

IoT Network Architecture and Design

Chapter 2 : Text Book 3

Introduction

- IoT networks require careful architectural planning, similar to building a house with blueprints.
- This chapter emphasizes scalable, secure, and manageable design.
- Architectures ensure integration of IT and OT, address scale, security, data, and legacy devices.

Drivers of IoT Architectures

- Scale: Billions of IoT devices
- Security: Larger attack surface
- Constrained Devices & Networks: Low power, limited resources
- Data Explosion: Huge volumes require management
- Legacy Device Support: Compatibility essential

IoT Standard Architectures

- oneM2M Standardized Architecture
- IoT World Forum (IoTWF) Model (7 layers)
- Aligns devices, connectivity, edge, applications, and business processes
- Defines IT and OT responsibilities

Simplified IoT Architecture

- Core IoT Functional Stack:
 - Things: Sensors & Actuators
 - Communications Network
 - Applications & Analytics
- IoT Data Management & Compute Stack:
 - Edge Computing
 - Fog Computing
 - Cloud Computing

Analytics and Business Value

- Data Analytics vs. Network Analytics
- From raw data to actionable insights
- Smart services improve efficiency, automation, and decision making
- Analytics drives business growth

Conclusion

- IoT requires unique architectures beyond traditional IT.
- A simplified functional and data stack approach makes IoT manageable.
- Edge, fog, and cloud computing are key for efficiency.
- Careful design ensures secure, scalable, and intelligent IoT solutions.

Section 2.2 – Comparing IoT Architectures

2.2 Introduction

- IoT requires standardized architectures to reduce fragmentation and ensure interoperability. This section compares two major frameworks: oneM2M and IoTWF. Both aim to address scalability, security, and heterogeneity while supporting business value creation.

2.2.1 Emergence of Standards (Part 1)

- The IoT ecosystem is fragmented with diverse devices, protocols, and platforms. ETSI initiated work on M2M communications in 2008, which laid the foundation for standardized IoT architectures.

2.2.1 Emergence of Standards (Part 2)

- The IoT World Forum, driven by Cisco and global partners, introduced a reference model to promote interoperability and scalability. These industry efforts are crucial in reducing fragmentation.

2.2.2 Foundational Concept (Part 1)

- IoT is fundamentally data-driven. Architectures focus on enabling data collection, transport, processing, and abstraction to create actionable business insights.

2.2.2 Foundational Concept (Part 2)

- The flow of IoT data: devices generate information → reliable transmission → aggregation and abstraction → analysis for decision-making. Real-time services depend on this architecture.

Transition to Standardized Architectures

- To address IoT's unique requirements, two standardized architectures are dominant: oneM2M and IoTWF.

2.2.3 oneM2M – Background

- ETSI launched oneM2M in 2012 with 13 founding members. It provides a global standard for IoT systems.

2.2.3 oneM2M – Common Services Layer

- The architecture defines a common services layer embedded in devices, supporting interoperability across applications and industries such as healthcare, utilities, and transport.

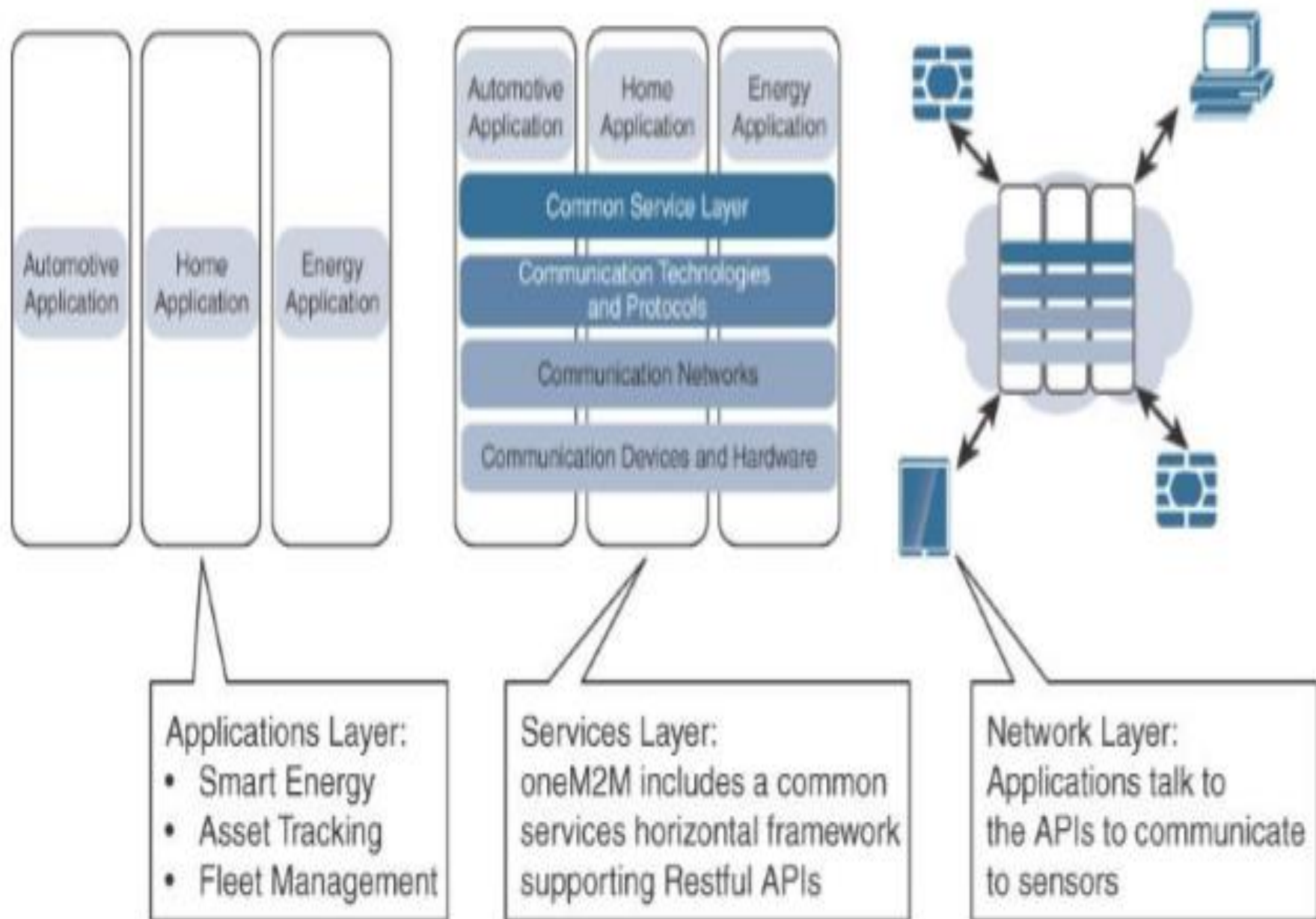


Figure 2-1 The Main Elements of the oneM2M IoT Architecture

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

Applications Layer (Verticals)

- The Applications Layer represents industry-specific IoT verticals such as Automotive, Home, and Energy Applications. Each vertical has its own data models and use cases, including Smart Energy, Asset Tracking, and Fleet Management. The purpose is to connect devices with business systems using northbound APIs, enabling integration into enterprise applications.

Services Layer (Horizontal Framework)

- The Services Layer is the core of oneM2M and provides a common horizontal framework across all vertical applications. It includes the Common Service Layer, Communication Technologies and Protocols, Communication Networks, and Communication Devices and Hardware. This layer ensures interoperability and supports RESTful APIs for cross-industry integration.

Common Services Layer – oneM2M Goal

- The Common Services Layer provides essential services such as security, data management, and device management. Its goal is to create a common M2M Service Layer that can be embedded in devices, gateways, and servers, linking field devices to application servers (often in the cloud). This supports industries such as telematics, healthcare, utilities, industrial automation, and smart homes.

Network Layer (Devices and Communications)

- The Network Layer encompasses IoT devices and the communication infrastructure linking them. Applications rely on APIs in this layer to communicate with sensors. Devices can interact peer-to-peer, through Field Area Networks (FANs), or via Gateways that connect to core networks. Technologies include IEEE 802.15.4 (mesh), IEEE 802.11ah (point-to-multipoint), and IEEE 1901 (power-line communications). The Gateway is the demarcation point between the device and network domains.

Purpose of oneM2M

- The purpose of oneM2M is to unify fragmented IoT ecosystems by providing a horizontal services framework. It allows applications across diverse verticals to interoperate seamlessly. For example, a LoRaWAN sensor network can interwork with a building management system such as BACnet using oneM2M's RESTful APIs and services. This makes IoT scalable, flexible, and interoperable across industries.

Challenge	Description	IoT Architectural Change Required
Scale	The massive scale of IoT endpoints (sensors) is far beyond that of typical IT networks.	The IPv4 address space has reached exhaustion and is unable to meet IoT's scalability requirements. Scale can be met only by using IPv6. IT networks continue to use IPv4 through features like Network Address Translation (NAT).
Security	IoT devices, especially those on wireless sensor networks (WSNs), are often physically exposed to the world.	Security is required at every level of the IoT network. Every IoT endpoint node on the network must be part of the overall security strategy and must support device-level authentication and link encryption. It must also be easy to deploy with some type of a zero-touch deployment model.
Devices and networks constrained by power, CPU, memory, and link speed	Due to the massive scale and longer distances, the networks are often constrained, lossy, and capable of supporting only minimal data rates (tens of bps to hundreds of Kbps).	New last-mile wireless technologies are needed to support constrained IoT devices over long distances. The network is also constrained, meaning modifications need to be made to traditional network-layer transport mechanisms.
The massive volume of data generated	The sensors generate a massive amount of data on a daily basis, causing network bottlenecks and slow analytics in the cloud.	Data analytics capabilities need to be distributed throughout the IoT network, from the edge to the cloud. In traditional IT networks, analytics and applications typically run only in the cloud.

The massive volume of data generated	The sensors generate a massive amount of data on a daily basis, causing network bottlenecks and slow analytics in the cloud.	Data analytics capabilities need to be distributed throughout the IoT network, from the edge to the cloud. In traditional IT networks, analytics and applications typically run only in the cloud.
Support for legacy devices	An IoT network often comprises a collection of modern, IP-capable endpoints as well as legacy, non-IP devices that rely on serial or proprietary protocols.	Digital transformation is a long process that may take many years, and IoT networks need to support protocol translation and/or tunneling mechanisms to support legacy protocols over standards-based protocols, such as Ethernet and IP.
The need for data to be analyzed in real time	Whereas traditional IT networks perform scheduled batch processing of data, IoT data needs to be analyzed and responded to in real-time.	Analytics software needs to be positioned closer to the edge and should support real-time streaming analytics. Traditional IT analytics software (such as relational databases or even Hadoop), are better suited to batch-level analytics that occur after the fact.

Table 2-1 *IoT Architectural Drivers*

Table: oneM2M Features

- Key Features:
 - - Horizontal services framework.
 - - Device/application interoperability.
 - - Cross-domain integration.
 - - Scalable architecture.
- Explanation: oneM2M supports broad IoT deployments.

2.2.4 IoTWF – Introduction

- IoTWF, supported by Cisco, proposes a 7-layer architecture. It provides a modular framework for IoT deployments.

IoTWF Layers

- Layer 1: Physical devices – sensors, controllers.
- Layer 2: Connectivity – reliable data transmission.
- Layer 3: Edge Computing – local data processing.
- Layer 4: Data Accumulation – stores raw data.
- Layer 5: Data Abstraction – converts data to usable formats.
- Layer 6: Applications – data-driven applications.
- Layer 7: Collaboration & Processes – integrates IoT insights into organizational workflows.

Figure 2-2 – IoT Reference Model (IoT World Forum)

Levels

- 7 Collaboration & Processes**
(Involving People & Business Processes)
- 6 Application**
(Reporting, Analytics, Control)
- 5 Data Abstraction**
(Aggregation & Access)
- 4 Data Accumulation**
(Storage)
- 3 Edge Computing**
(Data Element Analysis & Transformation)
- 2 Connectivity**
(Communication & Processing Units)
- 1 Physical Devices & Controllers**
(The "Things" in IoT)



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

Orientation of the IoTWF Model

- The IoTWF Reference Model divides IoT into seven layers. Data generally flows northbound from devices at the edge (Layer 1) to the center (Layer 7). Control flows southbound from the center to the devices. This layered approach ensures modularity, interoperability, and clarity.

Layer 1 – Physical Devices & Controllers

- This layer represents the 'things' in IoT: sensors, devices, and controllers. They generate data by sensing the environment and can also act as actuators. Devices vary from small sensors to large industrial machines, forming the foundation of IoT.

Layer 2 – Connectivity

- Provides communication between devices and higher layers. Ensures reliable and timely data transmission. Includes last-mile networks (e.g., Zigbee, Wi-Fi), gateways, and backhaul (e.g., cellular, MPLS). Handles communication protocols and processing units to move data upward.

Layer 3 – Edge Computing

- Also known as fog computing, this layer processes data close to the source. It reduces latency and bandwidth usage. Functions include filtering, aggregation, threshold checks, and preliminary analytics. Example: a gateway detecting anomalies locally without sending all raw data to the cloud.

Layer 4 – Data Accumulation

- Acts as a temporary holding area for IoT data before processing. Collects and stores raw data from multiple devices. Ensures persistence for future access and analysis. Examples include distributed storage systems and databases.

Layer 5 – Data Abstraction

- Converts raw stored data into usable formats. Provides aggregation, indexing, and access mechanisms. Abstracts away heterogeneity so applications can query data uniformly. Ensures scalability and efficient data retrieval.

Layer 6 – Application

- Provides applications that use IoT data, including dashboards, analytics engines, and predictive maintenance tools. Can be industry-specific applications such as healthcare monitoring or fleet management. Turns data into usable information for end-users.

Layer 7 – Collaboration & Processes

- This layer integrates IoT insights into business processes. Data-driven decisions, workflow automation, and human collaboration occur here. Examples include supply chain automation, scheduling maintenance, and customer-facing services. Ensures IoT contributes real business value.

Importance of the IoTWF Model

- The IoTWF model decomposes IoT into smaller parts, enabling modular design and deployment. Different vendors can implement different layers while maintaining interoperability. It enforces a tiered security model at each layer and highlights IoT's role in enabling business transformation.

