systems use the user interfaces (UIs), application programming interfaces (APIs), identification data, sensor data and communication ports.


The linked image cannot be displayed. The file may have been moved, renamed, or deleted. Verify that the link points to the correct file and location.

IBM IoT conceptual framework

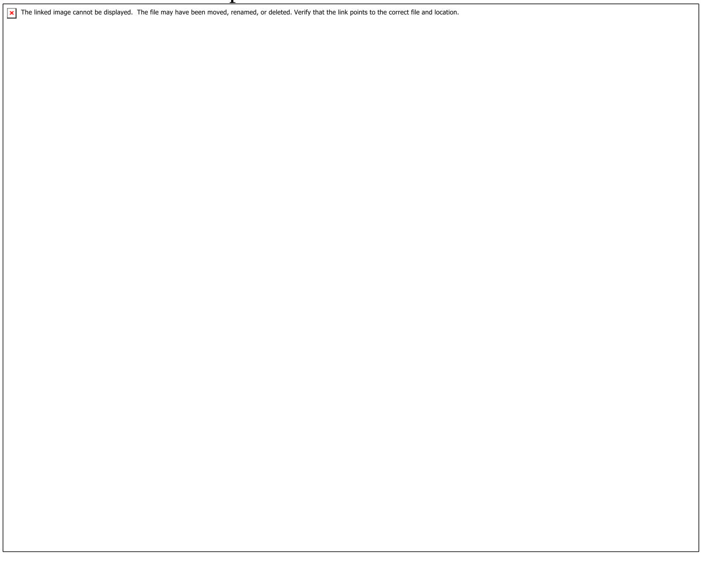
1.3 IoT ARCHITECTURAL VIEW

An IoT system has multiple levels . 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 .


The linked image cannot be displayed. The file may have been moved, renamed, or deleted. Verify that the link points to the correct file and location.

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

A reference model could be identified to specify reference architecture. Several reference architectures are expected to co-exist in the IoT domain.


The linked image cannot be displayed. The file may have been moved, renamed, or deleted. Verify that the link points to the correct file and location.

Oracle's IoT architecture

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. IoT architecture by Oracle

● 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 . A number of models (CISCO, Purdue and other models) have been proposed at SWG (Sub Working Group) Teleconference of December 2014. Standards for an architectural framework for the IoT have been developed under IEEE project P2413. IEEE working group is working on a set of guidelines for the standards.

IEEE suggested P24133 standard for architecture of IoT. It is a reference architecture which builds upon the reference model(s). The reference architecture covers the definition of basic architectural building blocks and their integration capability into multi-tiered systems. P2413 architectural framework4 is a reference model that defines relationships among various IoT verticals, for example, transportation and healthcare. P2413 provides for the following:

● Follows top-down approach (consider top layer design first and then move to the lowest)

● Does not define new architecture but reinvent existing architectures congruent with it

● Gives a blueprint for data abstraction

● Specifies abstract IoT domain for various IoT domains

● Recommends quality ‘quadruple’ trust that includes protection, security, privacy and safety

● Addresses how to document

● Strives for mitigating architecture divergence(s) Scope of IEEE P2413 standard defines an architectural framework for the IoT. It includes descriptions of various IoT domains, definitions of IoT domain abstractions and identification of commonalities between different IoT domains. Smart manufacturing, smart grid, smart buildings, intelligent transport, smart cities and e-health are different IoT domains. P2413 leverages existing applicable standards. It identifies planned or ongoing projects with a similar or overlapping scope.

1.4 TECHNOLOGY BEHIND IoT

The following entities provide a diverse technologyenvironment and are examples of technologies, which are involved in 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. An example of machine-learning software is GROK from Numenta Inc. that uses machine intelligence to analyse the streaming data from clouds and uncover anomalies, has the ability to learn continuously from data and ability to drive action from the output of GROK’s data models and perform high level of automation for analysing streaming data.6 The following five entities can be considered for the five levels behind an IoT system :

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

1.4.2 Major Components of IoT System

Major components of IoT devices are:

1. Physical object with embedded software into a hardware.

2. consisting of a microcontroller, firmware, sensors, control unit, actuators and communication module.

3. Communication module: Software consisting of device APIs and device interface for communication over the network and communication circuit/port(s), and middleware for creating communication stacks using 6LowPAN, CoAP, LWM2M, IPv4, IPv6 and other protocols.

4. for actions on messages, information and commands which the devices receive and then output to the actuators, which enable actions such as glowing LEDs, robotic hand movement etc.

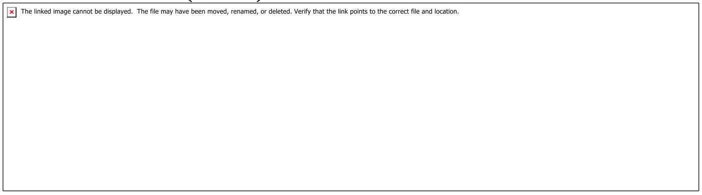
Sensors and Control Units

Sensors

Sensors are electronic devices that sense the physical environments. An industrial automation system or robotic system has multiple smart sensors embedded in it. Sensor-actuator pairs are used in control systems. A smart sensor includes computing and communication circuits.Each light has sensors for measuring surrounding light-intensity and surrounding traffic-proximity for sensing and transmitting the data after aggregation over a period. Sensors are used for measuring temperature, pressure, humidity, light intensity, traffic proximity, acceleration in an accelerometer, signals in a GPS, proximity sensor, magnetic fields in a compass, and magnetic intensity in a magnetometer. Sensors are of two types. The first type gives analog inputs to the control unit. Examples are thermistor, photoconductor, pressure gauge and Hall sensor. The second type gives digital inputs to the control unit. Examples are touch sensor, proximity sensor, metal sensor, traffic presence sensor, rotator encoder for measuring angles and linear encoders for measuring linear displacements.

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).


The linked image cannot be displayed. The file may have been moved, renamed, or deleted. Verify that the link points to the correct file and location.

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.

The linked image cannot be displayed. The file may have been moved, renamed, or deleted. Verify that the link points to the correct file and location.

IoT software components for device hardware

AllJoyn is an open-source OS created by Qualcomm. It is a cross platform OS with APIs available for Android, iOS, OS X, Linux and Windows OSs. It includes a framework and a set of services. It enables the manufacturers to create compatible devices. Spark is a distributed, cloud-based IoT operating system and web-based IDE. It includes a command-line interface, support for multiple languages and libraries for working with several different IoT devices. Contiki OS7 is an open-source multitasking OS. It includes 6LowPAN, RPL, UDP, DTLS and TCP/IP protocols which are required in low-power wireless IoT devices. Example of applications are street lighting in smart cities, which requires just 30 kB ROM and 10 kB RAM.

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. Eclipse developed the IoT programming language—Lua. Eclipse website provides sandbox environments for experimenting with the tools and a live demo. Eclipse-related popular projects are Paho, Koneki and Mihini.

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 Create (kit for prototyping), Kinoma Studio development environment and Kinoma Platform Runtime are three different open-source projects. Kinoma Connect is a free app for iOS and Android smartphones and tablets with IoT devices. APIs and Device Interfacing Components Connectivity interface consists of communication APIs, device interfaces and processing units.

1.4.5 Platforms and Integration Tools

ThingSpeak is an open data platform with an open API. It consists of APIs that enable realtime data collection, geolocation data, data processing and visualisations. It enables device status messages and plugins. It can process HTTP requests and store and process data. It can integrate multiple hardware and software platforms. It supports Arduino, Raspberry Pi,9 ioBridge/RealTime.io, and Electric Imp. An important feature of ThingSpeak is the support to MATLAB data analytics, mobile, web applications and social networks.

Nimbits is a cloud platform which supports multiple programming languages, including Arduino, JavaScript, HTML or the Nimbits.io Java library.10 The software deploys on Google App Engine, any J2EE server on Amazon EC2 or Raspberry Pi. It processes a specific type of data and can also store the data. The data can be time- or geo-stamped.

IoT Toolkit offers Smart Object API, HTTP-to-CoAP Semantic mapping and a variety of tools for integrating multiple IoT-related sensor networks and protocols.11

SiteWhere provisions a complete platform for managing IoT devices. It enables gathering of data and integrating it with external systems. SiteWhere can be used on Amazon's cloud or downloaded. It also integrates MongoDB, ApacheHBase and multiple big data tools. Other platforms are Microsoft Azure, TCS connected universe platform (TCS CUP), Xively, smartliving, thethings.io and exosite.

1.5 SOURCES OF IoT

Examples of hardware sources for IoT prototype development are

Arduino Yún, Microduino, Beagle Board and RasWIK. Hardware prototype needs an IDE for developing device software, firmware and APIs.

1.5.1 Popular IoT Development Boards

Arduino Yún

Arduino Yún board uses microcontroller ATmega32u4 that supports Arduino and includes Wi-Fi, Ethernet, USB port, micro-SD card slot and three reset buttons. The board also combines with Atheros AR9331 that runs Linux.

Microduino

Microduino is a small board compatible with Arduino that can be stacked with the other boards. All the hardware designs are open source.

Intel Galileo

Intel Galileo is a line of Arduino-certified development boards. Galileo is based on Intel x86 architecture. It is open-source hardware that features the Intel SOC X1000 Quark based Soc. Galileo is pin-compatible with Arduino. It has 20 digital I/O (12 GPIOs fully native), 12-bit PWM for more precise control, six analog inputs and supports power over Ethernet (PoE).

Intel Edison

Intel Edison is a compute module. It enables creation of prototypes and fast development of prototyping projects and rapidly produces IoT and wearable computing devices. It enables seamless device internetworking and device-to-cloud communication. It includes foundational tools. The tools collect, store and process data in the cloud, and process rules on the data stream. It generates triggers and alerts based on advanced analytics.

Beagle Board

Beagle Bone based board has very low power requirement. It is a card-like computer which can run Android and Linux. Both the hardware designs and the software for the IoT devices are open source.

Raspberry Pi Wireless Inventors Kit (RasWIK)

RasWIK enables Raspberry Pi Wi-Fi connected devices. It includes documentation for 29 different projects or you can come up with one of your own. There is a fee for the devices but all of the included code is open source, and you can use it to build commercial products as well.

1.5.2 Role of RFID and IoT Applications

Earlier IoT systems were internet-connected RFID based systems. RFID enables tracking and inventory control, identification in supply chain systems, access to buildings and road tolls or secured store centre entries, and devices such as RFID-based temperature sensors. RFID networks have new applications in factory design, 3PL-management, brand protection, and anti-counterfeiting in new business processes for payment, leasing, insurance and quality management.

1.5.3 Wireless Sensor Networks (WSNs)

Sensors can be networked using wireless technology and can cooperatively monitor physical or environmental conditions. Sensors acquire data from remote locations, which may not be easily accessible. Each wireless sensor also has communication abilities for which it uses a radio-frequency transceiver. Each node either has an analog sensor with signal conditioner circuit or a digital sensor. Sensing can be done to monitor temperature, light intensity, presence of darkness, metal proximity, traffic, physical, chemical and biological data etc.

WSN Definition Wireless Sensor Network (WSN) is defined as a network in which each sensor node connects wirelessly and has capabilities of computations for data compaction, aggregation and analysis plus communication and networking. WSN node is autonomous. Autonomous refers to independent computing power and capability to send requests and receive responses, and data forward and routing capabilities. A web source defines the WSN as “a wireless network consisting of spatially distributed autonomous devices using sensors to cooperatively monitor physical or environmental conditions, such as temperature, sound, vibration, pressure, motion or pollutants, at different locations.”

WSN Node

A WSN node has limited computing power. It may change topology rapidly. The WSN network in the topology-changing environment functions as an ad-hoc network. A WSN network in that environment is generally self-configuring, self-organising, self-healing and self-discovering.

1.6 M2M COMMUNICATION

Machine-to-machine (M2M) refers to the process of communication of a physical object or device at machine with others of the same type, mostly for monitoring but also for control purposes. Each machine in an M2M system embeds a smart device. The device senses the data or status of the machine, and performs the computation and communication functions. A device communicates via wired or wireless systems. The communication protocols are 6LowPAN, LWM2M, MQTT, and XMPP. Each communication device is assigned 48-bits Ipv6 address.

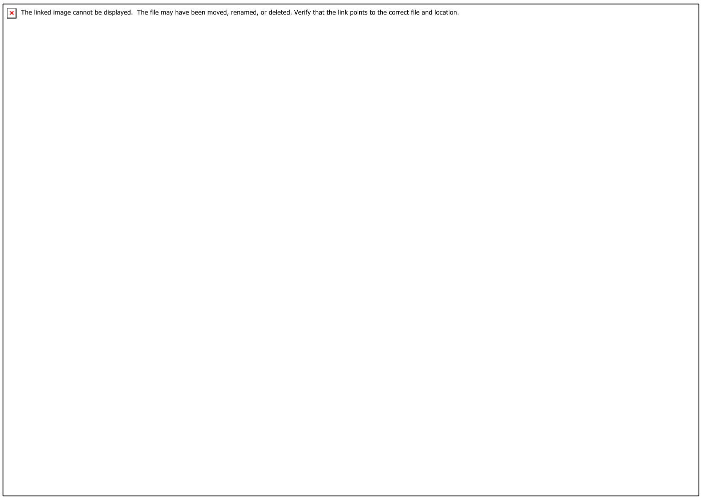
1.6.1 M2M to IoT

IoT technology in industry involves the integration of complex physical machinery M2M communication with the networks of sensors, and uses analytics, machine learning, and knowledge discovery software. M2M technology closely relates to IoT when the smart devices or machines collect data which is transmitted via the Internet to other devices or machines located remotely. The close difference between M2M and IoT is that M2M must deploy device to device, and carry out the coordination, monitoring, controlling of the devices and communicate without the usage of Internet whereas IoT deploys the internet, server, internet protocols and server or cloud end applications, services or processes. M2M has many applications in fields such as industrial automation, logistics, smart grid, smart cities, health and defence. Initial applications of M2M were found in automation and instrumentation only, but now these include telemetric applications and Industrial Internet of Things (IIoT) as well.

1.6.2 M2M Architecture

M2M architecture consists of three domains:

1. M2M device domain
2. M2M network domain
3. M2M application domain


The linked image cannot be displayed. The file may have been moved, renamed, or deleted. Verify that the link points to the correct file and location.

Three domains of M2M architecture

M2M device communication domain consists of three entities: physical devices, communication interface and gateway. Communication interface is a port or a subsystem, which receives the input from one end and sends the data received to another. M2M network domain consists of M2M server, device identity management, data analytics and data and device management similar to IoT architecture (connect + collect + assemble + analyse) level. M2M application domain consists of application for services, monitoring, analysis and controlling of devices networks.

1.6.3 Software and Development Tools

Examples of M2M software and development tools are as follows:

● **Mango** is an open source M2M web-based software. It supports multiple platforms, multiple protocols, databases, meta points, user-defined events and import/export.20

● **Mainspring** from M2MLabs is a development tool, and source framework for developing M2M applications.

It enables:

❍ Flexible modelling of devices and their configurations

❍ Communication between devices and applications

❍ Validation and normalisation of data

❍ Long-term data storage and data retrieval functions

❍ Programming in Java and Apache Cassandra

❍ Usages of no SQL database.

● **DeviceHive** is an M2M communication framework. It is an M2M platform and integration tool. It enables connecting devices to the IoT. It includes web-based management software that creates security-rules-based e-networks and monitoring devices. The web software enables prototype projects built with DeviceHub and online tests to find out how it works.

● **Open M2M protocols,Tools and Framework:** Following are the open protocols, tools and frameworks for M2M:

❍ XMPP, MQTT-OASIS standards group and OMA LWM2M-OMA standard group for protocol

❍ Various projects of Eclipse M2M industry working groups' are Koneki, Eclipse SCADA for open standards for communication protocols, tools and frameworks

❍ ITU-T Focus Group M2M global standardisation initiative for a common M2M service layer

❍ 3GPPP study group for security aspects of M2M equipment and automatic SIM activation covering remote provisioning and change of subscription.

❍ Weightless (wireless communications) group for standards and using wireless spaces for M2M.

