1. Definition of IOT ?

IoT is a technology transition in which devices will allow us to sense and control the physical world by making objects smarter and connecting them through an intelligent network.[1]

Q. evolutionary phases of internet?

Genenration

Genesis of IoT

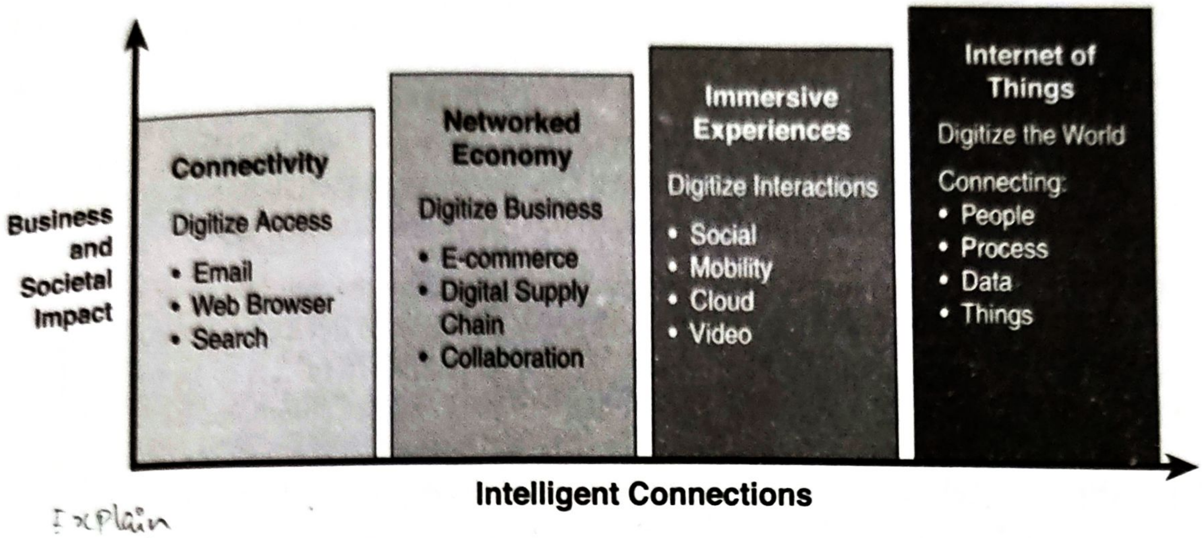
The age of IoT is often said to have started between the years 2008 and 2009. During this time period, the number of devices connected to the Internet eclipsed the world's population. With more "things" connected to the Internet than people in the world, a new age was upon us, and the Internet of Things was born.

The person credited with the creation of the term "Internet of Things" is Kevin Ashton. While working for Procter & Gamble in 1999, Kevin used this phrase to explain a new idea related to linking the company's supply chain to the Internet.

Kevin has subsequently explained that IoT now involves the addition of senses to computers. He was quoted as saying: "In the twentieth century, computers were brains without senses—they only knew what we told them." Computers depended on humans to input data and knowledge through typing, bar codes, and so on. IoT is changing this paradigm; in the twenty-first century, computers are sensing things for themselves.[2]

It is widely accepted that IoT is a major technology shift, but what is its scale and importance? Where does it fit in the evolution of the Internet?

As shown in Figure 1-1, the evolution of the Internet can be categorized into four phases. Each of these phases has had a profound impact on our society and our lives. These four phases are further defined in Table 1-1.



Explain

Figure 1-1 *Evolutionary Phases of the Internet*

Table 1-1 *Evolutionary Phases of the Internet*

Internet Phase	Definition
Connectivity (Digitize access)	This phase connected people to email, web services, and search so that information is easily accessed.
Networked Economy (Digitize business)	This phase enabled e-commerce and supply chain enhancements along with collaborative engagement to drive increased efficiency in business processes.
Immersive Experiences (Digitize interactions)	This phase extended the Internet experience to encompass widespread video and social media while always being connected through mobility. More and more applications are moved into the cloud.
Internet of Things (Digitize the world)	This phase is adding connectivity to objects and machines in the world around us to enable new services and experiences. It is connecting the unconnected.

Each of these evolutionary phases builds on the previous one. With each subsequent phase, more value becomes available for businesses, governments, and society in general.

The first phase, Connectivity, began in the mid-1990s. Though it may be hard to remember, or even imagine if you are younger, the world was not always connected as it is today. In the beginning, email and getting on the Internet were luxuries for universities and corporations. Getting the average person online involved dial-up modems, and even basic connectivity often seemed like a small miracle.

Even though connectivity and its speed continued to improve, a saturation point was reached where connectivity was no longer the major challenge. The focus was now on leveraging connectivity for efficiency and profit. This inflection point marked the beginning of the second phase of the Internet evolution, called the Networked Economy.

With the Networked Economy, e-commerce and digitally connected supply chains became the rage, and this caused one of the major disruptions of the past 100 years. Vendors and suppliers became closely interlinked with producers, and online shopping experienced incredible growth. The victims of this shift were traditional brick-and-mortar retailers. The economy itself became more digitally intertwined as suppliers, vendors, and consumers all became more directly connected.

The third phase, Immersive Experiences, is characterized by the emergence of social media, collaboration, and widespread mobility on a variety of devices. Connectivity is now pervasive, using multiple platforms from mobile phones to tablets to laptops and desktop computers. This pervasive connectivity in turn enables communication and collaboration as well as social media across multiple channels, via email, texting, voice, and video. In essence, person-to-person interactions have become digitized.

The latest phase is the Internet of Things. Despite all the talk and media coverage of IoT, in many ways we are just at the beginning of this phase. When you think about the fact that 99% of "things" are still unconnected, you can better understand what this evolutionary phase is all about. Machines and objects in this phase connect with other machines and objects, along with humans. Business and society have already started down this path and are experiencing huge increases in data and knowledge. In turn, this is now leading to previously unrecognized insights, along with increased automation and new process efficiencies. IoT is poised to change our world in new and exciting ways, just as the past Internet phases already have.

3. Challenges of IOT ?

IoT Challenges

While an IoT-enabled future paints an impressive picture, it does not come without significant challenges. Many parts of IoT have become reality, but certain obstacles need to be overcome for IoT to become ubiquitous throughout industry and our everyday life. Table 1-4 highlights a few of the most significant challenges and problems that IoT is currently facing.

Table 1-4 *IoT Challenges*

Challenge	Description
Scale	While the scale of IT networks can be large, the scale of OT can be several orders of magnitude larger. For example, one large electrical utility in Asia recently began deploying IPv6-based smart meters on its electrical grid. While this utility company has tens of thousands of employees (which can be considered IP nodes in the network), the number of meters in the service area is tens of millions. This means the scale of the network the utility is managing has increased by more than 1,000-fold! Chapter 5, "IP as the IoT Network Layer," explores how new design approaches are being developed to scale IPv6 networks into the millions of devices.
Security	With more "things" becoming connected with other "things" and people, security is an increasingly complex issue for IoT. Your threat surface is now greatly expanded, and if a device gets hacked, its connectivity is a major concern. A compromised device can serve as a launching point to attack other devices and systems. IoT security is also pervasive across just about every facet of IoT. For more information on IoT security, see Chapter 8, "Securing IoT."
Privacy	As sensors become more prolific in our everyday lives, much of the data they gather will be specific to individuals and their activities. This data can range from health information to shopping patterns and transactions at a retail establishment. For businesses, this data has monetary value. Organizations are now discussing who owns this data and how individuals can control whether it is shared and with whom.
Big data and data analytics	IoT and its large number of sensors is going to trigger a deluge of data that must be handled. This data will provide critical information and insights if it can be processed in an efficient manner. The challenge, however, is evaluating massive amounts of data arriving from different sources in various forms and doing so in a timely manner. See Chapter 7 for more information on IoT and the challenges it faces from a big data perspective.

Challenge	Description
Interoperability	As with any other nascent technology, various protocols and architectures are jockeying for market share and standardization within IoT. Some of these protocols and architectures are based on proprietary elements, and others are open. Recent IoT standards are helping minimize this problem, but there are often various protocols and implementations available for IoT networks. The prominent protocols and architectures—especially open, standards-based implementations—are the subject of this book. For more information on IoT architectures, see Chapter 2, "IoT Network Architecture and Design." Chapter 4, "Connecting Smart Objects," Chapter 5, "IP as the IoT Network Layer," and Chapter 6, "Application Protocols for IoT," take a more in-depth look at the protocols that make up IoT.

4. one M2M or IOTWF architecture?

machine to machine

The oneM2M IoT Standardized Architecture

In an effort to standardize the rapidly growing field of machine-to-machine (M2M) communications, the European Telecommunications Standards Institute (ETSI) created the M2M Technical Committee in 2008. The goal of this committee was to create a common architecture that would help accelerate the adoption of M2M applications and devices. Over time, the scope has expanded to include the Internet of Things.

Other related bodies also began to create similar M2M architectures, and a common standard for M2M became necessary. Recognizing this need, in 2012 ETSI and 13 other founding members launched oneM2M as a global initiative designed to promote efficient M2M communication systems and IoT. The goal of oneM2M is to create a common services layer, which can be readily embedded in field devices to allow communication with application servers.[1] oneM2M's framework focuses on IoT services, applications, and platforms. These include smart metering applications, smart grid, smart city automation, e-health, and connected vehicles.

One of the greatest challenges in designing an IoT architecture is dealing with the heterogeneity of devices, software, and access methods. By developing a horizontal platform architecture, oneM2M is developing standards that allow interoperability at all levels of the IoT stack. For example, you might want to automate your HVAC system by connecting it with wireless temperature sensors spread throughout your office. You decide to deploy sensors that use LoRaWAN technology (discussed in Chapter 4, "Connecting Smart Objects"). The problem is that the LoRaWAN network and the BACnet system that your HVAC and BMS run on are completely different systems and have no natural connection point. This is where the oneM2M common services architecture comes in. oneM2M's horizontal framework and RESTful APIs allow the LoRaWAN system to interface with the building management system over an IoT network, thus promoting end-to-end IoT communications in a consistent way, no matter how heterogeneous the networks.

HVAC - heating, ventilation & AC

Building mgmt syst - BMS

Figure 2-1 illustrates the oneM2M IoT architecture.