Figure 2-3 – IoT Reference Model Connectivity Layer Functions

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

Layer 2 Functions:

- Communications Between Layer 1 Devices
- Reliable Delivery of Information Across the Network
- Switching and Routing
- Translation Between Protocols
- Network Level Security



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

Position of Layer 2 in IoTWF Model

- The Connectivity Layer (Layer 2) sits above Physical Devices (Layer 1) and below Edge Computing (Layer 3). It forms the communication backbone that links IoT devices with higher-level processing and applications.

Function 1 – Communications Between Layer 1 Devices

- This function ensures that physical devices such as sensors, actuators, and controllers can exchange data. It covers both device-to-device communication and device-to-network communication, enabling seamless interaction at the edge.

Function 2 – Reliable Delivery of Information

- This function guarantees the accurate and timely delivery of IoT data packets across the network. It addresses issues like packet loss, interference, and latency, ensuring that IoT applications receive consistent data streams.

Function 3 – Switching and Routing

- Switching and routing ensure that data packets follow optimal paths through the network. In large IoT deployments such as smart cities, routing is crucial for handling massive numbers of devices and ensuring efficient communication.

Function 4 – Translation Between Protocols

- IoT devices use multiple communication standards such as Zigbee, Wi-Fi, LTE, Bluetooth, and LoRa. The Connectivity Layer translates between these protocols, ensuring interoperability among heterogeneous devices.

Function 5 – Network Level Security

- This function enforces security at the network layer through authentication, encryption, and integrity checks. It protects IoT data in transit against threats such as eavesdropping, spoofing, and tampering.

Importance of the Connectivity Layer

- The Connectivity Layer is critical because IoT devices are heterogeneous and often incompatible. By providing routing, protocol translation, and security, this layer ensures that higher layers receive reliable, secure, and unified data flows.

Table: IoTWF Layer Functions

- Table lists each IoTWF layer and its main functions, such as device management, data storage, abstraction, and analytics.

Figure 2-4 – IoT Reference Model Layer 3 (Edge/Fog Computing)

③ **Edge (Fog) Computing**
(Data Element Analysis and Transformation)

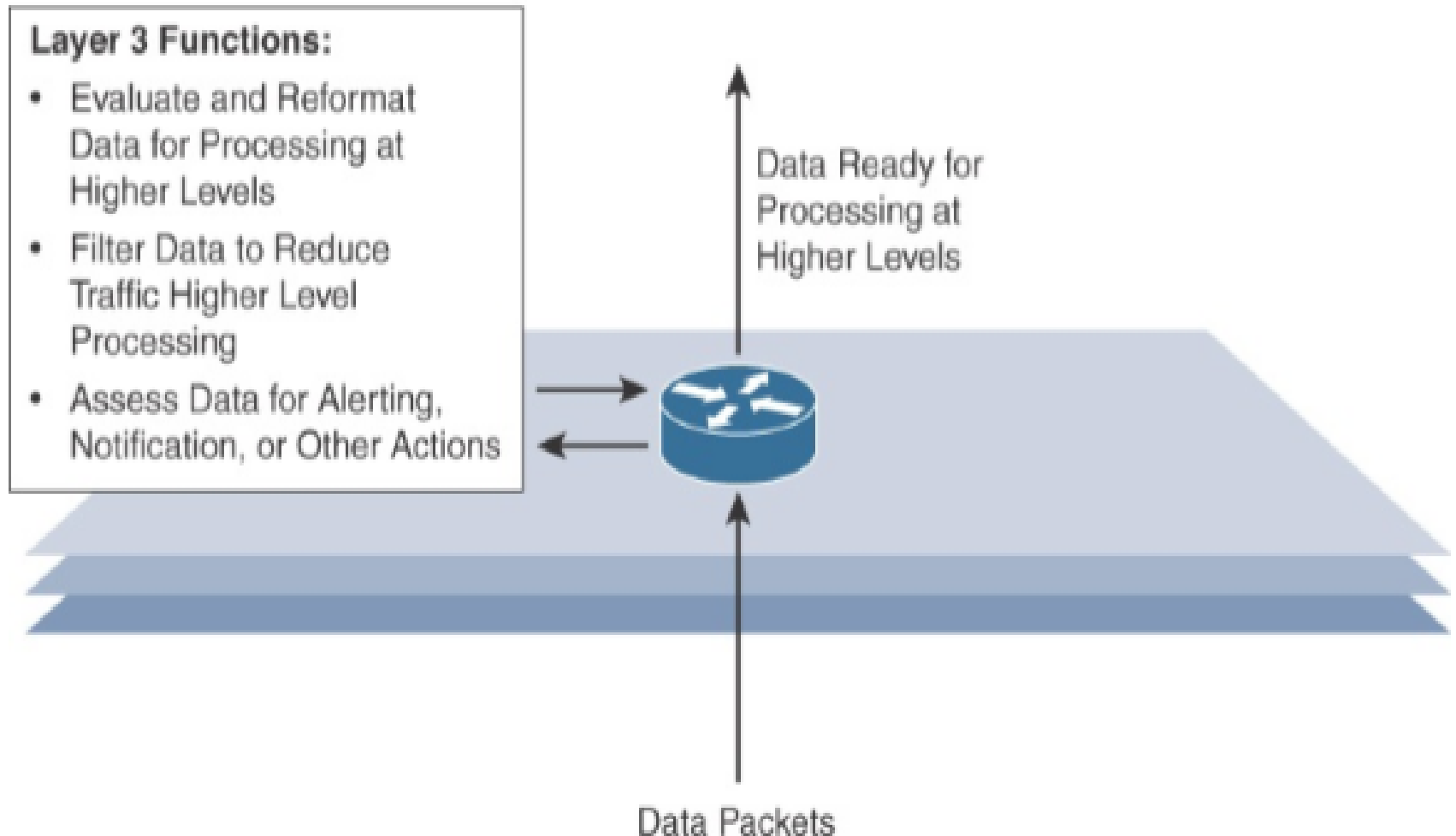


Figure 2-4 *IoT Reference Model Layer 3 Functions*

Position of Layer 3 in IoTWF Model

- Layer 3, called Edge or Fog Computing, sits above Connectivity (Layer 2) and below Data Accumulation (Layer 4). Its primary role is to process information as early and as close to the source as possible, reducing the burden on higher layers.

Function 1 – Evaluate and Reformat Data

- Raw data from IoT devices often comes in diverse formats. At this layer, data is cleaned, normalized, and reformatted to be compatible with higher layers. Example: converting proprietary sensor data into standardized formats like JSON or XML.

Function 2 – Filter Data to Reduce Traffic

- Not all raw sensor data is useful for higher-level processing. The Edge layer filters out redundant or irrelevant data, ensuring only important information is sent upstream. This reduces bandwidth usage, network congestion, and storage requirements. Example: only sending temperature anomalies instead of all readings.

Function 3 – Assess Data for Alerts and Actions

- The Edge layer can analyze data in real-time to identify threshold breaches or abnormal patterns. This allows immediate alerts, notifications, or automated actions at the edge without needing cloud involvement. Example: shutting down a machine when vibration exceeds safe thresholds.

Importance of Edge/Fog Computing

- Edge computing reduces latency by processing data locally, optimizes bandwidth by transmitting only filtered data, and increases reliability by enabling IoT devices to function even if cloud connectivity is interrupted. It is vital for real-time IoT use cases in healthcare, smart grids, autonomous vehicles, and industrial automation.

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.

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

IoT Reference Model	Description
Purdue Model for Control Hierarchy	<p>The Purdue Model for Control Hierarchy (see www.cisco.com/c/en/us/td/docs/solutions/Verticals/EttF/EttFDIG/ch2_EttF.pdf) is a common and well-understood model that segments devices and equipment into hierarchical levels and functions. It is used as the basis for ISA-95 for control hierarchy, and in turn for the IEC-62443 (formerly ISA-99) cyber security standard. It has been used as a base for many IoT-related models and standards across industry. The Purdue Model's application to IoT is discussed in detail in Chapter 9, "Manufacturing," and in Chapter 10, "Oil & Gas."</p>
Industrial Internet Reference Architecture (IIRA) by Industrial Internet Consortium (IIC)	<p>The IIRA is a standards-based open architecture for Industrial Internet Systems (IISs). To maximize its value, the IIRA has broad industry applicability to drive interoperability, to map applicable technologies, and to guide technology and standard development. The description and representation of the architecture are generic and at a high level of abstraction to support the requisite broad industry applicability. The IIRA distills and abstracts common characteristics, features and patterns from use cases well understood at this time, predominantly those that have been defined in the IIC.</p> <p>For more information, see www.iiconsortium.org/IIRA.htm.</p>
Internet of Things–Architecture (IoT-A)	<p>IoT-A created an IoT architectural reference model and defined an initial set of key building blocks that are foundational in fostering the emerging Internet of Things. Using an experimental paradigm, IoT-A combined top-down reasoning about architectural principles and design guidelines with simulation and prototyping in exploring the technical consequences of architectural design choices.</p> <p>For more information, see https://vdivde-it.de/en.</p>

Table 2-3 *Alternative IoT Reference Models*

Figure 2-5 – IoT Reference Model Separation of IT and OT

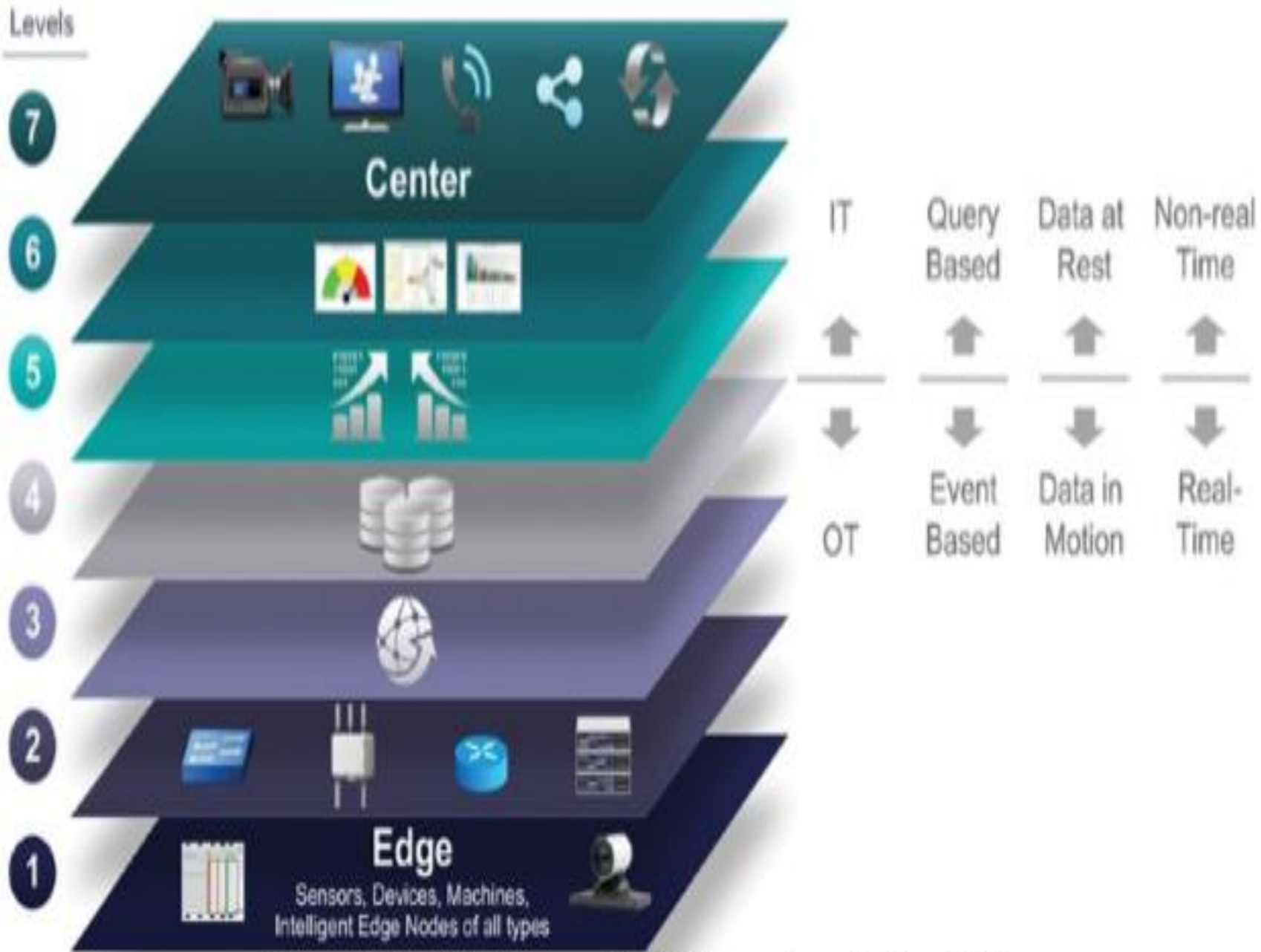


Figure 2-5 IoT Reference Model Separation of IT and OT

Context of Figure 2-5

- This figure extends the IoTWF 7-layer reference model by showing the separation of IT (Information Technology) and OT (Operational Technology). It highlights their distinct characteristics in terms of data type, time sensitivity, and processing models.

Operational Technology (OT) Layers

- Layers 1–3 (Edge, Connectivity, Edge Computing) belong to OT. They are event-based, handle continuous data-in-motion, and require real-time responsiveness. Example: machine safety systems or real-time sensor alerts.

Information Technology (IT) Layers

- Layers 5–7 (Data Abstraction, Applications, Collaboration & Processes) belong to IT. They are query-based, work primarily with data-at-rest, and operate in non-real-time contexts. Example: dashboards, reporting systems, and business analytics.

The Middle Ground – Data Accumulation (Layer 4)

- Layer 4 (Data Accumulation/Storage) is the boundary between OT and IT. It converts data-in-motion from OT into stored data-at-rest for IT. This enables the transition between real-time operations and longer-term analytics.

Why IT/OT Separation Matters

- IT and OT historically operated separately, but IoT integrates them. OT ensures operational safety and real-time control, while IT ensures data management and business optimization. IoT convergence allows both domains to complement each other while respecting their different requirements.

2.2.5 Common Goals

- Both oneM2M and IoTWF share goals: interoperability, scalability, layered security, IT/OT integration. They provide foundations for robust IoT ecosystems.

