MANIPAL INSTITUTE OF TECHNOLOGY
BENGALURU
(A constituent unit of MAHE, Manipal)

INTERNET OF THINGS

MODULE 2: M2M to IoT – An Architectural Overview

An IoT architecture outline, Standards considerations. IoT data Management, IoT architecture-State of art solution, IoT reference model, IoT deployment and operational view. [7 Hours]

An IoT architecture outline

- Attempting to produce a single architecture consequently results in a number of optional and conditional requirements, all depending on the particular problem at hand or application in focus.
- Nevertheless, the identified key features that are needed when building an M2M or IoT solution can now be put together into a larger context by proposing a single view of the main functional capabilities (see Figure 4.3).
- This is not a strict and formal functional architecture but provides a conceptual overview.
- It also follows the approach of looking at the system capabilities from a layered point of view, including highlighting key functions that go across the layers.

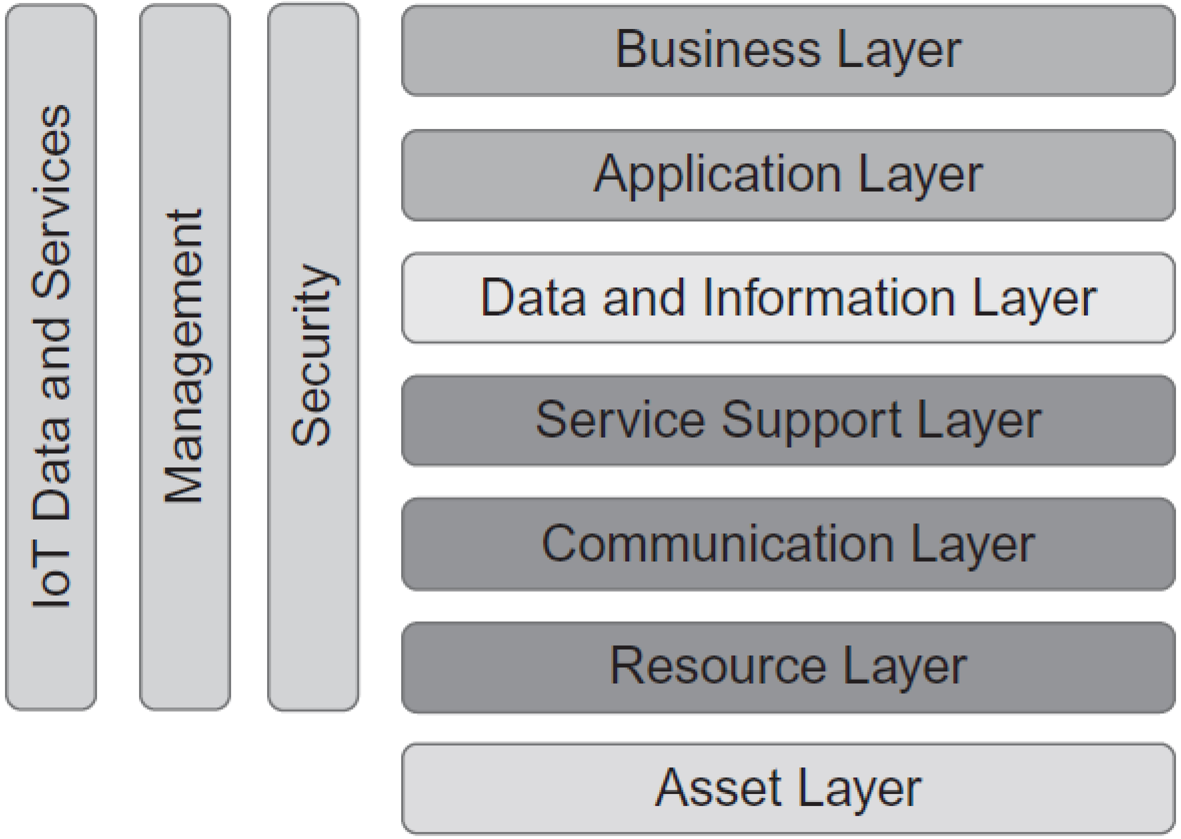


FIGURE 4.3

Functional layers and capabilities of an IoT solution.

1. **Asset Layer:** This is the foundational layer, consisting of physical devices, sensors, and actuators that interact with the environment. These assets collect data and perform actions based on instructions received from higher layers.
2. **Resource Layer:** This layer manages the resources provided by the assets. It includes data collection, preliminary processing, and resource allocation. It ensures that the collected data is properly stored and accessible for further processing.
3. **Communication Layer**: Responsible for transmitting data between devices and other layers. This layer encompasses the networking protocols and technologies required for communication, such as Wi-Fi, Bluetooth, Zigbee, and cellular networks.
4. **Service Support Layer**: Provides the necessary services and support for the IoT system to function effectively. This includes middleware services that facilitate communication, data management, and integration of different components within the system.

5. **Data and Information Layer**: Focuses on the processing, analysis, and storage of data collected from the lower layers. It transforms raw data into meaningful information through data analytics, machine learning, and other data processing techniques.

6. **Application Layer:** This layer hosts the applications and user interfaces that interact with the IoT system. It includes software that allows users to monitor, control, and manage IoT devices and services. Applications are tailored to specific use cases, such as smart homes, healthcare, and industrial automation.

7. **Business Layer**: The topmost layer, which deals with business processes and objectives. It ensures that the IoT solution aligns with business goals and adds value. This layer includes business analytics, decision support systems, and other tools that help in making strategic decisions based on data insights.

Horizontal Capabilities:

- **IoT Data and Services**:
  - This encompasses the overall management of data and services across all layers. It ensures that data is collected, processed, and utilized efficiently and securely.
- **Management**:
  - Refers to the administration and orchestration of the entire IoT solution. This includes device management, network management, and the coordination of various IoT components to ensure smooth operation.
- **Security**:
  - A critical capability that spans all layers, ensuring the confidentiality, integrity, and availability of data and services. It involves implementing security measures such as encryption, authentication, and access control to protect the IoT system from threats and vulnerabilities.

Standards considerations

- The primary objective of any technology oriented standardization activity is to provide a set of agreed-upon specifications that typically address issues like achieving interoperability in a market with many actors and suppliers.

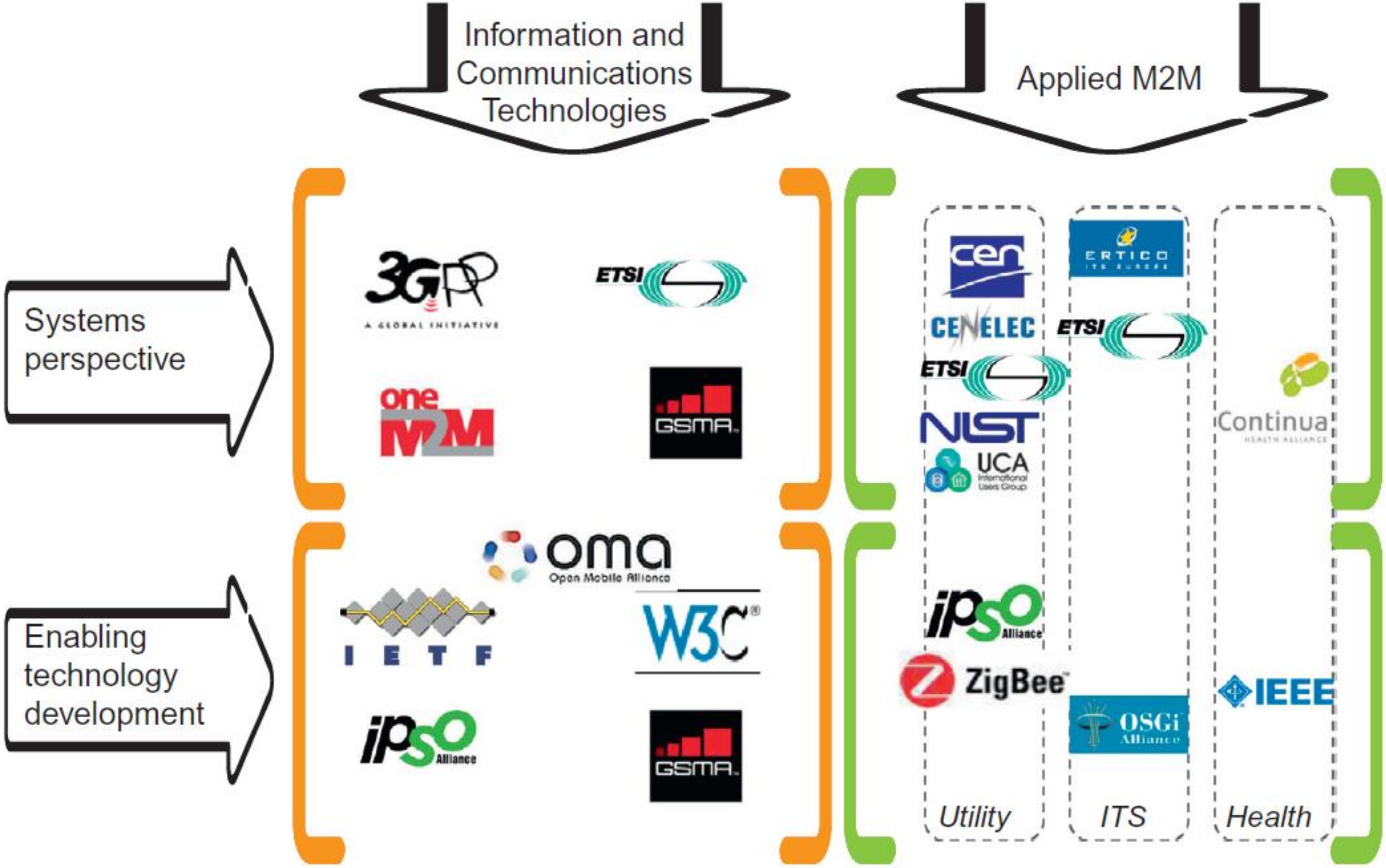


FIGURE 4.4

The landscape of M2M and IoT standardization.

The diagram illustrates the landscape of M2M (Machine to Machine) and IoT (Internet of Things) standardization. It categorizes various organizations into two main perspectives:

1. **Systems Perspective:**

- **3GPP:** Focuses on mobile telecommunications standards.
- **oneM2M**: Develops standards for M2M and IoT interoperability.
- **ETSI**: European Telecommunications Standards Institute, involved in various ICT standards.
- **Enabling Technology Development**: Includes organizations like IETF (Internet Engineering Task Force) and OMA (Open Mobile Alliance), which develop foundational technologies.

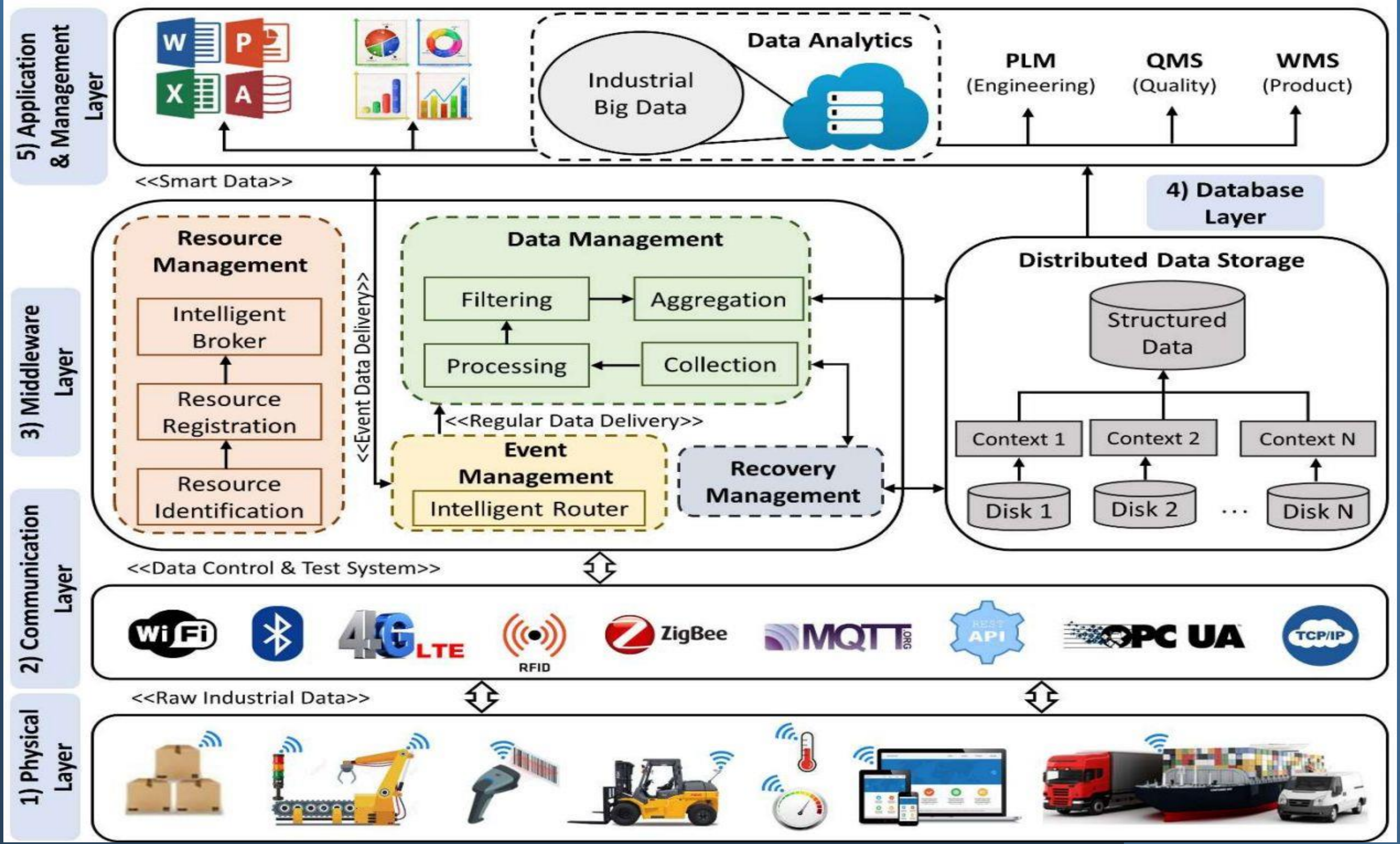
2. **Information and Communication Technologies:**

