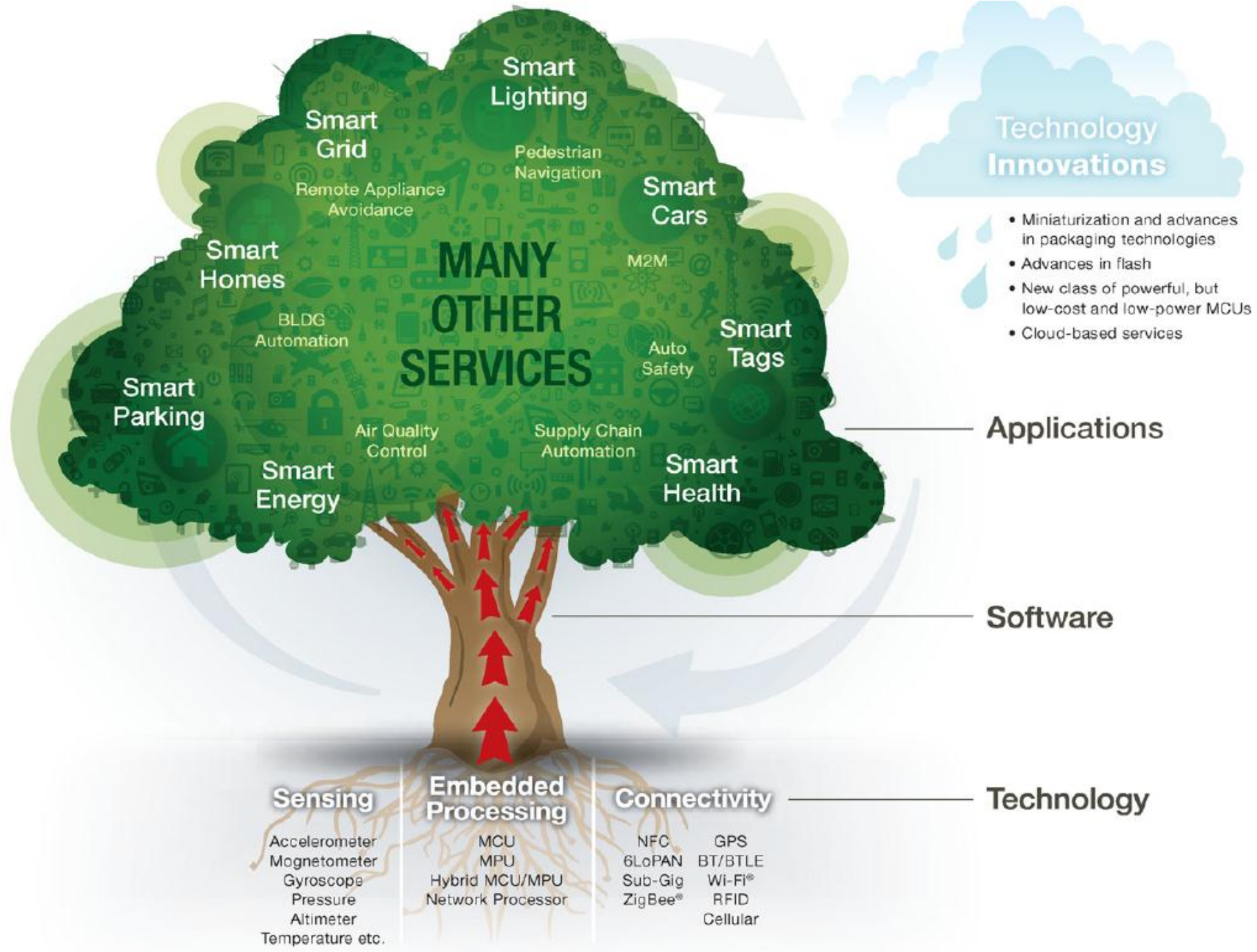


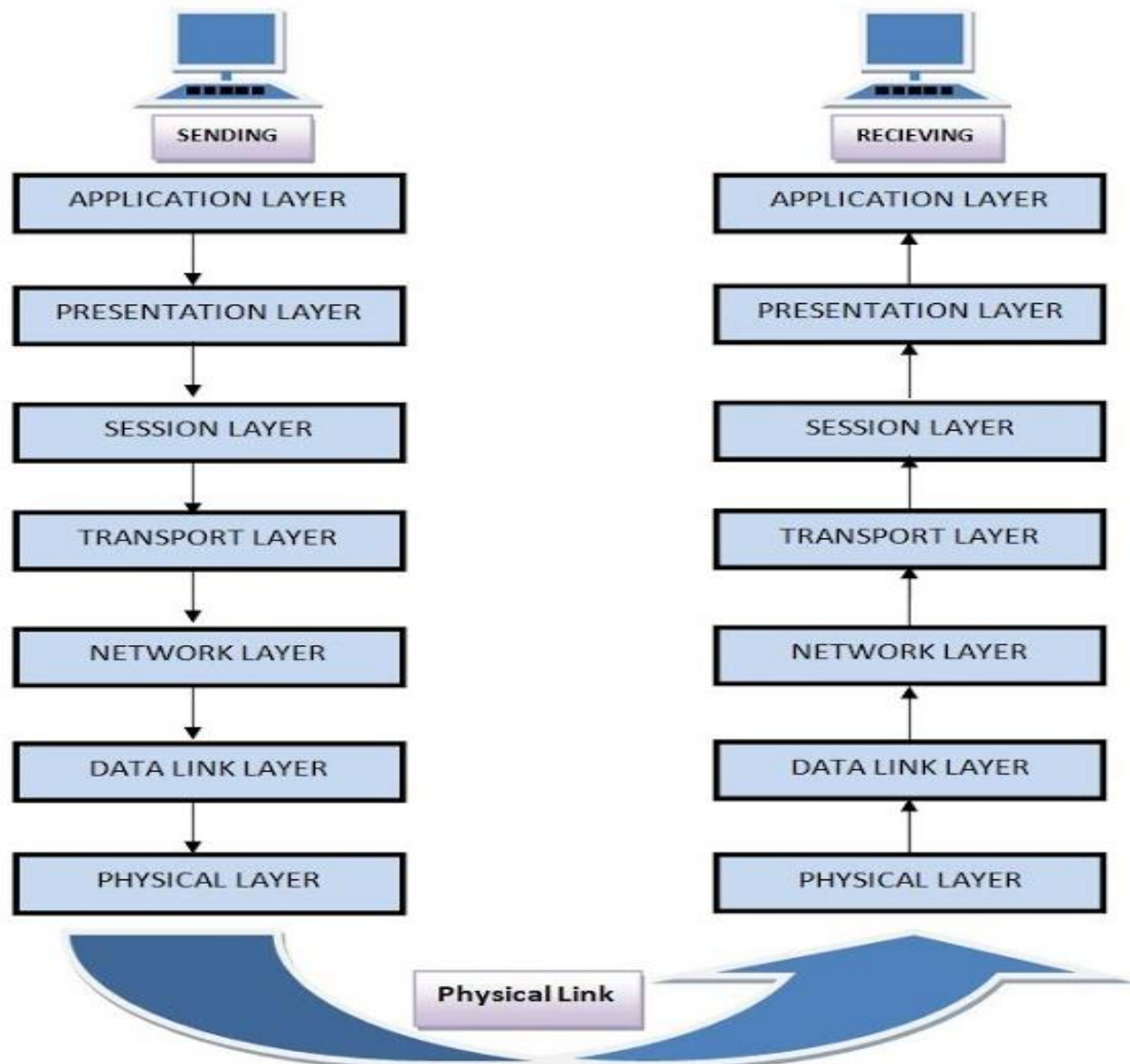
Internet of Things

IoT Protocols - Unit - 2

- Sri TDNSSSRAO, SVEC
- Functionality of Layers in IOT
- Study of protocols
 1. Wireless HART
 2. Z-Wave
 3. 6LoWPAN
 4. RPL
 5. CoAP
 6. MQTT



ISO/OSI Model Layers



Layer and Protocol

- **Layering** is the organization of programming into separate functional components that interact in some sequential and hierarchical way, with each layer usually having an interface only to the layer above it and the layer below it.
- **Layer** provides end-to-end sequenced delivery of data. It is the lowest **layer** that provides applications and higher **layers** with end-to-end service. **Layers** hides the topology and characteristics of the underlying network from users.
- In networking, **layering** means to break up the sending of messages into separate components and activities. Each component handles a different part of the communication. Examples: ISO/OSI model or Transmission Control **Protocol**/Internet **Protocol** (TCP/IP) model.
- **Protocols** provide us with a medium and set of rules to establish communication between different devices for the exchange of data and other services. **Protocols** are needed in every field like society, science & technology, Data Communication, media, etc.
- A **protocol** is a set of rules and conventions that describe how information is to be exchanged between two entities. Each **layer** adds the necessary information to the data so that the receiving system understands how to handle the data and is able to route the data.
- The division of network **protocols** and services into **layers** not only helps simplify networking **protocols** by breaking them into smaller, more manageable units, but also offers greater flexibility. By dividing **protocols** into **layers**, **protocols** can be designed for interoperability.

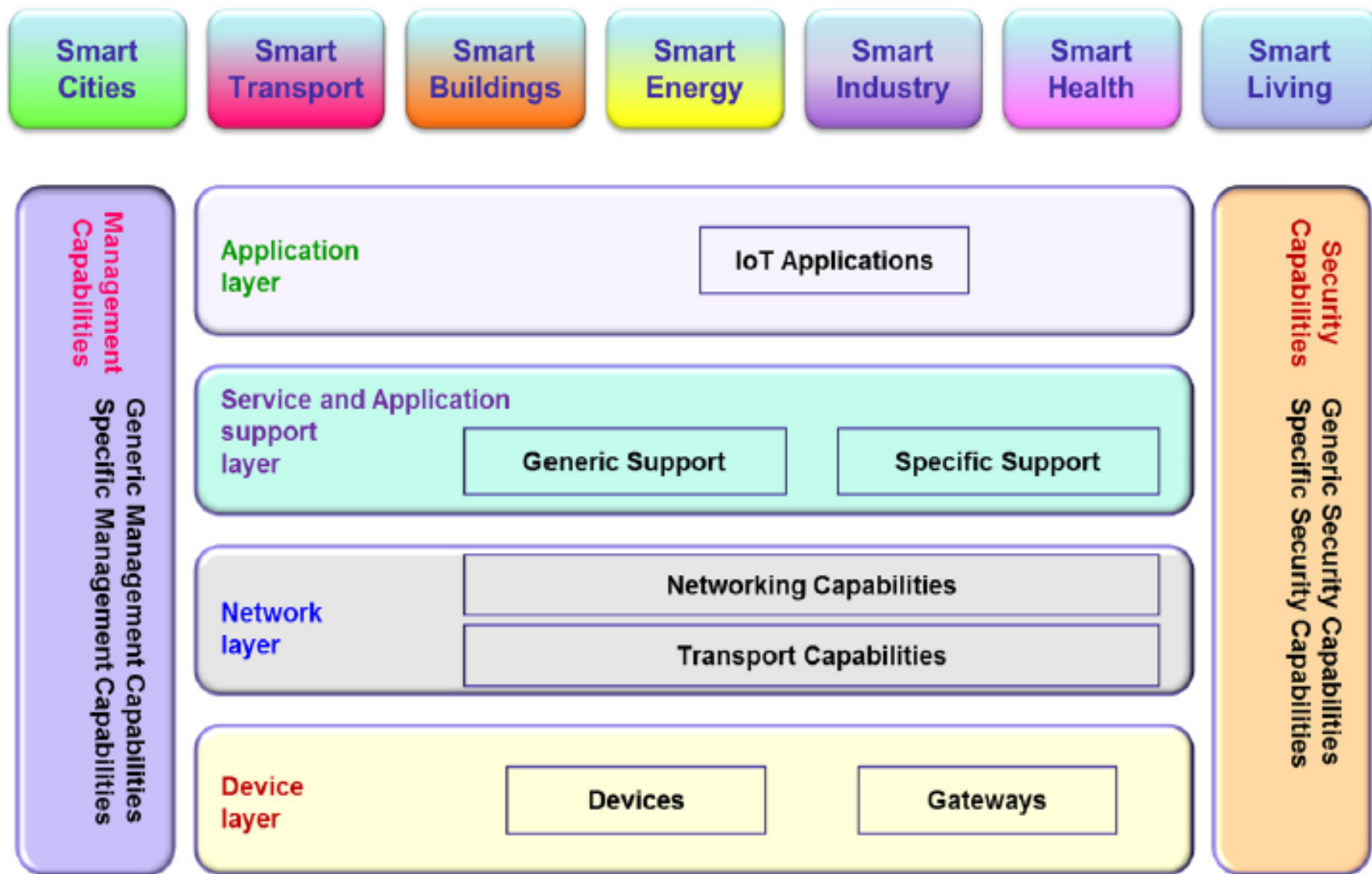


Figure 3.4 IoT Layered Architecture (Source: ITU-T)

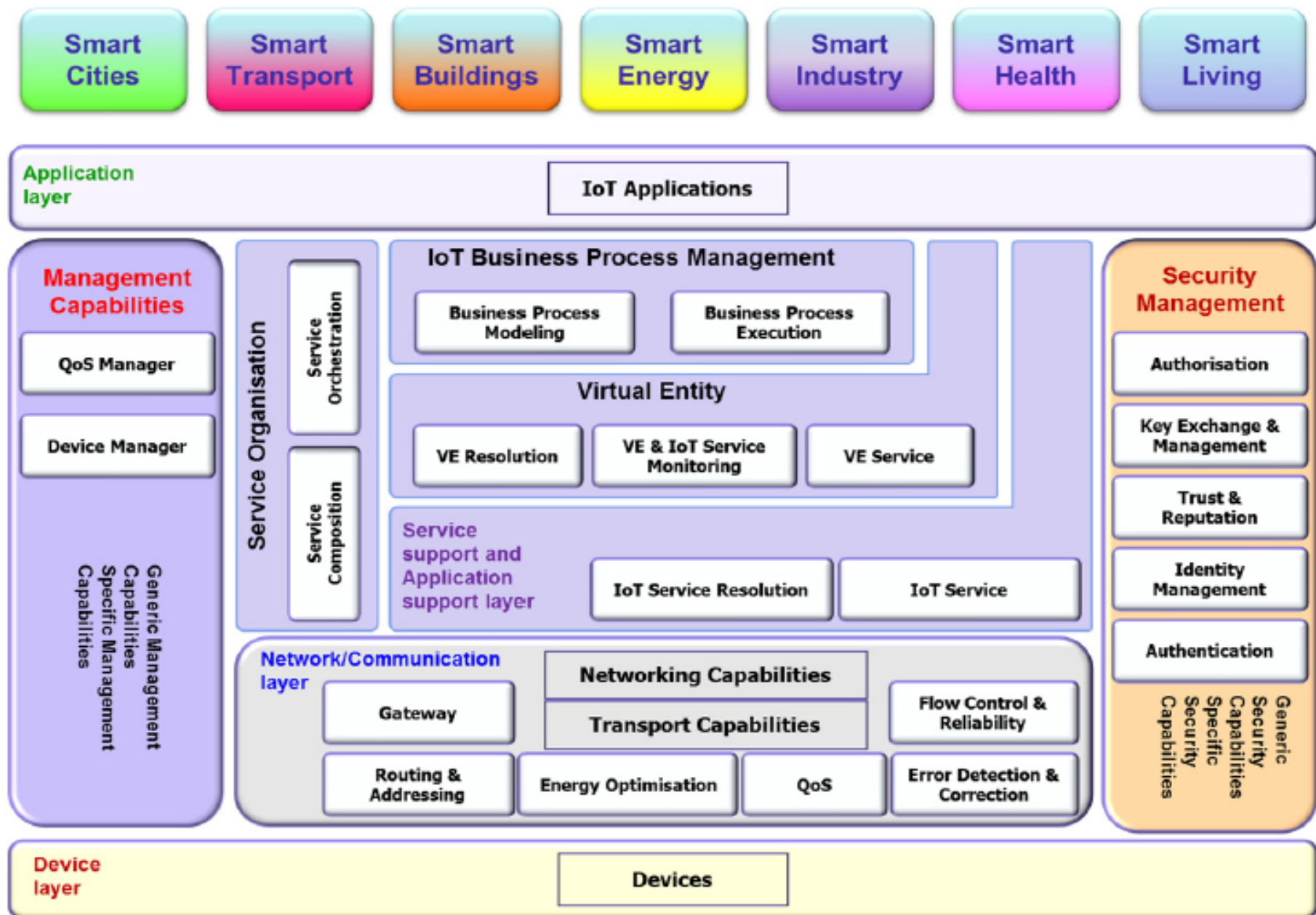
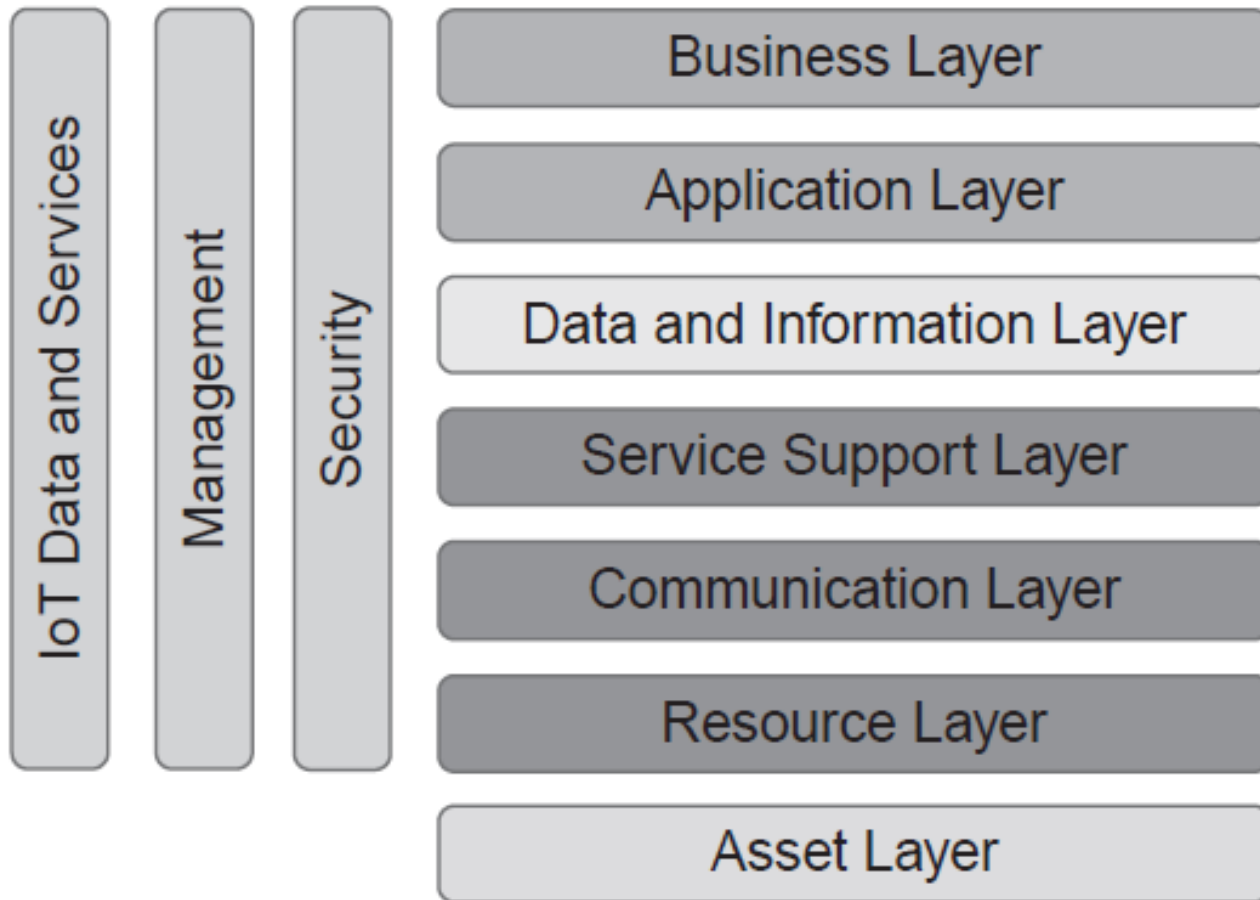


Figure 3.5 Detailed IoT Layered Architecture (Source: IERC)

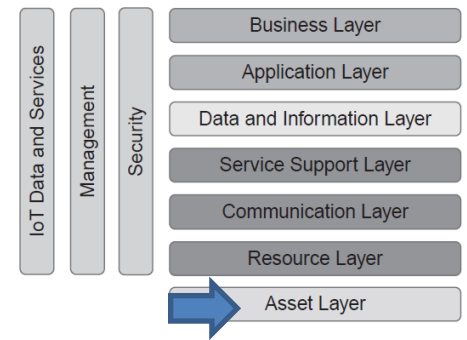
An IoT architecture outline

Functional layers



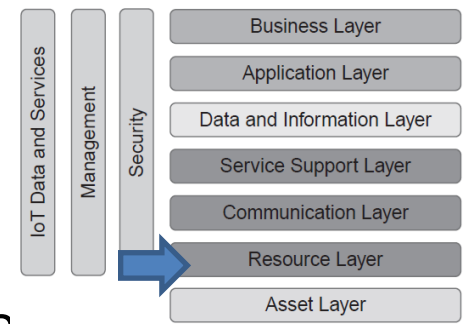
Functional layers and capabilities of an IoT solution.

Asset Layer



- At the lowest level is the Asset Layer.
- The assets of interest are the real world objects and entities that are subject to being monitored and controlled, as well as having digital representations and identities. The typical examples include vehicles and machinery, fixed infrastructures such as buildings and utility systems, homes, and people themselves thus being inanimate as well as animate objects.
- Assets can also be of a more virtual character, being subjective representations of parts of the real world that are of interest to a person or an organization.
- A typical example of the latter is a set of particular routes used by trucks in a logistics use case. Information of interest may then be traffic intensity, roadwork, or road conditions based on the actual weather situation.

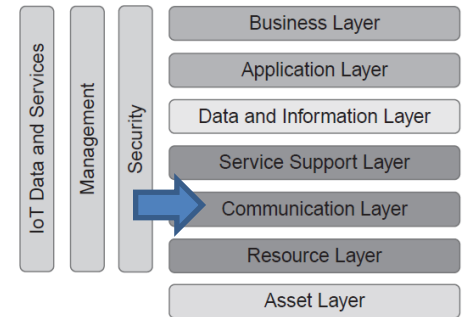
Resource Layer



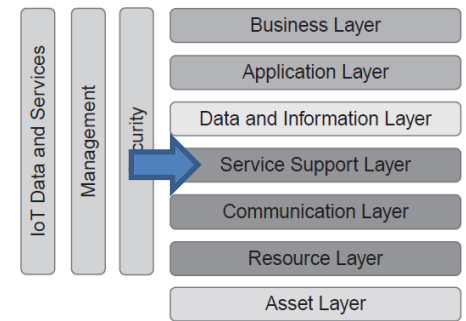
- Assets are instrumented with embedded technologies that bridge the digital realm with the physical world, and that provide the capabilities to monitor and control the assets as well as providing identities to the assets.
- The Resource Layer provides the main functional capabilities of sensing, actuation, and embedded identities. Sensors and actuators in various devices that may be smartphones or Wireless Sensor Actuator Networks (WSANs), M2M devices like smart meters, or other sensor/actuator nodes, deliver these functions.
- Gateways of different types are placed that can provide aggregation or other capabilities that are closely related to these basic resources. Identification of assets can be provided by different types of tags; for instance, Radio Frequency Identification (RFID) as in (ISO/IEC RFID 2013), or optical codes like bar codes or Quick Response (QR) codes.

Communication Layer

- The purpose of the Communication Layer is to provide the means for connectivity between the resources on one end and the different computing infrastructures that host and execute service support logic and application logic on the other end.
- Different types of networks realize the connectivity, and it is customary to differentiate between the notion of a Local Area Network (LAN) and a Wide Area Network (WAN). WANs can be realized by different wired or wireless technologies, for instance, fiber or Digital Subscriber Line (DSL) for the former, and cellular mobile networks, satellite, or microwave links for the latter.
- WANs can also be provided by different actors, where some networks can be regarded as public (i.e. offered as commercial services for the general public) or as private (i.e. dedicated networks that provide services in a more closed business or entirely company internal environment).

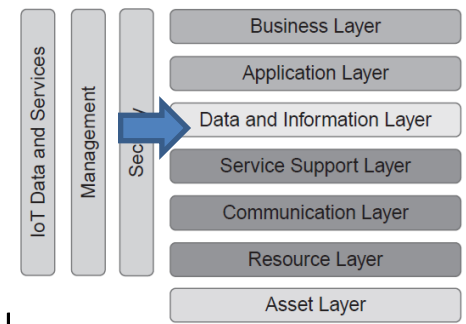


Service Support Layer



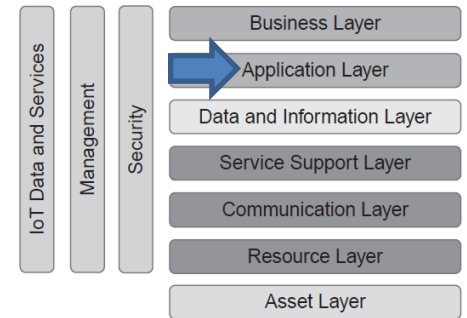
- IoT applications benefit from simplification by relying on support services that perform common and routine tasks. These support services are provided by the **Service Support Layer** and are typically executing in data centers or server farms inside organizations or in a cloud environment.
- These support services can provide uniform handling of the underlying devices and networks, thus hiding complexities in the communications and resource layers.
- Examples include remote device management that can do remote software upgrades, remote diagnostics or recovery, and dynamically reconfigure application processing such as setting event filters.
- Communication-related functions include selection of communication channels if different networks can be used in parallel, for example, for reliability purposes, and publish/subscribe and message queue mechanisms. Location Based Service (LBS) capabilities and various Geographic Information System (GIS) services are also important for many IoT applications.

Data and Information Layer



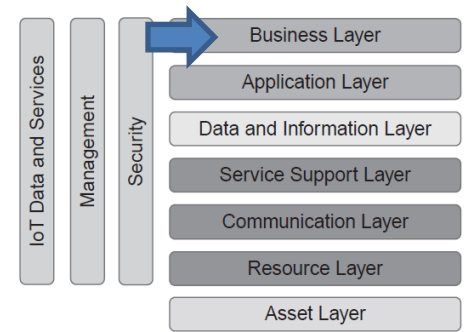
- The Data and Information Layer provides a more advanced set of functions as its main purposes are to capture knowledge and provide advanced control logic support.
- Key concepts here include data and information models and knowledge representation in general, and the focus is on the organization of information.
- Knowledge Management Framework (KMF) as a collective term to include data, information, domain-specific knowledge, actionable services descriptions as, for example, represented by single actuators or more complex composite sensing and actuation services, service descriptors, rules, process or workflow descriptions, etc.
- The KMF needs to integrate anything from single pieces of data from individual sensors to highly domain-specific expert knowledge into a common knowledge fabric. Key concepts to construct the KMF include semantic annotation, Linked Data and building different ontologies.

