Internet Of Things

Christian Rossi

Academic Year 2024-2025

Abstract

The course provides an overview of the four main components of IoT systems: sensors, communication technologies, management platforms, and data processing and storage platforms for sensor data. In the first part, the course covers the characteristics of the hardware components of sensor nodes (microcontrollers/microprocessors, memory, sensors, and communication devices). It then delves into communication technologies used in IoT systems, distinguishing between short-range solutions (ZigBee, 6LoWPAN) and long-range solutions (LoRaWAN, NB-IoT). Finally, the course focuses on application-level protocols for IoT systems (COAP, MQTT) and the analysis of IoT management platforms. The course includes hands-on development activities and is delivered through flipped classroom and/or blended learning formats.

Contents

1 Introduction **1**
- 1.1 Internet . . . 1
  - 1.1.1 Internet of Things . . . 1
  - 1.1.2 Industrial IoT . . . 2
  - 1.1.3 Architecture . . . 2
- 1.2 Hardware . . . 2
  - 1.2.1 Processor . . . 3
  - 1.2.2 Sensor . . . 4
  - 1.2.3 Power consumption . . . 5
- 1.3 Communication . . . 7
  - 1.3.1 Key Performance Indicators . . . 7
  - 1.3.2 Communication technology . . . 7
  - 1.3.3 Communication protocol . . . 8

2 Application layer **9**
- 2.1 Introduction . . . 9
  - 2.1.1 Taxonomy . . . 9
- 2.2 Hyper Text Transfer Protocol . . . 9
  - 2.2.1 Communication . . . 9
- 2.3 Constrained Application Protocol . . . 10
  - 2.3.1 Packets . . . 11
  - 2.3.2 Observation . . . 12
  - 2.3.3 Block transfer . . . 12
  - 2.3.4 Resource discovery . . . 13
- 2.4 Message Queuing Telemetry Transport . . . 13
  - 2.4.1 Publisher . . . 13
  - 2.4.2 Subscriber . . . 14
  - 2.4.3 Session . . . 14
  - 2.4.4 Messages . . . 14
  - 2.4.5 MQTT for Sensor Networks . . . 15

3 Physical layer **16**
- 3.1 Introduction . . . 16
  - 3.1.1 Controller Area Network bus . . . 16
  - 3.1.2 Ethernet . . . 16
  - 3.1.3 Wireless . . . 17
- 3.2 Low Power Wide Area Network . . . 17

3.2.1 End devices 18
3.2.2 Messages 19
3.2.3 Uplink 19
3.3 NarrowBand Internet of Things 19
3.3.1 Coverage 20
3.3.2 Features 20

4 Short range communication technologies **21**
4.1 Introduction 21
4.2 ZigBee 22
4.2.1 Physical layer 23
4.2.2 Network layer 24
4.2.3 Application profiles 26
4.3 THREAD 26
4.3.1 IPv6 over Low-power Wireless Personal Area Networks 28
4.4 Bluetooth 29
4.4.1 Physical layer 30
4.4.2 Profiles 31
4.4.3 Bluetooth Low Energy 32
4.5 Radio Frequency Identification 33
4.5.1 History 33
4.5.2 Architecture 33
4.5.3 Physical communication 34
4.5.4 Tag arbitration 34

CHAPTER 1

Introduction

1.1 Internet

Definition (*Internet*)**.** The Internet is a global network that connects various types of networks, enabling communication and data exchange.

Traditionally, the internet was primarily used for fixed, stationary clients accessing well-defined services. However, modern internet usage has shifted significantly with the rise of mobile clients. These mobile devices, often equipped with sensing and actuating capabilities, are no longer just consumers of information and services.

Technological advancements Several breakthroughs have paved the way for the rapid growth of the Internet of Things (IoT). The miniaturization of hardware has enabled the development of compact yet powerful smart devices. At the same time, improvements in energy solutions have enhanced the efficiency and autonomy of these devices. Increased mobility has further expanded the reach and functionality of IoT applications.

In parallel, communication protocols have evolved to support low-power wireless technologies, ensuring efficient and reliable connectivity. The widespread adoption of cloud computing has also played a crucial role, providing scalable architectures and vast processing power. Additionally, the rise of Artificial Intelligence has unlocked new possibilities for intelligent data analysis, automation, and decision-making within IoT ecosystems.

1.1.1 Internet of Things

Definition (*Internet of Things*)**.** Internet of Things (IoT) is a worldwide network of uniquely addressable interconnected objects, based on standard communication.

The IoT is based on:

- *Smart objects*: devices embedded with sensors, actuators, and connectivity.
- *Data*: continuous collection and processing of information.
- *Pervasiveness*: seamless integration into everyday life.

- *Seamless communication*: reliable and efficient interaction between devices, networks, and services.

The IoT primarily consists of connected low-cost endpoints, such as consumer devices and everyday smart objects, which focus on accessibility and widespread adoption.

1.1.2 Industrial IoT

Definition (*Industrial IoT*)**.** The IIoT refers to a network of interconnected sensors, instruments, and devices integrated with industrial computing applications, including manufacturing, energy management, and automation.

The IIoT consists of connected industrial assets that are typically medium to high-cost. These devices are more expensive but also more responsive.

Cybersecurity is a central concern in the IIoT, where even minor disruptions can have severe consequences. Unlike consumer IoT, IIoT systems must operate with continuous availability, robustness, and resiliency, ensuring that industrial processes remain uninterrupted. IIoT environments often coexist with a significant amount of legacy operational technologies. These legacy systems, designed for reliability rather than cybersecurity, introduce additional challenges in securing industrial networks.

While usability and user experience are critical in consumer IoT, they are not primary concerns in IIoT. Instead, the focus is on system integrity, fault tolerance, and maintaining operational continuity in complex industrial ecosystems.

1.1.3 Architecture

The IoT endpoints require strong security and reliability. These devices are not just about connectivity; they depend on a combination of smart objects, reliable connectivity, data collection, and advanced analytics to function properly. The security of IoT endpoints is critical. Ensuring reliability ensures that these devices can perform their tasks without interruption.

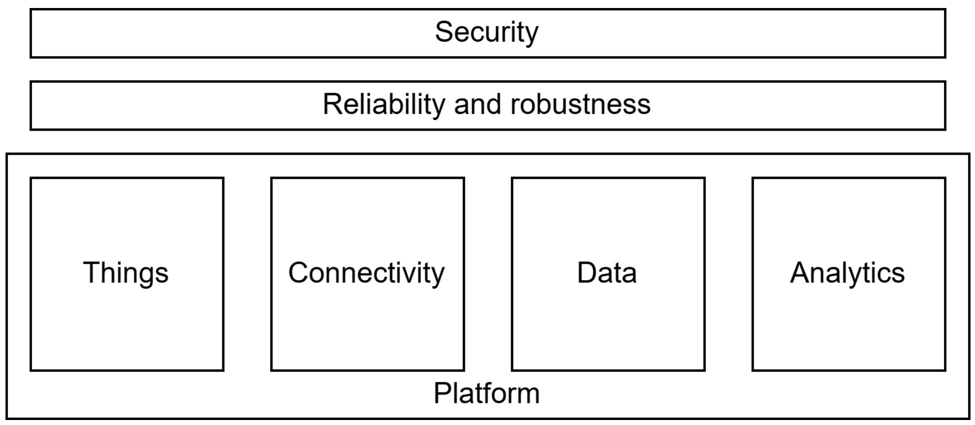

Figure 1.1: IoT architecture

1.2 Hardware

Definition (*Sensor node*)**.** A sensor node (or mote) is a device with capable of sensing external phenomena and process, store and communicate the data collected by the sensors.

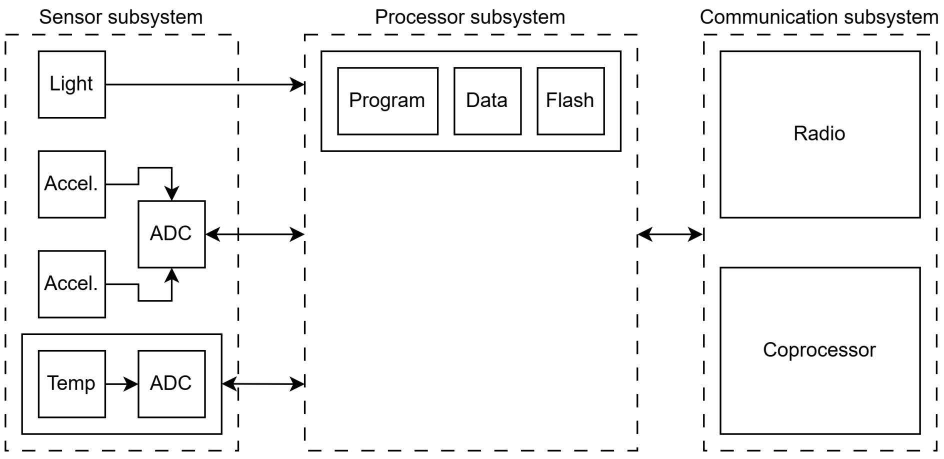

Figure 1.2: Sensor node architecture

Definition (*Actuator*)**.** An actuator is a device capable of acting on processes in order to modify conditions based on the input data.

1.2.1 Processor

The processor subsystem of a sensor node is often designed based on the SHARC architecture.

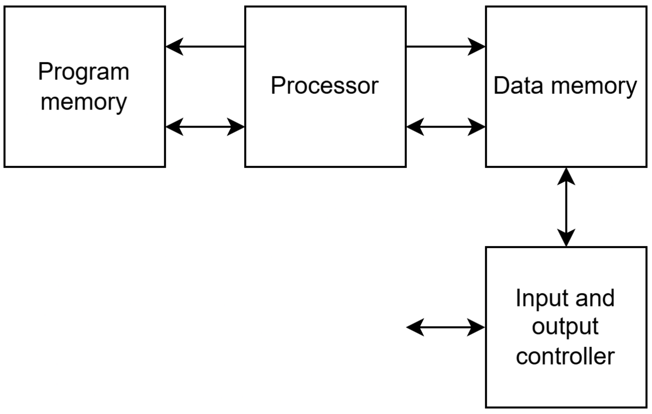

Figure 1.3: Processor SHARC architecture

A microcontroller is generally used as the processor in sensor nodes.