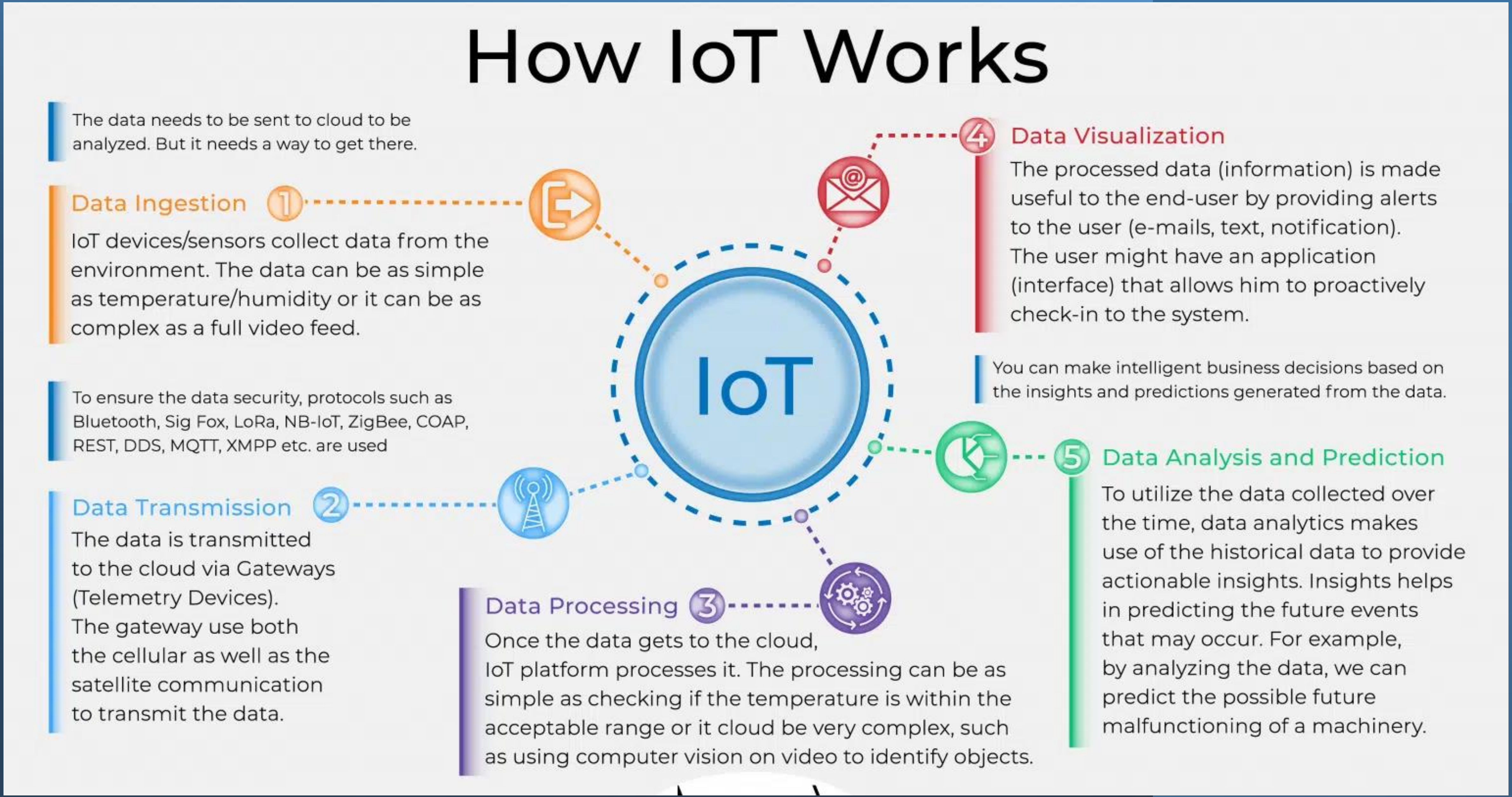
- **Applied M2M**: Includes specific applications like Smart M2M (CEN/CENELEC/ETSI), Continua Health Alliance, and ITS (Intelligent Transport Systems) with ZigBee and IEEE for health standards.
- **Utility:** Although not specified with logos, this likely refers to standards for utility services like smart grids.

IoT data Management

- IoT data management refers to the comprehensive process of handling data generated by IoT devices from collection to final use.
- This involves several critical components to ensure that data is
  - efficiently collected
  - Transmitted
  - Stored
  - processed and
  - Analyzed

ultimately providing valuable insights and facilitating decision-making.


5) Application & Management Layer
Data Analytics
Industrial Big Data
PLM (Engineering)
QMS (Quality)
WMS (Product)
<<Smart Data>>
4) Database Layer
3) Middleware Layer
Resource Management
Intelligent Broker
Resource Registration
Resource Identification
<<Event Data Delivery>>
Data Management
Filtering
Aggregation
Processing
Collection
<<Regular Data Delivery>>
Event Management
Intelligent Router
Recovery Management
Distributed Data Storage
Structured Data
Context 1
Context 2
Context N
Disk 1
Disk 2
...
Disk N
<<Data Control & Test System>>
2) Communication Layer
WiFi
4G LTE
RFID
ZigBee
MQTT.ORG
REST API
OPC UA
TCP/IP
<<Raw Industrial Data>>
1) Physical Layer



How IoT Works

The data needs to be sent to cloud to be analyzed. But it needs a way to get there.

Data Ingestion ①

IoT devices/sensors collect data from the environment. The data can be as simple as temperature/humidity or it can be as complex as a full video feed.

To ensure the data security, protocols such as Bluetooth, Sig Fox, LoRa, NB-IoT, ZigBee, COAP, REST, DDS, MQTT, XMPP etc. are used

Data Transmission ②

The data is transmitted to the cloud via Gateways (Telemetry Devices).
The gateway use both the cellular as well as the satellite communication to transmit the data.

Data Processing ③

Once the data gets to the cloud,
IoT platform processes it. The processing can be as simple as checking if the temperature is within the acceptable range or it cloud be very complex, such as using computer vision on video to identify objects.

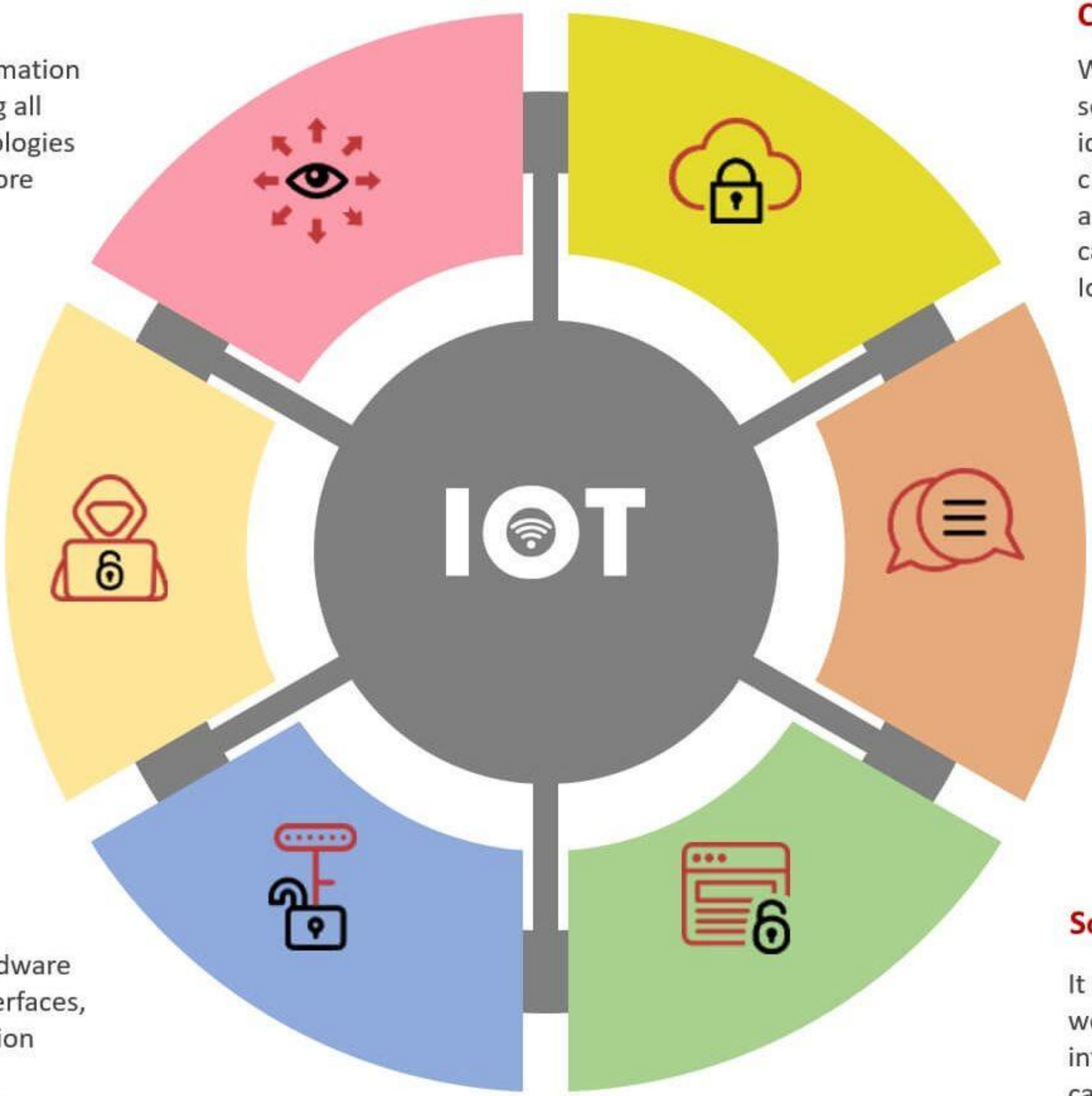
④ Data Visualization

The processed data (information) is made useful to the end-user by providing alerts to the user (e-mails, text, notification).
The user might have an application (interface) that allows him to proactively check-in to the system.

You can make intelligent business decisions based on the insights and predictions generated from the data.

⑤ Data Analysis and Prediction

To utilize the data collected over the time, data analytics makes use of the historical data to provide actionable insights. Insights helps in predicting the future events that may occur. For example, by analyzing the data, we can predict the possible future malfunctioning of a machinery.



Reconnaissance

Gathering most of the information about the target considering all the components and technologies in use. This is first stage before hacking the target

Firmware Hacking

Analyzing the firmware for hardcoded secrets such as API keys, Sensitive URLs, Credentials, Sensitive Services running on the device along with analyzing binaries to discover web/API related flaws

Hardware Hacking

It covers exploiting the hardware device against exposed interfaces, internal serial communication ports, firmware extraction techniques and tamper resistance testing

Cloud Security

With the popularity of the Cloud services, additional severe flaws are identified and with a misconfigured cloud server, an attacker can gain access to sensitive information and can successfully compromise entire IoT Infrastructure

Communication

IoT devices often uses popular communication protocols like MQTT, Zigbee, BLE, CoAP, etc. which can be tested for cryptographic security, ability to sniff traffic and modify it from an attacker's perspective

Software Hacking

It consists of penetrating testing if web and mobile applications interacting with the things which can lead to facilitate compromise of IoT Infrastructure

Key Components of IoT Data Management:

1. **Data Collection:**
   - Gathering data from various IoT devices and sensors.
   - Using APIs, IoT gateways, and direct device communication protocols to capture data.
2. **Data Transmission**:
   - Transferring collected data from devices to storage or processing systems.
   - Utilizing communication protocols like MQTT, CoAP, HTTP, and networking technologies such as Wi-Fi, Bluetooth, Zigbee, and cellular networks to ensure reliable data transfer.
3. **Data Storage**:
   - Storing the vast amounts of data generated by IoT devices.
   - Employing databases (SQL and NoSQL), data lakes, cloud storage solutions, and edge computing to store data efficiently.

4. **Data Processing**:

- Analyzing and processing raw data to derive meaningful insights.
- Utilizing data processing frameworks, stream processing (e.g., Apache Kafka, Apache Storm), and batch processing (e.g., Hadoop) to handle and process data in real-time or in batches.

5. **Data Integration:**

- Combining data from multiple sources to provide a unified view.
- Using integration platforms and middleware to ensure interoperability between IoT devices and data sources.

6. **Data Analytics:**

- Analyzing data to extract insights, patterns, and trends.
- Applying machine learning, artificial intelligence, and statistical methods to process and analyze data, providing predictive analytics and actionable insights.

