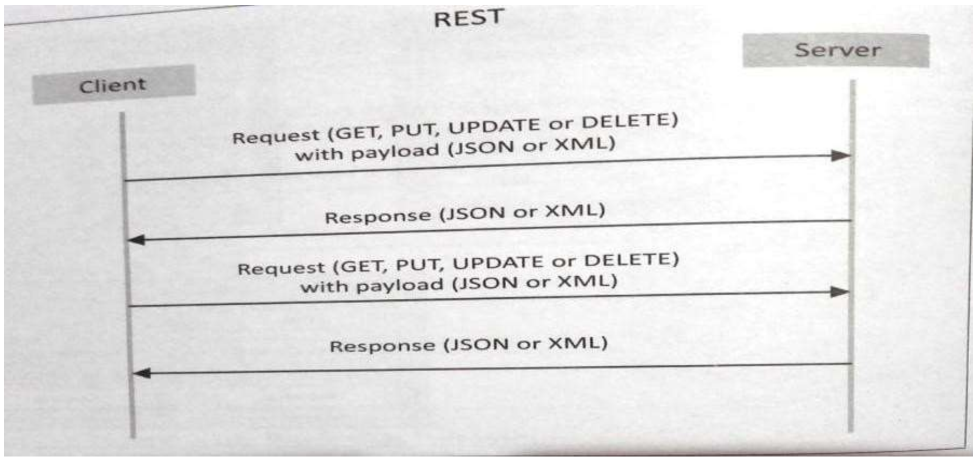
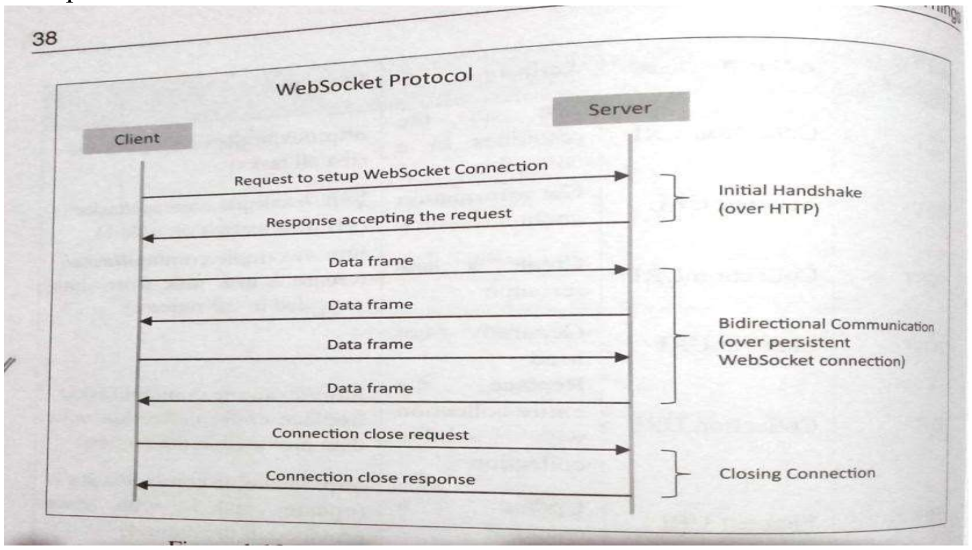
Request-Response model used by REST:


REST
Client
Server
Request (GET, PUT, UPDATE or DELETE) with payload (JSON or XML)
Response (JSON or XML)
Request (GET, PUT, UPDATE or DELETE) with payload (JSON or XML)
Response (JSON or XML)

REST full web service is a collection of resources which are represented by URIs (Uniform Resource Identifiers). REST full web API has a base URI (e.g: http://example.com/api/tasks/). The clients and requests to these URIs using the methods defined by the HTTP protocol (e.g: GET, PUT, POST or DELETE). A REST full web service can support various internet media types.

a) Web Socket Based Communication APIs: Web Socket APIs allow bi-directional, full duplex communication between clients and servers. Web Socket APIs follow the exclusive pair communication model.


38
WebSocket Protocol
Client
Server
Request to setup WebSocket Connection
Response accepting the request
Initial Handshake (over HTTP)
Data frame
Data frame
Data frame
Data frame
Bidirectional Communication (over persistent WebSocket connection)
Connection close request
Connection close response
Closing Connection

It doesn't require a new connection to be set up for each message sent.
It begins with connection request by client.
This type of API reduces the traffic and redundancy of data and make sure that each time when we request a new data, it cannot terminate the request.

IoT Enabling Technologies

IoT is enabled by several technologies including Wireless Sensor Networks, Cloud Computing, Big Data Analytics, Embedded Systems, Security Protocols and architectures, Communication Protocols, Web Services, Mobile internet and semantic search engines.

1) **Wireless Sensor Network (WSN):** Comprises of distributed devices with sensors which are used to monitor the environmental and physical conditions. Zig Bee is one of the mostpopular wireless technologies used by WSNs.
   WSNs used in IoT systems are described as follows:
   - Weather Monitoring System: in which nodes collect temperature, humidity and otherdata, which is aggregated and analyzed.
   - Indoor air quality monitoring systems: to collect data on the indoor air quality andconcentration of various gases.
   - Soil Moisture Monitoring Systems: to monitor soil moisture at various locations.
   - Surveillance Systems: Use WSNs for collecting surveillance data (motion data detection).
   - Smart Grids: Use WSNs for monitoring grids at various points.
   - Structural Health Monitoring Systems: Use WSNs to monitor the health of structures (building, bridges) by collecting vibrations from sensor nodes deployed at various points in the structure.
2) **Cloud Computing:** Services are offered to users in different forms.
   - Infrastructure-as-a-service (IaaS): Provides users the ability to provision computing and storage resources. These resources are provided to the users as a virtual machine instances and virtual storage.
   - Platform-as-a-Service (PaaS): Provides users the ability to develop and deploy application in cloud using the development tools, APIs, software libraries and services provided by the cloud service provider.
   - Software-as-a-Service (SaaS): Provides the user a complete software application orthe user interface to the application itself.
3) **Big Data Analytics:** Some examples of big data generated by IoT are:
   - Sensor data generated by IoT systems.
   - Machine sensor data collected from sensors established in industrial and energysystems.
   - Health and fitness data generated IoT devices.
   - Data generated by IoT systems for location and tracking vehicles.
   - Data generated by retail inventory monitoring systems.
4) **Communication Protocols:** Form the back-bone of IoT systems and enable network connectivity and coupling to applications.
   - Allow devices to exchange data over network.
   - Define the exchange formats, data encoding addressing schemes for device and routing of packets from source to destination.
   - It includes sequence control, flow control and retransmission of lost packets.
5) **Embedded Systems:** It is a computer system that has computer hardware and software embedded to perform specific tasks. Embedded System range from low cost miniaturized devices such as digital watches to devices such as digital cameras, POS terminals, vending machines, appliances etc.

Embedded system + Internet = IOT

IoT Levels and Deployment Templates

IOT system consists of following components:

Devices: An IOT device allows identification, remote sensing, and remote monitoring capacities.

Resources: Software components on IOT devices for accessing, processing and storing data, controlling actuators, and enabling n/w access devices.

Controller Services: sends data from the device to the web service and receives commands from the application for controlling device.

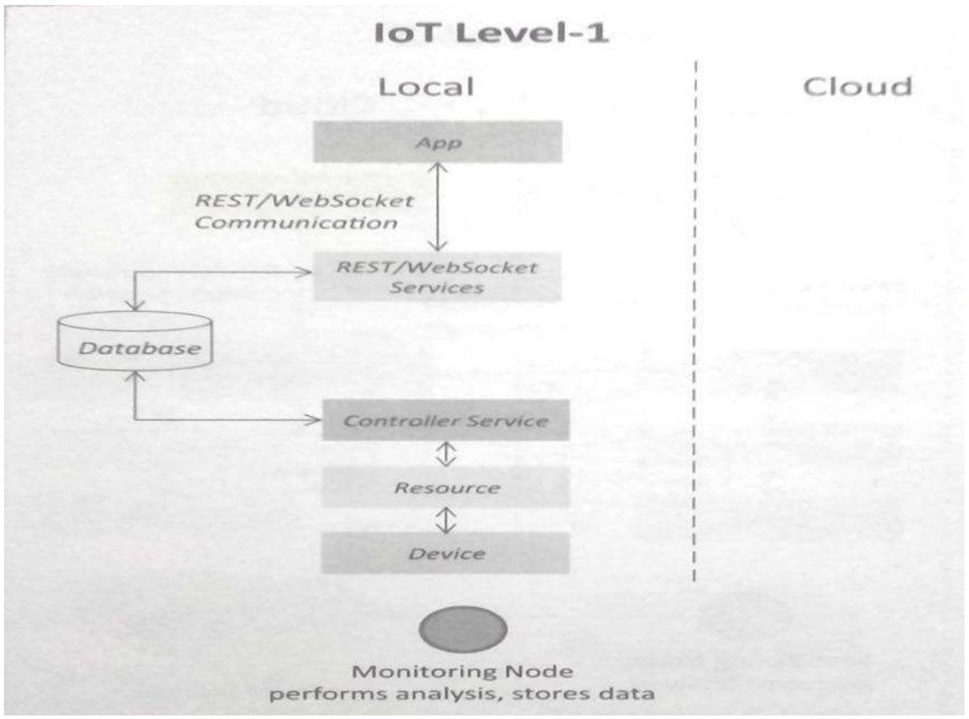
Database: can be local or cloud based and stores the data generated by IOT devices.

Web Service: Serve as a link b/w the IOT device, application, database and analysis components. Can be implemented by HTTP & REST principles or using web Socket protocol.

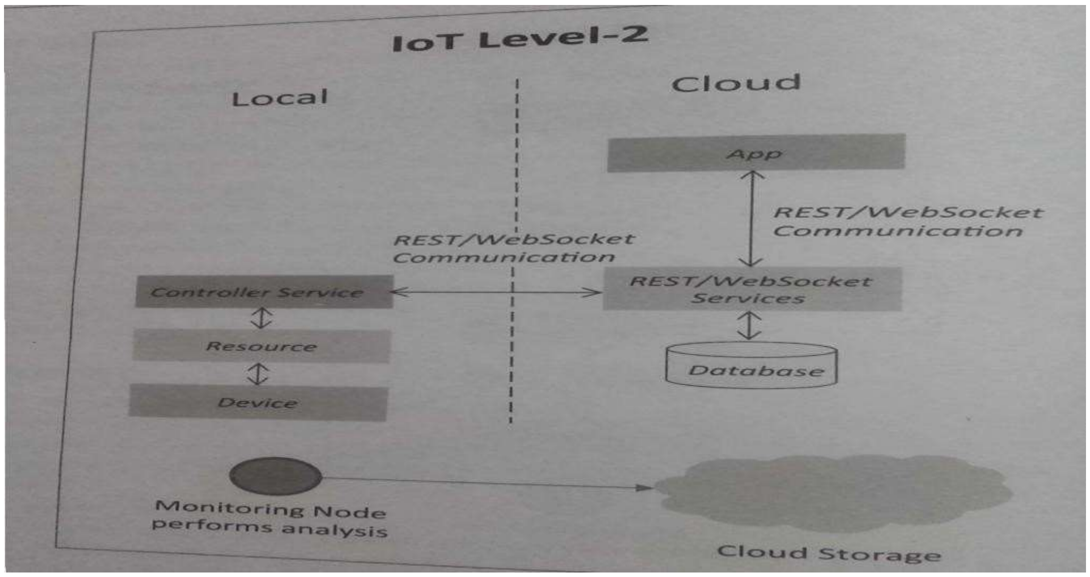
Analysis Component: responsible for analyzing the IOT data and generating results in the form that is easy to understand for users.