1.7 EXAMPLES OF IoT

Examples of IoT usages are wearable devices such as watches, fitness trackers, sleep monitors and heart monitors etc. Fitbit (for example, Fitbit Alta fitness tracker), Garmin and other companies manufacture many such devices. Microsoft (Microsoft band might soon be discontinued), Xiaomi and other manufacturers make tracking bands. A fitness tracker wearable band has the following functions:

● Track steps, distance, calories burned and active minutes

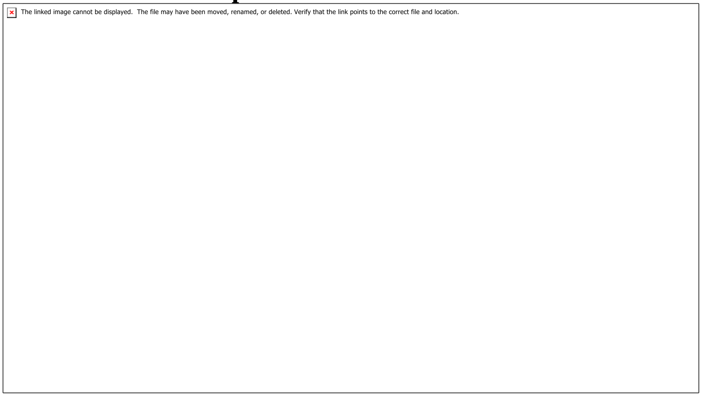
● See stats and time with a bright OLED tap display

● Automatically track how long and how well you sleep and set a silent, vibrating alarm

● Personalize with interchangeable metal, leather and classic bands

● Get calls, texts and calendar notifications at a glance when the phone is in a defined range. Clothing and accessories nowadays incorporate computer and advanced electronic technologies. The design of watches, rings and bands often includes practical functions and features.

1.7.1 Wearable Smart Watch Following are the examples of wearable watches based on IoT concept:


The linked image cannot be displayed. The file may have been moved, renamed, or deleted. Verify that the link points to the correct file and location.

Features of hyperconnected wearable smart watches

1.7.2 Smart Home

Sensors and actuators manage a smart home with an Internet connection. Wired and wireless sensors are incorporated into the security sensors, cameras, thermostats, smart plugs, lights and entertainment systems. Do-it-Yourself (DIY) sensors and actuators, include smart plug, motion detector, door/window detector, smoke detector, energy meter interface (electric, gas, water), remote control (built-in authentication), smart relay, surveillance camera, Wireless Hi-Fi speakers, HUE LED lights, electric utility meter etc.

A connected home has the following applications deployed in a smart home:

● Mobile, tablets, IP-TV, VOIP telephony, video-conferencing, video-on-demand, videosurveillance, Wi-Fi and internet

● Home security: Access control and security alerts

● Lighting control

● Home healthcare

● Fire detection or Leak detection

- Energy efficiency
- Solar panel monitoring and control
- Temperature monitoring and HVAC control
- Refrigerator network with maintenance and service centres
- Automated meter reading

Home Automation Software

Intel based intelligent gateway enables creation of a home automation system offered by service providers for telephony, mobile, cable, broadband and security. OpenHAB (open Home Automation Bus) enables the smart home devices that communicate via home. It has a companion cloud computing service called my.openHAB. It runs on a Java-enabled system. The Thing System enables the smart home devices which communicate and are controlled via home. The language used is Node.js. It deploys a Raspberry Pi and consists of software components and network protocols for all Internet-connected things at home.

1.7.3 Smart Cities

The IoT concept extends to Internet of Everything (IoE) for developing smart cities.

1. Layer 1 consists of sensors, sensor networks and devices network in parking spaces, hospitals, streets, vehicles, banks, water supply, roads, bridges and railroads. Bluetooth, ZigBee, NFC, WiFi are the protocols used at this layer.
2. Layer 2 captures data at distributed computing points where data is processed, stored and analysed.
3. Layer 3 is meant for central collection services, connected data centres, cloud and enterprise servers for data analytics applications.
4. Layer 4 consists of new innovative applications, such as waste containers' monitoring, WSNs for power loss monitoring, bike sharing management and smart parking. Smart parking refers to services for motorists that informs them about the nearby parking services with vacant spaces in advance.

CHAPTER II
Design Principles for Connected Devices

2.1 INTRODUCTION

When a letter is written then it is written according to a protocol (etiquette). To send a letter, it is first put in an envelope, and then the envelope is marked with the receiver's address at the centre, sender's address at left hand bottom area, stamp(s) is/are affixed at right hand top corner and the type of post is mentioned on top line in the centre. All letters are then gathered (stacked) and the stack is sent to the target city. Each action takes place according to a specified protocol at each stage (layer). Similarly, when data is transferred from a sensor, then functional units create a stack for data communication to an application or service.

IoT or M2M device data refers to the data meant for communication to an application, service or process. Data also refers to data received by a device for its monitoring or for actions at actuator in it.

Data stack denotes the data received after the actions at various in-between layers (or levels or domains). Layers in Open Systems Interconnection (OSI) model are Application, Presentation, Session, Transport, Network, Data-link and Physical.

Actions at the data-adaptation or other layers can be related to data privacy, data security, data consolidation, aggregation, compaction and fusion. An action can be a gateway action—using one protocol for reception and another one for transmission.

The actions at a layer can be adding additional header after appropriate data formatting or transformation at functional units. For example, data from a device can be used by an application or in-between layers when a header (specifying device ID or address, destination addresses and other fields) adds and encrypts at the physical (device) layer for its security on the network.

Actions besides appending a header can be of appending the additional bits at the various in-between units. For example, appending additional message(s) or an acknowledgement.

Following are the key terms which need to be understood to learn the design principles of connected devices for IoTs:

Layer refers to a stage during a set of actions at which the action is taken as per a specific protocol or method, and then the result passes to the next layer until the set of actions complete. A layer may consist of various sublayers.

Physical layer refers to a layer at transmitting-node or at the receiving node for the data bits. The transfer uses physical systems and refers to wireless or wired transmission. This layer is the lowest layer.

Application layer refers to a layer for transmitting or receiving the data bits of an application. Data bits route across the network and transfer takes place as follows: application data from the application layer transfers after passing through several in-between layers to the physical layer, and from there it transmits to the receiving-end physical layer. Then, the data at the receiving node transfers from the physical layer to the application layer after passing through several in-between layers.

Level refers to a stage from the lowest to the highest. For example, acquiring device data and actions that may be considered at the lowest level and actions in business processes at the highest level.

Domain refers to a set of software, layers or levels having specific applications and capabilities. For example, CoRE network, access network, service capabilities and applications can be considered as one domain, say, network domain. A domain generally has limited interactions with other domains or outside the domain.

Gateway refers to software for connecting two application layers, one at the sender and the other at the receiver [application layer gateway (ALG)]. A gateway may be of different types. A communication gateway at device and gateway domain has capabilities as protocol-conversion during communication between two ends when each end uses distinct protocols. An Internet gateway may have capabilities besides protocol conversion, transcoding data, device management and data-enrichment before the data communicate over the Internet.

IP stands for Internet Protocol version 6 (IPv6) or Internet Protocol version 4 (IPv4) for the network layer (v6 means version 6, v4 version 4)

Header means a set of octets containing information about the data being sent. Header packs the data of a layer before transmission to the next layer during communication between two end-points. The size of a header and its fields are according to the protocol used for creating data stack at a layer. For example, IPv4 header has fields as per IP network layer, Universal Datagram Protocol (UDP) header as per UDP at the transport layer and so on. Each header field has distinct meanings. The field size can be between 1 and 32-bit in a packet. A field helps in processing the packet when transferring it from one layer to the next one.

Packet means packaged data-stack which routes over the network. Packet size limit is according to the protocol. For example, IPv4 packet size limit is 216 B (214 words with 1 word = 4 octet).

Protocol Data Unit (PDU) is a unit of data which is specified in a protocol of a given layer which transfers from one layer to another. For example, PDU is bit which transfers from physical layer; frame from data-link layer; packet from network layer; segment from transport layer and text (plain, encrypted or compressed) from application and other layers.

Maximum Transmission Unit (MTU) is the largest size frame or packet or segment specified in octets (1 octet = 1 byte = 8 bits) that can be sent in a packet or frame-based network such as the Internet. For example, consider transfers of a segment from the transport layer using the Transmission Control Protocol (TCP) to the network layer. The MTU determines the maximum size of each data stack in any transfer to the network layer. The network layer determines the maximum size of each frame in any transfer to the data-link layer and then uses MTU of the data-link layer.

Star network denotes the number of nodes interacting with a coordinator or master node.

Mesh network denotes the number of nodes that may interconnect with each other.

End-point device or node denotes the one that provides connectivity to a coordinator or router.

Coordinator denotes the one that connects to a number of end-points as well as routers in a star topology and forwards the data stack from one attached end point/router to another.

Master refers to the one who initiates the pairing with the devices in a star topology network.

Slave means one that pairs with a master, uses the clock signals from master for synchronisation and uses address assigned by the master at the beginning.

Router refers to a device or node capable of storing paths to each destination to which it has logical links. The router sends the data stack according to the available path or paths at a receiving instance.

ISM band means Industrial, Scientific and Medical (ISM) radio frequency (RF) bands. 2.4 GHz and the frequencies are 915 MHz for North America, 868 MHz for Europe and 433 MHz band for Asia in ISM bands.

Application means software for specific tasks, such as streetlight monitoring or control.

Service means service software, for example, report generation or chart visualisation service.

Process means a software component, which processes the input and generates the output; for example after analysing the data or acquiring the data. An operating system controls a process, memory for the process and other parameters of the process

Suggested models, conceptual frameworks, reference models and architectures for the IoT/M2M suggest the following:

● Need of designing a communication framework for connecting devices, for their local area networking and provisions for data gathering.

● Need of designing a data enrichment, data consolidation and data transformation framework.

● Need of designing gateway components for connecting the device's network with the web/Internet.

● Need of application and applications-support frameworks for services, applications and processes.

2.2 IoT/M2M SYSTEMS, LAYERS AND DESIGNS STANDARDISATION

A number of international organisations have taken action for IoT design standardisation. Following are the examples

Internet Engineering Task Force (IETF), an international body initiated actions for addressing and working on the recommendations for the engineering specifications for the Internet of Things. IETF suggests the specifications for the layers, and the engineering aspects for the IoT communication, networks and applications.

International Telecommunication Union for Telecommunication (ITU-T) suggested a reference model for IoT domain, network and transport capabilities for the IoT services and the applications at the application and application-support layers.

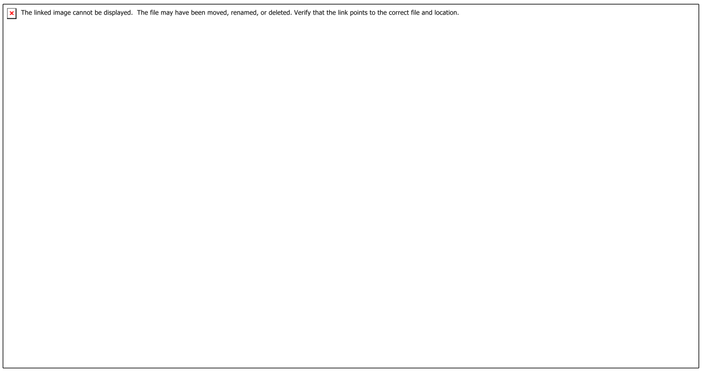
European Telecommunication Standards Institute (ETSI) initiated the development of a set of standards for the network, and devices and gateway domains for the communication between machines (M2M). ETSI proposed high-level architecture for applications and service capabilities. Open Geospatial Consortium (OGC), an International Industry Consortium, has also suggested open standards for sensors' discovery, capabilities, quality and other aspects with support to geographical information web support.

2.2.1 Modified OSI Model for the IoT/M2M Systems

OSI protocols mean a family of information exchange standards developed jointly by the ISO and the ITU-T. The seven-layer OSI model is a standard model. It gives the basic outline for designing a communication network. Various models for data interchanges consider the layers specified by the OSI model, and modify it for simplicity according to the requirement. Similarly, IETF suggests modifications in the OSI model for the IoT/M2M.

Data communicates from device end to application end. Each layer processes the received data and creates a new data stack which transfers it to the next layer. The processing takes place at the in-between layers, i.e. between the bottom functional-layer to the top layer. Device end also receives data from an application/service after processing at the in-between layers.

Gather + Enrich + Stream + (Manage + Acquire + Organise +Analyse) = IoT Applications and Services


The linked image cannot be displayed. The file may have been moved, renamed, or deleted. Verify that the link points to the correct file and location.

Seven-layer generalised OSI model (on left) and IETF six layer modified OSI model for IoT/ M2M

New applications and services are present at the application layer 6. A modification to this is that the application-support layer 5 uses protocols, such as CoAP. IoT applications and services commonly use them for network communication. The CoAP protocol at the layer is used for the request/response interactions between the client and server at the network. Similarly, the application-support layer may include processes for data managing, acquiring, organising and analysing which are mostly used by applications and services.

Modifications are also at the data-link layer 2 (L2) and physical layer 1 (L1). The new layers are data-adaptation (new L2) and physical cum data-link (new L1). The dataadaptation layer includes a gateway. The gateway enables communication between the devices network and the web.

2.2.2 ITU-T Reference Model

ITU-T reference model RM1. It also shows correspondence of the model with the six-layers modified OSI model. The figure also shows a comparison with CISCO IoT reference model RM2. RM1 considers four layers which are

Lowest layer, L1, is the device layer and has device and gateway capabilities.

● Next layer, L2, has transport and network capabilities.

● Next layer, L3, is the services and application-support layer. The support layer has two types of capabilities—generic and specific service or application-support capabilities.

● Top layer, L4, is for applications and services. ITU-T recommends four layers, each with different capabilities. A comparison of ITU-T RM1 with the six-layer OSI model can be made as follows:

● RM1 device layer capabilities are similar to data-adaptation and physical cum datalink layers.

● RM1 network layer capabilities are similar to transport and network layers.

● RM1 upper two layer capabilities are similar to top two layers. A comparison with the CISCO IoT reference model (RM2) can be made as follows:

● RM1 L4 capabilities are similar to RM2 collaborations and processes, and application top two levels.

● RM1 L3 capabilities are similar to RM2 three middle-level functions of data abstraction, accumulation, analysis and transformation.

● RM1 L2 layer capabilities are similar to RM2 functions at connectivity level.

● RM1 L1 device layer capabilities are similar to RM2 functions at physical devices level.

The linked image cannot be displayed. The file may have been moved, renamed, or deleted. Verify that the link points to the correct file and location.

ITU-T reference model RM1, its correspondence with six layers of modified OSI and a comparison with seven levels suggested in CISCO IoT reference model RM2

2.2.3 ETSI M2M Domains and High-level Capabilities

A domain specifies the functional areas. High-level architecture means architecture for functional and structural views. ETSI M2M domains and architecture, and the high-level capabilities of each domain. It also shows that the architecture correspondences with the six-layer modified OSI model as well as the four layers of the ITU-T reference model.

The ETSI network domain has six capabilities and functions:

1. M2M applications
2. M2M service capabilities
3. M2M management functions
4. Network management functions
5. CoRE network (for example, 3G and IP networks, network control functions, interconnections among networks)
6. Access network (for example, LPWAN (low power wide area network), WLAN (Wi-Fi) and WiMax networks).

The ETSI device and gateway domain has the following functional units:

● Gateway between M2M area network, and CoRE and access network, possessing M2M service capabilities and applications

● M2M area network (for example, Bluetooth, ZigBee NFC, PAN, LAN)

● M2M devices

The linked image cannot be displayed. The file may have been moved, renamed, or deleted. Verify that the link points to the correct file and location.

ETSI M2M domain architecture and its high-level capabilities, and its correspondences with six layers of modified OSI and four layers of ITU-T reference model

2.3 COMMUNICATION TECHNOLOGIES

Physical cum data-link layer in the model consists of a local area network/personal area network. A local network of IoT or M2M device deploys one of the two types of technologies— wireless or wired communication technologies.number of devices present in an IoT or M2M devices network. The figure shows the local area network of devices. The connectivity between the devices (left-hand side) is by using RF, Bluetooth Smart Energy, ZigBee IP, ZigBee NAN (neighbourhood area network), NFC or 6LoWPAN or mobile. Tens of bytes communicate at an instance between the device and local devices network.

2.3.1 Wireless Communication Technology

Physical cum data-link layer uses wired or wireless communication technologies. Examples of wireless communication technologies are NFC, RFID, ZigBee,Bluetooth (BT), RF transceivers and RF modules. Following subsections describe these wireless communication technologies.

Near-Field Communication Near-Field communication (NFC) is an enhancement of ISO/IEC2 14443 standard for contact-less proximity-card.

NFC is a short distance (20 cm) wireless communication technology. It enables data exchange between cards in proximity and other devices. Examples of applications of NFC are proximity-card reader/RFID/IoT/M2M/mobile device, mobile payment wallet, electronic keys for car, house, office entry keys and biometric passport readers.