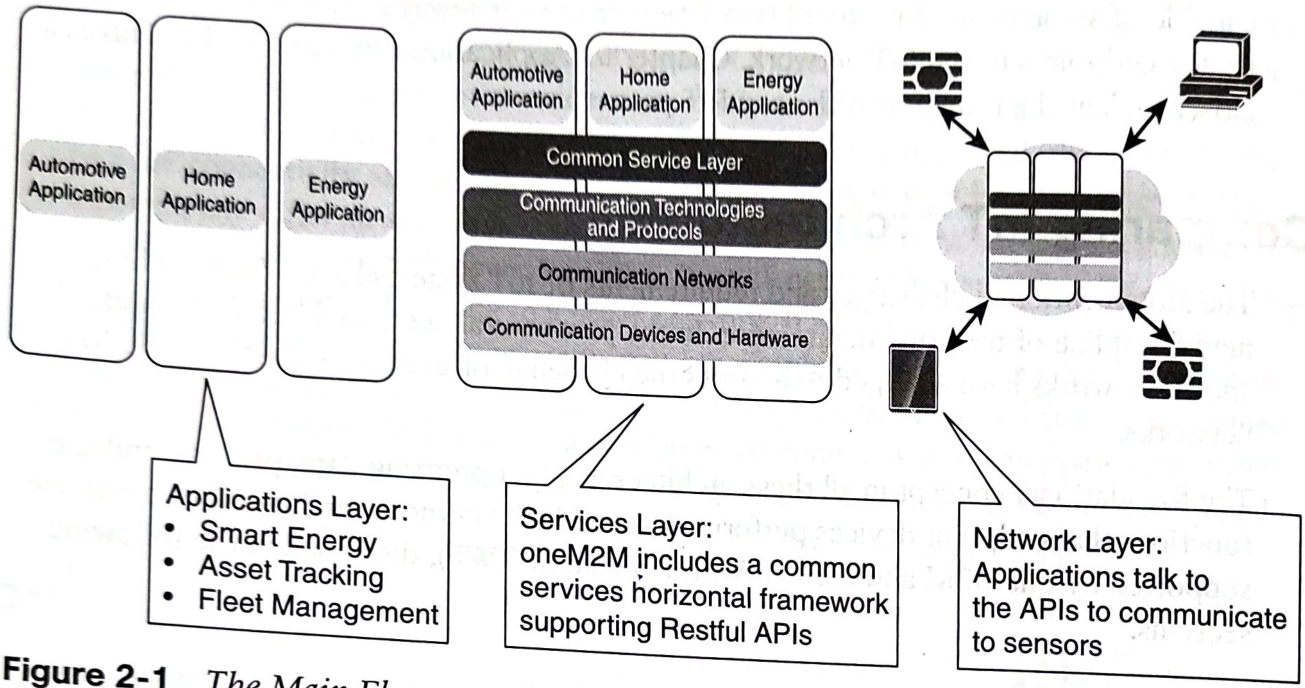


Figure 2-1 *The Main Elements of the oneM2M IoT Architecture*

The oneM2M architecture divides IoT functions into three major domains: the application layer, the services layer, and the network layer. While this architecture may seem simple and somewhat generic at first glance, it is very rich and promotes interoperability through IT-friendly APIs and supports a wide range of IoT technologies. Let's examine each of these domains in turn:

- **Applications layer:** The oneM2M architecture gives major attention to connectivity between devices and their applications. This domain includes the application-layer protocols and attempts to standardize northbound API definitions for interaction with business intelligence (BI) systems. Applications tend to be industry-specific and have their own sets of data models, and thus they are shown as vertical entities.
- **Services layer:** This layer is shown as a horizontal framework across the vertical industry applications. At this layer, horizontal modules include the physical network that the IoT applications run on, the underlying management protocols, and the hardware. Examples include backhaul communications via cellular, MPLS networks, VPNs, and so on. Riding on top is the common services layer. This conceptual layer adds APIs and middleware supporting third-party services and applications. One of the stated goals of oneM2M is to "develop technical specifications which address the need for a common M2M Service Layer that can be readily embedded within various hardware and software nodes, and rely upon connecting the myriad of devices in the field area network to M2M application servers, which typically reside in a cloud or data center." A critical objective of oneM2M is to attract and actively involve organizations from M2M-related business domains, including telematics and intelligent transportation, healthcare, utility, industrial automation, and smart home applications, to name just a few.[2]

14469582

- **Network layer:** This is the communication domain for the IoT devices and endpoints. It includes the devices themselves and the communications network that links them. Embodiments of this communications infrastructure include wireless mesh technologies, such as IEEE 802.15.4, and wireless point-to-multipoint systems, such as IEEE 801.11ah. Also included are wired device connections, such as IEEE 1901 power line communications. Chapter 4 provides more details on these connectivity technologies.

  In many cases, the smart (and sometimes not-so-smart) devices communicate with each other. In other cases, machine-to-machine communication is not necessary, and the devices simply communicate through a field area network (FAN) to use-case-specific apps in the IoT application domain. Therefore, the device domain also includes the gateway device, which provides communications up into the core network and acts as a demarcation point between the device and network domains.

Technical Specifications and Technical Reports published by oneM2M covering IoT functional architecture and other aspects can be found at www.onem2m.org.

The IoT World Forum (IoTWF) Standardized Architecture

In 2014 the IoTWF architectural committee (led by Cisco, IBM, Rockwell Automation, and others) published a seven-layer IoT architectural reference model. While various IoT reference models exist, the one put forth by the IoT World Forum offers a clean, simplified perspective on IoT and includes edge computing, data storage, and access. It provides a succinct way of visualizing IoT from a technical perspective. Each of the seven layers is broken down into specific functions, and security encompasses the entire model. Figure 2-2 details the IoT Reference Model published by the IoTWF.

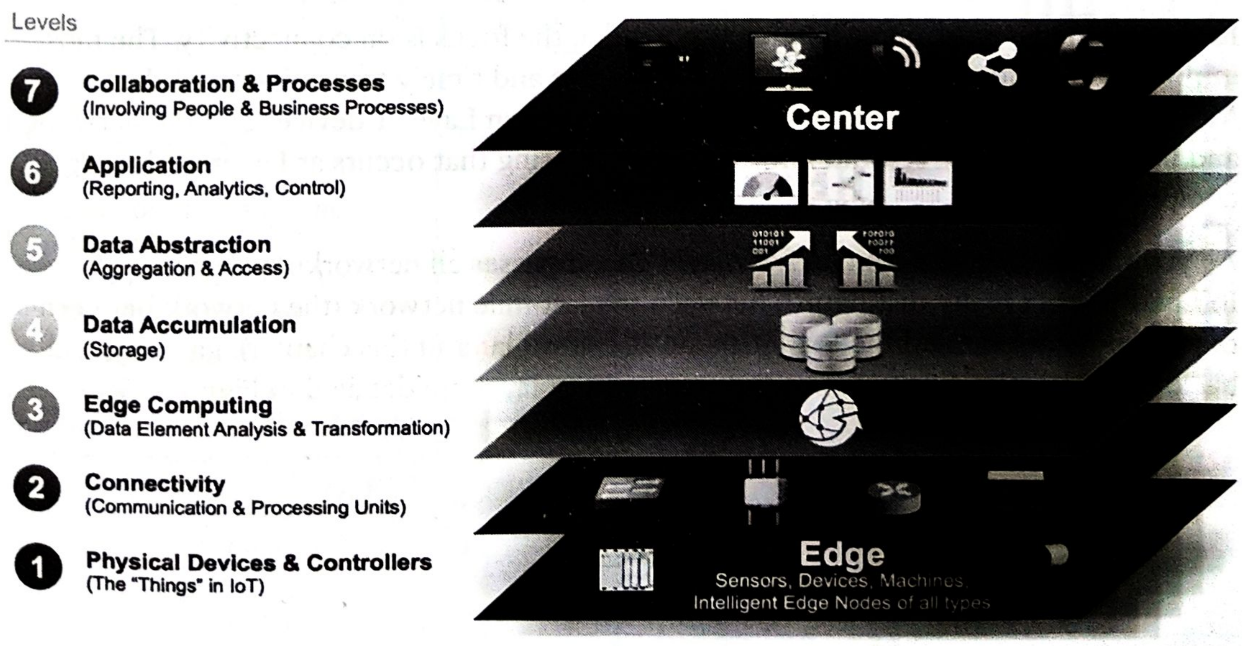


Figure 2-2 *IoT Reference Model Published by the IoT World Forum*

As shown in Figure 2-2, the IoT Reference Model defines a set of levels with control flowing from the center (this could be either a cloud service or a dedicated data center), to the edge, which includes sensors, devices, machines, and other types of intelligent end nodes. In general, data travels up the stack, originating from the edge, and goes northbound to the center. Using this reference model, we are able to achieve the following:

- Decompose the IoT problem into smaller parts
- Identify different technologies at each layer and how they relate to one another
- Define a system in which different parts can be provided by different vendors
- Have a process of defining interfaces that leads to interoperability
- Define a tiered security model that is enforced at the transition points between levels

The following sections look more closely at each of the seven layers of the IoT Reference Model.

Layer 1: Physical Devices and Controllers Layer

The first layer of the IoT Reference Model is the physical devices and controllers layer. This layer is home to the "things" in the Internet of Things, including the various endpoint devices and sensors that send and receive information. The size of these "things" can range from almost microscopic sensors to giant machines in a factory. Their primary function is generating data and being capable of being queried and/or controlled over a network.

Layer 2: Connectivity Layer

In the second layer of the IoT Reference Model, the focus is on connectivity. The most important function of this IoT layer is the reliable and timely transmission of data. More specifically, this includes transmissions between Layer 1 devices and the network and between the network and information processing that occurs at Layer 3 (the edge computing layer).

As you may notice, the connectivity layer encompasses all networking elements of IoT and doesn't really distinguish between the last-mile network (the network between the sensor/endpoint and the IoT gateway, discussed later in this chapter), gateway, and backhaul networks. Functions of the connectivity layer are detailed in Figure 2-3.

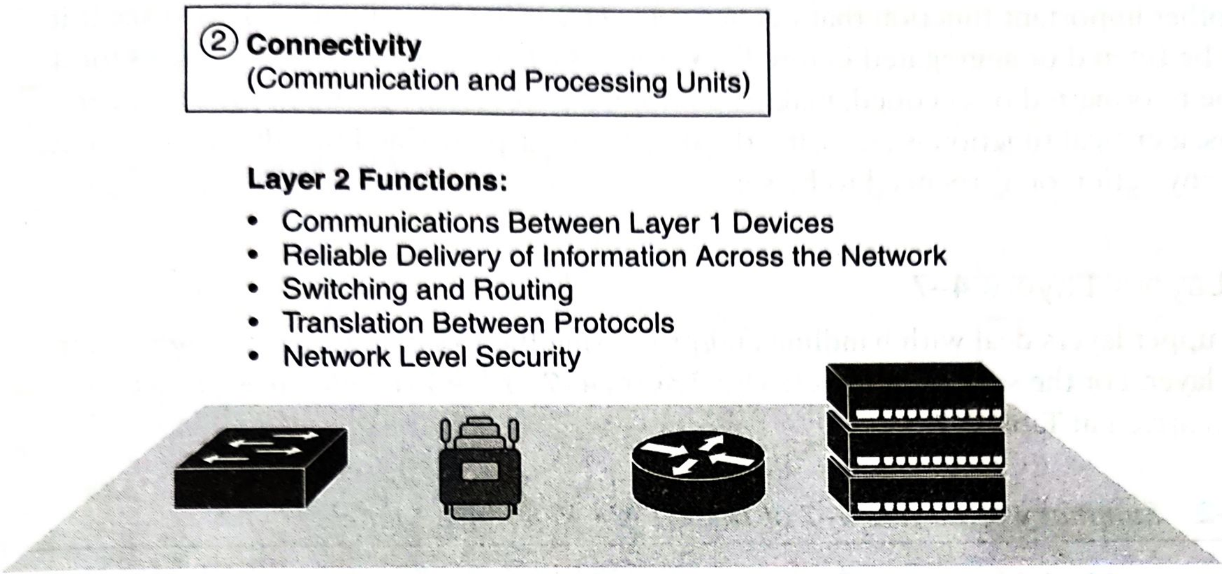


Figure 2-3 *IoT Reference Model Connectivity Layer Functions*

Layer 3: Edge Computing Layer

Edge computing is the role of Layer 3. Edge computing is often referred to as the "fog" layer and is discussed in the section "Fog Computing," later in this chapter. At this layer, the emphasis is on data reduction and converting network data flows into information that is ready for storage and processing by higher layers. One of the basic principles of this reference model is that information processing is initiated as early and as close to the edge of the network as possible. Figure 2-4 highlights the functions handled by Layer 3 of the IoT Reference Model.