Application: provides an interface that the user can use to control and monitor various aspects of IOT system. It also allows users to view the system status and the processed data.

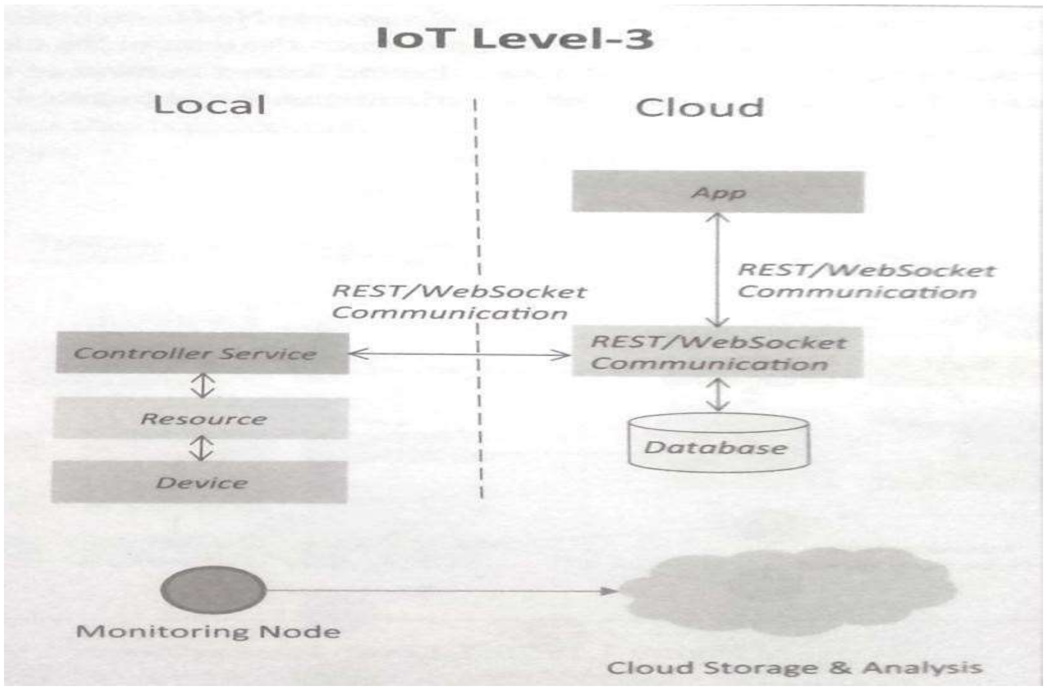
1) **IoT Level 1:** System has a single node that performs sensing and/or actuation, stores data, performs analysis and host the application as shown in figure. Suitable for modeling low cost and low complexity solutions where the data involved is not big and analysisrequirement are not computationally intensive. An e.g., of IoT Level1 is Home automation.



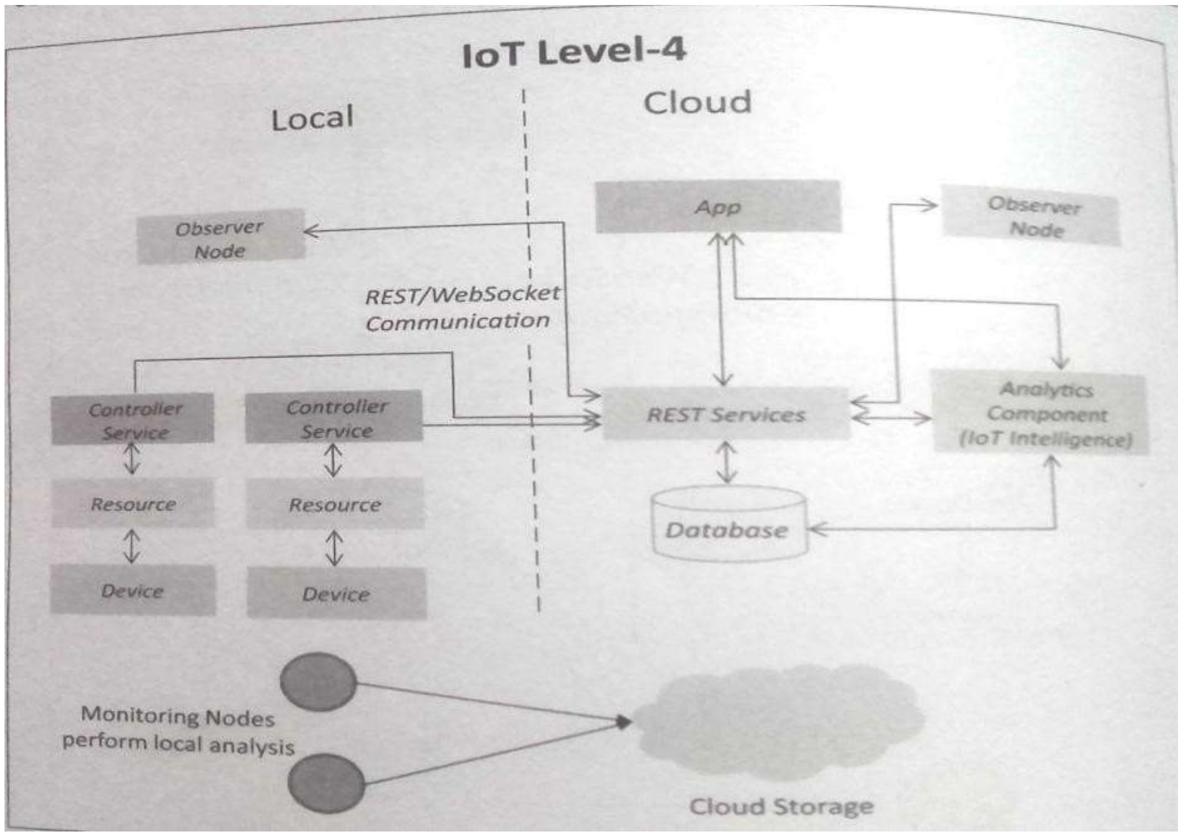
2) **IoT Level 2:** It has a single node that performs sensing and/or actuating and local analysis as shown in fig. Data is stored in cloud and application is usually cloud based. Level2 IoTsystems are suitable for solutions where data are involved is big, however, the primary analysis requirement is not computationally intensive and can be done locally itself. An e,g., of Level2 IoT system for Smart Irrigation.



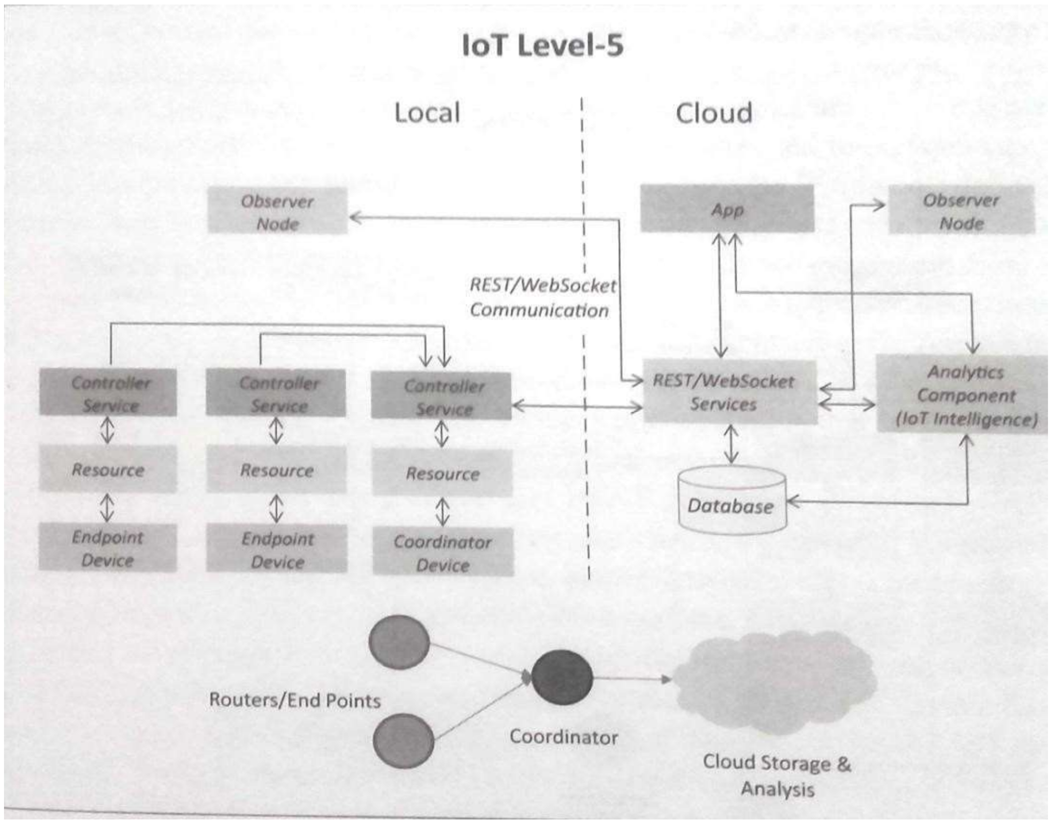
3) **IoT Level 3:** System has a single node. Data is stored and analyzed in the cloud, application is cloud based as shown in fig. Level 3 IoT systems are suitable for solutions where the data involved is big and analysis requirements are computationally intensive. An example of IoT level3 system for tracking package handling.



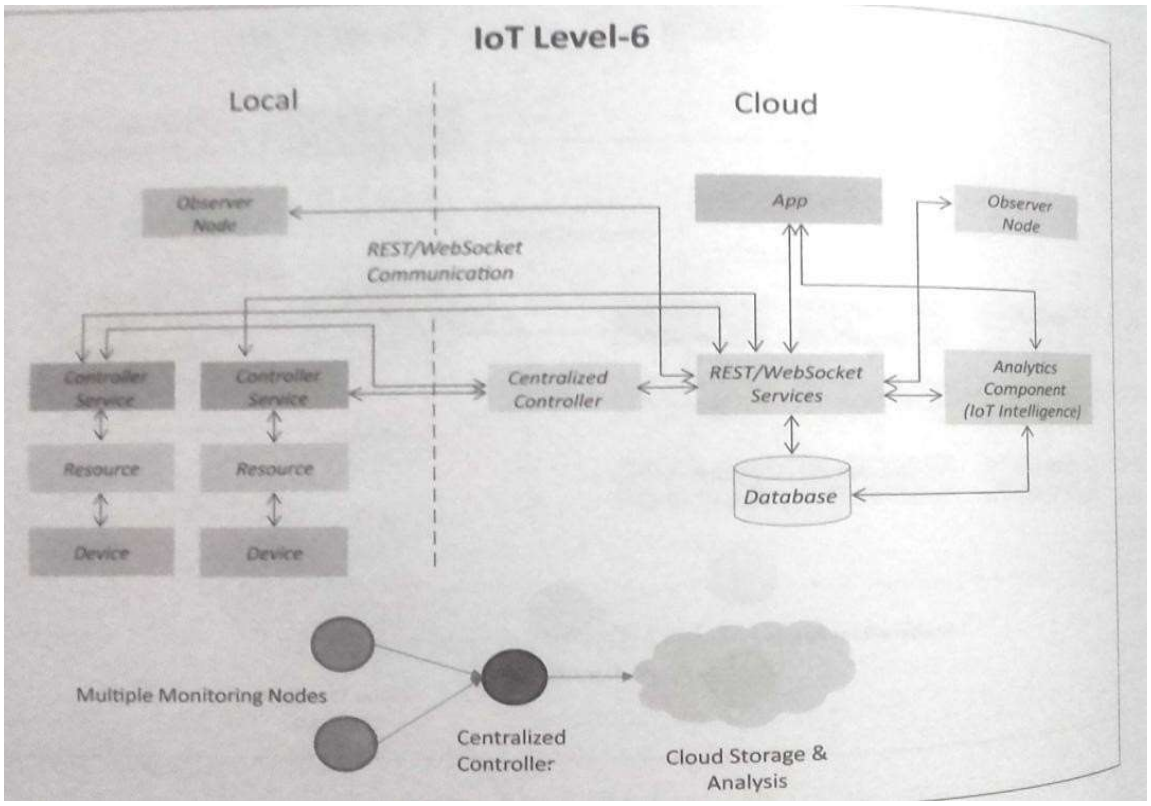
4) **IoT Level 4:** System has multiple nodes that perform local analysis. Data is stored in the cloud and application is cloud based as shown in fig. Level 4 contains local and cloud based observer nodes which can subscribe to and receive information collected in the cloud from IoT devices. An example of a Level4 IoT system for Noise Monitoring.