IoT Network Architecture and Design

Section 2.3 – A Simplified IoT Architecture

2.3.1 Introduction

- After studying complex frameworks like oneM2M and IoTWF, this section introduces a simplified IoT architecture. It reduces IoT into two parallel stacks: the Core IoT Functional Stack and the IoT Data Management and Compute Stack. The purpose is to provide a clear foundation for understanding IoT systems.

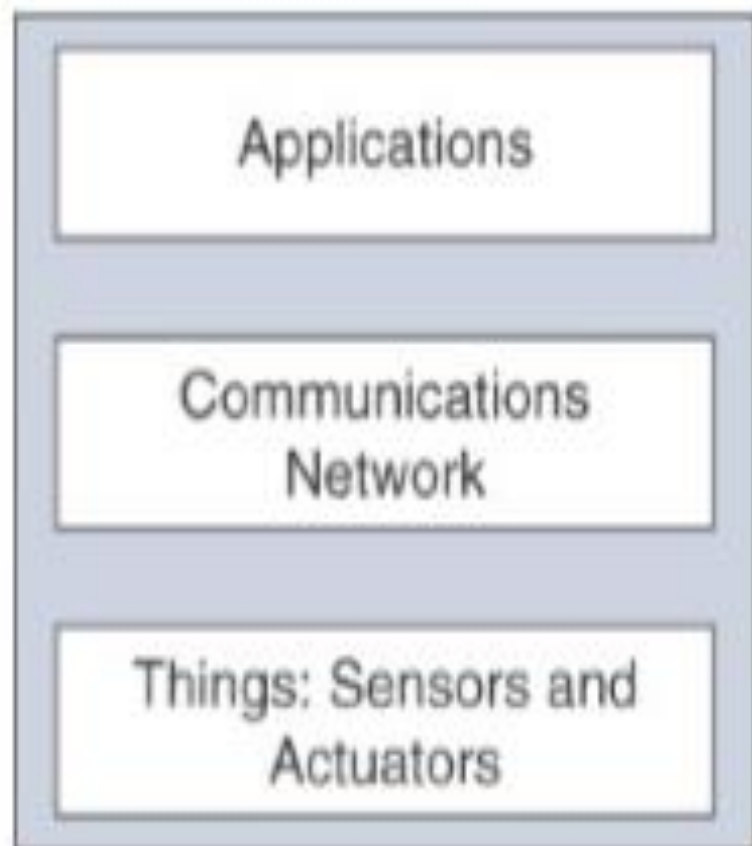
2.3.2 Core IoT Functional Stack

This stack has three layers:

1. Things (Sensors & Actuators): IoT endpoints generating or acting on data.
2. Communications Network: Includes Access, Gateways & Backhaul, Transport, and Network Management.
3. Applications & Analytics: Provides applications, visualization, and automation, not only in the cloud but also at edge and fog layers.

Figure 2-6 – Simplified IoT Architecture

Core IoT Functional Stack



IoT Data Management and Compute Stack

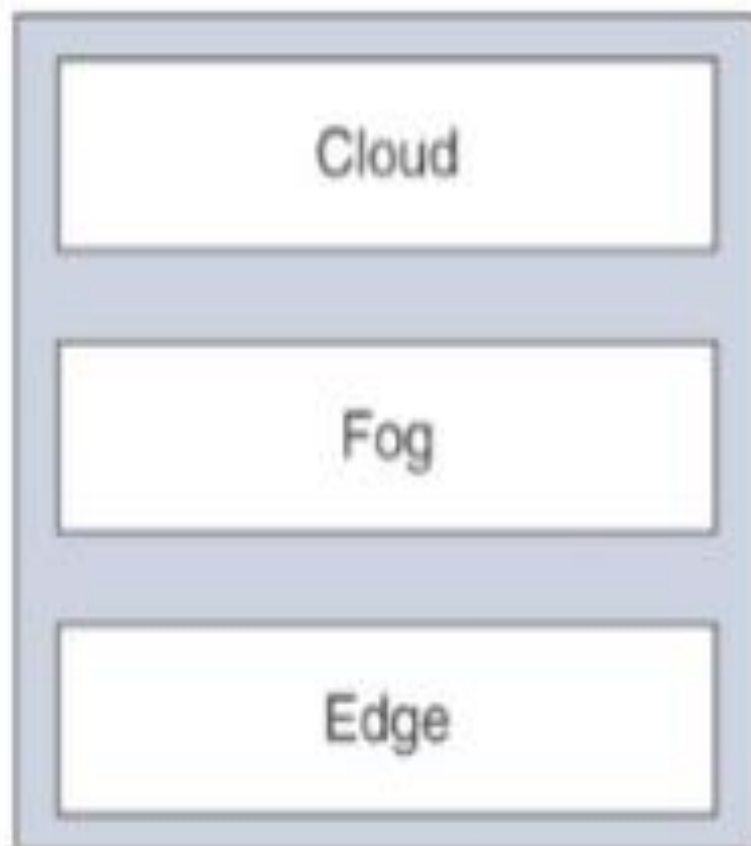


Figure 2-6 *Simplified IoT Architecture*

Purpose of the Simplified Model

- This simplified IoT model organizes IoT into two parallel stacks: the Core IoT Functional Stack and the IoT Data Management and Compute Stack. It captures the essential building blocks of IoT without unnecessary complexity.

Core IoT Functional Stack

- This stack represents the fundamental IoT functions:
 - - Things: Sensors and Actuators – capture data and act on commands.
 - - Communications Network – enables connectivity through Wi-Fi, Zigbee, LoRa, Bluetooth, or cellular.
 - - Applications – software solutions such as energy management, predictive maintenance, and healthcare monitoring.

IoT Data Management and Compute Stack

- This stack represents IoT data processing and storage:
 - - Edge – processes data locally, filtering and aggregating before sending upstream.
 - - Fog – intermediate layer for localized compute/storage, reduces latency and bandwidth.
 - - Cloud – centralized processing and storage, supports advanced analytics, ML, and enterprise integration.

Security Across the Stacks

- Security spans both stacks vertically. It must protect devices, networks, storage, and applications, ensuring confidentiality, integrity, and availability of IoT data. Security is applied end-to-end across the entire architecture.

Importance of the Simplified IoT Architecture

- This model distills complex architectures into a clear structure. It emphasizes that IoT requires both Core Functional Stack (devices, network, apps) and Data Management Stack (edge, fog, cloud). Security integrates the two, making the system scalable, reliable, and trustworthy.

Figure 2-7 – Expanded View of the Simplified IoT Architecture

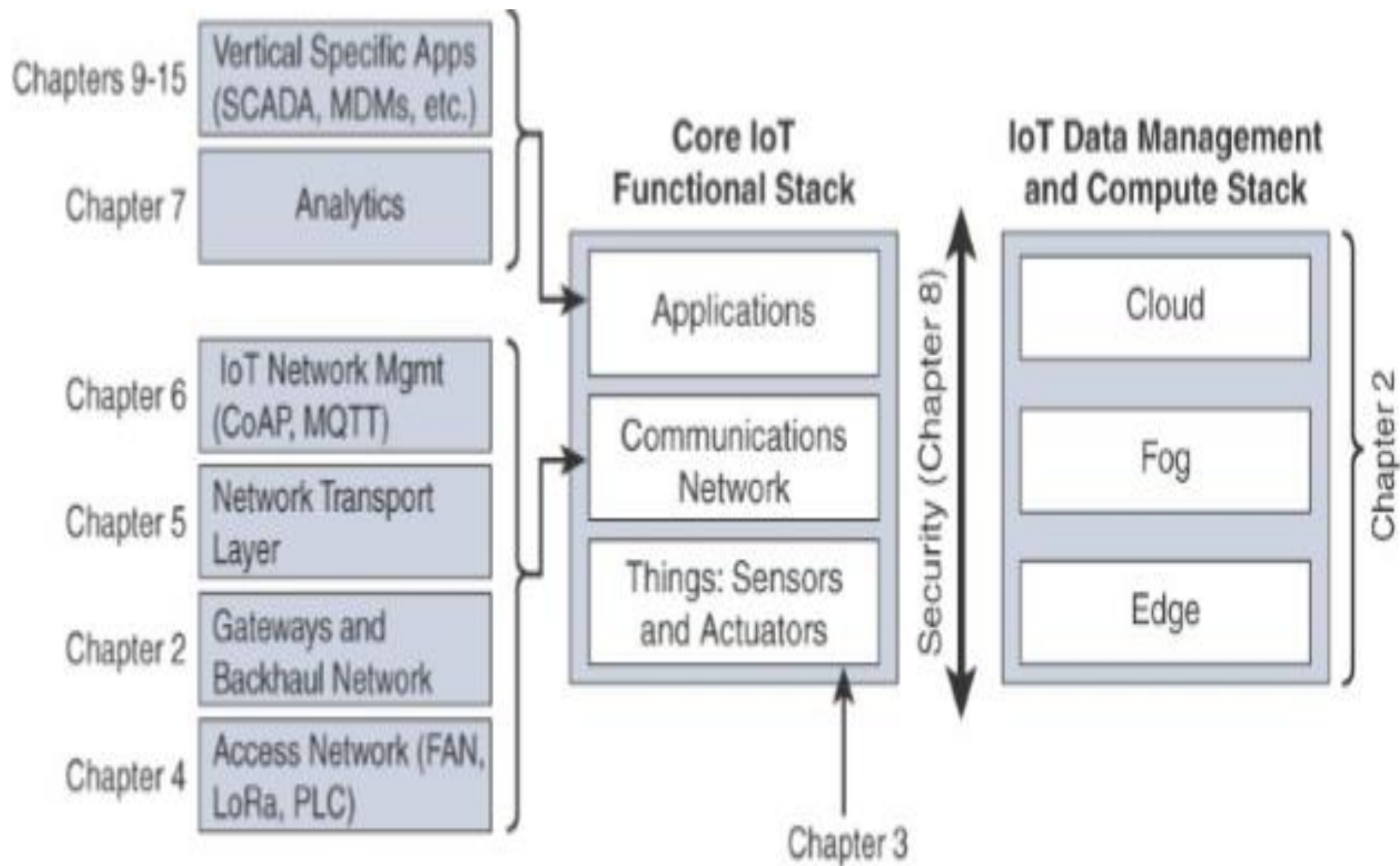


Figure 2-7 Expanded View of the Simplified IoT Architecture

Purpose of the Expanded Model

- This figure expands the simplified IoT architecture by breaking down the communications network into detailed layers, mapping textbook chapters to components, and emphasizing analytics and security as integral parts of the IoT ecosystem.

Core IoT Functional Stack

- The Core Functional Stack includes Applications, Communications Network, and Things (Sensors/Actuators). In this expanded model, the Communications Network is subdivided into multiple layers for clarity.

Expanded Communication Layers – Part 1

- 1. Access Network (Chapter 4): Provides last-mile connectivity via FAN, LoRa, PLC, etc.
- 2. Gateways and Backhaul Network (Chapter 2): Connects local networks to core networks and the internet, ensuring reliable upstream data transfer.

Expanded Communication Layers – Part 2

- 3. Network Transport Layer (Chapter 5):
Handles data transport protocols, switching, and routing for efficient delivery.
- 4. IoT Network Management (Chapter 6):
Includes protocols like CoAP and MQTT, enabling device management, registration, addressing, and resource optimization.

Applications Layer (Chapters 9–15)

- Covers general and industry-specific IoT applications. Examples include Smart Energy, Healthcare, Industrial Automation, SCADA, and MDMs. These applications provide business value by transforming IoT data into actionable insights.

IoT Data Management and Compute Stack (Chapter 2)

- This stack includes Edge (local data processing), Fog (distributed/local computing), and Cloud (centralized analytics and storage). It ensures scalability and integration with enterprise systems.

Analytics (Chapter 7)

- Analytics sits above applications, providing tools for data analysis, visualization, and decision-making. It transforms IoT data into actionable intelligence for operations and business processes.

Security (Chapter 8)

- Security spans across all layers and both stacks. It protects devices, networks, applications, storage, and cloud systems. Security is not an isolated block but an end-to-end concern ensuring confidentiality, integrity, and availability.

Conclusion – Importance of the Expanded Model

- Figure 2-7 highlights how IoT integrates functional and data management stacks with security and analytics. By mapping layers to textbook chapters, it reinforces the learning structure and clarifies how each element supports IoT ecosystems.

Core IoT Functional Stack

Overview of the Core IoT Functional Stack

- IoT networks are built around smart objects ('things') that perform functions and deliver connected services. These objects act based on contextual information and configured goals. They may be self-contained but often interact with external systems and management platforms for data exchange and guidance.

Things Layer (Sensors and Actuators)

- At this layer, physical devices operate within environmental constraints while providing needed information. These devices can sense, collect, and sometimes act upon data, forming the foundation of IoT systems.

Communications Network Layer

- When smart objects are not self-contained, they must connect externally. This often uses wireless technologies. The communications network is divided into four sublayers for efficiency and interoperability.

Access Network Sublayer

- Represents the last mile of the IoT network. Includes wireless technologies such as 802.11ah, 802.15.4g, and LoRa, though some sensors may be wired. Provides direct connectivity between sensors and the IoT system.

Gateways and Backhaul Network Sublayer

- Gateways organize multiple smart objects in an area, collecting their data. They forward information through backhaul links to central stations. Gateways perform application-layer functions and act as routers in IP networks.

Network Transport Sublayer

- Implements network and transport protocols like IP and UDP. Supports diverse devices and transmission media, ensuring reliable communication.

IoT Network Management Sublayer

- Provides management protocols to enable data exchange between headend applications and sensors. Examples: CoAP and MQTT, which ensure efficient communication and resource management.

Application and Analytics Layer

- At this top layer, applications process collected data. They control smart objects, make intelligent decisions, and adapt system behaviors. Analytics transform IoT data into actionable insights for operations and business processes.

Layer 1: Things (Classification of Smart Objects)

- Smart objects vary in type, shape, and requirements. Their classification drives protocol and architecture choices. Key classification dimensions include power source, mobility, reporting frequency, data richness, reporting range, and density.

Classification – Power Source

- Smart objects may be battery-powered or line-powered. Battery-powered devices are mobile but limited in lifetime and transmission power. Line-powered devices have continuous energy supply but less mobility.

Classification – Mobility

- Smart objects can be mobile (e.g., attached to moving goods or relocated sensors) or static (e.g., embedded road sensors). Mobility range and frequency influence power source choices and communication technologies.