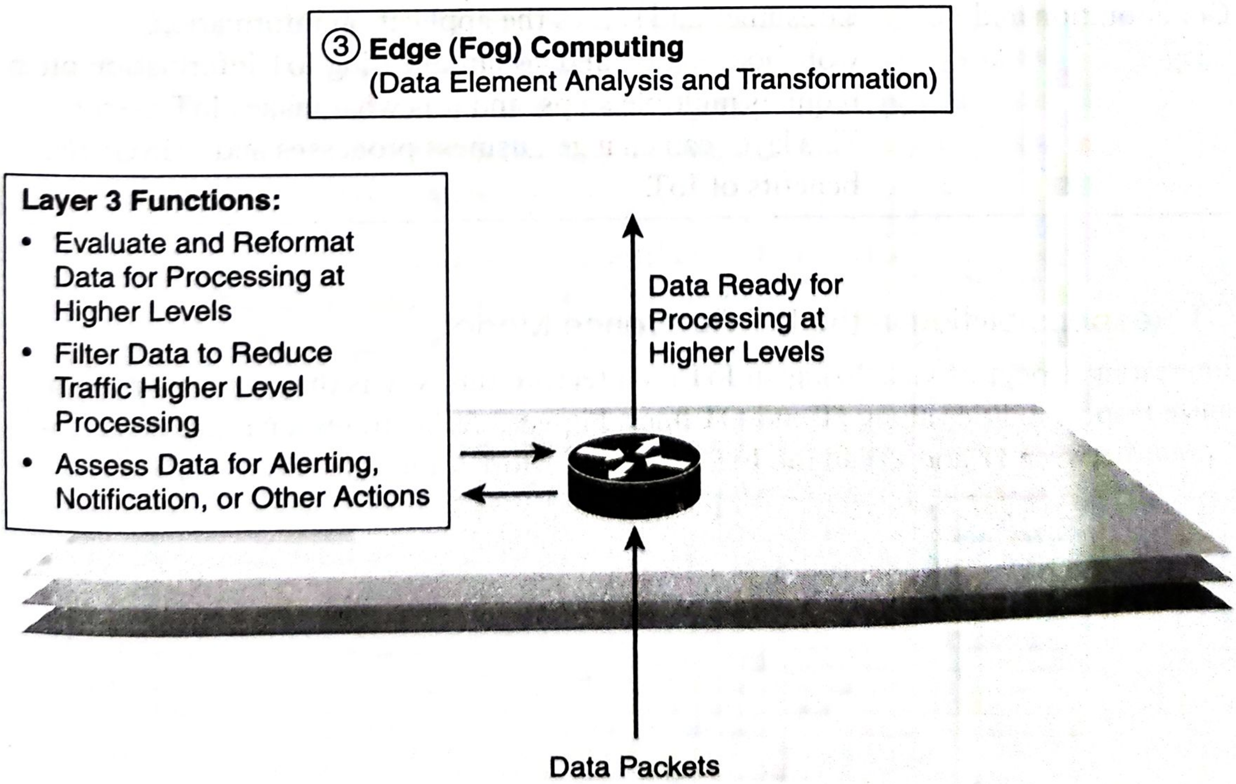


Figure 2-4 *IoT Reference Model Layer 3 Functions*

Another important function that occurs at Layer 3 is the evaluation of data to see if it can be filtered or aggregated before being sent to a higher layer. This also allows for data to be reformatted or decoded, making additional processing by other systems easier. Thus, a critical function is assessing the data to see if predefined thresholds are crossed and any action or alerts need to be sent.

Upper Layers: Layers 4–7

The upper layers deal with handling and processing the IoT data generated by the bottom layer. For the sake of completeness, Layers 4–7 of the IoT Reference Model are summarized in Table 2-2.

Table 2-2 *Summary of Layers 4–7 of the IoTWF Reference Model*

IoT Reference Model Layer	Functions
Layer 4: Data accumulation layer	Captures data and stores it so it is usable by applications when necessary. Converts event-based data to query-based processing.
Layer 5: Data abstraction layer	Reconciles multiple data formats and ensures consistent semantics from various sources. Confirms that the data set is complete and consolidates data into one place or multiple data stores using virtualization.
Layer 6: Applications layer	Interprets data using software applications. Applications may monitor, control, and provide reports based on the analysis of the data.
Layer 7: Collaboration and processes layer	Consumes and shares the application information. Collaborating on and communicating IoT information often requires multiple steps, and it is what makes IoT useful. This layer can change business processes and delivers the benefits of IoT.

5. IoT data management & compute stack ?

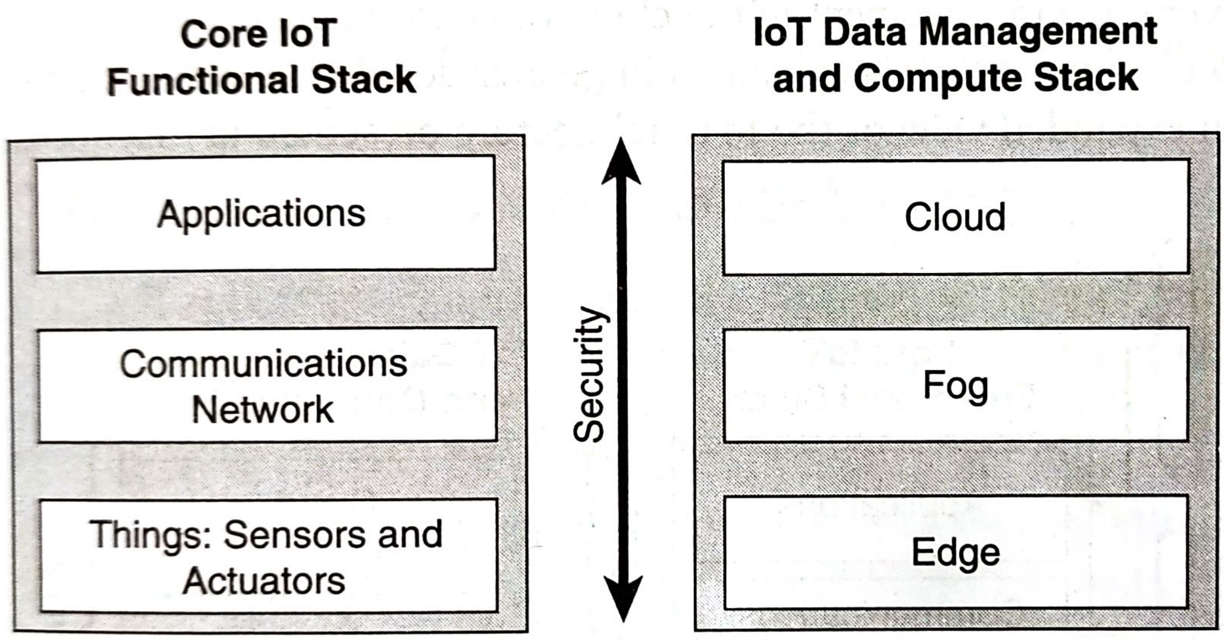


Figure 2-6 *Simplified IoT Architecture*

IoT Data Management and Compute Stack

One of the key messages in the first two chapters of this book is that the massive scale of IoT networks is fundamentally driving new architectures. For instance, Figure 1-2 in Chapter 1 illustrates how the "things" connected to the Internet are continuing to grow exponentially, with a prediction by Cisco that by 2020 there will be more than 50 billion devices connected to some form of an IP network. Clearly, traditional IT networks are not prepared for this magnitude of network devices. However, beyond the network architecture itself, consider the data that is generated by these devices. If the number of devices is beyond conventional numbers, surely the data generated by these devices must also be of serious concern.

In fact, the data generated by IoT sensors is one of the single biggest challenges in building an IoT system. In the case of modern IT networks, the data sourced by a computer or server is typically generated by the client/server communications model, and it serves the needs of the application. In sensor networks, the vast majority of data generated is unstructured and of very little use on its own. For example, the majority of data generated by a smart meter is nothing more than polling data; the communications system simply determines whether a network connection to the meter is still active. This data on its own is of very little value. The real value of a smart meter is the metering data read by the meter management system (MMS). However, if you look at the raw polling data from a different perspective, the information can be very useful. For example, a utility may have millions of meters covering its entire service area. If whole sections of the smart grid start to show an interruption of connectivity to the meters, this data can be analyzed and combined with other sources of data, such as weather reports and electrical demand in the grid, to provide a complete picture of what is happening. This information can help determine whether the loss of connection to the meters is truly a loss of power or whether some other problem has developed in the grid. Moreover, analytics of this data can help the utility quickly determine the extent of the service outage and repair the disruption in a timely fashion.

In most cases, the processing location is outside the smart object. A natural location for this processing activity is the cloud. Smart objects need to connect to the cloud, and data processing is centralized. One advantage of this model is simplicity. Objects just need to connect to a central cloud application. That application has visibility over all the IoT nodes and can process all the analytics needed today and in the future.

However, this model also has limitations. As data volume, the variety of objects connecting to the network, and the need for more efficiency increase, new requirements appear, and those requirements tend to bring the need for data analysis closer to the IoT system. These new requirements include the following:

- **Minimizing latency:** Milliseconds matter for many types of industrial systems, such as when you are trying to prevent manufacturing line shutdowns or restore electrical service. Analyzing data close to the device that collected the data can make a difference between averting disaster and a cascading system failure.
- **Conserving network bandwidth:** Offshore oil rigs generate 500 GB of data weekly. Commercial jets generate 10 TB for every 30 minutes of flight. It is not practical to transport vast amounts of data from thousands or hundreds of thousands of edge devices to the cloud. Nor is it necessary because many critical analyses do not require cloud-scale processing and storage.
- **Increasing local efficiency:** Collecting and securing data across a wide geographic area with different environmental conditions may not be useful. The environmental conditions in one area will trigger a local response independent from the conditions of another site hundreds of miles away. Analyzing both areas in the same cloud system may not be necessary for immediate efficiency.

An important design consideration, therefore, is how to design an IoT network to manage this volume of data in an efficient way such that the data can be quickly analyzed and lead to business benefits. The volume of data generated by IoT devices can be so great that it can easily overrun the capabilities of the headend system in the data center or the cloud. For example, it has been observed that a moderately sized smart meter network of 1 million meters will generate close to 1 billion data points each day (including meter reads and other instrumentation data), resulting in 1 TB of data. For an IT organization that is not prepared to contend with this volume of data storage and real-time analysis, this creates a whole new challenge.

The volume of data also introduces questions about bandwidth management. As the massive amount of IoT data begins to funnel into the data center, does the network have the capacity to sustain this volume of traffic? Does the application server have the ability to ingest, store, and analyze the vast quantity of data that is coming in? This is sometimes referred to as the "impedance mismatch" of the data generated by the IoT system and the management application's ability to deal with that data.

As illustrated in Figure 2-14, data management in traditional IT systems is very simple. The endpoints (laptops, printers, IP phones, and so on) communicate over an IP core network to servers in the data center or cloud. Data is generally stored in the data center, and the physical links from access to core are typically high bandwidth, meaning access to IT data is quick.