5) **IoT Level 5:** System has multiple end nodes and one coordinator node as shown in fig. The end nodes that perform sensing and/or actuation. Coordinator node collects data from the end nodes and sends to the cloud. Data is stored and analyzed in the cloud and application is cloud based. Level 5 IoT systems are suitable for solution based on wireless sensor network, in which data involved is big and analysis requirements are computationally intensive. An example of Level5 system for Forest Fire Detection.



6) **IoT Level 6:** System has multiple independent end nodes that perform sensing and/or actuation and sensed data to the cloud. Data is stored in the cloud and application is cloud based as shown in fig. The analytics component analyses the data and stores the result in the cloud data base. The results are visualized with cloud based application. The centralized controller is aware of the status of all the end nodes and sends control commands to nodes. An example of a Level 6 IoT system for Weather Monitoring System.



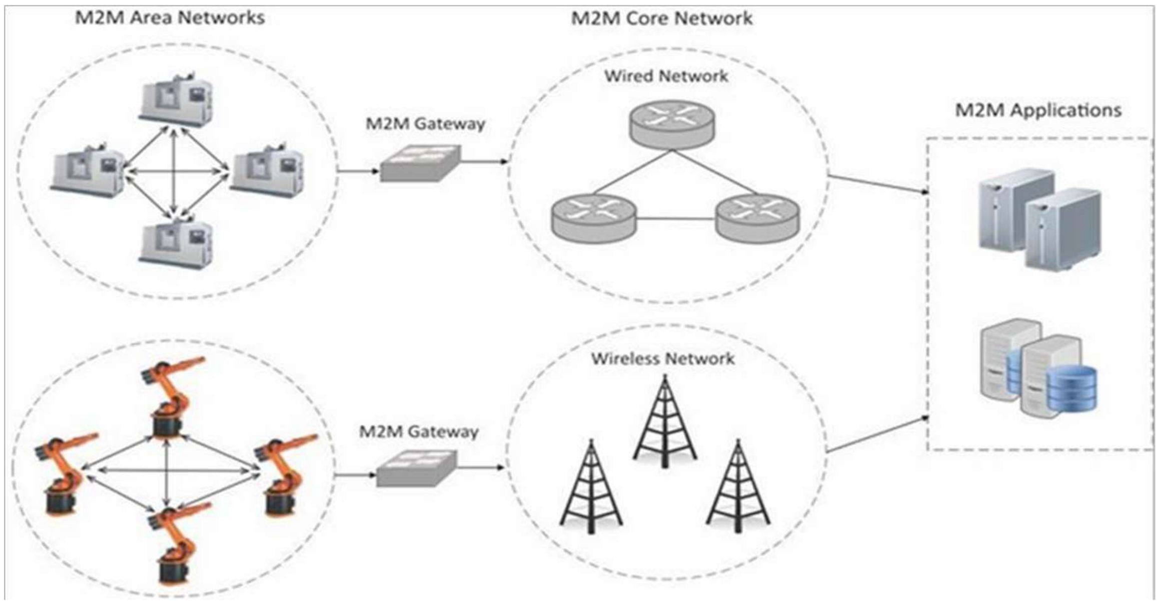
IoT and M2M

M2M:

Machine-to-Machine (M2M) refers to networking of machines (or devices) for the purposeof remote monitoring and control and data exchange.

- Term which is often synonymous with IoT is Machine-to-Machine (M2M).
- IoT and M2M are often used interchangeably.

Fig. Shows the end-to-end architecture of M2M systems comprises of M2M area networks, communication networks and application fomain.



- An M2M area network comprises of machines (or M2M nodes) which have embedded network modules for sensing, actuation and communicating, various communication protocols can be used for M2M LAN such as ZigBee, Bluetooth, M-bus, Wireless M-Bus etc., These protocols provide connectivity between M2M nodes within an M2M area network.
- The communication network provides connectivity to remote M2M area networks. The communication network provides connectivity to remote M2M area network. The communication network can use either wired or wireless network (IP based). While the M2M are networks use either properietorary or non-IP based communication protocols, the communication network uses IP-based network. Since non-IP based protocols are used within M2M area network, the M2M nodes within one network cannot communicate with nodes in an external network.
- To enable the communication between remote M2M are network, M2M gateways are used.

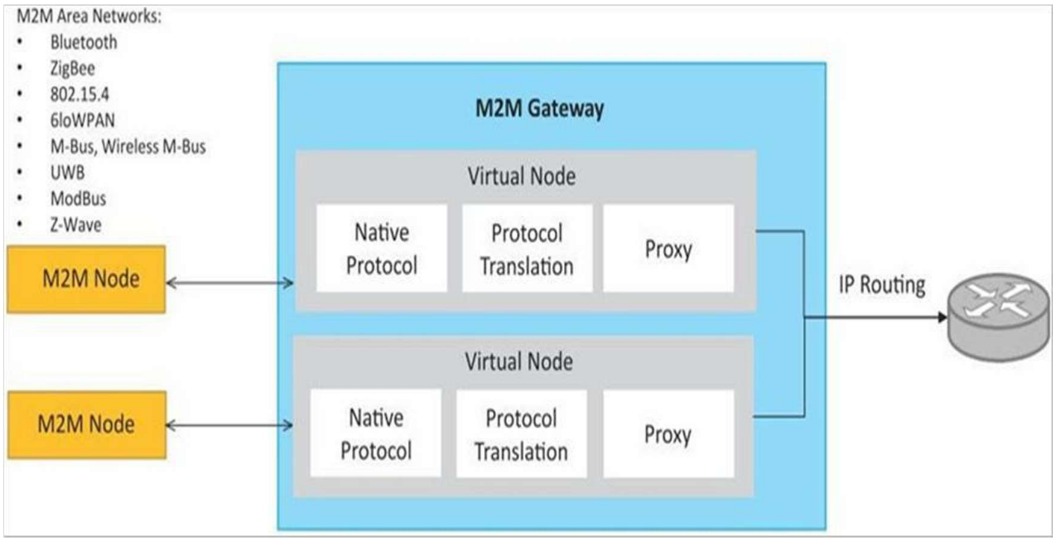


Fig. Shows a block diagram of an M2M gateway. The communication between M2M nodes and the M2M gateway is based on the communication protocols which are naive to the M2M are network. M2M gateway performs protocol translations to enable IP-connectivity for M2M are networks. M2M gateway acts as a proxy performing translations from/to native protocols to/fromInternet Protocol(IP). With an M2M gateway, each mode in an M2M area network appears as a virtualized node for external M2M area networks.

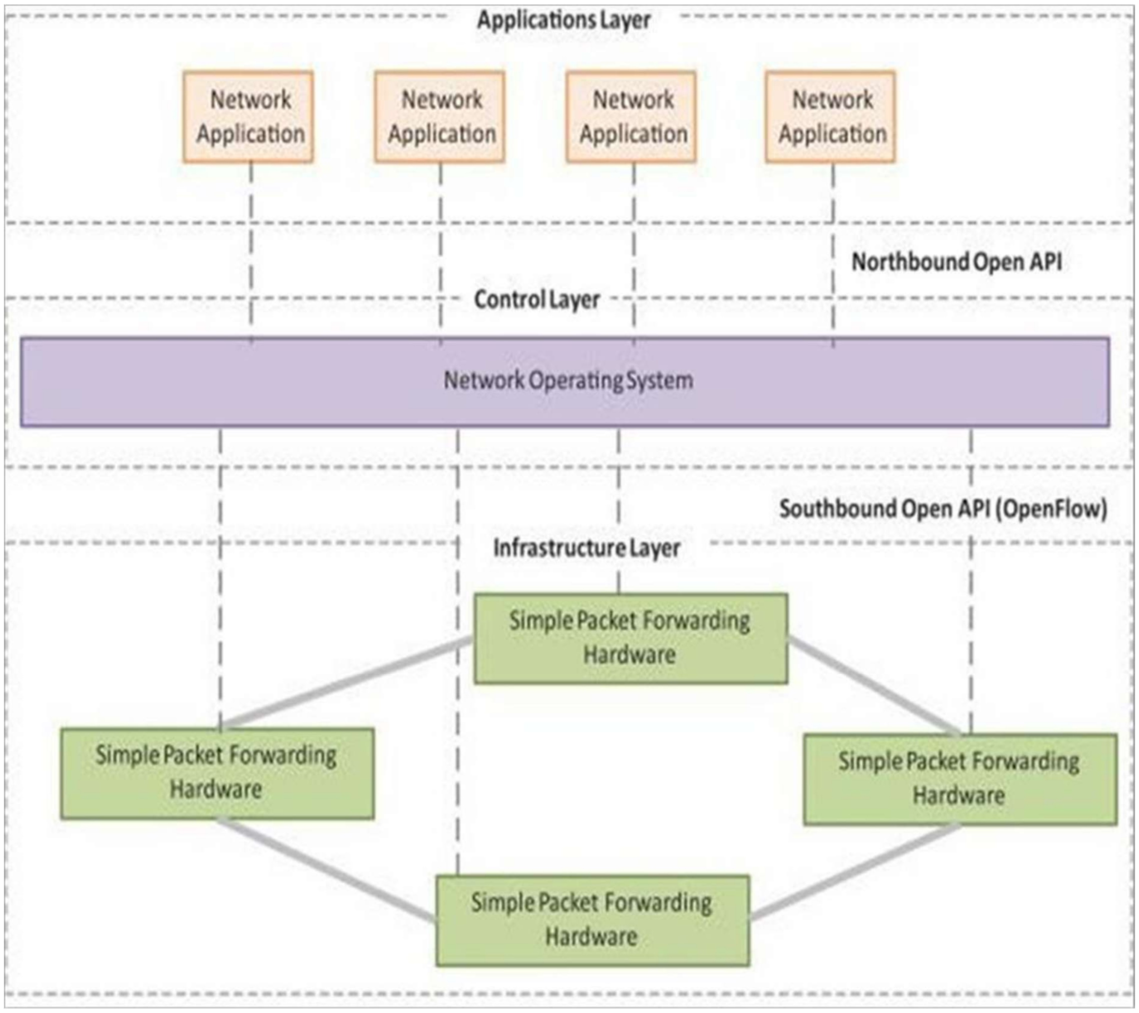
Differences between IoT and M2M

1) Communication Protocols:
    1. Commonly uses M2M protocols include ZigBee, Bluetooth, ModBus, M-Bus,Wireless M-Bustec.,
    2. In IoT uses HTTP, CoAP, WebSocket, MQTT,XMPP,DDS,AMQPetc.,
2) Machines in M2M Vs Things inIoT:
    1. Machines in M2M will be homogenous whereas Things in IoT will beheterogeneous.
3) Hardware Vs Software Emphasis:
    1. The emphasis of M2M is more on hardware with embedded modules, the emphasisof IoT is more on software.
4) Data Collection &Analysis
    1. M2M data is collected in point solutions and often in on-premises storageinfrastructure.
    2. The data in IoT is collected in the cloud (can be public, private orhybrid cloud).
5) Applications
    1. M2M data is collected in point solutions and can be accessed by on-premises applications such as diagnosis applications, service management applications, andon- premisis enterprise applications.
    2. IoT data is collected in the cloud and can be accessed by cloud applications such as analytics applications, enterprise applications, remote diagnosis and management applications,etc.