Definition (*Microcontroller*)**.** A microcontroller is a single integrated circuit designed for a specific application.

A microcontroller is usually compose by a Central Processing Unit and a clock generator. It is usually equipped with RAM, flash memory and an EEPROM. It is connected with a serial BUS, I/O interfaces and analogic and digital converters. While microcontrollers are flexible and low-cost, they can compromise speed in certain use cases.

Definition (*Digital Signal Processor*)**.** A Digital Signal Processor (DSP) is a specialized microprocessor optimized for processing discrete signals using digital filters.

DSPs excel at performing complex mathematical operations with extremely high efficiency. While they are well-suited for data-intensive operations, they are less flexible than microcontrollers.

Definition (*Application Specific Integrated Circuit*)**.** An Application Specific Integrated Circuit (ASIC) is a custom integrated circuit tailored for a specific application.

ASICs offer high speed and can be tailored for specific tasks, but they come with high development costs and limited flexibility once designed.

Definition (*Field Programmable Gate Array*)**.** A Field Programmable Gate Array (FPGA) has a high level architecture that allows for some degree of reconfigurability after manufacturing.

FPGAs offer high-speed performance, supporting parallel programming, and moderate reconfigurability. However, they are more complex and costly than microcontrollers or DSPs.

1.2.2 Sensor

Sensors samples the data from the real worlds and traduce them in digital format

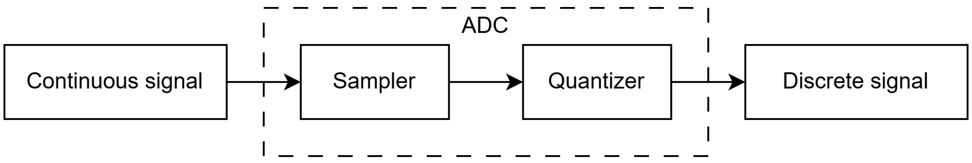

Figure 1.4: Analog to digital converter

The Nyquist analog to digital converter involves reading a time-continuous signal at specific points in time. The sampling rate, or bandwidth, is the inverse of the sampling interval:

$$f_s = \frac{1}{T}$$

The key idea is that if the sampling frequency is properly set, the original signal can be losslessly reconstructed from its samples.

Theorem 1.2.1 (Nyquist theorem)**.** *Given the signal bandwidth B, we have that the sampling frequency must be chosen as:*

$$f_s = 2B$$

Quantization is used to approximate the input voltage V_{in} in a digital codeword. An ideal quantizer maps input to output with the smallest variation in the input causing a change in the codeword.

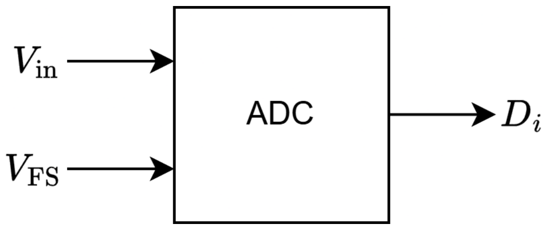

Figure 1.5: Quantization

The resolution refers to the smallest input variation that causes a change in the codeword:

$$\text{LBS} = \frac{V_{\text{FS}}}{2^n}$$

Here, V_{FS} is the full-scale voltage and n is the number of bits of resolution. The quantization error occurs when the output voltage either overestimates or underestimates the input voltage. This error decreases as the resolution increases.

1.2.3 Power consumption

The absence of cables in wireless sensors and actuators means no wired power or connectivity. A sensor node typically operates with a limited power source, and its lifetime directly depends on the battery lifetime. The goal is to maximize the energy provided while minimizing the cost, volume, weight, and recharge requirements. There are two main types of batteries used:

- Primary batteries, which are not rechargeable.
- Secondary batteries, which are rechargeable, but only make sense when paired with some form of energy harvesting.

Power cycle The power cycle of an IoT device consists of sleep and active states (wake-up and work). During the sleep state, power consumption is minimal, with only essential components running.

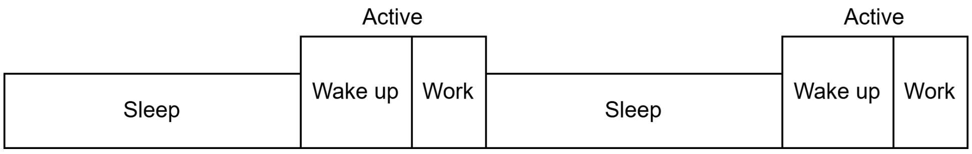

Figure 1.6: IoT device life cycle

The average power consumption is then defined as:

$$P_{\text{avg}} = f_{\text{sleep}}P_{\text{sleep}} + f_{\text{wake up}}P_{\text{wake up}} + f_{\text{work}}P_{\text{work}}$$

Here, f_{sleep}, f_{wakeup}, and f_{work} are the fractions of time spent in sleep, wake-up, and work states, respectively. Then, the lifetime of the device is given by:

$$\text{lifetime} = \frac{\text{energy stored}}{P_{\text{avg}} - P_{\text{gen}}}$$

Here, P_{gen} is the power generated.

1.2.3.1 Data transmission

When data needs to be sent, the device first wakes up and then performs the actual transmission. The total energy consumption for this process is given by:

$$E_{\text{tx}} = P_{\text{tx}}(T_{\text{wu}} + T_{\text{tx}}) + P_0 T_{tx}$$

Here, P_{tx} is the power consumed by the transmitter, P_0 is the output power of the transmitter, T_{tx} is the time taken to transmit a packet, and T_{wu} is the wake-up time.

1.2.3.2 Data reception

When the device needs to receive data, it first wakes up and then performs the reception. The total energy consumption in this case is:

$$E_{\text{tx}} = P_{\text{rx}}(T_{\text{uw}} + T_{\text{rx}})$$

Here, P_{rx} is the power consumed by the receiver, T_{rx} is the time taken to receive a packet, and T_{wu} is the wake-up time.

Sensitivity Each receiver is characterized by a sensitivity parameter, which is the minimum input signal power required for the receiver to demodulate the data correctly. Knowing this sensitivity, the required emitted power at the transmitter can be determined by inverting the propagation law of the communication channel.

1.2.3.3 Emitted power

The emitted power is often a tunable parameter, and it is generally considered good practice to set it to the lowest value that still allows for reliable reception. The quality of the reception process is typically measured using metrics like:

- *Bit Error Rate*: the fraction of bits that are incorrectly received.
- *Packet Error Rate*: the fraction of packets that are not received correctly.

The relationship between BER and PER, for a packet of length l with independent errors, is given by:

$$\text{PER} = 1 - (1 - \text{BER})^l$$

Both BER and PER are influenced by the level of noise in the transmission and reception channels, which in turn is determined by the transmitted and received power. The Signal-to-Interference-plus-Noise Ratio is a key factor in determining this quality, and is calculated as:

$$\text{SINR} = 10 \log \left(\frac{P_{\text{recv}}}{N_0 + \sum_{i=1}^{k} I_i} \right)$$

Here, N_0 is the thermal noise (KTB), P_{recv} is the received power, and I_i are the interference contributions from other signals. Given the SINR and the specific modulation of the channel, BER can be computed.

1.2.3.4 Processing power

The power dissipation of the CPU is due to several factors, including:

$$P_{\text{p}} = P_{\text{dyn}} + P_{\text{sc}} + P_{\text{leak}}$$

Here, $P_{\text{dyn}} = CfV^2$ is the power consumed by the work done, P_{sc} is the power lost due to short circuits, and P_{leak} is the power lost due to leakage. Local data processing is crucial in minimizing power consumption, especially in multi-hop networks.

1.2.3.5 Sensor power

The power consumption due to sensing is highly dependent on the type of sensor used. A rough model for the power consumption of an analog to digital converter can be expressed as:

$$P_s \approx f_s 2^n$$

Here, f_s is the sampling frequency, and n is the resolution of the analog to digital converter in bits.

1.3 Communication

In IoT, various devices and systems need to exchange information seamlessly. Industrial environments present unique challenges for communication networks such as timeliness, determinism, and reliability.

Definition (*Cyclic traffic*)**.** Cyclic traffic involves periodic transmissions, ensuring continuous process monitoring.

Definition (*Acyclic traffic*)**.** Acyclic traffic occurs in response to unpredictable events.

Definition (*Multimedia traffic*)**.** Multimedia traffic includes images, video streams, and other data-intensive transmissions.

Definition (*Backhand traffic*)**.** Backhand traffic aggregates data flows from multiple sources, consolidating information for centralized processing and analysis.

1.3.1 Key Performance Indicators

Definition (*Throughput*)**.** Throughput refers to the rate at which data can be transmitted over a network link, typically measured in bits per second or bytes per second.

The actual throughput depends on the type of link being used, as different technologies offer varying transmission capacities.

Definition (*Delivery time*)**.** Delivery time represents the total time required to transfer a service data unit from the source to the destination.

Measured in seconds, it is influenced by multiple factors, including transmission time, propagation time, and the execution time of network protocols.

Definition (*Minimum cycle time*)**.** Minimum Cycle Time defines the shortest duration needed to complete one full cycle within a control loop.

This metric is crucial in real-time systems, where consistent and predictable execution times are essential for maintaining stability and efficiency.

Definition (*Jitter*)**.** Jitter measures the precision and reliability of periodic operations, particularly in time-sensitive applications.