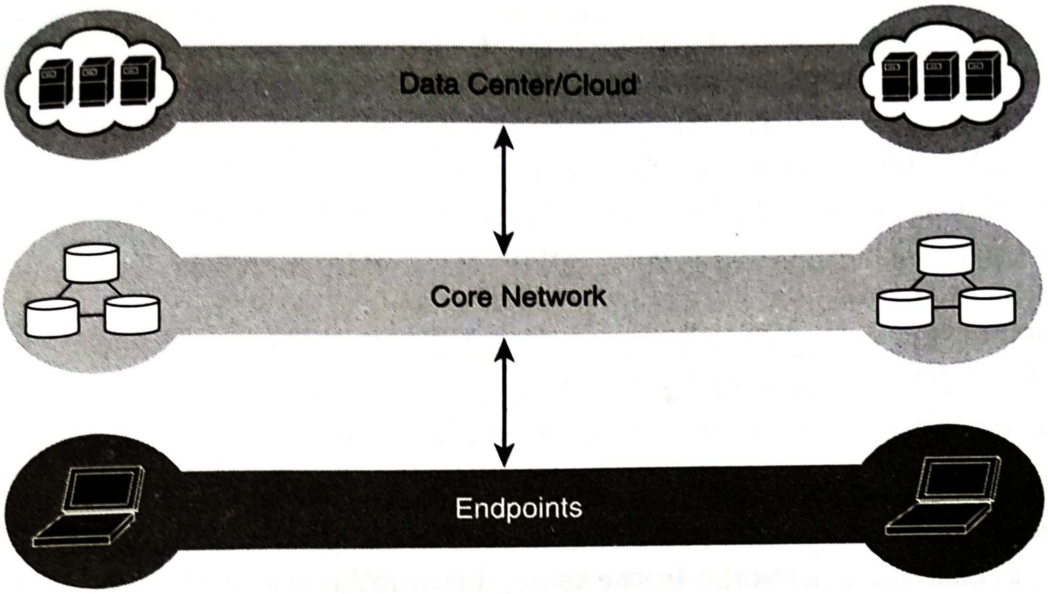


Figure 2-14 *The Traditional IT Cloud Computing Model*

IoT systems function differently. Several data-related problems need to be addressed:

- Bandwidth in last-mile IoT networks is very limited. When dealing with thousands/millions of devices, available bandwidth may be on order of tens of Kbps per device or even less.
- Latency can be very high. Instead of dealing with latency in the milliseconds range, large IoT networks often introduce latency of hundreds to thousands of milliseconds.
- Network backhaul from the gateway can be unreliable and often depends on 3G/LTE or even satellite links. Backhaul links can also be expensive if a per-byte data usage model is necessary.
- The volume of data transmitted over the backhaul can be high, and much of the data may not really be that interesting (such as simple polling messages).
- Big data is getting bigger. The concept of storing and analyzing all sensor data in the cloud is impractical. The sheer volume of data generated makes real-time analysis and response to the data almost impossible.

Fog Computing

The solution to the challenges mentioned in the previous section is to distribute data management throughout the IoT system, as close to the edge of the IP network as possible. The best-known embodiment of edge services in IoT is fog computing. Any device with computing, storage, and network connectivity can be a fog node. Examples include industrial controllers, switches, routers, embedded servers, and IoT gateways. Analyzing IoT data close to where it is collected minimizes latency, offloads gigabytes of network traffic from the core network, and keeps sensitive data inside the local network.

> **Note** The concept of fog was first developed by Flavio Bonomi and Rodolfo Milito of Cisco Systems. In the world of IoT, fog gets its name from a relative comparison to computing in the cloud layer. Just as clouds exist in the sky, fog rests near the ground. In the same way, the intention of fog computing is to place resources as close to the ground—that is, the IoT devices—as possible. An interesting side note is that the term "fog" was actually coined by Ginny Nichols, Rodolfo's wife. Although not working directly in IoT, she had an excellent grasp of what her husband was developing and was able to quickly draw the comparison between cloud and edge computing. One day she made the suggestion of simply calling it the "fog layer." The name stuck.

An advantage of this structure is that the fog node allows intelligence gathering (such as analytics) and control from the closest possible point, and in doing so, it allows better performance over constrained networks. In one sense, this introduces a new layer to the traditional IT computing model, one that is often referred to as the "fog layer." Figure 2-15 shows the placement of the fog layer in the IoT Data Management and Compute Stack.

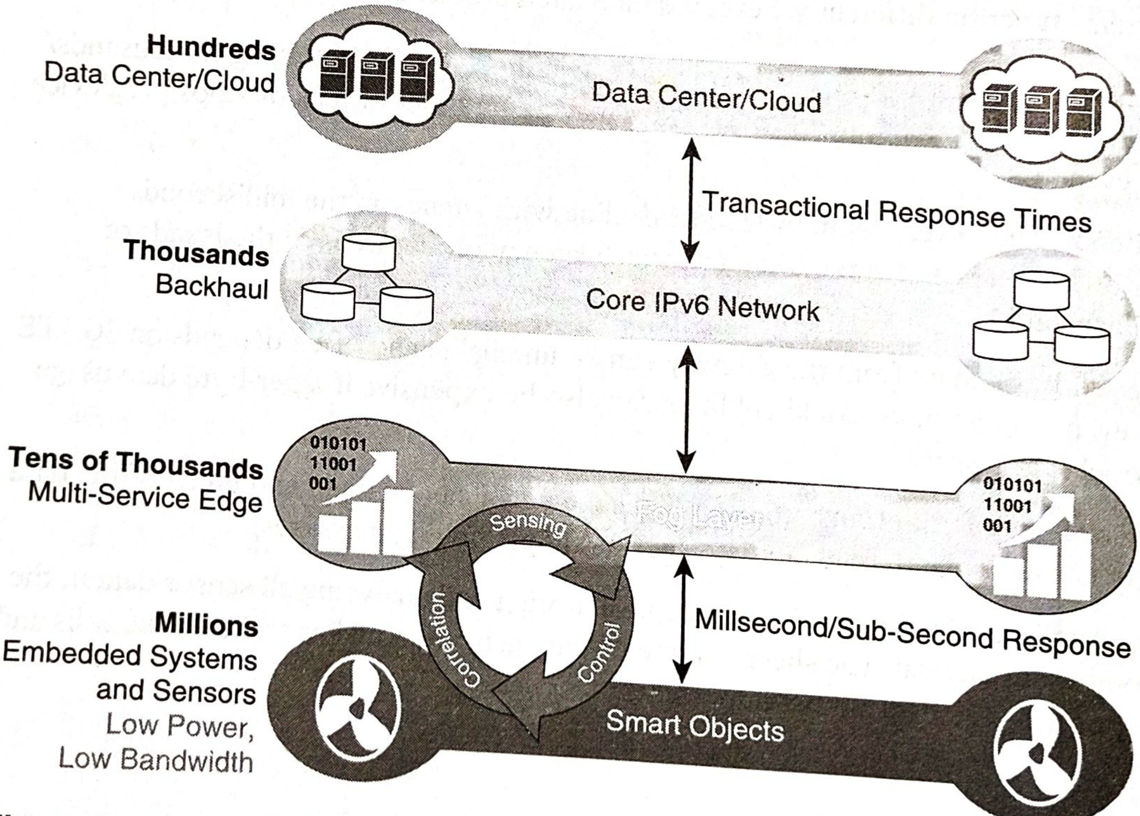


Figure 2-15 *The IoT Data Management and Compute Stack with Fog Computing*

Fog services are typically accomplished very close to the edge device, sitting as close to the IoT endpoints as possible. One significant advantage of this is that the fog node has contextual awareness of the sensors it is managing because of its geographic proximity to those sensors. For example, there might be a fog router on an oil derrick that is monitoring all the sensor activity at that location. Because the fog node is able to analyze information from all the sensors on that derrick, it can provide contextual analysis of the messages it is receiving and may decide to send back only the relevant information over the backhaul network to the cloud. In this way, it is performing distributed analytics such that the volume of data sent upstream is greatly reduced and is much more useful to application and analytics servers residing in the cloud.

In addition, having contextual awareness gives fog nodes the ability to react to events in the IoT network much more quickly than in the traditional IT compute model, which would likely incur greater latency and have slower response times. The fog layer thus provides a distributed edge control loop capability, where devices can be monitored, controlled, and analyzed in real time without the need to wait for communication from the central analytics and application servers in the cloud.

The value of this model is clear. For example, tire pressure sensors on a large truck in an open-pit mine might continually report measurements all day long. There may be only minor pressure changes that are well within tolerance limits, making continual reporting to the cloud unnecessary. Is it really useful to continually send such data back to the cloud over a potentially expensive backhaul connection? With a fog node on the truck, it is possible to not only measure the pressure of all tires at once but also combine this data with information coming from other sensors in the engine, hydraulics, and so on. With this approach, the fog node sends alert data upstream only if an actual problem is beginning to occur on the truck that affects operational efficiency.

IoT fog computing enables data to be preprocessed and correlated with other inputs to produce relevant information. This data can then be used as real-time, actionable knowledge by IoT-enabled applications. Longer term, this data can be used to gain a deeper understanding of network behavior and systems for the purpose of developing proactive policies, processes, and responses.

Fog applications are as diverse as the Internet of Things itself. What they have in common is data reduction—monitoring or analyzing real-time data from network-connected things and then initiating an action, such as locking a door, changing equipment settings, applying the brakes on a train, zooming a video camera, opening a valve in response to a pressure reading, creating a bar chart, or sending an alert to a technician to make a preventive repair.

The defining characteristic of fog computing are as follows:

- **Contextual location awareness and low latency:** The fog node sits as close to the IoT endpoint as possible to deliver distributed computing.
- **Geographic distribution:** In sharp contrast to the more centralized cloud, the services and applications targeted by the fog nodes demand widely distributed deployments.
- **Deployment near IoT endpoints:** Fog nodes are typically deployed in the presence of a large number of IoT endpoints. For example, typical metering deployments often see 3000 to 4000 nodes per gateway router, which also functions as the fog computing node.

- **Wireless communication between the fog and the IoT endpoint:** Although it is possible to connect wired nodes, the advantages of fog are greatest when dealing with a large number of endpoints, and wireless access is the easiest way to achieve such scale.
- **Use for real-time interactions:** Important fog applications involve real-time interactions rather than batch processing. Preprocessing of data in the fog nodes allows upper-layer applications to perform batch processing on a subset of the data.

Edge Computing

Fog computing solutions are being adopted by many industries, and efforts to develop distributed applications and analytics tools are being introduced at an accelerating pace. The natural place for a fog node is in the network device that sits closest to the IoT endpoints, and these nodes are typically spread throughout an IoT network. However, in recent years, the concept of IoT computing has been pushed even further to the edge, and in some cases it now resides directly in the sensors and IoT devices.

Note Edge computing is also sometimes called "mist" computing. If clouds exist in the sky, and fog sits near the ground, then mist is what actually sits on the ground. Thus, the concept of mist is to extend fog to the furthest point possible, right into the IoT endpoint device itself.