Classification – Reporting Frequency

- Some sensors report rarely (e.g., rust sensor once per month), while others report frequently (e.g., motion sensors hundreds of times per second). High reporting frequency increases energy use and may constrain transmission range.

Classification – Data Richness

- Smart objects may produce simple or rich datasets. Example: a humidity sensor may report one daily value, while an engine sensor can output hundreds of parameters. Richer data increases energy use and throughput requirements.

Classification – Reporting Range

- The required communication distance influences technology choices. Example: fitness bands connect only a few meters, while road-embedded moisture sensors may need to reach hundreds of meters.

Classification – Object Density

- Density depends on the number of objects in a given area. Examples: sparse oil pipeline sensors vs. thousands of sensors in observatories or industrial sites. Density influences gateway design and network architecture.

Conclusion – Core IoT Functional Stack

- IoT systems integrate smart objects, communication networks, and applications. The classification of devices influences architecture choices and technologies used. These foundational layers ensure that IoT networks are scalable, flexible, and aligned with specific industry needs.

Figure 2-8 – Sensor Applications Based on Mobility and Throughput

Industrial (Pumps, Motors, etc.)
Environment (Weather Sensors, etc.)
Home (Fire and Safety, Security, Control)
Retail (Vending Systems, PoS, Signage)

Vehicle Telematics, Fleet Management,
Battlefield Communications

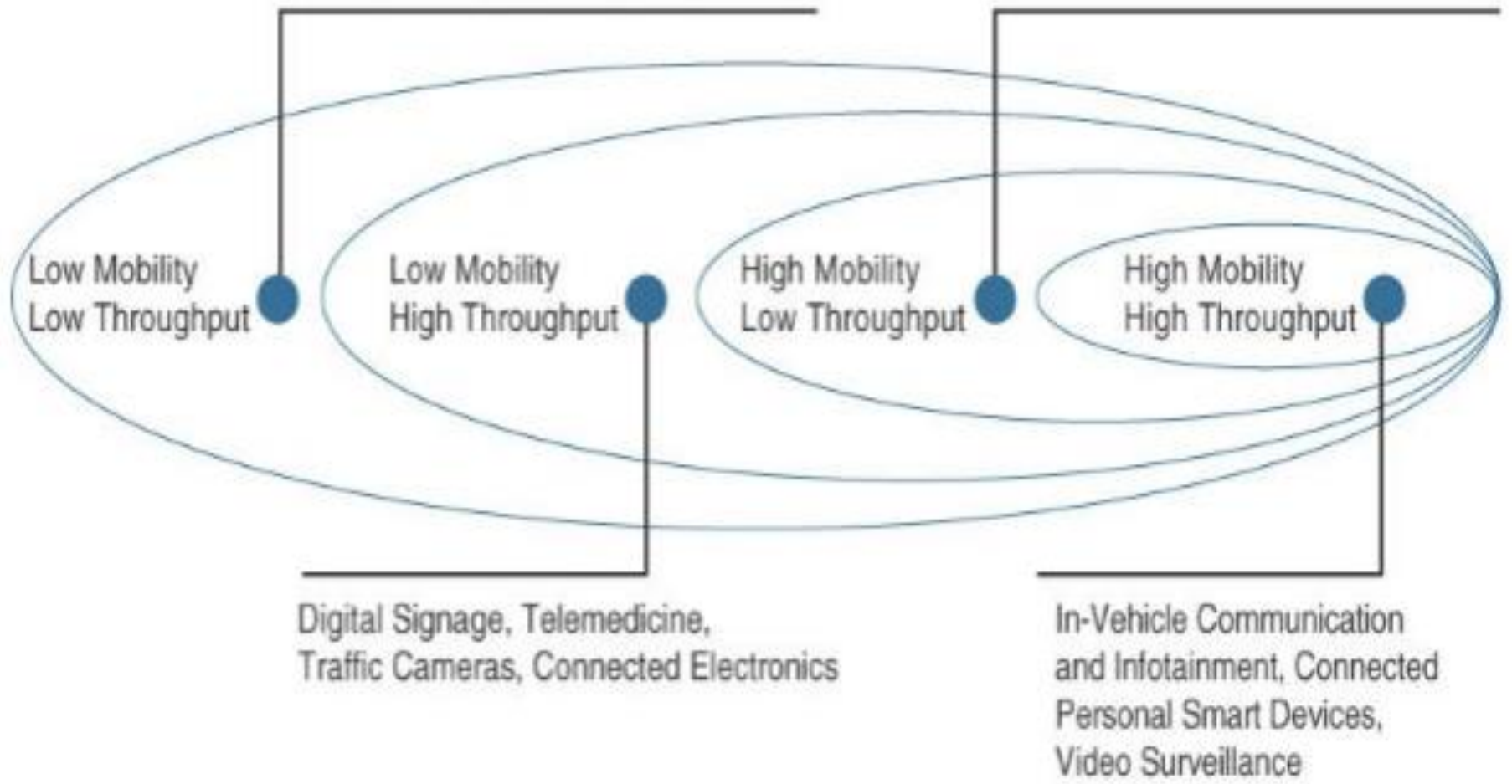


Figure 2-8 Example of Sensor Applications Based on Mobility and Throughput

Context of Figure 2-8

- IoT applications can be classified based on two key dimensions: mobility (static vs mobile) and throughput (low vs high). This figure maps real-world sensor applications to these categories, helping select the right technologies and architectures.

Low Mobility – Low Throughput

- Devices are static and produce small amounts of data at infrequent intervals.
- Examples:
 - - Industrial equipment sensors (pumps, motors)
 - - Environmental sensors (weather stations)
 - - Home safety/security sensors
 - - Retail devices (POS, vending machines, signage)
- Characteristics: Long life, stability, energy efficiency.

Low Mobility – High Throughput

- Devices are fixed but generate large continuous data volumes.
- Examples:
 - - Digital signage
 - - Telemedicine devices
 - - Traffic cameras
 - - Connected electronics
- Characteristics: Require high bandwidth and reliable, high-capacity connectivity.

High Mobility – Low Throughput

- Devices are mobile but produce small volumes of data.
- Examples:
 - - Vehicle telematics
 - - Fleet management
 - - Battlefield communication sensors
- Characteristics: Need wide-area wireless connectivity and power efficiency for continuous operation.

High Mobility – High Throughput

- Devices are mobile and generate large volumes of data continuously.
- Examples:
 - - In-vehicle infotainment systems
 - - Connected personal smart devices (wearables, AR/VR)
 - - Video surveillance on moving platforms (e.g., drones)
- Characteristics: Very demanding on bandwidth and reliability, critical for transport, defense, and security.

Importance of the Classification

- This classification helps IoT architects determine the most suitable communication technologies.
 - Low mobility/low throughput: LPWAN (LoRa, Zigbee)
 - High mobility/high throughput: 5G, satellite links
- By aligning mobility and throughput needs, IoT systems achieve better scalability, efficiency, and reliability.

Layer 2 – Communications Network Layer

Introduction

- Once smart objects have been classified by form factor, transmission range, and throughput needs, they must be connected for communication. The Communications Network Layer provides this connectivity, forming the bridge between IoT devices and higher-level systems.

IoT vs IT Environments

- Compute and network assets in IoT differ significantly from those in IT environments. The differences are driven by physical environments and operational requirements. IoT devices are built to withstand more rugged conditions and operational variances compared to IT systems.

Environmental Challenges – Temperature and Humidity

- IoT deployments often face harsh environmental conditions. Devices must withstand extremes of temperature, such as the heat of the Arabian Gulf or the cold of the Alaskan North Slope. Humidity variations can also affect system longevity, from wet delta regions to arid deserts, and in some cases devices may face direct liquid contact in marine deployments.

Environmental Challenges – Vibration and Particulates

- IoT devices may encounter constant vibration, sudden shocks, or seismic activity. For example, manufacturing systems face low-amplitude but continuous vibrations, while mobile systems face high-impact forces. Dust and particulates can also accumulate, and fanless cooling solutions with larger heat fins are often used to reduce buildup.

Environmental Challenges – Corrosive and Hazardous Conditions

- In hazardous environments, caustic materials may corrode connections and reduce heat transfer efficiency. In areas with volatile gases, spark suppression is critical to prevent explosions. Such conditions demand robust, specialized designs for IoT communication devices.

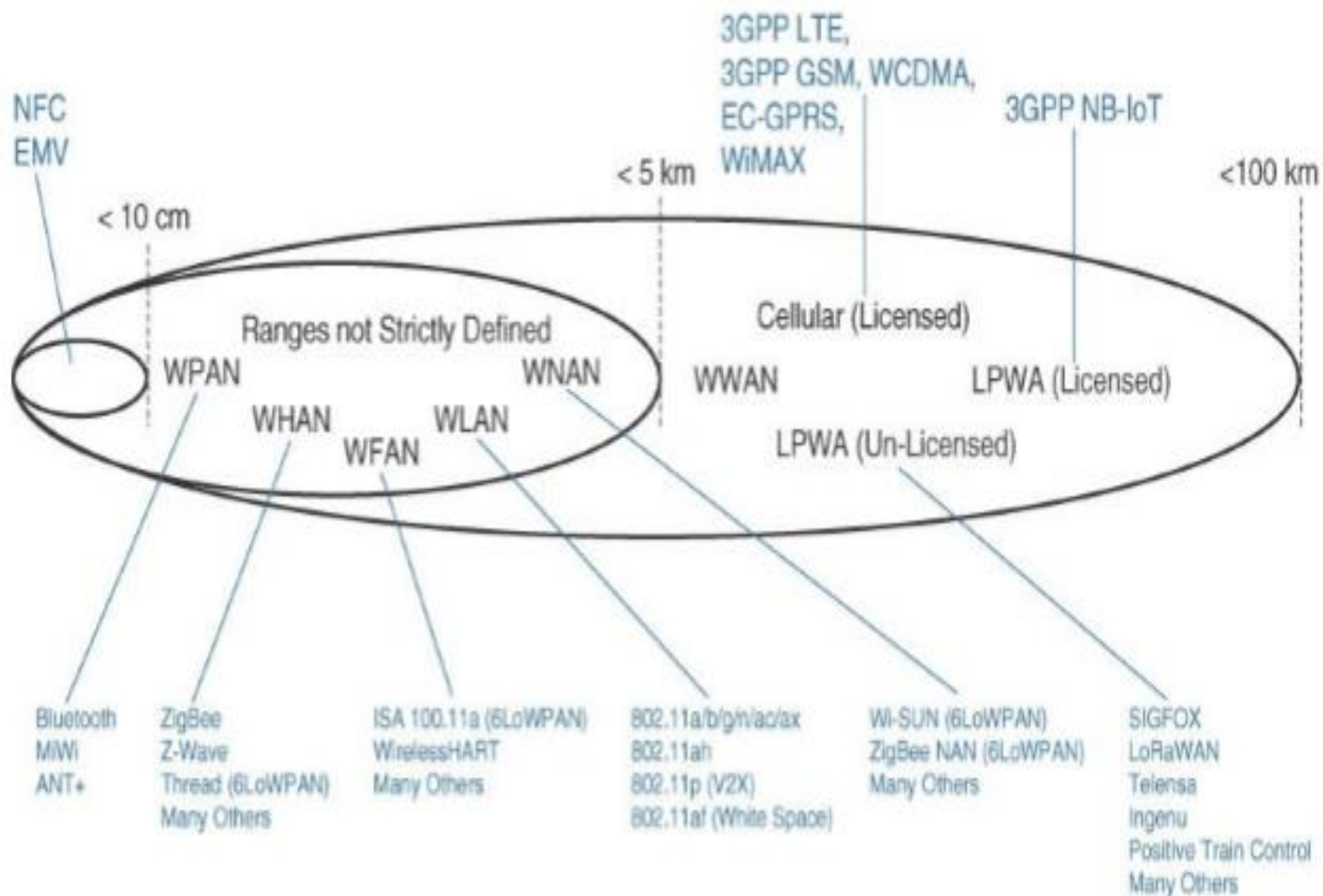
Device Mounting Differences

- In IT environments, devices are often rack-mounted. In OT environments, devices may be mounted in control cabinets on DIN rails, or horizontally on walls and fences. This reflects different industrial practices and safety requirements.

External Connectivity

- Industrial IoT systems often use alarm channels and indicators like stack lights to show process status. Devices may also receive external inputs to trigger actions. This external connectivity differentiates OT systems from IT assets.

IoT Wireless Access Technologies by Range



Purpose of the Diagram

- This diagram categorizes IoT access network technologies based on their transmission range and spectrum usage. IoT applications vary from very short-range (cm-level) connections to wide-area (tens of km). Each range has corresponding technologies suited for power, throughput, and mobility requirements.

Very Short Range (< 10 cm)

- Technologies: NFC, EMV.
- Use Cases: Payment systems, secure access control, identity verification.
- Characteristics: Operates at extremely close range for security, low latency, and minimal energy use.

Short Range (< 100 m) – WPAN/WFAN

- Technologies: Bluetooth, BLE, ANT+, ZigBee, Z-Wave, Thread, WirelessHART, ISA 100.11a.
- Use Cases: Wearables, home automation, industrial field sensors.
- Characteristics: Low power, low data rate, suitable for dense short-range IoT deployments.

Medium Range (< 1 km) – WLAN/WHAN

- Technologies: Wi-Fi (802.11a/b/g/n/ac/ax), Wi-Fi HaLow (802.11ah), 802.11p (V2X).
- Use Cases: Smart homes, offices, industrial automation, vehicle-to-vehicle communication.
- Characteristics: Higher throughput, reliable but more power-intensive compared to WPAN.

Wide Range (< 5 km) – WWAN

- Technologies: 3GPP LTE, GSM, WCDMA, EC-GPRS, WiMAX.
- Use Cases: Smart cities, fleet management, mobile IoT applications.
- Characteristics: Operates in licensed spectrum, offers guaranteed quality of service (QoS).

Very Wide Range (< 100 km) – LPWA

- Technologies: Licensed (NB-IoT, LTE-M), Unlicensed (LoRaWAN, Sigfox, Telensa, Ingenu).
- Use Cases: Agriculture, smart metering, environmental monitoring, utility grids.
- Characteristics: Very low power, long-range, optimized for small data packets, supports massive IoT deployments.