Application Layer



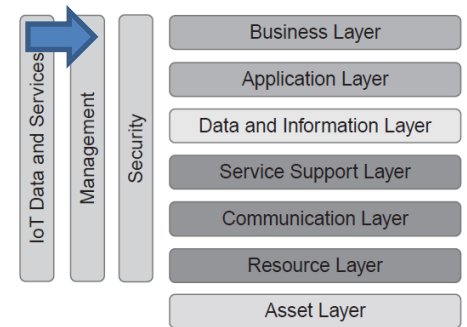
- The Application Layer in turn provides the specific IoT applications.
- There is an open-ended array of different applications, and typical examples include smart metering in the Smart Grid, vehicle tracking, building automation, or participatory sensing (PS).

Business Layer



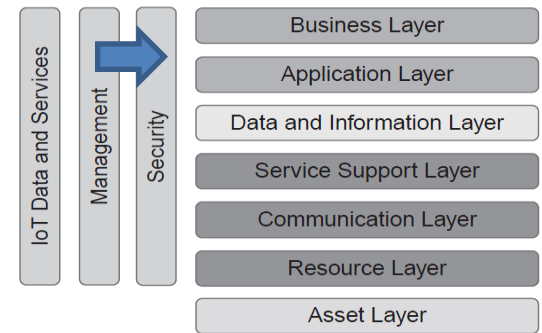
- Business Layer, which focuses on supporting the core business or operations of any enterprise, organization, or individual that is interested in IoT applications.
- This is where any integration of the IoT applications into business processes and enterprise systems takes place.
- The enterprise systems can, for example, be Customer Relationship Management (CRM), Enterprise Resource Planning (ERP), or other Business Support Systems (BSS). The business layer also provides exposure to APIs for third parties to get access to data and information, and can also contain support for direct access to applications by human users.

Management



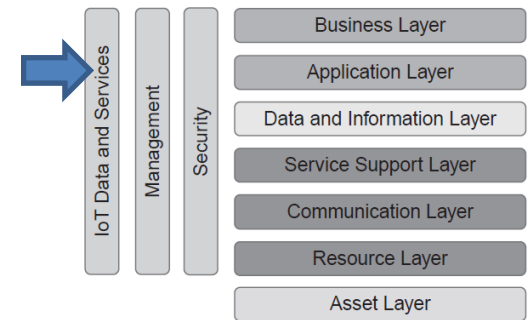
- Management, as the name implies, deals with management of various parts of the system solution related to its operation, maintenance, administration, and provisioning. This includes management of devices, communications networks, and the general Information Technology (IT) infrastructure as well as configuration and provisioning data, performance of services delivered, etc.

Security



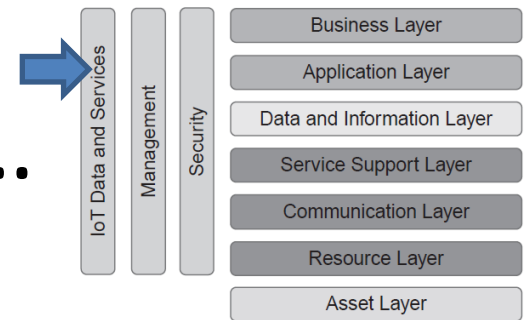
- Security is about protection of the system, its information and services, from external threats or any other harm.
- Security measures are usually required across all layers, for instance, providing communication security and information security.
- Trust and identity management, and authentication and authorization, are key capabilities. From an IoT perspective, management of privacy via, for example, anonymization, is in many instances a specific requirement.

Data and Services



- The final functional group of outlined architecture is denoted as Data and Services.
- Data and Service processing can, from a topological perspective, be done in a very distributed fashion and at different levels of complexity. Basic event filtering and simpler aggregation, such as data averaging, can take place in individual sensor nodes in WSNs, contextual metadata such as location and temporal information can be added to sensor readings, and further aggregation can take place higher up in the network topology.

Data and Services Contd...



- Embedded processing is evolving, not only towards higher capabilities and processing speeds, but also towards allowing the smallest of applications to run on them. There is a growing market for small-scale embedded processing such as 8-, 16-, and 32-bit microcontrollers with on-chip RAM and flash memory, I/O capabilities, and networking interfaces such as IEEE 802.15.4 that are integrated on tiny System-on-a-Chip (SoC) solutions.

Wireless - Highway Addressable Remote Transducer Protocol(W-HART)

Applications in Process Industry

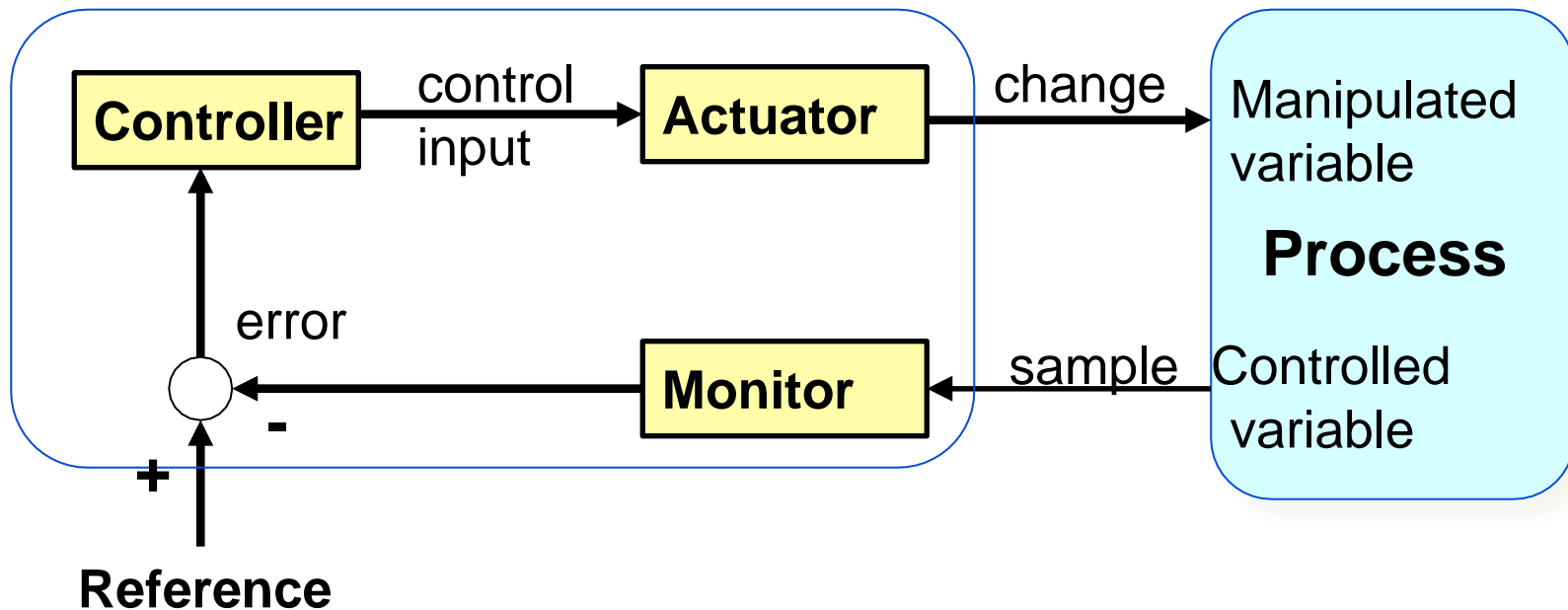
- Most widely used field communication protocol for intelligent process instrumentation
- 30 million devices worldwide
- Health, Safety, and the Environment (HSE) regulations.
- A secure networking protocol operating in the 2.4GHz ISM radio band.
- On the physical layer, W-HART utilizes IEEE 802.15.4 compatible DSSS radios.
- On the data link layer, W-HART defines a secure and reliable MAC protocol.
- The WirelessHART network layer supports mesh networking technology.
- A central network manager is responsible for the configuration of the network and scheduling of all activities in the network.
- WirelessHART networks support a wide variety of devices from different manufactures.
- Enhance safety, optimize process, protect environment
- Detect leaks before they lead to environmental problems.
- Monitor the status of manually operated valves.
- Monitor safety relief valves to detect venting to avoid accidents.

Process Control

➤ Feedback control loop controls a physical process.

Example: control temperature by manipulating heat supply.

➤ Centralized vs. peer-to-peer control.



Why Wireless

- **Cost reduction:** wiring is economically infeasible
- **Easier installation:** inaccessible locations
- **Easier maintenance**
 - ❑ Wired networks cannot handle severe heat or exposure of chemicals
 - ❑ A wireless infrastructure can remain in place for many years.

Challenges in Wireless

- Strict timing requirement
- High security concerns
- Reliable communication despite wireless deficiencies
- Plant environments are inherently unreliable
 - ❑ Interference, obstacles, power failures, lightening, storms...

Wireless Technologies

➤ Existing standards fail in industrial environments

- ❑ ZigBee: static channel
- ❑ Bluetooth: quasi-static star network

➤ WirelessHART

- ❑ For process measurement and control applications
- ❑ First open and interoperable wireless standard to address the critical needs of real-world industrial applications

➤ WirelessHART released in Sep 2007 (as a part of HART 7)

- ❑ Adds wireless capabilities to the HART protocol while maintaining compatibility with existing devices, commands and tools.

Wireless HART

- ***Backed by the Power of HART***
 - Built on proven industry standards
 - Created by industry and technology experts
 - Multi-vendor support and interoperable devices
 - Uses existing devices, tools and knowledge
- **Flexible Applications**
 - Reduced installation costs – no wires!
 - Provides a ***simple, reliable*** and ***secure*** way to deploy new points of measurement and control.
 - Process monitoring, control and asset management
 - Health, safety and environmental compliance monitoring
- **Supports All Phases of the Plant Life Cycle**
 - Fast engineering, deployment and commissioning
 - Cost-effective move from scheduled to predictive maintenance
 - Easy diagnosing and troubleshooting

Simple

- *WirelessHART* technology allows users to access the vast amount of unused information stranded in 85% of the installed HART devices.
- *WirelessHART* is a robust technology that is simple to implement. Benefits of wireless technology, maintaining compatibility with existing HART devices, tools and systems.
- **Easy Installation and Commissioning**
 - Familiar tools, work flow and procedures
 - Multiple power options
 - Reduced installation and wiring costs
 - Coexistence with other wireless networks
 - Supports both star and mesh topologies
 - Add devices one at a time
- **Automatic Network Features**
 - Self-organizing and self-healing
 - Always-on security
 - Adjusts as new instruments are added
 - Adjusts to changes in plant infrastructure

Reliable

- Industrial facilities with dense infrastructures, frequent movement of large equipment, changing conditions, or numerous sources of radio-frequency and electromagnetic interference may have communication challenges. Provides built-in 99.9% end-to-end reliability in all industrial environments.
- **Standard Radio with Channel Hopping**
 - Radios comply with IEEE 802.15.4-2006
 - 2.4GHz license free frequency band
 - “Hops” across channels to avoid interference
 - Delivers high reliability in challenging radio environments
- **Coexistence with Other Wireless Networks**
 - Clear Channel Assessments tests for available channels
 - Blacklisting avoids frequently used channels
 - Optimizes bandwidth and radio time
 - Time synchronization for on-time messaging
- **Self-Healing Network**
 - Adjusts communication paths for optimal performance
 - Monitors paths for degradation and repairs itself
 - Finds alternate paths around obstructions
 - Mesh network and multiple access points

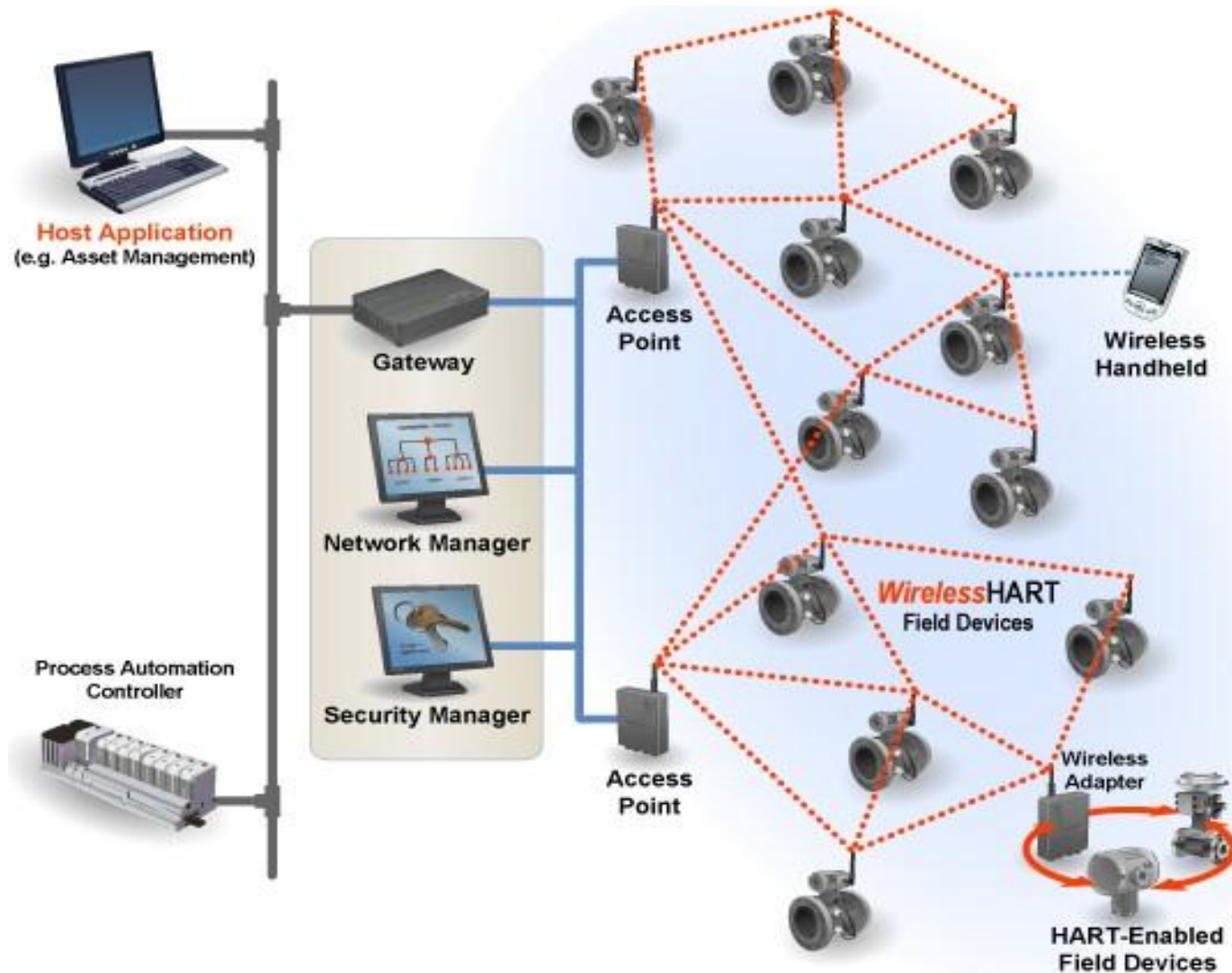
Secure

- [WirelessHART](#) employs robust security measures to protect the network and secure the data at all times. Latest security techniques to provide the highest levels of protection available.
- **Protects Valuable Information**
 - Robust, multi-tiered, always-on security
 - Industry standard 128-bit AES encryption
 - Unique encryption key for each message
 - Data integrity and device authentication
 - Rotate encryption keys used to join the network
- **Protects Wireless Network**
 - Channel hopping
 - Adjustable transmit power levels
 - Multiple levels of security keys for access
 - Indication of failed access attempts
 - Reports message integrity failures
 - Reports authentication failures
 - Safe from Wi-Fi type Internet attacks

Types of Security

- **Wireless Sensor Network Security** can be broken down into two main categories:
- **Data Security** or Confidentiality deals with maintaining the Privacy and Integrity of the information being passed over the network.
- **Network Security** or Availability deals with maintaining the functionality of the network in the face of internal and/or external attacks (intentional or unintentional).

Network Architecture



How it works



- Each *WirelessHART* network includes three main elements:
- **Wireless field devices** connected to process or plant equipment. This device could be a device with *WirelessHART* built in or an existing installed HART-enabled device with a *WirelessHART* adapter attached to it.
- **Gateways** enable communication between these devices and host applications connected to a high-speed backbone or other existing plant communications network.
- **A Network Manager** is responsible for configuring the network, scheduling communications between devices, managing message routes, and monitoring network health. The Network Manager can be integrated into the gateway, host application, or process automation controller.

Measuring devices(HART enabled)

- A **Repeater** is a device which routes messages but may have no process connection of its own. Extend the range of a *WirelessHART* network or help “go around” an existing or new obstacle (New process vessel). All instruments in a *WirelessHART* network have routing capability which simplifies planning and implementation of a wireless network.
- The **Adapter** is a device which plugs into an existing HART-enabled instrument to pass the instrument data through a *WirelessHART* network to the host. The adapter could be located anywhere along the instrument 4-20mA cable; it could be battery powered or obtain its power from the 4-20Ma cable. Some adapters will be battery powered and use the same battery to power the instrument as well – in this case there will be no 4-20mA signal to the host – all process data will be reported via *WirelessHART*
- A **Handheld Terminal** may come in two versions. In the first case, the handheld will be a standard HART FSK configuration unit (just add new device DDs or DOF files), just like the one used for everyday tasks such as routine maintenance and calibration checks. In the case of wireless support, the handheld is used to join a new instrument to an existing *WirelessHART* network.

Network Manager

➤ Centralized brain

➤ Manages the network and its devices

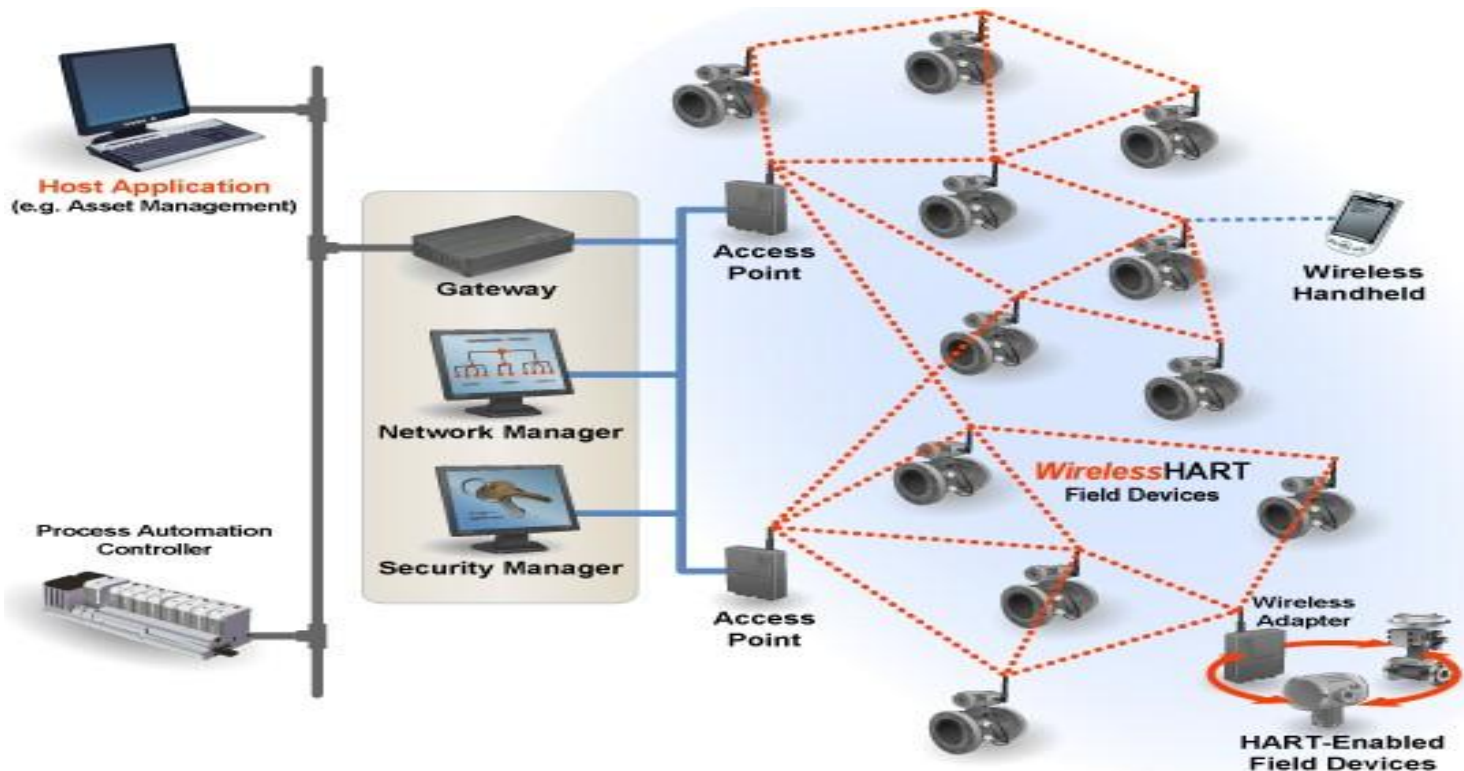
- ❑ Routing, scheduling
- ❑ User/administrator interacts with the Network Manager
- ❑ Generates network management packets to devices

➤ Redundant Network Managers supported (only one active)



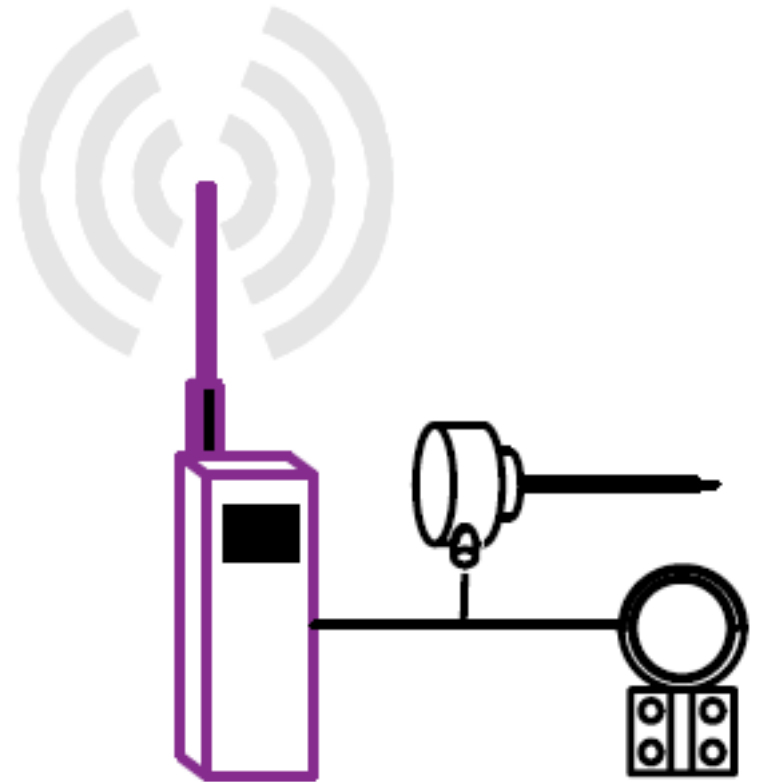
Field Devices

- Sensor/actuator/both
- Connected to the process or plant equipment
- Combines wireless communication with traditional HART field device capabilities
- Maybe line or battery-powered



WirelessHART Adapter

- Enables communication with a non-native device through a WirelessHART Network. .



Gateway

➤ One gateway can support up to 80 devices

➤ A Gateway provides

- ❑ One or more Access Points providing the physical connection into the WirelessHART network
- ❑ One or more Host Interfaces connecting the Gateway to backbone networks (e.g., the plant automation network)
- ❑ A connection to the Network Manager
- ❑ Buffering and local storage for publishing data, event notification, and common commands
- ❑ Time synchronization sourcing

Other Devices

➤ Handheld devices

- ❑ Portable applications used to configure, maintain or control plant assets.
- ❑ Typically belong to networks of different standards

➤ Plant Automation Network

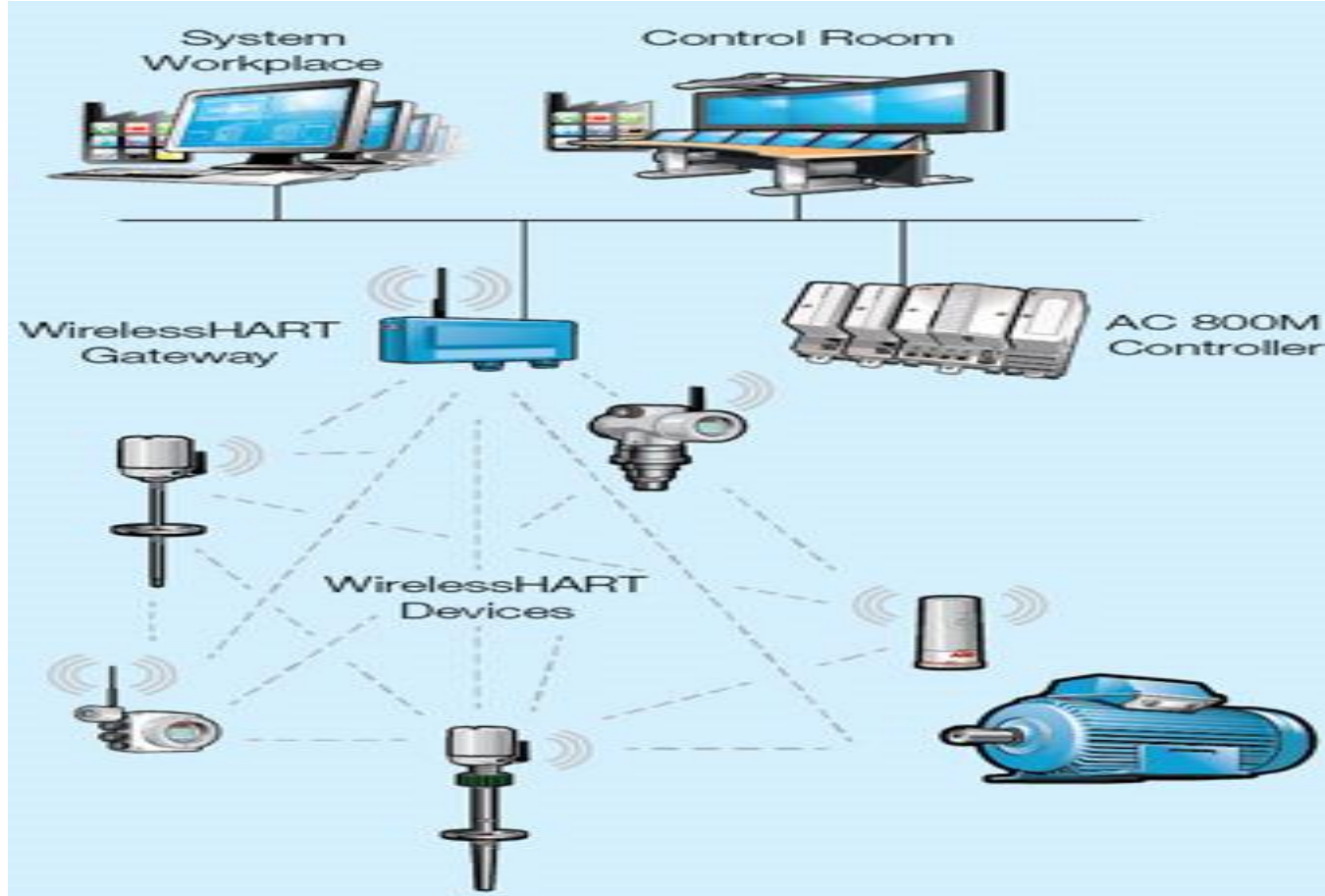
- ❑ Connects client applications to the gateway