NFC devices transmit and receive data at the same instance and the setup time (time taken to start the communication) is 0.1 s. The device or its reader can generate RF fields for the nearby passive devices such as passive RFID. An NFC device can check RF field and detect collision of transmitted signals. The device can check collision when the received signal bits do not match with the transmitted signal bits. Features of an NFC device are:

Range of functioning is within 10 to 20 cm. The device can also communicate with Bluetooth and Wi-Fi devices in order to extend the distance from 10 cm to 30 m or higher. The device is able to receive and pass the data to a Bluetooth connection or standardised LAN or Wi-Fi using information handover functions. Device data transfer rates are 106 kbps, 212 kbps, 424 kbps and 848 kbps (bps stands for bit per second, kbps for kilo bit per second). Three modes of communication are:

1. Point-to-point (P2P) mode: Both devices use the active devices in which RF fields alternately generate when communicating.

2. Card-emulation mode: Communication without interruption for the read and write as required in a smart card and smart card reader. FeliCa™ and Mifare™ standards are protocols for reading and writing data on the card device and reader, and then the reader can transfer information to Bluetooth or LAN.

3. Reader mode: Using NFC the device reads passive RFID device. The RF field is generated by an active NFC device. This enables the passive device to communicate.

RFID

Radio Frequency Identification (RFID) is an automatic identification method. RFIDs use the Internet. RFID usage is, therefore, in remote storage and retrieval of data is done at the RFID tags. An RFID device functions as a tag or label, which may be placed on an object. The object can then be tracked for the movements. The object may be a parcel, person, bird or an animal. IoT applications of RFID are in business processes, such as parcels tracking and inventory control, sales log-ins and supply-chain management.

Bluetooth BR/EDR and Bluetooth Low Energy

Bluetooth devices follow IEEE 802.15.1 standard protocol for L1 (physical cum data-link layer). BT devices form a WPAN devices network. Two types of modes for the devices are Bluetooth BR/EDR (Basic Rate 1 Mbps/Enhanced Data Rate 2 Mbps and 3 Mbps) and Bluetooth low energy (BT LE 1Mbps). A latest version is Bluetooth v4.2. BT LE is also called Bluetooth Smart. Bluetooth v4.2 (December 2014) provides the LE data packet length extension, link layer privacy and secure connections, extended scanner and filter link layer policies and IPSP. BT LE range is 150 m at 10 mW power output, data transfer rate is 1 Mbps and setup time is less than 6 s.

Bluetooth v5, released in June 2016, has increased the broadcast capacity by 800%, quadrupled the range and doubled the speed. A device may have provisions for single mode BT LE or dual mode BT BR/EDR (Mbps stands for Million Bits per second). Its features are:

● Auto-synchronisation between mobile and other devices when both use BT. BT network uses features of self-discovery, self-configuration and self-healing.

● Radio range depending on class of radio; Class 1 or 2 or radios: 100 m, 10 m or 1 m used in device BT implementation.

● Support to NFC pairing for low latency in pairing the BT devices.

● Two modes—dual or single mode devices are used for IoT/M2M devices local area network.

● IPv6 connection option for BT Smart with IPSP (Internet Protocol Support Profile). ● Smaller packets in LE mode.

● Operation in secured as well as unsecured modes (devices can opt for both link-level as well as service-level security or just service level or unsecured level).

● AES-CCM 128 authenticated encryption algorithm for confidentiality and authentication (Refer Example 2.4).

● Connection of IoT/M2M/mobile devices using BT EDR device to the Internet with 24 Mbps Wi-Fi 802.11 adaptation layer (AMP: Alternative MAC/PHY layer) or BT-enabled wire-bound connection ports or device. MAC stands for media access control sublayer at a data-link layer/sublayer.

ZigBee IP/ZigBee SE 2.0

ZigBee devices follow the IEEE 802.15.4 standard protocol L1 (physical cum data-link layer). ZigBee devices form a WPAN devices network. ZigBee end-point devices form a WPAN of embedded sensors, actuators, appliances, controllers or medical data systems which connect to the Internet for IoT applications, services and business processes. ZigBee Neighbourhood Area Network (NAN) is a version for a smart grid. ZigBee smart energy version 2.0 has energy management and energy efficiency capabilities using an IP network.

The features of ZigBee IP are:

● L1 layer PDU = 127 B

● Used for low-power, short-range WPAN

● The device can function in six modes—end point, ZigBee-ZigBee devices router, ZigBee network coordinator, ZigBee-IP coordinator, ZigBee-IP router and IP host.

● ZigBee IP enhancement provisions the IPv6 connectivity. A ZigBee IP device is a Reduced Function Device (RFD). RFD means one that functions for the ‘sleepy’/ battery-operated device. Sleepy means one that wakes up infrequently, sends data and then goes back to sleep. ZigBee IP supports IPv6 network with 6LoWPAN header compression, connection for Internet communication and control of low power devices, TCP/UDP transport layer and TLSv1.2 public key (RSA and ECC) and PSK cipher suite for end-to-end security protocol, end-to-end means application layer to physical layer.

● The ZigBee router uses reactive and proactive protocols for routing mode, which enable applications in big-scale automation and remote controls.

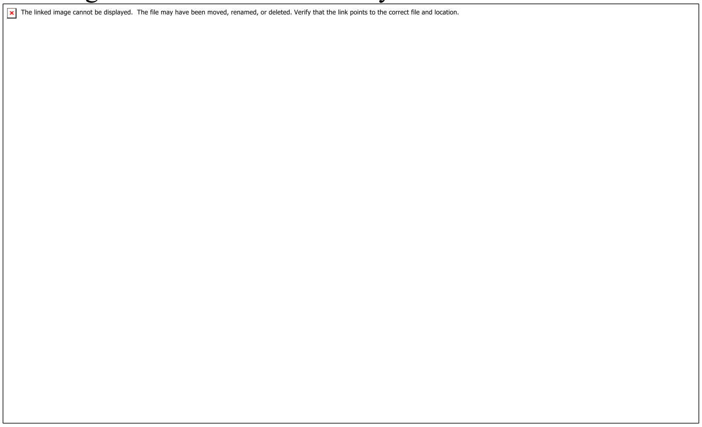
● A self-configuring and self-healing dynamic pairing mesh network, supports both multicast and unicast options.

● Multicast forwarding to support multicast Domain Name System (mDNS) based service discovery (SD)

● Three end devices, two routers, one sensor node connected to coordinator ZigBee devices forming a star network. ● One end device, two routers and one coordinator forming a mesh network.

● Mesh network router connects to an AP/gateway, which in turn connects to a cellular network.

● Coordinator of mesh network connects to ZigBee IP border router, which enables local ZigBee networks' connectivity to the Internet.


The linked image cannot be displayed. The file may have been moved, renamed, or deleted. Verify that the link points to the correct file and location.

ZigBee end point, coordinator, router, ZigBee IP router nodes forming the star, mesh and IP networks of ZigBee sensors, end devices, and ZigBee router devices which interconnect to Internet IPv4, IPv6 and cellular networks

Features of a ZigBee network are:

● A router in star network connects to 6LoWPAN, which connects an IEEE 802.15.4 devices network to IPv6 network.

● 1000s of byte communicate between the network layer and IoT web objects.

● 127 B communication between the adaptation layer IEEE 802.15.4 devices at single data transfer.

● IETF ND (Neighbour Discovery), ROLL (Routing Over Low power Loss Network), RPL routing, IPv6/IPv4 network, TCP/UDP/ICMP transport, SSL/TLS security layer protocols for the communication between web object/application and ZigBee devices.

Wi-Fi

Wi-Fi is an interface technology that uses IEEE 802.11 protocol and enables the Wireless Local Area Networks (WLANs). Wi-Fi devices connect enterprises, universities and offices through home AP/public hotspots. Wi-Fi connects distributed WLAN networks using the Internet.

Automobiles, instruments, home networking, sensors, actuators, industrial device nodes, computers, tablets, mobiles, printers and many devices have Wi-Fi interface. They network using a Wi-Fi network.

Wi-Fi is very popular. The issues of Wi-Fi interfaces, APs and routers are higher power consumption, interference and performance degradation.

Wi-Fi interfaces connect within themselves or to an AP or wireless router using Wi-Fi PCMCIA or PCI card or built-in circuit cards and through the following:

● A WLAN transceiver or BS can connect one or many wireless devices simultaneously to the Internet.

● Peer-to-peer nodes without access point: Client devices within Independent Basic Service Set (IBSS) network can communicate directly with each other. It enables fast and easy setting of an 802.11 network.

● Peer to multipoint nodes with Basic Service Sets (BSSs) using one in-between AP point or distributed BSSs connect through multiple APs.

The linked image cannot be displayed. The file may have been moved, renamed, or deleted. Verify that the link points to the correct file and location.

Three WLAN networks for sensor device nodes, mobiles, tablets, laptops, computers and Internet connectivity of WLAN networks with the IP4 networks

● Connectivity range of each BSS depends on the range of wireless bridges and antennae used and environmental conditions.

● Each BSS is a Service Set Identifier (SSID).

Diagram shows the following:

● Sensor nodes connected to BT with Wi-Fi adaptation, 802.11 interfaces in a WLAN network 1 (WLAN1).

● Tablets, Wi-Fi, computers also connect in WLAN 1 though an AP.

● AP1 connects to a broadband router 1 and to the IP4 network 2.

● WLAN1 and WLAN2 function as BSS.

● WLAN 2 also consists of AP2, Wi-Fi router and other Wi-Fi enabled interfaces.

● Wi-Fi router connects to multiple Wi-Fi nodes as well as to a broadband router 2.

● Broadband routers 1 and 2 connect using wires, to IP4 networks and web objects for IoT apps, services and processes. The Wi-Fi interfaces, access points, routers features are as follows:

● Generally used are the 2.4 GHz IEEE 802.11b adapters or 5 GHz (802.11a or 802.11g) or 802.11n or other 802.11 series protocols.

● Interfaces use 2.4 GHz or 5 GHz antenna ● Offers mobility and roaming.

RF Transceivers and RF Modules

RF transmitters, receivers, and transceivers are the simplest RF circuits. A transceiver transmits the RF from one end and receives the RF from the other end, but internally has an additional circuit, which separates the signals from both ends. An oscillator generates RF pulses of required active duty cycle and connects to a transmitter. BT, ZigBee, and WiFi radios deploy ISM band transceivers, which have comparatively complex circuits.

IoT/M2M applications deploy ISM band RF modules with transceivers or just transmitter or receiver. A number of systems use RF modules for applications needing wireless connectivity; for example, security, telemetry, telematics, fleet management, home automation, healthcare, automobiles wireless tire pressure monitors, back-up cameras and GPS navigation service, payment wallet, RFID and maintenance.

RF technology consists of the following elements:

● RF interface/physical layer, RF signals transmit between the nodes or endpoints, i.e. the sensors, actuators, controllers and a gateway where signals are received. Physical layer specifications consist of signal aspects and characteristics, including the frequencies, modulation format, power levels, the transmitting and receiving mode and signalling between end-point elements.

● RF network architecture includes the overall system architecture, backhaul, server and bidirectional end-devices with radio duty cycling in the applications. Radio duty cycling means managing the active intervals, transmission and receiving schedule, and time-intervals actions on an event during the active intervals and actions during the inactive (sleep) intervals using the RF Integrated Circuits (RFICs).

GPRS/GSM Cellular Networks-Mobile Internet

An IoT/M2M communication gateway can access a Wireless Wide Area Network (WWAN). The network access may use a GPRS cellular network or new generation cellular network for Internet access. A mobile phone provisions for a USB wired port, BT and Wi-Fi connectivity. Wireless connectivity for Internet uses data connectivity using GSM, GPRS, UMTS/LTE and WiMax services of a mobile service provider or Wi-Fi using a modem. A phone, generally, provisions for number of sensors also; for example, acceleration, GPS and proximity.

Wireless USB

Wireless USB is a wireless extension of USB 2.0 and it operates at ultra-wide band (UWB) 5.1 GHz to 10.6 GHz frequencies. It is for short-range personal area network (high speed 480 Mbps 3 m or 110 Mbps 10 m channel). FCC recommends a host wire adapter (HWA) and a device wire adapter (DWA), which provides wireless USB solution. Wireless USB also supports dual-role devices (DRDs). A device can be a USB device as well as limited capability host.

2.3.2 Wired Communication Technology

Wired communication can be serial asynchronous communication (for example, UART interface) or synchronous serial communication (for example, SPI interface or parallel input, output and input-output ports at the devices). Communication can be over a bus when a number of systems (chips, units, integrated circuits or ports or interfacing circuits) connect through a common set of interconnections.

\Wired communication can be done using Ethernet IEEE 802.2 bus specifications. A MAC sublayer data frame may be according to Ethernet Protocol. Wired communication can also use a USB port, a microUSB or a USB 3.0 adapter.

UART/USART Serial Communication

A Universal Asynchronous Transmitter (UART) enables serial communication (transmission) of 8 bits serially with a start bit at the start of transmission of a byte on serial Transmitter Data (TxD) output line. Serial means present one after another at successive time intervals.

Asynchronous refers to all bytes in a frame transmit, which can result in variation in time interval spacing or phase differences between successive bytes and in-between wait interval. This is because clock information of transmitter does not transmit along with the data. The receiver clock also does not synchronise with the data. Further, successive set of bytes may wait after transmission till an acknowledgement is received from the receiving end.

The intervals of each transmit bit are controlled by a clock. When the clock period is T = 0.01 ms, then period for transmission of a byte on TxD = 10 T = 0.1 ms. A byte's transfer rate is 1 MBps. Reciprocal of T in UART is called Baud rate. When an additional bit appends between stop bit and last bit of the byte, then T = 0.11 ms. An additional bit can be used to identify the received byte on the serial line as the address or data. It can be used for error detection. UART receives the byte at RxD (transmitter data) input line.

Serial Peripheral

Interface Serial Peripheral Interface (SPI) is one of the widely used serial synchronous communication methods. Source of serial synchronous output or input is called master when it also controls the synchronising clock information to the receiver. A receiver of serial synchronous input or output is called a slave, when along with the serial data it also receives the synchronising clock information from the master. Four sets of signals, viz., SCLK, MISO, MOSI, and SS (slave select) are used on four wires. When SS is active, then the device functions as a slave.

Serial Peripheral Interface Serial Peripheral Interface (SPI) is one of the widely used serial synchronous communication methods. Source of serial synchronous output or input is called master when it also controls the synchronising clock information to the receiver. A receiver of serial synchronous input or output is called a slave, when along with the serial data it also receives the synchronising clock information from the master. Four sets of signals, viz., SCLK, MISO, MOSI, and SS (slave select) are used on four wires. When SS is active, then the device functions as a slave.

I2C Bus

A number of device integrated circuits for sensors, actuators, flash memory and touchscreens need data exchanges in a number of processes. ICs mutually network through a common synchronous serial bus, called inter-integrated circuit (I2C). Four potential modes of operation (viz. master transmit, master receive, slave transmit and slave receive) for I2C bus device and generally most devices have a single role and use two modes only.

The I2C was originally developed at Philips Semiconductors. There are three I2C bus standards: Industrial 100 kbps I2C, 100 kbps SM I2C and 400 kbps I2C. I2C bus has two lines that carry the signals—one line is for the clock and one is for bidirectional data. I2C bus protocol has specific fields. Each field has a specific number of bits, sequences and time intervals between them.

Wired USB

Universal Serial Bus (USB) is for fast serial transmission and reception between the hosts, the embedded system and distributed serial devices; for example, like connecting a keyboard, printer or scanner. USB is a bus between the host system and a number of interconnected peripheral devices. Maximum 127 devices can connect with a host. USB standard provides a fast (up to 12 Mbps) as well as a low-speed (up to 1.5 Mbps) serial transmission and reception between the host and serial devices. Both the host and device can function in a system.

Features of a USB are:

USB data format and transfer serial signals are Non Return to Zero (NRZI) and the clock is encoded by inserting synchronous code (SYNC) field before each packet. The receiver synchronises bit recovery clock continuously. The data transfer is of four types—controlled data transfer, bulk data transfer, interrupt driven data transfer and isosynchronous transfer.

USB is a polled bus. Polling mode functions as: A host controller regularly polls the presence of a device as scheduled by the software. It sends a token packet. The token consists of fields for type, direction, USB device address and device end-point number.

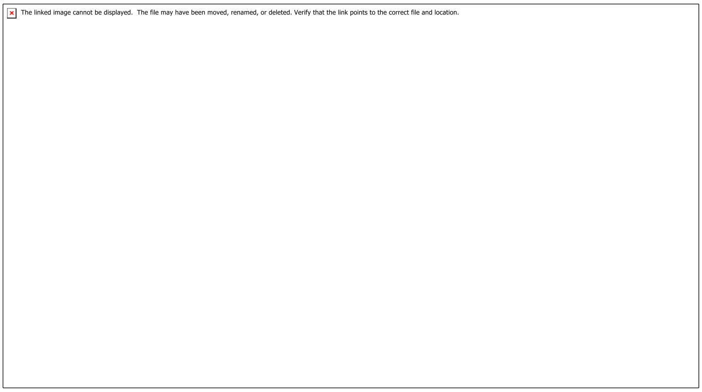
A USB supports three types of pipes—Stream with no USB-defined protocol is used when the connection is already established and the data flow starts. Default control is for providing access. Message is for control functions of the device. The host configures each pipe with the data bandwidth to be used, transfer service type and buffer sizes.