Key Observations

1. Range vs Throughput Trade-off: Shorter range technologies offer higher throughput, while longer range technologies prioritize coverage and energy efficiency.
2. Licensed vs Unlicensed Spectrum: Licensed (e.g., NB-IoT) ensures reliability, while unlicensed (e.g., LoRa) offers cost flexibility.
3. Overlapping Ranges: Ranges are not strictly defined, with overlaps among WPAN, WLAN, and WWAN technologies.

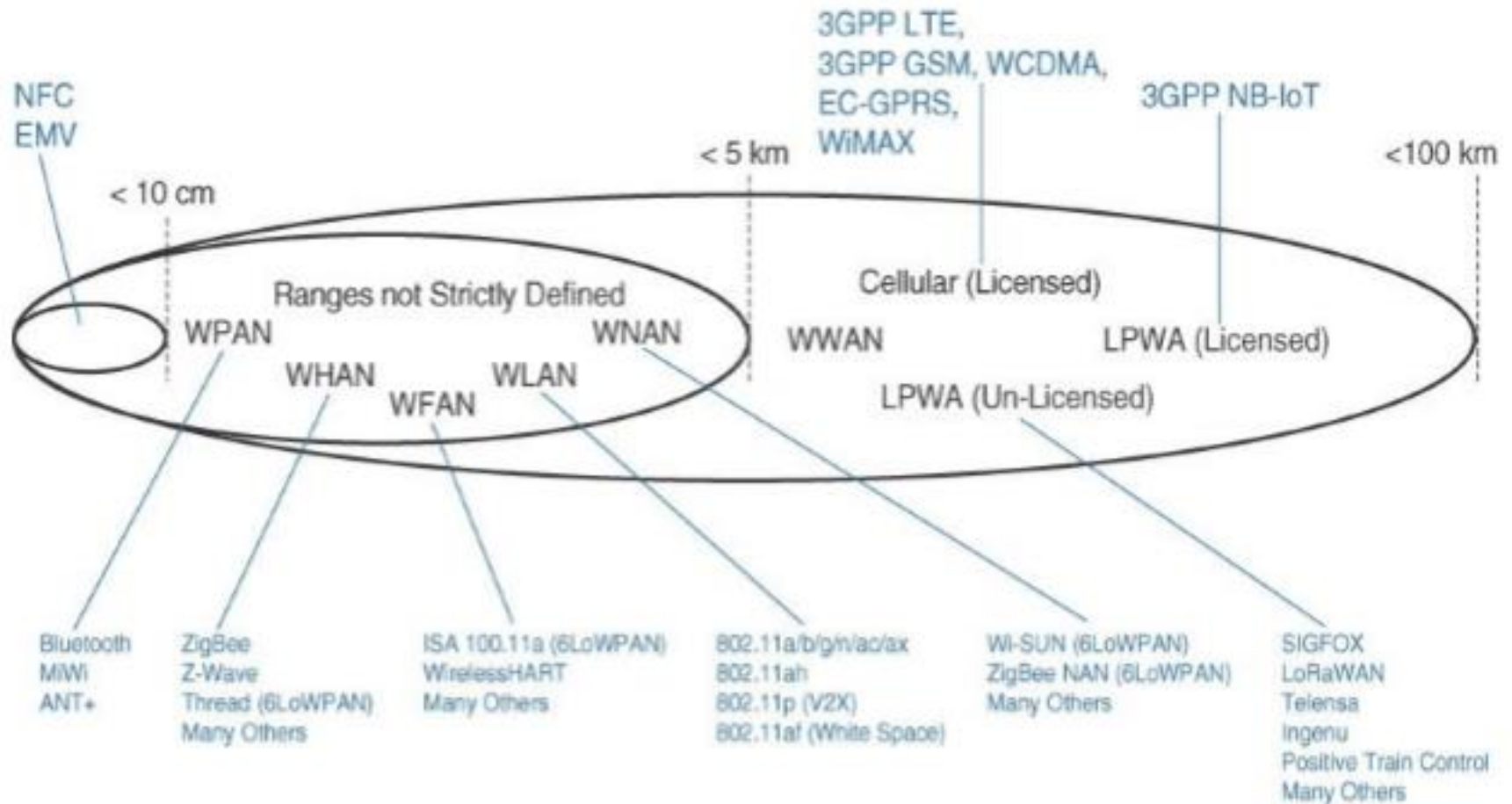
Conclusion

- The Access Network Sublayer provides the 'last mile' connectivity in IoT. Choosing the right wireless technology depends on application requirements such as range, power, mobility, throughput, and reliability.

Layer 2 – Communications Network Layer

Range Categories, Trade-offs, and
Topologies

Figure 2-9 – Inclusive Ranges of IoT Technologies



Explanation of Figure 2-9

- Figure 2-9 shows that IoT communication ranges are inclusive. Cellular is indicated for >5 km, but it also works at shorter ranges (e.g., 100 m). ZigBee is efficient only over tens of meters, not kilometers. Range estimates are grouped into categories like PAN, HAN, NAN, FAN, LAN, MAN, and WAN.

PAN and HAN

- PAN (Personal Area Network) – a few meters around a person, typically Bluetooth.
- HAN (Home Area Network) – a few tens of meters, common for ZigBee and BLE in smart homes.

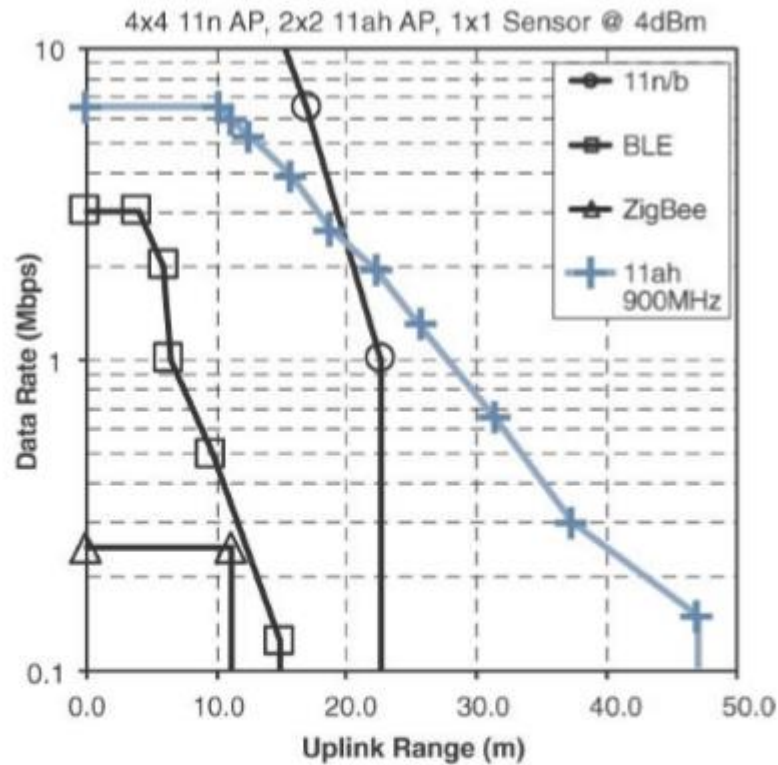
NAN and FAN

- NAN (Neighborhood Area Network) – hundreds of meters, often a group of house units.
- FAN (Field Area Network) – outdoor space of tens to hundreds of meters, sometimes groups of HANs or NANs.

LAN, MAN, WAN

- LAN (Local Area Network) – up to 100 m, e.g., Ethernet or Wi-Fi.
- MAN (Metropolitan Area Network) – a few kilometers.
- WAN (Wide Area Network) – more than a few kilometers. Wireless versions are denoted with a 'W' prefix, such as WLAN or WWAN.

Figure 2-10 – Throughput vs Range



Simulation Assumptions: 1% PER, 4dB NF, 32 Bytes, D-NLOS Fading, Indoor-to-Outdoor PL Model. 900MHz has 12dB propagation gain.

Sensor Antenna Gain: 11ah (-6.5dB) and 11n (-4dB). AP antenna gain = 2dB.

* BT Long Range Adds 125 kbps and 500 kbps Modes

Figure 2-10 Range Versus Throughput for Four WHAN to WLAN Technologies

Explanation of Figure 2-10

- Figure 2-10 compares throughput degradation across technologies at WHAN to WLAN ranges. Even with same frame size, transmit power, and antenna gain, throughput declines at different rates with distance. This limits data capacity as range increases.

Trade-offs

- Increasing throughput and range generally requires higher power consumption.
Designers must balance mobility, range, and throughput with power constraints to select appropriate IoT technology.

Figure 2-11 – Cost, Range, Power, and Bandwidth Trade-offs

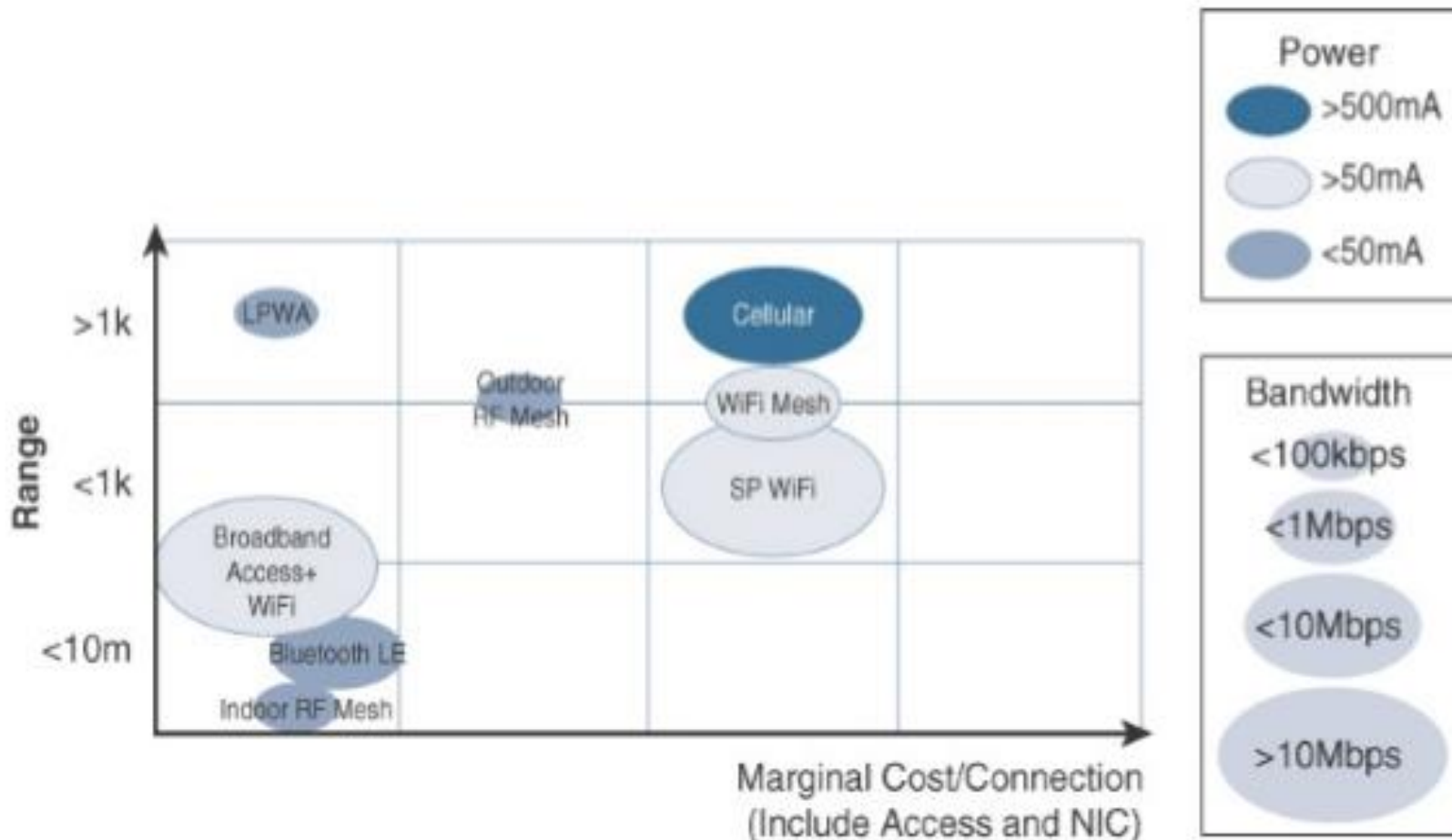


Figure 2-11 Comparison Between Common Last-Mile Technologies in Terms of Range Versus Cost, Power, and Bandwidth

Explanation of Figure 2-11

- Figure 2-11 integrates cost, range, power consumption, and available bandwidth. It shows that choosing IoT technology is not just about range or throughput, but also cost efficiency and energy requirements.