Variability in timing can lead to performance issues, making jitter a critical factor in industrial and communication networks that require consistent and predictable timing.

1.3.2 Communication technology

Defining a communication technology involves several key aspects, starting with the physical infrastructure. IoT connectivity integrates multiple heterogeneous technologies to ensure seamless communication across diverse environments. The classification of communication technologies often depends on their range.

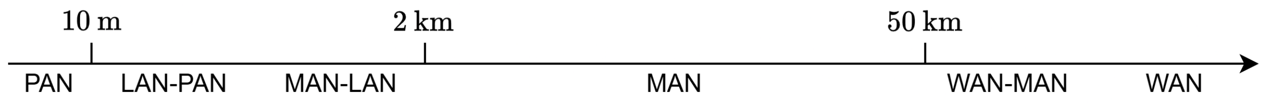

Figure 1.7: Connection range

Network topology also can vary a lot:

- *Ring topology*: structured flow of data, while linear chains connect devices sequentially.
- *Tree topology*: hierarchical communication, whereas star topology centralizes connections around a single node.
- *Mesh topology*: robust, decentralized connectivity, enhancing reliability and redundancy in complex systems.

1.3.3 Communication protocol

Communication protocols define the set of rules governing the exchange of information between two entities. In IoT communication networks, protocols facilitate packet-based data exchange. Most communication protocols follow a layered architecture. This structured approach enhances interoperability, modularity, and scalability in network communication. The layers are:

- *Application layer*: handles messages exchanged between applications, enabling services such as HTTP, MQTT, and CoAP.
- *Transport layer*: manages end-to-end communication through segmentation and reassembly, with protocols like TCP and UDP.
- *Network layer*: routes packets across different networks, using IP-based addressing and routing mechanisms.
- *Data link layer*: organizes data into frames, ensuring reliable point-to-point or point-to-multi point transmission.
- *Physical layer*: deals with the actual transmission of raw bits over physical media such as wired or wireless connections.

CHAPTER 2

Application layer

2.1 Introduction

Definition (*Application*)**.** An application refers to a program running on a host that can communicate with another program over a network.

However, communication technologies are highly fragmented. Different protocols and languages coexist, often creating compatibility challenges. Many field-level and factory-level technologies do not natively support Internet-based communication. A practical solution to this fragmentation is the adoption of service oriented architectures.

2.1.1 Taxonomy

The application layer protocols can be classified as:

- *Client and server* (HTTP and CoAP): a client requests data from a server.
- *Publish and subscribe* (MQTT): devices publish data to a central broker, which then distributes updates to subscribed clients.

2.2 Hyper Text Transfer Protocol

Web pages are composed of multiple resources which are uniquely identified by a Uniform Resource Locator (URL) or a Uniform Resource Identifier (URI). When a client requests a resource using a URL, the request is resolved through Domain Name Resolution (DNS), followed by the retrieval of the requested data from the corresponding web server.

On the web, resources are managed by servers, identified through URIs, and accessed synchronously by clients using a request and response paradigm. This model aligns with the REpresentational State Transfer (REST) architectural style.

2.2.1 Communication

HTTP follows a client-server architecture, where communication occurs as follows:

- The client sends an HTTP request for a specific resource, identified by its URI.

- The server processes the request and responds with the requested resource or an appropriate status message.

HTTP is a stateless protocol, meaning each request is independent, and the server does not retain memory of previous interactions.

Requests HTTP requests are ASCII-encoded and follow a format that includes various fields.

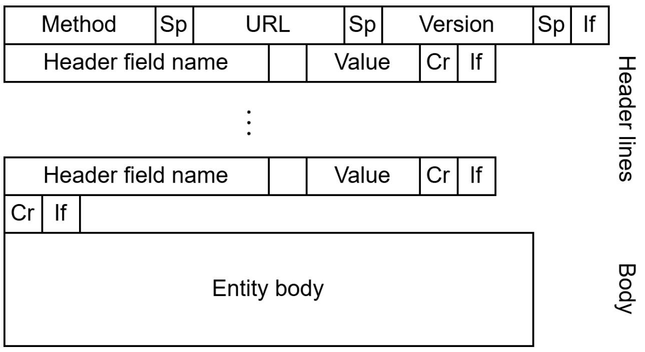

Figure 2.1: HTTP request

Responses HTTP responses follow a similar format, including a status code that indicates the outcome of the request:

- *Informational* (1xx): request received and processing continues.
- *Success* (2xx): the request was successfully processed.
- *Redirection* (3xx): further action is needed to complete the request.
- *Client-side error* (4xx): the request contains incorrect syntax or cannot be fulfilled.
- *Server-side error* (5xx): the server encountered an issue while processing the request.

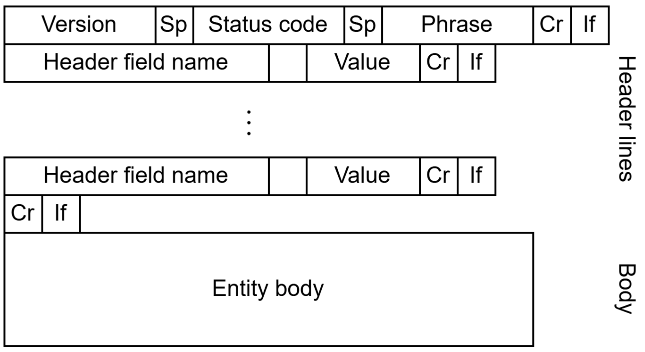

Figure 2.2: HTTP response

2.3 Constrained Application Protocol

Constrained Application Protocol (CoAP) is designed to enable web-based services in constrained wireless networks. These constraints include: low-power devices, limited memory and processing capabilities, and energy-efficient, low-bandwidth networks. To overcome these challenges, CoAP redesigns web-based communication to suit constrained environments while maintaining compatibility with standard web technologies.

CoAP is an embedded web transfer protocol optimized for lightweight, efficient communication.

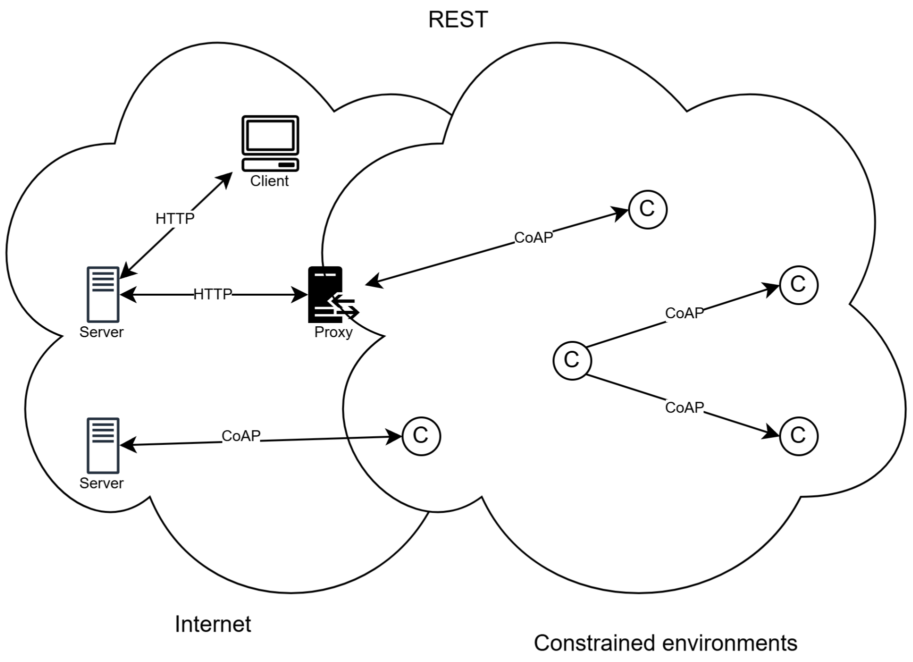

Figure 2.3: CoAP architecture

CoAP plays a critical role in IoT ecosystems, enabling lightweight, low-latency communication between resource-constrained devices while maintaining compatibility with traditional web technologies. Its UDP-based nature makes it well-suited for low-power wireless networks.

Proxy CoAP proxies act as intermediaries between clients and servers, forwarding requests and responses. They help with load balancing, security, and NAT traversal, improving network efficiency and scalability.

Caching Caching: CoAP supports caching responses to reduce repeated network requests. Clients and proxies can store responses and reuse them if the resource hasn't changed, saving bandwidth and improving speed.

2.3.1 Packets

CoAP primarily relies on UDP for lightweight communication, making it ideal for constrained networks. Message exchange occurs between endpoints with a compact 4-byte header, ensuring minimal overhead. Each message contains a 16-bit Message ID, enabling mechanisms for both reliable and unreliable transmission:

- *Confirmable messages*: require acknowledgment (`ACK`) or may trigger a reset message (`RST`) if lost. A stop-and-wait retransmission strategy with exponential backoff ensures reliability.

- *Non-confirmable messages*: sent without requiring acknowledgment, suitable for non-critical or frequent updates.
- *Duplicate detection*: both confirmable and non-confirmable messages use message IDs to identify and discard duplicates.

CoAP follows a RESTful architecture, embedding request and response semantics directly into its messages.

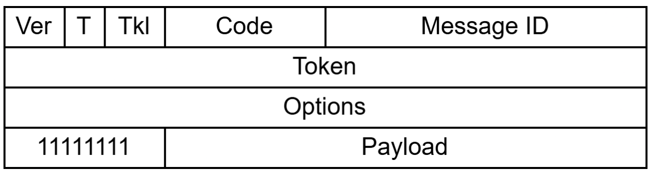

Figure 2.4: CoAP message

The CoAP 4-byte message header consists of several key fields such as version, message type, token length, code, and message ID.