2.4 DATA ENRICHMENT, DATA LO 2.3 CONSOLIDATION AND DEVICE MANAGEMENT AT GATEWAY

A gateway at a data-adaptation layer has several functions. These are data privacy, data security, data enrichment, data consolidation, transformation and device management. IoT or M2M gateway consisting of data enrichment, consolidation and device management, and communication frameworks .

Recall ITU-T reference model. The model's lowest layer is the device layer. This layer has device and gateway capabilities. Also, recall device and gateway domainin ETSI IoT architecture. The domain consists of a gateway between M2M area network and CoRE and access network. A gateway consists of the data enrichment, consolidation and IoT communication frameworks. The communication gateway enables the devices to communicate and network with the web. The communication gateway uses message transport protocols and web communication protocols for the Internet.

The gateway includes two functions viz. data management and consolidation, and connected device management. The following subsections describe the framework for data enrichment and consolidation.


The linked image cannot be displayed. The file may have been moved, renamed, or deleted. Verify that the link points to the correct file and location.

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

2.4.1 Data Management and Consolidation Gateway

Gateway includes the provisions for one or more of the following functions: transcoding and data management. Following are data management and consolidation functions:

● Transcoding

● Privacy, security

● Integration

● Compaction and fusion

Transcoding

Transcoding means data adaptation, conversion and change of protocol, format or code using software. The gateway renders the web response and messages in formats and representations required and acceptable at an IoT device. Similarly, the IoT device requests are adapted, converted and changed into required formats acceptable at the server by the transcoding software.For example, use of transcoding enables the message request characters to be in ASCII format at the device and in Unicode at the server. It also enables the use of XML format database at the device, while the server has a DB2, Oracle or any other database.

Privacy

Data such as patient medical data, data for supplying goods in a company from and to different locations, and changes in inventories, may need privacy and protection from conscious or unconscious transfer to untrustworthy destinations using the Internet. Privacy is an aspect of data management and must be remembered while designing an application.

The design should ensure privacy by ensuring that the data at the receiving end is considered anonymous from an individual or company. Following are the components of the privacy model:

- Devices and applications identity-management
- Authentication
- Authorisation
- Trust
- Reputation

When data is transfered from one point to another, it should be ensured that the stakeholder in future may not misuse the device end data or the application data. These static and dynamic relationships are components which depend on trust and reputation.

Secure Data Access

Access to data needs to be secure. The design ensures the authentication of a request for data and authorisation for accessing a response or service. It may also include auditing of requests and accesses of the responses for accountability in future. A layer provides the confidentiality and authorisation using AES-128 and CCM. End-to-end security is another aspect while implies using a security protocol at each layer, physical, logical link and transport layers during communication at both ends in a network.

Data Gathering and Enrichment

IoT/M2M applications involve actions such as data-gathering (acquisition), validation, storage, processing, reminiscence (retention) and analysis.

Data gathering refers to data acquisition from the devices/devices network. Four modes of gathering data are:

1. Polling refers to the data sought from a device by addressing the device; for example, waste container filling information in a waste management system

2. Event-based gathering refers to the data sought from the device on an event; for example, when the device reaches near an access point or a card reaches near the card reader or an initial data exchange for the setup of peer-to-peer or master-slave connection of BT device using NFC

3. Scheduled interval refers to the data sought from a device at select intervals; for example, data for ambient light condition in Internet of streetlights

4. Continuous monitoring refers to the data sought from a device continuously; for example, data for traffic presence in a particular street ambient light condition in Internet of streetlights Data enrichment refers to adding value, security and usability of the data.

Data Dissemination

Consider the following three steps for data enrichment before the data disseminates to the network as aggregation, compaction and fusion. Aggregation refers to the process of joining together present and previously received data frames after removing redundant or duplicate data. Compaction means making information short without changing the meaning or context; for example, transmitting only the incremental data so that the information sent is short.

Fusion means formatting the information received in parts through various data frames and several types of data (or data from several sources), removing redundancy in the received data and presenting the formatted information created from the information parts. Data fusion is used in cases when the individual records are not required and/or are not retrievable later.

Energy Dissipation in Data Dissemination

Energy consumption for data dissemination is an important consideration in many devices in WPANs and in wireless sensor nodes (WSNs). This is due to limited battery life. Energy is consumed when performing computations and transmissions. Higher the data rate, the greater will be the energy consumed. Higher is RF used, the greater will be the energy consumed.

Higher the gathering interval, the lower will be the energy consumed. Energy efficient computations can be done by using concepts of data aggregation, compaction and fusion. Lesser the data bytes communication, greater the acquisition intervals, and lower the data rate for data transfer, lesser the energy dissipation.

Data Characteristics, Formats and Structures

Data characteristics can be in terms of temporal data (dependent on the time), spatial data (dependent on location), real-time data (generated continuously and acquired continuously at the same pace), real-world data (from physical world for example, traffic or streetlight, ambient condition), proprietary data (copy right data reserved for distribution to authorised enterprises) and big data (unstructured voluminous data). Data received from the devices, formats before transmission onto Internet. The format can be in XML, JSON and TLV.

2.5 EASE OF DESIGNING AND AFFORDABILITY

Design for connected devices for IoT applications, services and business processes considers the ease in designing the devices' physical, data-link, adaption and gateway layer.

It means availability of SDKs (software development kits), prototype development boards with smart sensors, actuators, controllers and IoT devices which are low in cost and hardware which embeds and are preferably open source software components and protocols. Hardware which includes the device should embed minimum number of components and use ready solutions for ease in designing local devices personal area network and secure connectivity with the Internet.

Designing also considers ease as well as affordances for example, RFID or card. The card has an embedded microcontroller, memory, OS, NFC peripheral interfaces, access point-based device activation, RF module and transceiver at low cost. A wireless sensor uses, for example, a mobile terminal (Mote) which is a low cost device with an open-source OS (tiny OS) and software components. Usages of Motes provide ease and affordance in a WSN network.

Devices of smart homes and cities use ZigBee IP or BT LE 4.2 (dual mode or single mode) due to their affordability, ease of designing, usage and low cost.

An IoT/M2M device network gateway needs connectivity to web servers. A communication gateway enables web connectivity, while IoT/M2M specific protocols and methods enable web connectivity for a connected devices network. A server enables IoT device data accumulation (storage). Application (reporting, analysis and control), collaboration, service and processes (involving peoples and business processes) use this data. Following are some key terms, which need to be understood for learning web connectivity and communication between the connected devices network and the web for IoT:

Application or App refers to a software for applications such as creating and sending an SMS, measuring and sending the measured data, receiving a message from a specific sender etc.

Application Programming Interface (API) refers to a software component, which receives messages from one end; for example, from an application or client or input. An API may consist of GUIs (button, check box, text box, dialog box). An API may get input to or from a server or a user. It then initiates actions and may send the messages, for example, to application software, server or a client at the other end etc. For example, consider an API for a parcel tracking application. The API displays the fields for the user inputs required for tracking, accepts the user inputs, displays a 'wait message' to the user and sends the input parameters to an application at a server.

The application in turn displays the response from the application or server. An API also refers to software components, which enable easier development of an application. An API defines the functionalities for a programmer, which puts the building blocks together to develop an application. A building block consists of the operations, inputs, outputs and underlying types.

Web service refers to a servicing software which uses web protocols, web objects or WebSockets; for example, weather reports service, traffic density reports, streetlights monitoring and controlling service.

Object refers to a collection of resources; for example, collection of data and methods (or functions or procedures) to operate on that data. Take for instance, Time_Date object with second, minutes, hour, day, month and year fields and update methods. An object instance can be just one or more than one for an object.

An example of an object instance is birth_date. Multiple object instances, abc_birth_date, pqr_birth_date, xyz_birth_date and many instances can be created from birth_date object in JavaScript. An example of object instance is weather report object for reporting the rains.

Object model is defined as the usage of objects for values, messages, data or resource transfer, and creation of one or more object-instances. Class: Java uses concept of class, which creates one or more object instances. Communication gateway is one that functions as communication protocol translator (convertor) for provisioning communication capabilities. For example, the gateway for communication between ZigBee and IP networks. Client refers to a software object which makes request (or an API associated with it makes request) for data, messages, resources or objects. A client can have one or more object instances. A client may also have an API or many APIs for enabling the communication to the server. A client can be at a device or application on a network or Internet connected web, enterprise server or cloud. Server is defined as a software which sends a response on a request. The server also sends messages, alerts or notifications. The server has access to resources, databases and objects. A server can be on a device or can be on a separate computer system, not necessarily on Internet connected web. Web object is the one that retrieves a resource from the web object at other end using a web protocol. Broker denotes an object, which arranges the communication between two ends; for example, between the message publisher and subscriber or for example taking the request from a source and sending the response received back for that source after arranging the response from another object, such as a server. Proxy refers to an application which receives a response from the server for usage of a client or application and which also receives requests from the client for the responses retrieved or saved at proxy. Communication protocol defines the rules and conventions for communication between networked devices and between systems. The protocol includes mechanisms for devices or systems to identify and make connections with each other. The protocol also includes formatting rules that specify how data is packaged into sent and received messages and header, its field and their meanings. Web protocol is a protocol that defines the rules and conventions for communication between the web server and web clients. It is a protocol for web connectivity of web objects, clients, servers and intermediate servers or firewalls. It includes mechanisms for a web object to identify and make connections with other objects. The protocol also includes web object formatting rules that specify how that object packs into it the sent and received messages. Firewall is one that protects the server from unauthentic resources. A header consists of a set of words. The words contain the information and parameters about the processing at a communicating layer. The words are placed over a data stack received from a previous layer and create a new data stack for transmission to next lower layer. Header words are placed at the sending layer lower in the hierarchy during transmission and removed at the receiving layer higher in hierarchy after using the information contained in the header.

A state refers to an aspect related to someone or something, or a form at a particular time (Collins English Language Dictionary). State, in reference to data interchange between web objects, refers to an aspect related to data or resource or object received at a particular instance at the server or client end. Resource denotes something that can be read (used), written (created or changed) or executed (processed). A path specification is also a resource. The resource is atomic (not-further divisible) information, which is usable during computations. A resource may have multiple instances or just a single instance. Single instance example is a contact list. Examples of multiple instances of a contact are name, surname, city, address and email ID. Each resource instance has a resource ID. A resource is accessible using a resource identifier. A resource identifier can be a path specification, such as a Universal Resource Locator (URL) or Universal Resource Identifier (URI).

A resource may have resource registry, which means it is usable after a resource instance is registered at the registry, and unusable when not-registered or when de-registered. A resource can be updated. A resource can be used when an application uses resource discovery.

A resource may have a resource repository. Repository refers to subfolder, folder or directory, which contains the resources and their instances. A resource can have a resource directory, which directs to resource repository. Resource structure or resource directory refers to a structure or collection of resources, applications,

containers and groups, which may each have an attribute, access right, subscription and discovery.
Path denotes a navigation path between two ends when accessing a resource. Path specification can be of URI or URL type. The structure of URI is hierarchical. Entity after the sign '/' is a child of the parent before the sign '/'. Universal Resource Identifier is generally used for saved resources, such as contacts or address book. Example of a URI is /Contacts/First_Character_R/ for a set of resource directory contacts having resource repository First_Character_R for contacts with first character R and resources giving information about a contact. Another example is URI sensorNetwork_J/sensorID_N/temperature for a temperature value. The value is at a resource directory sensorNetwork_J for a sensor network, which stores sensor data for a sensor of id sensorID_N. Universal Resource Locator is generally used for retrieving a resource(s) by a client. The saved resources may be at a document or at a remote server accessed using Internet protocols. An example of a URL is http://www.mhhe.com/ for a set of resource directories, resource repositories and resources on the McGraw Hill Higher Education server.Datagram refers to a limited size data (216 byte). It is used for stateless connectionless transfer from a web object. Stateless means each single datagram transfer is independent of previous data interchanges and connectionless means there is no need of pre-establishing the connection for resource exchanges between the web objects and no connection closing after finishing the resource exchanges. Representational State Transfer (REST) is a software architecture referring to ways of defining the identifiers for the resources, methods, access methods and data transfer during interactions. REST is a software architecture which also specifies the practices, constraints, characteristics and guidelines for creating scalable web services. Scalable means can be used as per the size. The architecture is used during the design of web software components, clients and web APIs . REST also refers to usage of defined resource types when transferring the objects between two ends—URIs or URLs for representations of the resources. REST also refers to the usage of use verbs (commands), POST, GET, PUT and DELETE and files ofMIME types.
Multipurpose Internet Mail Extensions or MIME refers to the type of files that are widely used on the Internet by web objects, applications and services. RESTful

refers to one which follows REST constraints and characteristics. User Datagram Protocol (UDP) is a protocol at the transport layer for the web using Internet and Constrained RESTful Environment (CoRE). UDP specifies the way of enveloping the datagram by header words, which just specifies the ports and addresses of two ends between which the datagram transfers.

Hypertext means text embedded with hyperlinks. HyperText Transfer Protocol (HTTP) means an application layer protocol for use of hypertext as app data transfer protocol.Clients and servers use the URLs http://... . Hyperlink refers to a specification of the URL for a resource path, so that a link can be established between two objects. For example, hyperlink URL specifications for the author’s earlier book’s supplementary web-resources at McGraw-Hill Higher Education server are http://www.mhhe.com/kamal/emb3. Retrieval of a resource at a web object by the other object uses the click at a link shown on a displayed text on browser. HyperText Markup Language (HTML) is a language for creating a hypertext which refers to text that embeds text, images, audio and video, image frames, forms, lists, tables, navigation links (reference to resources), APIs, Java Script and other codes for dynamic actions. Extensible Markup Language (XML) is a language, which enables creation, sending and receiving documents, messages, commands, query responses, queries, and creation of forms. The form data, response or message uses a new tag (not defined earlier in HTML) with new data type definitions than the ones at HTML.

3.2 WEB COMMUNICATION PROTOCOLS FOR CONNECTED DEVICES

Data of connected devices routes over the web in two types of communication environments. The environments are:● Constrained RESTful Environment (CoRE): IoT devices or M2M devices communicate between themselves in a Local Area Network. A device typically sends or receives 10s of bytes. The data gathered after enriching and consolidating from a number of devices consists of 100s of bytes. A gateway in the communication framework enables the data of networked devices that communicate over the Internet using the REST software architecture.
❍ Devices have the constraint in the sense that their data is limited in size compared to when data interchange between web clients and web servers takes

place using HTTP, Transmission Control Protocol (TCP) and Internet Protocol (IP).

❍ Another constraint is data-routing when Routing Over a network of Low power and (data) Loss (ROLL). ROLL network is a wireless network with low power transceiver. Another constraint is that the devices may sleep most of the time in low power environment and awaken on an event or when required (on client initiative). Devices' connectivity may also break for long periods, have limited up intervals in loss environment, and have limited data size.

Unconstrained Environment: Web applications use HTTP and RESTful HTTP for web client and web server communication. A web object consists of 1000s of bytes. Data routes over IP networks for the Internet. Web applications and services use the IP and TCP protocols for Internet network and transport layers (Section 4.3). Figure 3.1 shows device's local area network connectivity, web connectivity in constrained and unconstrained RESTful environments and the protocols for communication.

The linked image cannot be displayed. The file may have been moved, renamed, or deleted. Verify that the link points to the correct file and location.

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

Assume that a web object refers to a web client or web server. The web object communicates a request of the client or a response of the server. Communication is over the ROLL network or Internet.

Communication between web objects (right-hand side)

● IETF CoRE specifications, which include CoAP and UDP

● Web objects' protocols for sending a request or response; for example, RESTful CoAP, CoAP client and CoAP server communication over the network and transport layers to other end CoAP client and server. Client/server use the URIs coap://… in place of http://… .

● Transport layer protocols used are Datagram Transport Layer Security (DTLS) and UDP. Data between web objects route using ROLL network specifications of IETF.

● 100s of bytes communicate between the IoT web objects

● Web objects HTTP client and HTTP server communication over the Internet using the IP and client/server use the URLs http://… .

● 1000s of bytes communicate between HTTP web objects using certain protocols for sending a request or response; for example, RESTful HTTP. IPv6 or IP is the network layer protocol used. The transport layer protocols used are TLS and TCP. IoT device or machine applications need constrained environment protocols such as CoAP and LWM2M.