IoT devices and sensors often have constrained resources, however, as compute capabilities increase. Some new classes of IoT endpoints have enough compute capabilities to perform at least low-level analytics and filtering to make basic decisions. For example, consider a water sensor on a fire hydrant. While a fog node sitting on an electrical pole in the distribution network may have an excellent view of all the fire hydrants in a local neighborhood, a node on each hydrant would have clear view of a water pressure drop on its own line and would be able to quickly generate an alert of a localized problem. The fog node, on the other hand, would have a wider view and would be able to ascertain whether the problem was more than just localized but was affecting the entire area. Another example is in the use of smart meters. Edge compute–capable meters are able to communicate with each other to share information on small subsets of the electrical distribution grid to monitor localized power quality and consumption, and they can inform a fog node of events that may pertain to only tiny sections of the grid. Models such as these help ensure the highest quality of power delivery to customers.

6. Definition of sensors & actuators, how they interact with physical world?

Sensors, Actuators, and Smart Objects

The following sections describe the capabilities, characteristics, and functionality of sensors and actuators. They also detail how the economic and technical conditions are finally right for IoT to flourish. Finally, you will see how to bring these foundational elements together to form smart objects, which are connected to form the sensor and actuator networks that make most IoT use cases possible.

Sensors

A sensor does exactly as its name indicates: It senses. More specifically, a sensor measures some physical quantity and converts that measurement reading into a digital representation. That digital representation is typically passed to another device for transformation into useful data that can be consumed by intelligent devices or humans.

Naturally, a parallel can be drawn with humans and the use of their five senses to learn about their surroundings. Human senses do not operate independently in silos. Instead, they complement each other and compute together, empowering the human brain to make intelligent decisions. The brain is the ultimate decision maker, and it often uses several sources of sensory input to validate an event and compensate for "incomplete" information.

Sensors are not limited to human-like sensory data. They can measure anything worth measuring. In fact, they are able to provide an extremely wide spectrum of rich and diverse measurement data with far greater precision than human senses; sensors provide superhuman sensory capabilities. This additional dimension of data makes the physical world an incredibly valuable source of information. Sensors can be readily embedded in any physical objects that are easily connected to the Internet by wired or wireless networks. Because these connected host physical objects with multidimensional sensing capabilities communicate with each other and external systems, they can interpret their environment and make intelligent decisions. Connecting sensing devices in this way has ushered in the world of IoT and a whole new paradigm of business intelligence.

Actuators

Actuators are natural complements to sensors. Figure 3-4 demonstrates the symmetry and complementary nature of these two types of devices. As discussed in the previous section, sensors are designed to sense and measure practically any measurable variable in the physical world. They convert their measurements (typically analog) into electric signals or digital representations that can be consumed by an intelligent agent (a device or a human). Actuators, on the others hand, receive some type of control signal (commonly an electric signal or digital command) that triggers a physical effect, usually some type of motion, force, and so on.

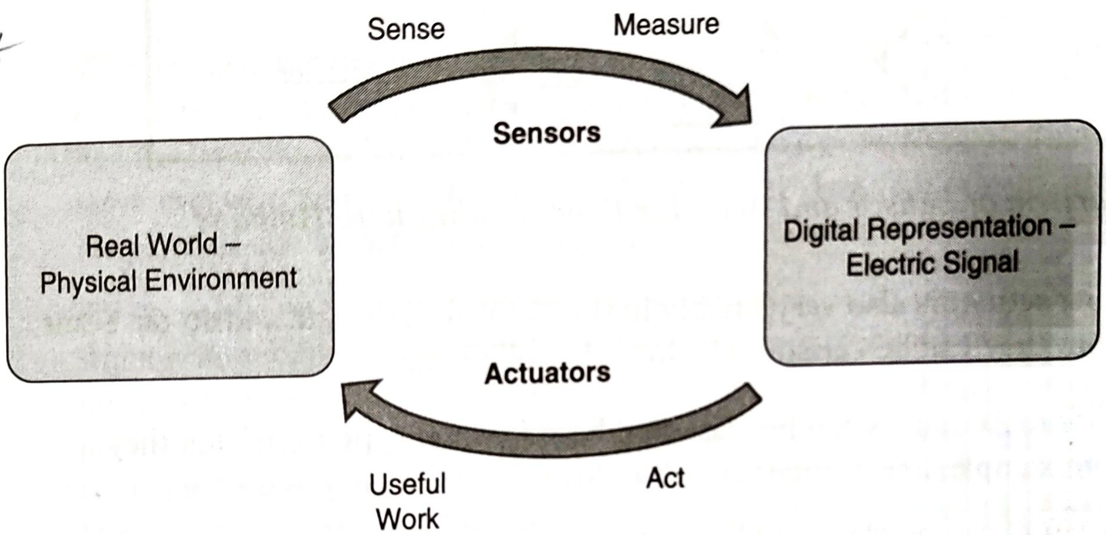


Figure 3-4 *How Sensors and Actuators Interact with the Physical World*

The previous section draws a parallel between sensors and the human senses. This parallel can be extended to include actuators, as shown in Figure 3-5. Humans use their five senses to sense and measure their environment. The sensory organs convert this sensory information into electrical impulses that the nervous system sends to the brain for processing. Likewise, IoT sensors are devices that sense and measure the physical world and (typically) signal their measurements as electric signals sent to some type of microprocessor or microcontroller for additional processing. The human brain signals motor function and movement, and the nervous system carries that information to the appropriate part of the muscular system. Correspondingly, a processor can send an electric signal to an actuator that translates the signal into some type of movement (linear, rotational, and so on) or useful work that changes or has a measurable impact on the physical world. This interaction between sensors, actuators, and processors and the similar functionality in biological systems is the basis for various technical fields, including robotics and biometrics.

7. Wireless sensor n/ws.

Wireless Sensor Networks (WSNs)

Wireless sensor networks are made up of wirelessly connected smart objects, which are sometimes referred to as *motes*. The fact that there is no infrastructure to consider with

WSNs is surely a powerful advantage for flexible deployments, but there are a variety of design constraints to consider with these wirelessly connected smart objects. Figure 3-8 illustrates some of these assumptions and constraints usually involved in WSNs.

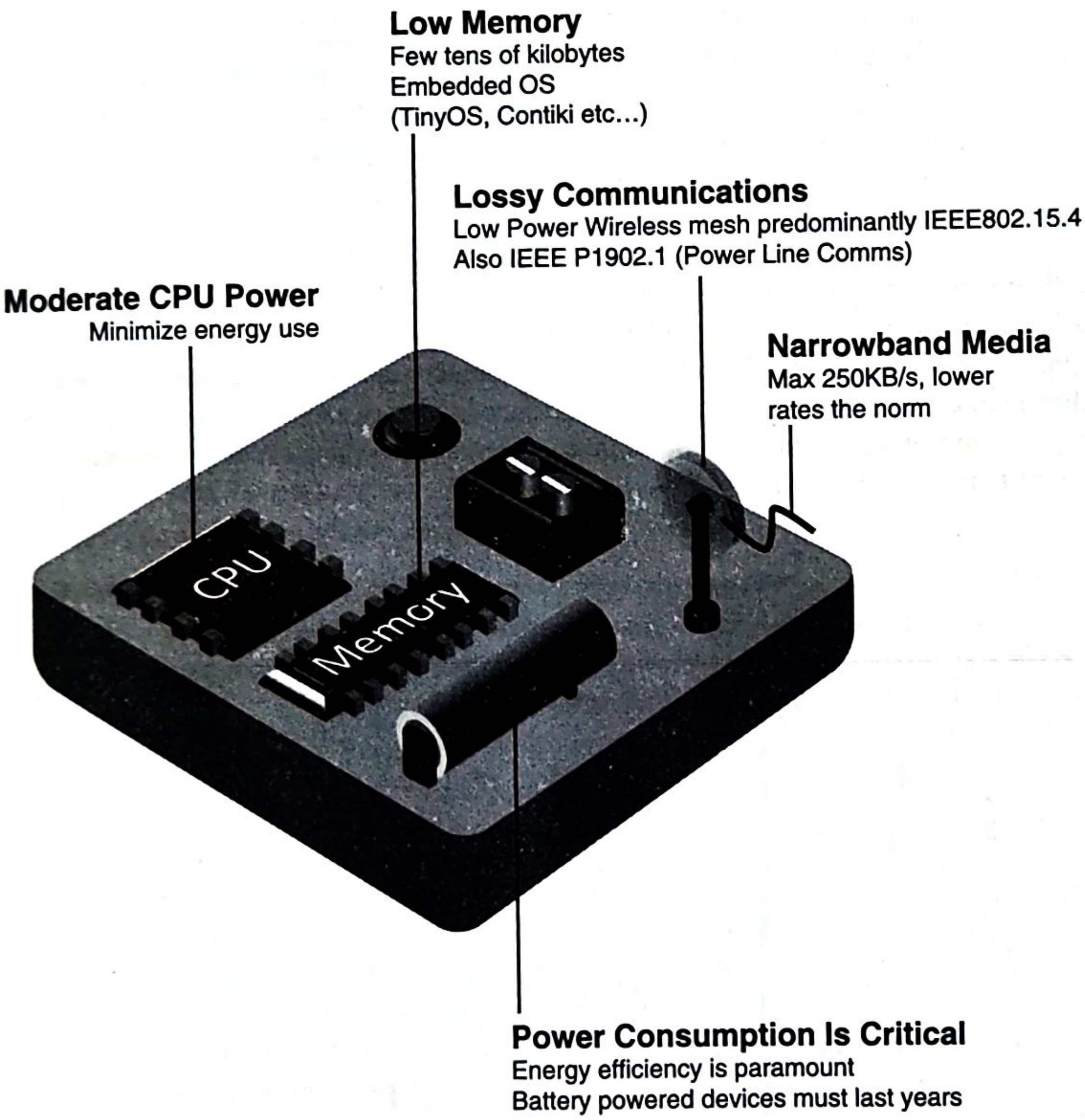


Figure 3-8 *Design Constraints for Wireless Smart Objects*

The following are some of the most significant limitations of the smart objects in WSNs:

- Limited processing power
- Limited memory
- Lossy communication
- Limited transmission speeds
- Limited power

These limitations greatly influence how WSNs are designed, deployed, and utilized. The fact that individual sensor nodes are typically so limited is a reason that they are often deployed in very large numbers. As the cost of sensor nodes continues to decline, the ability to deploy highly redundant sensors becomes increasingly feasible. Because many sensors are very inexpensive and correspondingly inaccurate, the ability to deploy smart objects redundantly allows for increased accuracy.

Note Smart objects with limited processing, memory, power, and so on are often referred to as *constrained nodes*. Constrained nodes are discussed in more detail in Chapter 5.

Such large numbers of sensors permit the introduction of hierarchies of smart objects. Such a hierarchy provides, among other organizational advantages, the ability to aggregate similar sensor readings from sensor nodes that are in close proximity to each other. Figure 3-9 shows an example of such a data aggregation function in a WSN where temperature readings from a logical grouping of temperature sensors are aggregated as an average temperature reading.