➤ Security Manager

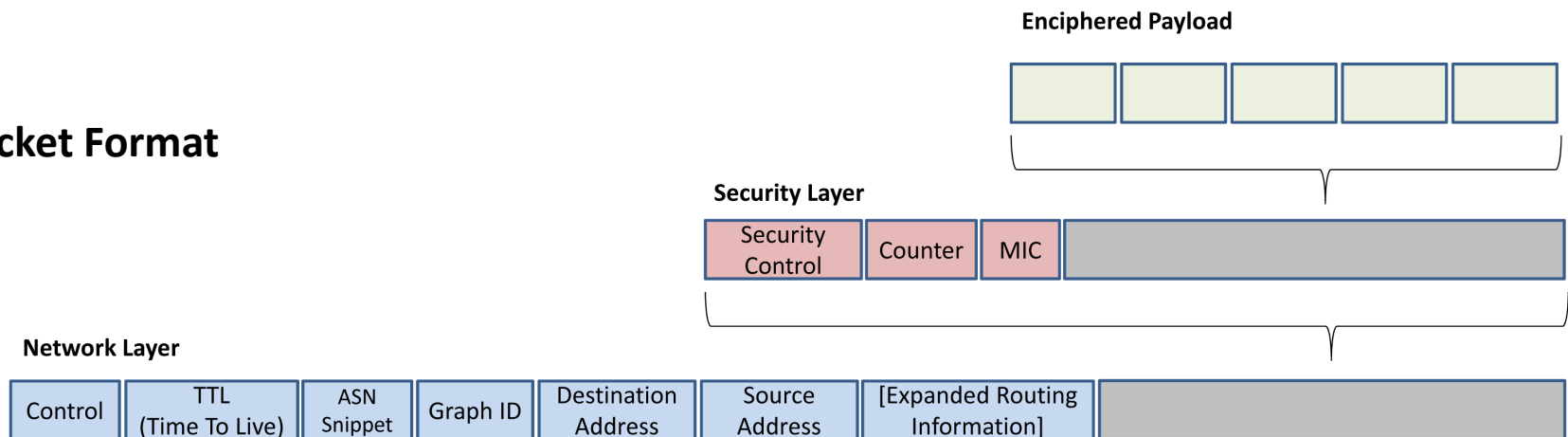
- ❑ Industry standard AES-128 ciphers/keys

- **PROTECTS VALUABLE INFORMATION – IT'S AUTOMATIC**
 - Robust, multi-tiered, always-on security
 - Industry standard 128-bit AES encryption
 - Unique encryption key for each message
 - Data integrity and device authentication
 - Rotate encryption keys used to join the network – automatic or on-demand
- **PROTECTS WIRELESS NETWORK**
 - Channel hopping for security protection and co-existence
 - Multiple levels of security keys for access
 - Indication of failed access attempts – a rogue device
 - Reports message integrity and authentication failures
 - Safe from Wi-Fi type Internet attacks

Use Case Example



Packet Format



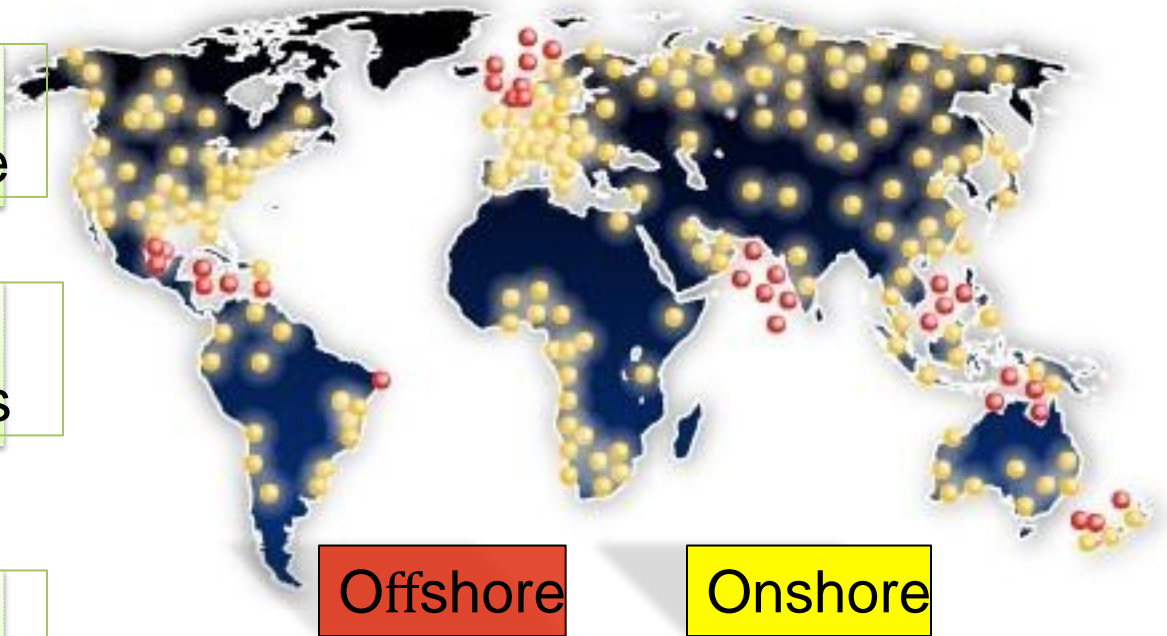
Wireless for Process Automation

➤ World-wide adoption of wireless in process industries

1.5+ billion hours
operating experience

100,000s of smart
wireless field devices

10,000s of wireless
field networks



Courtesy: Emerson Process Management

Killer App of Sensor Networks!

WirelessHART Use Cases

- Improved control of plant steam supply by detecting “cool spots” in cross plant steam lines
- Reducing risk of overfilling tanks by adding redundant level measurements (in oil and petroleum refineries)
- Monitor and control safety valves
- Monitor and control pressure and temperature of process fluids and gases

What is special?

➤ Reliable: 99.9%

➤ Secure

➤ Self-organizing, self-healing

➤ Interoperable

➤ Supports both star and mesh topologies

➤ Built-in time synchronization

WirelessHART PHY

➤ Adopts IEEE 802.15.4

- ❑ Same 16 mutually orthogonal channels
- ❑ Operates in the 2.4GHz ISM band
- ❑ Data rate of up to 250 Kbps

➤ Radio transceivers

- ❑ Omni-directional
- ❑ Half-duplex
- ❑ 100 meters LOS @ 0 dB
- ❑ Time to switch between channels: 0.192 ms
- ❑ Radio turn-on time: 4 ms

How to achieve reliability?

➤ Time diversity

➤ Channel diversity

- ❑ Channel hopping

- ❑ Channel blacklisting

➤ Route diversity

- ❑ Graph routing

➤ Power Diversity

Routing

➤ WirelessHART supports both Graph and Source routing

➤ Graph routing: provides redundant paths

➤ Routing graphs

- ❑ Uplink graph: upstream communication
- ❑ Downlink graph: Downstream communication
- ❑ Broadcast graph



Z-Wave Technology

Z-Wave - Wireless Communication Protocol

- Smart homes need wireless connectivity, and Z-wave has emerged as the ultimate solution **for home automation**. The Z-wave protocol is a wireless, radio frequency protocol designed primarily for smart home networks.
- All the existing wireless communication protocols had one or the other problem. Bluetooth and Zigbee often shortfall of range while Wi-Fi poses its own limitations in a low-power ecosystem.
- Interoperability has been another major issue as popular wireless standards have different protocols and implementations for different applications.
- No one solution could cater to the requirements of an automated home. Z-wave is, now, the solution for all those lingering issues.
- Z-wave devices are interoperable and can be easily accessed through the internet or a Z-wave gateway. With a range of around 40 meters, a Z-wave network limited to four hops can connect at most 232 devices.
- Irrespective of their make or application, all devices can have simultaneous two-way communication over the Z-wave network secured using AES. With sufficient range, optimum data speed, AES security, low-power wireless solution, and interoperable protocol. Thousands of Z-wave products in the market, serving as intelligent devices for smart home ecosystems.

What is Z-wave

- Z-wave is a wireless communications protocol developed for **smart home networks** and best wireless solution for home automation applications. More than 3000 Z-wave enabled products in the global market, and more than 100 million Z-wave devices operating in smart homes worldwide.
- Z-wave uses low-energy radio waves within the 800-900 MHz ISM frequency band and allows data communication between devices at a data rate of 100 Kbits/sec.
- The wireless network has a range of 40 meters and can use up to 4 nodes for the extension. By allowing data communication over a greater range using low-energy waves at optimum data rates, Z-wave is simply the best solution for home automation devices compared to Wi-Fi or Bluetooth.
- Almost all sorts of control and sensor applications like lighting control, remote-operated locks, smart switchboards, heating control, garage door openers, etc. can utilize Z-wave for data communication. Z-wave chips are exclusively built by Silicon Labs, and any home automation device can get Z-wave enabled using these chips.

Z-Wave	
International standard	800-900 MHz radio frequency range
Developed by	Zensys
Introduced	1999
Industry	Home automation
Physical range	100 meters



How Z-wave works

- A Z-wave network consists of two types of devices – **Controllers/Masters and Slaves**. The controller is generally a Z-wave gateway that controls data communication between other nodes and connects them to the internet.
- The controller comes with a pre-programmed networkID called HomeID. This is a 32-bit ID that identifies a particular Z-wave network. The slaves are included in the network by assignment of HomeID as well as a NodeID. The NodeID is an 8-bit ID that identifies the particular node or the Z-wave device in the network.
- The process of including a slave device in the network by assigning NodeID and acknowledging the HomeID (specific to a particular Z-wave network) is called 'inclusion'. When a slave device has to be removed from the network, HomeID and NodeID are deleted from it, and the device is the factory reset.

How Z-wave works

- The Z-wave uses radio signals in range 800-900 MHz. The actual frequency depends on the country where the device is being used.
- For example, in the USA, Canada, Mexico, and Chile uses Z-wave signals use 908.40, 908.42, and 916 MHz. UK and Europe, 868.40, 868.42, and 869.45 MHz are used as Z-wave signals.
- In India, 865.20 MHz is used by Z-wave signals.

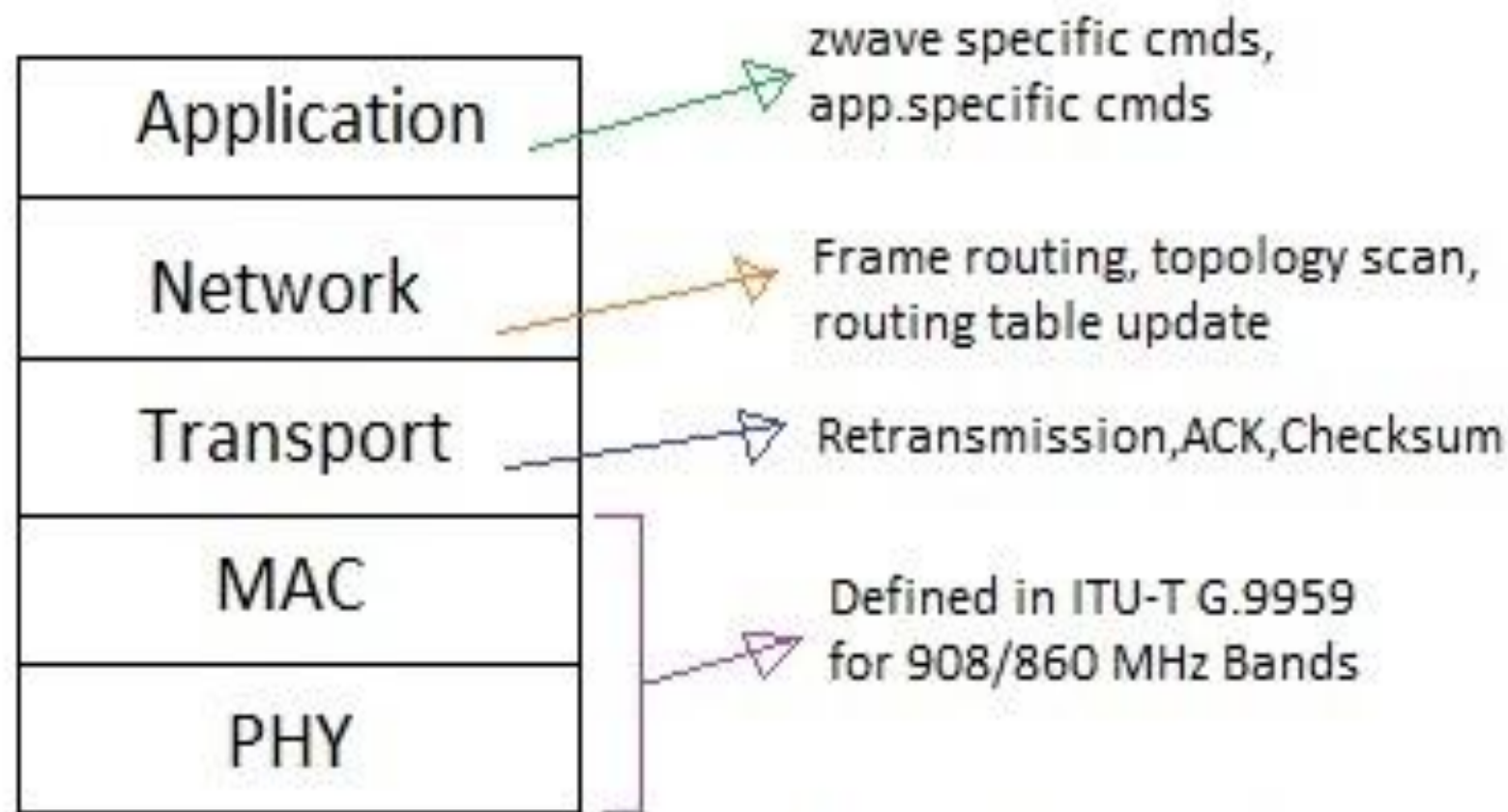
- The Z-wave devices can communicate with each other only when they have the same HomeID. Two Z-wave slave devices allotted different HomeID belong to different Z-wave networks and cannot communicate with each other.
- The controller that comes with the particular HomeID pre-programmed maintains a routing table to manage data communication between the included nodes, that is, the Z-wave devices included in its network.
- The Z-wave uses mesh network topology. Therefore, the devices (slaves) do not necessarily need to connect with the controller to communicate with other devices (slaves) of the network. They can communicate data with each other directly.
- Even some Z-wave devices transmit their own Z-wave signals. Such devices are plug-in devices and act as **repeaters**. With such Z-wave devices, a controller can communicate to the devices beyond its original range. Generally, Z-wave devices cannot directly connect to the internet. They connect to the internet only via the controller, which is also a Z-wave gateway to the internet.

Z-Wave protocol

- The first layer Application control SW is the OEM application SW like the thermostat, Sensor, Door, Lock etc.
- The routing layer uses the source routing algorithm and uses one of the Singlecast, Multicast, Broadcast, Routed Singlecast methods.
- The Transfer Layer uses the Locally Administered Addressing scheme with unique Home ID and Node ID.
- The MAC layer uses short frames of size 20-30 bytes while transmitting.
- The Z-Wave protocol uses checksum for error checking and ensure accurate data to be transferred.

Simple wireless control

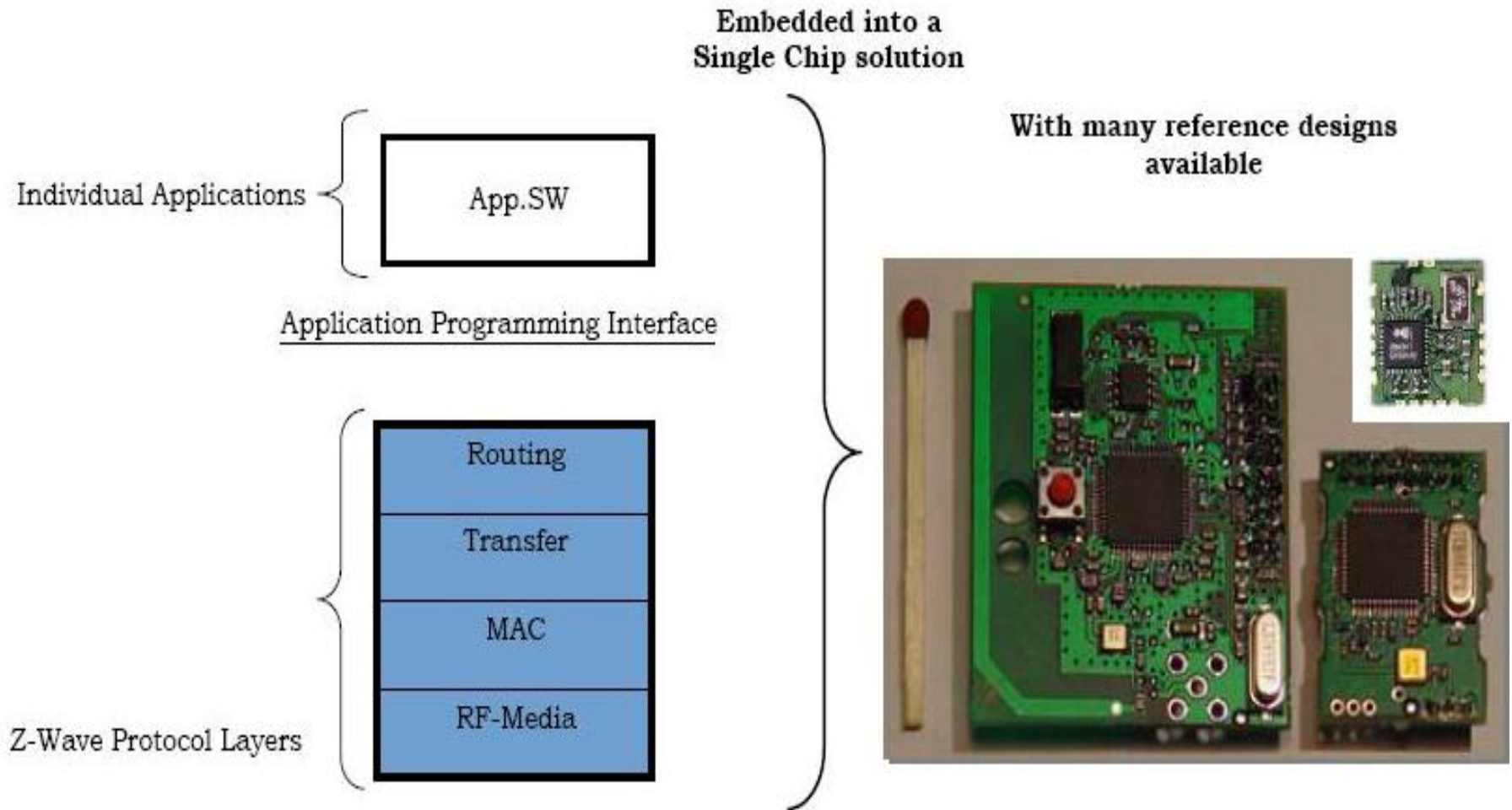
- The Z-Wave protocol is a wireless, radio frequency (RF) based communications technology designed particularly for control, monitoring and status reading of household applications.
- Z-Wave supports full mesh networks, enabling numerous Z-Wave devices to communicate with each other simultaneously. Z-Wave allows for secure and low power consuming communication between approved Z-Wave devices.
- Due to its interoperability, Z-Wave encompasses a broad ecosystem of intelligent products that work together between brands and models. With the advanced technology of Z-Wave, there is no interference from Wi-Fi, Zigbee, or other 2.4GHz wireless technologies in a similar band.



Z-Wave Protocol Stack

*security layer (implementation specific)

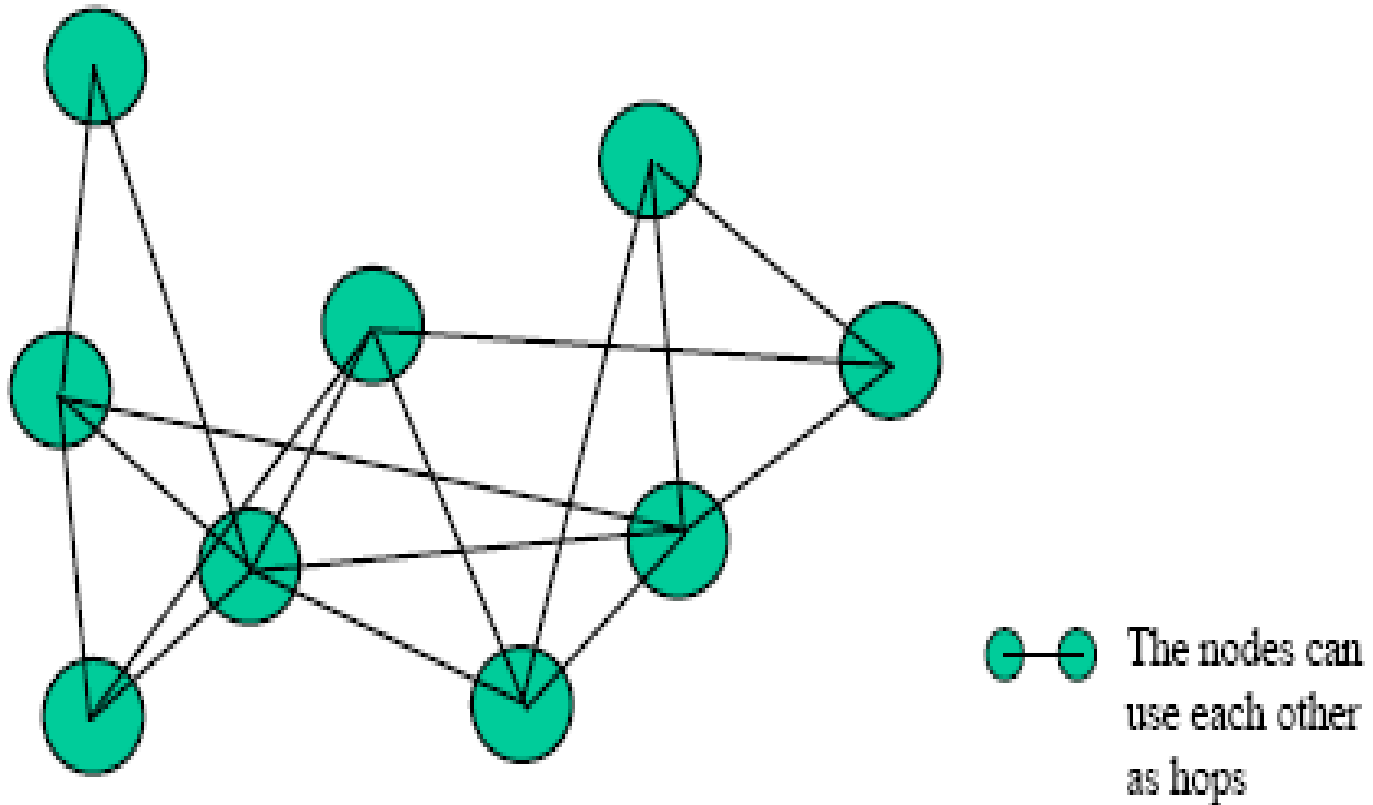
Protocol Layers



Z-Wave Characteristics

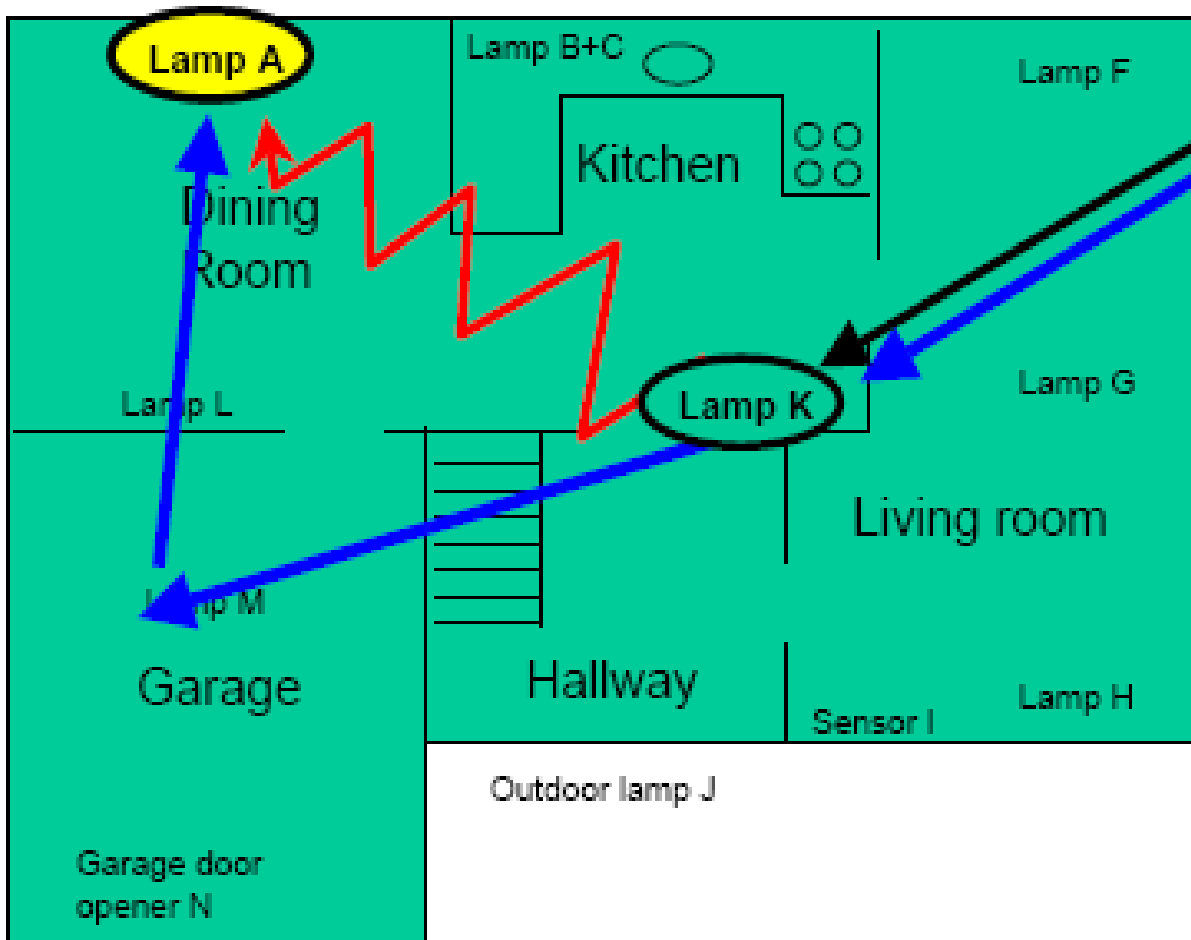
- The Z-Wave Network is of the mesh architecture.
- Efficiency of the Z-Wave Network is because of the Routing Protocol it uses.
- More than one Z-Wave Network can co-exist.
- A Z-Wave network can consist of 232 nodes to the max.

Typical Z-Wave Network



Note: Z-Wave networks devices manufactured by different companies are compatible with one another.

Real life example



Suppose if a person wants to control the Lamp A from the living room the Lamp A can be reached either through the kitchen devices or the garage devices. If either of the routes is having a dead spot then the Z-wave uses the alternative node to control the Lamp A.

Note: A dead spot is a node through with the network is not completed due to its open state. The fridge maybe in repair condition and hence can be a dead spot.