7. Data Visualization:

- Presenting data in an understandable and actionable format.
- Using dashboards, graphs, and other visualization tools to display data analytics results to stakeholders.

8. Data Security:

- Ensuring the protection of data throughout its lifecycle.
- Implementing encryption, authentication, access control, and other security measures to protect data from unauthorized access and breaches.

9. Data Governance:

- Managing the availability, usability, integrity, and security of data.
- Establishing policies, procedures, and standards to ensure data quality and compliance with regulations.

How IoT Data Management is Implemented in an IoT Solution:

Edge Computing: Processing data at the edge of the network (near the data source) to reduce latency and bandwidth usage.

Cloud Computing: Utilizing cloud platforms for scalable data storage and processing capabilities.

Data Lakes: Creating centralized repositories that allow for the storage of structured and unstructured data at any scale.

Stream Processing: Analyzing data in real-time as it is ingested, allowing for immediate insights and actions.

Batch Processing: Handling and processing large volumes of data at scheduled intervals for comprehensive analysis.

IoT Data and Services Management:

- **IoT Data Management**: Ensuring data is collected, transmitted, stored, processed, and analyzed efficiently and securely.
- **IoT Services Management**: Orchestrating and managing IoT services, including device management, network management, and service provisioning.

IoT architecture-State of art solution

- The M2M system architectures are naturally more communication-oriented, while the IoT-related reference architectures and models are more holistic in their scope.

1. **European Telecommunications Standards Institute M2M/oneM2M**

- In 2009, the European Telecommunications Standards Institute (ETSI) formed a Technical Committee (TC) on M2M (Machine-to-Machine) topics to create end-to-end communication standards among machines.
- This committee included representatives from telecom network operators, equipment vendors, administrations, research bodies, and specialist companies.
- The ETSI M2M specifications were based on standards from ETSI and other organizations like the IETF, 3GPP, OMA, and BBF. ETSI released the first M2M standards in early 2012. Later in 2012, seven leading ICT standards organizations (ARIB, TTC, ATIS, TIA, CCSA, ETSI, TTA) formed the oneM2M Partnership Project to develop global M2M specifications and promote M2M business.

ETSI M2M high-level architecture

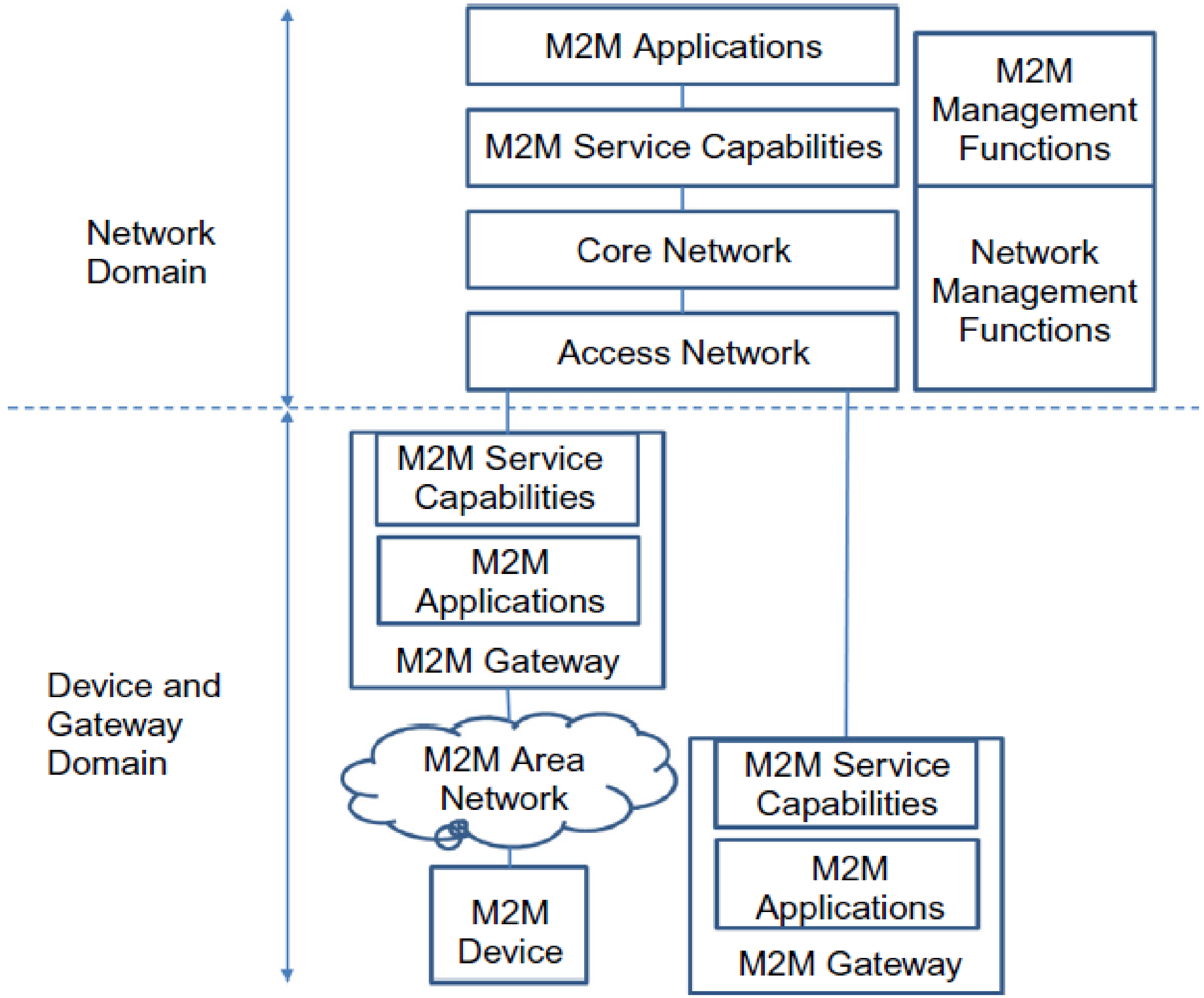


FIGURE 6.1

ETSI M2M High-Level Architecture.

This architecture combines functional and topological views, distinguishing between physical infrastructure (e.g., M2M Devices, Gateways) and other functional groups. It consists of two main domains: the Network Domain and the Device and Gateway Domain.

Device and Gateway Domain:

- **M2M Device**: A device with M2M applications and service capabilities (e.g., a temperature sensor). It connects to the Network Domain either directly or through an M2M Gateway.
    - **Direct Connection**: The M2M Device handles registration, authentication, authorization, management, and provisioning, and communicates with the Access Network.
    - **Through M2M Gateway**: Used when the M2M device lacks appropriate physical layers for the Access Network. The gateway acts as a proxy, performing necessary procedures.
- **M2M Area Network**: Provides connectivity between M2M Devices and M2M Gateways using technologies like Bluetooth, ZigBee, MBUS, KNX, etc.
- **M2M Gateway**: Connects M2M Devices to the Network Domain, containing M2M applications and service capabilities, and may serve legacy devices.

Network Domain:

- **Access Network**: Allows devices in the Device and Gateway Domain to communicate with the Core Network using technologies like xDSL, Satellite, WLAN, WiMAX, etc.
- **Core Network**: Provides IP connectivity, service, network control, interconnection, and roaming. Examples include the 3GPP Core Network and ETSI TISPAN Core Network.
- **M2M Service Capabilities**: Exposes functions to M2M Applications through open interfaces, abstracting network functions.
- **M2M Applications**: Specific applications (e.g., smart metering) utilizing M2M Service Capabilities.
- **Network Management Functions**: Manage the Access and Core Network (e.g., provisioning, fault management).
- **M2M Management Functions**: Manage M2M Service Capabilities on the Network Domain, including:
  - **M2M Service Bootstrap Function (MSBF)**: Facilitates bootstrapping of security credentials.
  - **M2M Authentication Server (MAS)**: Stores permanent security credentials in a secure environment.

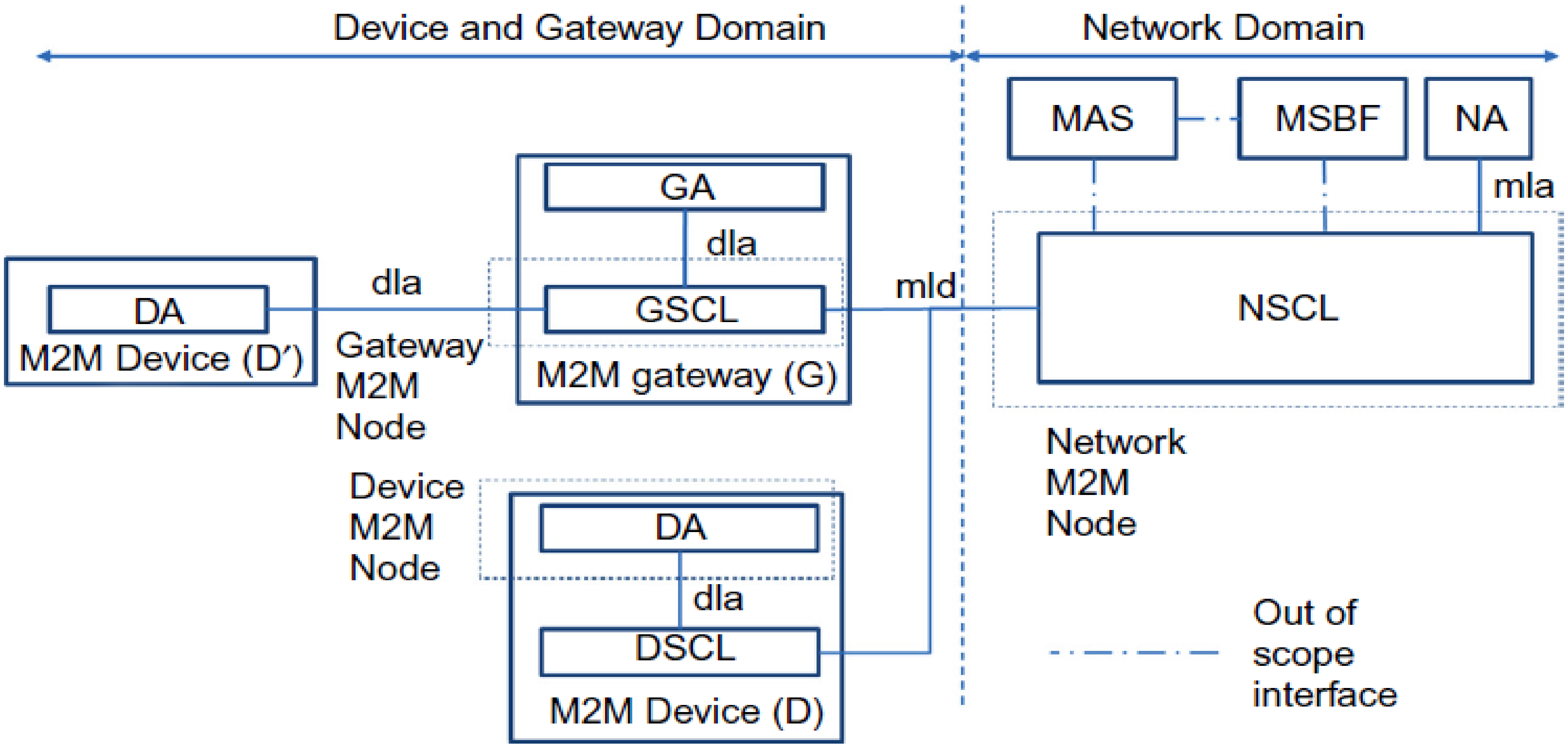


FIGURE 6.2

M2M Service Capabilities, M2M Nodes and Open Interfaces.

Device and Gateway Domain

- **M2M Device (D and D')**:
  - **DA (Device Application)**: This is the application running on the M2M device.
  - **DSCL (Device Service Capability Layer)**: Provides the service capabilities for the M2M device D.
  - **dla (Device Local Application)**: Represents the interface between the DA and DSCL.
- **M2M Gateway (G)**:
  - **GA (Gateway Application)**: This is the application running on the M2M gateway.
  - **GSCL (Gateway Service Capability Layer)**: Provides the service capabilities for the M2M gateway.
  - **dla**: Represents the interface between the GA and GSCL.
  - **mld (M2M Local Domain)**: Represents the interface between the GSCL and NSCL in the network domain.

Network Domain

- **NSCL (Network Service Capability Layer)**: Provides service capabilities at the network level, interacting with the GSCL in the device and gateway domain.
- **MAS (M2M Application Server)**: This is a server running M2M applications.
- **MSBF (M2M Service Bootstrap Function)**: Responsible for the initial setup and configuration of M2M devices.
- **NA (Network Application)**: Represents applications running at the network level.
- **mla (M2M Local Application)**: Interface within the network domain.

Interfaces

- **dla (Device Local Application Interface)**: Used for communication between applications and service capabilities within the same device or gateway.
- **mld (M2M Local Domain Interface)**: Used for communication between the gateway service capability layer and the network service capability layer.
- **mla (M2M Local Application Interface)**: Used for communication between different network applications and service capabilities.