Loss CoAP employs a stop-and-wait retransmission mechanism to handle packet loss in unreliable networks. This approach ensures that lost messages are resent while maintaining low overhead. When sending a confirmable message, the sender expects an acknowledgment or a reset message. If no response is received within a timeout period, the request is retransmitted. If no response is received after the initial timeout, and the transmission counter is less than a threshold the transmission counter is increased, the timeout value is doubled (exponential backoff), and the message is retransmitted. This process continues until one of the following conditions is met:

- An `ACK` is received, confirming successful delivery.
- An `RST` message is received, indicating rejection.
- The transmission counter exceeds `MAX_RETRANSMIT` (4 retransmissions).
- The total attempt duration surpasses `MAX_TRANSMIT_WAIT` (93 seconds), at which point the transmission is aborted.

2.3.2 Observation

In Wireless Sensor Network data is often periodic or triggered by specific events. CoAP introduces the concept of observation to address this. Observation allows clients to subscribe to resources and receive updates automatically whenever the resource's state changes.

CoAP clients use the `observe` option in their requests to subscribe to a resource. When the resource state changes, the server sends an update to the client. The client is notified asynchronously, receiving updates as they occur without needing to send explicit requests.

2.3.3 Block transfer

In networks using IPv6, large payloads are often fragmented at the lower layers. CoAP uses block transfer at the application layer to handle large messages by splitting them into smaller blocks, avoiding fragmentation at lower layers.

2.3.4 Resource discovery

CoAP supports resource discovery, allowing clients to discover and interact with available resources on CoAP servers. The goal is to enable clients to discover the links hosted by a CoAP server. The server returns resource details in a link-header format, which includes:

- *URL*: the resource location.
- *Relation*: describes the relationship between resources.
- *Type*: specifies the resource type.
- *Content type*: indicates the format of the resource.
- *Interface*: describes the access protocol.

2.4 Message Queuing Telemetry Transport

Message Queuing Telemetry Transport (MQTT) is a client-server, publish and subscribe messaging protocol designed for lightweight and efficient communication. In MQTT, clients do not communicate directly with each other. Instead, they publish messages to topics, and other clients subscribe to those topics. A single client can publish a message, and all clients subscribed to that topic will receive the message. Unlike CoAP's pull-based request and response model, MQTT uses a push model where updates are automatically sent to clients when new information is available.

Connection In MQTT, each client establishes a single connection (which supports push capabilities) to the MQTT broker for communication. MQTT can also work even through firewalls or NAT devices, thanks to its use of TCP and well-defined protocols. When a client connects to the broker, it sends a `CONNECT` message with several fields. When the broker responds to the connect message, it sends a `CONNACK` message which indicates wether the client's session was retained and the result of the connection attempt.

2.4.1 Publisher

In MQTT, the `PUBLISH` message is used by the publisher to send data to the broker. The Quality of Service (QoS) of a publisher can be:

- *QoS 0*: the message is delivered at most once, with no guarantee of delivery.
- *QoS 1*: the message is guaranteed to be delivered, but it may be delivered multiple times. The client stores the message and retransmits it until the broker acknowledges receipt. Once the broker receives the publish message, it sends a `PUBACK` message back to the client to acknowledge receipt.
- *QoS 2*: ensures that the message is delivered exactly once, and involves a 4-step handshake:
  1. *Publish reception* (broker): the MQTT broker processes the packet and sends a `PUBREC` message back.

2. *Reception* (client): upon receiving the PUBREC message, the client discards the original packet and sends a PUBREL message to the broker.
3. *Acknowledgment* (broker): the broker clears any current state and sends a PUBCOMP message to confirm the delivery of the message.
4. *Completion* (broker): the client receives the PUBCOMP message.

2.4.2 Subscriber

In MQTT, the SUBSCRIBE message is used by the client to subscribe to one or more topics. Once the broker receives the SUBSCRIBE message, it responds with a SUBACK message.

To unsubscribe from topics, the client sends an UNSUBSCRIBE message. When the broker processes the UNSUBSCRIBE request, it sends an UNSUBACK message.

2.4.3 Session

In MQTT, the session management behavior depends on the type of session:

- *Non-persistent session*: when a client disconnects, all client-related information is cleared from the broker. This means that when the client reconnects, it needs to re-subscribe to topics and will not receive messages sent during its disconnection.

- *Persistent session*: the client and the broker maintain the session state even if the client disconnects. This ensures that the client can resume its communication without losing its subscription information or missing messages. With a persistent session, even if the client is offline, the broker will hold the state and deliver any relevant messages when the client reconnects.

2.4.4 Messages

In MQTT, publishing and subscribing are asynchronous operations. Retained messages are PUBLISH messages where the `retained` flag is set to one. These messages are stored by the broker, and whenever a new client subscribes to the topic, the broker immediately sends the last retained message on that topic to the subscribing client. This ensures that subscribers receive the latest available message, even if no new message has been published.

Last will The Last Will and Testament (LWT) message in MQTT allows a client to notify other clients about an unexpected disconnect. When a client connects to the broker, it can specify a LWT message. The broker stores the LWT message and only sends it if it detects that the client has disconnected unexpectedly. When the client experiences a hard disconnection, the broker sends the stored LWT message to all subscribed clients on the specified topic. If the client disconnects gracefully, the broker will discard the LWT message.

Keep alive It is the client's responsibility to maintain an active MQTT connection. If no other interactions occur within the specified keep-alive interval, the client sends a ping request to the broker. The broker, upon receiving the ping, responds with a ping response to confirm the connection is still active. This mechanism ensures the connection remains stable and prevents unnecessary disconnections due to inactivity.

2.4.5 MQTT for Sensor Networks

MQTT for Sensor Networks (MQTT-SN) is a variation of the standard MQTT protocol designed to better suit the constraints of sensor networks and environments with limited resources. Here are the main differences compared to the standard MQTT protocol:

- Extended architecture with gateways and forwarders.
- New gateway discovery procedures and messages.
- Some messages are more.
- Extended keep alive procedures to support sleeping clients.

CHAPTER 3

Physical layer

3.1 Introduction

Fieldbus technology is designed to connect control systems with field devices, enabling frequent exchange of short messages. A hallmark of fieldbus systems is their demand for low latency and deterministic communication, ensuring predictable and timely message delivery.

One of the central challenges in fieldbus networks lies in managing access to a shared communication medium among multiple devices. Two main strategies address this:

1. *Scheduled access*: devices transmit in a predefined order, coordinated either through centralized scheduling or polling mechanisms. This guarantees collision-free communication, predictable delays, and consistent throughput. However, it introduces system complexity due to the need for synchronization or a central authority.

2. *Random access*: Devices independently decide when to transmit. While easier to implement and more flexible, this approach only offers statistical performance guarantees and can suffer under heavy traffic due to increased collision rates.

3.1.1 Controller Area Network bus

The Controller Area Network (CAN) bus uses a shared physical medium where all connected devices receive all transmitted messages. It employs Carrier Sense Multiple Access (CSMA): devices sense the bus before transmitting.

To resolve contention, messages are assigned priorities based on their identifier fields. During transmission, each device monitors the bus. If it detects a higher-priority message while transmitting, it stops and defers to the higher-priority sender.

3.1.2 Ethernet

Originally introduced in 1976, Ethernet began as a shared-bus network using coaxial cables. Through the 1990s, it shifted to star topologies using hubs and twisted-pair cabling. Since the 2000s, Ethernet has evolved to fully switched, full-duplex systems capable of high-speed communication.

Ethernet manages traffic using techniques like multiplexing, frame filtering, and multiple access protocols. A typical Ethernet frame includes a synchronization preamble and a frame delimiter to mark the start of the transmission.

MAC address Each network interface has a unique MAC address, composed of a manufacturer identifier (first 3 bytes) and a device-specific identifier (last 3 bytes). A MAC address of all ones is used for broadcast communication.

Collision detection Legacy Ethernet relied on Carrier Sense Multiple Access with Collision Detection (CSMA/CD). Devices monitored the channel before sending data. If a collision occurred, all senders aborted and waited a random interval before retrying. While effective, this method could introduce delays, especially under load.

Early Ethernet networks used hubs, where all connected devices shared the same bandwidth and collisions were common. In contrast, network switches enable full-duplex, collision-free communication by establishing dedicated links between devices and the switch. As a result, CSMA/CD is no longer necessary in modern switched Ethernet networks.

Industry 4.0 As part of Industry 4.0, Ethernet has been tailored to meet a range of time-sensitive industrial needs. It is classified into three categories based on maximum cycle time:

- *Class A*: non-critical applications with relaxed timing constraints.
- *Class B*: time-sensitive tasks requiring faster responses.
- *Class C*: high-priority, ultra-low-latency communication for critical operations.

3.1.3 Wireless

The expansion of wireless networking has been fueled by the proliferation of smart objects—connected devices ranging from sensors to everyday electronics, especially within the IoT ecosystem. Modern wireless systems support diverse use cases, including: mobile radio networks, cellular IoT operators, low power long range technology, and capillary multi-hop networks.

3.2 Low Power Wide Area Network

Long Range Wide Area Network (LoRaWAN) is a communication protocol tailored for low-power, long-range IoT applications. Unlike traditional cellular networks, LoRaWAN employs an association-less communication model, allowing devices to transmit without establishing persistent links to the network infrastructure. The architecture comprises three primary components: end devices, gateways, and a network server.

End devices are typically deployed in the field to collect and transmit data. Gateways act as intermediaries, bridging the wireless transmission from end devices to the network server, which hosts most of the system intelligence. The network server is responsible for de-duplicating messages, managing acknowledgments, adapting radio parameters, and orchestrating overall communication to ensure reliable and energy-efficient operation.