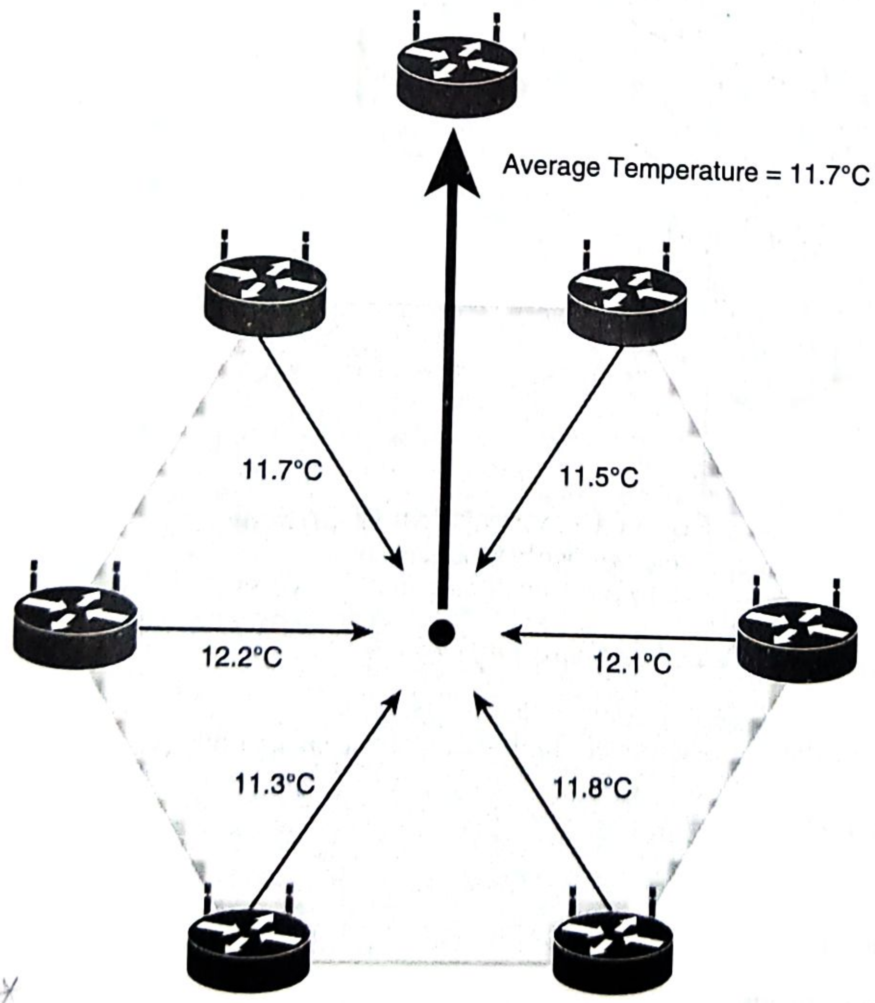


Figure 3-9 *Data Aggregation in Wireless Sensor Networks*

These data aggregation techniques are helpful in reducing the amount of overall traffic (and energy) in WSNs with very large numbers of deployed smart objects. This data aggregation at the network edges is where fog and mist computing, discussed in Chapter 2, "IoT Network Architecture and Design," are critical IoT architectural elements needed to deliver the scale and performance required by so many IoT use cases. While there are certain instances in which sensors continuously stream their measurement data, this is typically not the case. Wirelessly connected smart objects generally have one of the following two communication patterns:

- **Event-driven:** Transmission of sensory information is triggered only when a smart object detects a particular event or predetermined threshold.
- **Periodic:** Transmission of sensory information occurs only at periodic intervals.

The decision of which of these communication schemes is used depends greatly on the specific application. For example, in some medical use cases, sensors periodically send postoperative vitals, such as temperature or blood pressure readings. In other medical use cases, the same blood pressure or temperature readings are triggered to be sent only when certain critically low or high readings are measured.

As WSNs grow to very large numbers of smart objects, there is a trend toward ever-increasing levels of autonomy. For example, manual configuration of potentially thousands of smart objects is impractical and unwieldy, so smart objects in a WSN are typically self-configuring or automated by an IoT management platform in the background. Likewise, additional levels of autonomous functions are required to establish cohesive communication among the multitudinous nodes of large-scale WSNs that are often ad hoc deployments with no regard for uniform node distribution and/or density. For example, there is an increasing trend toward "smart dust" applications, in which very small sensor nodes (that is, MEMS) are scattered over a geographic area to detect vibrations, temperature, humidity, and so on. This technology has practically limitless capabilities, such as military (for example, detecting enemy troop movement), environmental (for example, detecting earthquakes or forest fires), and industrial (for example, detecting manufacturing anomalies, asset tracking). Some level of self-organization is required for networking the scads of wireless smart objects such that these nodes autonomously come together to form a true network with a common purpose. This capability to self-organize is able to adapt and evolve the logical topology of a WSN to optimize communication (among nodes as well as to centralized wireless controllers), simplify the introduction of new smart objects, and improve reliability and access to services.

Additional advantages of being able to deploy large numbers of wireless low-cost smart objects are the inherent ability to provide fault tolerance, reliability, and the capability to extend the life of a WSN, especially in scenarios where the smart objects have limited battery life. Autonomous techniques, such as self-healing, self-protection, and self-optimization, are often employed to perform these functions on behalf of an overall WSN system. IoT applications are often mission critical, and in large-scale WSNs, the overall system can't fail if the environment suddenly changes, wireless communication is temporarily lost, or a limited number of nodes run out of battery power or function improperly.

8. IoT access technology

Zigbee, LoRaWAN

ZigBee

Based on the idea of ZigBee-style networks in the late 1990s, the first ZigBee specification was ratified in 2004, shortly after the release of the IEEE 802.15.4 specification the previous year. While not released as a typical standard, like an RFC, ZigBee still had industry support from more than 100 companies upon its initial publication. This industry support has grown to more than 400 companies that are members of the ZigBee Alliance. Similar to the Wi-Fi Alliance, the Zigbee Alliance is an industry group formed to certify interoperability between vendors and it is committed to driving and evolving ZigBee as an IoT solution for interconnecting smart objects.

ZigBee solutions are aimed at smart objects and sensors that have low bandwidth and low power needs. Furthermore, products that are ZigBee compliant and certified by the ZigBee Alliance should interoperate even though different vendors may manufacture them.

The Zigbee specification has undergone several revisions. In the 2006 revision, sets of commands and message types were introduced, and increased in number in the 2007 (called Zigbee pro) iteration, to achieve different functions for a device, such as metering, temperature, or lighting control. These sets of commands and message types are called clusters. Ultimately, these clusters from different functional domains or libraries form the building blocks of Zigbee application profiles. Vendors implementing pre-defined Zigbee application profiles like Home Automation or Smart Energy can ensure interoperability between their products.

The main areas where ZigBee is the most well-known include automation for commercial, retail, and home applications and smart energy. In the industrial and commercial automation space, ZigBee-based devices can handle various functions, from measuring temperature and humidity to tracking assets. For home automation, ZigBee can control lighting, thermostats, and security functions. ZigBee Smart Energy brings together a variety of interoperable products, such as smart meters, that can monitor and control the use and delivery of utilities, such as electricity and water. These ZigBee products are controlled by the utility provider and can help coordinate usage between homes and businesses and the utility provider itself to provide more efficient operations.

The traditional ZigBee stack is illustrated in Figure 4-3. As mentioned previously, ZigBee utilizes the IEEE 802.15.4 standard at the lower PHY and MAC layers. (The 802.15.4 PHY and MAC layers are covered in detail later in this chapter.) ZigBee specifies the network and security layer and application support layer that sit on top of the lower layers.

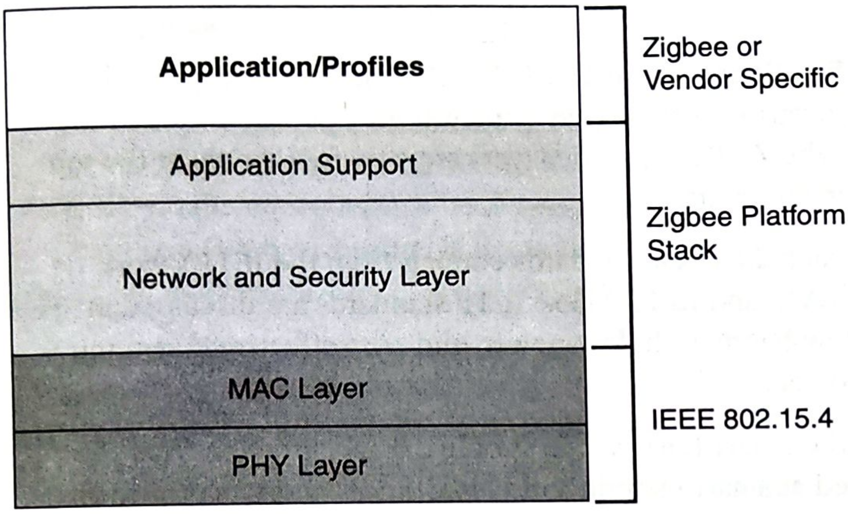


Figure 4-3 *High-Level ZigBee Protocol Stack*

The ZigBee network and security layer provides mechanisms for network startup, configuration, routing, and securing communications. This includes calculating routing paths in what is often a changing topology, discovering neighbors, and managing the routing tables as devices join for the first time. The network layer is also responsible for forming the appropriate topology, which is often a mesh but could be a star or tree as well. From a security perspective, ZigBee utilizes 802.15.4 for security at the MAC layer, using the Advanced Encryption Standard (AES) with a 128-bit key and also provides security at the network and application layers.

Note ZigBee uses Ad hoc On-Demand Distance Vector (AODV) routing across a mesh network. Interestingly, this routing algorithm does not send a message until a route is needed. Assuming that the next hop for a route is not in its routing table, a network node broadcasts a request for a routing connection. This causes a burst of routing-related traffic, but after a comparison of various responses, the path with the lowest number of hops is determined for the connection. This process is quite different from standard enterprise routing protocols, which usually learn the entire network topology in some manner and then store a consolidated but complete routing table.

The application support layer in Figure 4-3 interfaces the lower portion of the stack dealing with the networking of ZigBee devices with the higher-layer applications. ZigBee predefines many application profiles for certain industries, and vendors can optionally create their own custom ones at this layer. As mentioned previously, Home Automation and Smart Energy are two examples of popular application profiles.

ZigBee is one of the most well-known protocols built on an IEEE 802.15.4 foundation. On top of the 802.15.4 PHY and MAC layers, ZigBee specifies its own network and security layer and application profiles. While this structure has provided a fair degree of interoperability for vendors with membership in the ZigBee Alliance, it has not provided interoperability with other IoT solutions. However, this has started to change with the release of ZigBee IP, which is discussed next.

ZigBee IP

With the introduction of ZigBee IP, the support of IEEE 802.15.4 continues, but the IP and TCP/UDP protocols and various other open standards are now supported at the network and transport layers. The ZigBee-specific layers are now found only at the top of the protocol stack for the applications.

ZigBee IP was created to embrace the open standards coming from the IETF's work on LLNs, such as IPv6, 6LoWPAN, and RPL. (These IETF standards are discussed in Chapter 5.) They provide for low-bandwidth, low-power, and cost-effective communications when connecting smart objects.

ZigBee IP is a critical part of the Smart Energy (SE) Profile 2.0 specification from the ZigBee Alliance. SE 2.0 is aimed at smart metering and residential energy management systems. In fact, ZigBee IP was designed specifically for SE 2.0 but it is not limited to this use case. Any other applications that need a standards-based IoT stack can utilize Zigbee IP. The ZigBee IP stack is shown in Figure 4-4.