SDN IoT

Software Defined Networking (SDN):

- Software-Defined Networking (SDN) is a networking architecture that separates thecontrol plane from the data plane and centralizes the network controller.
- Software-based SDN controllers maintain a unified view of the network
- The underlying infrastructure in SDN uses simple packet forwarding hardware asopposed to specialized hardware in conventional networks.



SDN Architecture

Key elements of SDN:

1) **Centralized Network Controller**

   With decoupled control and data planes and centralized network controller, the network administrators can rapidly configure the network.

2) Programmable Open APIs

   SDN architecture supports programmable open APIs for interface between the SDN application and control layers (Northbound interface).

3) Standard Communication Interface (OpenFlow)

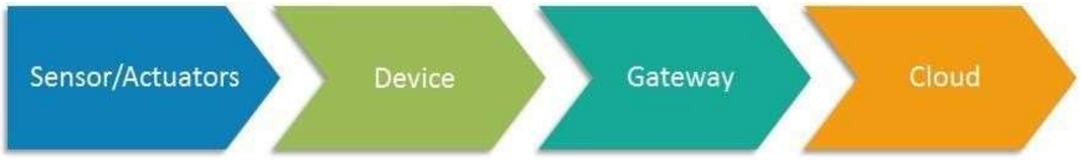
   SDN architecture uses a standard communication interface between the control and infrastructure layers (Southbound interface). Open Flow, which is defined by the Open Networking Foundation (ONF) is the broadly accepted SDN protocolfor the South bound interface.

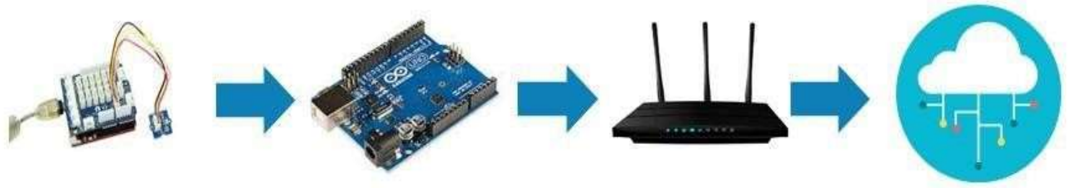
IOT ARCHITECTURE

State of the art

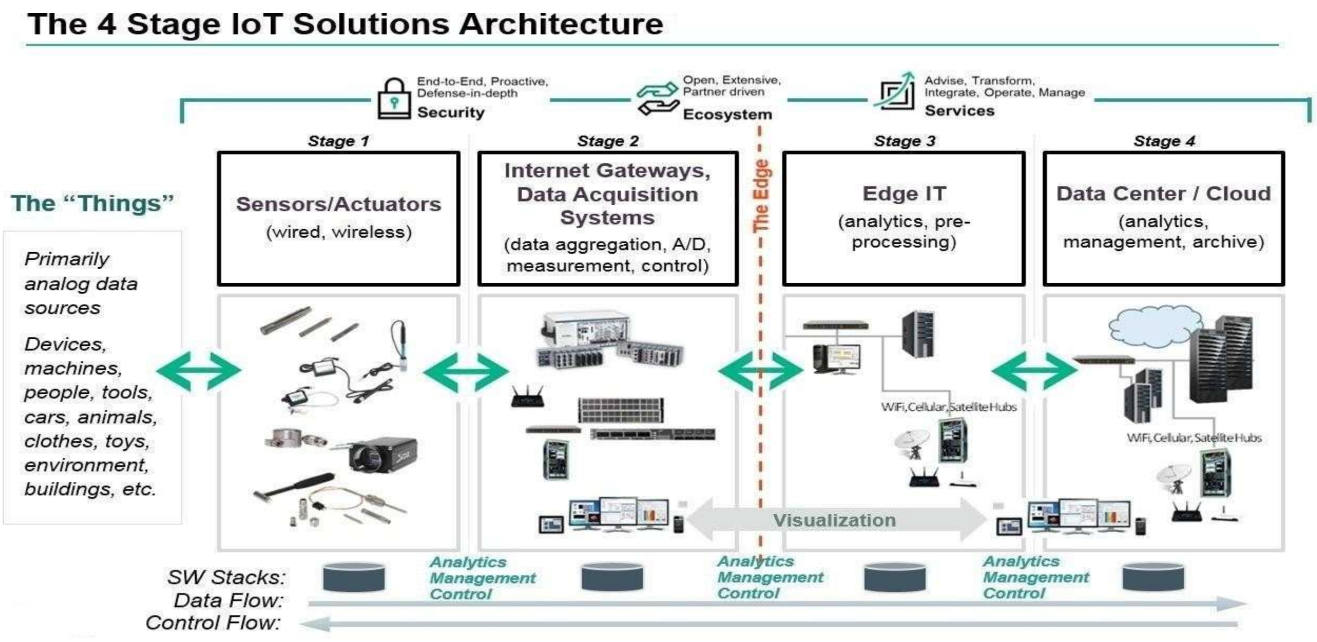
IoT architecture varies from solution to solution, based on the type of solution which we intend to build. IoT as a technology majorly consists of four main components, over which an architecture is framed.

1) Sensors
2) Devices
3) Gateway
4) Cloud





Stages of IoT Architecture


The 4 Stage IoT Solutions Architecture
End-to-End, Proactive, Defense-in-depth
Security
Open, Extensive, Partner driven
Ecosystem
Advise, Transform, Integrate, Operate, Manage
Services
Stage 1
Stage 2
Stage 3
Stage 4
The "Things"
Primarily analog data sources
Devices, machines, people, tools, cars, animals, clothes, toys, environment, buildings, etc.
Sensors/Actuators (wired, wireless)
Internet Gateways, Data Acquisition Systems (data aggregation, A/D, measurement, control)
The Edge
Edge IT (analytics, pre-processing)
Data Center / Cloud (analytics, management, archive)
WiFi, Cellular, Satellite Hubs
WiFi, Cellular, Satellite Hubs
Visualization
Analytics Management Control
Analytics Management Control
Analytics Management Control
SW Stacks:
Data Flow:
Control Flow:

Stage1:-
Sensors/actuators

Sensors collect data from the environment or object under measurement and turn it into useful data. Think of the specialized structures in your cell phone that detect the directional pull of gravity and the phone's relative position to the ―thing‖ we call the earth and convert it into data that your phone can use to orient the device.

Actuators can also intervene to change the physical conditions that generate the data. An actuator might, for example, shut off a power supply, adjust an air flow valve, or move a robotic gripper in an assembly process.

The sensing/actuating stage covers everything from legacy industrial devices to robotic camera systems, water level detectors, air quality sensors, accelerometers, and heart rate monitors. And the scope of the IoT is expanding rapidly, thanks in part to low-power wireless sensor network technologies and Power over Ethernet, which enable devices on a wired LAN to operate without the need for an A/C power source.

Stage 2:-
The Internet gateway

The data from the sensors starts in analog form. That data needs to be aggregated and converted into digital streams for further processing downstream. Data acquisition systems(DAS) perform these data aggregation and conversion functions. The DAS connects to the sensor network, aggregates outputs, and performs the analog-to-digital conversion. The Internet gateway receives the aggregated and digitized data and routes it over Wi-Fi, wired LANs, or the Internet, to Stage 3 systems for further processing. Stage 2 systems often sit in close proximity tothe sensors and actuators.

For example, a pump might contain a half-dozen sensors and actuators that feed data into a data aggregation device that also digitizes the data. This device might be physically attached to the pump. An adjacent gateway device or server would then process the data and forward it to the Stage 3 or Stage 4 systems. Intelligent gateways can build on additional, basic gateway functionality by adding such capabilities as analytics, malware protection, and data management services. These systems enable the analysis of data streams in real time.

Stage 3:-
Edge IT

Once IoT data has been digitized and aggregated, it's ready to cross into the realm of IT. However, the data may require further processing before it enters the data center. This is where edge IT systems, which perform more analysis, come into play. Edge IT processing systems maybe located in remote offices or other edge locations, but generally these sit in the facility or location where the sensors reside closer to the sensors, such as in a wiring closet. Because IoT data can easily eat up network bandwidth and swamp your data center resources, it's best to have systems at the edge capable of performing analytics as a way to lessen the burden on core IT infrastructure. You'd also face security concerns, storage issues, and delays processing the data. With a staged approach, you can preprocess the data, generate meaningful results, and pass only those on. For example, rather than passing on raw vibration data for the pumps, you could aggregate and convert the data, analyze it, and send only projections as to when each device will fail or need service.

Stage 4:-
The data center and cloud

Data that needs more in-depth processing, and where feedback doesn't have to be immediate, gets forwarded to physical data center or cloud-based systems, where more powerful IT systems can analyze, manage, and securely store the data. It takes longer to get results when you wait until data reaches Stage 4, but you can execute a more in-depth analysis, as well as combine your sensor data with data from other sources for deeper insights. Stage 4 processing may take place on-premises, in the cloud, or in a hybrid cloud system, but the type of processing executed in this stage remains the same, regardless of the platform.

REFERENCE MODEL AND ARCHITECTURE

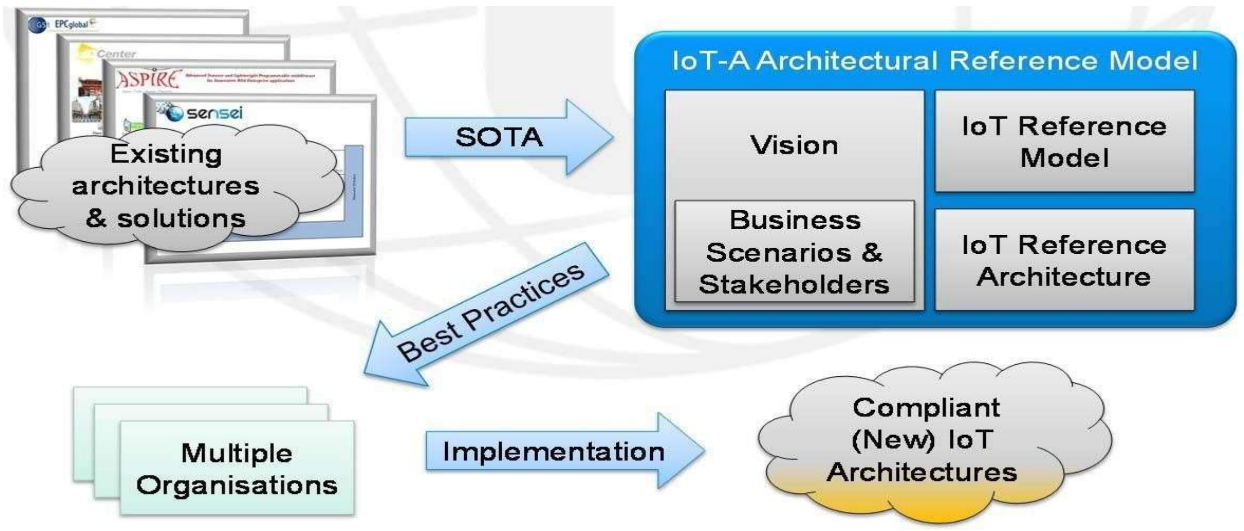
Reference Architecture that describes essential building blocks as well as design choices to deal with conflicting requirements regarding functionality, performance, deployment and security. Interfaces should be standardised, best practices in terms of functionality and information usage need to be provided.