Z-Wave Network Addressing

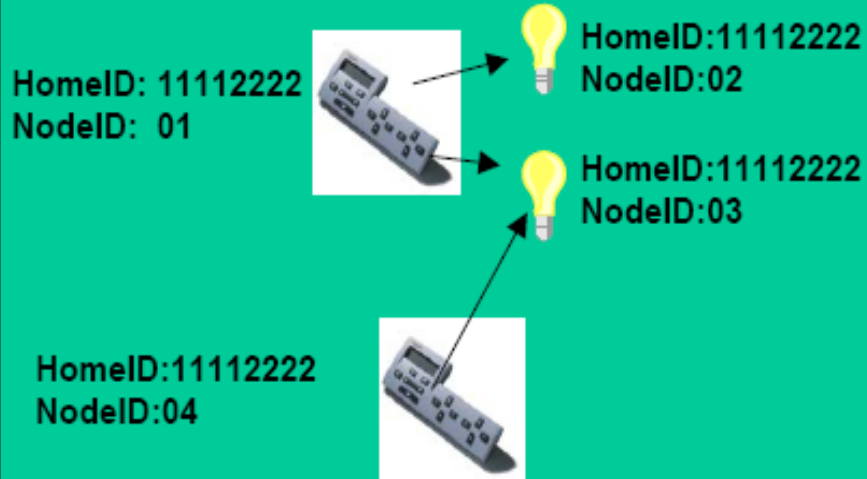
- The Z-Wave protocol uses a 32 bit identifier to address the devices it controls. i.e. if a device needs to be controlled by a microcontroller then every device controlled by that microcontroller is assigned a common Home ID. Once the device is added to the network the device looks for other devices and adds them to the same network.
- When devices are being added to the network through devices that have already been added the device in turn return assigns the same Home ID but a different Node ID.
- The node ID is a 8 bit identifier.
- A device can be spotted by a microcontroller only if the Home ID on the microcontroller is same as the one on the device. Hence the path to be chosen by the microcontroller to send a signal is through devices which have the same Home ID only.
- When two different networks are present they know each other are present but they are not interoperable.

Typical Network

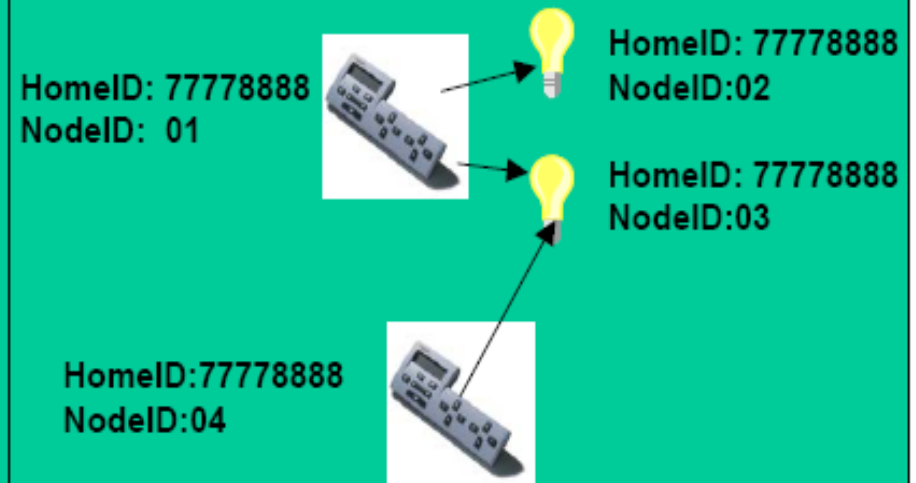
Home A

Home B


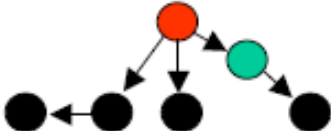




Light control system



Light control system



Z-Wave Node Archetypes

Archetypes	Illustration	Main Network System Functions
Controllers 		<ul style="list-style-type: none">• Have a complete topology map and can calculate routes• Can address all nodes in the network• Can act as repeater in the network• Can be portable or static in the network• Can have the added functionalities of Bridge and/or SUC + SIS
Routing Slaves 		<ul style="list-style-type: none">• Have a partial topology map and cannot calculate routes• Can address a subset of the network• Can act as a repeater in the network
Slaves 		<ul style="list-style-type: none">• Have no topology map and cannot calculate routes• Can only respond to requests from Controllers or Routing Slaves

Network Node Type Controllers

There are two types of controllers:

- Portable Controllers – Controllers those which can be moved around in a Network.
- Static Controllers – Controllers those which cannot be moved around in a Network.
- Bridge – Makes it possible for the Z-Wave to control up to 128 devices.

A “Virtual Node” is created by a Bridge which act as proxies for the non Z-Wave devices present on the other network which thereby can be controlled by the Z-Wave Network.

NOTE: A non Z-Wave Network can be imagined as a Bluetooth Network.

Network Node Types – Routing Slaves

- Routing Slave is a AC powered which is present in a fixed location. It can receive commands and replies from/to all nodes. It can send unsolicited routed frames to up to 5 other nodes.
- Enhanced Slave is the same as a routing slave but can support an external EEPROM.

Network Node Types - Slaves

- Slave – Must be in a fixed position and should be listening every time. It can receive commands and send reply using the same route.

Typical Node type controllers



A Portable
Controller



Static Controller



Bridge



Routing
Slave



Slave

Conclusion

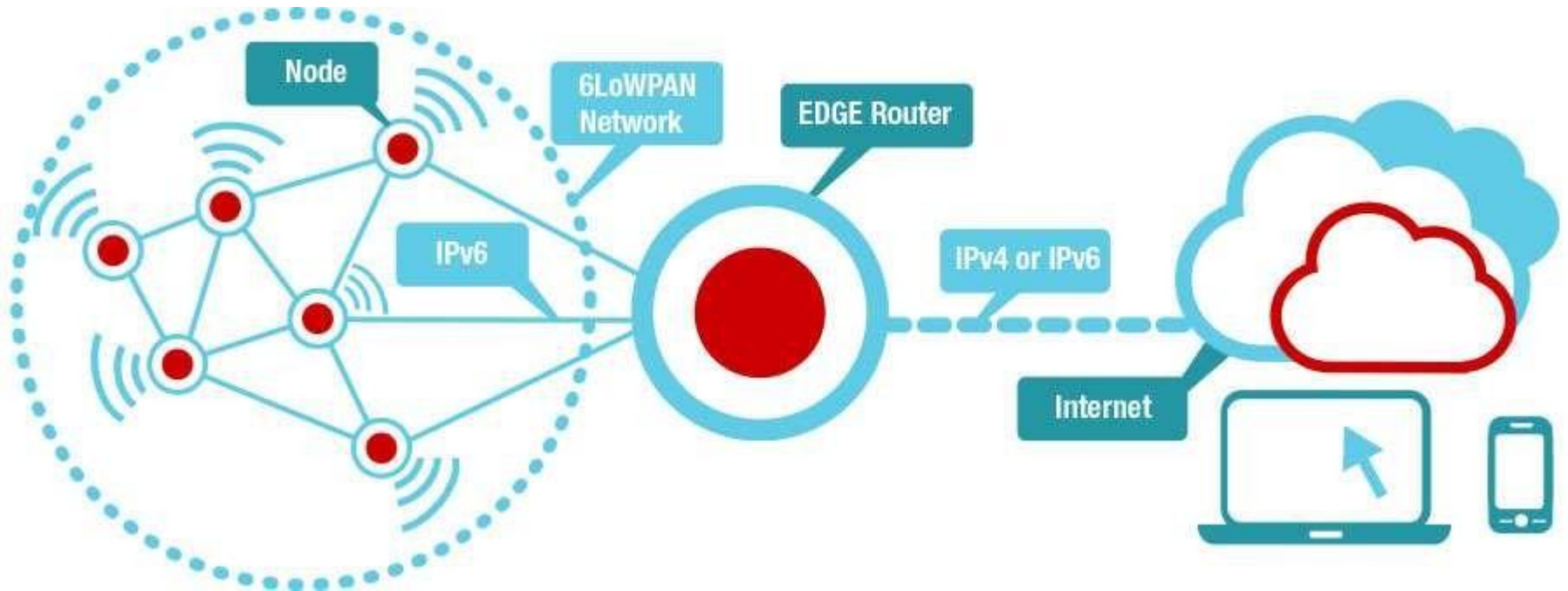
- The **Z-Wave protocol** is a wireless, radio frequency (RF) based communications technology designed particularly for control, monitoring and status reading of household applications. **Z-Wave** supports full mesh networks, enabling numerous **Z-Wave** devices to communicate with each other simultaneously.
- **Other Z-wave related protocols Z-wave plus and Z-wave LR**

6LoWPAN

*IPv6 LoW Power wireless
Personal Area Networks*

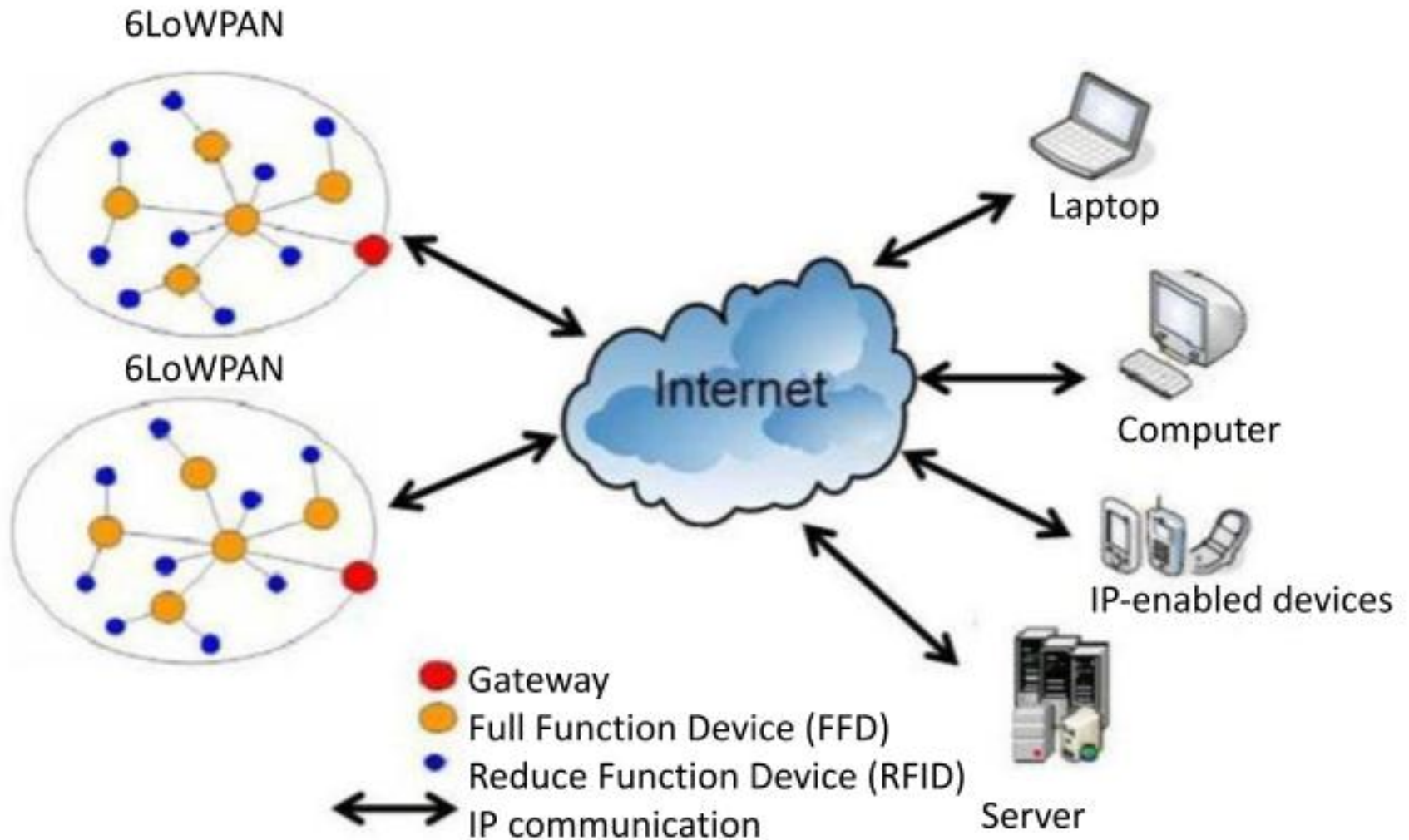
Internet Of Things and Sensors Network

The 6LoWPAN concept originated from the idea that "the Internet Protocol could and should be applied even to the smallest devices," and that low-power devices with limited processing capabilities should be able to participate in the Internet of Things.

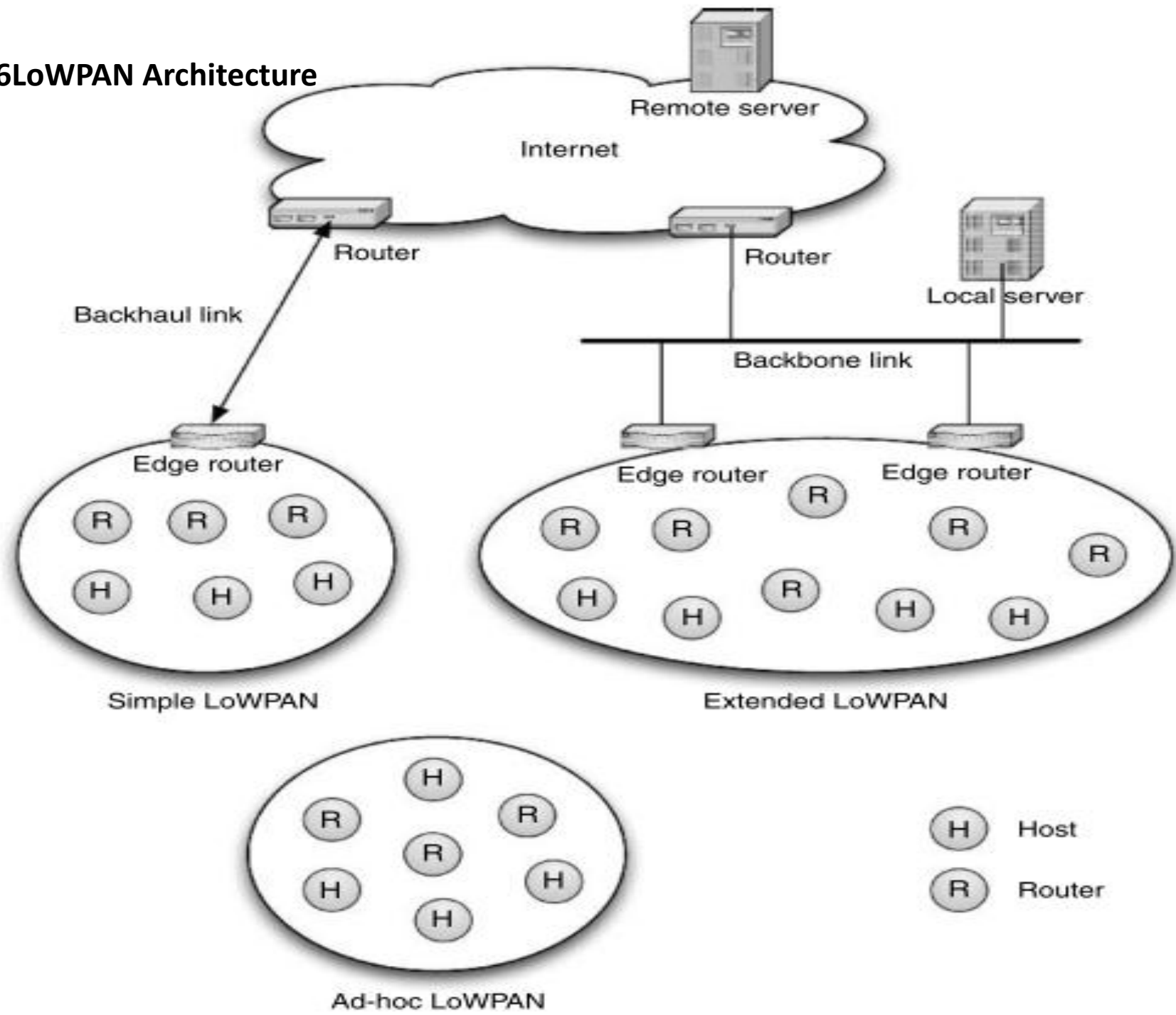


"The Internet Protocol could and should be applied even to the smallest devices"

6LoWPAN Architecture



6LoWPAN Architecture



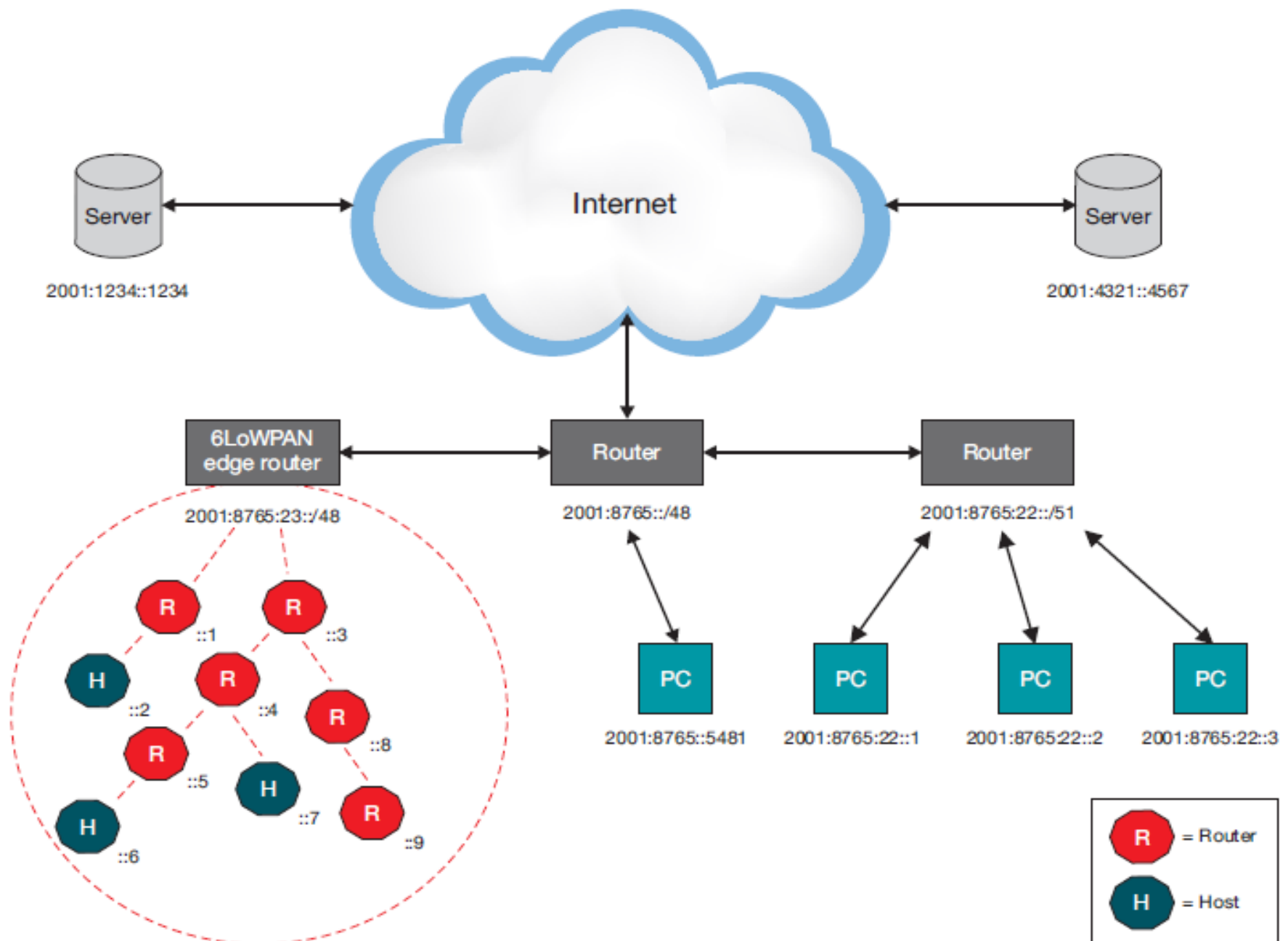


Figure 1. An example of an IPv6 network with a 6LoWPAN mesh network

6LoWPAN application areas

- *6LoWPAN provides the upper layer system for use with low power wireless communications for IoT and M2M, originally intended for 802.15.4*
- Compare with many low power WSNs and other forms of ad hoc wireless networks, uses IP in particular IPv6 to carry the data.
- The overall system is aimed at providing wireless internet connectivity at low data rates and with a low duty cycle. Some applications where 6LoWPAN is being used:
- **General Automation:** There are enormous opportunities for 6LoWPAN to be used in many different areas of automation.
- **Home automation:** There is a large market for home automation. By connecting using IPv6, it is possible to gain distinct advantages over other IoT systems. The Thread initiative has been set up to standardize on a protocol running over 6LoWPAN to enable home automation.
- **Smart Grid:** Smart grids enable smart meters and other devices to build a micro mesh network and they are able to send the data back to the grid operator's monitoring and billing system using the IPv6 backbone.
- **Industrial monitoring:** Automated factories and industrial plants provide a great opportunity for 6LoWPAN and using automation, can enable major savings to be made. The ability of 6LoWPAN to connect to the cloud opens up many different areas for data monitoring and analysis.

6LoWPAN basics

- The 6LoWPAN technology utilises IEEE 802.15.4 to provide the lower layers for this low power wireless network system. There are incompatibilities between IPv6 format and the formats allowed by IEEE 802.15.4. These differences are overcome within 6LoWPAN and this allows the system to be used as a layer over the basic 802.15.4.
- In order to send packet data, IPv6 over 6LoWPAN, it is necessary to have a method of converting the packet data into a format that can be handled by the IEEE 802.15.4 lower layer system.
- IPv6 requires the **maximum transmission unit (MTU)** to be at least **1280 bytes** in length. This is considerably longer than the IEEE802.15.4's standard packet size of 127 octets which was set to keep transmissions short and thereby reduce power consumption.
- To overcome the address resolution issue, IPv6 nodes are given 128 bit addresses in a hierarchical manner. The IEEE 802.15.4 devices may use either of IEEE 64 bit extended addresses or 16 bit addresses that are unique within a PAN after devices have associated. PAN-ID used for a group of physically co-located IEEE802.15.4 devices.

- **6LoWPAN security**

- It is anticipated that the Internet of Things, IoT will offer hackers a huge opportunity to take control of poorly secured devices and also use them to help attack other networks and devices.
- Accordingly security is a major issue for any standard like 6LoWPAN, and it uses **AES-128** link layer security which is defined in IEEE 802.15.4. This provides link authentication and encryption.
- Further security is provided by the transport layer security mechanisms that are also included. This is defined in RFC 5246 and runs over TCP.
- For systems where UDP is used the transport layer protocol defined under RFC 6347 can be used, although this may require some specific hardware requirements.

- **6LoWPAN interoperability**

- One key issue of any standard is that of interoperability. It is vital that equipment from different manufacturers operates together.
- When testing for interoperability, it is necessary to ensure that all layers of the OSI stack are compatible. To ensure that this can be achieved there several different specifications that are applicable.
- Any item can be checked to conform it meets the standard, and also directly tested for interoperability.

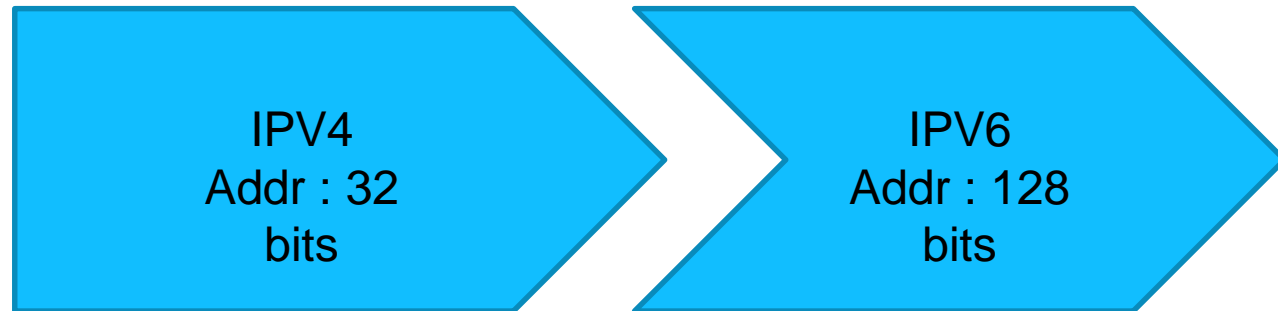
6Lowpan : what does it mean?

6

Low Power
Low Datarate
Low Cost



Wireless Personal Area Networks



Why IPv6?	IPv4	IPv6
IPv6 has more addresses	4.3 billion addresses (32bits)	340 trillion trillion trillion addresses (128bits)
IPv6 networks are easier and cheaper to manage	Networks must be configured manually or with DHCP. IPv4 has had many overlays to handle Internet growth, which demand increasing maintenance efforts.	IPv6 networks provide autoconfiguration capabilities. They are simpler, flatter and more manageable for large installations.
IPv6 restores end-to-end transparency	Widespread use of NAT devices means that a single NAT address can mask thousands of non-routable addresses, making end-to-end integrity unachievable.	Direct addressing is possible due to vast address space – the need for network address translation devices is effectively eliminated.
IPv6 has improved security features	Security is dependent on applications – IPv4 was not designed with security in mind.	IPSEC is built into the IPv6 protocol, usable with a suitable key infrastructure.
IPv6 has improved mobility capabilities	Relatively constrained network topologies restrict mobility and interoperability capabilities in the IPv4 Internet.	IPv6 provides interoperability and mobility capabilities which are already widely embedded in network devices.
IPv6 encourages innovation	IPv4 was designed as a transport and communications medium, and increasingly any work on IPv4 is to find ways around the constraints.	Given the numbers of addresses, scalability and flexibility of IPv6, its potential for triggering innovation and assisting collaboration is unbounded.