IoT Reference Model

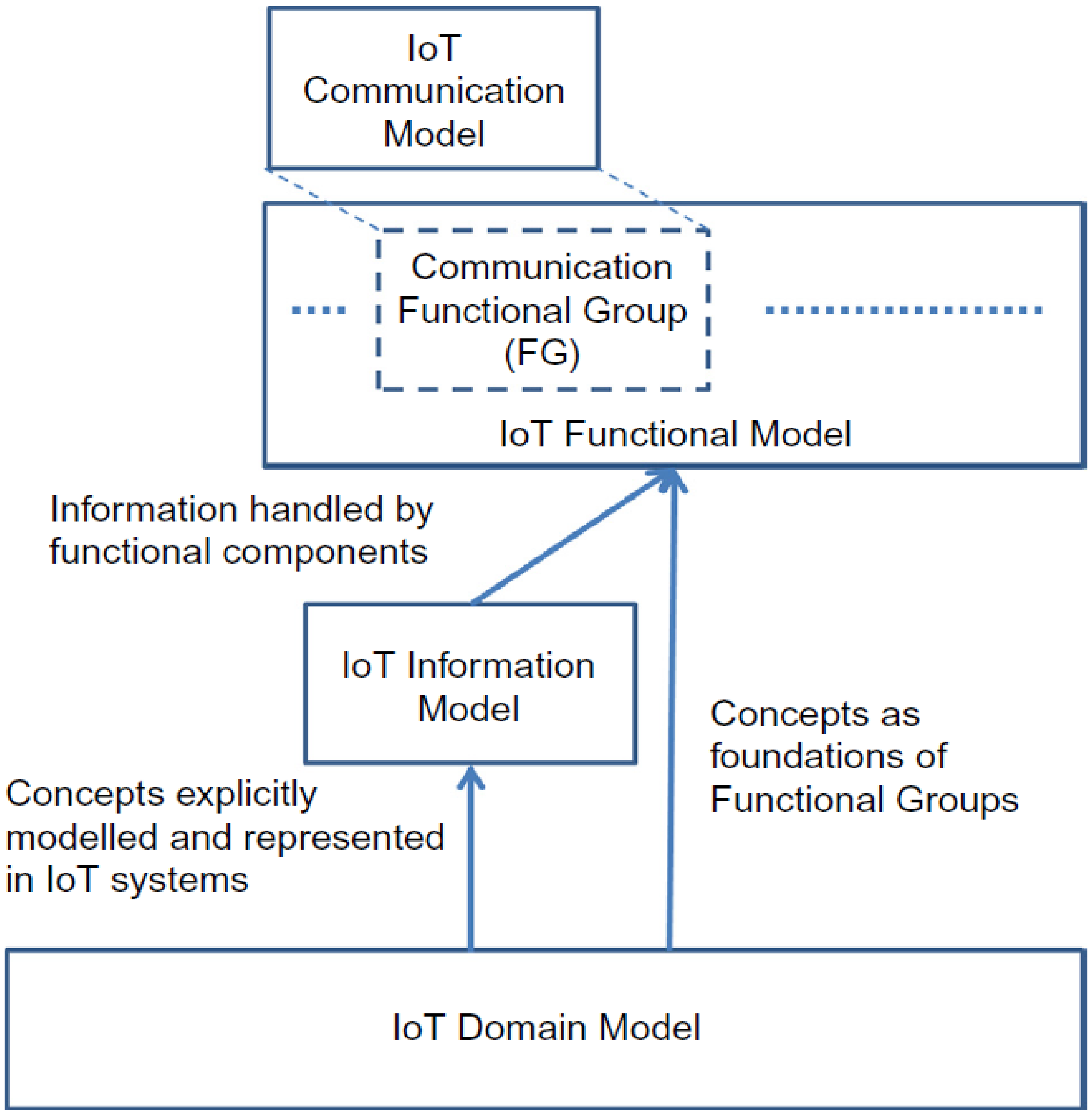

IoT Communication Model
Communication Functional Group (FG)
IoT Functional Model
Information handled by functional components
IoT Information Model
Concepts explicitly modelled and represented in IoT systems
Concepts as foundations of Functional Groups
IoT Domain Model

FIGURE 7.1

IoT Reference Model.

IoT Communication Model

- This model represents the communication aspects of the IoT system.
- It is part of the broader IoT Functional Model and interacts with the Communication Functional Group (FG).

IoT Functional Model

- Encompasses various functional groups, including the Communication Functional Group (FG).
- It handles the information provided by the IoT Information Model.
- Defines how different functional components operate and interact within the IoT system.

IoT Information Model

- This model deals with the information managed by the functional components of the IoT system.
- Acts as a bridge between the IoT Functional Model and the IoT Domain Model.
- Provides a structured representation of the concepts explicitly modeled and represented in IoT systems.

IoT Domain Model

- Serves as the foundation for the concepts used in the IoT Information Model.
- Provides the basic concepts and relationships that underpin the functional groups and components of the IoT system.
- These concepts are used as the foundations for creating functional groups within the IoT Functional Model.

Relationships and Interactions

- The **IoT Domain Model** provides the fundamental concepts that are used to develop the **IoT Information Model**.
- The **IoT Information Model** takes these concepts and structures them into information that can be managed and used by the functional components within the **IoT Functional Model**.
- The **IoT Functional Model** includes various functional groups (such as the Communication Functional Group) that handle and process the information structured by the IoT Information Model.
- The **IoT Communication Model**, as part of the IoT Functional Model, specifies the communication mechanisms and protocols used in the IoT system.

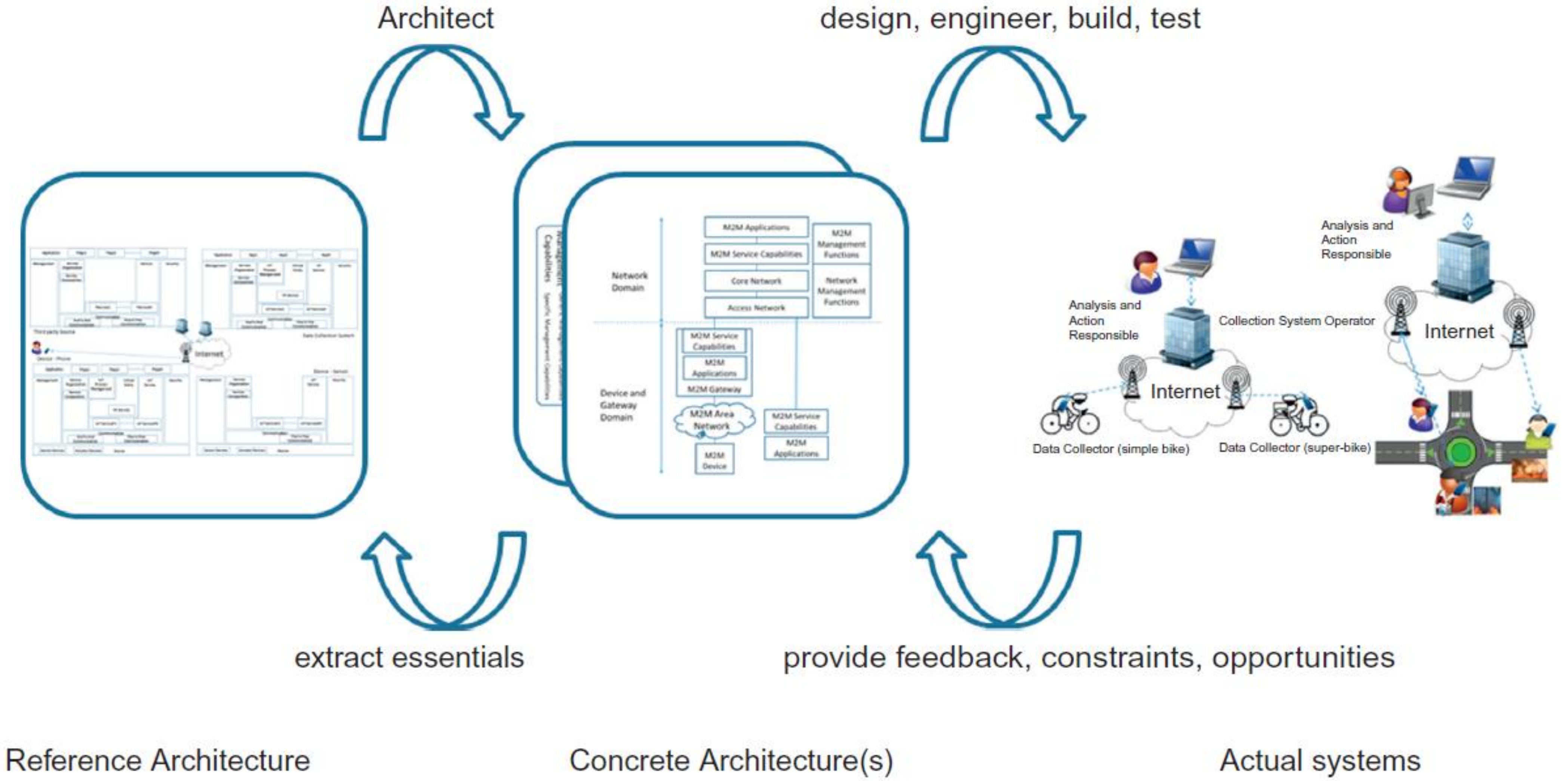


FIGURE 7.2

From reference to concrete architectures and actual systems.

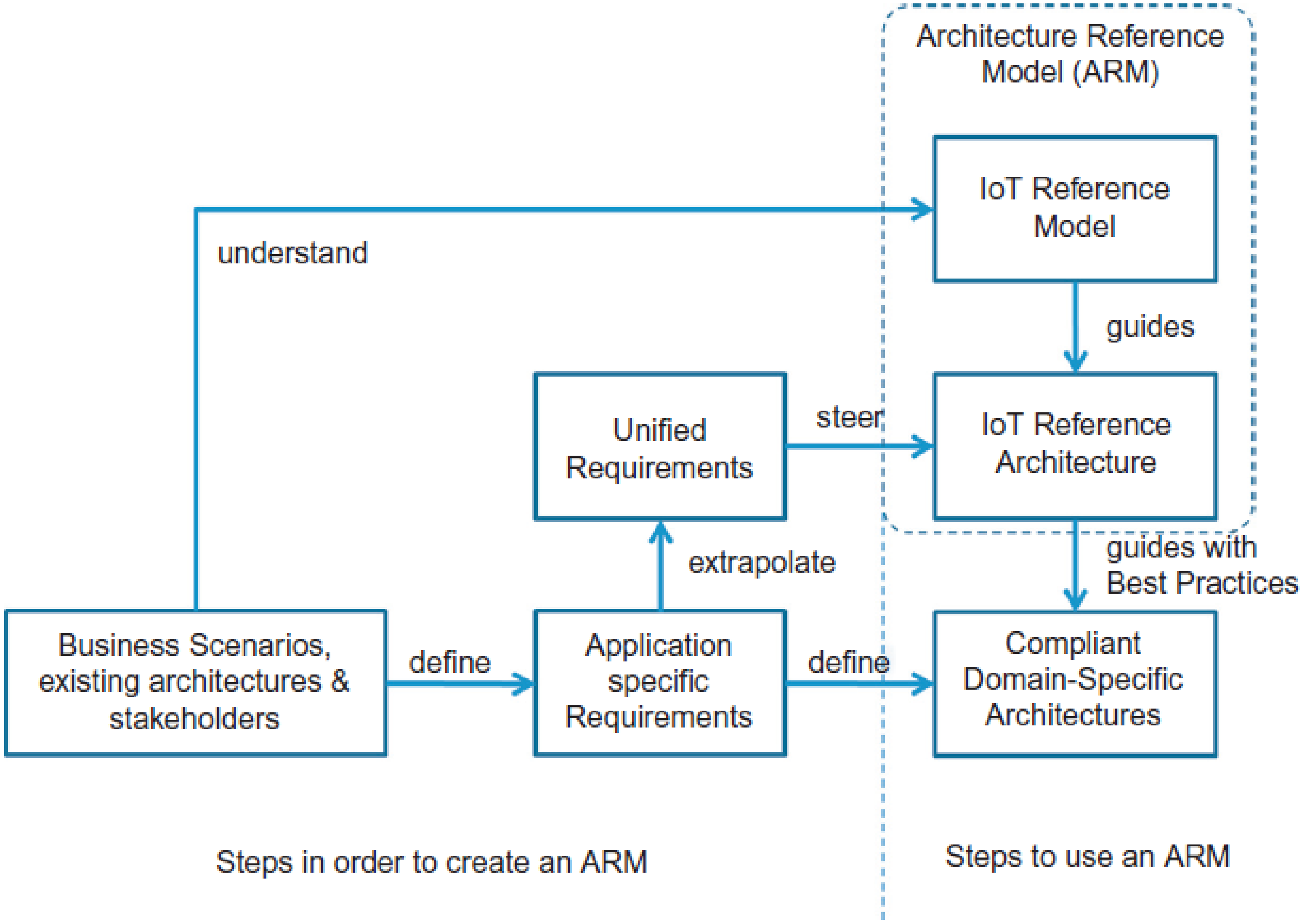

Architecture Reference Model (ARM)
IoT Reference Model
understand
guides
Unified Requirements
steer
IoT Reference Architecture
extrapolate
guides with Best Practices
Business Scenarios, existing architectures & stakeholders
define
Application specific Requirements
define
Compliant Domain-Specific Architectures
Steps in order to create an ARM
Steps to use an ARM

FIGURE 7.3

IoT Reference Model and Reference Architecture dependencies.