The central choice of the IoT-A project was to base its work on the current state of the art, rather than using a clean-slate approach. Due to this choice, common traits are derived to form the base line of the Architectural Reference Model (ARM). This has the major advantage of ensuring backward compatibility of the model and also the adoption of established, working solutions to various aspects of the IoT. With the help of end users, organised into a stakeholders group, new requirements for IoT have been collected and introduced in the main model building process. This work was conducted according to established architecture methodology.

A*Reference Architecture (RA) can be visualised as* the –Matrix ‖that eventually gives birth ideally to all concrete architectures. For establishing such a Matrix, based on a strong and exhaustive analysis of the State of the Art, we need to envisage the superset of all possible functionalities, mechanisms and protocols that can be used for building such concrete architecture and to show how interconnections could take place between selected ones (as no concrete system is likely to use all of the functional possibilities). Giving such a foundationalong with a set of design-choices, based on the characterisation of the targeted system w.r.t. various dimensions (like distribution, security, real-time, semantics) it becomes possible for a system architect to select the protocols, functional components, architectural options, needed to build their IoT systems.

As any metaphoric representation, this tree does not claim to be fully consistent in its depiction; it should therefore not be interpreted too strictly. On the one hand, the roots of this tree are spanning across a selected set of communication protocols (6LoWPAN, Zigbee, IPv6,…) and device technologies (sensors, actuators, tags,..) while on the other hand the blossoms / leaves of the tree represent the whole set of IoT applications that can be built fromthe sap (i.e., data and information) coming from the roots. The trunk of the tree is of utmost importance here, as it represent the Architectural Reference Model (ARM). The ARM is the combination of the Reference Model and the Reference Architecture, the set of models, guidelines, best practices, views and perspectives that can be used for building fully interoperable concrete IoT architectures and systems. In this tree, we aim at selecting a minimal set of interoperable technologies (the roots) and proposing the potentially necessary set of enablers or building blocks (the trunk) that enable the creation of a maximal set of interoperable IoT systems (the leaves).

IoT-A architectural reference model building blocks.


EPCglobal
Center
ASPIRE
sensei
Existing architectures & solutions
SOTA
IoT-A Architectural Reference Model
Vision
IoT Reference Model
Business Scenarios & Stakeholders
IoT Reference Architecture
Best Practices
Multiple Organisations
Implementation
Compliant (New) IoT Architectures

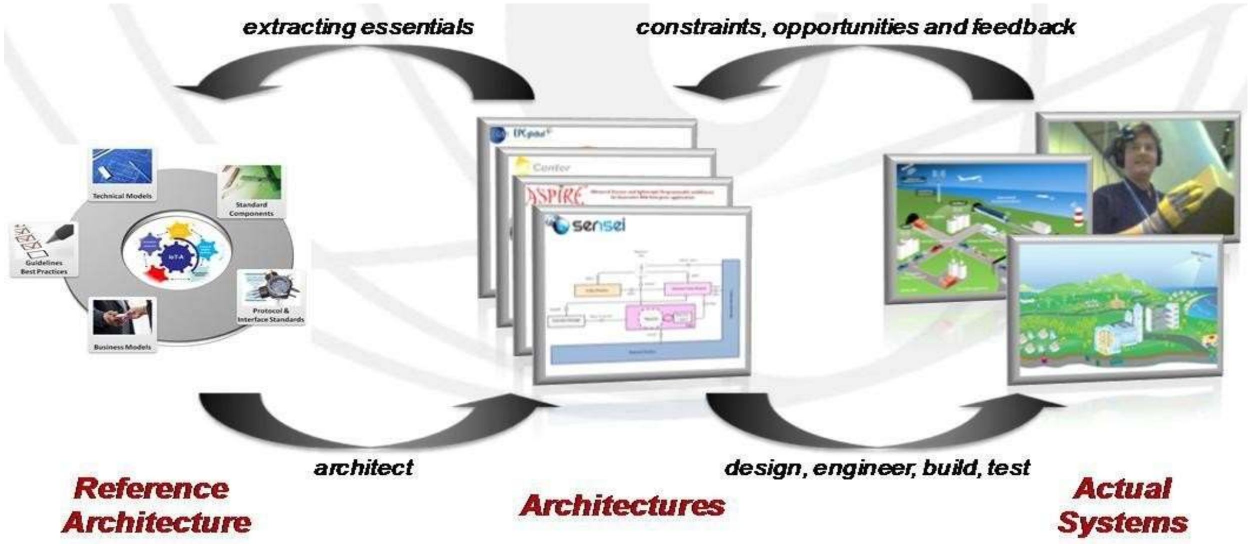
Starting with existing architectures and solutions, generic baseline requirements can be extracted and used as an input to the design. The IoT-A ARM consists of four parts:

The vision summarises the rationale for providing an architectural reference model for the IoT. At the same time it discusses underlying assumptions, such as motivations. It also discusses how the architectural reference model can be used, the methodology applied to the architecture modelling, and the business scenarios and stakeholders addressed.

Business scenarios defined as requirements by stakeholders are the drivers of the architecture work. With the knowledge of businesses aspirations, a holistic view of IoT architectures can be derived.

The IoT Reference Model provides the highest abstraction level for the definition of the IoT-A Architectural Reference Model. It promotes a common understanding of the IoT domain. The description of the IoT Reference Model includes a general discourse on the IoT domain, an IoT Domain Model as a top-level description, an IoT Information Model explaining how IoT information is going to be modelled, and an IoT Communication Model in order to understand specifics about communication between many heterogeneous IoT devices and the Internet as a whole.

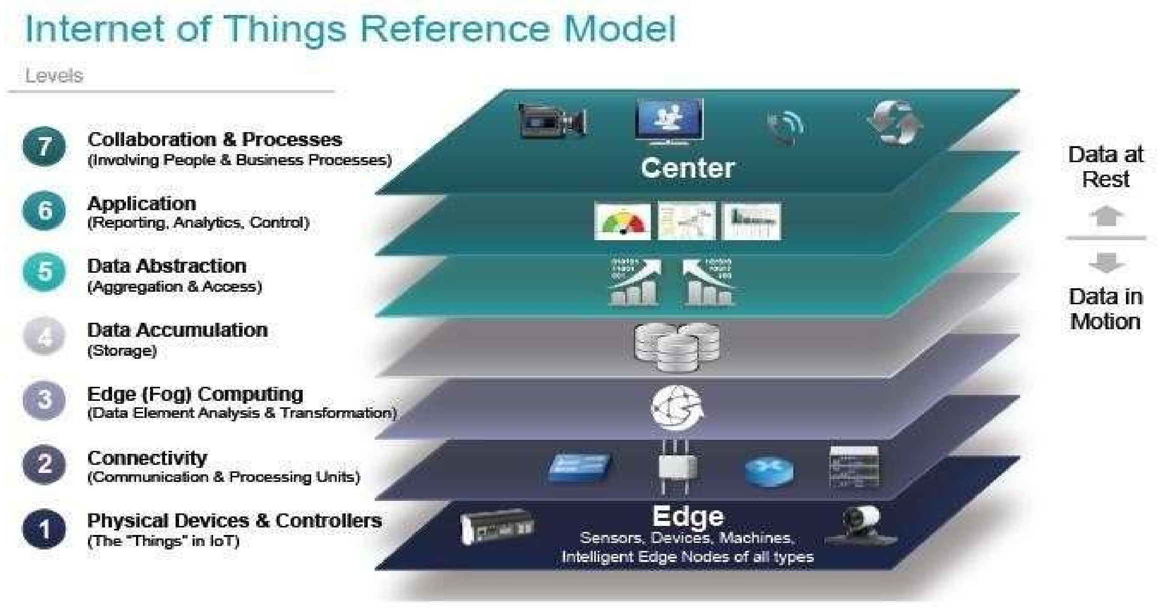
The IoT Reference Architecture is the reference for building compliant IoT architectures. As such, it provides views and perspectives on different architectural aspects that are of concern to stakeholders of the IoT. The terms view and perspectives are used according to the general literature and standards the creation of the IoT Reference Architecture focuses on abstract sets of mechanisms rather than concrete application architectures. To organisations, an important aspect is the compliance of their technologies with standards and best practices, so that interoperability across organisations isensured.



In an IoT system, data is generated by multiple kinds of devices, processed in different ways, transmitted to different locations, and acted upon by applications. The proposed IoT reference model is comprised of seven levels. Each level is defined with terminology that can be standardized to create a globally accepted frame of reference.

Simplifies: It helps break down complex systems so that each part is moreunderstandable. **Clarifies**: It provides additional information to precisely identify levelsof the IoT and to establish common terminology.

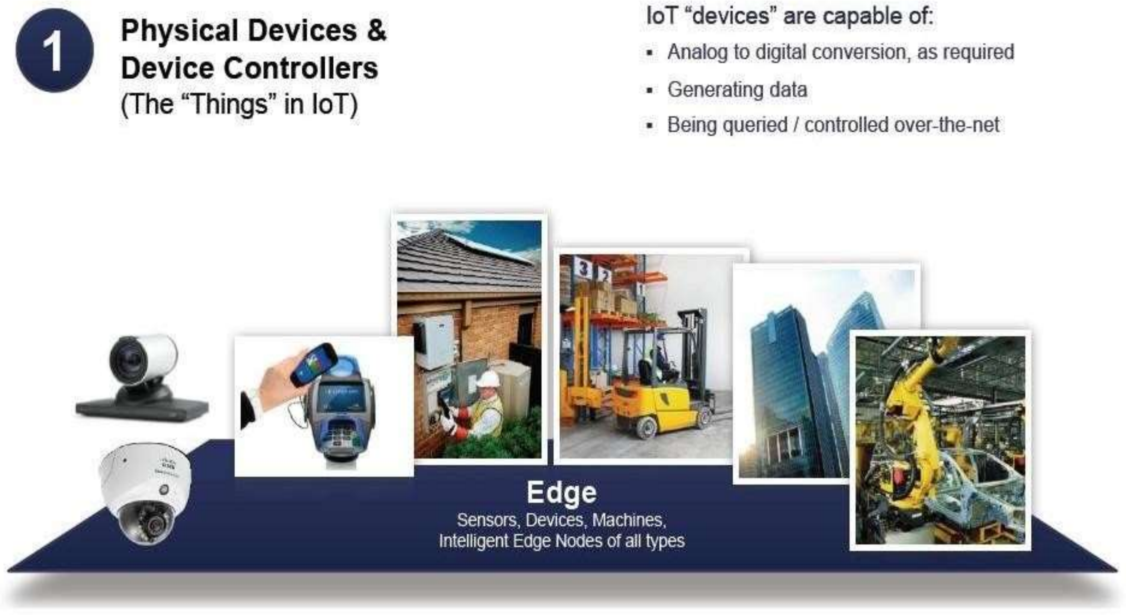
- ✓ **Identifies**: It identifies where specific types of processing is optimized across different parts of the system.
- ✓ **Standardizes**: It provides a first step in enabling vendors to create IoT products that work with each other.
- ✓ **Organizes**: It makes the IoT real and approachable, instead of simply conceptual.


Internet of Things Reference Model
Levels
7 Collaboration & Processes (Involving People & Business Processes)
6 Application (Reporting, Analytics, Control)
5 Data Abstraction (Aggregation & Access)
4 Data Accumulation (Storage)
3 Edge (Fog) Computing (Data Element Analysis & Transformation)
2 Connectivity (Communication & Processing Units)
1 Physical Devices & Controllers (The "Things" in IoT)
Center
Edge
Sensors, Devices, Machines, Intelligent Edge Nodes of all types
Data at Rest
Data in Motion

Level 1: Physical Devices and Controllers

The IoT Reference Model starts with Level 1: physical devices and controllers that might control multiple devices.These are the ―things in the IoT, and they include a wide range of endpoint devices that send and receive information. Today, the list of devices is already extensive.