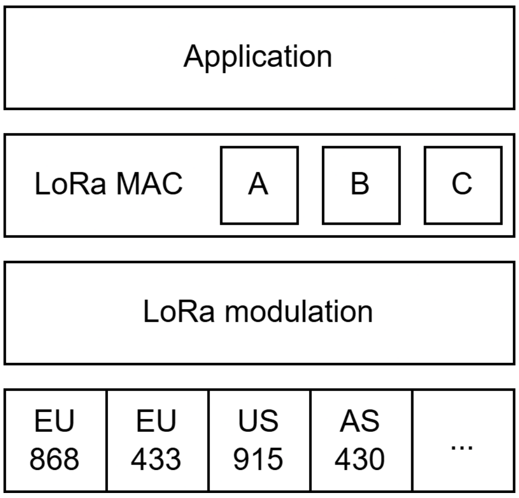

Figure 3.1: LoraWAN protocol stack

LoRaTM employs a proprietary chirp spread spectrum modulation technique. This method modulates information onto chirped signals, enhancing resilience against interference and enabling long-range, low-power communication. The chip rate R_C and bit rate R_b are governed by:

$$R_C = \text{BW} \qquad R_b = \text{SF}\frac{4\text{BW}}{2^{\text{SF}}(4+\text{CR})}$$

Here, BW is the bandwidth, SF is the spreading factor, and CR is the coding rate.

There is an inherent trade-off between data rate and transmission reliability:

- Lower spreading factors yield higher data rates but reduced robustness.
- Higher spreading factors enhance signal sensitivity and reliability, ideal for long-distance or noisy environments.

The minimum required received power for successful decoding is given by:

$$P_{\text{min}} = -174 + 10\log_{10}\text{BW} + \text{NF} + \text{SNR}$$

Here, BW is the reference bandwidth, NF is the noise, and SNR is the signal-to-noise ratio.

3.2.1 End devices

LoRaWAN defines three operational classes for end devices, optimized for different application needs and energy constraints:

- *Class A*: the most energy-efficient mode. Devices initiate uplinks at will and open two short downlink reception windows afterward.
- *Class B*: devices synchronize with periodic beacons and offer scheduled receive windows, enabling more predictable downlink communication.
- *Class C*: devices continuously listen for downlink messages, offering the lowest latency at the cost of higher power consumption.

In the European LoRaWAN specification, the timing of Class A receive windows is fixed. Upon sending an uplink, the end device opens two sequential receive windows. If a valid preamble is detected in the first window, the device remains awake until the message is demodulated. If the message passes address and integrity checks, the second window is skipped, minimizing power usage.

3.2.2 Messages

LoRaWAN messages are organized across several protocol layers to ensure synchronization, integrity, and proper routing.

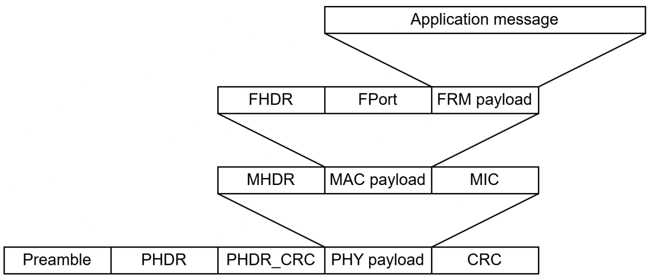

Figure 3.2: LoraWAN messages

3.2.3 Uplink

Uplink transmissions in LoRaWAN follow an un-slotted ALOHA approach. When a confirmed message is transmitted but no acknowledgment (ACK) is received, the message is retransmitted after a randomly chosen delay.

LoraWAN does not rely on explicit channel feedback. Instead, devices infer success or failure based on ACK receipt. Time is modeled as continuous, and packets are transmitted as soon as they are queued. Retransmissions are delayed by a randomized backoff interval.

To model network performance, it is assumed that:

- Traffic is stationary, meaning input and output rates are equal.
- Packet arrivals follow a Poisson distribution with rate λ.
- Each packet transmission lasts T seconds.
- The total channel load is $G = \lambda T$.

The success probability Pr_s, i.e., the probability that no other transmission overlaps within a vulnerable period $2T$, is:

$$\Pr_s = \Pr(N(t-T, t+T) = 0) = e^{-2G}$$

The throughput S, defined as successful transmissions per time unit, becomes:

$$S = \mathrm{G} \Pr_s = Ge^{-2G}$$

This relationship illustrates the classic ALOHA performance curve, with a peak throughput at $G = 0.5$, beyond which performance degrades due to increasing collisions.

3.3 NarrowBand Internet of Things

NarrowBand Internet of Things (NB-IoT) is a cellular technology purpose-built to address the unique requirements of large-scale IoT deployments. It is designed for low-power, low-throughput applications and offers several advantages. Operating within the licensed spectrum, NB-IoT ensures robust, interference-free connectivity, making it especially suitable for industrial, urban, and rural IoT use cases. Moreover, it benefits from seamless integration with existing LTE infrastructure, allowing mobile operators to deploy it cost-effectively.

Power Saving Mode (PSM) allows devices to stay registered with the network without needing to re-initiate connections. This avoids the energy-intensive signaling associated with re-attachments, significantly reducing power consumption. While in PSM, devices are unreachable for mobile-terminated traffic, but remain logically connected, enabling long-term sleep periods and dramatically extending battery life without compromising network accessibility.

The random access procedure governs how a device initiates communication with the network. This mechanism ensures efficient resource allocation, especially important when a large number of devices attempt to connect simultaneously. To maintain reliability even in weak signal conditions, NB-IoT supports extensive message repetition ensuring that critical messages are delivered even in deep indoor or remote environments.

3.3.1 Coverage

NB-IoT offers robust coverage capabilities through Coverage Enhancement (CE) levels, allowing reliable operation across diverse signal environments. Devices in strong signal areas operate at CE Level 0, while those in medium and poor signal areas transition to CE Levels 1 and 2, respectively. These levels determine the number of repetitions for both uplink and downlink transmissions, increasing signal robustness. This adaptability allows NB-IoT to achieve up to 164 dB of maximum coupling loss, significantly extending coverage into basements, tunnels, and remote outdoor locations.

3.3.2 Features

NB-IoT minimizes device and deployment costs by simplifying traditional LTE features. It reduces modem complexity through the use of smaller transport block sizes, single-stream transmissions, and single-antenna support. Additionally, turbo decoding is eliminated from the device (UE) side, as only Turbo Block Coding (TBC) is used for downlink channels.

NB-IoT supports in-band operation by embedding its carriers within existing LTE carriers. This approach enables efficient reuse of spectrum and infrastructure, allowing flexible scaling by adding more NB-IoT carriers as needed. However, care must be taken to manage near-far interference, especially with legacy LTE base stations that don't support NB-IoT. A network-wide upgrade is typically required to ensure optimal coexistence. Although in-band deployment may slightly reduce LTE capacity, strategies like increasing NB-IoT transmit power or dynamic base band resource sharing can mitigate performance degradation.

To further optimize energy efficiency, NB-IoT employs Enhanced Discontinuous Reception (eDRX). This mechanism allows devices to remain dormant for extended intervals significantly reducing the frequency at which they must wake up to monitor paging messages. The eDRX configuration is negotiated between the device and the network, enabling flexible trade-offs between responsiveness and battery life. This feature is particularly beneficial for delay-tolerant applications like smart meters and environmental sensors.

CHAPTER 4

Short range communication technologies

4.1 Introduction

In the world of short-range communication, a variety of technologies and protocols exist, each designed to serve specific needs. While these solutions work well for small-scale applications and localized coverage, they often struggle with interoperability.

Communication technologies today are highly fragmented, with different standards and protocols optimized for different use cases. While some technologies provide effective coverage for small areas, seamless integration across networks remains a challenge. This has led to the need for a new, unified communication stack that can bridge these gaps.

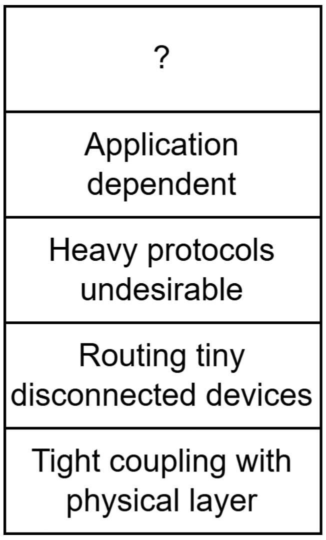
?
Application dependent
Heavy protocols undesirable
Routing tiny disconnected devices
Tight coupling with physical layer

Figure 4.1: New communication stack

Several communication solutions are available, and they can generally be categorized based on:

- Proprietary (WirelessHART) or open standards (WiFi, ZigBee, 6LowPAN, THREAD).
- Application-specific or application-agnostic.
- IP-compliant or not IP-compliant.

ZigBee and 6LowPAN Among the many existing technologies, ZigBee and 6LowPAN are two of the most widely used. While they share similarities at the lower communication layers, their fundamental difference lies in their approach to IP connectivity:

- 6LowPAN extends IP to IoT devices, making it easier to integrate with existing internet infrastructure.
- ZigBee, on the other hand, operates independently of IP, requiring additional gateways for internet connectivity.

4.2 ZigBee

ZigBee is a widely used wireless communication technology designed for low-power, low-data-rate applications.

History In the mid-1990s, IoT solutions were highly fragmented, with numerous proprietary protocols leading to major compatibility and interoperability challenges. This also drove up costs, making large-scale adoption difficult. To address this, IEEE launched Working Group 4 in 2001 to develop a standardized reference technology. This effort resulted in the IEEE 802.15.4 standard, which was officially published in March 2003.