IoT domain model

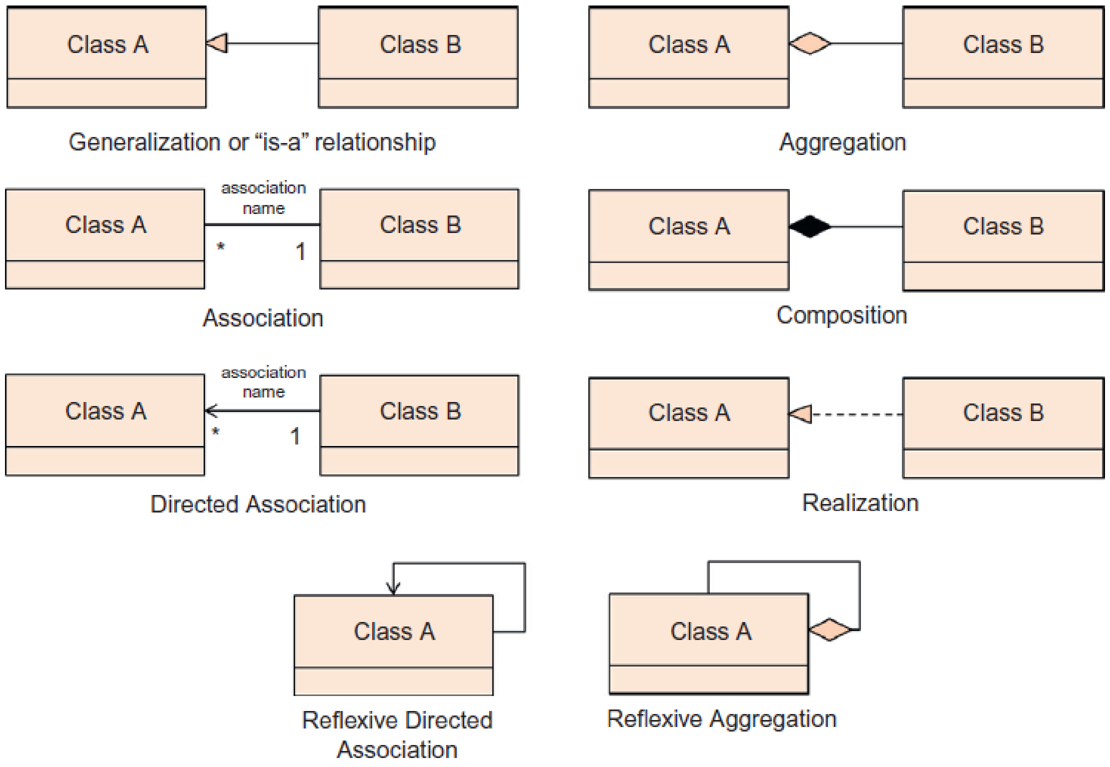


FIGURE 7.4

UML Class diagram main modeling concepts.

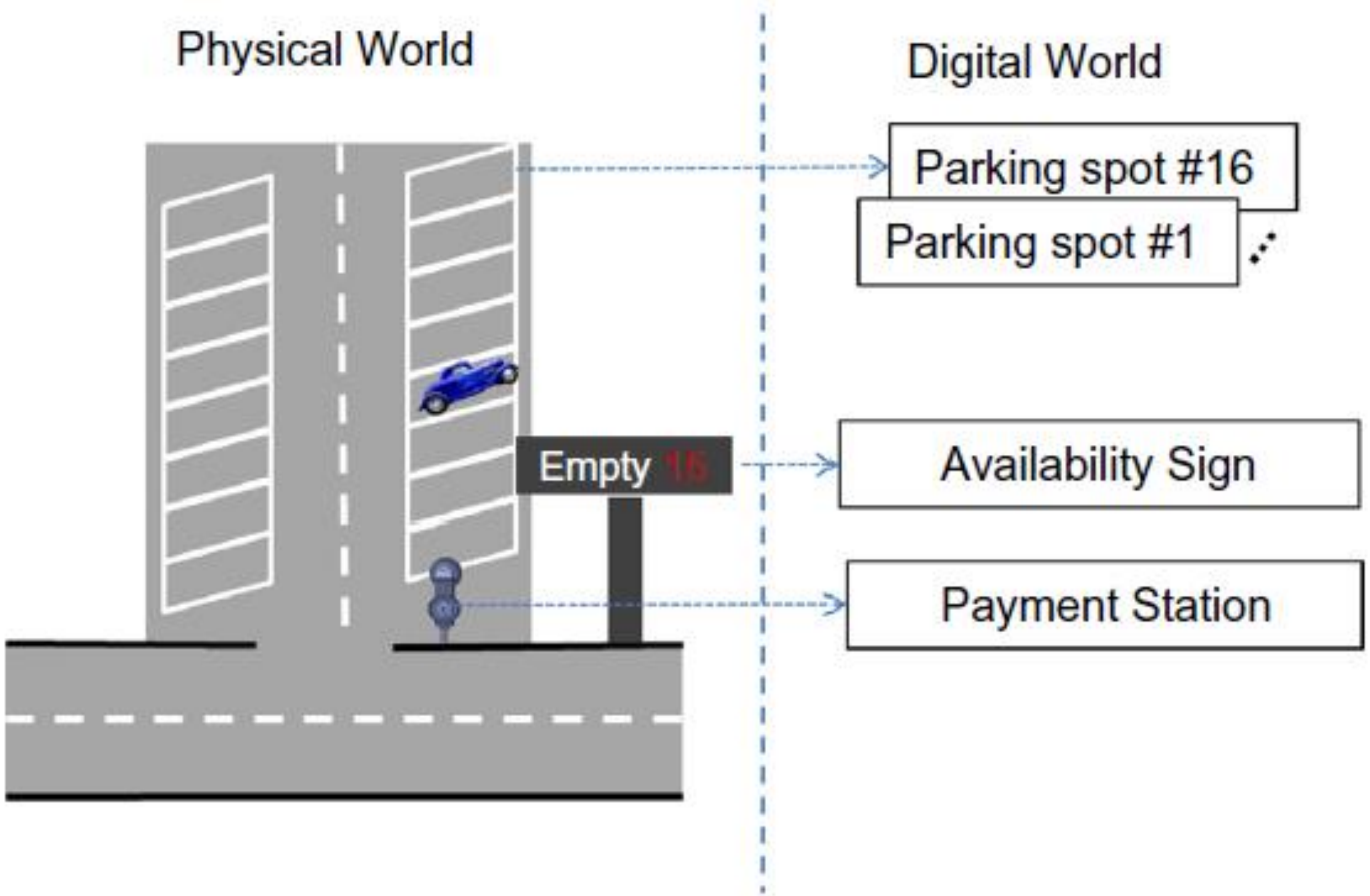


FIGURE 7.5

Physical vs. Virtual World.

- Imagine monitoring a parking lot with 16 spots, a payment station, an electronic sign displaying available spots, and a smartphone app for frequent customers to check availability before arriving.
- Physical objects, such as parking spots and the payment station, are represented digitally as variables or database objects.
- This allows the software to detect unpaid parking, inform drivers about spot availability, and generate occupancy statistics.
- Each spot has a sensor with a digital representation (e.g., "Parking spot #1" to "Parking spot #16"), indicating whether it's available or occupied.
- The payment station and availability sign are also digitally represented to ensure payments are tracked and the sign functions correctly.

- ***This example highlights a key difference between IoT and today's Internet: while the current Internet focuses on virtual content and services, IoT emphasizes interaction with physical Things.***
- ***In M2M (Machine-to-Machine) communication, devices are accessed through a network, but IoT prioritizes Thing-oriented interaction over communication-oriented interaction.***

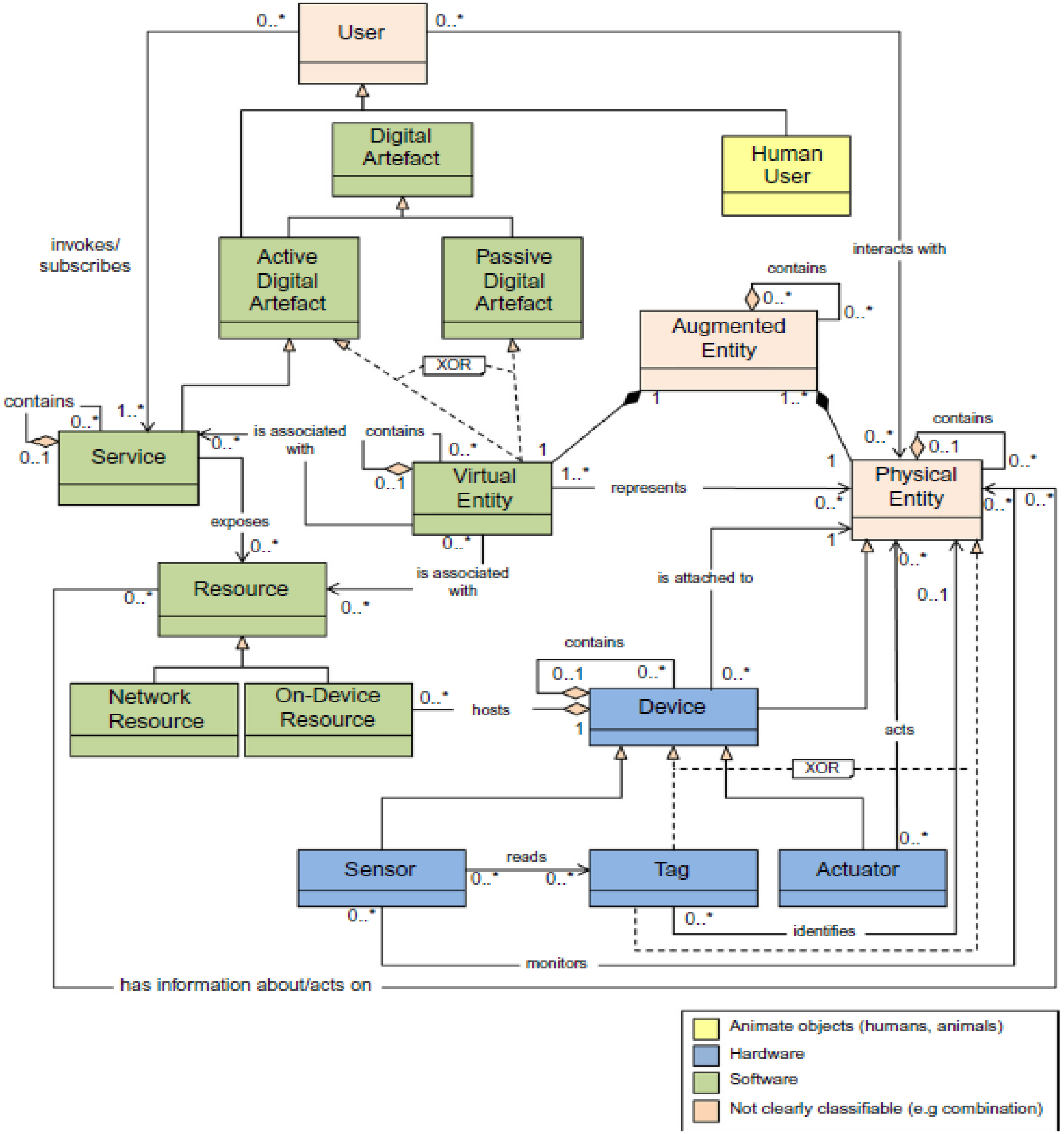


FIGURE 7.6

IoT Domain Model.

Key Elements and Relationships:

1. **User**:
   Represents the person who interacts with digital artifacts in the IoT system.
2. **Human User**:
   A specific type of user that can interact with augmented entities and physical entities.
3. **Digital Artefact**:
   1. Can be either an **Active Digital Artefact** or a **Passive Digital Artefact**.
   2. An **Active Digital Artefact** contains and is associated with services.
   3. A **Passive Digital Artefact** is associated with a virtual entity.
4. **Service**:
   Exposes and is associated with resources, which can be either network or on-device resources.
5. **Resource**:
   Network Resource and **On-Device Resource** are types of resources that services can expose and interact with.

6. Virtual Entity:
- Represents a digital representation of a physical entity.
- It is associated with both active and passive digital artifacts.

7. Augmented Entity:
- A combination of virtual and physical entities.
- Interacts with users and is attached to devices.

8. Physical Entity:
- Represents a real-world object that can be monitored or acted upon by devices.

9. Device:
- Hosts and contains sensors, tags, and actuators.
- Can monitor, read, and identify physical entities through these components.

10. Sensor:
- Reads and gathers data from physical entities and sends it to devices.

11. Tag:
- Identifies physical entities and can be read by devices.

12. Actuator:
- Acts upon physical entities based on commands received from devices.

For the IoT Domain Model, three kinds of Device types are the most important:

1. Sensors:

- These are simple or complex Devices that typically involve a transducer that converts physical properties such as temperature into electrical signals.
- These Devices include the necessary conversion of analog electrical signals into digital signals, e.g. a voltage level to a 16-bit number, processing for simple calculations, potential storage for intermediate results, and potentially communication capabilities to transmit the digital representation of the physical property as well receive commands.
- A video camera can be another example of a complex sensor that could detect and recognize people.

2. Actuators:

- These are also simple or complex Devices that involve a transducer that converts electrical signals to a change in a physical property (e.g. turn on a switch or move a motor).
- These Devices also include potential communication capabilities, storage of intermediate commands, processing, and conversion of digital signals to analog electrical signals.