It will become almost unlimited as more equipment is added to the IoT over time. Devices are diverse, and there are no rules about size, location, form factor, or origin. Some devices will be the size of a silicon chip. Some will be as large as vehicles. The IoT must supportthe entire range. Dozens or hundreds of equipment manufacturers will produce IoT devices. To simplify compatibility and support manufacturability, the IoT Reference Model generally describes the level of processing needed from Level 1devices.


1
Physical Devices & Device Controllers
(The "Things" in IoT)
IoT "devices" are capable of:
Analog to digital conversion, as required
Generating data
Being queried / controlled over-the-net
Edge
Sensors, Devices, Machines, Intelligent Edge Nodes of all types

Level 2: Connectivity

Communications and connectivity are concentrated in one level—Level 2. The most important function of Level 2 is reliable, timely information transmission. This includes transmissions:

● Between devices (Level 1) and the network
● Across networks(east-west)
● Between the network (Level 2) and low-level information processing occurring at Level 3

Traditionaldatacommunicationnetworkshavemultiplefunctions,asevidencedbythe International Organization for Standardization (ISO) 7-layer reference model. However, a complete IoT system contains many levels in addition to the communications network. One objective of the IoT Reference Model is for communications and processing to be executed by existing networks. The IoT Reference Model does not require or indicate creation of a different network—it relies on existing networks. As Level 1 devices proliferate, the ways in which they interact with Level 2 connectivity equipment may change. Regardless of the details, Level 1 devices communicate through the IoT system by interacting with Level 2 connectivity equipment.

2 **Connectivity**
(Communication & Processing Units)

Level 2 functionality focuses on East-West communications

Connectivity includes:

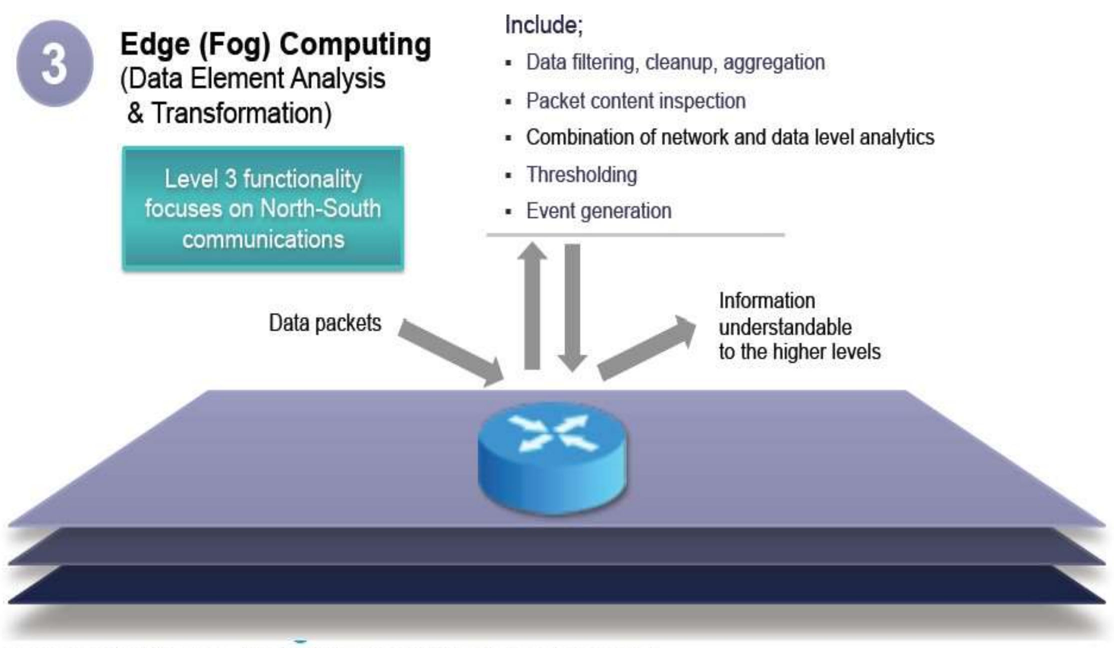
- Communicating with and between the Level 1 devices
- Reliable delivery across the network(s)
- Implementation of various protocols
- Switching and routing
- Translation between protocols
- Security at the network level
- (Self Learning) Networking Analytics

Level 3: Edge (Fog) Computing

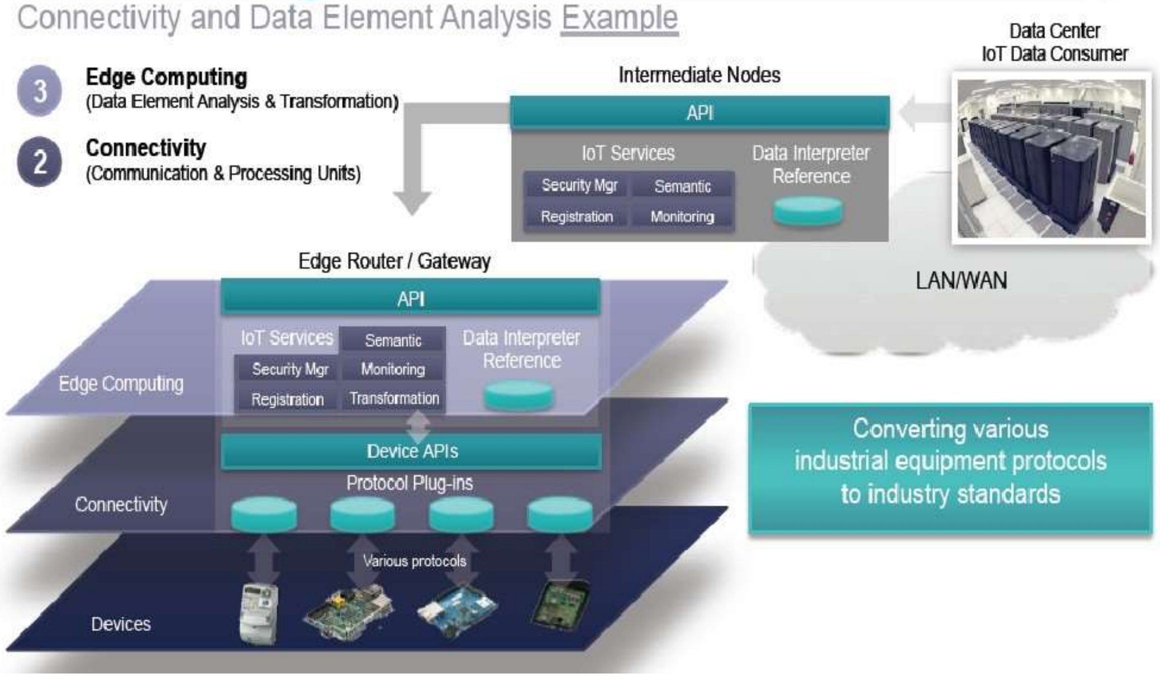
The functions of Level 3 are driven by the need to convert network data flows into information that is suitable for storage and higher level processing at Level 4 (data accumulation). This means that Level 3 activities focus on high-volume data analysis and transformation. For example, a Level 1 sensor device might generate data samples multiple times per second, 24 hours a day, 365 days a year. A basic tenet of the IoT Reference Model is that the most intelligent system initiates information processing as early and as close to the edge of the network as possible. This is sometimes referred to as fog computing. Level 3 is where this occurs.

Given that data is usually submitted to the connectivity level (Level 2) networking equipment by devices in small units, Level 3 processing is performed on a packet-by-packet basis. This processing is limited, because there is only awareness of data units—not “sessions” or “transactions.” Level 3 processing can encompass many examples, such as:

- Evaluation: Evaluating data for criteria as to whether it should be processed at a higher level
- Formatting: Reformatting data for consistent higher-level processing
- Expanding/decoding: Handling cryptic data with additional context (such as the origin)
- Distillation/reduction: Reducing and/or summarizing data to minimize the impact of data and traffic on the network and higher-level processing systems
- Assessment: Determining whether data represents a threshold or alert; this could include redirecting data to additional destinations


3
Edge (Fog) Computing
(Data Element Analysis & Transformation)
Level 3 functionality focuses on North-South communications
Include;
▪ Data filtering, cleanup, aggregation
▪ Packet content inspection
▪ Combination of network and data level analytics
▪ Thresholding
▪ Event generation
Data packets
Information understandable to the higher levels

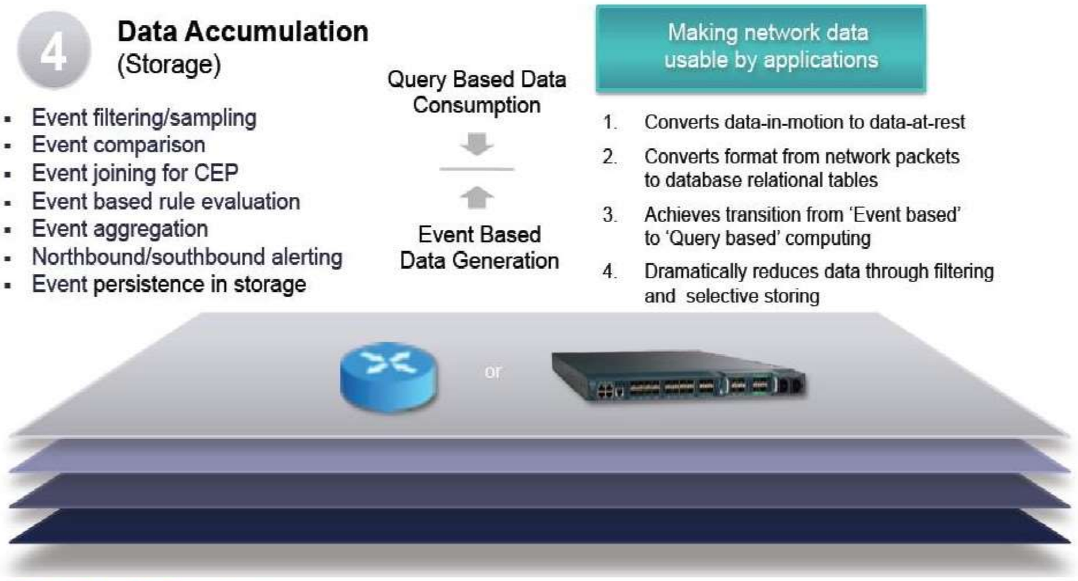
Connectivity and Data Element Analysis Example



Level 4: Data Accumulation

Networking systems are built to reliably move data. The data is "in motion." Prior to Level 4, data is moving through the network at the rate and organization determined by the devices generating the data. The model is event driven. As defined earlier, Level 1 devices do not include computing capabilities themselves. However, some computational activities could occur at Level 2, such as protocol translation or application of network security policy. Additional compute tasks can be performed at Level 3, such as packet inspection. Driving computational tasks as close to the edge of the IoT as possible, with heterogeneous systems distributed across multiple management domains represents an example of fog computing. Fog computing and fog services will be a distinguishing characteristic of the IoT.

As Level 4 captures data and puts it at rest, it is now usable by applications on a non-real-time basis. Applications access the data when necessary. In short, Level 4 converts event-based data to query-based processing. This is a crucial step in bridging the differences between the real-time networking world and the non-real-time application world. Figure 6 summarizes the activities that occur at Level 4.