6LoWPAN Layers and Protocol Stack

<i>Application</i>	MQTT	CoAP
<i>Transport</i>	TCP	UDP
<i>Network</i>	IPv6	
<i>Adaptation</i>	6LoWPAN	
<i>Link</i> <i>Physical</i>	IEEE 802.15.4	Bluetooth Low Energy

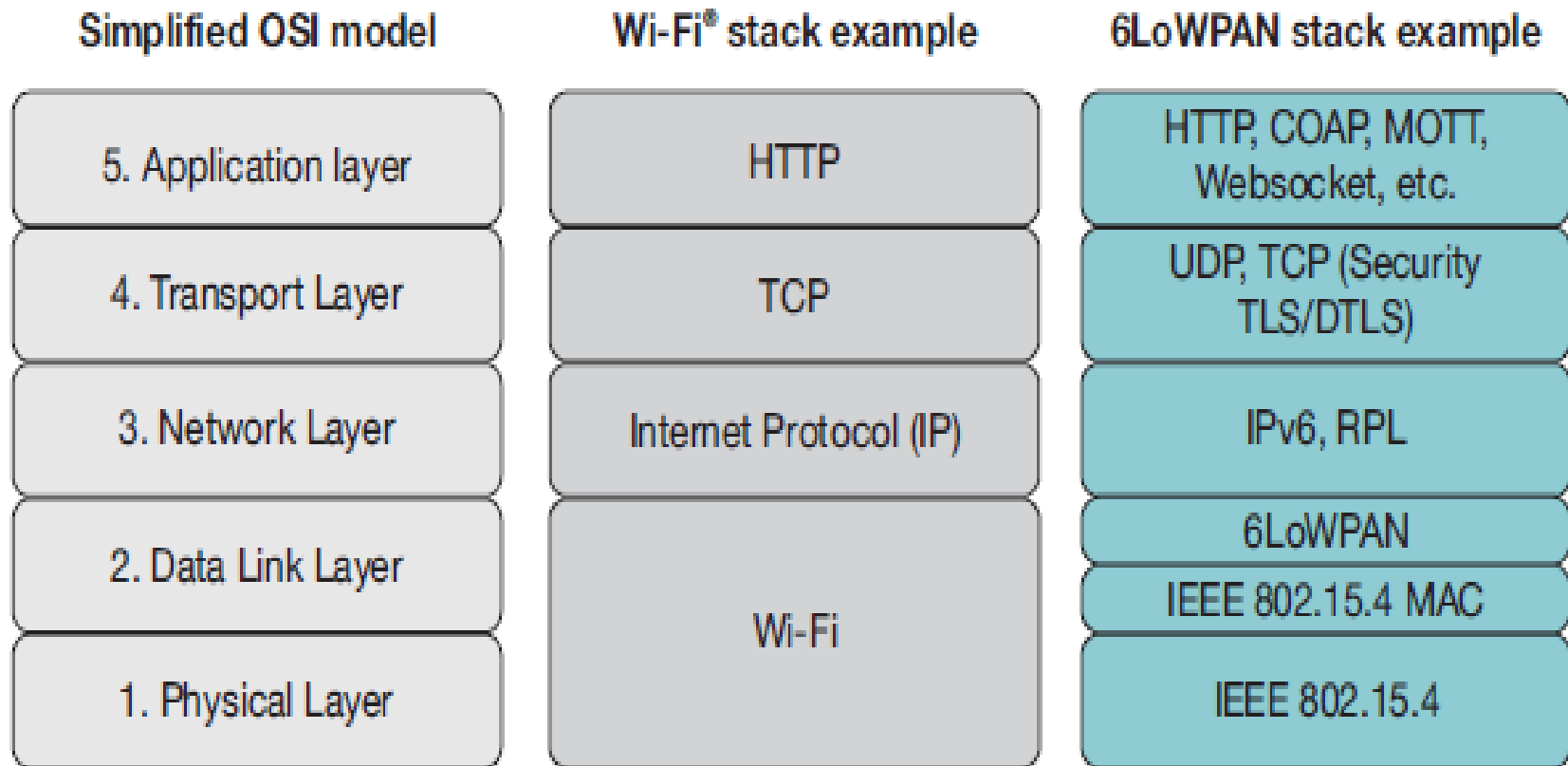
TCP/IP vs. 6LoWPAN Protocol Stacks

TCP/IP Protocol Stack
HTTP, FTP, DNS, SMTP
TCP, UDP, ICMP
IP
Ethernet, PPP
Ethernet

Layers
Application
Transport
Network
Data Link
Physical

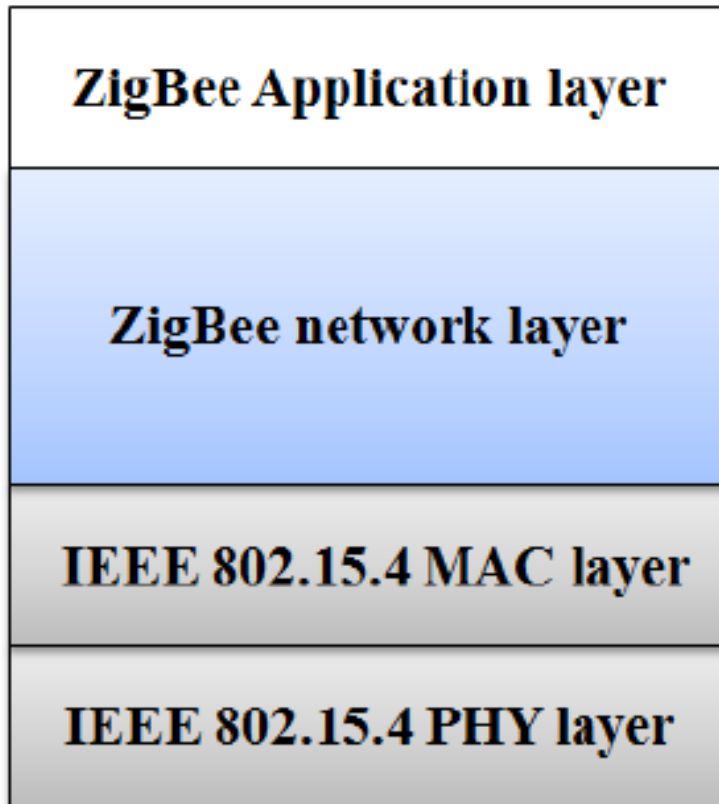
6LoWPAN Protocol Stack
COAP, MQTT
UDP, ICMP, DTLS
IPv6 with <u>LoWPAN</u>
IEEE 802.15.4 MAC
IEEE 802.15.4 PHY

WiFi vs. 6LoWPAN Protocol Stacks

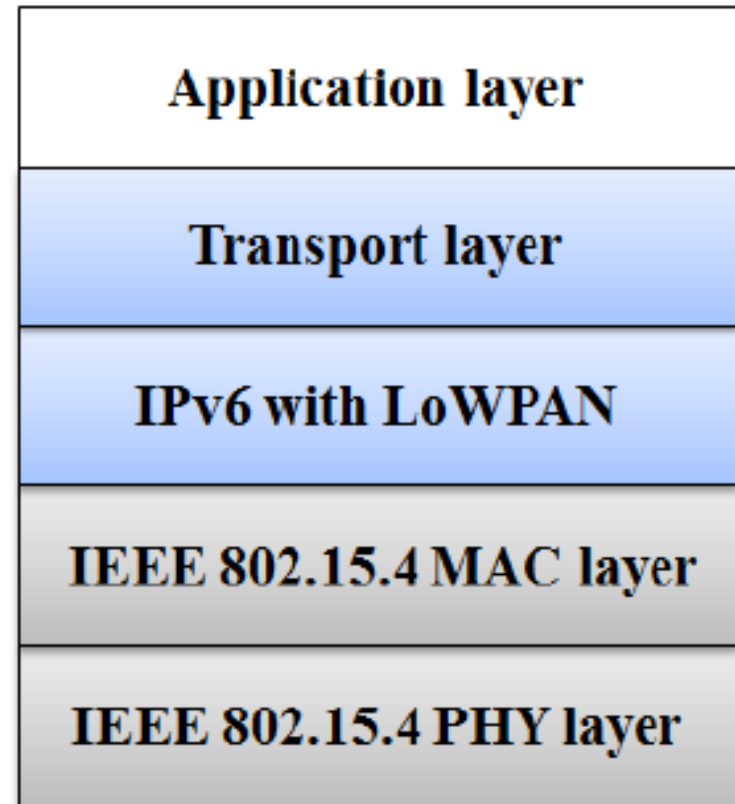


The OSI model, a Wi-Fi® stack example and the 6LoWPAN stack

Zigbee Vs. 6LoWPAN Protocol Stacks



ZigBee



6LoWPAN

6LowPAN : IOT solution

IPV6 packet in 802.15.4 container

IPV6 packet
1280 Bytes



802.15.4 packet
127 Bytes



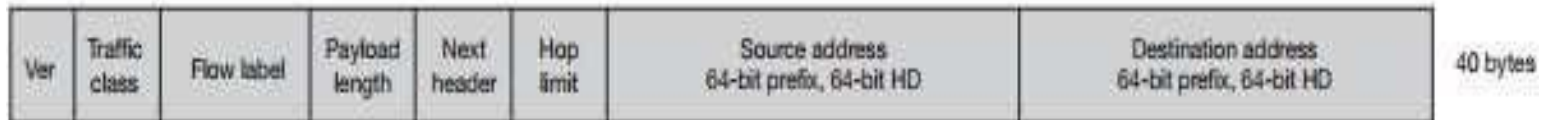
6LowPAN : IOT solution

Optimizing the transmission of IPv6 packets over low-power and lossy networks

IPv6 Header
40 Bytes

802.15.4 Packet
127 Bytes

IPv6 header

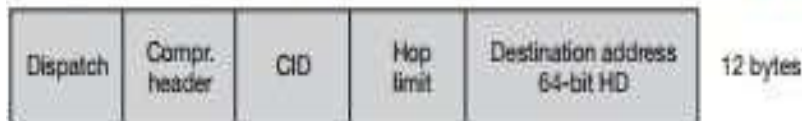


Header compression

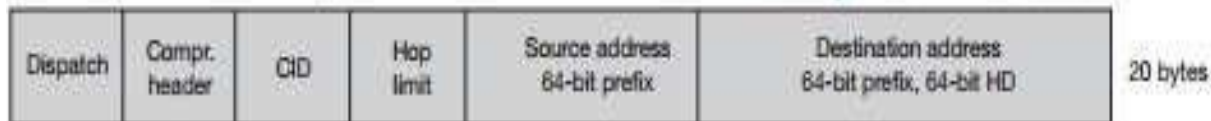
1. Compressed header, FE80::CAFE:00FF:FE00:0100 → FE80::CAFE:00FF:FE00:0200



2. Compressed header, 2001::DEC4:E3A1:FE24:9600 → 2001::4455:84C6:39BB:A2DD

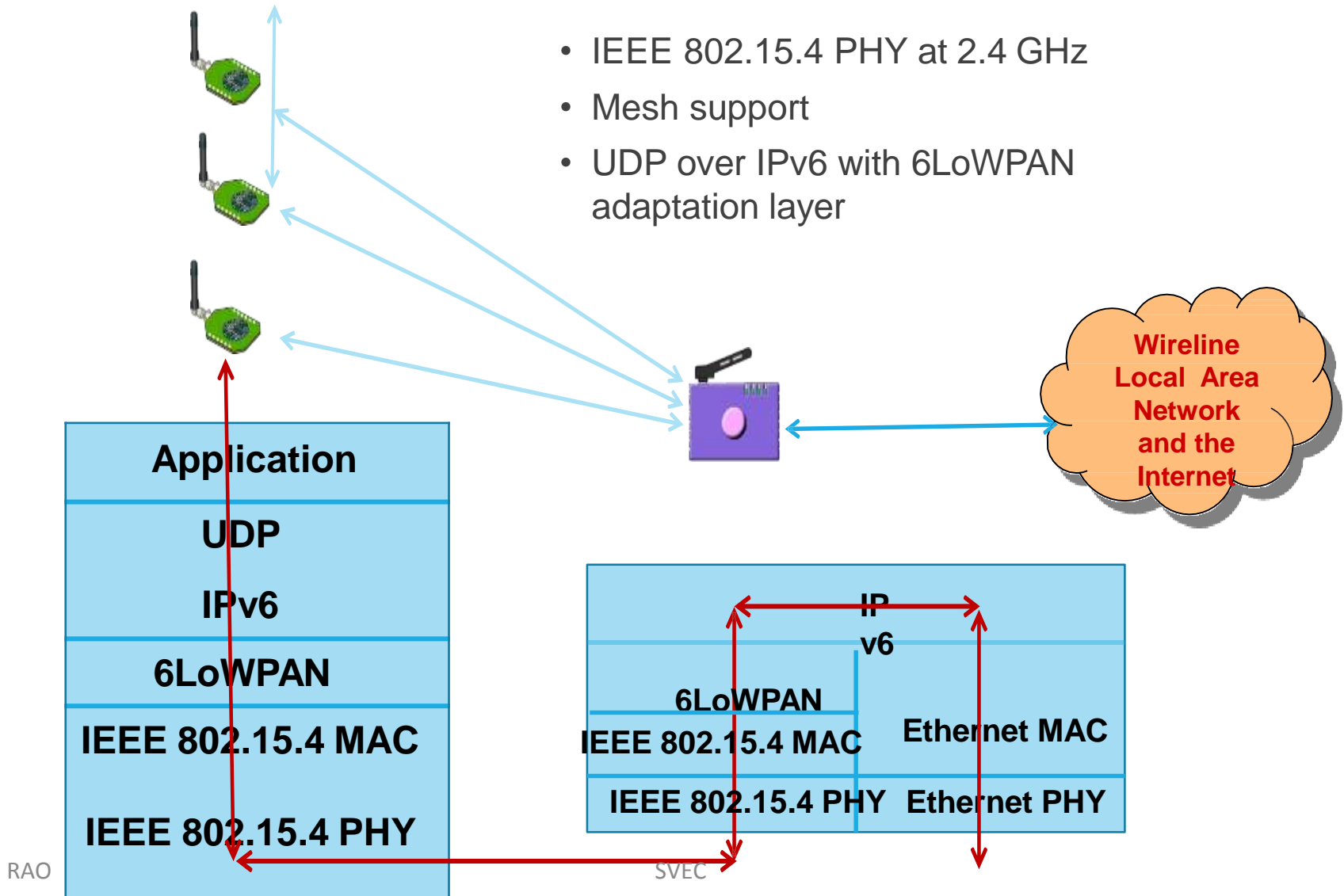


3. Compressed header, 2001::DEC4:E3A1:FE24:9600 → 2001::4455:84C6:39BB:A2DD



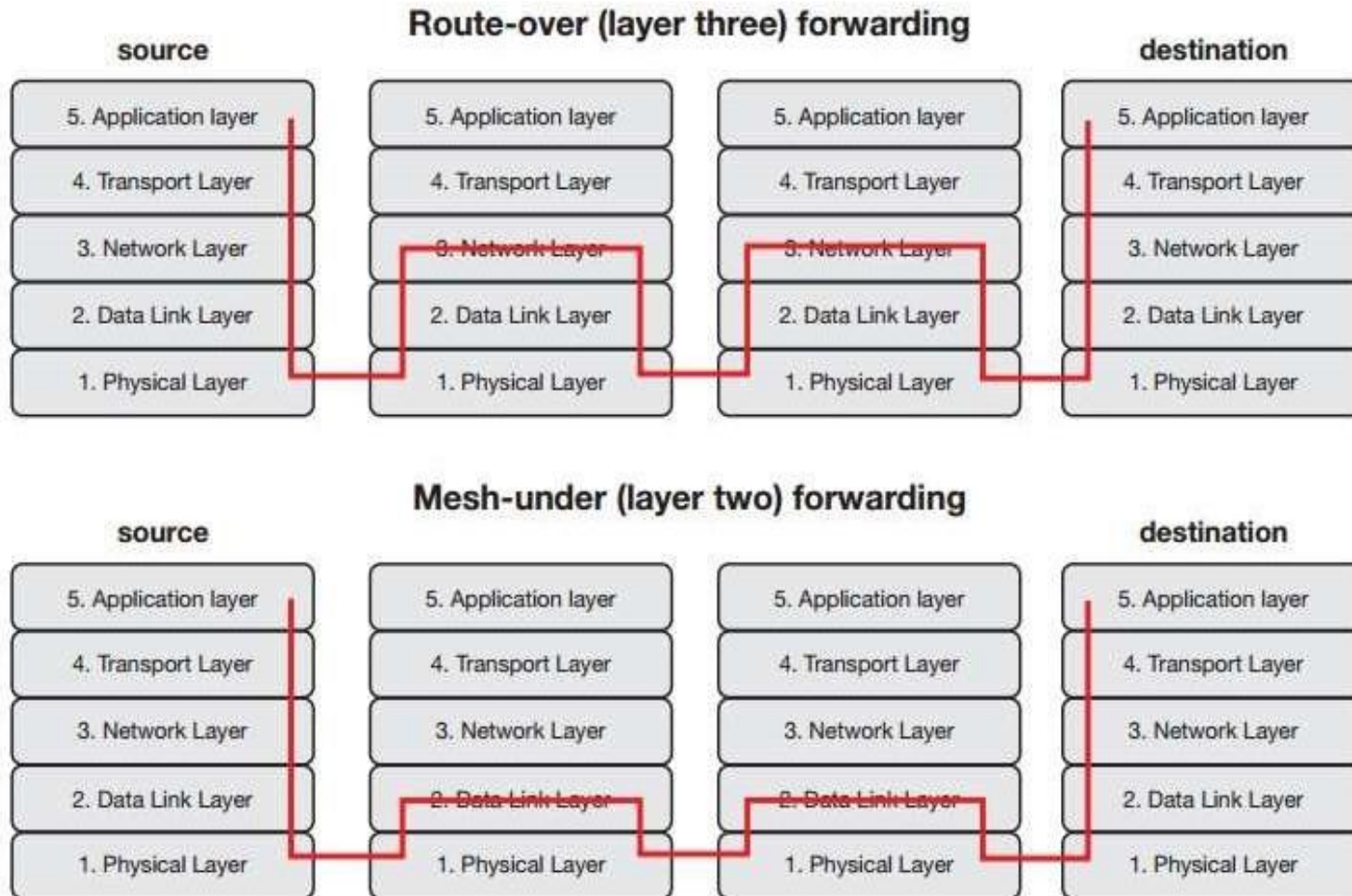
The Network Protocol Stack

- IEEE 802.15.4 PHY at 2.4 GHz
- Mesh support
- UDP over IPv6 with 6LoWPAN adaptation layer



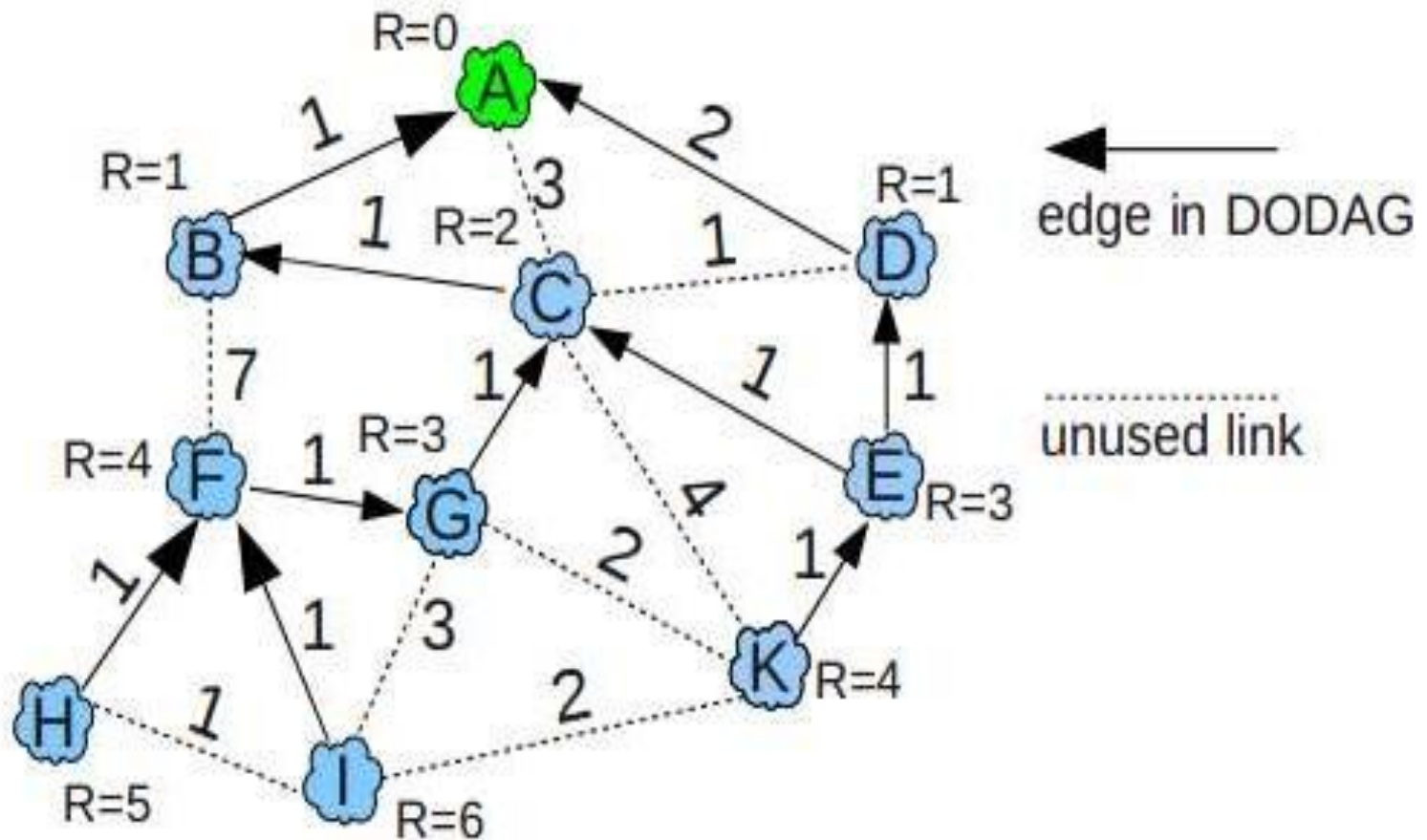
6LowPAN : IOT solution

Routing : Mesh Over (Qos) and Mesh Under (Low latency)



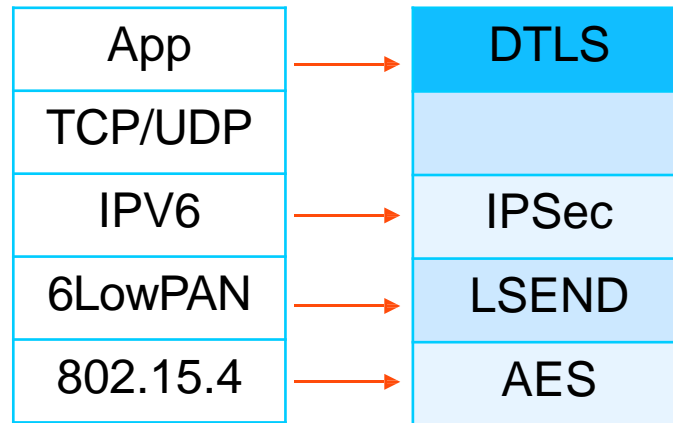
6LowPAN : IOT solution

Routing: Build RPL tree (DODAG) and communicate through nodes



6LowPAN : IOT solution

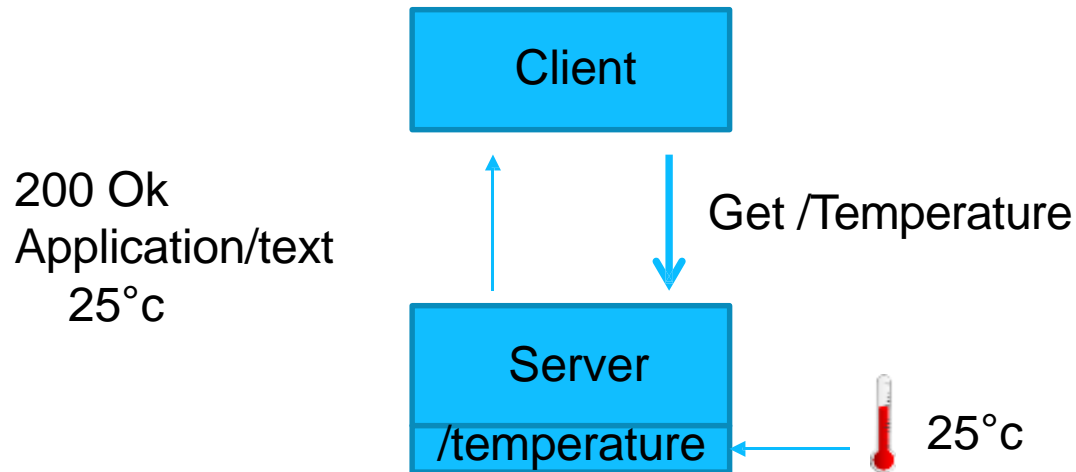
Security: Add security mechanism on network layer and application layer