3. Tags:

- Tags in general identify the Physical Entity that they are attached to. In reality, tags can be Devices or Physical Entities but not both, as the domain model shows.
- An example of a Tag as a Device is a Radio Frequency Identification (RFID) tag, while a tag as a Physical Entity is a paper-printed immutable barcode or Quick Response (QR) code.
- Either electronic Devices or a paper-printed entity tag contains a unique identification that can be read by optical means (bar codes or QR codes) or radio signals (RFID tags).
- The reader Device operating on a tag is typically a sensor, and sometimes a sensor and an actuator combined in the case of writable RFID tags.

As shown in Figure 7.6

- Devices can aggregate other devices, such as a sensor node containing a temperature sensor, an LED (actuator), and a buzzer (actuator).
- Every IoT device needs to either have energy reserves (like a battery), be connected to the power grid, or perform energy scavenging (e.g., converting solar energy).
- The device's communication, processing, storage, and energy reserve capabilities influence design decisions, such as whether resources should be on-device, if the device should enter sleep mode, and if data should be saved locally or transmitted immediately.

IoT services can be classified into three main classes based on their level of abstraction:

1. **Resource-Level Services**: Expose the functionality of a device by exposing on-device resources, handling aspects like security, availability, and performance. They also include network resources hosted on powerful machines or in the cloud, such as historical databases of specific resource measurements.
2. **Virtual Entity-Level Services**: Provide information or interaction capabilities about virtual entities, with service interfaces typically including the identity of the virtual entity.
3. **Integrated Services**: Compositions of resource-level and virtual entity-level services, or any combination of both.

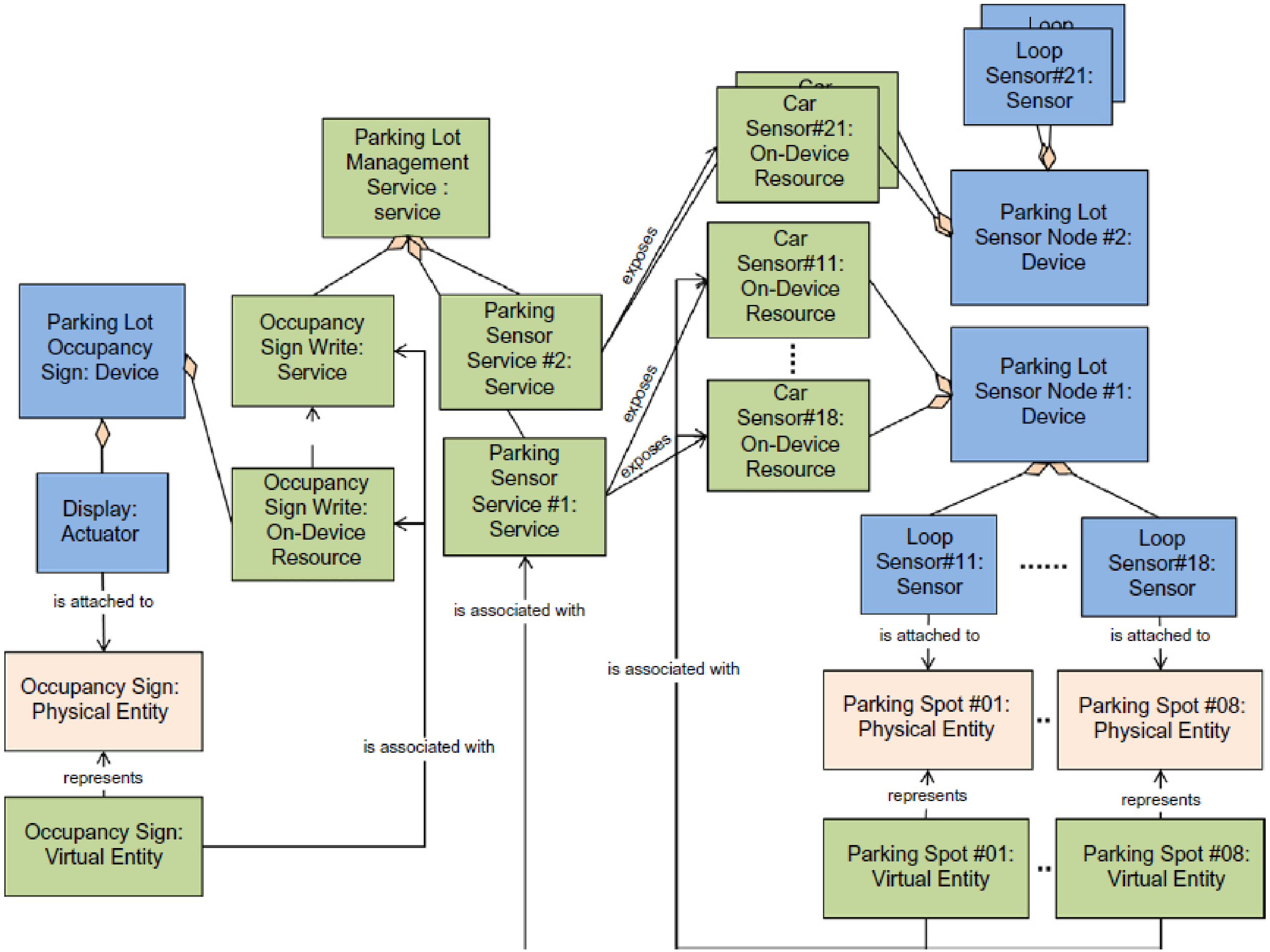

Loop
Loop Sensor#21: Sensor
Car
Car Sensor#21: On-Device Resource
Parking Lot Management Service : service
Parking Lot Sensor Node #2: Device
exposes
Car Sensor#11: On-Device Resource
Parking Lot Occupancy Sign: Device
Occupancy Sign Write: Service
Parking Sensor Service #2: Service
Parking Lot Sensor Node #1: Device
exposes
Car Sensor#18: On-Device Resource
exposes
Parking Sensor Service #1: Service
Display: Actuator
Occupancy Sign Write: On-Device Resource
Loop Sensor#11: Sensor
Loop Sensor#18: Sensor
is attached to
is associated with
is attached to
is attached to
is associated with
Occupancy Sign: Physical Entity
Parking Spot #01: Physical Entity
Parking Spot #08: Physical Entity
is associated with
represents
represents
represents
Occupancy Sign: Virtual Entity
Parking Spot #01: Virtual Entity
Parking Spot #08: Virtual Entity

FIGURE 7.7

IoT Domain Model instantiation.

Information model

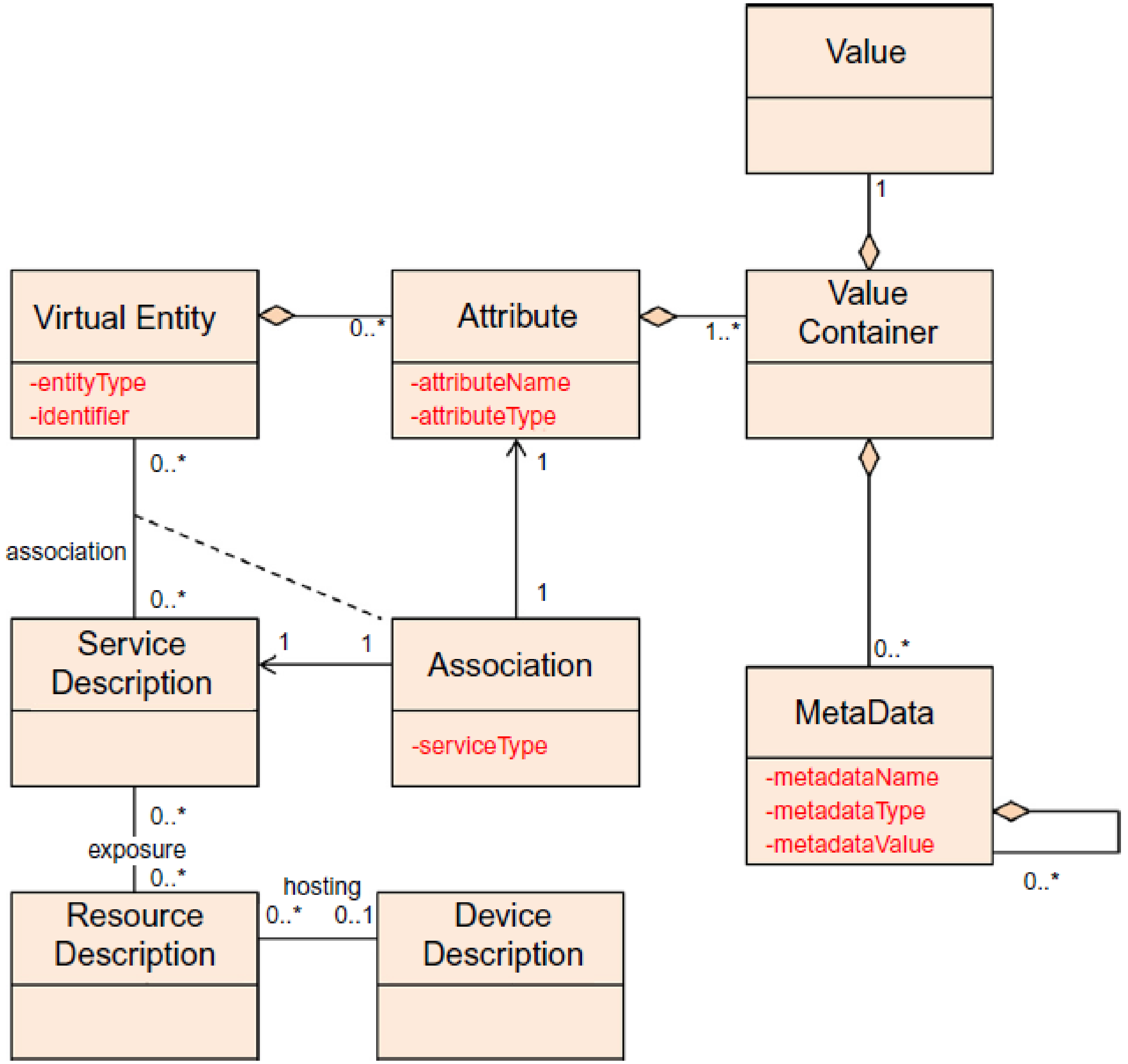


FIGURE 7.8

High-level IoT Information Model.

- The IoT Information Model, like the IoT Domain Model, uses UML diagrams to represent classes and their attributes.
- Attributes are usually simple types (e.g., integers or strings) and are displayed in red text under the class name.
- Complex attributes are represented as separate classes with aggregation relationships.
- This model maintains information about virtual entities and their properties, which can be static or dynamic, entered manually, or read from sensors.
- Virtual entity attributes may also reflect the state of actuators.

Key points of the IoT Information Model:

1. **Resource Exposure**:

Resources expose functionality as services with standardized interfaces, abstracting low-level details. Users interact with physical entities through virtual entities associated with these services.

2. **Service Classes**:

- ***Resource-Level Services***: Expose device functionality, handling security, availability, and performance. They include both on-device and network resources.
- ***Virtual Entity-Level Services***: Provide information or interaction capabilities about virtual entities.
- ***Integrated Services***: Combinations of resource-level and virtual entity-level services.

3. **Attributes:**

- Simple attributes (e.g., entityType, identifier) are associated with virtual entities.
- Complex attributes are grouped under the class Attributes, which includes sub-attributes like attributeName and attributeType.
- ValueContainer holds multiple values for an attribute, annotated with metadata (e.g., timestamp).

4. **Associations**:

- The Association class captures relationships between virtual entities and services, linking attributes to service providers (e.g., sensors or actuators).

Functional model

- The IoT Functional Model aims to describe mainly the Functional Groups (FG) and their interaction with the ARM.
- In contrast, the Functional View of a Reference Architecture describes the functional components of an FG, interfaces, and interactions between the components.
- The Functional View is typically derived from the Functional Model in conjunction with high-level requirements.