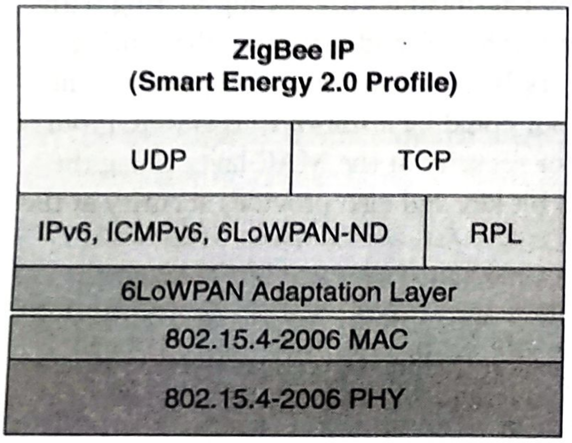


Figure 4-4 *ZigBee IP Protocol Stack*

Unlike traditional ZigBee, discussed in the previous section, ZigBee IP supports 6LoWPAN as an adaptation layer. (The 6LoWPAN protocol is covered in Chapter 5.) The 6LoWPAN mesh addressing header is not required as ZigBee IP utilizes the mesh-over or route-over method for forwarding packets. ZigBee IP requires the support of 6LoWPAN's fragmentation and header compression schemes.

At the network layer, all ZigBee IP nodes support IPv6, ICMPv6, and 6LoWPAN Neighbor Discovery (ND), and utilize RPL for the routing of packets across the mesh network. IPv6 and RPL are discussed in more detail in Chapter 5. Both TCP and UDP are also supported, to provide both connection-oriented and connectionless service.

As you can see, ZigBee IP is a compelling protocol stack offering because it is based on current IoT standards at every layer under the application layer. This opens up opportunities for ZigBee IP to integrate and interoperate on just about any 802.15.4 network with other solutions built on these open IoT standards. The following sections take a deeper dive into 802.15.4 and its PHY and MAC layers.

LoRaWAN

In recent years, a new set of wireless technologies known as Low-Power Wide-Area (LPWA) has received a lot of attention from the industry and press. Particularly well adapted for long-range and battery-powered endpoints, LPWA technologies open new business opportunities to both services providers and enterprises considering IoT solutions. This section discusses an example of an unlicensed-band LPWA technology, known as LoRaWAN, and the next section, "NB-IoT and Other LTE Variations," reviews licensed-band alternatives from the 3rd Generation Partnership Project (3GPP).

Note Other technologies could have been covered in this section of the book from an LPWA perspective, but currently this part of the IoT world is still evolving, and there are a lot of available options. We chose to cover LoRaWAN because it is one of the few options that is established, and it is backed by an industry alliance supported by a substantial number of companies.

Standardization and Alliances

Initially, LoRa was a physical layer, or Layer 1, modulation that was developed by a French company named Cycleo. Later, Cycleo was acquired by Semtech. Optimized for long-range, two-way communications and low power consumption, the technology evolved from Layer 1 to a broader scope through the creation of the LoRa Alliance. For more information on the LoRa Alliance, visit www.lora-alliance.org.

The LoRa Alliance quickly achieved industry support and currently has hundreds of members. Published LoRaWAN specifications are open and can be accessed from the LoRa Alliance website.

Semtech LoRa as a Layer 1 PHY modulation technology is available through multiple chipset vendors. To differentiate from the physical layer modulation known as LoRa, the LoRa Alliance uses the term LoRaWAN to refer to its architecture and its specifications that describe end-to-end LoRaWAN communications and protocols.

Figure 4-15 provides a high-level overview of the LoRaWAN layers. In this figure, notice that Semtech is responsible for the PHY layer, while the LoRa Alliance handles the MAC layer and regional frequency bands.

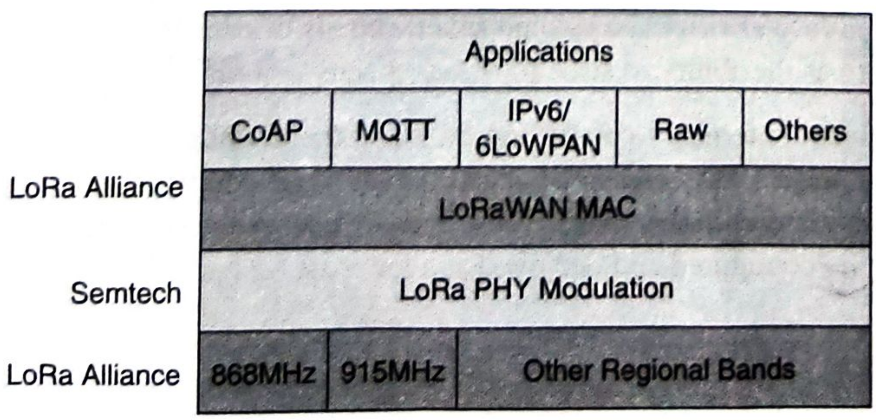


Figure 4-15 *LoRaWAN Layers*

Overall, the LoRa Alliance owns and manages the roadmap and technical development of the LoRaWAN architecture and protocol. This alliance also handles the LoRaWAN endpoint certification program and technology promotion through its certification and marketing committees.

Physical Layer

Semtech LoRa modulation is based on chirp spread spectrum modulation, which trades a lower data rate for receiver sensitivity to significantly increase the communication distance. In addition, it allows demodulation below the noise floor, offers robustness to noise and interference, and manages a single channel occupation by different spreading factors. This enables LoRa devices to receive on multiple channels in parallel.

LoRaWAN 1.0.2 regional specifications describe the use of the main unlicensed sub-GHz frequency bands of 433 MHz, 779–787 MHz, 863–870 MHz, and 902–928 MHz, as well as regional profiles for a subset of the 902–928 MHz bandwidth. For example, Australia utilizes 915–928 MHz frequency bands, while South Korea uses 920–923 MHz and Japan uses 920–928 MHz.

Note Semtech LoRa chipsets support additional frequency bands, such as 169 MHz, that are not supported by LoRaWAN specifications. Additional regional profiles and bands are under development and are being considered by the LoRa Alliance.

Understanding LoRa gateways is critical to understanding a LoRaWAN system. A LoRa gateway is deployed as the center hub of a star network architecture. It uses multiple transceivers and channels and can demodulate multiple channels at once or even demodulate multiple signals on the same channel simultaneously. LoRa gateways serve as a transparent bridge relaying data between endpoints, and the endpoints use a single-hop wireless connection to communicate with one or many gateways.

The data rate in LoRaWAN varies depending on the frequency bands and adaptive data rate (ADR). ADR is an algorithm that manages the data rate and radio signal for each endpoint. The ADR algorithm ensures that packets are delivered at the best data rate possible and that network performance is both optimal and scalable. Endpoints close to the gateways with good signal values transmit with the highest data rate, which enables a shorter transmission time over the wireless network, and the lowest transmit power. Meanwhile, endpoints at the edge of the link budget communicate at the lowest data rate and highest transmit power.

Note LoRaWAN best practices recommend the use of ADR for fixed endpoints, and a fixed data rate or spreading factor for mobile endpoints. Data rate management is not practical when mobile endpoints cause quick changes in their radio environment.

An important feature of LoRa is its ability to handle various data rates via the spreading factor. Devices with a low spreading factor (SF) achieve less distance in their communications but transmit at faster speeds, resulting in less airtime. A higher SF provides slower transmission rates but achieves a higher reliability at longer distances. Table 4-4 illustrates how LoRaWAN data rates can vary depending on the associated spreading factor for the two main frequency bands, 863–870 MHz and 902–928 MHz.

Table 4-4 *LoRaWAN Data Rate Example*

Configuration	863–870 MHz bps	902–928 MHz bps
LoRa: SF12/125 kHz	250	N/A
LoRa: SF11/125 kHz	440	N/A
LoRa: SF10/125 kHz	980	980
LoRa: SF9/125 kHz	1760	1760
LoRa: SF8/125 kHz	3125	3125
LoRa: SF7/125 kHz	5470	5470
LoRa: SF7/250 kHz	11,000	N/A
FSK: 50 kbps	50,000	N/A
LoRa: SF12/500 kHz	N/A	980
LoRa: SF11/500 kHz	N/A	1760
LoRa: SF10/500 kHz	N/A	3900
LoRa: SF9/500 kHz	N/A	7000
LoRa: SF8/500 kHz	N/A	12,500
LoRa: SF7/500 kHz	N/A	21,900

In Table 4-4, notice the relationship between SF and data rate. For example, at an SF value of 12 for 125 kHz of channel bandwidth, the data rate is 250 bps. However, when the SF is decreased to a value of 7, the data rate increases to 5470 bps.

Channel bandwidth values of 125 kHz, 250 kHz, and 500 kHz are also evident in Table 4-4. The effect of increasing the bandwidth is that faster data rates can be achieved for the same spreading factor.

MAC Layer

As mentioned previously, the MAC layer is defined in the LoRaWAN specification. This layer takes advantage of the LoRa physical layer and classifies LoRaWAN endpoints to optimize their battery life and ensure downstream communications to the LoRaWAN endpoints. The LoRaWAN specification documents three classes of LoRaWAN devices:

- **Class A:** This class is the default implementation. Optimized for battery-powered nodes, it allows bidirectional communications, where a given node is able to receive downstream traffic after transmitting. Two receive windows are available after each transmission.

- **Class B:** This class was designated "experimental" in LoRaWAN 1.0.1 until it can be better defined. A Class B node or endpoint should get additional receive windows compared to Class A, but gateways must be synchronized through a beaconing process.
- **Class C:** This class is particularly adapted for powered nodes. This classification enables a node to be continuously listening by keeping its receive window open when not transmitting.

LoRaWAN messages, either uplink or downlink, have a PHY payload composed of a 1-byte MAC header, a variable-byte MAC payload, and a MIC that is 4 bytes in length. The MAC payload size depends on the frequency band and the data rate, ranging from 59 to 230 bytes for the 863–870 MHz band and 19 to 250 bytes for the 902–928 MHz band. Figure 4-16 shows a high-level LoRaWAN MAC frame format.

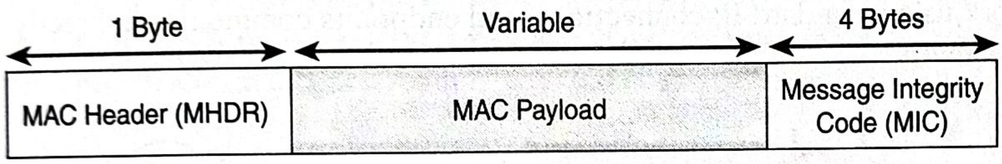


Figure 4-16 *High-Level LoRaWAN MAC Frame Format*

In version 1.0.x, LoRaWAN utilizes six MAC message types. LoRaWAN devices use join request and join accept messages for over-the-air (OTA) activation and joining the network. The other message types are unconfirmed data up/down and confirmed data up/down. A "confirmed" message is one that must be acknowledged, and "unconfirmed" signifies that the end device does not need to acknowledge. "up/down" is simply a directional notation identifying whether the message flows in the uplink or downlink path. Uplink messages are sent from endpoints to the network server and are relayed by one or more LoRaWAN gateways. Downlink messages flow from the network server to a single endpoint and are relayed by only a single gateway. Multicast over LoRaWAN is being considered for future versions.

LoRaWAN endpoints are uniquely addressable through a variety of methods, including the following:

- An endpoint can have a global end device ID or DevEUI represented as an IEEE EUI-64 address.
- An endpoint can have a global application ID or AppEUI represented as an IEEE EUI-64 address that uniquely identifies the application provider, such as the owner, of the end device.
- In a LoRaWAN network, endpoints are also known by their end device address, known as a DevAddr, a 32-bit address. The 7 most significant bits are the network identifier (NwkID), which identifies the LoRaWAN network. The 25 least significant bits are used as the network address (NwkAddr) to identify the endpoint in the network.