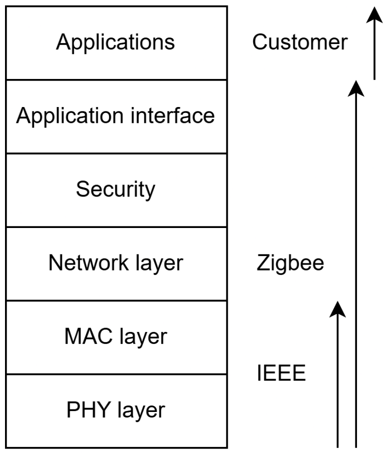

Figure 4.2: ZigBee communication stack

Taxonomy ZigBee networks consist of different types of devices, each playing a specific role:

- *Full function device*: can send beacons to manage network communication, communicate with other FFDs, route network frames, and act as a PAN coordinator. Typically powered by an external power supply.
- *Reduced function device*: cannot route frames or communicate with other RFDs. Communicates only with an FFD. Designed for low-power, battery-operated applications.
- *PAN coordinator*: handles device association and de-association.

ZigBee networks support three different topologies, each suited for specific use cases:

- *Star topology*: all devices connect to a central coordinator. Simple, but limited in scalability. Common in home automation and industrial monitoring.
- *Mesh topology*: devices communicate with multiple neighboring nodes. Highly reliable and scalable, as data can take multiple paths. Used in large-scale IoT deployments.
- *Cluster-tree topology*: an hybrid between star and mesh. Devices are organized in clusters, with a coordinator managing each cluster. Efficient for applications requiring structured hierarchies, such as industrial automation.

4.2.1 Physical layer

The physical layer (PHY) of ZigBee, based on IEEE 802.15.4, is responsible for managing radio communication and ensuring reliable data transmission. Its main functions include: radio transceiver control, energy detection, link quality indicator, clear channel assessment, channel frequency selection, and data transmission and reception. ZigBee operates across multiple frequency bands, offering flexibility based on regional regulations and application requirements: 868 MHz, 915 MHz, and 2.4 GHz.

4.2.1.1 Medium Access Control layer

The MAC sub-layer provides key functionalities such as: beacon management, channel access management, guaranteed time slot management, frame validation, acknowledged frame delivery, association and disassociation, and sSecurity hooks. ZigBee defines two operation modes to balance power efficiency and communication reliability:

1. *Beacon-enabled mode*: the PAN Coordinator periodically transmits beacons to synchronize devices. Commonly used in star topologies. Uses a combination of slotted CSMA-CA and scheduled transmissions to manage channel access.
2. *Non beacon-enabled mode*: devices communicate using un-slotted CSMA-CA, meaning no predefined transmission schedule. Provides uncoordinated access, reducing overhead but increasing the risk of collisions.

Channel access mechanism Since multiple devices share the same frequency, ZigBee uses a mix of scheduled and random access to prevent interference:

- *Scheduled Access*: managed by the PAN Coordinator (only in beacon-enabled mode).
- *Random Access*: used for communication between RFDs or between RFD/FFD and the PAN Coordinator. Allowed in both operation modes.

4.2.1.2 CSMA/CA

ZigBee devices rely on CSMA/CA to reduce transmission conflicts. Each device tracks key variables to manage communication:

- The number of back-offs, which counts transmission delays.
- The contention window, controlling how long the channel must remain clear before transmission.

- The backoff exponent, determining the wait time before attempting access.

The CSMA/CA process begins with the transmitting node waiting for a random backoff period. If the channel remains clear for the required CW duration, the device proceeds with transmission and awaits an acknowledgment. However, if the channel is busy, the node increases both BE and NB, repeating the process until either a successful transmission occurs or the retry limit is reached, at which point the packet is discarded. All transmissions, including ACKs, must complete within the Contention Access Period. While slotted CSMA/CA requires synchronization, un-slotted CSMA/CA operates without it. Certain packets bypass CSMA entirely and transmit immediately. If a collision occurs the device restarts the procedure.

4.2.1.3 Network formation

To join a network, ZigBee devices scan for available PANs:

- Active scanning, used by full-function devices, involves sending a beacon request, prompting nearby PAN coordinators to respond with beacons. After scanning, the device selects a PAN ID and joins.
- Passive scanning, available to both FFDs and reduced-function devices, operates similarly but without actively requesting beacons. Instead, the device listens for beacon transmissions from coordinators.

4.2.1.4 Extensions

Enhancements to the IEEE 802.15.4 standard have expanded ZigBee's capabilities for specialized applications. IEEE 802.15.4e introduces slotted channel access for improved efficiency. IEEE 802.15.4g targets smart utility applications while IEEE 802.15.4k optimizes communication for critical infrastructure monitoring, including power grids and industrial control systems.

4.2.2 Network layer

The network layer provides is used to: configure anew device, start a network, join and leave a network, address, neighbor discovery, route discovery, reception control, and routing. In ZigBee, there are three types of devices:

- *ZB coordinator* (FFD): the central coordinator that initiates and manages the network.
- *ZB router* (FFD): routers that relay messages and allow other devices to join the network.
- *ZB end-device* (RFD or FFD): end devices.

ZigBee routing integrates two primary protocols: Ad-hoc On-demand Distance Vector (AODV) and cluster tree algorithm.

4.2.2.1 Cluster tree algorithm

The cluster tree algorithm defines how a ZigBee network organizes itself into a tree structure for efficient data routing. The most important operations are:

- *Tree initialization by FFD*: a ZigBee FFD begins the tree formation by scanning available channels and selecting one with minimal interference. It sets its PAN identifier and assigns itself the network address 0, indicating its role as the coordinator. The ZigBee coordinator fixes the maximum number of routers (R_m), end-devices (D_m) that each router may have as children and the maximum depth of the tree (L_m).

- *Association of devices*: other devices can then join the network by associating with the coordinator through standard association procedures. These devices may be routers or end-devices.

- *Address assignment*: during the association process, devices are assigned 16-bit short addresses. The parent device assigns address ranges to its child devices. Each router receives a consecutive block of addresses based on its depth in the tree, while the first address in this block is used as the node's address, and the remaining addresses are available for its child devices. The size of the address range $A(d)$ assigned to a router node at depth is calculated using the following formula:

$$A(d) = \begin{cases} 1 + D_m + R_m & \text{if } d = L_m - 1 \\ 1 + D_m + R_m A(d+1) & \text{if } 0 \leq d < L_m - 1 \end{cases}$$

Routing within a ZigBee network follows the tree structure formed by the coordinator and its child devices. Messages are routed through the tree in the following manner:

- If the destination address belongs to one of the child end-devices, the message is routed directly to the destination.

- If the destination address corresponds to a child router, the message is sent to that router for further forwarding.

- If the destination address is not found among the children, the message is forwarded to the parent node for further routing.

4.2.2.2 Ad-hoc On-demand Distance Vector

AODV is a reactive routing protocol used when a device needs to send data to an unknown destination. Here's how AODV works:

1. *Route request* (RREQ): when a device wants to communicate with a destination, it broadcasts a RREQ message.

2. *Flooding*: the RREQ message is flooded across the network, and each node that relays the request stores the address of the node that forwarded it, setting up a reverse path to the source.

3. *Route reply* (RREP): when the destination node receives the RREQ, it sends a RREP message back to the source, traveling along the reverse path set up during the flooding process.

Routing tables are updated as nodes process these RREQ and RREP messages. A node's routing table contains the destination address, the nex-hop address and the entry status (active or not). Additionally, a Routing Discovery Table (RDT) keeps track of RREQ messages.

Routing cost The cost of a path P is the sum of the individual costs of the links that make up the path:

$$C\{P\} = \sum_{i=1}^{L-1} C\left\{[D_i, D_{i+1}]\right\}$$

The cost for each link l follows a standard cost function:

$$C(l) = \begin{cases} 7 \\ \min\left(7, \text{round}\left(\frac{1}{p_l^4}\right)\right) \end{cases}$$

4.2.3 Application profiles

ZigBee also provides predefined application profiles that ensure interoperability across different devices and manufacturers. These profiles define a common language for data exchange with a set of processing actions for devices. This allows device interoperability across manufacturers and simplicity and reliability for end users. Each profile is made up of:

- A set of devices needed for the application.
- A set of clusters that implement the required functionality.
- A set of attributes representing device state.
- A set of commands enabling communication between devices.

4.3 THREAD

THREAD is a low-power, low-latency wireless mesh networking protocol designed for secure and reliable IoT communication. It is based on open and proven standards, ensuring broad interoperability and long-term viability.

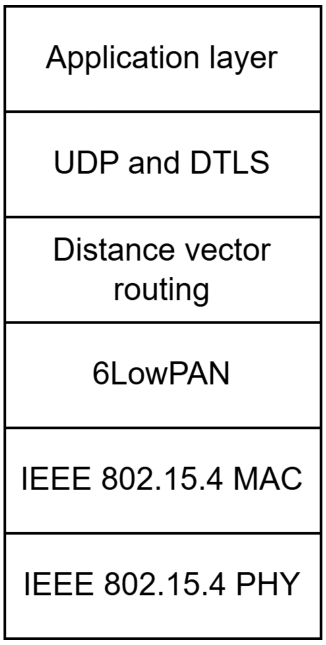

Figure 4.3: 6LowPAN protocol stack

THREAD devices are categorized based on their capabilities and roles within the mesh network:

- *Full THREAD Devices* (FTD): fully participate in routing and maintain a complete network topology. Can be divided as:
  - *Router*: forward packets for other devices and provide critical services such as network joining and security. Routers are always powered and can dynamically downgrade to Routing-Eligible End Devices (REEDs) if necessary.
  - *Leader*: special role assigned to one router, responsible for managing the overall network structure. It coordinates upgrades and other critical decisions. If the leader fails, another router automatically assumes the role, ensuring network stability.
  - *Routing-Eligible End Device* (REED): a non-routing that can upgrade to a Router if the network requires it.
  - *Full End Device* (FED): does not perform routing and cannot upgrade to a router, but otherwise maintains full communication capabilities.
- *Minimal THREAD Devices* (MTD): a basic non-sleepy device that communicates through a parent router but does not perform routing. Can be divided as:
  - *Sleepy End Device* (SED): a battery-powered device that conserves energy by sleeping for extended periods and only periodically waking up to communicate.
  - *Synchronized Sleepy End Device:* (SSED): a variant of the SED that synchronizes its sleep cycles for more predictable communication patterns.
  - *Bluetooth End Device*: a device that supports Bluetooth Low Energy (BLE) alongside THREAD to enable flexible connectivity options.
- *Border router*: connects the THREAD network to external networks, allowing seamless communication between THREAD devices and the broader Internet.