Security Mecanism

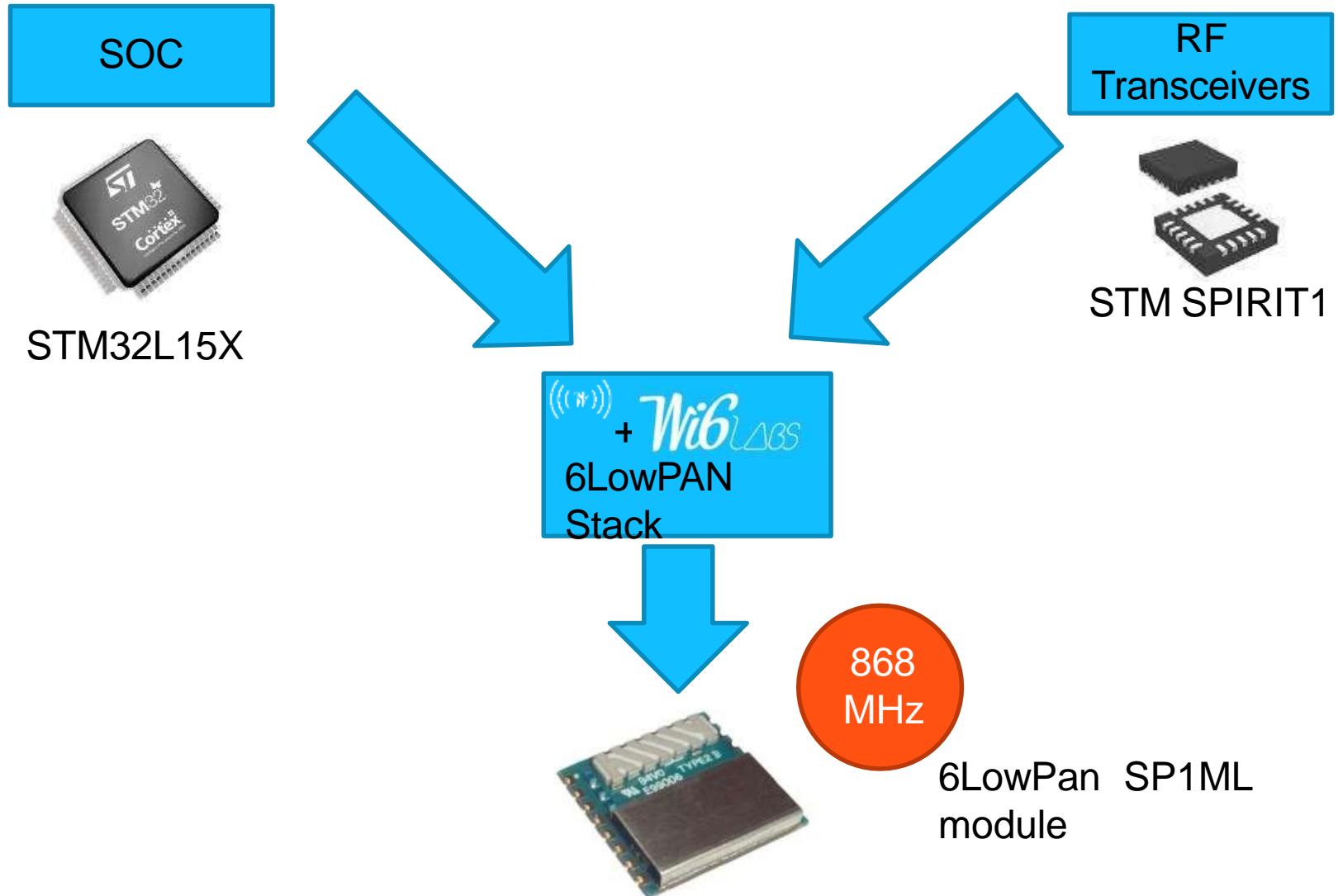
6LowPAN : IOT solution

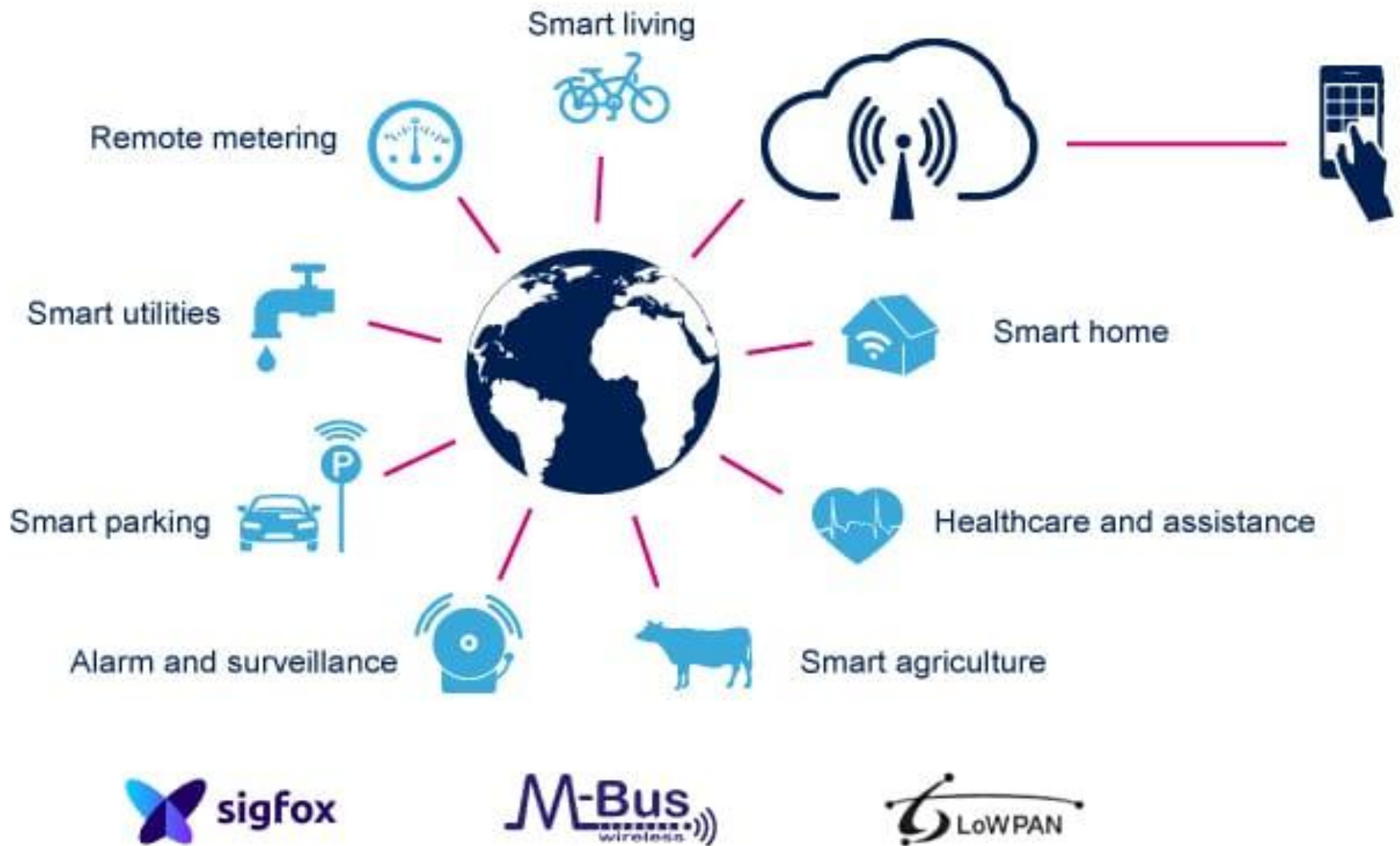
HTTP is chatty : REST request – COAP (*Constrained Application Protocol*)



Rest request example

6LowPAN implementation example





OTHER APPLICATIONS: Home building automation, Health care, Industry Automation & Logistics, Vehicular Automation, Environmental monitoring

RPL - Routing Protocol for Low power and lossy Networks

Overview

- Introduction
- Routing Requirements
- RPL instance and DODAG
- RPL Ranks
- Route construction
- Objective Function and Control messages

Introduction

- Low Power and Lossy Networks (LLN) are resource constrained
- Routers are usually limited in terms of processing power, battery and memory, and their interconnects are characterised by unstable links with high loss rates, low data rates and low packet delivery rates
- The traffic patterns could be P2P or P2MP or MP2P
- Lossy means the packet drop rate will be high.

Introduction

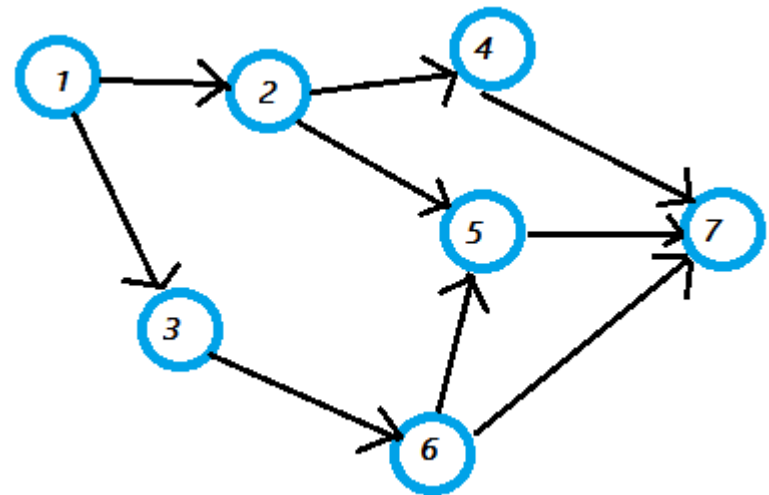
- RPL is a distance vector routing protocol
- RPL mainly targets collection-based networks, where nodes periodically send measurements to a collection point.
- The protocol was designed to be highly adaptive to network conditions and to provide alternate routes, whenever default routes are inaccessible.
- RPL provides a mechanism to disseminate information over the dynamically formed network topology
- Contains thousands of nodes...

RPL topology

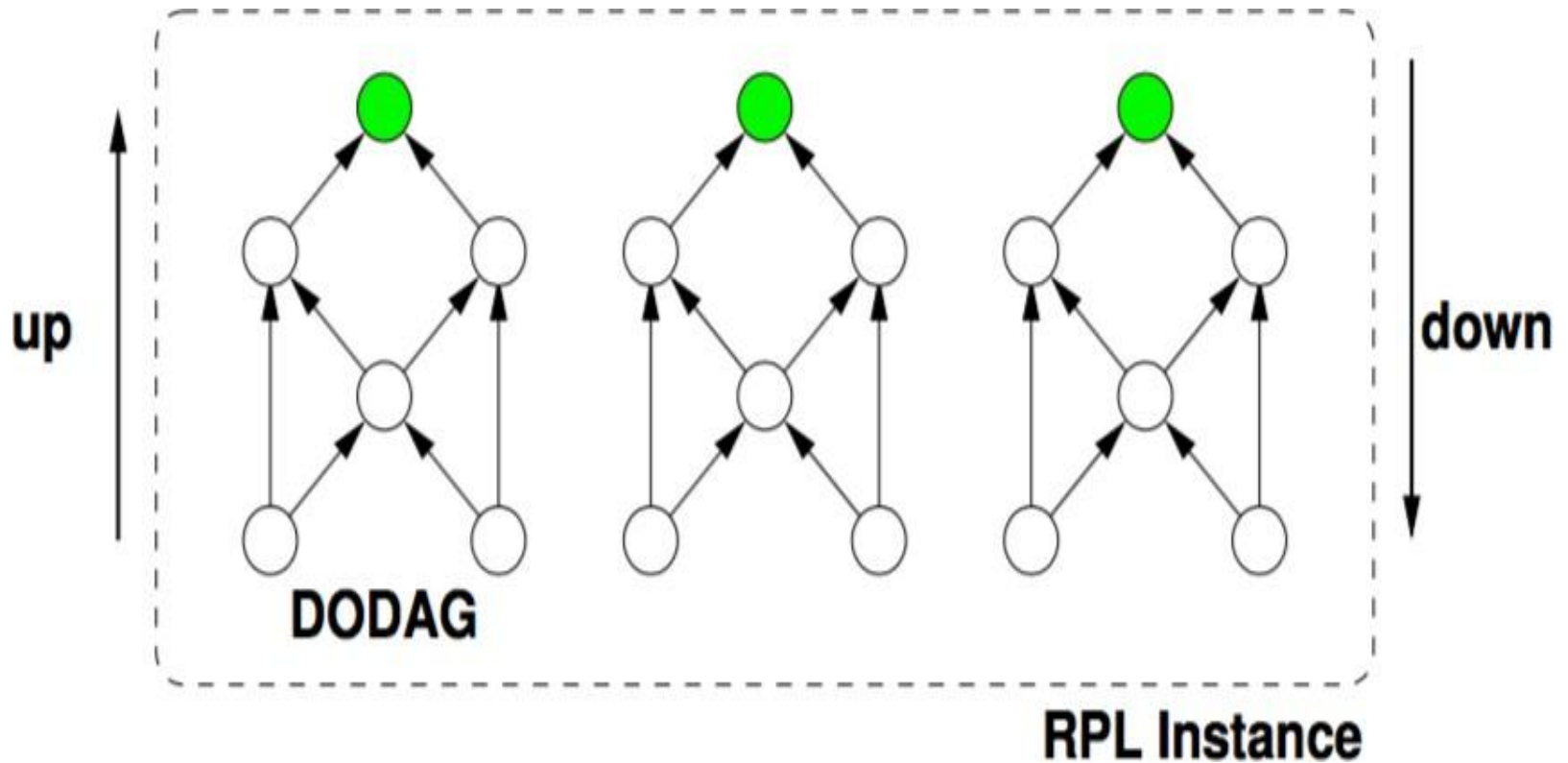
- **DODAG (Destination Oriented Directed Acyclic Graphs)**
 - A DODAG is a DAG rooted at a single destination. The DODAG root has no outgoing edges. A DODAG is uniquely identified by a combination of RPL Instance ID and DODAG ID.
- **Rank**
 - A nodes Rank defines the nodes individual position relative to other nodes with respect to a DODAG root. Rank strictly increases in the Down¹ direction and strictly decreases in the Up² direction.
- **DODAG Root**
 - The DODAG root is the DAG root of the DODAG. The DODAG root may act as a *border router* for the DODAG, and aggregate routes in the DODAG and may redistribute DODAG routes into other routing protocols

Vertices and Edges

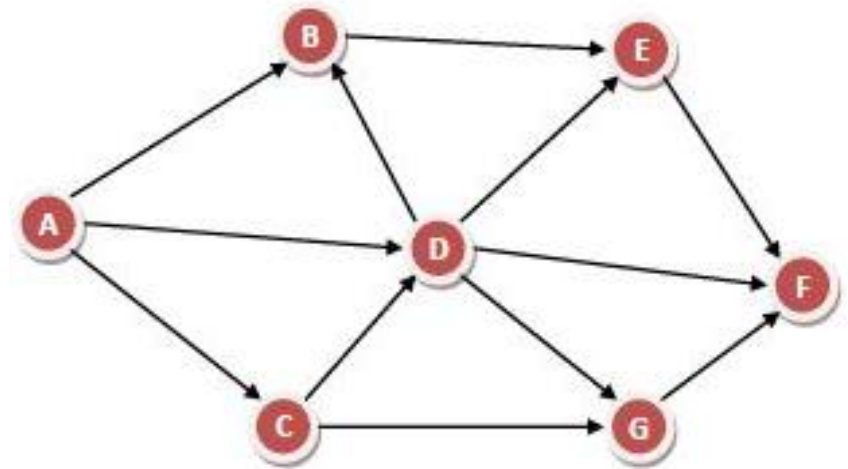
- Integer = the set for the Vertices.
- **Vertices set** = $\{1,2,3,4,5,6,7\}$.
- **Edge set** = $\{(1,2), (1,3), (2,4), (2,5), (3,6), (4,7), (5,7), (6,7)\}$.



RPL topology

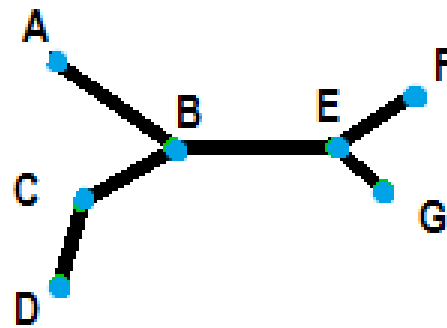


DAG(Directed Acyclic Graphs)



- In graph theory, a graph refers to a set of vertices which are connected by lines called edges. In a directed graph or a digraph, each edge is associated with a direction from a start vertex to an end vertex. If we traverse along the direction of the edges and we find that no closed loops are formed along any path, we say that there are no directed cycles. The graph formed is a directed acyclic graph.
- A DAG is always topologically ordered, i.e. for each edge in the graph, the start vertex of the edge occurs earlier in the sequence than the ending vertex of the edge.
- In the above directed graph, if we find the paths from any node, say u , we will never find a path that come back to u .

DAG

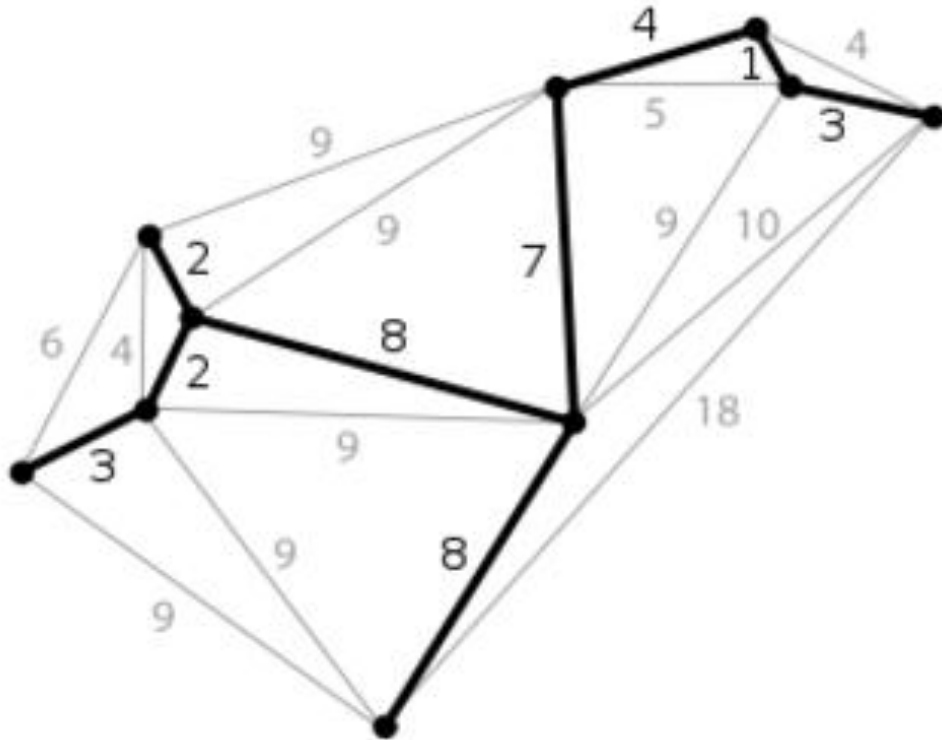


*A tree with nodes
A B C D E F and G.*

- Directed acyclic graphs (DAGs) are used to model probabilities, connectivity, and causality. A “graph” in this sense means a structure made from nodes and edges.
- **Nodes** are usually denoted by circles or ovals (although technically they can be any shape of your choosing).
- **Edges** are the connections between the nodes. An edge connects two nodes. They are usually represented by lines, or lines with arrows.
- DAGs are based on basic acyclic graphs.

What is an Acyclic Graph?

- An acyclic graph is a graph without cycles (a cycle is a complete circuit). When following the graph from node to node, you will never visit the same node twice.



This graph (the thick black line) is acyclic, as it has no cycles (complete circuits).

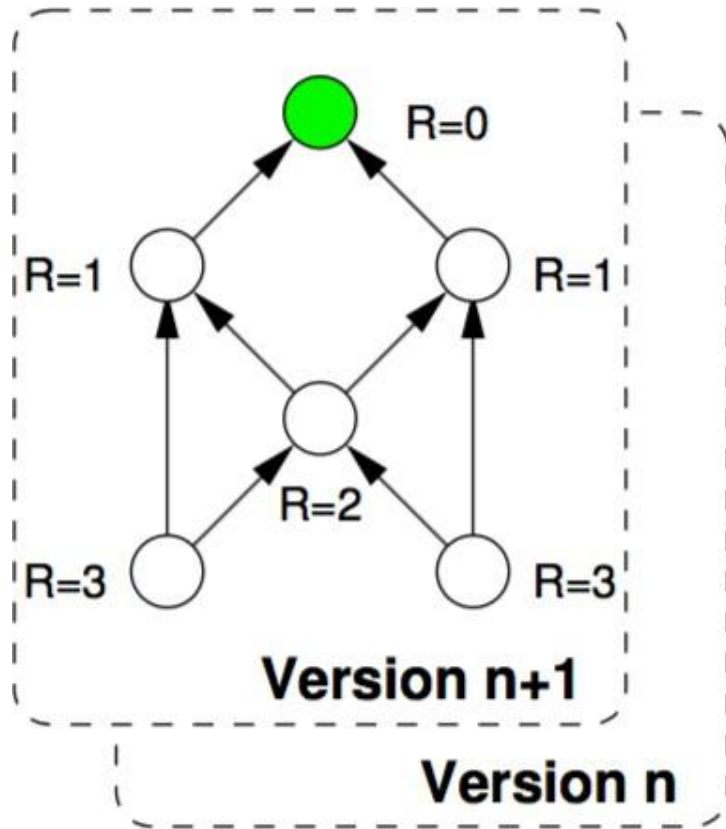
RPL topology

- Upward path is so common (mp2p)
- Downward path is optional mainly for p2p and p2mp
- An RPL Instance consists of multiple Destination Oriented Directed Acyclic Graphs (DODAGs). Traffic moves either up towards the DODAG root or down towards the DODAG leafs

RPL instance

- DODAGS are disjoint (no shared nodes)
- Link properties: (reliability, latency, . . .)\Node properties: (powered or not, . . .)
- RPL Instance has an optimization objective
- Multiple RPL Instances with different optimization objectives can coexist

RPL Rank



- A node's Rank defines the node's individual position relative to other nodes with respect to a DODAG root. The scope of Rank is a DODAG Version.

Forwarding and routing

- Up routes towards nodes of decreasing rank (parents),
Down routes towards nodes of increasing rank
- Nodes inform parents of their presence and reachability to descendants.
- All routes go upwards and/or downwards along a DODAG
- When going up, always forward to lower rank when possible, may forward to sibling if no lower rank exists
- When going down, forward based on down routes

RPL Control Messages

- RPL DODAG formation and maintained by the control messages. The control messages are:
- **DODAG Information Solicitation (DIS)** : .DIS message broadcasted by new nodes joining network, may be used to solicit a DODAG Information Object from a RPL node.
- **DODAG Information Object (DIO)** : Carries information that allows a node to discover a RPL Instance, learn its configuration parameters, select a DODAG parent set, and maintain the DODAG.
- **DODAG Advertisement Object (DAO)** : Downward route are maintained by DAO messages
- **DAO-ACK** - DAO Acknowledgement (unicast packet by a DAO recipient)
- **CC** - Consistency Check (Checking for consistency in the messages)

Constrained Application Protocol (CoAP)

- **Constrained Application Protocol (CoAP)** is a specialized web transfer **protocol** for use with **constrained** nodes and **constrained** networks in the Internet of Things. **CoAP** is designed to enable simple, **constrained** devices to join the **IoT** even through **constrained** networks with low bandwidth and low availability.
- CoAP has the following main features:
- Constrained web protocol fulfilling M2M requirements.
- UDP binding with optional reliability supporting unicast and multicast requests.
- Asynchronous message exchanges.
- Low header overhead and parsing complexity.
- URI and Content-type support.
- Simple proxy and caching capabilities.
- A stateless HTTP mapping, allowing proxies to be built providing access to CoAP resources via HTTP in a uniform way or for HTTP simple interfaces to be realized alternatively over CoAP.
- Security binding to Datagram Transport Layer Security (DTLS).

Background

: to enable web-based services in
strained wireless networks

8 bit micro-controllers

limited memory

low-power networks

Problem: WEB solutions are hardly applicable

Solution: re-design web-based services for
strained networks → COAP

Application layer	HTTP, CoAP, EBHTTP, LTP, SNMP, IPfix, DNS, NTP, SSH, DLMS, COSEM, DNP, MODBUS
Network/Communication layer	IPv6/IPv4, RPL, TCP/UDP, uIP, SLIP, 6LoWPAN,
PHY/MAC layer	IEEE 802.11 Series, 802.15 Series, 802.3, 802.16, WirelessHART, Z-WAVE, UWB, IrDA, PLC, LonWorks, KNX

Table 1 protocols in different layers

Table 2 CoAP features

Constrained web protocol fulfilling M2M requirements

Security binding to Datagram Transport Layer Security (DTLS)

Asynchronous message exchanges

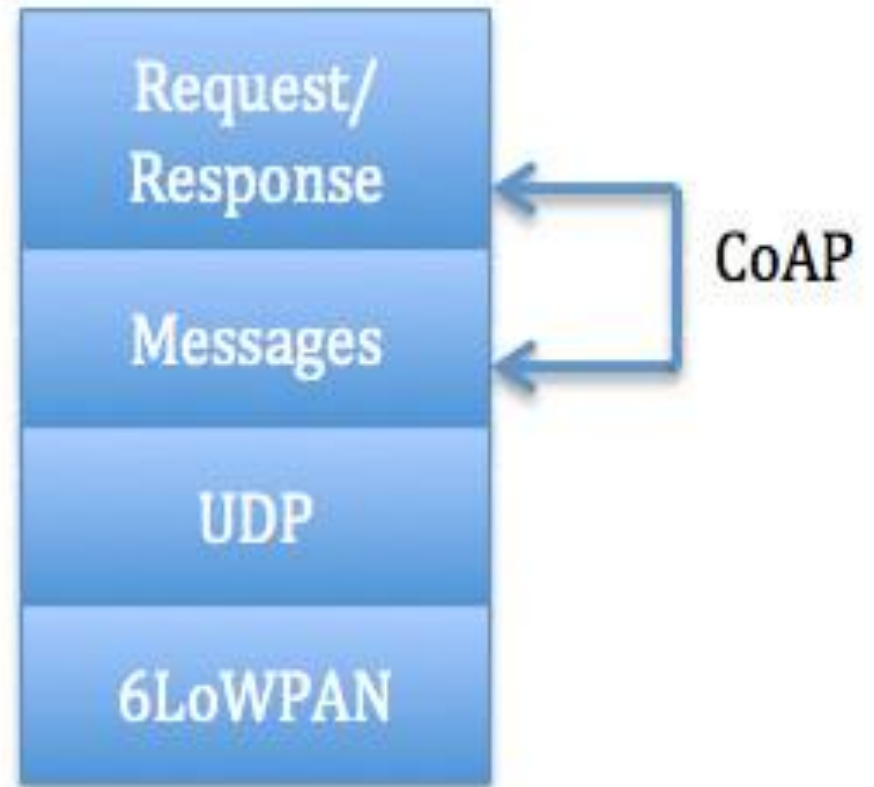
Low header overhead and parsing complexity.

URI and Content-type support.

Simple proxy and caching capabilities

UDP binding with optional reliability supporting unicast and multicast requests.

A stateless HTTP mapping, allowing proxies to be built providing access to CoAP resources via HTTP in a uniform way or for HTTP simple interfaces to be realized alternatively over CoAP.



HTTP and CoAP protocol stacks



CoAP Structure Model

<https://www.cse.wustl.edu/~jain/cse574-14/ftp/coap/index.html>

Table 3 Message Format

0		1								2								3													
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Ver	T	OC		Code								MessageID																			
Token (if any, TKL bytes)...																															
Options (if any)...																															
Payload (if any)...																															

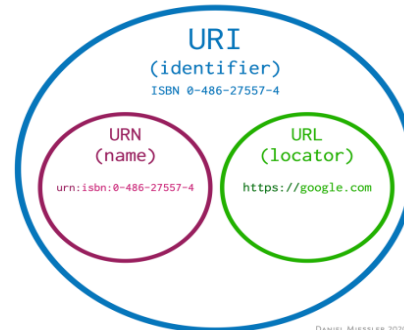
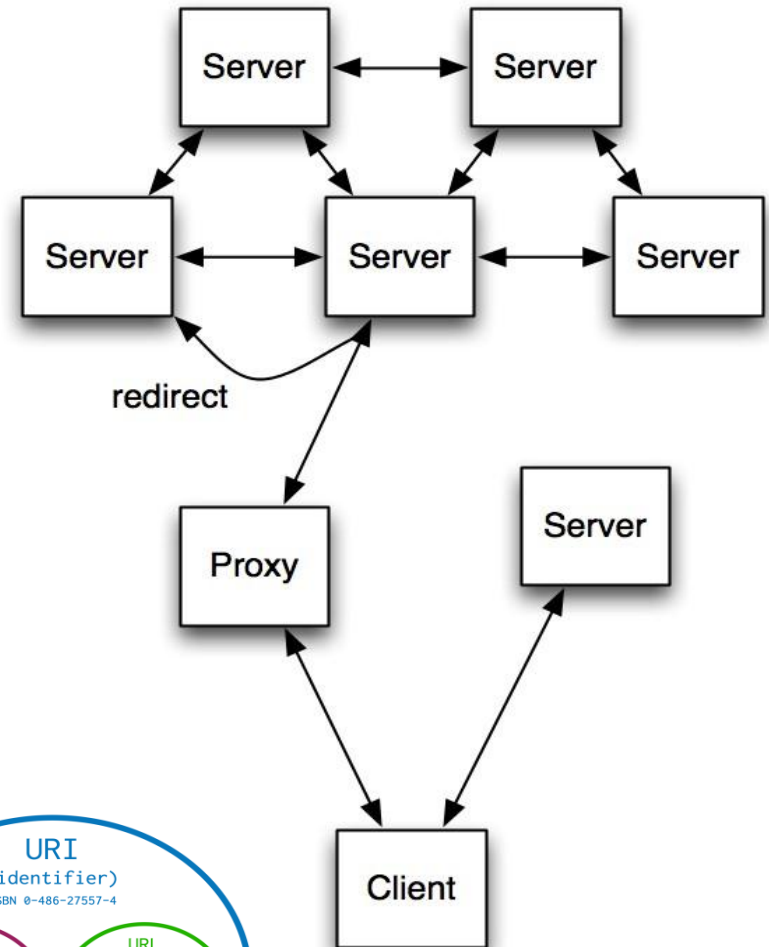
CoAP is based on the exchange of compact messages that, by default, are transmitted over UDP (i.e. each CoAP message occupies the data section of one UDP datagram). Message of CoAP uses simple binary format. Message= fixed-size 4-byte header plus a variable-length Token plus a sequence of CoAP options plus payload.

How Does the Web Work?