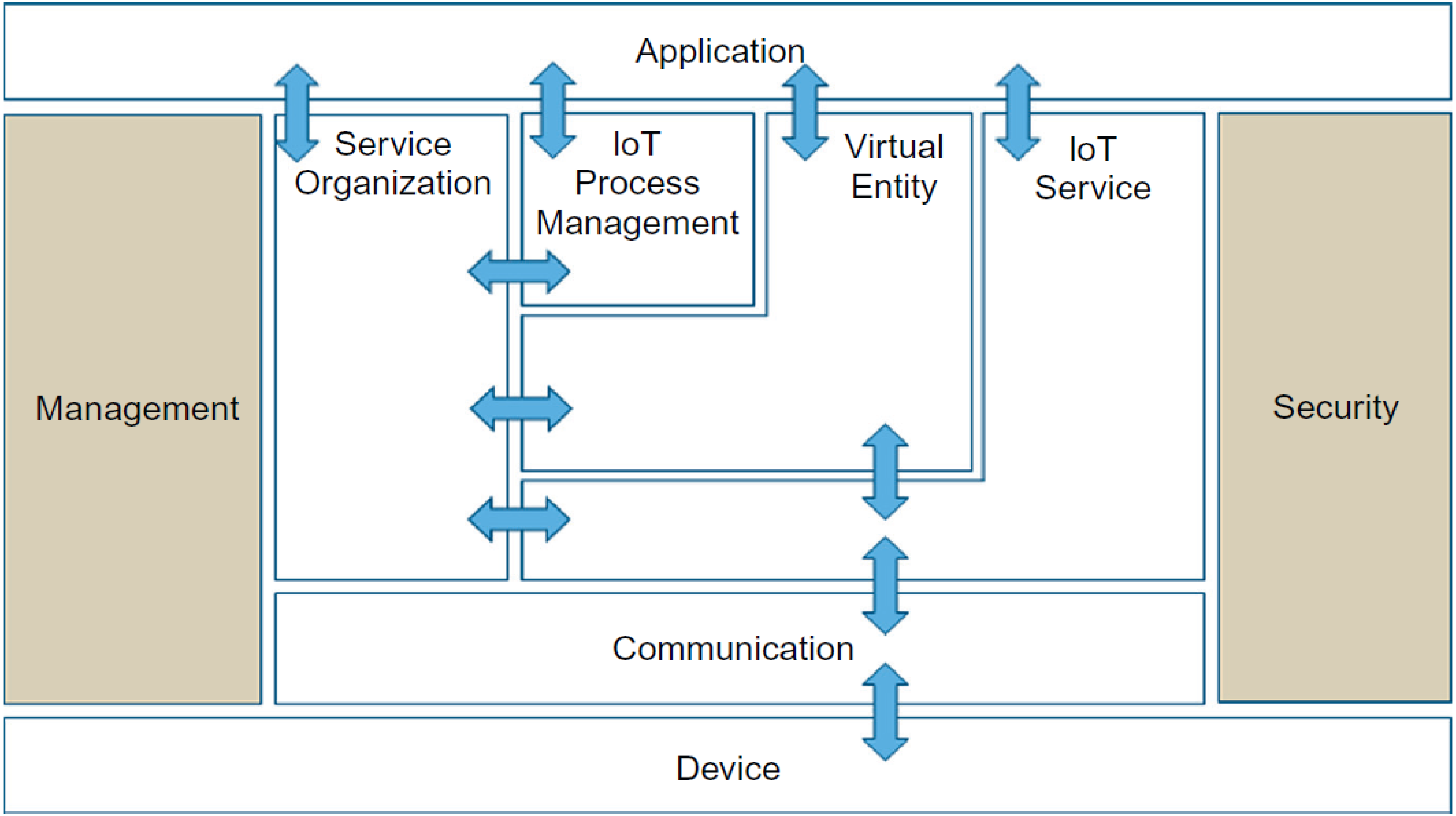


FIGURE 7.11

IoT-A Functional Model.

- **Device functional group**
- The Device FG contains all the possible functionality hosted by the physical Devices used for instrumenting the Physical Entities.
- This Device functionality includes sensing, actuation, processing, storage, and identification components, the sophistication of which depends on the Device’s capabilities.
- **Communication functional group**
- The Communication FG abstracts all the possible communication mechanisms used by the relevant Devices in an actual system to transfer information to the digital world components or other Devices.
- Examples of such functions include wired bus or wireless mesh technologies through which sensor Devices are connected to Internet Gateway Devices.

- **IoT Service functional group**
- The IoT Service FG corresponds mainly to the Service class from the IoT Domain Model and contains single IoT Services exposed by Resources hosted on Devices or in the Network (e.g. processing or storage Resources).
- Support functions such as directory services, which allow discovery of Services and resolution to Resources, are also part of this FG.

- **Virtual Entity functional group**
- The Virtual Entity FG corresponds to the Virtual Entity class in the IoT Domain Model and manages associations between Virtual Entities and IoT Services.
- These associations can be static, like a building containing floors and rooms, or dynamic, like a car moving between city blocks.
- A key distinction between IoT Services and Virtual Entity Services lies in their request-response semantics.
- For example, a Parking Sensor Service may simply return a binary value ("0" or "1"), while a Virtual Entity, such as Parking Spot #01, would provide more meaningful information ("free").
- IoT Services deliver data linked to specific devices, whereas Virtual IoT Services offer richer, more human-readable information.

- **IoT Service Organization functional group**
- The purpose of the IoT Service Organization FG is to host all functional components that support the composition and orchestration of IoT and Virtual Entity services.
- For example, service requests from Applications or the IoT Process Management are directed to the Resources implementing the necessary Services.
- Therefore, the Service Organization FG supports the association of Virtual Entities with the related IoT Services and contains functions for discovery, composition, and choreography of services.
- Simple IoT or Virtual Entity Services can be composed to create more complex services, e.g. a control loop with one Sensor Service and one Actuator service to control the temperature in a building.
- Choreography is the brokerage of Services so that Services can subscribe to other services in a system.

- **IoT Process Management functional group**
- The IoT Process Management FG is a collection of functionalities that allows smooth integration of IoT-related services (IoT Services, Virtual Entity Services, Composed Services) with the Enterprise (Business) Processes.
- **Management functional group**
- The Management FG includes the necessary functions for enabling fault and performance monitoring of the system, configuration for enabling the system to be flexible to changing User demands and accounting for enabling subsequent billing for the usage of the system.
- Support functions such as management of ownership, administrative domain, rules and rights of functional components, and information stores are also included in the Management FG.

- **Security functional group**
- The Security FG contains the functional components that ensure the secure operation of the system as well as the management of privacy.
- The Security FG contains components for the Authentication of Users (Applications, Humans), Authorization of access to Services by Users, and secure communication (ensuring integrity and confidentiality of messages) between entities of the system such as Devices, Services, Applications, and last but not least, assurance of privacy of sensitive information relating to Human Users.
- These include privacy mechanisms such as anonymization of collected data, anonymization of resource and Service accesses (Services cannot deduce which Human User accessed the data), and un-linkability (an outside observer cannot deduce the Human User of a service by observing multiple service requests by the same User).
- **Application functional group**
- The Application FG is just a placeholder that represents all the needed logic for creating an IoT application. The applications typically contain custom logic tailored to a specific domain such as a Smart Grid.
- An application can also be a part of a bigger ICT system that employs IoT services such as a supply chain system that uses RFID readers to track the movement of goods within a factory to update the Enterprise Resource Planning (ERP) system.

Modular IoT functions

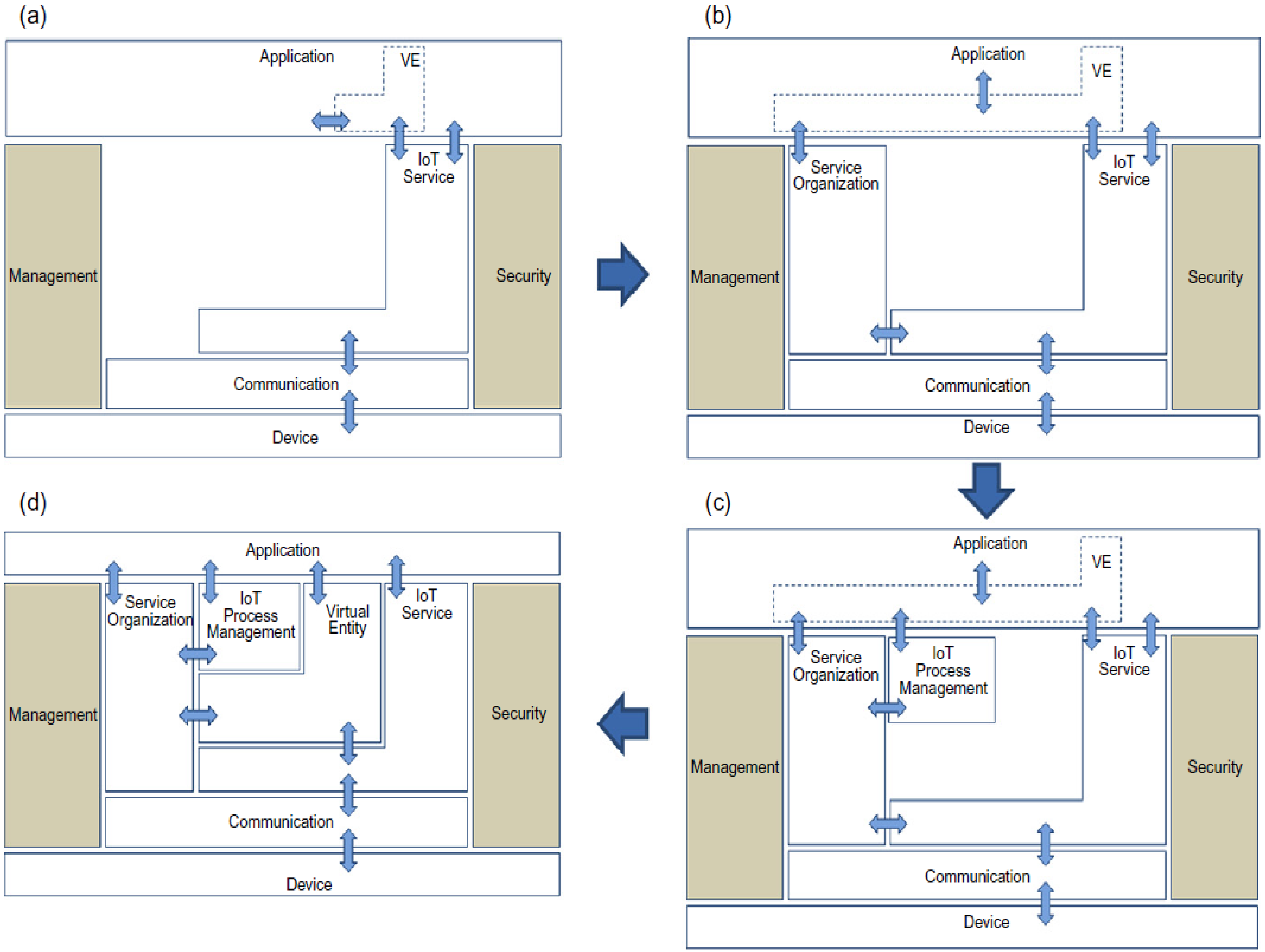


FIGURE 7.12

Building progressively complex IoT Systems.

Communication model

- The communication model for an IoT Reference Model consists of the identification of the endpoints of interactions, traffic patterns (e.g. Unicast vs. multicast), and general properties of the underlying technologies used for enabling such interactions.
- In the IoT Domain Model, communication endpoints include Users (Human Users and Active Digital Artifacts like services and applications), Resources, and Devices.
- Devices with Human-Machine Interfaces facilitate interactions between humans and the physical world but are not considered communication endpoints themselves.

- Communication occurs between Users and Services, as well as between Services and Resources or Devices.
- While most communication uses standard Internet protocols, constrained devices (like embedded systems) may use different communication stacks, requiring gateways to bridge disparate technologies.
- When devices are too limited to host services or resources, these components are moved to more powerful devices or cloud infrastructure, necessitating varied communication stacks to manage these interactions.

Safety, privacy, trust, security model

- An IoT system enables interactions between Human Users and Active Digital Artifacts (Machine Users) with the physical environment.
- The fact that Human Users are part of the system that could potentially harm humans if malfunctioning, or expose private information, motivates the Safety and Privacy needs for the IoT Reference Model and Architecture.
- The Trust and Security Model is needed in every ICT system to protect the digital world.

Safety

- System safety in IoT is highly specific to the application and is critical when actuators could potentially harm humans or animals, such as in an elevator system where safety failures could be dangerous.
- Safety concerns also extend to critical infrastructure, where attacks could cause significant harm, like power outages in hospitals.
- While the IoT Reference Model offers general guidelines for safety, the responsibility lies with system designers to identify hazards and create mitigation plans.
- This process is similar to security threat modeling and includes adding safety checks at key interaction points, such as ensuring elevator doors only open when the elevator is present, with additional mechanical safeguards for added security.

Privacy

Protecting user privacy is crucial in IoT systems, particularly because interactions often involve humans. The IoT-A Privacy Model relies on four key components:

1. **Identity Management**: Derives multiple identities for an entity to anonymize the original user.
2. **Authentication**: Verifies the identity of users, whether original or derived.
3. **Authorization**: Enforces access rights during interactions between users and IoT components.
4. **Trust & Reputation**: Manages trust relationships between entities, affecting their behavior. For instance, an untrusted device might have its data rejected by other components. Trust and reputation are often represented by scores that rank entities based on their reliability and behavior.

Trust

In managing trust within ICT and IoT systems:

- **Trust Model Domains**: Groups of entities with similar trust properties can be organized into different trust domains to simplify management, rather than maintaining trust relationships for every pair.
- **Trust Evaluation Mechanisms**: Define how to compute trust scores for entities, considering the sources of information and concepts like federated trust and trust anchors.
- **Trust Behavior Policies**: Govern interactions based on trust levels, such as how users should handle data from low-trust sensors.
- **Trust Anchor**: A default trusted entity used to evaluate the trustworthiness of other entities within the same model.
- **Federation of Trust**: Rules that manage trust relationships between entities across different trust models, crucial for large-scale systems.