4
Data Accumulation
(Storage)
Event filtering/sampling
Event comparison
Event joining for CEP
Event based rule evaluation
Event aggregation
Northbound/southbound alerting
Event persistence in storage
Query Based Data Consumption
Event Based Data Generation
Making network data usable by applications
1. Converts data-in-motion to data-at-rest
2. Converts format from network packets to database relational tables
3. Achieves transition from 'Event based' to 'Query based' computing
4. Dramatically reduces data through filtering and selective storing
or

Level 5: Data Abstraction

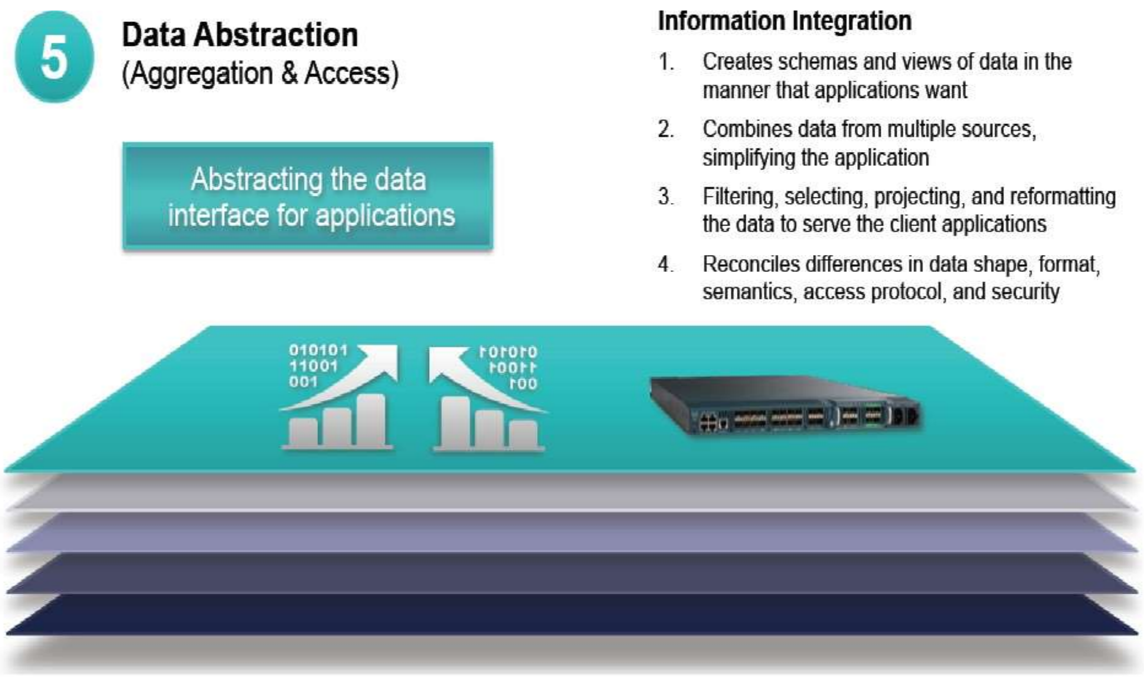
IoT systems will need to scale to a corporate—or even global—level and will require multiple storage systems to accommodate IoT device data and data from traditional enterprise ERP, HRMS, CRM, and other systems. The data abstraction functions of Level 5 are focused on rendering data and its storage in ways that enable developing simpler, performance-enhanced applications.

With multiple devices generating data, there are many reasons why this data may not land in the same data storage:

- There might be too much data to put in one place.
- Moving data into a database might consume too much processing power, so that retrieving it must be separated from the data generation process. This is done today with online transaction processing (OLTP) databases and data warehouses.
- Devices might be geographically separated, and processing is optimized locally.
- Levels 3 and 4 might separate "continuous streams of raw data" from "data that represents an event." Data storage for streaming data may be a big data system, such as Hadoop. Storage for event data may be a relational database management system (RDBMS) with faster query times.
- Different kinds of data processing might be required. For example, in-store processing will focus on different things than across-all-stores summary processing.

For these reasons, the data abstraction level must process many different things. These include:

- Reconciling multiple data formats from different sources
- Assuring consistent semantics of data across sources
- Confirming that data is complete to the higher-level application



5 Data Abstraction
(Aggregation & Access)

Abstracting the data interface for applications

Information Integration

1. Creates schemas and views of data in the manner that applications want
2. Combines data from multiple sources, simplifying the application
3. Filtering, selecting, projecting, and reformatting the data to serve the client applications
4. Reconciles differences in data shape, format, semantics, access protocol, and security

Level 6: Application

Level 6 is the application level, where information interpretation occurs. Software at this level interacts with Level 5 and data at rest, so it does not have to operate at network speeds.

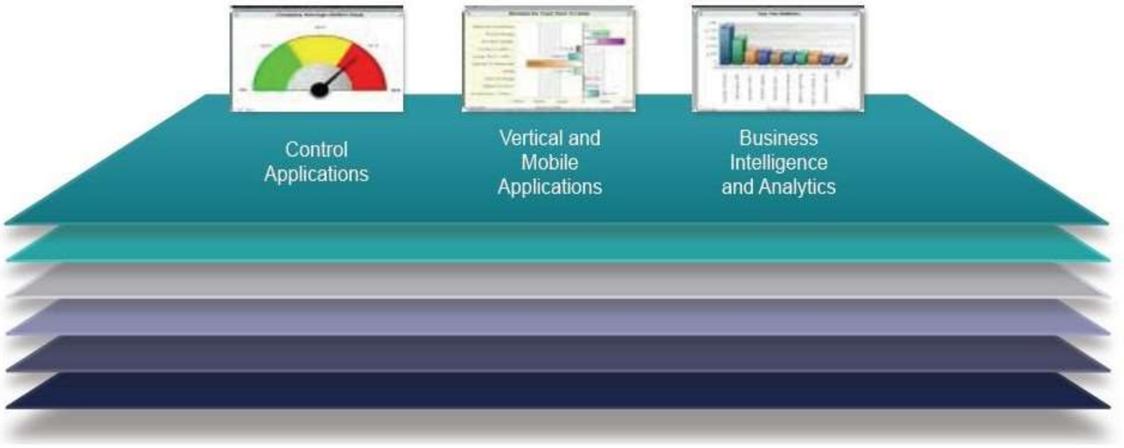
The IoT Reference Model does not strictly define an application. Applications vary based on vertical markets, the nature of device data, and business needs. For example, some applications will focus on monitoring device data. Some will focus on controlling devices. Some will combine device and non-device data. Monitoring and control applications represent many different application models, programming patterns, and software stacks, leading to discussions of operating systems, mobility, application servers, hypervisors, multi-threading, multi-tenancy, etc. These topics are beyond the scope of the IoT Reference Model discussion. Suffice it to say that application complexity will vary widely.

Examples include:

- Mission-critical business applications, such as generalized ERP or specialized industry solutions
- Mobile applications that handle simple interactions
- Business intelligence reports, where the application is the BI server
- Analytic applications that interpret data for business decisions

- System management/control center applications that control the IoT system itself and don't act on the data produced by it

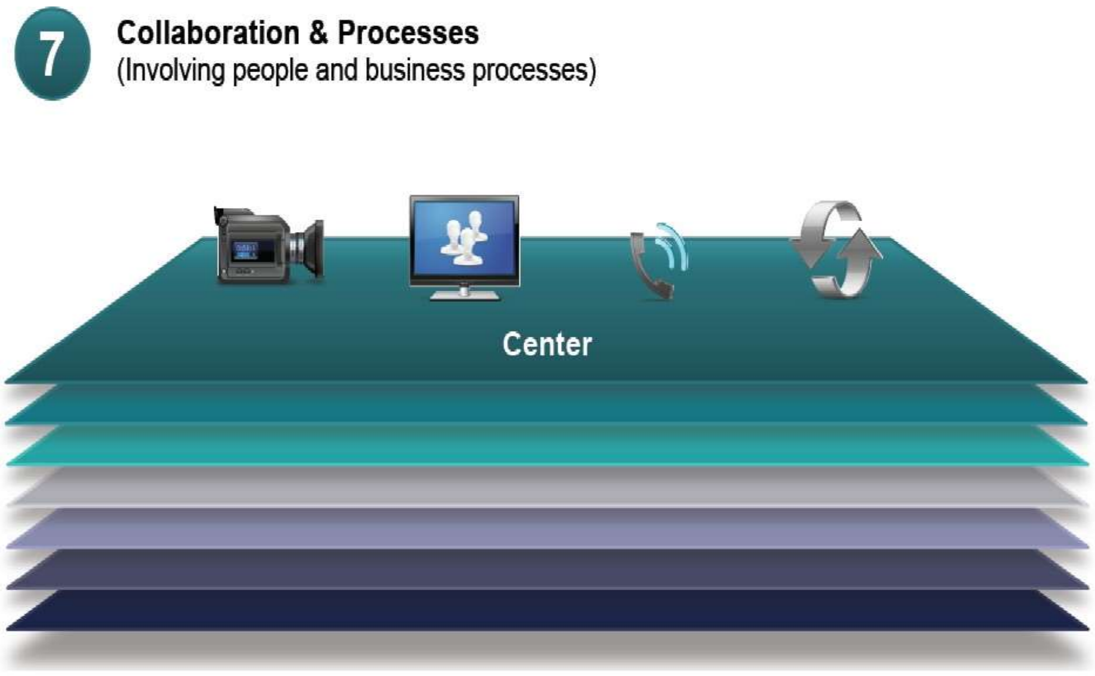
6 **Application**
(Reporting, Analytics, Control)


Control Applications
Vertical and Mobile Applications
Business Intelligence and Analytics

Level 7: Collaboration and Processes

One of the main distinctions between the Internet of Things (IoT) and IoT is that IoT includes people and processes. This difference becomes particularly clear at Level 7: Collaboration and Processes. The IoT system, and the information it creates, is of little value unless it yields action, which often requires people and processes. Applications execute business logic to empower people. People use applications and associated data for their specific needs. Often, multiple people use the same application for a range of different purposes. So the objective is not the application—it is to empower people to do their work better. Applications (Level 6) give business people the right data, at the right time, so they can do the right thing.