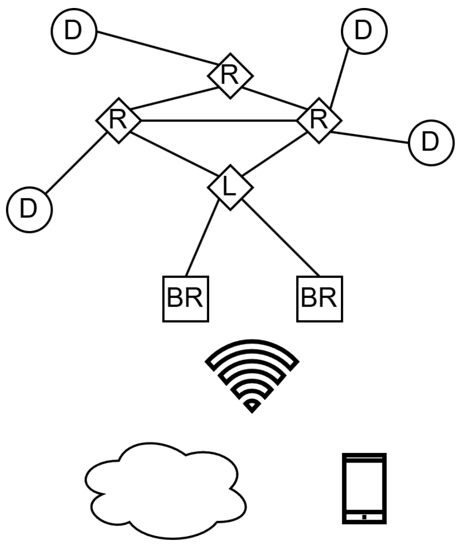

Figure 4.4: THREAD network structure

4.3.1 IPv6 over Low-power Wireless Personal Area Networks

6LowPAN is an adaptation layer designed to enable IPv6 communication over low-power wireless networks. It addresses several critical challenges inherent in these networks, such as limited frame size, energy efficiency, and the need for auto-configuration. By providing a framework for efficient communication, 6LowPAN bridges the gap between traditional IP-based networking and resource-constrained wireless environments.

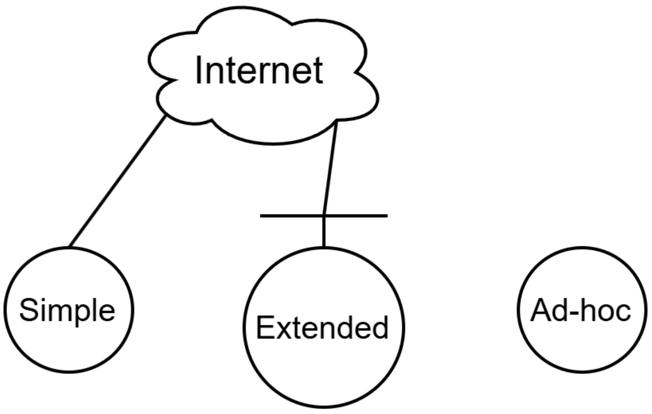

Figure 4.5: 6LowPAN architecture

The architecture of 6LowPAN involves various types of networks. A simple LoWPAN typically consists of a single edge router, while an extended LoWPAN incorporates multiple edge routers connected via a common backbone link. An Ad-hoc LoWPAN , on the other hand, operates independently without external routing. These networks face significant challenges when integrating with the broader Internet application protocol compatibility, IPv4 interconnectivity, firewalls, NATs, and security concerns.

4.3.1.1 Header compression

One of the key features of 6LowPAN is its ability to perform efficient header compression. This allows the large IPv6 headers, along with extension headers and UDP headers, to be compressed into smaller formats suitable for low-power wireless links. The initial specification for this was defined in RFC4944, with ongoing updates provided by draft-ietf-6LowPAN-hc. Additionally, 6LowPAN supports fragmentation, breaking down larger IPv6 packets into smaller chunks that fit within the constrained frame sizes of technologies like IEEE 802.15.4.

4.3.1.2 Addressing

Addressing in 6LowPAN is optimized for low-power environments. Nodes within a 6LowPAN network often share a common prefix, allowing for efficient compression of IPv6 addresses. Interface identifiers are typically derived from the link-layer address during auto-configuration, further reducing the overhead of address management. Network auto-configuration is achieved through neighbor discovery mechanisms, enabling seamless integration into existing infrastructures. Communication support includes unicast, multicast, and broadcast modes, with multicast being compressed and mapped to broadcast for efficiency.

4.3.1.3 Routing

Routing in 6LowPAN leverages protocols like IETF RPL (Routing Protocol for Low-Power and Lossy Networks), which is specifically designed for resource-constrained environments. RPL builds a Destination-Oriented Directed Acyclic Graph (DODAG) based on specific objective functions, which take into account metrics. This proactive distance-vector approach ensures adaptability to different application requirements, whether they involve home automation, commercial building automation, industrial automation, or urban environments. When it comes to routing protocols for wireless sensor networks (WSNs), the ideal solution must balance shortest-path routes, overlap avoidance, and energy efficiency. Distance-vector algorithms, which associate costs with links to determine the shortest path, are commonly used. Each router maintains local next-hop information, ensuring efficient packet forwarding. Alternatively, link-state algorithms require each node to acquire complete network information, typically through flooding, and calculate a shortest-path tree to each destination. Proactive routing acquires information before it is needed, while reactive routing discovers paths dynamically as required.

In the context of low-power and lossy networks, the IETF's ROLL working group has developed RPL, a proactive distance-vector routing protocol tailored for embedded applications. RPL supports unicast, anycast, and multicast communication, adapting to various metrics such as reliability, latency, and throughput. It also incorporates constraint-based routing, allowing parallel paths to be established based on specific criteria. Scalability is a key consideration, as RPL must accommodate diverse application requirements, from time-sensitive alarm systems to energy-optimized telemetry.

4.3.1.4 Summary

The 6LowPAN format plays a crucial role in enabling IPv6 over low-power wireless links. It employs stateless compression techniques to reduce the size of IPv6 and UDP headers. Payload length is derived from the underlying link-layer header, and source and destination addresses can be elided or compressed based on the context of the transmission. This approach significantly reduces overhead, making it feasible to transmit IPv6 packets over constrained wireless networks.

4.4 Bluetooth

Bluetooth is an industrial specification designed for Wireless Personal Area Networks (WPANs). It was developed as a low-cost, low-power wireless communication technology. The technology was initially conceived as an internal project by Ericsson in 1996-1997, with the goal of creating a universal short-range wireless communication standard. In 1998, the Bluetooth Special Interest Group (SIG) was formed. By 1999, additional industry leaders joined the SIG, further solidifying Bluetooth's position as a global standard.

Bluetooth operates in the ISM 2.4 GHz band and is characterized by its small range, low complexity, and compact size. While the original Bluetooth specifications were developed by the industrial consortium, only the first two layers (PHY and MAC) were later standardized by the IEEE under the 802.15.1 working group.

4.4.1 Physical layer

The physical layer of Bluetooth operates in the globally available ISM band at 2.4 GHz. This band is divided into 79 channels (or 23 in countries like France and Japan), each spaced 1 MHz apart, covering the frequency range from 2402 MHz to 2480 MHz. The modulation technique used is Gaussian Frequency-Shift Keying (G-FSK).

A key feature of Bluetooth's physical layer is its use of Frequency Hopping Spread Spectrum (FHSS). The system hops between frequencies at a rate of 1600 hops per second, with each hop lasting 625 microseconds. The hopping sequence is pseudo-random and determined by the clock and address of the ”master” device, which regulates channel access. All other devices in the network, referred to as slaves, follow the hopping sequence defined by the master. Packets can be transmitted over durations of 1, 3, or 5 time intervals, depending on the application requirements.

4.4.1.1 Piconet

The simplest network architecture in Bluetooth is called a piconet. A piconet is an ad hoc network composed of two or more devices, where one device acts as the master and the others act as slaves. Communication within a piconet occurs exclusively between the master and the slaves; direct communication between slaves is not allowed.

A piconet can support up to seven active slaves simultaneously. Devices that are part of the piconet but not actively participating in communication can enter a parked state, allowing up to 256 devices to be associated with the piconet in this manner. Devices not part of the piconet remain in a standby state.

4.4.1.2 Architecture

Bluetooth defines two types of connections to accommodate different communication needs:

- *Synchronous Connection-Oriented* (SCO): fixed-rate, bi-directional connection that operates like a circuit-switched link. Forward Error Correction (FEC) is used to improve transmission quality. The data rate is suitable for voice communication.
- *Asynchronous Connection-less* (ACL): packet-switched connection shared between the master and active slaves, based on a polling access scheme. Multiple packet formats and physical layer codes are supported, with packets spanning 1, 3, or 5 time slots.

The original Bluetooth protocol architecture did not align with the IEEE 802 structure. However, it was later adapted by the IEEE to conform to the 802.15.1 specifications.

4.4.1.3 Packets

Bluetooth packets are structured into three main components, each serving a specific purpose in the communication process:

- *Access code*: used for synchronization and piconet identification. It ensures that devices within a piconet can align their frequency hopping sequences and communicate effectively. There are three types of access codes: Channel Access Code (CAC), Device Access Code (DAC), and Inquiry Access Code (IAC).
- *Header*: contains essential control information for managing the link.

- *Payload*: carries the actual data being transmitted. Its format depends on the type of connection and the packet configuration.

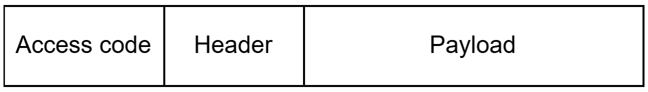

Figure 4.6: Packet format

Bluetooth devices operate in several states, each corresponding to different phases of communication:

- *Stand-by*: the device is inactive, and its radio is powered off to conserve energy.
- *Connection*: the device is actively connected to other devices. This state includes sub-states for managing ongoing communication.
- *Inquiry*: the device is searching for nearby devices by broadcasting an Inquiry Access Code (IAC).
- *Inquiry scan*: the device periodically listens for inquiry requests on specific channels with a low duty cycle.
- *Page*: the device attempts to establish a connection with another specific device using the Device Access Code (DAC).
- *Page scan*: the device listens for page requests at regular intervals.