Note The LoRa Alliance maintains a list of companies that have registered for network identifiers (NwkIDs). The LoRa Alliance also allocates new NwkIDs. When the LoRaWAN 1.1 specification is released, a NetID field will uniquely identify a network operator. This code is also managed by the LoRa Alliance. The seven least significant bits of the NetID contain the NwkID. The NwkIDs in the DevAddr field and the NetID field are the same so the 7 most significant bits found in the DevAddr field must match the 7 least significant bits of the NetID.

Topology

LoRaWAN topology is often described as a "star of stars" topology. As shown in Figure 4-17, the infrastructure consists of endpoints exchanging packets through gateways acting as bridges, with a central LoRaWAN network server. Gateways connect to the backend network using standard IP connections, and endpoints communicate directly with one or more gateways.

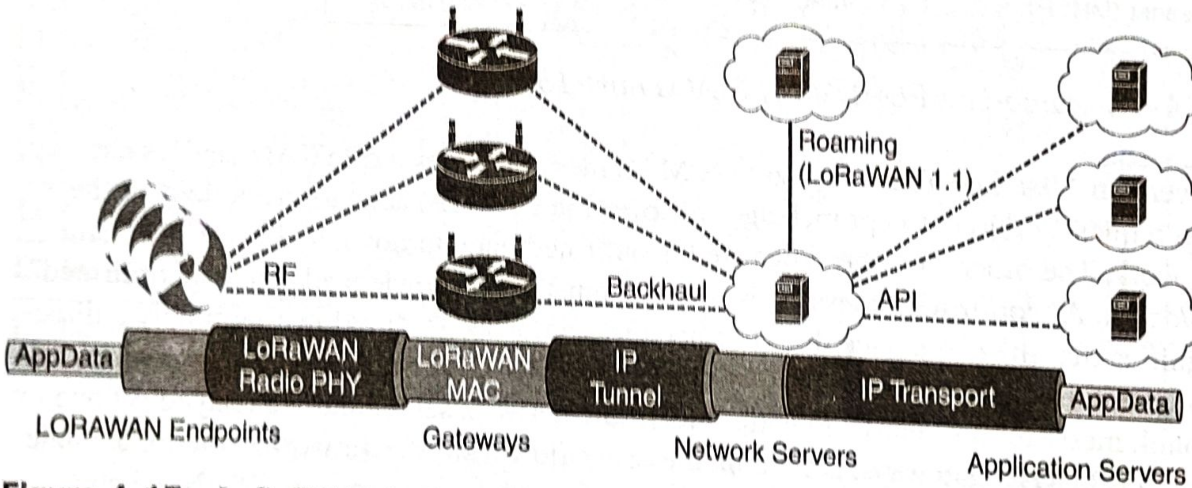


Figure 4-17 *LoRaWAN Architecture*

In Figure 4-17, LoRaWAN endpoints transport their selected application data over the LoRaWAN MAC layer on top of one of the supported PHY layer frequency bands. The application data is contained in upper protocol layers. These upper layers are not the responsibility of the LoRa Alliance, but best practices may be developed and recommended. These upper layers could just be raw data on top of the LoRaWAN MAC layer, or the data could be stacked in multiple protocols. For example, you could have upper-layer protocols, such as ZigBee Control Layer (ZCL), Constrained Application Protocol (CoAP), or Message Queuing Telemetry Transport (MQTT), with or without an IPv6/6LoWPAN layer. (The CoAP and MQTT protocols are covered in Chapter 6.)

Figure 4-17 also shows how LoRaWAN gateways act as bridges that relay between endpoints and the network servers. Multiple gateways can receive and transport the same packets. When duplicate packets are received, de-duplication is a function of the network server.

Note Semtech, developer of the LoRa PHY, has specified two generations of LoRaWAN gateways. The first generation was simple, and the next generation, known as version 2, adds new features, such as geolocation. Geolocation works by having version 2 LoRaWAN gateways share an accurate time source and then adding a high-resolution timestamp to each received LoRa packet. The endpoint's geolocation can be determined by using time differential of arrival (TDoA) algorithms.

The LoRaWAN network server manages the data rate and radio frequency (RF) of each endpoint through the adaptive data rate (ADR) algorithm. ADR is a key component of the network scalability, performance, and battery life of the endpoints. The LoRaWAN network server forwards application data to the application servers, as depicted in Figure 4-17.

In future versions of the LoRaWAN specification, roaming capabilities between LoRaWAN network servers will be added. These capabilities will enable mobile endpoints to connect and roam between different LoRaWAN network infrastructures.

Security

Security in a LoRaWAN deployment applies to different components of the architecture, as detailed in Figure 4-18. LoRaWAN endpoints must implement two layers of security, protecting communications and data privacy across the network.

The first layer, called "network security" but applied at the MAC layer, guarantees the authentication of the endpoints by the LoRaWAN network server. Also, it protects LoRaWAN packets by performing encryption based on AES.

Each endpoint implements a network session key (NwkSKey), used by both itself and the LoRaWAN network server. The NwkSKey ensures data integrity through computing and checking the MIC of every data message as well as encrypting and decrypting MAC-only data message payloads.

The second layer is an application session key (AppSKey), which performs encryption and decryption functions between the endpoint and its application server. Furthermore, it computes and checks the application-level MIC, if included. This ensures that the LoRaWAN service provider does not have access to the application payload if it is not allowed that access.

Endpoints receive their AES-128 application key (AppKey) from the application owner. This key is most likely derived from an application-specific root key exclusively known to and under the control of the application provider.

For production deployments, it is expected that the LoRaWAN gateways are protected as well, for both the LoRaWAN traffic and the network management and operations over their backhaul link(s). This can be done using traditional VPN and IPsec technologies that demonstrate scaling in traditional IT deployments. Additional security add-ons are under evaluation by the LoRaWAN Alliance for future revisions of the specification.

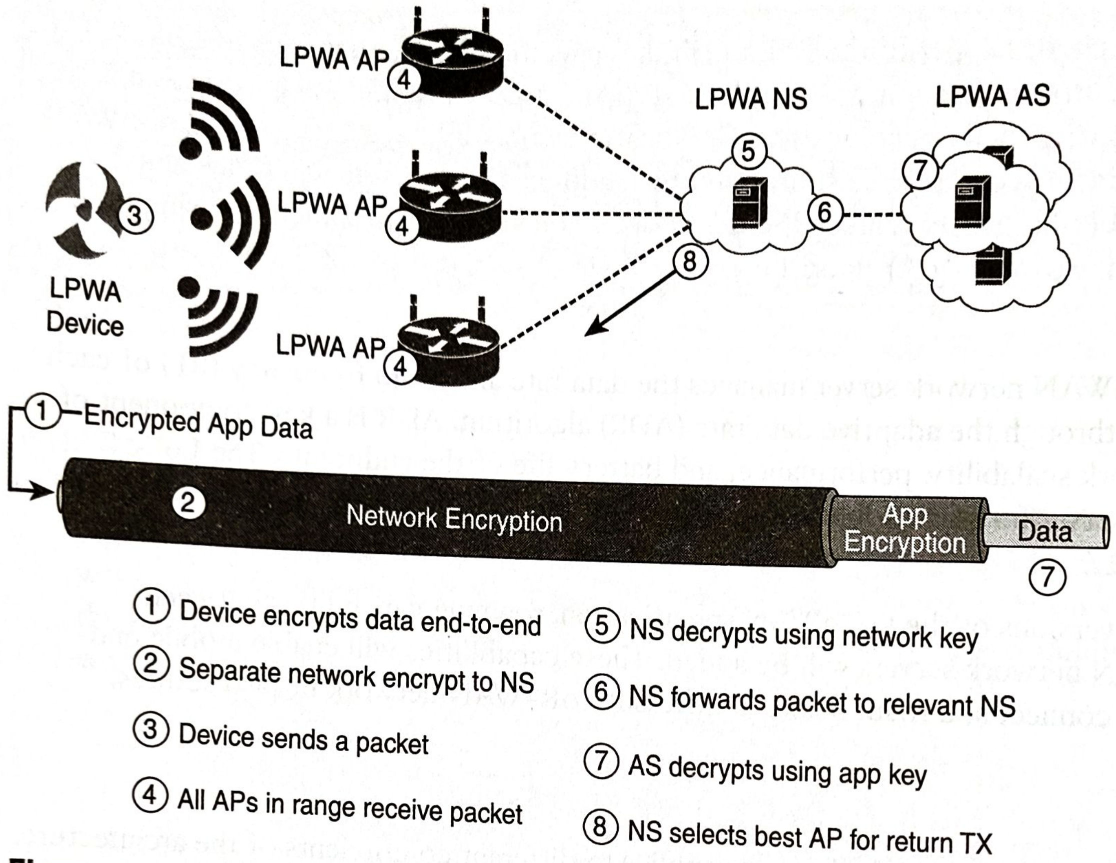


Figure 4-18 *LoRaWAN Security*

LoRaWAN endpoints attached to a LoRaWAN network must get registered and authenticated. This can be achieved through one of the two join mechanisms:

- **Activation by personalization (ABP):** Endpoints don't need to run a join procedure as their individual details, including DevAddr and the NwkSKey and AppSKey session keys, are preconfigured and stored in the end device. This same information is registered in the LoRaWAN network server.
- **Over-the-air activation (OTAA):** Endpoints are allowed to dynamically join a particular LoRaWAN network after successfully going through a join procedure. The join procedure must be done every time a session context is renewed. During the join process, which involves the sending and receiving of MAC layer join request and join accept messages, the node establishes its credentials with a LoRaWAN network server, exchanging its globally unique DevEUI, AppEUI, and AppKey. The AppKey is then used to derive the session NwkSKey and AppSKey keys.

Competitive Technologies

LPWA solutions and technologies are split between unlicensed and licensed bands. The licensed-band technologies are dedicated to mobile service providers that have acquired spectrum licenses; they are discussed in the next section. In addition, several technologies are targeting the unlicensed-band LPWA market to compete against LoRaWAN. The LPWA market is quickly evolving. Table 4-5 evaluates two of the best-established vendors known to provide LPWA options.

Table 4-5 *Unlicensed LPWA Technology Comparison*

Characteristic	LoRaWAN	Sigfox	Ingenu Onramp
Frequency bands	433 MHz, 868 MHz, 902–928 MHz	433 MHz, 868 MHz, 902–928 MHz	2.4 GHz
Modulation	Chirp spread spectrum	Ultra-narrowband	DSSS
Topology	Star of stars	Star	Star; tree supported with an RPMA extender
Data rate	250 bps–50 kbps (868 MHz) 980 bps–21.9 kbps (915 MHz)	100 bps (868 MHz) 600 bps (915 MHz)	6 kbps
Adaptive data rate	Yes	No	No
Payload	59–230 bytes (868 MHz) 19–250 bytes (915 MHz)	12 bytes	6 bytes–10 KB
Two-way communications	Yes	Partial	Yes
Geolocation	Yes (LoRa GW version 2 reference design)	No	No
Roaming	Yes (LoRaWAN 1.1)	No	Yes
Specifications	LoRA Alliance	Proprietary	Proprietary

Table 4-5 gives you a good overview of two of the most established LoRaWAN competitors. This is a good starting point, but you should perform additional research to further differentiate these technologies if you are interested in deploying an LPWAN.