Figure 2-12 – Star and Cluster Tree Topologies

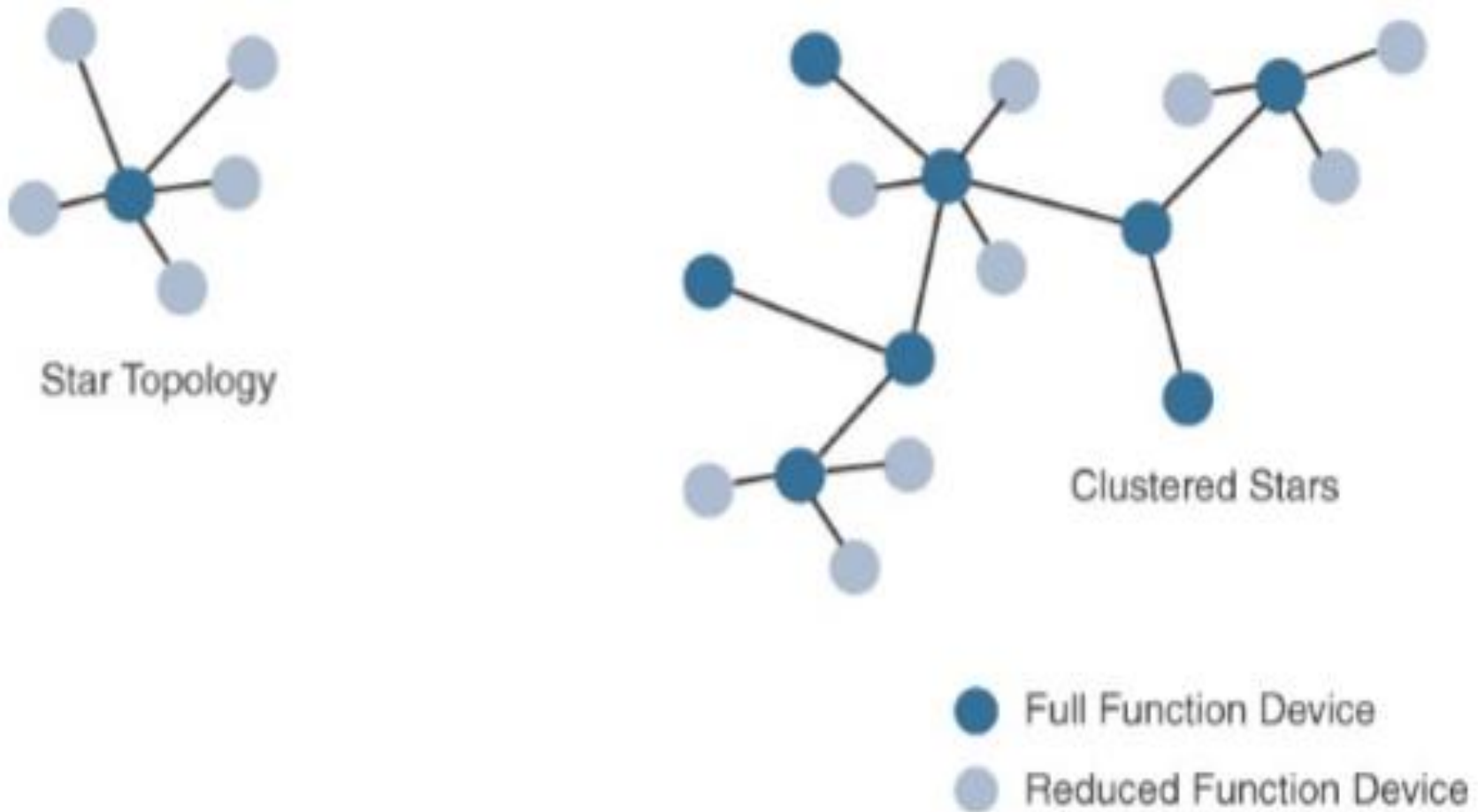


Figure 2-12 *Star and Clustered Star Topologies*

Explanation of Figure 2-12

- Figure 2-12 illustrates point-to-multipoint structures like star and cluster tree topologies. In a star, a central coordinator manages all sensors. In cluster tree, multiple full-function devices (FFDs) interconnect and may have reduced-function devices (RFDs) as leaves.

IEEE 802.15.4 Example

- In IEEE 802.15.4, RFDs are small, battery-powered end devices that only communicate with a coordinator. FFDs can connect with other FFDs and RFDs, forming cluster-tree topologies.

Figure 2-13 – Mesh Topology

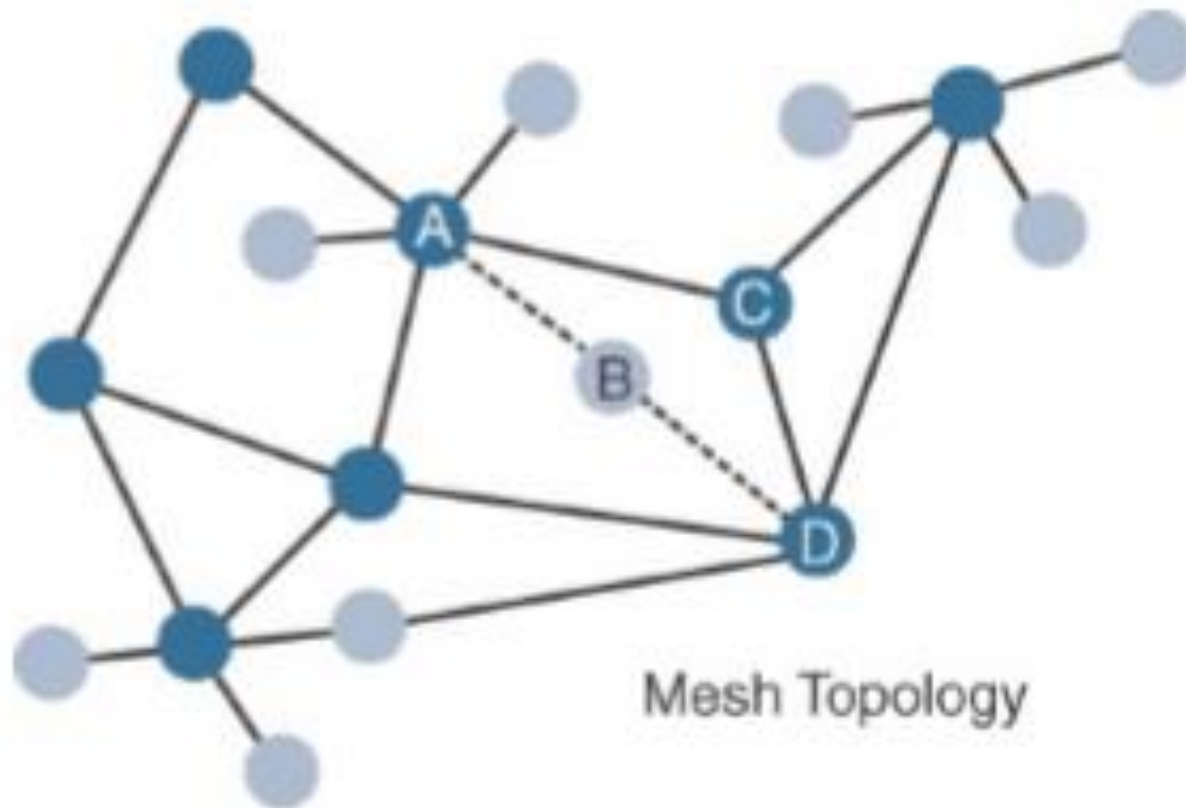


Figure 2-13 *Mesh Topology*

Explanation of Figure 2-13

- Figure 2-13 shows a mesh topology, where nodes may have multiple paths to one another. This redundancy ensures that if one relay fails, communication reroutes through another node. Example: Node A communicates with Node D via Node B or Node C.

Mesh Properties

- Mesh networks extend range by relaying data and provide resilience against node failure. They are slower due to relaying overhead but add fault tolerance and scalability.

Conclusion

- Figures 2-9 to 2-13 collectively illustrate IoT range categories, throughput-power trade-offs, and network topologies. Key insights:
 - - Range categories (PAN, HAN, NAN, FAN, LAN, MAN, WAN).
 - - Trade-offs between range, throughput, cost, and power.
 - - Topologies: Star, Cluster Tree, and Mesh for scalability and resilience.

Gateways and Backhaul Sublayer

Detailed Explanation (Based on
Textbook Section)

Introduction

- Data collected from IoT smart objects needs to be transported to a central station where it is processed. The gateway performs the role of forwarding data received through access technologies into the backhaul medium, which delivers it to the central station.

Gateway Role

- Smart objects are often static or mobile within a limited area, while gateways are typically static. The gateway manages inter-medium communication, allowing IoT data to move beyond the access layer. Communication between sensors and gateways may be wired or wireless.

Example – DSRC

- Dedicated Short-Range Communication (DSRC) supports vehicle-to-vehicle and vehicle-to-infrastructure communications. In this case, cars include both sensors and a gateway. DSRC uses the upper 5 GHz band for backhaul and can also operate peer-to-peer or in mesh between vehicles.

Choice of Backhaul Technology

- The choice of backhaul technology depends on communication distance and the amount of data to forward. For shorter and more stable environments, wired solutions may suffice. For dynamic or large areas, wireless solutions are necessary.

Stable Environments

- In controlled settings such as factories or oil and gas fields, Ethernet is a preferred backhaul technology. It provides stability and high throughput when cabling is feasible.

Unstable Environments

- In environments like open mines where cabling is impractical or unsafe, wireless backhaul is chosen. Wi-Fi is commonly used, often configured in mesh topologies to extend coverage.

Throughput and Hops

- In mesh networks, throughput decreases as hops increase. Typically, in a Wi-Fi mesh, throughput halves with each additional hop, limiting efficiency as the network expands.

802.11ah – Wi-Fi for IoT

- 802.11ah is a Wi-Fi variant designed for IoT, operating below 1 GHz instead of 2.4/5 GHz. It offers ranges of up to 2 km with better propagation characteristics, making it suitable for extended IoT backhaul.

WiMAX (802.16)

- WiMAX provides longer ranges, up to 50 km, and throughput of 70 Mbps at short range and 2–3 Mbps at maximum range. The 802.16d (Fixed WiMAX) version is widely implemented for backhaul. Although it can operate in unlicensed bands, licensed spectrum is usually preferred for reliability.

Cellular Backhaul

- Cellular technologies compete with WiMAX for backhaul solutions, offering similar range and throughput. The decision between cellular and WiMAX depends on the industry vertical, local regulations, and cost considerations.

Table 2-4 – Comparison of Backhaul Solutions

Technology	Type and Range	Architectural Characteristics
Ethernet	Wired, 100 m max	Requires a cable per sensor/sensor group; adapted to static sensor position in a stable environment; range is limited; link is very reliable
Wi-Fi (2.4 GHz, 5 GHz)	Wireless, 100 m (multipoint) to a few kilometers (P2P)	Can connect multiple clients (typically fewer than 200) to a single AP; range is limited; adapted to cases where client power is not an issue (continuous power or client battery recharged easily); large bandwidth available, but interference from other systems likely; AP needs a cable
802.11ah (HaloW, Wi-Fi in sub-1 GHz)	Wireless, 1.5 km (multipoint), 10 km (P2P)	Can connect a large number of clients (up to 6000 per AP); longer range than traditional Wi-Fi; power efficient; limited bandwidth; low adoption; and cost may be an issue
WiMAX (802.16)	Wireless, several kilometers (last mile), up to 50 km (backhaul)	Can connect a large number of clients; large bandwidth available in licensed spectrum (fee-based); reduced bandwidth in license-free spectrum (interferences from other systems likely); adoption varies on location
Cellular (for example, LTE)	Wireless, several kilometers	Can connect a large number of clients; large bandwidth available; licensed spectrum (interference-free; license-based)

Table 2-4 Architectural Considerations for WiMAX and Cellular Technologies

Conclusion

- The Gateways and Backhaul Sublayer ensures IoT systems can transport sensor data reliably to processing centers. Different technologies—Ethernet, Wi-Fi, 802.11ah, WiMAX, and cellular—serve different use cases based on distance, environment, throughput needs, and cost factors.

Network Transport Sublayer

Introduction

- The Network Transport Sublayer builds on the gateway and backhaul model, moving beyond strict hierarchies. Instead, it enables flexible communication with multiple paths, adapting to IoT use cases such as smart grids.

Example – Energy Grid

- In smart energy grids, home meters report energy consumption via gateways to central stations. These reports are low-throughput and infrequent, typically using 802.11ah, 802.15.4, or LPWA technologies. Neighboring areas form FANs, providing broader collection coverage.

Beyond Hierarchical Models

- While the structure appears hierarchical, in practice, IoT systems support more dynamic flows. For example, a headend may request on-demand updates (downstream pull) when consumption spikes, changing the model from simple push to bidirectional communication.

Distribution Automation (DA)

- DA enables smart meters to communicate with neighboring meters for load balancing. By coordinating air conditioning or appliance cycles between houses, energy consumption is stabilized, avoiding network spikes and inefficiencies.

Household Mesh

- Smart meters may communicate directly with appliances. For example, washing machines may run at night during low consumption, while EV charging schedules adapt dynamically. This often creates a partial mesh, with some nodes linking to multiple others.

Public Charging Stations

- Smart car charging systems link to energy accounts via public charging stations. The headend system can query multiple stations for status updates, reflecting a non-vertical communication model.

Top-Down Flows

- IoT communication is not limited to bottom-up reporting. Headend systems may push updates downstream, such as software upgrades for meters and appliances. This creates top-down flows over the same network infrastructure.

Transport Models

- IoT transport structures include:
 - - Peer-to-peer (meter-to-meter)
 - - Point-to-point (meter-to-headend)
 - - Point-to-multipoint (headend to many meters)
 - - Unicast and multicast (e.g., software updates to one or many devices).

Multitenant Environments

- In some deployments, multiple utilities (electricity, gas, water) may share the same IoT communication pathways. This requires careful orchestration and resource allocation to avoid conflicts.

Multiple Media

- IoT communication traverses multiple media: power lines within homes, short-range wireless (Wi-Fi, ZigBee), longer-range wireless to gateways, and wired or wireless backhaul to central stations.

Need for Protocols

- To support such flexible communication, protocols must be open, standard-based, secure, and scalable. They must accommodate thousands to millions of devices and diverse media.