3.2.1 Constrained Application Protocol

IETF recommends Constrained Application Protocol (CoAP) which is for CoRE using ROLL data network. Features of CoAP are:

● An IETF defined application-support layer protocol

● CoAP web-objects communicate using request/response interaction model.

● A specialised web-transfer protocol which is used for CoRE using ROLL network.

● It uses object-model for the resources and each object can have single or multiple instances.

● Each resource can have single or multiple instances.

● An object or resource use CoAP, DTLS (security binding with PSK, RPK and Certificate) and UDP protocols for sending a request or response.

● Supports the resource directory and resource-discovery functions

● The resource identifiers use the URIs as follow coap://… .

● Has small message header, 4 bytes are for ver (version), T (message type) [Types are confirmable, non-confirmable, acknowledgement and reset], TKL (token length), code (request method or response code), message ID 16-bit identifier, token (optional response matching token). Confirmable means server must send an acknowledgement, while non-confirmable means no acknowledgement is required.

● CoRE communication is asynchronous communication over the ROLL.

● Integrates easily with the web using CoAP application cross-protocol proxies. This is facilitated because HTTP and CoAP, both share the REST model. A web client may not even be aware that it just accessed an IoT device sensor resource or sent data to an IoT device actuator resource.

● Use of REST: The access by a CoAP object or its resource is thus using:

the URI

❍ a subset of MIME types

❍ a subset of response codes which are used for an HTTP object or resource.

CoAP Client Web Connectivity

A proxy is an intermediate server, which accepts a request from a client and sends the response to the client using a protocol. It also passes the request to the server and accepts a response from the server using the same or an other protocol. HTTP-CoAP proxy accepts requests from HTTP client using HTTP protocol and sends the request to the server using CoAP protocol. CoAP-HTTP proxy accepts requests from CoAP client using CoAP protocol and sends the request to the server using HTTP protocol.protocol used for securing the TCP-based Internet data interchanges. DTLS is the TLS for datagram. The features of DTLS are:

● DTLS provisions for three types of security services—integrity, authentication and confidentiality.

● DTLS protocol derives from TLS protocol and binds UDP for secured datagram transport.

● DTLS is well suited for securing applications; for example, tunnelling applications (VPN), applications that tend to run out of file descriptors or socket buffers or applications which are delay sensitive (and thus use UDP).

● A part of DTLS is OpenSSL repository openssl-0.9.8 security based on PSK, RPK and certificate.

3.2.2 Lightweight Machine-to-Machine Communication Protocol

Lightweight Machine-to-Machine Communication (LWM2M) protocol is an application layer protocol specified by Open Mobile Alliance (OMA) for transfer of service data/messages. It finds applications in M2M. It enables functionalities for device management in cellular or sensor networks. Communication protocol 'light weight' means that it does not depend on call to the system resources during execution. Example of call to a system resource is calling an API for display menu or network function in the system software (OS). Lightweight presently means data transfer formats between client and server are binary and has Tag Length Value (TLV) or Java Script Object Notation (JSON) batches of object arrays or resource arrays and transfers up to 100s of bytes unlike the webpages of 1000s of bytes.

The protocol enables communication between LWM2M client at IoT device and an LWM2M server at the M2M application and service capability layer. The protocol is a compact one, meaning small header. It has an efficient data model. It is generally used in conjunction with CoAP.

● Local M2M constrained devices use, for example, the Bluetooth low energy, 6LowPAN (IPv6 over low power WPAN), CoRE, ROLL, NFC, ZigBee PAN, cellular, Wi-Fi or ZigBee IP (left-hand side) network technologies. M2M area network functions as PAN for connectivity between devices and the M2M gateway. ● 10s of bytes communicate between a device and the PAN.

● Communication between LWM2M objects (right-hand side). LWM2M client refers to object instances as per the OMA standard LWM2M protocol. A client object sends a request or receives a response of the LWM2M server over the access and CoRE networks.

● CoRE network, for example, 3GPP or other networks for the IP connectivity

● Communication from an object instance using interface functions. The functions are bootstrapping; registration, deregister or update a client and its objects; reporting the notifications with new resource values; and service and management access through the server.

● Use of the CoAP, DTLS, and UDP protocols by the object or resource.

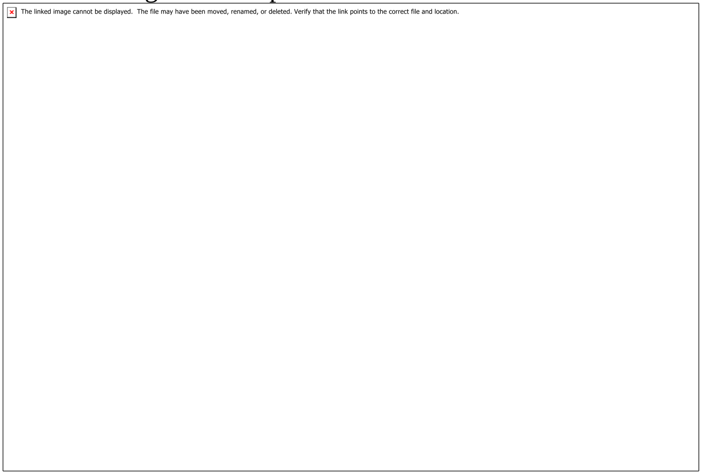
● 100s of bytes communicate between objects at the client or server for plain text or JSON or binary TLV format data transfer. LWM2M specifications and features are as follows:

● An object or resource use CoAP, DTLS, and UDP or SMS protocols for sending a request or response

Use of plain text for a resource or use of JSON during a single data transfer or binary TLV format data transfer for a package for a batch of resource representations in a single data transfer.

● An object or its resource access using an URI.

● Interface functions are for—bootstrapping; registration, deregister or updating a client and its objects; reporting the notifications with new resource values; and service and management access through the server. ● Use of object model for resources and each object can have single or multiple instances. Each resource can have single or multiple instances.



The linked image cannot be displayed. The file may have been moved, renamed, or deleted. Verify that the link points to the correct file and location.

M2M devices local area network connectivity, and constrained devices network connectivity with M2M applications and services using LWM2M OMA standard specifications of LWM2M

● OMA or other standard specifying organisation defines the LWM2M objects for usages in M2M communication. An LWM2M client has object instances as per OMA standards. LWM2M client-server interactions are over the access and

CoRE networks which generally use CoAP over ROLL. LWM2M device client and server, generally use CoAP client server interactions for data interchanges. Wi-Fi and WiMax are examples of access networks. Examples of CoRE network are GSM, GPRS, 3GPP and 4G LTE or other networks for IP connectivity as well as roaming

3.2.3 JSON Format

Features of a JSON are:

● JSON is an open-standard format used primarily to transmit data between a server and web application, as an alternative to XML. The text is human readable. It transmits data objects as text to transmit. It consists of attribute–value pairs.

● Originally derived from the JavaScript scripting language, JSON, is now a languageindependent data format, the coding of which can be in Java or C or another programming language used for parsing and generating JSON data. Example 3.2 describes codes used for data transfer in JSON.

3.2.4 Tag Length Value Format

In a TLV format the first two bytes are used to identify the parameter. The first and fourth bytes indicate the length of the actual data which follows directly after these bytes.3 Following is an example of codes used for data transfer in TLV format.

3.2.5 MIME Type

Generic-type files are application/octet-stream; however, MIME-type files are used in web applications and services.4 Features of MIME are:

● An Internet standard which is for description of the contents of various files.

● An Internet standard which extends the Simple Mail Transfer Protocol (SMTP) format of email to support the text in character set besides ASCII, and the non-text attachments such as audio, video, images and application programs.

● Supports messages with multiple parts and header information in non-ASCII character sets

Initially designed for mail, now it is used as Internet media type. Web client header specifies the content types using a MIME type.

● List of MIME type files is exhaustive. Few examples of MIME type files are—doc (application/msword), html (text/html), htm (text/htm), gif (image/gif), js (application/ x-javascript), css (Application is text/css), mpeg (video/mpeg), pdf (application/pdf) and exe (application/octet-stream). HTML links, objects, and script and style tags, each has a type attribute the value of which can be set equal to a MIME type. The MIME type for HTML is text/html.

3.3 MESSAGE COMMUNICATION PROTOCOLS FOR CONNECTED DEVICES

A device/node/end-point/client/server sends and receives message(s). A communication module includes a protocol handler, message queue and message cache. The protocol handler functions during transmission and reception of messages according to the communication protocols for these actions. Message queue forms to keep a message until it is transmitted towards its destination. Message cache keeps an incoming message until it is saved into the module. A device message-queue inserts (writes) the message into the queue and deletes (reads) the message from the queue. Device message-cache stores the received messages. The following section explains the terms used by messagecommunication protocols.

3.3.1 Terminology

Request/Response (Client/Server)

A request/response message exchange refers to an object (client) requesting for resource(s) and an object (server) sending the response(s). The objects, for example, HTTP objects may use the REST functions. When sending a request, a protocol adds the header words. Each header has fields. Each field interprets the request at the receiver object. Publish/Subscribe (pubsub) Publish/subscribe or PubSub message exchanges differ from request/response.

A service publishes the messages; for example, a weather information service publishes the messages of weather reports for the potential receivers. A group controller publishes the messages; for example, measured values of ambient light conditions, traffic density or traffic presence . A service can be availed by one or more clients or brokers. When a client subscribes to the service, it receives messages from that service. A publish/subscribe messaging protocol provisions for publication of messages and reception on subscription (PUT and GET methods) by the registered or authenticated devices. Publication may be for measured values, for state information or resources of one or more types. Subscription is for a resource-type (or for a topic).

A separate subscription is required for each resource-type or topic. An example of resource type is measured values of ambient light condition in the smart streetlights example. Another resource type is traffic presence or absence on the street. Another resource type is lighting function report; functioning is proper or fault exists in the light.

Resource Directory

Resource directory (RD) maintains information and values for each resource type. A resource of a resource type accesses from the RD using URI for the resource.

Resource Discovery

Resource discovery service may advertise (publish) at regular intervals, the availability of the resources or types of the resources available and their states. A client discovers the resource type and registers for the RD service.

Registration/Registration Update

Registration means a receiver registers with a service, such as an a RD service. When one or more endpoints or devices or nodes registers, then that device gets the access to the resources and receives published messages. Security considerations may require authentication of both ends (service provider and receiver) before registration. A separate registration is required for each endpoint (client or server). Registration updates means updating for one or more endpoints or devices or nodes. It also includes unregistering for one or more endpoints.

Pull (Subscribe/Notify) Data

Pull means pulling a resource, value, message or data of a resource-type by registering and subscribing. Pull may be using GET or on initiating OBSERVE. The server maintains state information for a resource and notifies on change of state. Client pulls again the resource on the change.

Polling or Observing

Polling means finding from where new messages would be available or whether new messages are available or updates are available or whether the information needs to be refreshed or finding the status if the state information has changed or not. When messages store at a database-server, then polling can be done by a client who uses the REST architecture GET method and server uses the POST. A state may refer to a connection or disconnection, sleep, awake, created, alive (not deleted), old values persisting or updated with new values (GET + OBSERVE). Observing means looking for change, if any, of a state at periodic intervals (OBSERVE).

Push (Publish/Subscribe)

Data Push means a service that pushes the messages or information regularly. Interested device or endpoint or potential receiver receives these pushes. For example, a mobile service provider pushes the temperature and location information regularly for the potential receivers (registered mobile services subscribers) (PUT). Push is efficient compared to polling, particularly when notifying or sending alerts. This is because there can be many instances when polling returns no data. Pull is efficient compared to polling or PUSH when certainty exists that within a reasonable time interval the server returns no data.

Message Cache

Cache means storing when available and can be used later on when required. Messages cache is useful in an environment of short or prolonged disconnections of a service. A message can be accessed once or more times from a cache.

Message Queue

Message queuing means storing the messages (data) in sequence from devices or endpoints so that when connection state changes then messages can be forwarded. Forwarding is done using the first-in first-out method for a resource type. A message forwards only once from a queue. Separate queues are formed for each resource type. The messages are forwarded to the registered devices or endpoints and to the subscribed devices or endpoints. A separate registered device or endpoint list and a separate subscription list is maintained and used for each resource type. Forwarding takes place only after matching the subscription from a list.

Information/Query

The method is that an object (client) requests information using a query while another end-object (server) responds by replying to the query. The responding application processes the query using the query optimiser and retrieval plan. The query processing uses a database or resource directory resources

3.3.2 Communication Protocols

Following are the protocols used in message communication.

CoAP-SMS and CoAP-MQ M2M or IoT device uses SMS quite frequently. SMS is identified as the transport protocol for transmission of small data (up to 160 characters). It is used for communicating with a GSM/GPRS mobile device. M2M or IoT device uses message queuing quite frequently due to ROLL environment and constrained devices (awake only when initiated) or connection-breaks for long periods. CoAP-SMS and CoAP-MQ are two protocols drafted and recommended by IETF. CoAP-SMS CoAP-SMS is a protocol when CoAP object uses IP as well as cellular networks and uses SMS. It is an alternative to UDP-DTLS over ROLL for CoAP object messages and when using cellular communication. SMS is used instead of UDP + DTLS by a CoAP client or server. A CoAP client communicates to a mobile terminal (MT) endpoint over the General Packet Radio Service (GPRS), High Speed Packet Access (HSPA) or Long Term Evolution (LTE) networks using CoAP-SMS protocol.

Following is the IETF recommended terminology for use:

● SMS-C: SMS service centre

● SMS-SP SMS service provider

● CIMD: Computer interface to message distribution

● MS: Mobile station at a cellular network functioning as CoAP client or CoAP sever

● MO: Machine or IoT device or mobile origin functioning as CoAP client

● MT: Machine or IoT device or mobile terminal functioning as CoAP sever

● SMPP: Short message peer-to peer for CoAP-Data. Peer-to-peer means sending requests as well as receiving requests

● SS7: Signalling Service Protocol

● UCP/UMI: Universal computer interface protocol/machine interface for short message delivery contained in a CoAP request or CoAP response

The CoAP-SMS features are as follows:

● An URI used as coap+sms:// in place of coap://. For example, URI may be coap+sms:// telNum/carLocatiobObject/latitude for latitude of location of a car after LocationObject measures location parameters using the GPS. URI is used when sending the SMS to the specified telephone number—telNum. ● A CoAP message encodes with alphabets for SMS communication. An SMS message consists of 160 characters in 7-bit encoding of a character. Maximum length for a

CoAP message is thus 140 B (= 160 × 7 bits/8 bits) when an SMS-C supports 8-bit encoding. A CoAP message is thus 70 (= 160 × 7 bits/16 bits) when an SMS-C supports 16-bit encoding and multilingual alphabets. Concatenated short messages can be up to 255B, provided the SMS-C supports this.

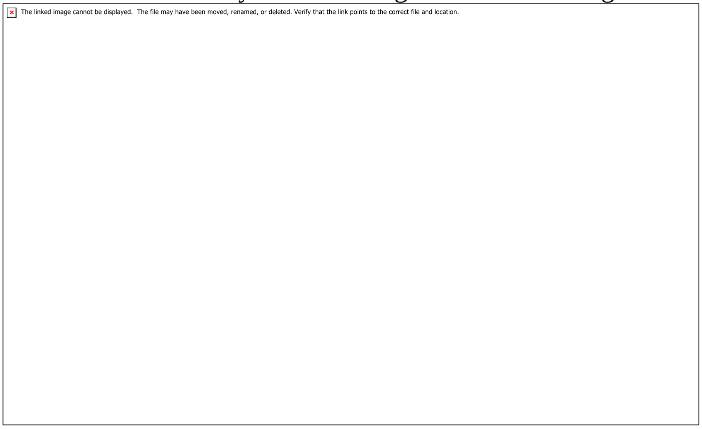
● CoAP endpoints have to work with a Subscriber Identity Module (SIM) card for SMS in cellular networks. The points are addressed by using Mobile Station ISDN (MSISDN) number. A TP-DATA-Coding-Scheme inclusion enables the CoAP client to find that the short message contains a CoAP message.

● Does not support multi-casting

● Two additional options are Response-to-URI-Host (RUH) and Response-to-URI-Port (RUP) which make initiating CoAP client aware of the presence of the alternative interface CIMD and SMPP and UCP/UMI. RUH is string of size 0 to 255B. RUP is of 2B with default port number 5683. IANA (Internet Assigned Number Authority) registered TBD as CoAP Option Numbers for registry.

● Data interchange sequences are as follows: An MS/CoAP client sends a SMS request (SMS-SUBMIT) to SMS-C; SMS-C reports using SMS-SUBMIT-REPORT; SMS-C sends SMS (SMS-DELIVER) to MS/CoAP server; the server reports using SMS-DELIVERREPORT; and SMS-C sends SMS-STATUS-REPORT to the client.

● Authentication of a client by the server provides the security. MSISDN of the MS and SIM based security is used during SMS data exchanges.