Security

- The Security Model for IoT consists of communication security that focuses mostly on the confidentiality and integrity protection of interacting entities and functional components such as
  - Identity Management
  - Authentication
  - Authorization
  - Trust
  - Reputation

IoT deployment and operational view

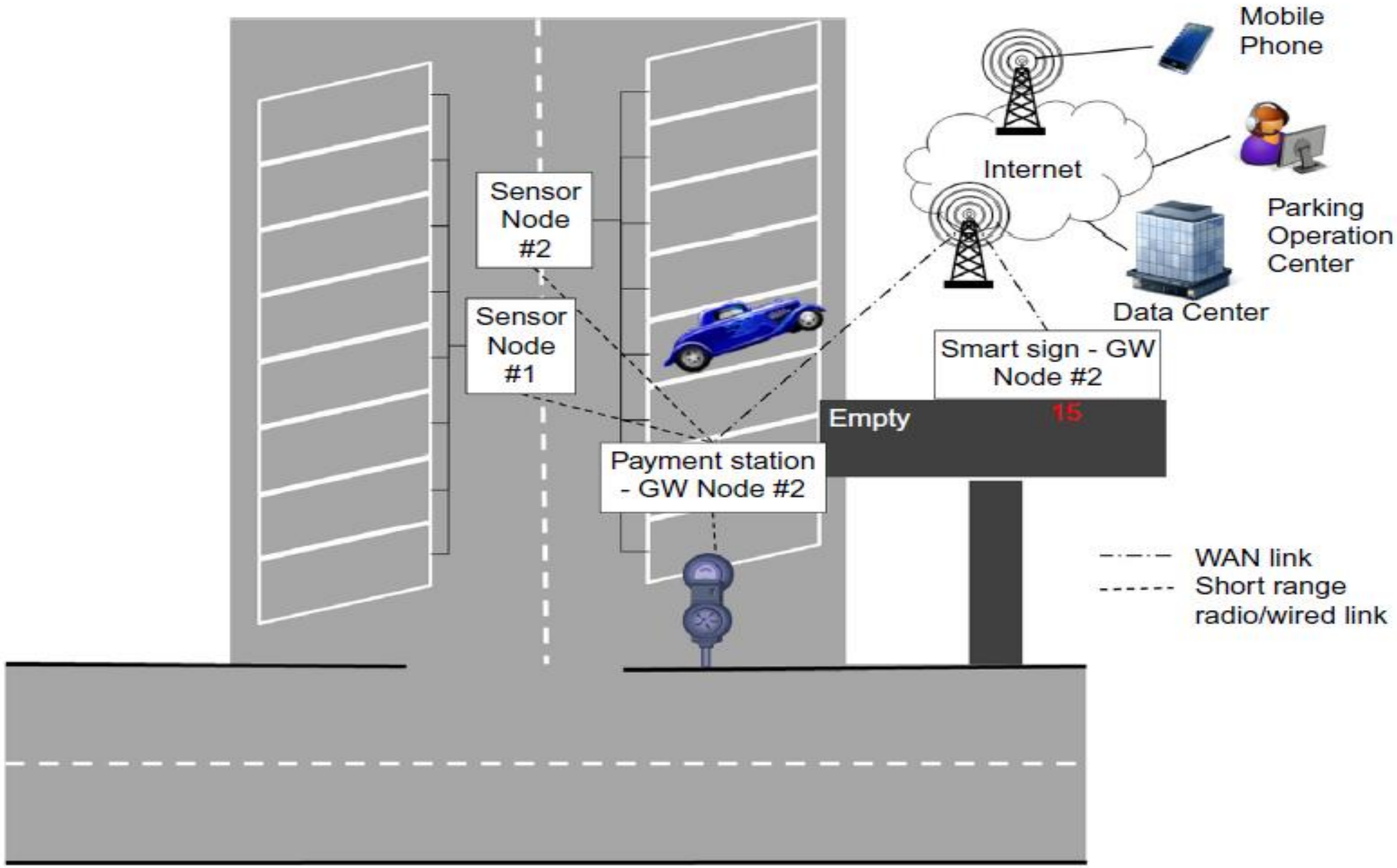


FIGURE 8.8

Parking Lot Deployment and Operational View, Devices.

- In the Parking Lot example, two sensor nodes, each connected to metal/car presence sensors, communicate with a payment station.
- This station serves as both a user interface for payment and a gateway connecting the sensors and payment devices to the Internet via WAN.
- An occupancy sign, acting as a communication gateway for displaying parking availability, connects through WAN to the payment station, avoiding direct wiring due to cost or vandalism risks.
- The system connects to a data center where parking management software is hosted on a PaaS configuration.
- The management system supports mobile applications for users and operational applications for the parking center, which manages multiple lots with similar infrastructure.

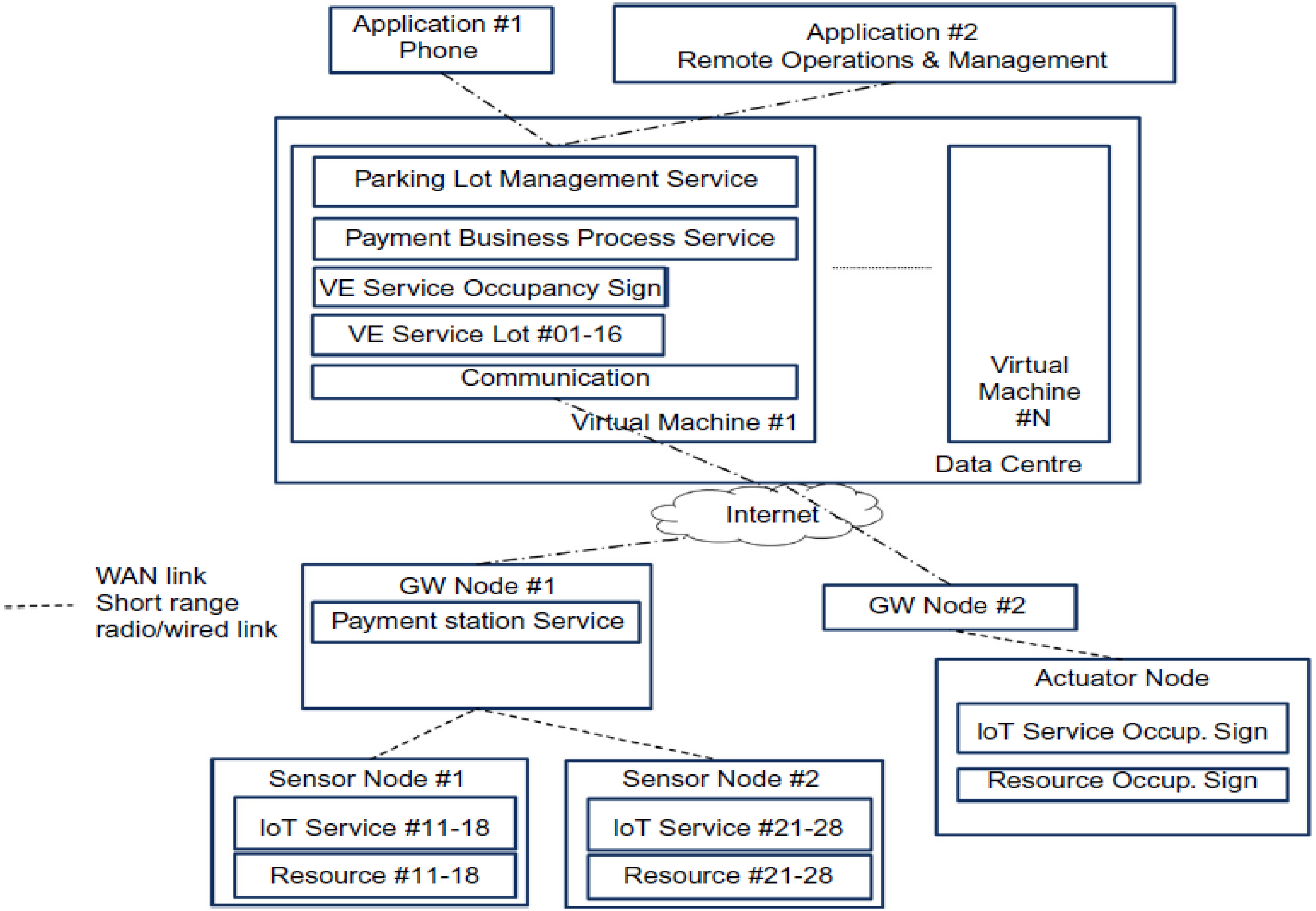


FIGURE 8.9

Parking Lot Deployment & Operational View, Resources, Services, Virtual Entities, Users.

In the IoT parking lot system:

- **Sensor Nodes**: Two nodes host sensors for parking spots and IoT services. Sensor Node #1 manages sensors for parking spots #01–#08, while Sensor Node #2 manages sensors for spots #09–#16.
- **Gateway Devices**: One gateway device connects the sensor nodes and hosts the payment service, while another gateway controls the occupancy sign actuator.
- **Management System**: Deployed on a virtual machine in a data center, it includes communication capabilities, Virtual Entity services for parking spots and the occupancy sign, a payment business process, and access control for parking data.
- **Virtual Entities**: Represent parking spot states and occupancy sign data. They map sensor node identifiers to parking spot identifiers and manage updates to the occupancy display.
- **Additional Services**: Include historical data for planning and machine learning. The IoT Domain Model maps these components to their physical counterparts, with sensors and actuators close to the entities they monitor or control.

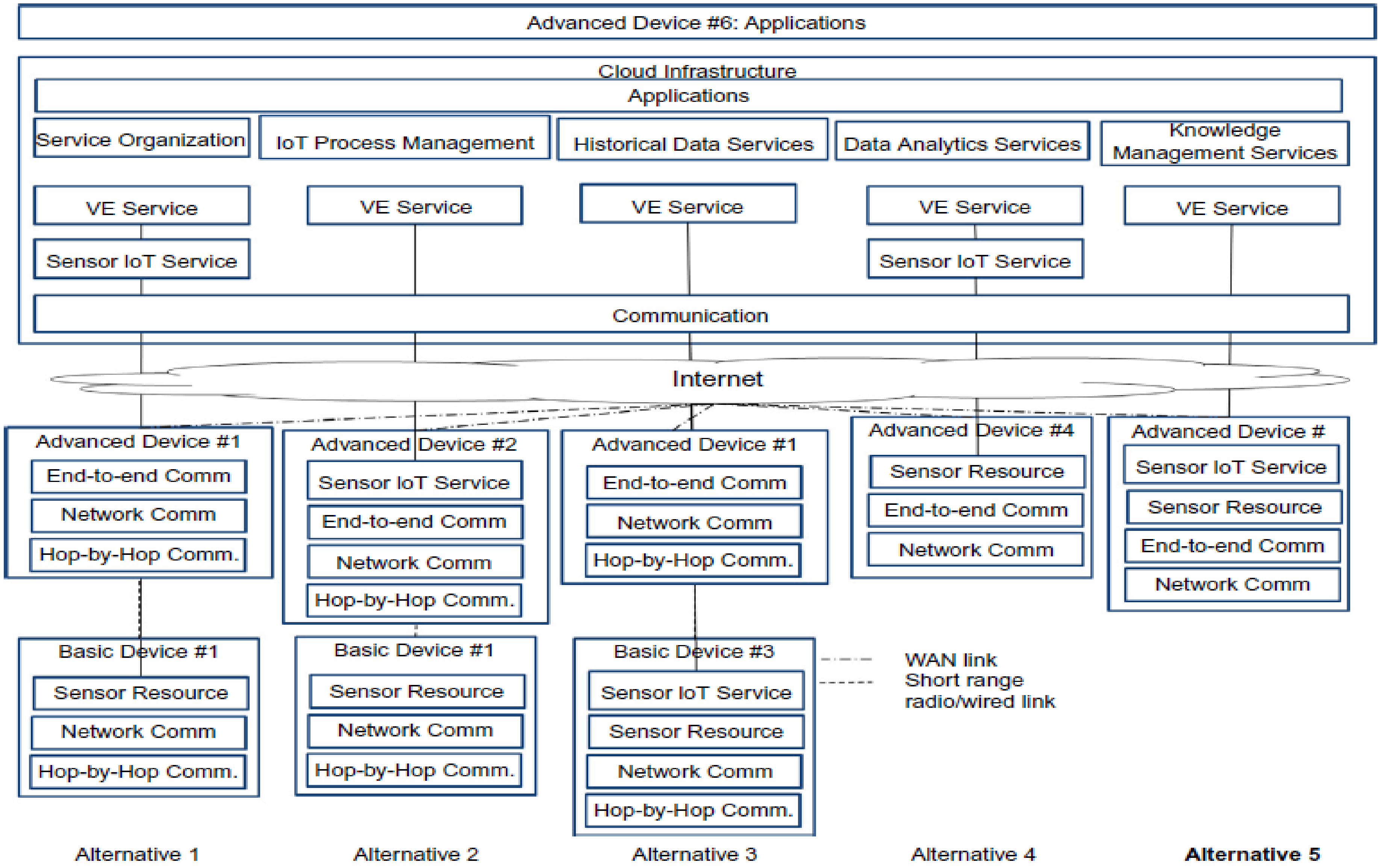


FIGURE 8.10

Mapping IoT Domain Model concepts to Deployment View.

Figure 8.10 illustrates various deployment scenarios for IoT systems connecting to cloud infrastructure:

1. **Alternative 1**: Basic Device #1, which only supports simple sensor functions and short-range communication, requires Advanced Device #1 for protocol adaptation to WAN technology. The cloud hosts the Virtual Entity for this setup.
2. **Alternative 2**: Advanced Device #2 can host both the Sensor IoT Service and communicate directly with Sensor Resource on Basic Device #1. The cloud only hosts the Virtual Entity Service for the Sensor IoT Service.
3. **Alternative 3**: Basic Device #3 provides both the Sensor Resource and IoT Service but still needs Advanced Device #6 for secure communication with Users and cloud services.
4. **Alternative 4**: Advanced Devices with WAN interfaces host only the Sensor Resource. The Virtual Entity Service is still in the cloud.
5. **Alternative 5**: Advanced Devices with WAN interfaces host both the Sensor Resource and IoT Service, while the Virtual Entity Service remains in the cloud.

Thank You