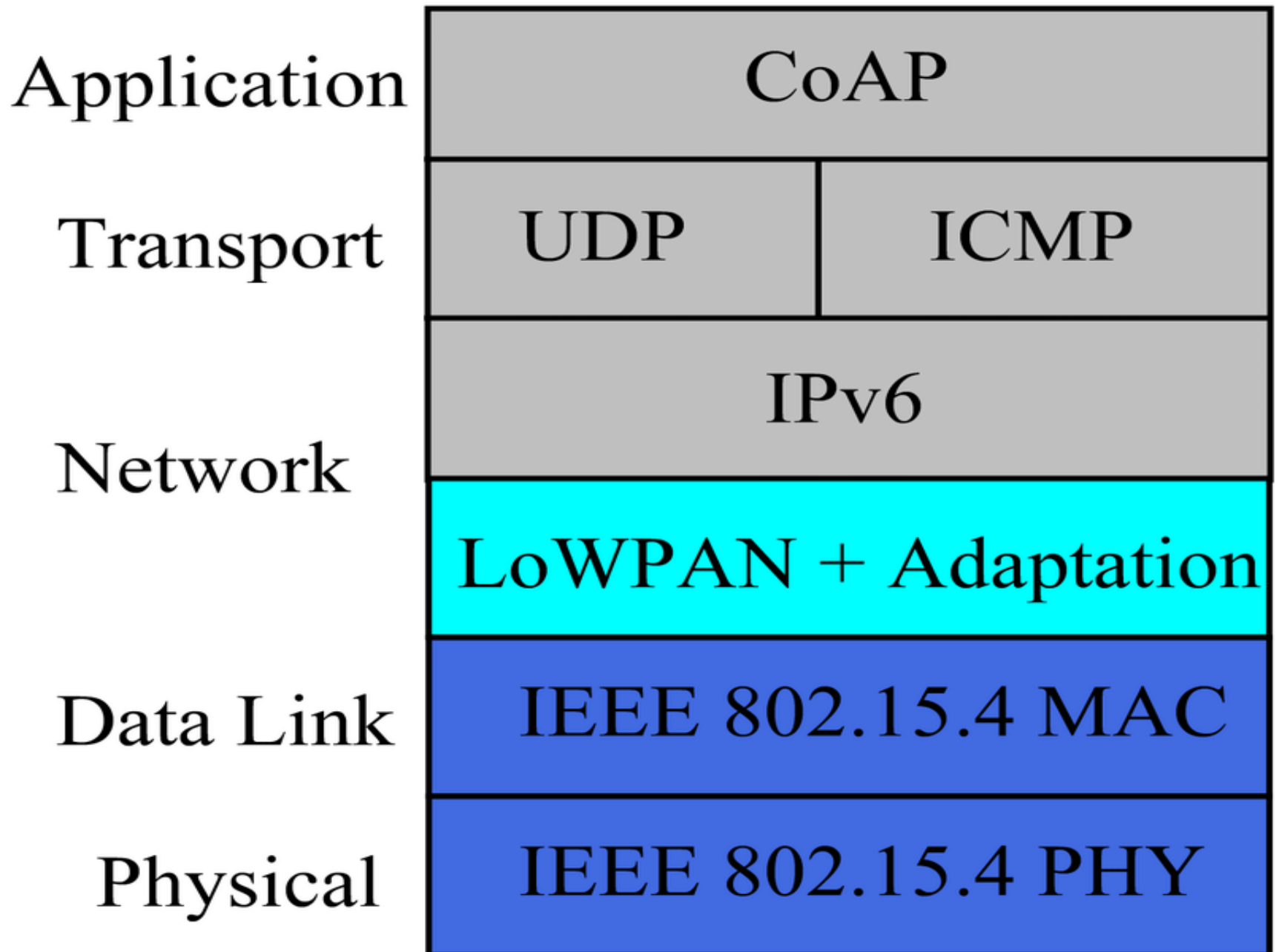
IP as Transport Foundation

- IP satisfies IoT requirements of scalability, openness, and cross-media compatibility. Its flexibility allows embedding in diverse devices, even over low-power, lossy, and low-bandwidth networks.

IPv6 Examples

- RFC 2464 defines IPv6 over Ethernet. IPv6 also works over IEEE 802.11 Wi-Fi. The IETF 6LoWPAN standard adapts IPv6 packets for efficient use over lossy, constrained IoT networks.

Figure – 6LoWPAN Adaptation for IPv6 in IoT



Transport Layer Protocols

- Built on top of IP, transport protocols like TCP and UDP manage packet delivery. TCP ensures reliable delivery but is heavy, while UDP is lighter, faster, and more suitable for constrained IoT systems.

TCP

- TCP guarantees ordered and reliable data delivery. It is suited for critical applications but consumes more resources, making it less ideal for low-power IoT devices.

UDP

- UDP is a lightweight alternative, offering faster transmission with lower overhead. However, it does not guarantee delivery, leaving reliability to the application layer.

Security Extensions

- TCP connections can be secured with TLS/SSL, while UDP can be secured with DTLS. These extensions ensure secure communication even in lightweight IoT networks.

Conclusion

- The Network Transport Sublayer provides flexibility for IoT communication models: hierarchical, peer-to-peer, mesh, and multitenant. By using IP as a foundation and leveraging TCP/UDP with security, it enables scalable and interoperable IoT systems.

IoT Network Management Sublayer

Introduction

- The IoT Network Management Sublayer builds on IP, TCP, and UDP to manage data exchange. It provides mechanisms for push (regular sensor reports), pull (application queries), and hybrid models. Protocols at this layer handle reliability, efficiency, and resource constraints in IoT systems.

HTTP in IoT

- HTTP follows a client-server model where sensors can act as clients reporting to IoT applications. Although widely used, HTTP is a heavy protocol not optimized for IoT's constrained environments with low memory, limited power, and high packet loss.

WebSocket

- WebSocket, part of HTML5, enables simple bidirectional communication over a single connection. It is more lightweight than HTTP and is sometimes combined with MQTT to support IoT-specific needs.

XMPP (Extensible Messaging and Presence Protocol)

- XMPP was derived from instant messaging and supports data exchange, presence, and contact lists. It also enables publish/subscribe models, making it suitable for IoT information distribution. However, XMPP relies on TCP, requiring persistent sessions, which is a limitation for constrained devices.

CoAP (Constrained Application Protocol)

- CoAP was created by the IETF CoRE working group to address IoT needs. It uses HTTP-like methods (GET, POST, PUT, DELETE) but with smaller headers, running on UDP instead of TCP. CoAP adds 'observation', allowing streaming of state changes without constant queries.

MQTT (Message Queue Telemetry Transport)

- MQTT follows a publisher-subscriber-broker architecture. Sensors publish information, applications subscribe to it, and brokers relay between them. MQTT is lightweight but relies on TCP, requiring clients to keep connections open, which can strain limited devices.

TCP vs UDP Considerations

- TCP ensures reliability and session awareness but increases memory and processing overhead. UDP is lightweight and faster, leaving reliability management to the application. Protocol choice depends on application requirements.

Quality of Service (QoS)

- Different IoT messages may require different priority levels. Application protocols must support QoS differentiation to ensure critical data is delivered efficiently.

Security Requirements

- IoT protocols must balance security against overhead. Security measures like TLS/SSL for TCP or DTLS for UDP protect communication but may consume more resources. Selecting the right level of security is crucial for constrained IoT environments.

Conclusion

- The IoT Network Management Sublayer integrates protocols such as HTTP, WebSocket, XMPP, CoAP, and MQTT. Architectural choices depend on constraints, QoS, and security requirements. Proper selection ensures efficient, reliable, and secure IoT data communication.

Layer 3: Applications and Analytics Layer

Introduction

- Once IoT devices are connected, they exchange data with other systems. The real value of IoT lies in the applications that process and utilize this data, spanning from analytics to control functionalities.

Analytics vs Control Applications

- Applications in IoT can be broadly classified into two categories:
 - - Analytics applications: Focus on data collection and insights.
 - - Control applications: Directly influence device/system behavior.
- Many advanced IoT applications integrate both functions.

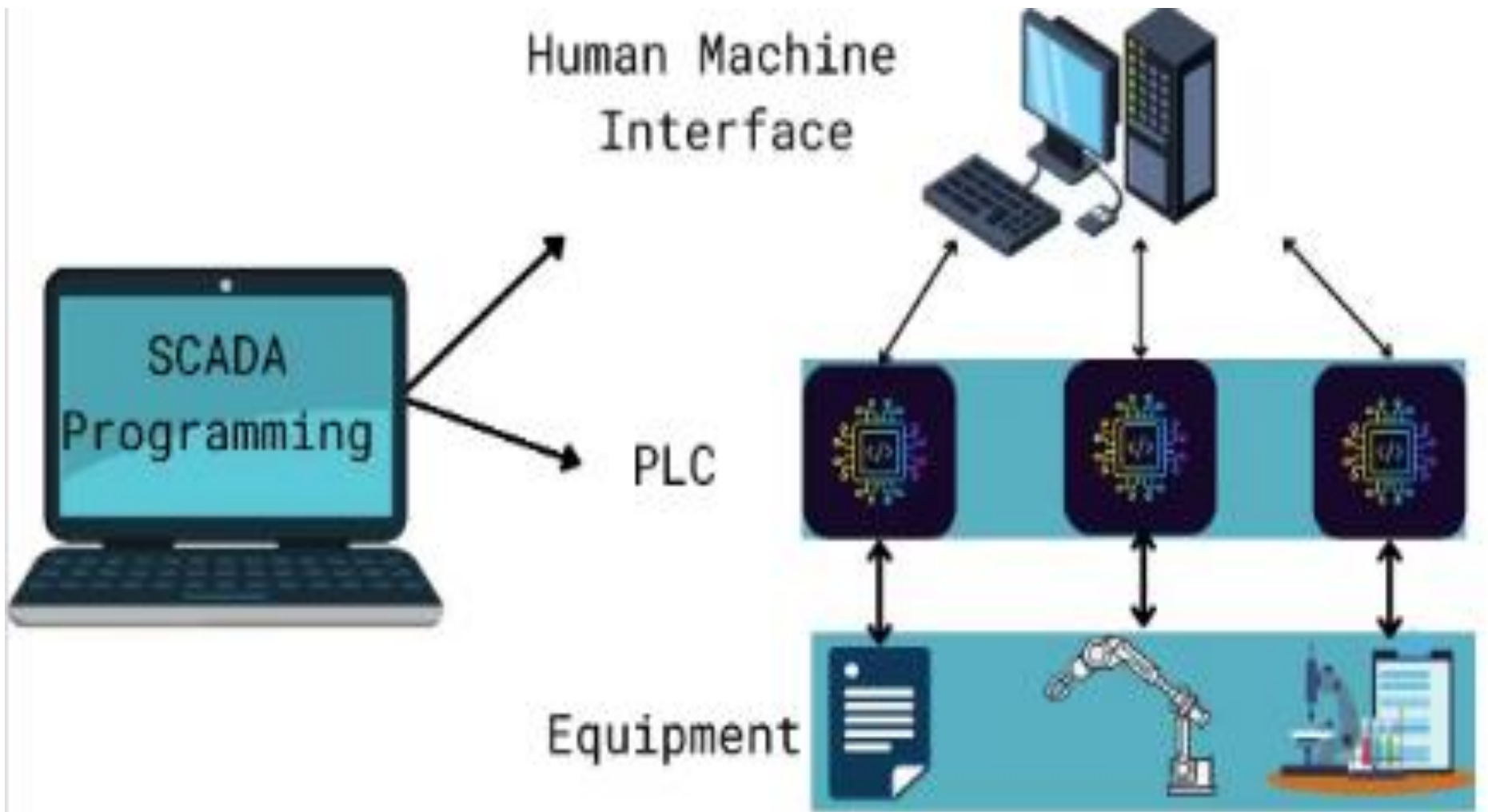
Analytics Applications

- Analytics applications collect data from multiple smart objects, process it, and display aggregated insights such as historical reports, statistics, trends, or system states. They reveal network-wide perspectives that single-device data cannot provide.

Control Applications

- Control applications guide the behavior of IoT objects or related systems. For example, a pressure sensor may control a pump, increasing its speed when pressure drops. These applications manage complex logic that cannot be embedded in a single IoT object.

Figure – SCADA Example Architecture



SCADA Example

- Supervisory Control and Data Acquisition (SCADA) systems represent a control application model. SCADA enables universal access to remote systems for monitoring and control. It is widely used in electrical distribution grids with Remote Terminal Units (RTUs).

Analytics + Control Integration

- Most modern IoT applications combine analytics and control. The analytics module processes sensor data, and the control module sends instructions to adjust device or system behavior based on analytics results.

Evaluation Considerations

- When evaluating IoT applications, consider:
 - - Depth of analytics (simple trends vs advanced AI-driven insights).
 - - Extent of control (basic commands vs orchestrated multi-device actions).
 - - Balancing analytics depth with control sophistication for the specific use case.

Conclusion

- Layer 3: Applications and Analytics Layer delivers actionable intelligence from IoT data. Analytics ensures visibility, reporting, and insights. Control ensures system responsiveness and optimization. Together, they maximize efficiency and overall IoT value.

Data Versus Network Analytics

- Overview
 - Analytics processes information to make sense of collected IoT data. In IoT, analytics splits into two broad classes: data analytics and network analytics; most real systems use both.
- Data Analytics
 - Processes measurements from many smart objects to produce an intelligent, system-wide view (from simple shelf-empty alarms to complex multi-sensor weather-path predictions). Also monitors asset health for predictive maintenance (e.g., robot movement degradation).
- Network Analytics
 - Assesses connectivity health and performance. Connectivity loss degrades efficiency or safety (e.g., autonomous haul trucks stopping in mines). Without network connectivity, data analytics and control loops stall. Depth of analysis depends on use case (basic status vs near-real-time control).

Data Analytics Versus Business Benefits

- Static vs Open Architectures
 - Static deployments predefine elements and analytics (typical in industrial monitoring). Smarter choice: open systems engineered for future sensors and bidirectional flows; enables new processing and deeper interaction for emergent business value.
- Example — Cisco Jasper (Vending Machines)
 - Start with error-state alerts to dispatch repair only when needed. Evolve by adding sales telemetry per item/location/time; optimize assortments by time/season/location; open platforms permit new applications over time.

Smart Services

- Definition & Goal
 - Generic term for IoT-enabled capabilities that aim for operational efficiency and added intelligence across domains.
- Compliance & Safety
 - Sensors verify regulatory/safety conformance: presence in hazardous areas, truck weight thresholds, etc.
- Operational Optimization
 - Measure machine output/speed/usage; optimize factory flows; in hospitality/retail, presence/motion guide staffing or assistance when a customer lingers.

Smart Services

- Home to Grid Coordination
 - Bulb presence sensors trigger lights; smarter systems learn patterns, anticipate presence, adjust color to mood/preferences; coordinate with alarm/HVAC. At grid scale, coordinate AC pulses and appliance cycles across homes to smooth demand.
- M2M & Smart City Logistics
 - Mining vehicles coordinate flows between drills, draglines, dozers, and trucks. In cities, vehicles and transit exchange data to anticipate congestion and dynamically reroute or scale service.
- Data Scale Challenge
 - IoT devices grow exponentially, and their data volumes can overwhelm headend/cloud systems. Most raw sensor traffic is unstructured (e.g., meter polling) yet becomes valuable when correlated with context (weather, demand) to infer outages and scope.

IoT Data Management & Compute Stack

- Cloud-Centric Model
 - Natural model centralizes processing in the cloud—simple connectivity and full visibility. But limitations arise as data volumes, device diversity, and efficiency needs increase.
- Why Push Processing Toward the Edge
 - Reduce latency for industrial control, conserve bandwidth (e.g., jets: ~10 TB per 30 minutes), and improve local efficiency where conditions and responses are site-specific.
- Impedance Mismatch
 - A 1M-meter network can generate ~1B data points per day (~1 TB). Last-mile bandwidth is limited, latency high, backhaul costly/unreliable, and much data is low-value if shipped raw; big data in cloud alone becomes impractical.