The linked image cannot be displayed. The file may have been moved, renamed, or deleted. Verify that the link points to the correct file and location.

(a) A CoAP request or response communication to a machine, IoT device or mobile terminal (MT), (b) A computer or machine interface using IP communication to a mobile service provider for data interchange with terminal, (c) A machine or IoT device or mobile origin (MO) communication of CoAP request or response communication, and (d) An origin communication using SS7/CIMD/SMPP with a computer or machine interface using IP communication

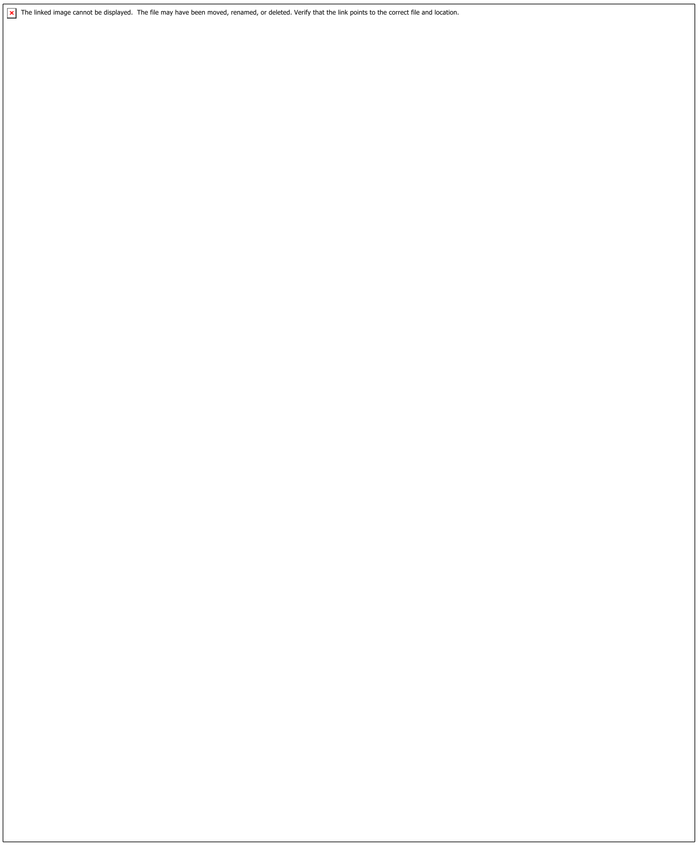
SMPP with machine or IoT device or MT with an in-between node SMS-C. The terminal sends response to the origin using SMS-C and CoAP-MQ Broker. Figure 3.4(c) shows a CoAP request or response communication.

MQTT Protocol

Message Queuing Telemetry Transport (MQTT) is an open-source protocol for machine-tomachine (M2M)/IoT connectivity. Word 'telemetry', in English dictionary, means measuring and sending values or messages to far off places by radio or other mechanism.

IBM first created it and then donated it to M2M 'Paho' project of Eclipse. A version is MQTT v3.1.1. MQTT has been accepted (2014) as OASIS (Organization for the Advancement of Structured Information Standards) standard6 MQTT protocol is used for connectivity in M2M/IoT communication. A version is MQTT-SN v1.2. Sensor networks and non-TCP/IP networks, such as ZigBee can use the MQTT-SN.

MQTT-SN is also a publish/subscribe messaging protocol. It enables extension of the MQTT protocol for WSNs, the sensor and actuator devices and their networks.


The linked image cannot be displayed. The file may have been moved, renamed, or deleted. Verify that the link points to the correct file and location.

Messages interchange between M2M/IoT device objects (publisher and subscriber) and web objects (publisher and subscriber) using an MQTT Broker

The objects communicate using the connected devices network protocols such as ZigBee. Web objects also use MQTT library functions and communicate using IP network and SSL and TLS security protocols for subscribing and publishing web APIs. MQTT Broker (also referred simply as MQ) does the following:

- Functions as a server node capable of storing messages from publishers and forwarding them to the subscribing clients.

● Receives topics from the publishers. Examples of topics are measured information of ambient light conditions, traffic density, nearby parking space availability and waste container status.

● Performs a store-and-forward function, stores topics from publishers and forwards to subscribers.

● Receives subscriptions from clients on the topics, matches subscriptions and publications in order to route messages to right endpoints.

● Recovers subscriptions on reconnect after a disconnection, unless the client explicitly disconnected.

● Acts as a broker between the publisher of the topics and their subscribers.

● Finds client disconnection until DISCONNET message receives, keeps message alive till explicit disconnection.

● Retains the last-received message from a publisher for a new connected subscriber on the same topic, when retain field in the header is set.

● Authentication by Username/Password in connect message and client security is through SSL/TLS. Security considerations are same as of CoAP, web-linking and CoRE resource directory.

● Support from Intelligent and business analyst server and other servers through a MQTT server with a gateway.

3.3.3 XMPP (Extensible Messaging and Presence Protocol) XML

XML is an open-source IETF recommended language. XML is widely used for encoding messages and texts. A text element in an XML document can correspond to the data, message, alert, notification object, command, method or value of an entity. There is a way to tag in which text is featured between the tag and its associated end tag. An XML tag specifies the type of encoded entity in an element. A tag may also associate an attribute(s) which is specified there itself. The interpretation and use of the text within a tag pair (a tag and its associated end tag) depends on the parser and associate application. The Parser uses XML file as input. The application uses the output from the parser. The parser and application can be in Java, C# or any other programming language. Following is an example of usage of XML tags and attributes in XML elements in a document.

XMPP

XMPP is an XML-based specification for messaging and presence protocols. XMPP is also an open-source protocol recommended specification which is accepted by IETF. RFC is an international organisation and stands for ’Recommended for Comments’. RFC 6120 document specifies the XMPP for CoRE. Another recommendation, RFC 6121 XMPP specifies the instant messaging (IM) and presence, and RFC 6122 XMPP specifies the (message) address format.

Messages notify presence for the IMs to one or many at the same time. It enables chatting and Multi-User Chat (MUC) after creation of a chat room, where different users can do the IMs. XMPP enables interoperable communication: for example, Google Talk.

XMPP enables IMs between many users as it uses presence-notifications and chat features. Chat room is an application, in which all those who have subscribed (meaning persons and objects initiating chatting and messaging to one another at the same time) are provided a room-like view and use the IMs among themselves. XMPP is extensible—XSF (XMPP standards foundation) develops and publishes the xeps (XMPP extension protocols). The xeps enable the addition of features and new applications. List of XMPP xeps for web objects is quite long. Examples of xeps are:

● xep-DataForms Format ● xep-XHTML-IM ● xep-Service Discovery ● xep-MUC ● xep-Publsih-Subscribe and Personal Eventing Protocol ● xep- File Transfer ● xep-Jingle for Voice and video

Simple Authentication and Security Layer (SASL) and TLS are security protocols for APIs and web object messages using TCP/IP network. The XML streams in XMPP format communicate between the devices, devices to web objects and between the web objects. Features of XMPP are:

● XMPP uses XML.

● XML elements are sent in the open-ended stream within the tag and corresponding end tag .

● Three basic types of XMPP stanzas (elements) are:

❍ message ❍ presence ❍ iq (information/query, request/response)

● Extensibility to constrained environment messaging and presence protocols as well as IP network messaging.

● Extensibility of request-response (client-server) architecture to iq (information through querying), PubSub messaging, Chat room MUC messaging and other architecture (where group of people exchange information when present in a chat room), decentralised XMPP server.

The linked image cannot be displayed. The file may have been moved, renamed, or deleted. Verify that the link points to the correct file and location.

Use of XMPP and XMPP extension protocols for connected devices and web objects for the messaging, presence notifications, responses on demand and service discoveries using XML streams

XMPP server set by anyone on the following standards recommended and using XSF xeps; for example, XMPP-IoT server, XMPP M2M server for messaging between the machines.

● Authentication by SASL/TLS, and support from intelligent and business analyst applications, and processing through XMPP server and gateway for connecting device network with IP network. XMPP does the following:

● Binary data is first encoded using base 64 and then transmitted in-band. Therefore, the file first transmits out-of-band between nodes on messages from XMPP server but not directly like IMs.

● No end-to-end encryption ● Higher overhead being text based in place of binary implementations

● No support for QoS like MQTT does.

3.4 WEB CONNECTIVITY FOR CONNECTEDDEVICES NETWORK USING GATEWAY, SOAP, REST, HTTP RESTFUL AND WEBSOCKETS

The following subsections describe the uses of gateway, SOAP, REST, RESTful HTTP and WebSockets for connecting web objects.

3.4.1 Communication Gateway

Communication gateway connects two application layers, one at sender and the other at receiver. The gateway also enables use of two different protocols, one at sender and the other at receiver ends. The gateway facilitates the communication between web server using the TCP/IP protocol conversion gateway and IoT devices. It also facilitates communication between the devices using CoAP client and server using HTTP. The gateway provisions for one or more of the following functions:

● Connects the sender and receiver ends using two different protocols. For example, IoT devices network maybe ZigBee network for connecting the devices. The network then connects to the web server through a gateway. The server posts and gets the data using HTTP. A gateway facilitates the communication between IoT devices and web server. For example, (i) ZigBee to SOAP and IP or (ii) CoAP protocol conversion gateway for RESTful HTTP

Functions as proxy between the system and server.

3.4.2 HTTP Request and Response Method

An application uses a protocol. The application layer in TCP/IP suite of protocols for Internet uses HTTP, FTP, SMTP, POP3, TELNET and a number of other protocols. HTTP is most widely used application layer protocol for communication over TCP/IP. An HTTP client connects to an HTTP server using TCP and then the client sends a resource after establishing an HTTP connection.

Data Exchanges Between HTTP Web-Objects

An HTTP connection enables a one-way communication at an instance from client API to a server or from server to the API. HTTP polling is a method for receiving new messages or updates from an HTTP server. Polling means finding whether new messages or updates are available and receiving them in case they are.

An HTTP transfer is stateless which means each data transfer is an independent request. Therefore, header overhead information and meta-data of previous state need to be present with each HTTP request. Metadata is data which describes the data for interpretation in future. Therefore, each data interchange needs large headers over 100s of byte, and thus greater latency in request-response exchanges. Ways of transferring both ways at the same instant are:

● Multiple TCP connection

● HTTP requests at short, regular intervals so that responses are nearly in real time

● Polling at successive intervals

● HTTP long polling means API sends a request to the server, which keeps the request open for a set period ● Stream hidden in iFrame Polling methods have high latencies and header sizes of 100s of bytes. An alternative to this is using Java Applets, Silverlight or Flash plugins.

3.4.3 SOAP

Applications need to exchange objects on the Internet using protocols such as HTTP. Applications may be using different languages and platforms. Simple Object Access (SOAP) is a W3C approved open-source protocol. SOAP is a protocol for exchange of objects between applications using XML. It is also a protocol for access to a web service. SOAP specifies the formats and way of communicating the messages. Its usage is independent of the application language and platform (OS and hardware). It is extensible and is also used for APIs for the web services and Service-Oriented Architecture (SOA).

SOAP enables development of applications and APIs. SOAP functions connect the GUI applications to web servers using the standards of the Internet—HTTP and XML. Microsoft's .NET architecture supports SOAP for Internet application development.

W3C recommended SOAP v1.2 second edition specifications—Part 0 is a primer, part 1 is a messaging framework and Part 2 is adjuncts. SOAP v1.2 specifications provision for assertions and test collection, XML-binary Optimized

Packaging, SOAP message transmission optimisation mechanism and resource representation SOAP header block.

A SOAP request could be an HTTP POST or an HTTP GET request. The HTTP POST request specifies at least two HTTP headers: content type and content length. A SOAP method uses HTTP request/response after the HTTP binding with the SOAP.

3.4.4 REST and RESTful HTTP Web Applications

W3C Technical Architecture Group (TAG) developed the Representational State Transfer (REST) architectural style.The group worked in parallel with HTTP 1.1. REST is a coordinated set of constraints which are used during design of software components in a distributed hypermedia, and the design depends on the characteristics of stateless, client-server, cacheable communication using a protocol. World Wide Web uses REST practices and constraints. REST is a simpler alternative for SOAP and Web Services Description Language (WSDL). REST style web resources and Resource-Oriented Architecture (ROA) have increasingly replacing SOAP.

The linked image cannot be displayed. The file may have been moved, renamed, or deleted. Verify that the link points to the correct file and location.

(a) A SOAP request or response communication between gateway weatherMsgG and weather service weatherMsgService, (b) SOAP message structure, (c) SOAP POST from weatherMsgG, (d) Request Get weatherMsgG, and (e) SOAP response weatherMsgID250715 at weatherMsgService to weatherMsgG using Internet and HTTP binding

The architectural properties of REST are realised by applying specific interaction constraints to data elements, components, connectors and objects. An architectural property of REST is separation of concern which means data storage at server is not a concern for the client, and client components can port on other objects.The resources themselves are conceptually separate from the representations that are returned to the client. For example, the server may send data HTML, XML, TLV or JSON from its database which can have other distinct internal representation at a connected database server.

REST software architecture style provisions for the use of specific practices, such as client-server mode of communication and layered system architectural approach. Client-server interactions have characteristics of performance and creation of scalable web objects and services. Scalability means ability to support greater number of interactions among components and greater number of components. Layered system refers to a client which can connect through an intermediate layer (proxy, firewall, gateway, proxy or intermediary server). REST enables intermediate layer processing by constraining the messages such that the interactions at each layer are selfdescriptive. A client may not ordinarily be made aware whether it is connected directly to an end-server. Intermediate layers may render assistance in transcoding and enabling usages of different protocols at two ends.

Use of the intermediate system improves performance when using bigger scales and shared caches. Shared caches means caches shared between two layers; for example intermediate layer and server or proxy and server. Representation of each application state contains links that may be used for a next-time interaction when the client selects those links and initiates a new state transition.

RESTful

When all interactions used in the applications conform fully to the REST constraints then these are called RESTful. RESTful APIs comply with these constraints and thus conform to the REST architectural style. Web services with RESTful APIs adhere to the REST architectural constraints. REST architectural style can be used for HTTP access by GET, POST, PUT and DELETE methods for resources and building web services.

3.4.5 WebSocket

RFC 6455 describes the specifications of the web protocol12. WebSocket is an IETF accepted protocol. WebSocket API (WSAPI) W3C standard is Web Interface Definition Language (Web IDL)13. Instant messaging and many applications need bidirectional data exchanges over the same connection. WebSocket enables bidirectional communication over a single TCP connection.

The WebSocket-protocol-based design usage now supersedes the existing bidirectional communication technologies that use HTTP at application layer. WSAPI attributes are url, protocol, readystate, buffered amount and are extendable. The API functions are send (frame) and close (socket). A frame contains header plus data. WebSocket frame contains 2 B header extended to 0, 2 or 4 B plus data bytes. Client side adds 4 B mark; for example, for identifying a frame.

The WebSockets enable easier usage of existing infrastructure, authentication, filtering and proxies. Number of popular browsers support the protocol. HTTP case polling at periodic intervals are not required due to no-wait feature of WebSocket.

The linked image cannot be displayed. The file may have been moved, renamed, or deleted. Verify that the link points to the correct file and location.

(a) Opcode and other fields at a WebSocket frame, (b) WebSocket API events, attributes and functions, and (c) Data bidirectional communication using WebSocket between web objects

Features of WebSocket are:

● Small header size (2B extended to 6 byte and above compared to 100s B and above for an HTTP header which results in high latency

● No new connection which will need a new header and thus no new latency period

● WSAPIs, because of very low connection latencies, facilitate live content and the creation of real-time games.

● Protocol is an independent TCP-based protocol. Its only relationship to HTTP is that its handshake is interpreted by HTTP servers as an upgrade request.

● Protocol uses default port 80 for regular WebSocket connections using ws:// and port 443 when using wss:// for WebSocket connections tunnelled over Transport Layer Security (TLS).

● Protocol is intended to be compatible with HTTP-based server-side software and intermediaries, so that a single port can be used by both HTTP clients talking to that server and WebSocket clients talking to that server. Therefore, WebSocket client's handshake is an HTTP upgrade request. Response from the server is also as an HTTP upgrade .

● Protocol specifies six frame types and leaves ten reserved for future use (version 13).

● Clients and servers exchange the ‘messages’ after a successful handshake. A WebSocket message does not necessarily correspond to a particular network layer framing. A frame has an associated type. Each frame belonging to the same message contains the same type of data. Six types are (i) textual data (UTF-8) and (ii) binary data (whose interpretation is left up to the application).

● Extensibility of request-response (client-server) architecture to iq (information through and querying) chat and super chat extensibility to cloud services.