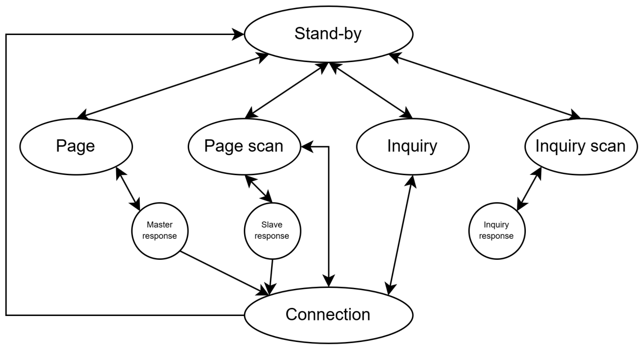

Figure 4.7: Link controller states

4.4.2 Profiles

Bluetooth profiles define basic operation modes tailored to specific applications, ensuring interoperability at the application layer. These profiles standardize how devices interact in various use cases, such as audio streaming, file transfer, and hands-free communication. By adhering to profiles, manufacturers ensure compatibility across devices from different vendors.

To conserve energy, Bluetooth supports several low-power modes for slave devices in the connection state:

- *Hold mode*: the slave temporarily stops listening to the channel for a negotiated period while retaining its AMA. This reduces power consumption without disconnecting the device.

- *Sniff mode*: the slave listens to the channel at regular intervals, maintaining its AMA. This mode balances power savings with responsiveness.
- *Park mode*: the slave releases its AMA and is assigned a Parked Member Address (PMA). It listens to the channel infrequently (very low duty cycle) and can be reactivated upon receiving an un-park message from the master.

4.4.3 Bluetooth Low Energy

Bluetooth Low Energy (BLE) is not intended to replace Bluetooth, but rather coexist alongside it. BLE is specifically designed to be energy-efficient, making it ideal for IoT applications.

4.4.3.1 Physical layer

The BLE physical layer operates in the same Industrial, Scientific, and Medical (ISM) band as BR/EDR but with distinct characteristics. It uses 40 channels, each 2 MHz wide, with three dedicated to advertisement and broadcast purposes and 37 used for data transfer via adaptive frequency hopping. It employs Gaussian Frequency-Shift Keying (GFSK) modulation and supports three PHY types.

4.4.3.2 Link layer

The BLE link layer is governed by a state machine and handles several critical duties. It manages different types of packets, supports both public and random addressing schemes, and implements adaptive frequency hopping on the data channels. It also manages Asynchronous Connection-Less (ACL) links and ensures reliable communication through sequence numbers (SN) and next expected sequence numbers (NESN) for packet ordering and acknowledgment.

4.4.3.3 Features

Error control in BLE is achieved through sequence numbers and retransmissions. Central and peripheral nodes play key roles in BLE communication. Peripheral nodes periodically transmit advertisements, ranging from 20 milliseconds to 10 seconds, while central nodes scan advertisement channels to detect these broadcasts. The central node determines how long to scan (scan window) and how often to perform scans (scan interval).

Adaptive Frequency Hopping (AFH) is a key feature of BLE, where the central node selects channels to avoid interference and shares the channel map with the peripheral node.

Advertisements in BLE can be directed or undirected, connectable or non-connectable, and scannable or non-scannable. Directed advertisements accept connections only from known devices, while undirected ones are open to all. Connectable advertisements allow connection establishment, and scannable advertisements permit receiving scan requests and responding accordingly.

A central device can issue a connection request to a peripheral that is broadcasting connectable advertisements. The peripheral responds with a connection reply, establishing a connection between the two BLE devices and enabling them to exchange data. Once connected, the two devices use the Attribute Protocol (ATT) for data exchange. In this protocol, the client can discover available attributes or resources on the server and perform read and write operations on these attributes.

4.5 Radio Frequency Identification

Radio Frequency Identification (RFID) is an advanced evolution of traditional bar codes, designed to enable faster and more efficient identification of assets. Unlike optical bar codes, RFID transitions to electronic bar codes with wireless communication capabilities and replaces optical readers with wireless ones. This technology allows for contactless identification, making it highly versatile and applicable across various industries.

4.5.1 History

The origins of RFID can be traced back to World War II, where the first passive RFID system was developed by the Luftwaffe. Around the same time, the RAF introduced the first active RFID system, the "Friend or Foe" identification system. In 1945, the Theremin Bug emerged as an early example of RFID-like technology, followed by H. Stockman's seminal paper in 1948, which introduced the concept of communication through reflected power.

The commercial deployment of RFID began in the 1960s and 1970s, with applications such as Electronic Article Surveillance (EAS) in retail and low-frequency (LF) animal identification. The first RFID patent was granted in 1973. By the 1980s, mass deployment of RFID systems began, including the introduction of active read/write systems with embedded processors. The 1990s saw the development of passive systems with EEPROM memory and automatic toll systems. The Auto-ID Center at MIT was founded in 1999, leading to standardization efforts by EPC Global (Electronic Product Code). Advances in conductive inks and low-power processors in the 2000s further propelled RFID adoption, with Walmart emerging as a major use case by 2004.

4.5.2 Architecture

An RFID system consists of two main components:

- *Reader*: the reader is responsible for transmitting radio waves to the tag and receiving data from it. It includes memory, a processing unit, control logic for collision arbitration, and interfaces such as Ethernet or Wi-Fi. It may also have its own power supply.

- *Tag*: the tag is attached to the object being identified and contains a battery scavenging circuitry (in active or semi-passive tags), an EEPROM to store the unique identifier, control logic, and RF antennas. Some advanced tags may include sensors or additional processing units. The tags can be:

  – *Passive tags*: these tags derive their operational power from the reader's radiated energy. They are low-cost but have a short range and are self-sustaining.

  – *Semi-passive tags*: these tags use a battery to power their internal circuits but rely on the reader for communication. They offer a medium range, average cost, and long life.

  – *Active tags*: these tags have their own power source and built-in transmitters, enabling high-range communication. However, they have a limited lifetime due to battery constraints.

  The tags may have 1 bit storage or more, based on the usage.

4.5.3 Physical communication

RFID systems operate using either near-field or far-field models:

- *Near field* (HF): based on inductive coupling between the reader and tag, typically operating at 125 kHz or 13.56 MHz. The reading range is comparable to the coil diameter, and functioning modes can be duplex (concurrent charging and transmission) or sequential (charging and transmission decoupled). HF systems are relatively immune to environmental interference but have limited range and are sensitive to orientation.

- *Far field* (UHF/SHF): based on electromagnetic coupling, where the tag scavenges energy from EM waves emitted by the reader. Transmission occurs via back-scattering. UHF systems offer tens of meters of read range, high bit rates, and are more affected by environmental conditions.

4.5.4 Tag arbitration

When multiple tags respond simultaneously, collision arbitration mechanisms are required. These mechanisms can be classified into vertical and horizontal categories:

- *Vertical classification*: these includes ALOHA-like mechanisms (includes Slotted ALOHA and Dynamic Frame ALOHA, where tags transmit in predefined slots and retransmit if collisions occur) and tree-based mechanisms (includes Binary Tree algorithms, where colliding tags are partitioned recursively).

- *Horizontal classification*: differentiates between centralized and distributed approaches and types of channel feedback.

The efficiency of arbitration is defined as the ratio of the tag population size to the length of the arbitration period:

$$\eta = \frac{N}{L_N}$$

4.5.4.1 Frame ALOHA

Frame ALOHA is an extension of the ALOHA protocol, where tags are allowed to transmit once per frame. Each frame consists of r slots, and every tag selects a slot randomly for transmission. If a transmission fails due to a collision, the tag retries in the next frame.

The average throughput $\mathbb{E}[S]$ in a single frame is given by:

$$\mathbb{E}[S] = n\left(1 - \frac{1}{r}\right)^{n-1}$$

Here, n is the number of tags and r is the number of slots in the frame. The efficiency η is then calculated as:

$$\eta = \frac{\mathbb{E}[S]}{r} = \frac{n}{r}\left(1 - \frac{1}{r}\right)^{n-1}$$

The efficiency is maximized when the number of slots equals the number of tags.

For multiple frames, the efficiency depends on the initial tag population N, the current backlog n, and the frame size r. In Dynamic Frame ALOHA, the frame size r is dynamically adjusted to match the current backlog n. The efficiency is expressed as:

$$\eta = \frac{N}{L_N}$$

The average tag resolution process L_n can be recursively calculated as:

$$L_n = r + \sum_{i=0}^{n-1} \Pr(S = i) L_{n-i}$$

A key challenge is that the initial population N and the backlog n are typically unknown. To address this, tag arbitration involves two modules:

- *Backlog Estimation Module*: estimates the current backlog n.
- *Collision Resolution Module*: runs Frame ALOHA with the estimated backlog n.

Schoute's method assumes that the frame size r is kept equal to the current backlog n. Under this assumption, the number of terminals transmitting in a slot follows a Poisson process with intensity 1 terminal per slot. The average number of terminals involved in a collided slot can be approximated as:

$$H = \frac{1 - e^{-1}}{1 - 2e^{-1}} = 2.39$$

The backlog n is then estimated as:

$$n_{\text{est}} = \lfloor H \cdot c \rceil$$

Here, c is the number of collided slots.

4.5.4.2 Binary tree

The Binary Tree algorithm uses random numbers to partition colliding tags into smaller groups. Tags maintain counters initialized to 1, and the reader broadcasts commands to manage the process:

- *Trigger command*: sent at the start or after successful/empty slots. Tags decrement their counters and transmit if their counter reaches 0.
- *Split command*: sent after a collision. Tags with a counter of 0 randomly choose a new counter value from [0, 1]. Tags with a counter greater than 0 increment their counter.