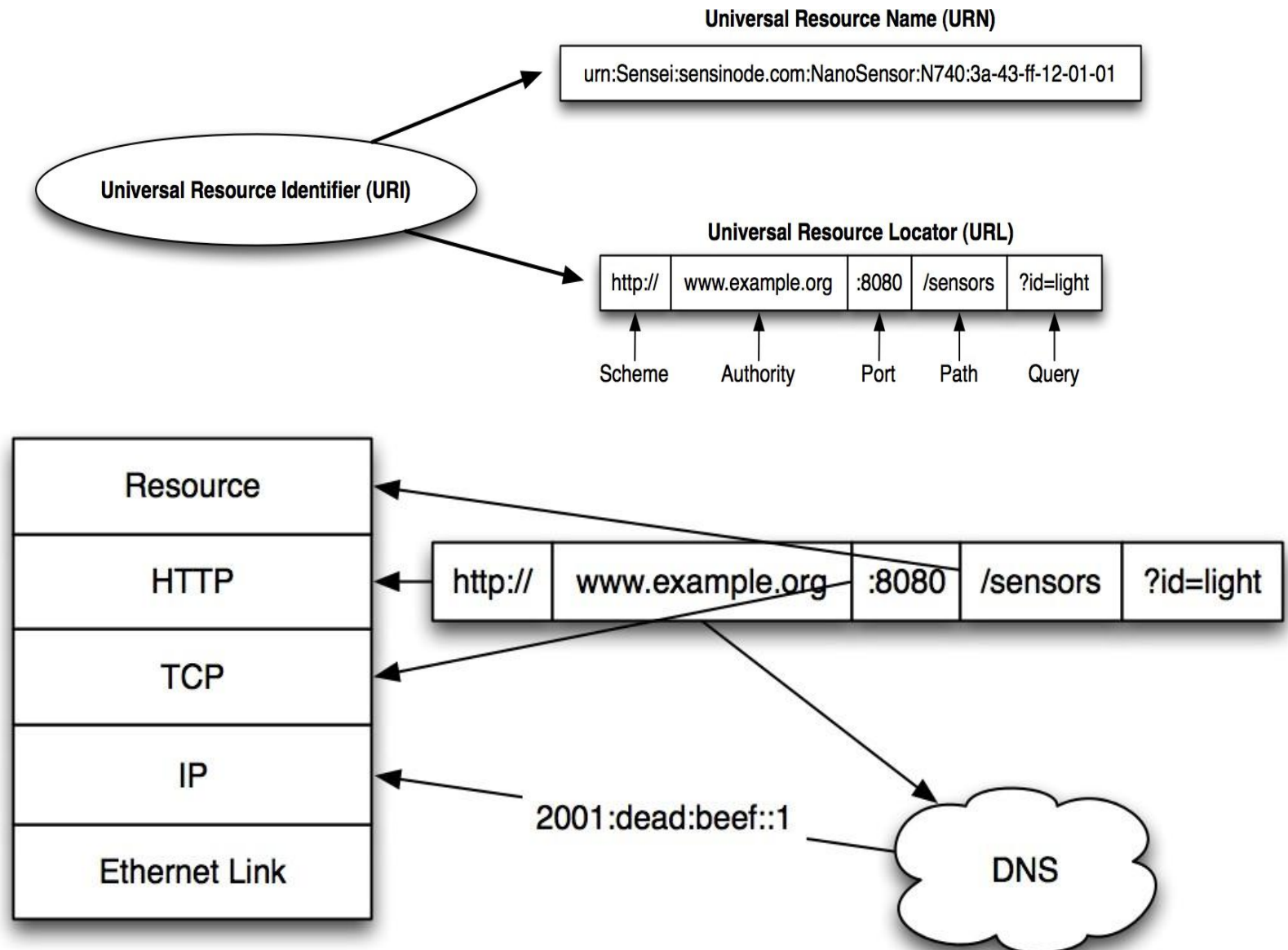
Resources in the Web are:

- managed by servers
- identified by URIs
- accessed synchronously by clients through request/response paradigms

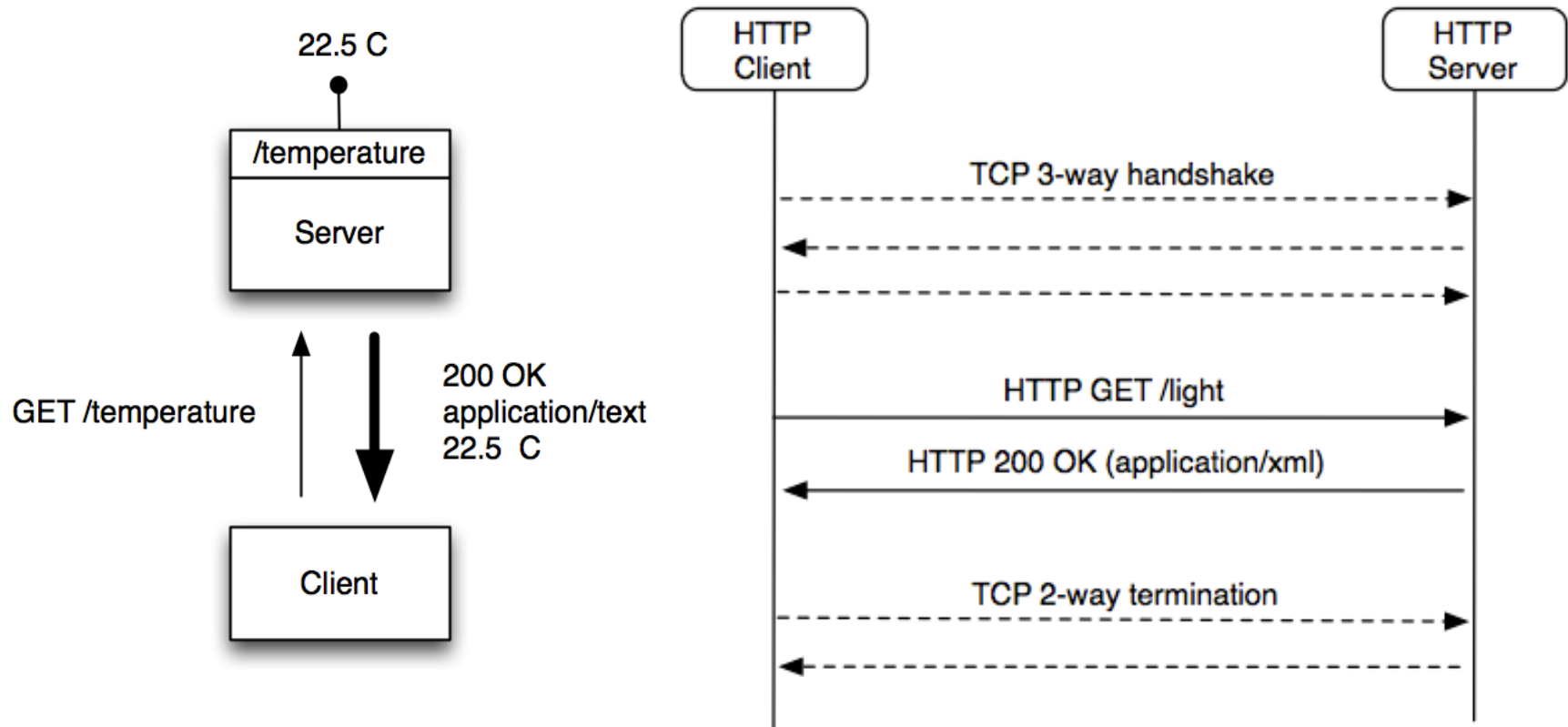
In a word, Representational State Transfer (REST)



URL Resolution

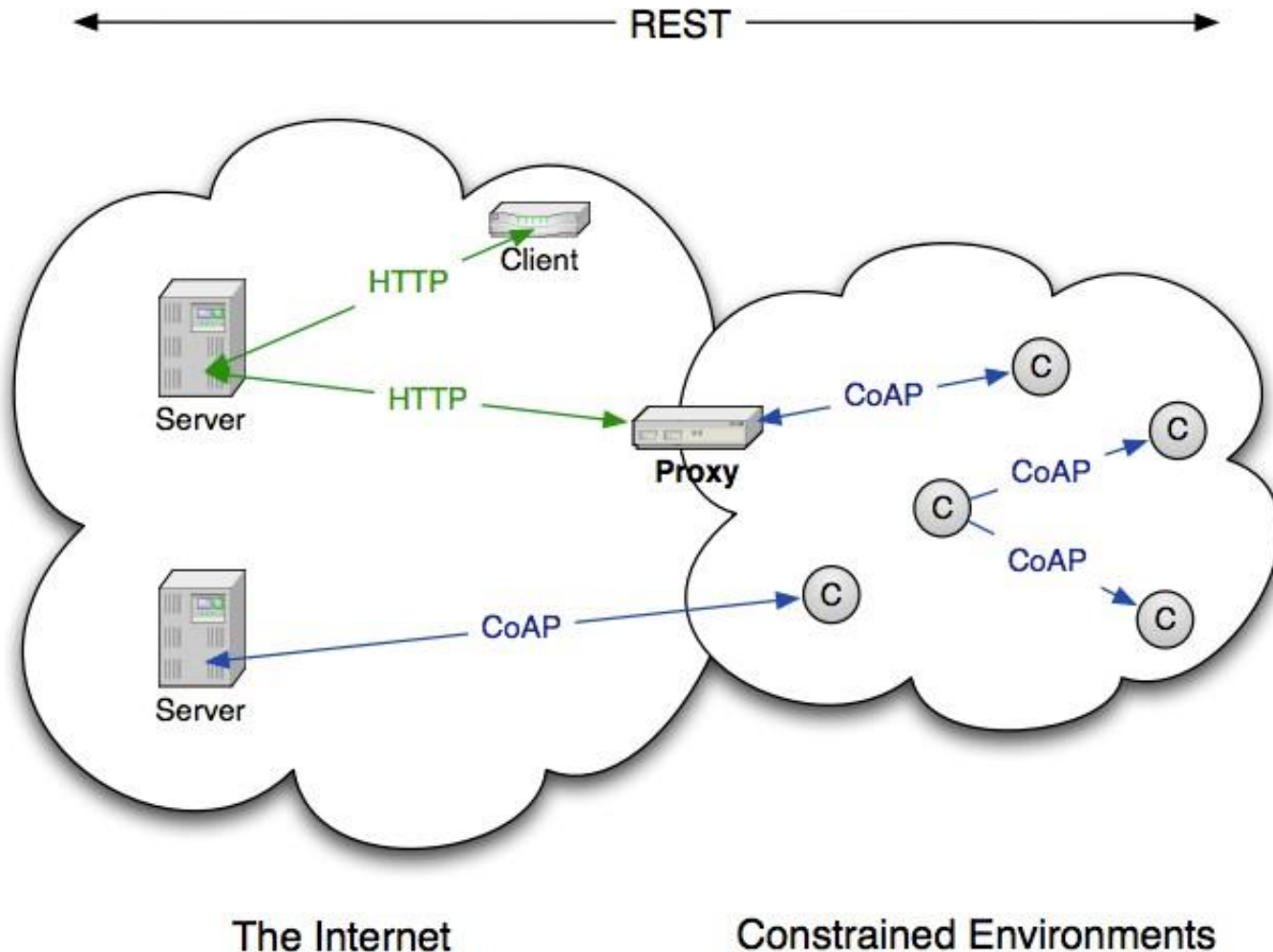


Request/Response Transaction

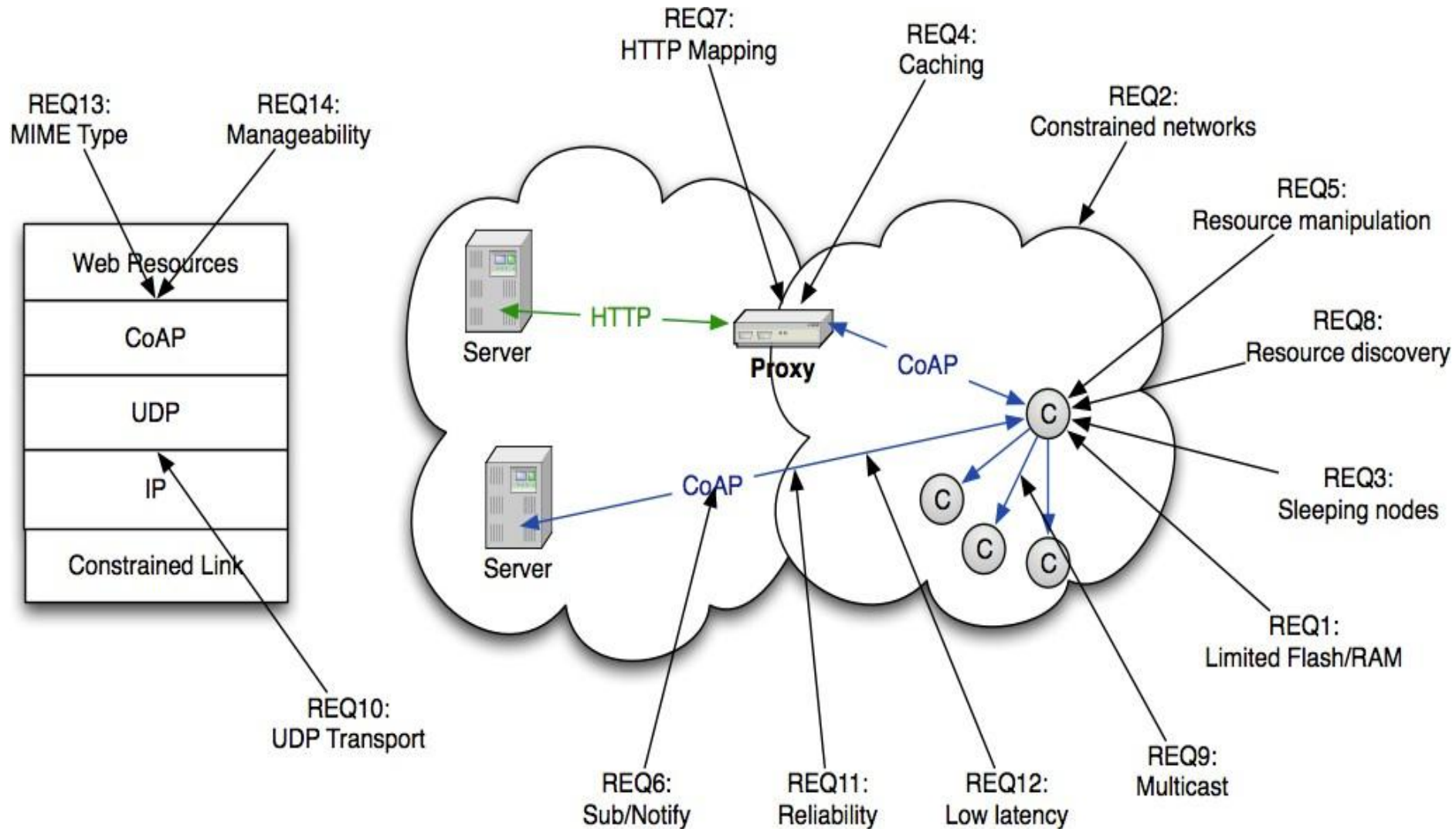


Other common HTTP methods: PUT, POST, DELETE

The CoAP Architecture



CoAP Design Requirements



CoAP At a Glance

- Embedded web transfer protocol (coap://)
- Asynchronous transaction model
- UDP binding with reliability and multicast support
- GET, POST, PUT, DELETE methods
- URI support
- 4 byte header
- Subset of MIME types and HTTP response codes
- Built-in discovery
- Optional observation and block transfer

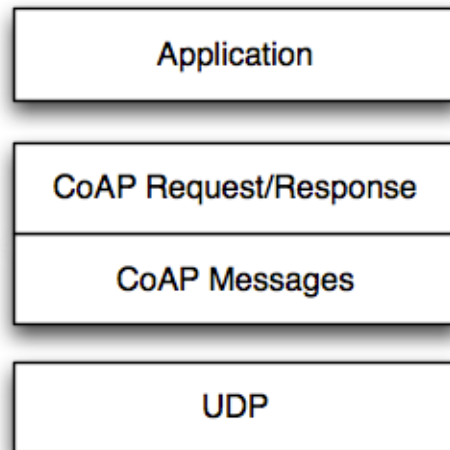
CoAP Messaging Basics

- Transport:
 - (mainly) UDP binding
- Message Exchange between Endpoints
 - Messages with 4 bytes header (shared by request and responses) containing a message ID (16 bits)
 - Reliable exchange through Confirmable Messages which must be acknowledged (through ACK or Reset Messages). Simple Stop-and-Wait retransmission with exponential back-off.
 - Unreliable exchange through Non-Confirmable Message
 - Duplicate detection for both confirmable and non-confirmable messages (through message ID)

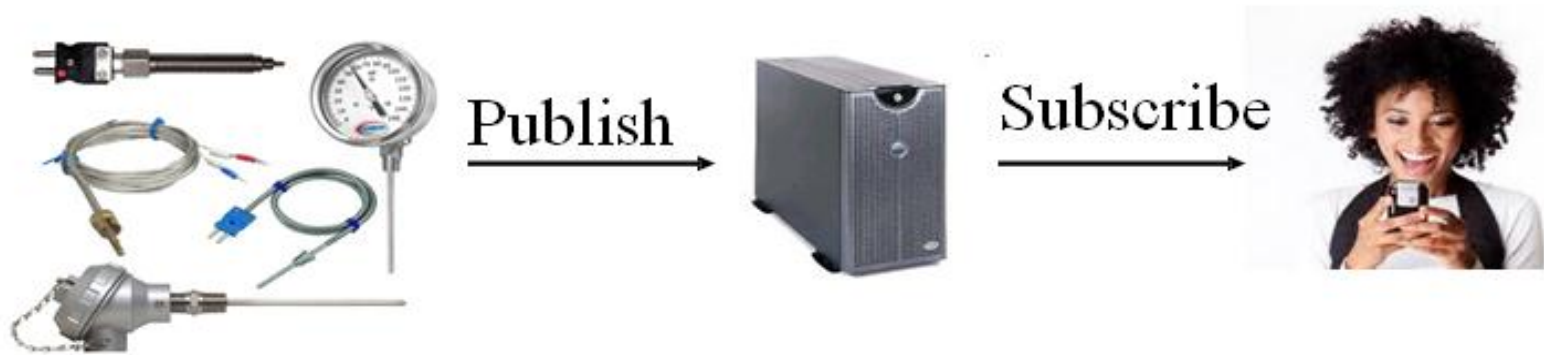
CoAP Message Semantics

REST Request/Response piggybacked
on CoAP Messages

Method, Response Code and Options
(URI, content-type etc.)



Messaging Protocols for Internet of Things: MQTT



MQTT: Message Queue Telemetry Transport

"A light weight event and message oriented protocol allowing devices to asynchronously communicate efficiently across constrained networks to remote systems"



- MQ Telemetry Transport (MQTT)
- MQTT Concepts
- MQTT Applications
- Designed for reliable communication over unreliable channel.
- Single-Board Microcontrollers
- MQTT vs. HTTP

MQ Telemetry Transport (MQTT)

- Lightweight messaging protocol for M2M communication.
- Telemetry = Tele-Metering = Remote measurements
- Invented and sponsored by IBM.
 - Now Open source. Open Source libraries available.
- MQ originated from “Message Queueing (MQ)” architecture used by IBM for service oriented networks. Queueing is based on QoS parameter.
- Telemetry data goes from devices to a server or broker. Uses a publish/subscribe mechanism.
- Lightweight = Low network bandwidth and small code footprint.

Why isn't HTTP enough?

- The HTTP standard revolutionized how *people* consume data
 - A single simple model: Send a request, read the response
 - Available via any tablet, laptop, phone, PC etc.
- The Internet of Things has fundamentally different challenges
 - HTTP remains ideal for requesting data from a known source
 - We also need an event-oriented paradigm:
 - Emitting information *one to many*
 - Listening for events *whenever they happen*
 - Distributing minimal packets of data in *huge volumes*
 - *Pushing* information over *unreliable networks*



MQTT – MQ Telemetry Transport

1. What is MQTT?

MQTT is a lightweight message queueing and transport protocol.

MQTT, as its name implies, is suited for the transport of telemetry data (sensor and actor data).

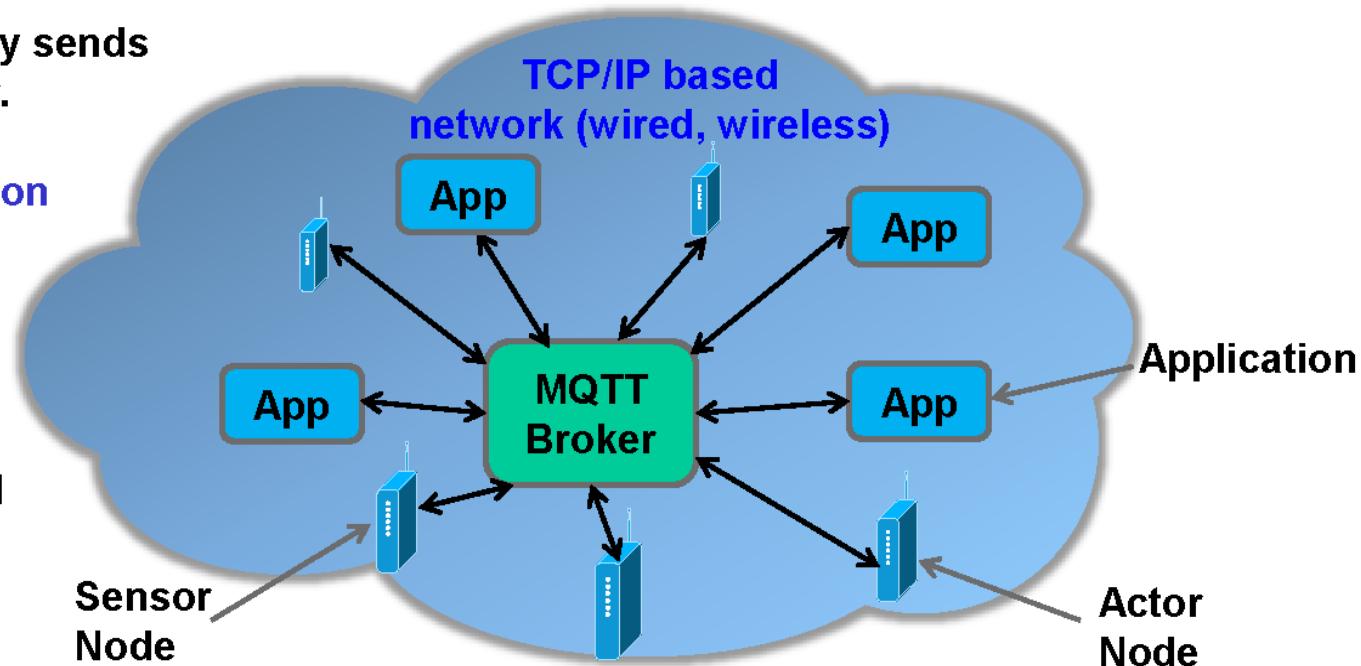
MQTT is very lightweight and thus suited for M2M (Mobile to Mobile), WSN (Wireless Sensor Networks) and ultimately IoT (Internet of Things) scenarios where sensor and actor nodes communicate with applications through the MQTT message broker.

Example:

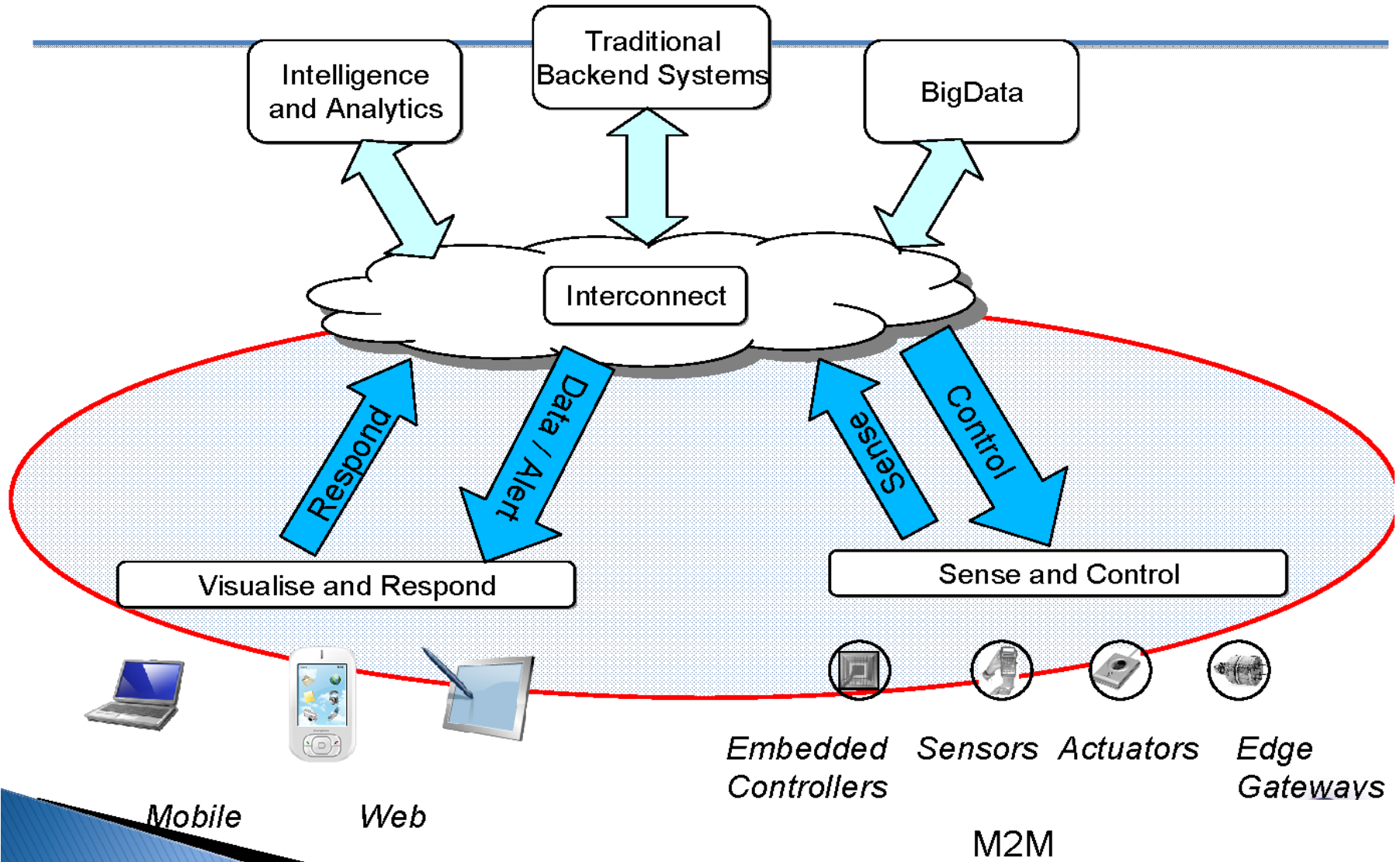
Light sensor continuously sends sensor data to the broker.

Building control application receives sensor data from the broker and decides to activate the blinds.

Application sends a blind activation message to the **blind actor node** through the broker.