● Support from intelligent and business analyst applications and processing through web server or XMPP server and gateway for connecting the device network with the IP network.

CHAPTER IV
Internet Connectivity Principles

4.1 INTRODUCTION

Internet is a global network with a set of connectivity protocols for:

● Connected devices gateway for sending the data frames of the devices or to the devices. The data communicate over the network as packets, which communicate through a set of routers at the Internet. The processes manage, acquire, organise and analyse the data of the IoT devices for applications, services and business processes.

● The devices perform the controlling and monitoring functions using the messages, data-stacks and commands sent through the Internet by the applications, services or business processes. Following are some of the key terms and their meanings, which are required for understanding Internet connectivity principles for communication between connected devices network and IoT applications.

Header refers to words, which are required for processing a received data stack at a layer and which envelopes the data stack of the preceding upper layer before transfer to the succeeding lower layer. Header consists of header fields. Each word has 32 bits. Each header word can have one or more fields. The fields in the words are as per the processing required at succeeding stages up to the destination.

IP header refers to header fields, which comprise parameters and their encodings as per the IP protocol. IP is Internet layer protocol at the source or destination.

TCP header denotes header fields containing parameters whose encoding is as per the TCP protocol. TCP is transport layer protocol at the source or destination.

Protocol Data Unit (PDU) is the unit of data stack maximum number of bytes, which can be processed at a layer as per the protocol at a layer or sublayer.

TCP stream is a sequence of bytes or words in the data stack created at the transport layer that transmits to the destination-end transport layer.

Maximum Transferable Unit (MTU) is the unit of data-stack maximum number of bytes, which can be transferred from a higher layer to lower layer or physical network.

Packet is a set of bytes with a fixed maximum specified size that transfers from network layer and communicates from one router to another, until it reaches at physical, data-link and network layer at the receiver's end. Internet technology provisions for packetising the received data stack at Internet layer for transmission to the next lower layer. Receiving-end Internet layer does de-packetising (unpacking) at the layer for sending it to the succeeding transport layer.

IP packet is a data stack, which includes IP header. It communicates from a source IP address through the routers to the destination IP address.

Data segment refers to data stack from application-support layer for transport. Application data is divided into the segments when its size is more than the transportable limit. Network interface is a system software component or hardware for facilitating communication between two protocol layers/computers/nodes in a network. The interface software-component provides standard functions. For example, connection establishment, closing and message passing. Network interface examples are port (software or hardware component), network interface device and socket. An interface can be addressed by a unique port number/socket name/node id.

Port is an interface to the network using a protocol that sends an application layer data stack to the lower layer for transmission. The port receives the data stack at the receiver's end from the lower layer. Each port uses an assigned number according to a protocol, which is used for transmission or reception at the application layer. For example, Port 80 is assigned number to HTTP, an application layer protocol.

Socket is a software interface to the network that links to data stack using a port protocol and an IP address. Internet data can be considered as communicating between the sockets. Application data can be considered to flow between the sockets at sender and receiver.

Host is a device or node that connects to a network of computers. It provides information, resources, services and applications to the other nodes on the network. A network layer assigns a host address to each host.

IP host is the one that uses the Internet protocol suite. An IP host has one or more IP addresses for the network interfaces.

Subnet is a subnetwork, which is logical and visible subdivision of an IP network. Subdivision enables addressing a set of networked computers in the subnet using a common and identical IP address. For example, an organisation has 1024 computers but uses a common and identical IP address(es). An IP address consists of two groups: msbs and lsbs, total 32-bits; msbs means most significant bits in a word, while lsbs means least significant bits in a set of bits.

Routing Prefix: Thirty-two bits IP address can be divided into the msbs consisting of 8, 16 or 24 bits and remaining lsbs. The division results in the logical division of an IP address into two fields—a network address or routing prefix field and a rest field or host identifier. The rest field is an identifier for a specific host or network interface.1

Host Identifier: The rest field may also have two sub-fields—one for subnet id and other for the host identifier. When a network subdivides into subnets and subnet has a number of hosts.

Data Flow Graph (DFG) denotes a graphical representation using arrows from one stage to another. A circle represents a stage and arrow characterises the direction of flow of data. Inputs (for example, incoming data for routing) at each stage compute (process) and cause the outputs from the stage. Inputs at the beginning of a circle continue to flow to the next circle, until the outputs (for example, outgoing data for routing) reach at the end of the circle. Data can be said to flow from beginning to end after being processed in the multiple stages in-between.

Acyclic Data Flow Graph (ADFG) refers to a DFG where only one set of inputs generate only one set of outputs for the given input-set in the DFG model. All inputs are instantaneously available in APDFG, (no delay between various inputs) at each stage, except the processing interval at the stage. Examples of non-acyclic data input are an event input, a status-flag set (= 1) or reset (= 0) in a device, and input as per output condition of the previous process.

Directed Acyclic Graph (DAG) means an ADFG in which none of the output cycles back to a previous processing stage or level or rank as an input during data flow. Following sections describe Internet-connectivity principles for connected devices. For details of the protocols, their usages and implementation, the reader can refer to any standard book on computer network or internet and web technology.

4.2 INTERNET CONNECTIVITY

Internet connectivity is through a set of routers in a global network of routers which carry data packets as per IP protocol from a source end to another and vice versa. A source sends data packets to a destination using IETF standardised formats.

The linked image cannot be displayed. The file may have been moved, renamed, or deleted. Verify that the link points to the correct file and location.

Source-end network-layer connected through a set of IP routers for data packets from an IP address and communicating with IoT/M2M IoT application and services layer using TCP/IP suite of application protocols

4.3 INTERNET-BASED COMMUNICATION

4.3.1 Internet Protocols

Internet Protocol Version 4 Let n = total number of header words. Figure 4.3 shows data stack received or transmitted at or to network layer and IP packet consists of IP header fields of n words (= 160 bits and extended header option words).

The extension is done when required actions and using data stack from or for the transport layer. Internet layer protocol is abbreviated as IP and refers to the process when a packet transmits data. The transmission is unacknowledged data flow. IP packet segment consists of the data which the Internet layer receives on transfer from the transport layer to the receiver's end, when using the IP protocol. Protocol data unit, PDUIP = 1 packet and has maximum 216 B. PDUIP is the maximum data unit which can transmit or receive at the layer when using IP packets.

The linked image cannot be displayed. The file may have been moved, renamed, or deleted. Verify that the link points to the correct file and location.

Data stack received or transmitted at or to network layer and IP packet consists of IP header field 160 bits and extended header up to bit q (extended when required) plus data stack from or for the transport layer

The features of IPv4 are:

● IP header consists of five words. The header can extend when using option and padding words. Data stack to the network layer has maximum v = (n + len) words where v <= (214 – n).

● Header first, second and third word fields are as shown in the figure (meaning of header fields in the first three words and option words are not given here and the reader can refer to the author’s Internet and Web Technology book).

● Header fourth and fifth words are source IP address and destination IP address.

● IP protocol transport is half duplex unacknowledged data flow from Internet layer at one end (End 1) to internet layer of other end (End 2).

● Each IP layer data stack is called IP packet and this packet is not guaranteed to reach the destination when the transport layer protocol is UDP, and guaranteed to reach the destination when the transport layer protocol is TCP.

● One packet communicates in one direction at an instance.

Internet Protocol Version 6

Internet Protocol Version 6 is a protocol with the following features:

● Provisions a larger addressing space

● Permits hierarchical address allocation, and thus route aggregation across the Internet, and limit the expansion of routing tables

● Provisions additional optimisation for the delivery of services using routers, subnets and interfaces ● Manages device mobility, security and configuration aspects

● Expanded and simple use of multicast addressing

● Provisions jumbo grams (big size datagram) ● Extensibility of options While an IPv4 address is of size 32-bit, the IPv6 address has a size of 128 bits.2 Therefore, IPv6 has a vastly enlarged address space compared to IPv4. The IPv6 address provides a numerical label.

It identifies a network interface of a node or other network nodes and subnets participating in IPv6 Internet. A device is the node in the network when it communicates on a network.

4.3.1 6LoWPAN

Internet layer IPv6 receives and transmits from/to adaptation layer . The data stack uses 6LoWPAN (IPv6 Over Low Power Wireless Personal Area Network) protocol at adaptation layer before the data stack transmits to IPv6 Internet layer. An IEEE 802.15.4 WPAN device has a 6LowPAN interface serial port for connectivity. 6LoWPAN is an adaptation-layer protocol for the IEEE 802.15.4

network devices. The devices are the nodes having low speed and low power. They are the WPAN nodes of a multiple device mesh network. Low-power devices need to limit data size per instance. Data compression reduces data size. Fragmentation of data also reduces data size per instance. Features of 6LoWPAN are header compression, fragmentation and reassembly. When data is fragmented before communication, the first fragment header has 27 bits which includes the datagram size (11 bits) and a datagram tag (16 bits). Subsequent fragments have header 8 bits which include the datagram size, datagram tag and the offset. Fragments reassembly time limit can be set equal to 60s.

The linked image cannot be displayed. The file may have been moved, renamed, or deleted. Verify that the link points to the correct file and location.

(a) Networked i devices at physical layer in IEEE 802.15.4 WPAN and (b) Adaptation layer 6LoWPAN protocol 127 B (maximum) fragmented frames reassembly into IPv6 maximum 1280 B or fragmentation of IPv6 MTU 1280 B into 127 B frames for transfer to a device.

IPv6 Maximum Transmission Unit (MTU) at link layer = 1280 B. Therefore, link-layer frame fragmentation is needed in order to communicate frame of 127 B over IEEE 802.15.4 nodes (device). The frame MTU is 1280 B for transmission to network layer. Fragments from frames from the device of 127 B each reassemble into an IPv6 frame. Also, IPv6 MTU at data-link layer 1280 B fragments into frame of 127 B each for single transfer to a device node.

6LoWPAN has the following features:

● Specifies the IETF recommended methods for reassembly of fragments, and IPv6 and UDP (or ICMP) headers compression (6LoWPAN-hc adaptation layer), neighbour discovery (6LoWPAN-nd adaptation layer), and

● Supports mesh routing ICMP stands for Internet Control Message Protocol. Routers or other devices on the network send error messages or relay the query messages. 6LoWPAN can be implemented using Berkley IP implementation with operating system TinyOS or 3BSD or other implementations for IoT nodes from Sensinode or Hitachi or others. IPv6 network layer has two options, viz. RH4 routing header and hop-to-hop header RPL option.

4.4 APPLICATION LAYER PROTOCOLS: HTTP, HTTPS, FTP, TELNET AND OTHERS

4.4.1 HTTP and HTTPS

Ports Hyper Text Transfer Protocol (HTTP) port number is 80. A web HTTP server listens to port 80 only and responds to port 80 only. An HTTP port sends application data stack at the output to the lower layer using the HTTP protocol.

An HTTP port uses a URL like http://www. mheducation.com/. The default port is taken as 80. The port number can be specified after the TLD. For example, after '.com' in URL http://www. theducation.com:80/.

HTTPS (HTTP over Secure Socket Layer or TLS) port number is 443. An HTTPS port sends a URL; for example, https://en.wikipedia.org/wiki/List_of_TCP_and_UDP_port_ numbers. Here, TLD is .org, domain name is wikipedia.org and Subdomain name is en. Resource URL is at /wiki/List_of_TCP_and_UDP_port_numbers.

The port receives the data stack at the input at the receiver end. Each port at the application layer uses a distinct protocol. A port is assigned a number according to protocol used for transmission and reception. The important features of HTTP are:

● HTTP is the standard protocol for· requesting a URL defined web-page resource, and for sending a response to the web server. An HTTP client requests an HTTP server on the Internet and the server responds by sending a response. The response may be with or without applying a process.

● HTTP is a stateless protocol. This is because for an HTTP request, the protocol assumes a fresh request. It means there is no session or sequence number field or no field that is retained in the next exchange. This makes a current exchange by an HTTP request independent of the previous exchanges. The later exchanges do not depend on the current one. E-commerce like application needs a state management mechanism. The stateless feature of HTTP is compensated by a method as: A cookie is a text file which creates during a particular pair of exchanges of HTTP request and response. The creation is either at a CGI or processing program. For example, JavaScript or script or at a client. A prior exchange may then depend on this cookie. The cookie thus provides an HTTP state management mechanism.

Basically, HTTP is a file transfer-like protocol. We use it more efficiently than the FTP (a protocol for a file transfer on the Internet) because in FTP, we have to give a certain command. Then, a communication establishes between two systems just to retrieve a specified file. On the other hand, HTTP is simple. There are no command line overheads. This makes it easy to explore a website URL. A request (from a client) and reply (from a server) is the paradigm.

● The HTTP protocol is very light (a small format) and thus speedy as compared to other protocols, such as FTP. HTTP is able to transfer any type of data to a client provided it is capable of handling that data.

● HTTP is flexible. Assume during a client web connection, the connection breaks. The client can start by re-connecting. Being a stateless protocol, HTTP does not keep track of the state as FTP does. Each time a connection establishes between the web server and the client, both these interpret this connection as a new connection. Simplicity is a must because a webpage has the URL resources distributed over a number of servers.

● HTTP protocol is based on Object Oriented Programming System (OOPS). Methods are applied to objects identified by a URL. It means as in the normal case of an Object Oriented Program, various methods apply on an object.

● Following features have been included from HTTP 1.0 and 1.1 version onwards: (a) Multimedia file access is feasible due to provision for the MIME (Multipurpose Internet Mail Extension) type file definition. ● Eight HTTP specific specified methods and extension methods included from HTTP 1.1 version onwards. The HTTP specific methods are as follows. 1. GET. 2. POST. 3. HEAD.

4. CONNECT. 5. PUT. 6. DELETE. 7. TRACE. 8. OPTIONS. (Last four from 1.1.)

● Earlier versions, GET follows a space and then the document name. The server returns the documents and closes the connection. The POST method permits form processing as using it the client transmits the form data or other information to the server from 1.1. Also, the server does not close the connection after response and thus response can be processed before it is sent.

● A provision of user authentication exits besides the basic authentication.

● Digest Access Authentication prevents the transmission of username and password as HTML or text from HTTP 1.1 version onwards.

● A host header field adds to support those ports and virtual hosts that do not accept or send IP packets. An Error report to Client when an HTTP request is without a Host header field from HTTP 1.1 version onwards.

● An absolute URL is acceptable to the server. In the earlier version, only a proxy server accepted that.

● Message Headers uses are: A message during a request from a client or during a response from a server consists of two parts—a start-line, none or several message-headers (fields) and empty line, and body of the message. Message headers are:

❍ Common (general) headers are added when requesting a server and when responding to a client. A header includes MIME version, OPTIONS, cacheable or not cacheable, and transfer to close or not close the connection.

❍ Request headers are for a request and client information to a server. The header includes acceptable media or preferred specification—specifies about acceptability of HTML or HTML or text or any other type informs if a special type of character set acceptable.

❍ Entity headers contain information about the entity body contained in the message or in case the body is not present then information about the entity is not present its body. For example, information for content-length in bytes.

❍ Response headers are present in the response for server information to a client.

● Status codes add in the response and caching of a resource provided at a server (and proxy). For example, status code returned as response when 400 means a bad request (the request unrespondable), 401 unauthorized request, 402 means request requires a payment before response feasible, 403 means request is for a forbidden resource and 404 means URL resource not found by the server.

● Byte range specification helps an HTTP server to send large response in parts. Length specification helps in presentation in chunks.

● Selection among various characteristics on retrieval by the client is feasible when a server sends a response to the client request. For example, the two characteristics, language and encoding can be specified in the server environment variables while the client sends the request header for retrieving a resource. The resource then retrieves in that language and with that encoding. The contents sent to the client do not change, only the way in which these are presented to the client change.

CHAPTER V
Data Acquiring, Organizing, Processing & Analytics

5.1 DATA ACQUIRING AND STORAGE

Following subsections describe devices data, and steps in acquiring and storing data for an application, service or business process.

5.1.1 Data Generation

Data generates at devices that later on, transfers to the Internet through a gateway. Data generates as follows:

● Passive devices data: Data generate at the device or system, following the result of interactions. A passive device does not have its own power source. An external source helps such a device to generate and send data. Examples are an RFID or an ATM debit card . The device may or may not have an associated microcontroller, memory and transceiver. A contactless card is an example of the former and a label or barcode is the example of the latter.

● Active devices data: Data generates at the device or system or following the result of interactions. An active device has its own power source. Examples are active RFID, streetlight sensor or wireless sensor node. An active device also has an associated microcontroller, memory and transceiver.