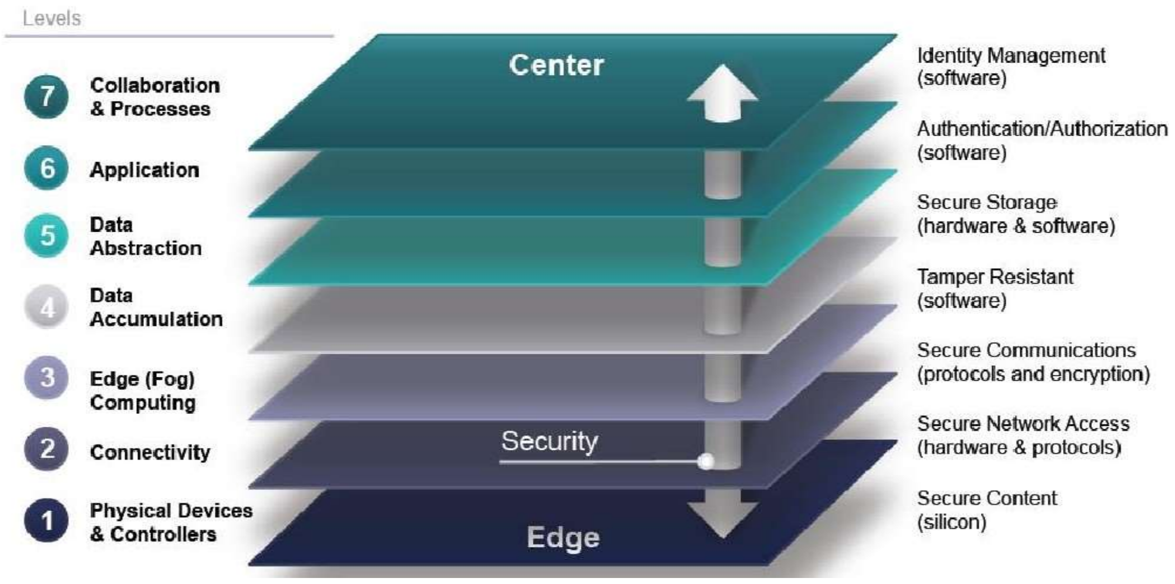
But frequently, the action needed requires more than one person. People must be able to communicate and collaborate, sometimes using the traditional Internet, to make the IoT useful. Communication and collaboration often requires multiple steps. And it usually transcends multiple applications. This is why Level 7, as shown in


7
Collaboration & Processes
(Involving people and business processes)
Center

Security in the IoT

Discussions of security for each level and for the movement of data between levels could fill a multitude of papers. For the purpose of the IoT Reference Model, security measures must:

- Secure each device or system
- Provide security for all processes at each level
- Secure movement and communication between each level, whether north- or south-bound


Levels
7 Collaboration & Processes
6 Application
5 Data Abstraction
4 Data Accumulation
3 Edge (Fog) Computing
2 Connectivity
1 Physical Devices & Controllers
Center
Security
Edge
Identity Management (software)
Authentication/Authorization (software)
Secure Storage (hardware & software)
Tamper Resistant (software)
Secure Communications (protocols and encryption)
Secure Network Access (hardware & protocols)
Secure Content (silicon)

Everything as a Service (XaaS) :

In cloud computing technology, cloud service providers were providing various cloud services to the customers. But now a new concept has emerged i.e Everything as a Service (XaaS) means anything can now be a service with the help of cloud computing and remote accessing. Where cloud computing technologies provide different kinds of services over the web networks. In Everything as a Service, various tools and technologies, and services are provided to users as a service. Before XaaS and cloud services, companies have to buy licensed products and install them, had to all securities on their site and provide infrastructure for business purposes. With XaaS, business is simplified as they have to pay for what they need. This Everything as a Service is also known as Anything as a Service.

Examples of XaaS :
As XaaS stands for "Everything as a service", There are many examples. There are many varieties of cloud computing models like –
1. Software as a Service (SaaS)
2. Platform as a Service (PaaS)
3. Disaster Recovery as a Service (DRaaS)
4. Infrastructure as a service (IaaS)
5. Communication as a Service (CaaS)
6. Network as a Service (NaaS)
7. Database as a Service (DBaaS)
8. Desktop as a Service (DaaS) etc.

SaaS provides many software applications like Google Apps, and Microsoft Office 365. Similarly, PaaS offers AWS, Heroku, Apache Stratos, and other sources relating to application development and testing. IaaS helps to deploy and configure virtual machines and manage these remotely. IaaS also provide services to Azure and Google Computer Engine.

Everything as a Service Model Examples
1. **Hardware as a Service (HaaS)**
   Managed Service Providers (MSP) provide and install some hardware on the customer's site on demand. The customer uses the hardware according to service level agreements. This model is very similar to IaaS as computing resources present at MSP's site are provided to users substituted for physical hardware.
2. **Communication as a Service (CaaS)**
   This model comprises solutions for different communication like IM, VoIP, and video conferencing applications which are hosted in the provider's cloud. Such a method is cost-effective and reduces time expenses.
3. **Desktop as a Service (DaaS) –**
   DaaS provider mainly manages storing, security, and backing up user data for desktop apps. And a client can also work on PCs using third-party servers.
4. **Security as a Service (SECaaS) –**
   In this method, the provider integrates security services with the company's

infrastructure through the internet which includes anti-virus software, authentication, encryption, etc.

5. **Healthcare as a Service (HaaS) –**
   The healthcare industry has opted for the model HaaS service through electronic medical records (EMR). IoT and other technologies have enhanced medical services like online consultations, health monitoring 24/7, medical service at the doorstep e.g. lab sample collection from home, etc.
6. **Transport as a Service (TaaS) –**
   Nowadays, there are numerous apps that help in mobility and transport in modern society. The model is both convenient and ecological friendly e.g. Uber taxi services is planning to test flying taxis and self-driving planes in the future.

Benefits in XaaS :

- **Cost Saving –**
  When an organization uses XaaS then it helps in cost-cutting and simplifies IT deployments.
- **Scalability –**
  XaaS can easily handle the growing amount of work by providing the required resources/service.
- **Accessibility –**
  It helps in easy accessing and improving accessibility as long as the internet connection is there.
- **Faster Implementation –**
  It provides faster implementation time to various activities of the organization.
- **Quick Modification –**
  It provides updates for modification as well as undergoes quick updating by providing quality services.
- **Better Security –**
  It contains improved security controls and is configured to the exact requirements of the business.
- **Boost innovation –**
  While XaaS is used it Streamlines the operations and frees up resources for innovation.
- **Flexibility –**
  XaaS provides flexibility by using cloud services and multiple advanced approaches.

Disadvantages in XaaS :

- **Internet Breakage –**
  Internet breaks sometimes for XaaS service providers where there can also be issues in internet reliability, provisioning, and managing the infrastructure resources.
- **Slowdown –**
  When too many clients are using the same resources at the same time, the system can slow down.

- **Difficult in Troubleshoot –**
  XaaS can be a solution for IT staff in day-to-day operational headaches, but if anywhere problem occurs it is harder to troubleshoot it as in XaaS multiple services are included with various technologies and tools.
- **Change brings problems –**
  If a XaaS provider discontinues a service or alters it gives an impact on XaaS users.

XaaS is on the rise now :
Public cloud services are growing at an exponential rate. Researchers assumed that global cloud computing revenue is going to reach $342 billion dollars by 2025. Through the XaaS model by servitization, products and services are combined through which businesses innovate faster and enhance their relationship with customers which further increases their revenue.
Future of XaaS :
A combination of cloud computing and good internet access allows accessing good quality XaaS services and better improvement of XaaS. Some companies are not confident to take XaaS because of security and business governance concerns. But service providers increasingly reveal these concerns which allow organizations which put additional workloads into the cloud.

Rise Of Things: IoT's Role In Business Processes

Consider these three key ways businesses can improve and transform their operations with Internet of things technologies.

The perfect digital storm of mobile, social, cloud, analytics and connected devices (a.k.a. "Things") is transforming businesses, governments, and consumers at an alarming rate. The societal impacts will be felt for generations to come and it's my firm belief that businesses will determine their own fate by either leveraging or ignoring this revolution.

But collaboration and orchestration between humans and "Things" are necessary to achieve this tremendous potential. Here's why.

Things -- including connected objects from smartphones with rich sensor capabilities to robots in homes, manufacturing plants and businesses -- are becoming a tour de force in digitization. A 2013 study by The Economist found that three quarters of businesses are interested in IoT; a staggering 96% plan to incorporate Things into their portfolio within the next three years.

[If IoT is going to work, networks must grant access to devices we'd refuse today. Read Internet Of Things Will Turn Networks Inside-Out]

There are huge opportunities to make Things better with the proliferation of IoT homes, cars (telematics), cities, healthcare, and manufacturing.

Here are three key ways businesses can transform their operations with IoT technologies:

1. Elevate the focus from technology to processes
The real digital transformation of IoT will happen through digital processes. Cisco (which characterizes IoT as Internet of Everything), believes people, data, and processes are essential components. A key requirement for the success of IoT is the end-to-end digitization of processes. What do I mean by processes? A process has (a) inputs; (b) execution of tasks; and (c) business outcome upon completion of the tasks.

Traditionally, process automation was coordinated around humans, business partners, or enterprise applications. That landscape is changing with the rise of Things. Take, for example, a complex and dynamic digital process: the coordination of the arrival and departure of an airplane in a busy airport. The business outcome is the timely departure. The participants include airport staff (baggage, refueling, catering) as well as Things that can be queried for sensor data or asked to carry out tasks. Autonomous or semi-autonomous Things are becoming active participants in *business* processes.

So IT in increasingly digital enterprises needs to provide not only the technology infrastructure for Things, but also a business process automation platform for making Things part of *business* outcomes. Thing tasks can be as simple as responding with a sensor value (e.g. what is the cabin temperature?) or as complex as how to circumnavigate a fast-moving storm.

2. Handling crisis events and digitizing change
How do these processes with Things get manifested? The airport example above illustrates a "happy path" coordinating humans and Things. However, one of the most pervasive use cases for Things is sensing (through IoT sensors) a crisis event and then activating a digitized end-to-end process to respond to it. This happens when there is a vehicle accident, boiler explosion, security alarm, or elevated blood pressure. The Thing autonomously senses and then either directly, or through a brokering layer, activates an *exception* process. This typically includes monitoring back-office and field workers to respond and resolve the problem.

3. Thing data analytics
Often, it's not merely an individual event that starts a process. Big data will eventually become "Thing Data." Through our connected homes, connected cities, and industries (such as power plants), Things are generating enormous amounts of data. Visualization of the data and analytics can be applied to streams of sensor data that are then handled via automated processes involving humans and Things. In a connected city that may have hundreds of thousands of sensors on city infrastructures, this could be applied for transportation, pollution sensing, or power grids.

For example, in a connected city application, multiple sensors could be monitoring pollution levels in the air or water. Then, if the critical levels are reached from this analysis of "thing data" over a period of time, exception handling processes would kick in. The difference from what I describe in the No. 2 point is that the exception here is detected and analyzed from multiple sources and typically over a period of time versus a single event. Both are important use cases in dealing with a situation through a digitized automated process.

While some dismiss it as hype, the pervasiveness of the Internet of Things is unquestionable. Estimates show there will be 25 billion to 1 trillion connected Things by 2020.

These all must be coordinated with people and applications within the enterprise through business processes. After all, isolated things have little value.

The Internet of Things demands reliable connectivity, but standards remain up in the air. Here's how to kick your IoT strategy into high gear. Get the new IoT Goes Mobile issue of InformationWeek Tech Digest today. (Free registration required.)

- Knowledge management (KM) is the process of identifying, organizing, storing and disseminating information within an organization.

How APIs work

APIs are made up of two related elements. The first is a specification that describes howinformation is exchanged between programs, done in the form of a request for processing and a return of the necessary data. The second is a software interface written to that specification and published in some way for use.The software that wants to access the features and capabilities of the API is said to call it, and the software that creates the API is said to publish it.

Why APIs are important for business

The web, software designed exchange information via the internet and cloud computinghave all combined to increase the interest in APIs in general and services in particular.Software that was

once custom-developed for a specific purpose is now often written referencing APIs that provide broadly useful features, reducing development time and cost and mitigating the risk of errors.APIs have steadily improved software quality over the last decade, and the growing number of web services exposed through APIs by cloud providers is also encouraging thecreation of cloud-specific applications, internet of things (IoT) efforts and apps to support mobiledevices and users.

Three basic types of APIs

APIs take three basic forms: local, web-like and program-like.

1. **Local APIs** are the original form, from which the name came. They offer OSor middleware services to application programs. Microsoft's .NET APIs, the TAPI (Telephony API) for voice applications, and database access APIs are examples of the local APIform.

2. **Web APIs** are designed to represent widely used resources like HTML pages and are accessed using a simple HTTP protocol. Any web URL activates a web API. Web APIs are often called REST (representational state transfer) or RESTful because the publisher of REST interfaces doesn't save any data internally between requests. As such, requests from many users can be intermingled as they would be on theinternet.

3. **Program APIs** are based on remote procedure call (RPC) technology that

makes a remote program component appear to be local to the rest of the software. Service orientedarchitecture (SOA) APIs, such as Microsoft's WS-series of APIs, are programAPIs.