The Realm of MQTT

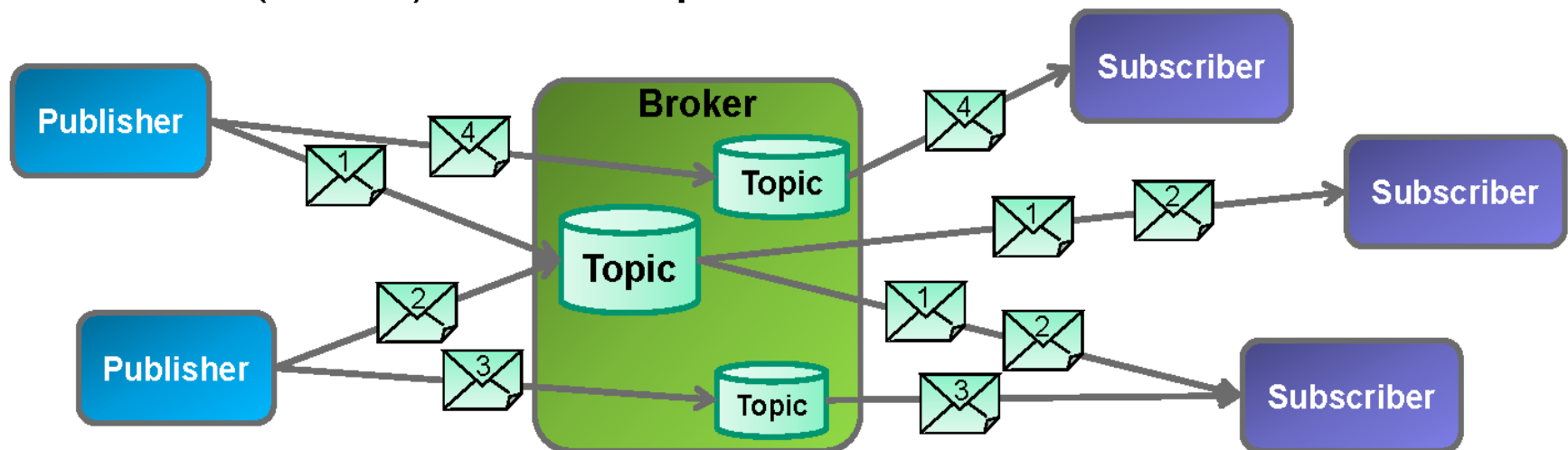


MQTT – MQ Telemetry Transport

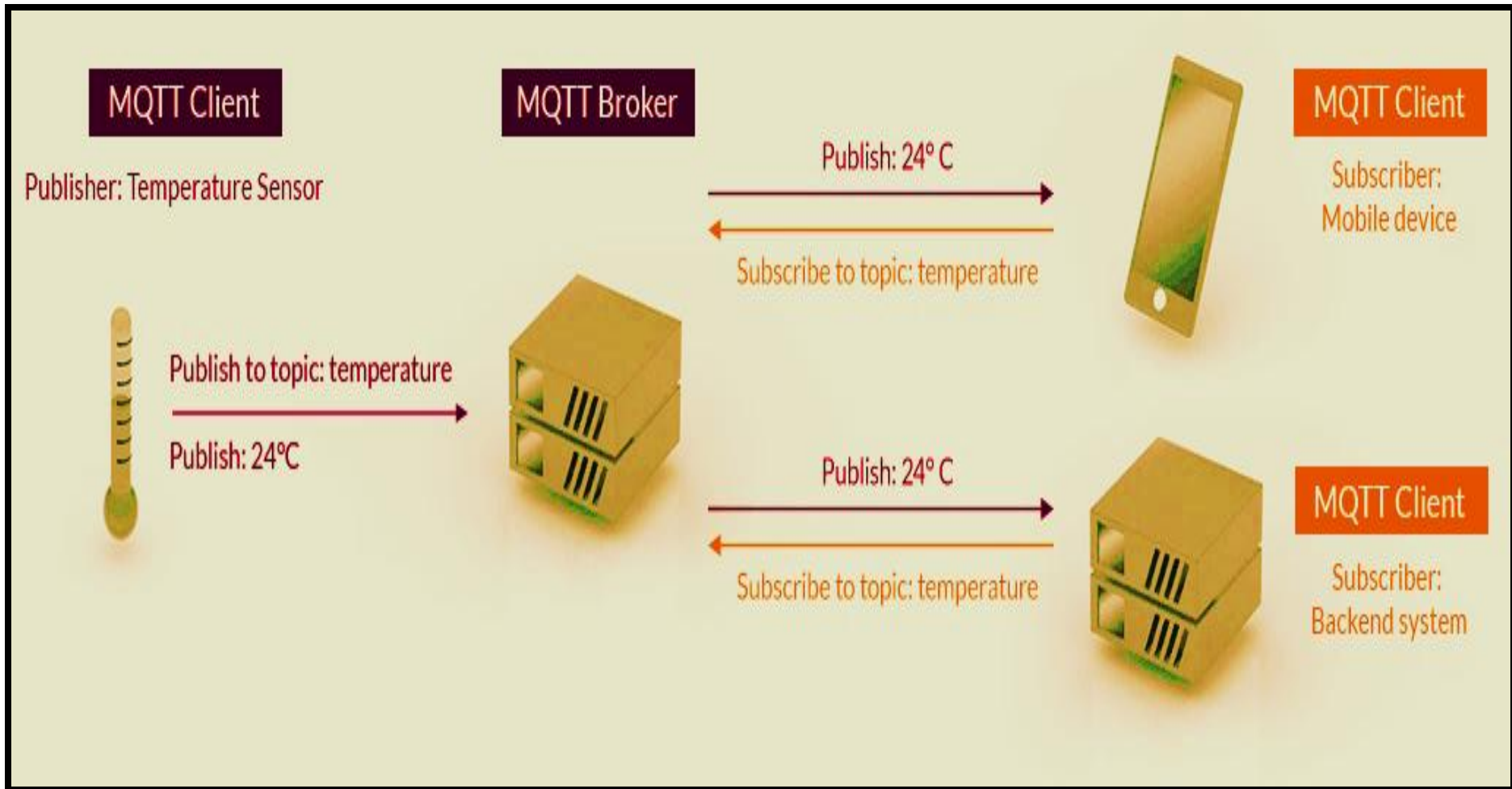
2. MQTT characteristics

MQTT Key features:

- Lightweight message queueing and transport protocol
- Asynchronous communication model with messages (events)
- Low overhead (2 bytes header) for low network bandwidth applications
- Publish / Subscribe (PubSub) model
- Decoupling of data producer (publisher) and data consumer (subscriber) through topics (message queues)
- Simple protocol, aimed at low complexity, low power and low footprint implementations (e.g. WSN - Wireless Sensor Networks)
- Runs on connection-oriented transport (TCP). To be used in conjunction with 6LoWPAN (TCP header compression)
- MQTT caters for (wireless) network disruptions

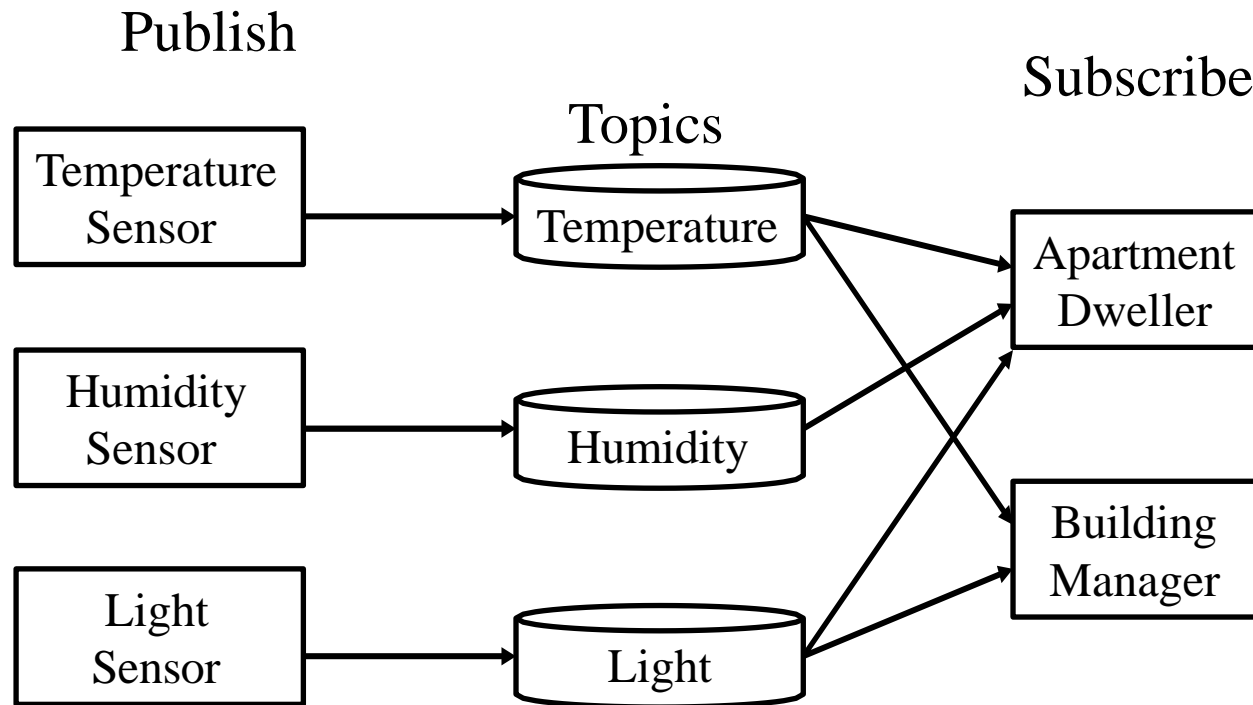


MQTT Architecture



MQTT Concepts

- ❑ **Topics/Subscriptions:** Messages are published to topics. Clients can subscribe to a topic or a set of related topics
- ❑ **Publish/Subscribe:** Clients can subscribe to topics or publish to topics.



Characteristics of MQTT

- **Uses Publish/subscriber message method:**
- The MQTT client who send the message to the MQTT Server is known as publisher and who receive the message from MQTT Server is known as subscriber.
- The MQTT Server is known as MQTT broker - Binary protocol that is built for machines (Not Text)
- Efficient (2 Byte Fixed Header)
- Bi-directional
- Data-agnostic (does not care about format of data)
- Scalable
- Built for push communication
- Suitable for constrained devices (Libraries – Arduino)

Bits	7	6	5	4	3	2	1	0
Byte 1	Packet Type				Dup Flag	QoS Level		RETAIN
Byte 2	Remaining Length							
	Options (variable length header)							
	Payload (if any)							

**Packet
format**

MQTT (Cont)

- ❑ MQTT is the **standard messaging and data exchange protocol** for the Internet of Things (IoT) and the top choice of major companies worldwide for **data exchange with constrained devices and server applications**.
- ❑ The MQTT protocol provides a **scalable and cost-efficient** way to connect devices over the Internet.
- ❑ It is **built upon TCP/IP**, it has **minimal overhead** and it is designed for reliable communication over unreliable channels.
- ❑ **Facebook messenger uses MQTT** to minimize battery usage. Several other applications in medical, environmental applications use MQTT.

Publish/Subscribe Pattern

- The pub/sub pattern provides an **alternative** to a traditional client-server architecture. In the client-server model, a client communicates directly with an endpoint.
- The pub/sub model **decouples** the client that sends a message (the publisher) from the client or clients that receive the messages (the subscribers).
- The publishers and subscribers **never contact** each other directly. In fact, they are not even aware that the other exists.
- The connection between them is **handled by a third component** (the broker).
- The job of the broker is to **filter** all incoming messages and **distribute** them correctly to subscribers.

Publish/Subscribe Characteristics

- Pub/Sub:
- Space decoupling
 - Publisher and subscribers need not be in same space.
- Time decoupling (queueing)
 - Produce of data and consuming decoupled.
- Synchronization decoupling
 - Publisher and subscribers don't wait for each other.
- Scalability
 - Broker cluster.
- Pub/Sub scales better than the traditional client-server approach.
- In brokered architecture, MQTT systems typically have a **single point of failure**. If this central component is not available, no MQTT communication is possible.
- Broker is the **Heart** of the communication protocol.

Components

- Many open source implementations of **clients** and **brokers** are available.
- **Client** Libraries – MQTT Clients
 - Eclipse Paho
 - C
 - Python
 - Hivemq
 - Java
- **Broker**
 - Really small message broker (RSMB):
 - Mosquitto
 - Micro broker: Java based for PDAs, notebooks

MQTT Clients

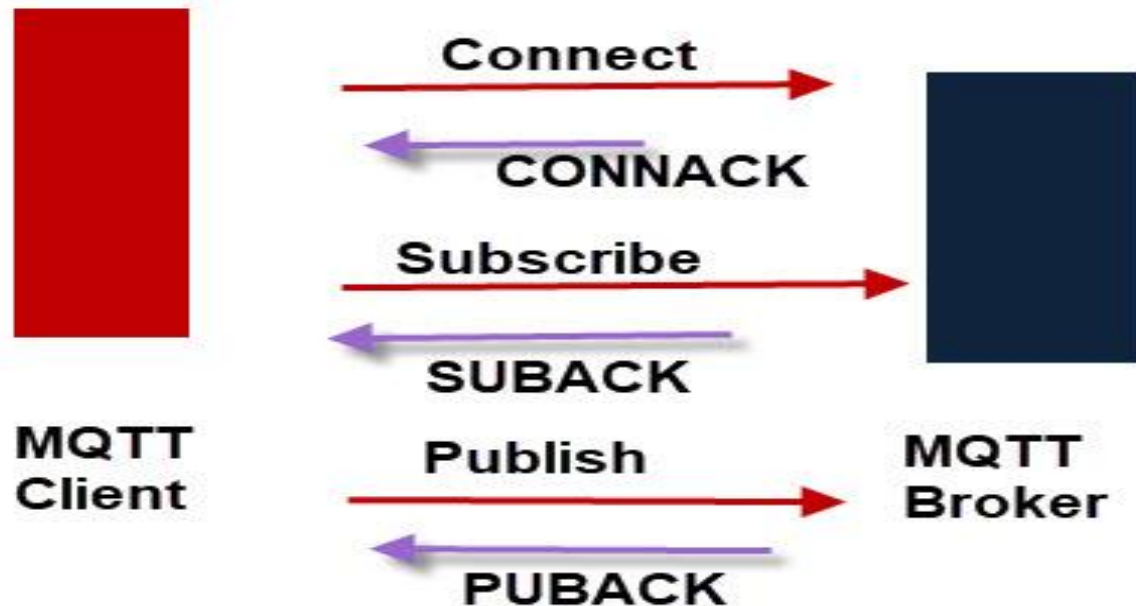
- An MQTT client is any device (from a micro controller up to a full-fledged server) that runs an MQTT library and connects to an MQTT broker over a network.
- **Both publishers and subscribers are MQTT clients.**
- The publisher and subscriber labels refer to whether the client is currently publishing messages or subscribing to messages, or does both.
- MQTT client libraries are available for a huge variety of programming languages. For example, Android, Arduino, C, C++, C#, Go, iOS, Java, JavaScript, and .NET.

MQTT Broker

- The MQTT broker is responsible for receiving all messages, filtering the messages, determining who is subscribed to each message, and sending the message to these subscribed clients.
- The MQTT protocol is based on TCP/IP. Both the client and the broker need to have a TCP/IP stack. The MQTT connection is always between one client and the broker.
- To initiate a connection, the client sends a CONNECT message to the broker. The broker responds with a CONNACK message and a status code.
- Once the connection is established, the broker keeps it open until the client sends a disconnect command or the connection breaks.

MQTT Client-Broker Connections

MQTT Message Flow



Note: MQTT is a command-response protocol each command is acknowledged. You cannot publish or subscribe unless you are connected.

MQTT Concepts (Cont)

- ❑ **Clean Sessions** and **Durable Connections**: At connection set up:
 - A clean session is one in which the broker isn't expected to remember anything about the client when it disconnects.
 - With a non clean session the broker will remember client subscriptions and may hold undelivered messages for the client.
 - Subsequent messages with high QoS are stored for delivery after reconnection
- ❑ **Wills**: At connection a client can inform that it has a will or a message that should be published if unexpected disconnection
⇒ Alarm if the client loses connection
- ❑ Periodic **keep alive** messages ⇒ If a client is still alive
- ❑ **Topic Trees**: Topics are organized as trees using / character
 - /# matches all sublevels
 - /+ matches only one sublevel.

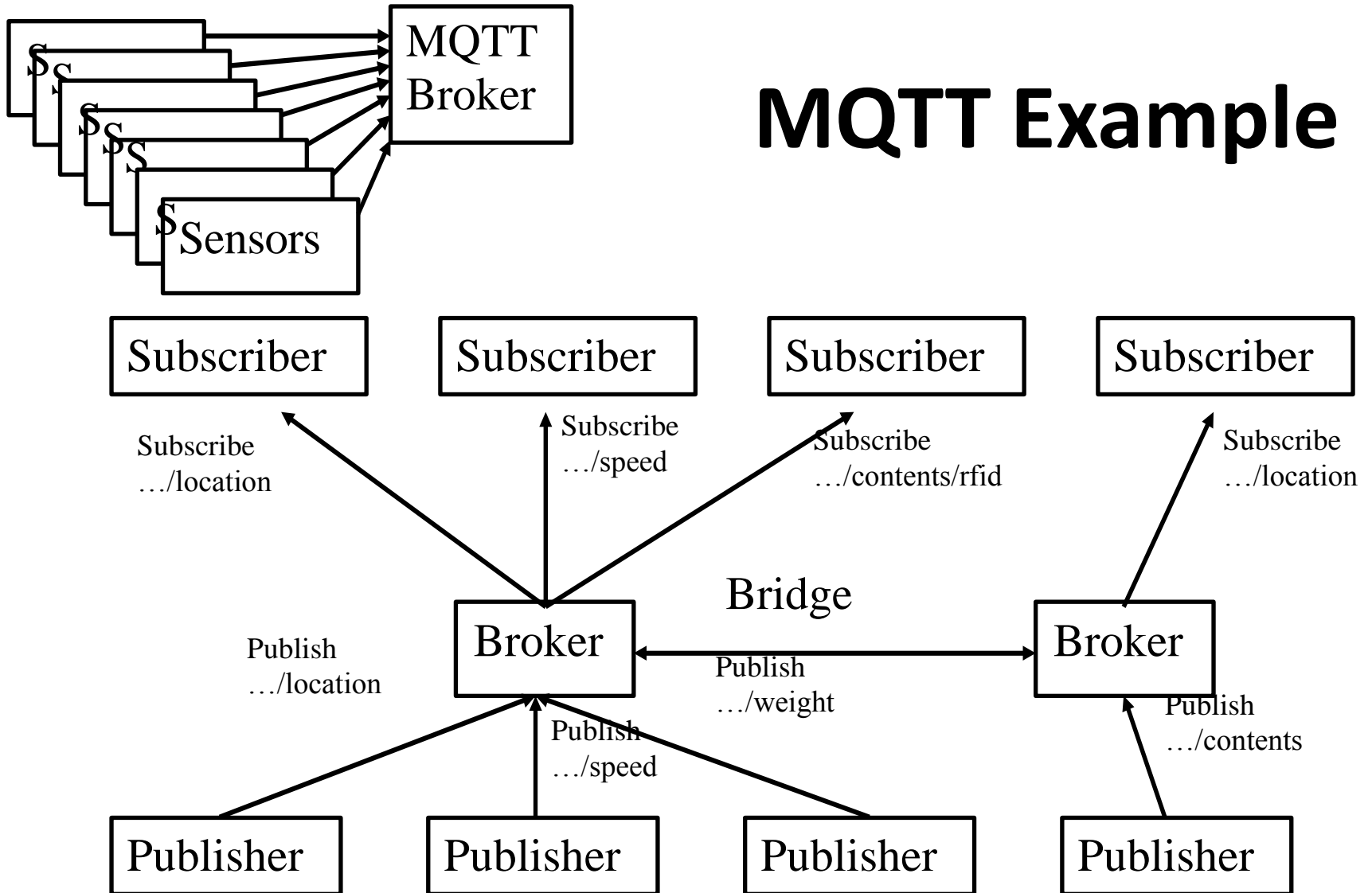
MQTT Concepts (Cont)

❑ **Quality of Service Levels:** Three levels:

- 0 = At most once (Best effort, No Ack),
- 1 = At least once (Ack-ed, retransmitted if ack not received),
- 2 = Exactly once [Request to send (Publish), Clear-to-send (Pubrec), message (Pubrel), ack (Pubcomp)]

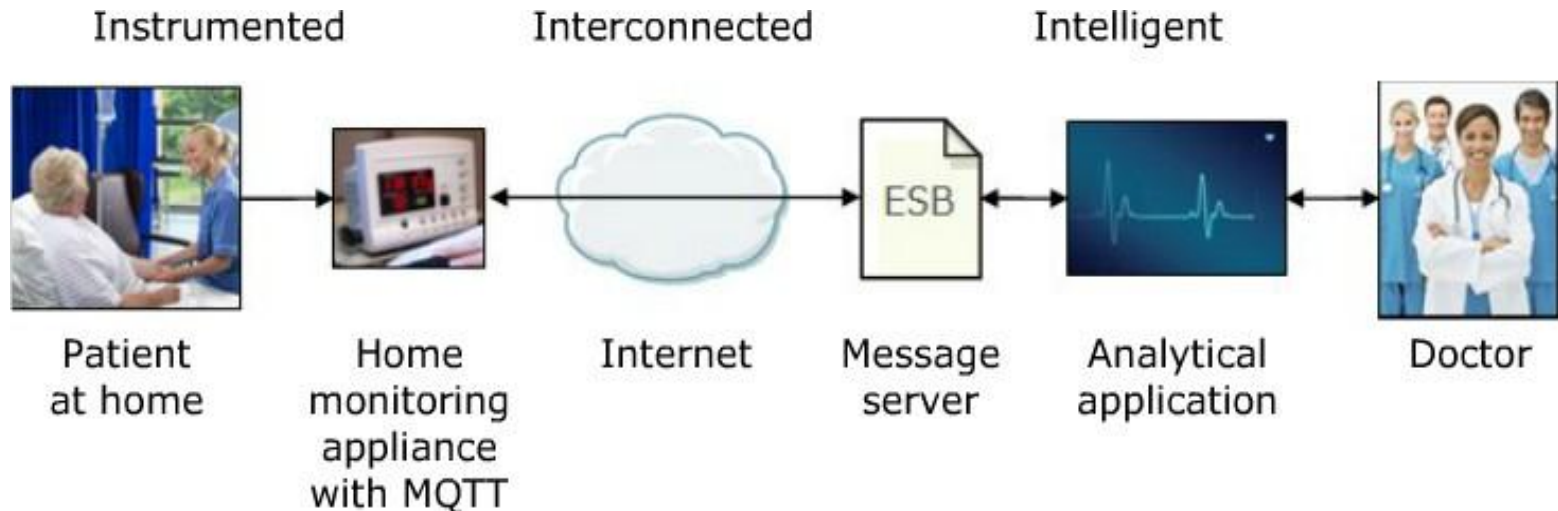
❑ **Retained Messages:** Server keeps messages even after sending it to all subscribers. New subscribers get the retained messages.

MQTT Example



MQTT Application Examples

- Home pacemaker monitoring solution
 - Sensors on patient
 - Collected by a monitoring equipment in home (broker) using MQTT
 - Subscribed by a computer in the hospital
 - Alerts the doctor if anything is out-of-order

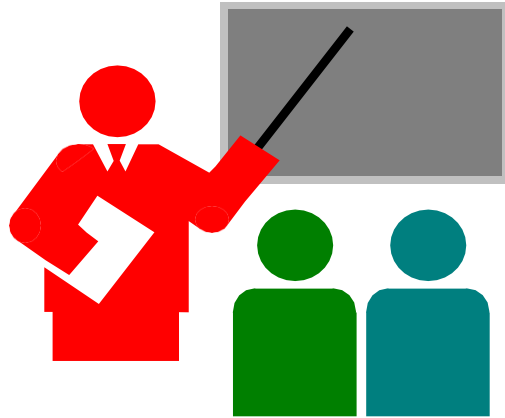


MQTT vs. HTTP

	MQTT	HTTP
Design	Data centric	Document centric
Pattern	Publish/Subscribe	Request /Response
Complexity	Simple	More Complex
Message Size	Small. Binary with 2B header	Large. ASCII
Service Levels	Three	One
Libraries	30kB C and 100 kB Java	Large
Data Distribution	1 to zero, one, or n	1 to 1 only

- ❑ Open source, <http://www.eclipse.org/paho/>
- ❑ Clients available in .NET, Perl, Python, REXX, Rube,
- ❑ Also for Arduino, Mbed, Nanode, Netduino

Summary



- ❑ MQTT is a protocol used to publish and subscribe sensor information
- ❑ Lightweight, low code size, open source

IoT Ecosystem

Applications	Smart Health, Smart Home, Smart Grid Smart Transport, Smart Workspaces, ...	Security TCG, Oath 2.0, SMACK, SASL, ISASecure, ace, CoAP, DTLS , Dice	Management IEEE 1905, IEEE 1451, ...
Session	MQTT , CoRE, DDS, AMQP, ...		
Routing	6LoWPAN , RPL , 6Lo, 6tsch, Thread, 6- to-nonIP, ...		
Datalink	WiFi, Bluetooth Smart, ZigBee Smart, Z- Wave, DECT/ULE, 3G/LTE, NFC, Weightless, HomePlug GP , 802.11ah, 802.15.4 , G.9959, WirelessHART, DASH7, ANT+ , LoRaWAN, ...		
Software	Mbed, Homekit, AllSeen, IoTivity, ThingWorks, EVERYTHING, ...		
Operating Systems	Linux, Android, Contiki-OS, TinyOS, ...		
Hardware	ARM, Arduino , Raspberry Pi, ARC-EM4, Mote, Smart Dust, Tmote Sky, ...		

Acronyms

❑ .NET	Microsoft's software framework
❑ 3G	Third Generation
❑ AMQP	Advanced Queueing Message Protocol
❑ ARC-EM4	Name of a Product
❑ ARM	Acorn RISC Machine
❑ ASCII	American Standard Code for Information Exchange
❑ AVR	Name of Atmel 8-bit RISC processor
❑ CoAP	Constrained Application Protocol
❑ DDS	Data Distribution Service
❑ DECT	Digital Enhanced Cordless Telecommunication
❑ DTLS	Datagram Transport Level Security
❑ GP	Green Physical Layer
❑ GPS	Global Positioning System
❑ HTTP	Hypertext Transfer Protocol
❑ IDE	Integrated Development Environment
❑ IEEE	Institution of Electrical and Electronics Engineers

Acronyms (Cont)

❑ IoT	Internet of Things
❑ IP	Internet Protocol
❑ ISASecure	Security Certification by ISCI
❑ ISCI	ISA Security Compliance Institute
❑ kB	Kilo Byte
❑ LoRaWAN	Long-Range Wide Area Network
❑ LTE	Long-Term Evolution
❑ MQ	Message Queueing
❑ MQTT	MQ Telemetry Transport
❑ NFC	Near Field Communication
❑ PDA	Personal Digital Assistant
❑ QoS	Quality of Service
❑ REXX	REstructured eXtended eXecutor (an interpreted programming language)
❑ RPL	Routing protocol over Low-Power and Lossy N/Ws
❑ RSMB	Really small message broker

Acronyms (Cont)

- SASL: Simple Authentication and Security Layer
- SMACK: Simplified Mandatory Access Control Kernel
- TCG: Trusted Control Group
- TinyOS: Tiny Operating System
- ULE: Ultra-Low Energy
- URL: Uniform Resource Locator
- WiFi: Wireless Fidelity

END