● Event data: A device can generate data on an event only once. For example, on detection of the traffic or on dark ambient conditions, which signals the event. The event on darkness communicates a need for lighting up a group of streetlights . A system consisting of security cameras can generate data on an event of security breach or on detection of an intrusion. A waste container with associate circuit can generate data in the event of getting it filled up 90% or above. The components and devices in an automobile generate data of their performance and functioning. For example, on wearing out of a brake lining, a play in steering wheel and reduced air-conditioning is felt. The data communicates to the Internet. The communication takes place as and when the automobile reaches near a Wi-Fi access point.

● Device real-time data: An ATM generates data and communicates it to the server instantaneously through the Internet. This initiates and enables Online Transactions Processing (OLTP) in real time.

● Event-driven device data: A device data can generate on an event only once. Examples are: (i) a device receives command from Controller or Monitor, and then performs action(s) using an actuator. When the action completes, then the device sends an acknowledgement; (ii) When an application seeks the status of a device, then the device communicates the status.

5.1.2 Data Acquisition

Data acquisition means acquiring data from IoT or M2M devices. The data communicates after the interactions with a data acquisition system (application). The application interacts and communicates with a number of devices for acquiring the needed data. The devices send data on demand or at programmed intervals. Data of devices communicate using the network, transport and security layers.

An application can configure the devices for the data when devices have configuration capability. For example, the system can configure devices to send data at defined periodic intervals. Each device configuration controls the frequency of data generation. For example, system can configure an umbrella device to acquire weather data from the Internet weather service, once each working day in a week (Example 1.1). An ACVM can be configured to communicate the sales data of machine and other information, every hour. The ACVM system can be configured to communicate instantaneously in event of fault or in case requirement of a specific chocolate flavour needs the Fill service.

Application can configure sending of data after filtering or enriching at the gateway at the data-adaptation layer. The gateway in-between application and the devices can provision for one or more of the following functions—transcoding, data management and device management. Data management may be provisioning of the privacy and security, and data integration, compaction and fusion.

Device-management software provisions for device ID or address, activation, configuring (managing device parameters and settings), registering, deregistering, attaching, and detaching.

5.1.3 Data Validation

Data acquired from the devices does not mean that data are correct, meaningful or consistent. Data consistency means within expected range data or as per pattern or data not corrupted during transmission. Therefore, data needs validation checks. Data validation software do the validation checks on the acquired data. Validation software applies logic, rules and semantic annotations. The applications or services depend on valid data. Then only the analytics, predictions, prescriptions, diagnosis and decisions can be acceptable. Large magnitude of data is acquired from a large number of devices, especially, from machines in industrial plants or embedded components data from large number of automobiles or health devices in ICUs or wireless sensor networks, and so on.

Validation software, therefore, consumes significant resources. An appropriate strategy needs to be adopted. For example, the adopted strategy may be filtering out the invalid data at the gateway or at device itself or controlling the frequency of acquiring or cyclically scheduling the set of devices in industrial systems. Data enriches, aggregates, fuses or compacts at the adaptation layer.

5.1.4 Data Categorisation for Storage

Services, business processes and business intelligence use data. Valid, useful and relevant data can be categorised into three categories for storage—data alone, data as well as results of processing, only the results of data analytics are stored.

Following are three cases for storage:

1. Data which needs to be repeatedly processed, referenced or audited in future, and therefore, data alone needs to be stored.

2. Data which needs processing only once, and the results are used at a later time using the analytics, and both the data and results of processing and analytics are stored. Advantages of this case are quick visualisation and reports generation without reprocessing. Also the data is available for reference or auditing in future.

3. Online, real-time or streaming data need to be processed and the results of this processing and analysis need storage.

5.1.5 Assembly Software for the Events

A device can generate events. For example, a sensor can generate an event when temperature reaches a preset value or falls below a threshold. A pressure sensor in a boiler generates an event when pressure exceeds a critical value which warrants attention. Each event can be assigned an ID. A logic value sets or resets for an event state. Logic 1 refers to an event generated but not yet acted upon. Logic 0 refers to an event generated and acted upon or not yet generated. A software component in applications can assemble the events (logic value, event ID and device ID) and can also add Date time stamp. Events from IoTs and logic-flows assemble using software.

5.1.6 Data Store

A data store is a data repository of a set of objects which integrate into the store. Features of data store are: ● Objects in a data-store are modeled using Classes which are defined by the database schemas.

● A data store is a general concept. It includes data repositories such as database, relational database, flat file, spreadsheet, mail server, web server, directory services and VMware

● A data store may be distributed over multiple nodes. Apache Cassandra is an example of distributed data store.

● A data store may consist of multiple schemas or may consist of data in only one scheme. Example of only one scheme data store is a relational database. Repository in English means a group, which can be related upon to look for required things, for special information or knowledge. For example, a repository of paintings of artists. A database is a repository of data which can be relied upon for reporting, analytics, process, knowledge discovery and intelligence. A flat file is another repository.

5.1.7 Data Centre Management

A data centre is a facility which has multiple banks of computers, servers, large memory systems, high speed network and Internet connectivity. The centre provides data security and protection using advanced tools, full data backups along with data recovery, redundant data communication connections and full system power as well as electricity supply backups.

5.1.8 Server Management

Server management means managing services, setup and maintenance of systems of all types associated with the server. A server needs to serve around the clock. Server management includes managing the following:

- Short reaction times when the system or network is down
- High security standards by routinely performing system maintenance and updation
- Periodic system updates for state-of-the art setups
- Optimised performance
- Monitoring of all critical services, with SMS and email notifications
- Security of systems and protection
- Maintaining confidentiality and privacy of data
- High degree of security and integrity and effective protection of data, files and databases at the organisation
- Protection of customer data or enterprise internal documents by attackers which includes spam mails, unauthorised use of the access to the server, viruses, malwares and worms
- Strict documentation and audit of all activities.

5.2 ORGANISING THE DATA

Data can be organised in a number of ways. For example, objects, files, data store, database, relational database and object oriented database. Following subsections describe these ways of organising and querying methods.

5.2.1 Databases

Required data values are organised as database(s) so that select values can be retrieved later. Database One popular method of organising data is a database, which is a collection of data. This collection is organised into tables. A table provides a systematic way for access, management and update. A single table file is called flat file database. Each record is listed in separate row, unrelated to each other.

Relational Database

A relational database is a collection of data into multiple tables which relate to each other through special fields, called keys (primary key, foreign key and unique key). Relational databases provide flexibility. Examples of relational database are MySQL, PostGreSQL, Oracle database created using PL/SQL and Microsoft SQL server using T-SQL. Object Oriented Database (OODB) is a collection of objects, which save the objects in objected oriented design.

Database Management System

Database Management System (DBMS) is a software system, which contains a set of programs specially designed for creation and management of data stored in a database. Database transactions can be performed on a database or relational database. Atomicity, Data Consistency, Data Isolation and Durability (ACID) Rules The database transactions must maintain the atomicity, data consistency, data isolation and durability during transactions. Let us explain these rules using Example 5.3 as follows: Atomicity means a transaction must complete in full, treating it as indivisible. When a service request completes, then the pending request field should also be made zero. Consistency means that data after the transactions should remain consistent. For example, sum of chocolates sent should equal the sums of sold and unsold chocolates for each flavour after the transactions on the database. Durability means after completion of transactions, the previous transaction cannot be recalled. Only a new transaction can affect any change.

Distributed Database

Distributed Database (DDB) is a collection of logically interrelated databases over a computer network. Distributed DBMS means a software system that manages a distributed database. The features of a distributed database system are: ● DDB is a collection of databases which are logically related to each other. ● Cooperation exists between the databases in a transparent manner. Transparent means that each user within the system may access all of the data within all of the databases as if they were a single database. ● DDB should be 'location independent', which means the user is unaware of where the data is located, and it is possible to move the data from one physical location to another without affecting the user.

5.2.2 Query Processing

Query means an application seeking a specific data set from a database. For example, a query at a relational database at bank server may be for the ATM transactions made in a month by a specific customer ID.

Query Processing

Query processing means using a process and getting the results of the query made from a database. The process should use a correct as well as efficient execution strategy. Five steps in processing are:

1. Parsing and translation: This step translates the query into an internal form, into a relational algebraic expression and then a Parser, which checks the syntax and verifies the relations.

2. Decomposition to complete the query process into micro-operations using the analysis (for the number of micro-operations required for the operations), conjunctive and disjunctive normalisation and semantic analysis.

3. Optimisation which means optimising the cost of processing. The cost means number of micro-operations generated in processing which is evaluated by calculating the costs of the sets of equivalent expressions.

4. Evaluation plan: A query-execution engine (software) takes a query-evaluation plan and executes that plan.

5. Returning the results of the query.

5.2.3 SQL

SQL stands for Structured Query Language. It is a language for viewing or changing (update, insert or append or delete) databases. It is a language for data querying, updating, inserting, appending and deleting the databases. It is a language for data access control, schema creation and modifications. It is also a language for managing the RDBMS. SQL was originally based upon the tuple relational calculus and relational algebra. SQL can embed within other languages using SQL modules, libraries and pre-compilers. SQL features are as follows:

● Create Schema is a structure that contains descriptions of objects created by a user (base tables, views, constraints). The user can describe and define the data for a database.

● Create Catalog consists of a set of schemas that constitute the description of the database.

● Use Data Definition Language (DDL) for the commands that depict a database, including creating, altering and dropping tables and establishing constraints. The user can create and drop databases and tables, establish foreign keys, create view, stored procedure, functions in a database.

● Use Data Manipulation Language (DML) for commands that maintain and query a database. The user can manipulate (INSERT, UPDATE or SELECT the data and access data in relational database management systems.

● Use Data Control Language (DCL) for commands that control a database, including administering privileges and committing data. The user can set (grant or add or revoke) permissions on tables, procedures, and views.

5.2.4 NOSQL

NOSQL stands for No-SQL or Not Only SQL that does not integrate with applications that are based on SQL. NOSQL is used in cloud data store. NOSQL may consist of the following:

● A class of non-relational data storage systems, flexible data models and multiple schemas Class consisting of uninterpreted key and value or 'the big hash table'. For example in [Dynamo (Amazon S3)]

● Class consisting of unordered keys and using the JSON. For example in PNUTS

● Class consisting of ordered keys and semi-structured data storage systems. For examples in the BigTable, Hbase and Cassandra (used in Facebook and Apache)

● Class consisting of JSON (Section 2.3). For example in MongoDb6 which is widely used for NOSQL)

● Class consisting of name and value in the text. For example in CouchDB

● May not require a fixed table schema

NOSQL systems do not use the concept of joins (in distributed data storage systems). Data written at one node replicates to multiple nodes, therefore identical and distributed system can be fault-tolerant, and can have partitioning tolerance. CAP theorem is applicable. The system offers relaxation in one or more of the ACID and CAP properties. Out of the three properties (consistency, availability and partitions), two are at least present for an application.

● Consistency means all copies have same value like in traditional DBs.

● Availability means at least one copy available in case a partition becomes inactive or fails. For example, in web applications, the other copy in other partition is available.

● Partition means parts which are active but may not cooperate as in distributed databases

5.3 TRANSACTIONS, BUSINESS PROCESSES, INTEGRATION AND ENTERPRISE SYSTEMS

A transaction is a collection of operations that form a single logical unit. For example, a database connect, insertion, append, deletion or modification transactions. Business transactions are transactions related in some way to a business activity.

5.3.1 Online Transactions and Processing

OLTP means process as soon as data or events generate in real time. OLTP is used when requirements are availability, speed, concurrency and recoverability in databases for real-time data or events.

Batch Transactions Processing

Batch transactions processing means the execution of a series of transactions without user interactions. Transaction jobs are set up so they can be run to completion. Scripts, command-line arguments, control files, or job control language predefine all input parameters. Batch processing means a transaction process in batches and in an non-interactive way. When one set of transactions finish, the results are stored and a next batch is taken up. A good example is credit card transactions where the final results at the end of the month are used. Another example is chocolate purchase transactions. The final results of sell figures from ACVMs can communicate on the Internet at the end of an hour or day.

Streaming Transactions Processing

Examples of the streams are log streams, event streams and twitter streams. Query and transactions processing on streaming data need specialised frameworks. Storm from Twitter, S4 from Yahoo, SPARK streaming, HStreaming and flume are examples of frameworks for real-time streaming computation frameworks.

Interactive Transactions Processing

Interactive transactions processing means the transactions which involve continual exchange of information between the computer and a user. For example, user interactions during e-shopping and e-banking. The processing is just the opposite of batch processing.

Real-time Transactions Processing

Real-time transaction processing means that transactions process at the same time as the data arrives from the data sources and data store. An example is ATM machine transactions. In-memory, row-format records enable real-time transaction processing. Row format means few rows and more columns. The CPU accesses all columns in single accesses in SIMD (single instruction multiple data) streams processing.

Event Stream Processing and Complex Event Processing

Event Stream Processing (ESP) is a set of technologies, event processing languages, Complex Event Processing (CEP), event visualisation, event databases and event-driven middleware. Apache S4 and Twitter Storm are examples of ESPs. SAP Sybase ESP and EsperTechEsper are examples of CEPs. ESP and CEP does the following:

- Processes tasks on receiving streams of event data
- Identifies the meaningful pattern from the streams
- Detects relationships between multiple events
- Correlates the events data
- Detects event hierarchies
- Detects aspects such as timing, causality, subscription membership
- Builds and manages the event-driven information systems.

5.3.2 Business Processes

A business process consists of a series of activities which serves a particular specific result. It is used when an enterprise has a number of interrelated processes which serve a particular result or goal. The results enable sales, planning and production. The BP is a representation or process matrix or flowchart of a sequence of activities with interleaving decision points. Internet of RFIDs enables a business process called tracking of RFID labelled goods which also enables inventory control process. IoT/M2M enables the devices' data in databases for business processes. The data supports the process. For example, consider a process, streetlights control and management . Each group of streetlights sends data in real time through the gateways. The gateways connect to the Internet. The control and management processes streetlights real time databases and group databases.

5.3.3 Business Intelligence

Business intelligence is a process which enables a business service to extract new facts and knowledge and then undertake better decisions. The new facts and knowledge follow from the earlier results of data processing, aggregation and then analysing those results. Example 5.5 shows business intelligence for Internet of ACVMs, whereas Example 5.6 shows business processes, intelligence and BP architecture reference model in Automotive Maintenance Application at the Service Centre.

5.3.4 Distributed Business Process

Several times, business processes need to be distributed. Distribution of processes reduces the complexity, communication costs, enables faster responses and smaller processing load at the central system. For example, recall Example 1.2, distribution of control process for each group of lights at the gateway itself reduces complexity, communication costs, faster responses and smaller processing load at the central system.

Distributed Business Process System (DBPS) is a collection of logically interrelated business processes in an Enterprise network. DBPS means a software system that manages the distributed BPs. DBPS features are: DBPS is a collection of logically related BPs like DDBS. DBPS exists as cooperation between the BPs in a transparent manner. Transparent means that each user within the system may access all of the process decisions within all of the processes as if they were a single business process. DBPS should possess ‘location independence’ which means the enterprise BI is unaware of where the BPs are located. It is possible to move the results of analytics and knowledge from one physical location to another without affecting the user.

5.3.5 Complex Applications Integration and Service Oriented Architecture

An enterprise has number of applications, services and processes. Heterogeneous systems have complexity when integrating them in the enterprise.

Following are the standardised business processes, as defined in the Oracle application integration architecture:

- Integrating and enhancing the existing systems and processes
- Business intelligence
- Data security and integrity
- New business services and products (web services)
- Collaboration and knowledge management
- Enterprise architecture and SOA
- e-commerce
- External customer services
- Supply chain automation and analytics results visualisation
- Data centre optimisation

IoT applications, services and processes enhance the existing systems in a number of enterprises. For example, an automobile enterprise has a number of divisions. Each division has Sales, Customer Relations Management, Automobile Maintenance Services, and Accounting. IoT-based services help in business intelligence, processes and systems, such as post-sales services and supply chain automation and analytics results in visualisation enhancement of the services from an enterprise.

Complex application integration means integration of heterogeneous application architectures and number of processes. SOA consists of services, messages, operations and processes. SOA components distribute over a network or the Internet in a high-level business entity. New business applications can be developed using a SOA.

5.3.6 Integration and Enterprise Systems

Complex applications integration architecture and SOA of cloud-based IoT services, web services, cloud services and services. Process orchestration means a number of business processes running in parallel and a number of processes running in sequence. The process matrix provides the decision points which indicate which processes should run in parallel and which in sequence. An SOA models the number of services and interrelationships.

The linked image cannot be displayed. The file may have been moved, renamed, or deleted. Verify that the link points to the correct file and location.

Complex applications integration architecture and SOA of cloud-based IoT services, web services, cloud services and services

Each service initiates on receipt of messages from a process or service. The service discovery and selection software components select the services for application integration. Service orchestration software coordinates the execution of the number of services, cloud services, cloud IoT services and web services. Services run in parallel and a number of processes in sequences.