Figure 2-14 — The Traditional IT Cloud Computing Model

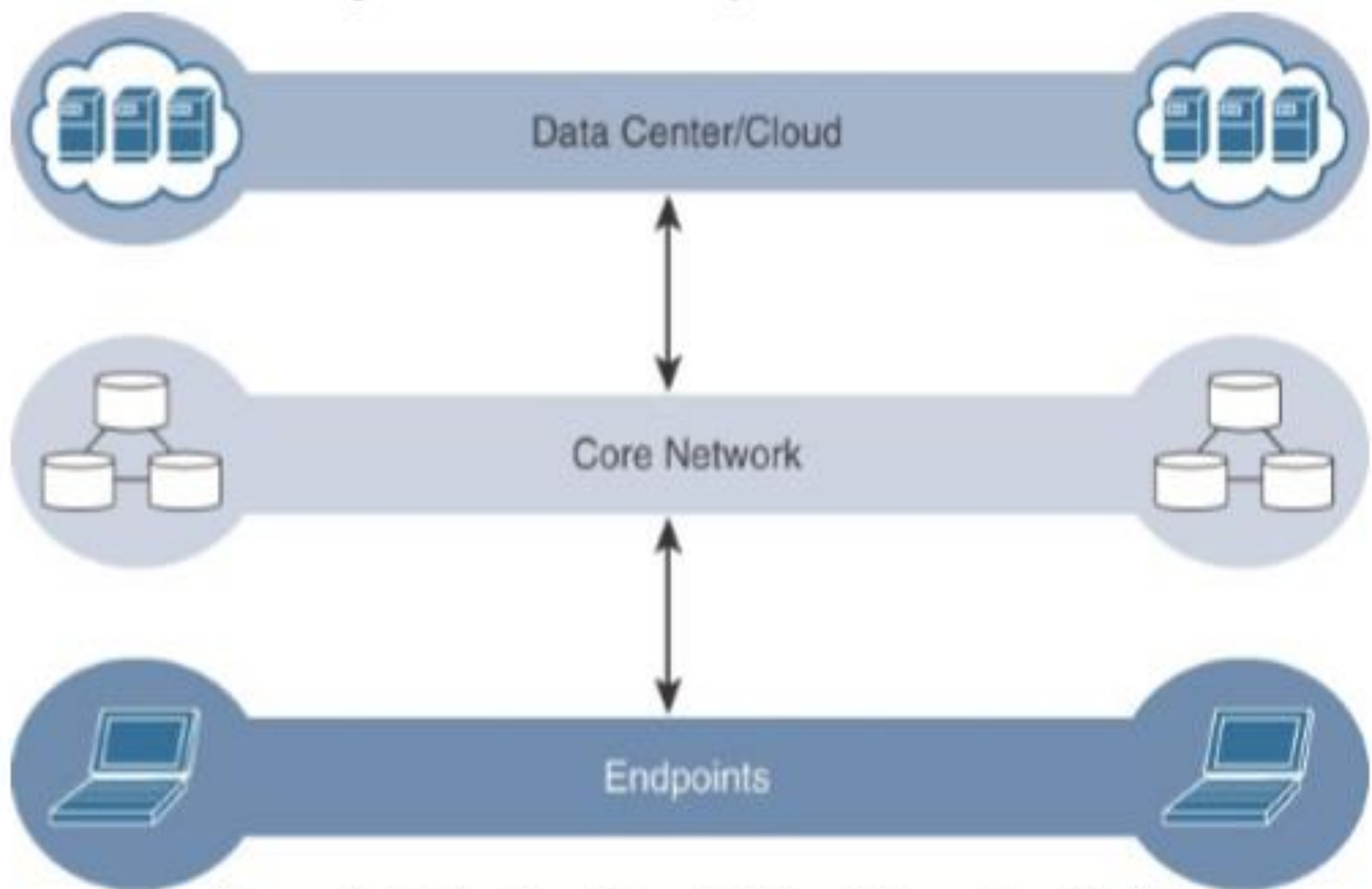


Figure 2-14 *The Traditional IT Cloud Computing Model*

Fog Computing

- Concept
 - Distribute data management close to the edge; any device with compute, storage, and networking can be a fog node (controllers, routers, gateways). Analyze locally to minimize latency, reduce core traffic, and keep sensitive data local.
- Origin of the Term (Note)
 - Fog sits near the ground—close to devices. The term was coined within Cisco; compares edge placement to clouds aloft.

Fog Computing

- Benefits & Placement
 - Fog nodes near sensors have contextual awareness (e.g., oil derrick router correlating local sensors). They forward only relevant summaries upstream, enabling distributed analytics and faster local control loops.
- Example — Mining Truck Tires
 - Local fog on the truck fuses tire pressure with engine/hydraulic data; sends alerts only when issues emerge, saving backhaul bandwidth and focusing attention on actionable events.

Figure 2-15 — IoT Data Management & Compute Stack with Fog Computing

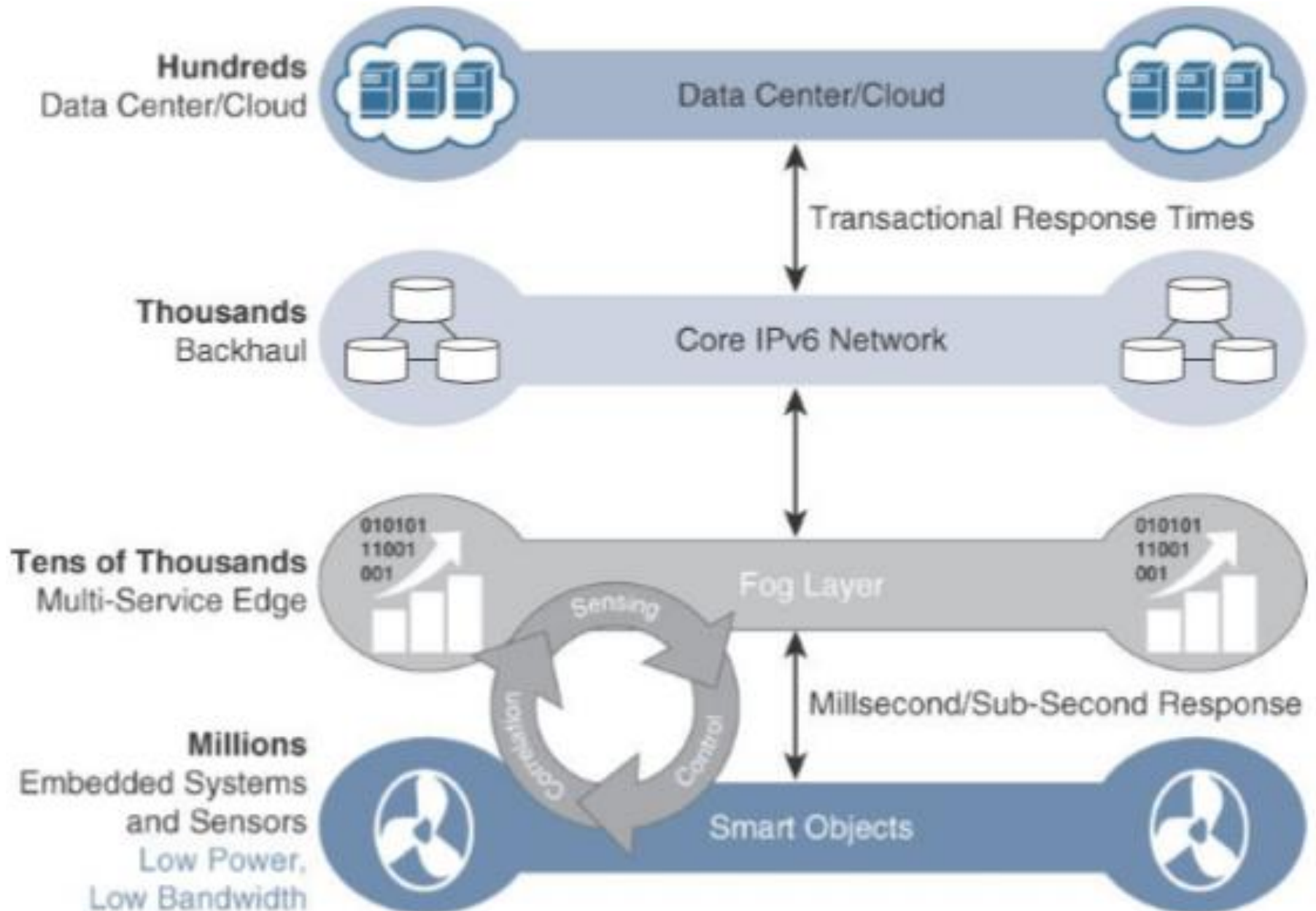


Figure 2-15 The IoT Data Management and Compute Stack with Fog Computing

Edge Computing (Mist)

- Pushing Compute into Endpoints
 - Some modern sensors/IoT devices have enough compute to perform low-level analytics/filters for rapid local decisions (e.g., fire hydrant water-pressure node raising immediate local alerts).
- Edge–Fog Synergy
 - Edge detects localized anomalies quickly; fog aggregates a wider view to determine if issues are systemic. Smart meters can peer-share local power quality and notify fog of micro-area events.

Hierarchy of Edge, Fog, and Cloud

- Complementary Layers
 - Edge/fog do not replace cloud; they filter and react fast near devices while cloud provides heavy analytics, historical storage, and global coordination. Heterogeneous edge/fog nodes require an abstraction layer and containerization/virtualization for portability and multitenancy (e.g., oneM2M APIs).
- Right-Place Processing
 - Most time-sensitive data: analyze at edge/fog; seconds-to-minutes latency: analyze at aggregation nodes; less time-sensitive: send to cloud for big data and long-term storage (periodic summaries from many fog nodes).

Hierarchy of Edge, Fog, and Cloud

- Design Guidance
 - Match data volume and time sensitivity to the compute layer. Fog reduces round trips, offloads bandwidth, and protects sensitive data by processing inside organizational boundaries.

Figure 2-16 — Distributed Compute & Data Management Across an IoT System

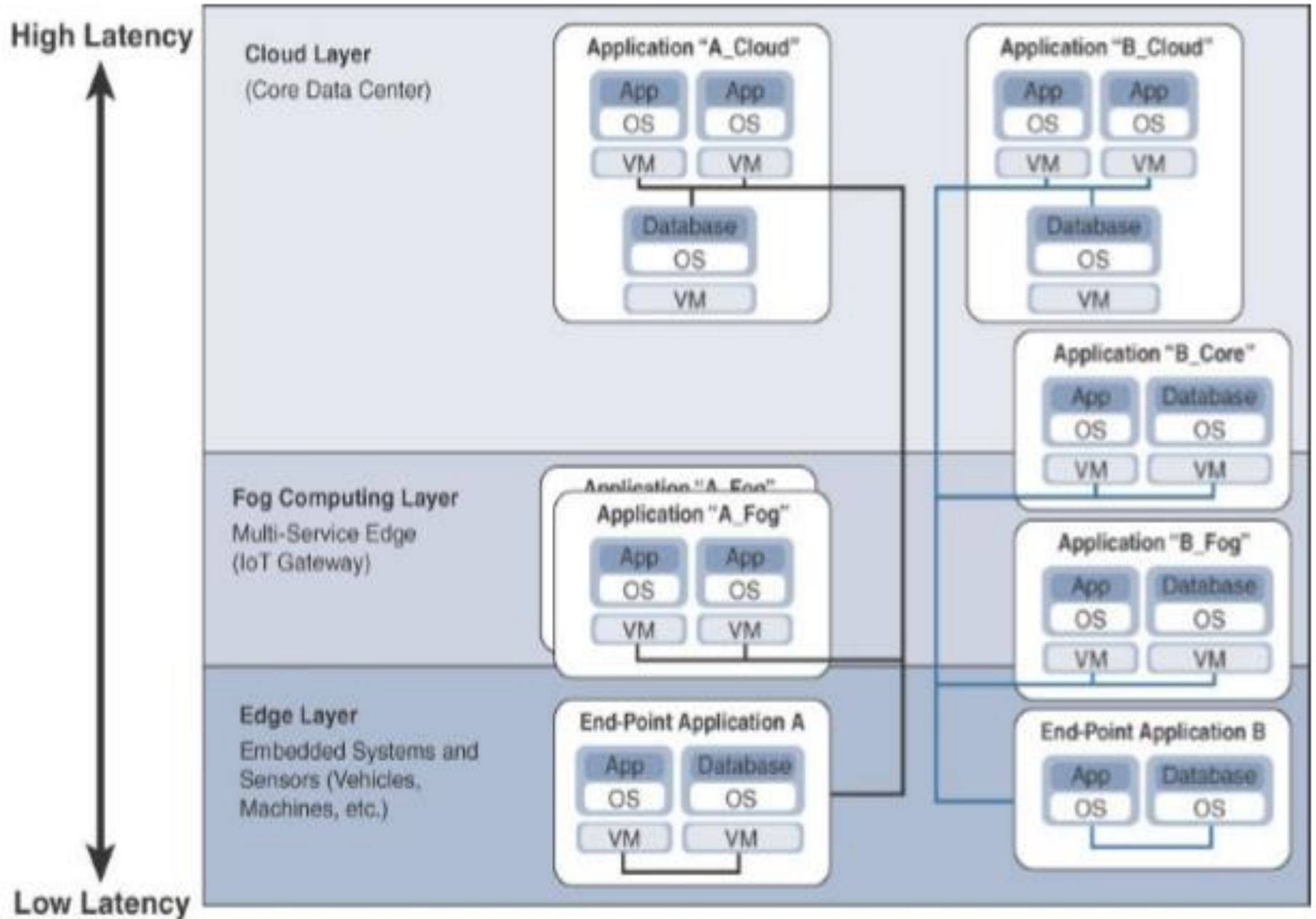


Figure 2-16 Distributed Compute and Data Management Across an IoT System

Summary

- Two Parallel Stacks
 - Core IoT Functional Stack: things, networking components (with sublayers), applications & analytics.
 - IoT Data Management & Compute Stack: where data is filtered, aggregated, stored, and analyzed (cloud + fog + edge).
- Architectural Takeaways
 - Design open systems to accommodate future sensors and bidirectional flows; distribute analytics per time-sensitivity; plan for bandwidth/latency limits and heterogeneous compute at edge/fog with standardized APIs and containers.