# Internet of Things (IoT): Protocols, Architectures and Standards

Prof. Wing C. Lau

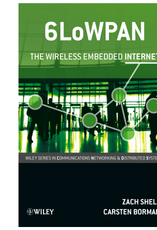
Spring 2017

# Acknowledgements

Many of the slides in this presentation are adapted from various sources including:

- Zach Shelby and Carsten Bormann, Companion Lecture Slides for the book titled: 6LowPAN: The Wireless Emdedded Internet, Published by Wiley, 2009.






Figures on slides with book symbol from 6LoWPAN: The Wireless Embedded Internet, Shelby & Bormann, ISBN: 978-0-470-74799-5, (c) 2009 John Wiley & Sons Ltd

- A. Brandt et al, “RPL: Routing Protocol for Low Power and Lossy Network,” Roll Design Team status report, IETF-75, July 2009.
- David E. Culler, Jonathan Hui and Zach Shelby, “6LowPAN”, IPSO Alliance Webinar, Nov. 2010.
- Jurgen Schonwalder, “Internet of Things: 802.15.4, 6LowPAN, RPL, CoAP,” Oct 2010
- Robert Cragie, “The ZigBee IP Stack,” IPv6-based stack for 802.15.4 networks, ZigBee Alliance 2011
- JP Vasseur, “RPL Tutorial”, IoT Workshop, Apr 2011
- JP Vasseur, “IP e2e in Smart Grid Networks/ Standardization Update”, Apr 2011
- Zach Shelby, “CoAP: The Internet of Things Protocol,” May 2013
- Kwok Wu, “Freescale Internet of Things Gateway Platform – Future Generations,” Sept 2013
- Zach Shelby, “OMA Lightweight M2M tutorial,” ARM IoT Tutorial, May 2014
- Bill Curtis, “IoT Device Standards,” ARM, 2014
- Michael Koster, “CoAP, OMA LWM2M and IPSO Smart Objects: Service and Application level Interoperability for IoT,” Tutorial in IETF-91, Nov 2014.
- Pratul Sharma, “Key requirements for Interoperable IoT systems,” Internet of Things Developers Conference, May 2014
- Julien Vermillard, “M2M, IoT, Device Management, CoAP and Lightweight M2M to rule them ALL ?”, Eclipse Day, Florence 2014
- Paolo Patierno, “Smart Home & Smart Factory systems: MQTT & IoT comparison,” Microsoft Embedded Conference, Feb 2014
- Paolo Patierno, “IoT protocols landscape,” 2014
- Vidhya Gholkar, “An Introduction to IoT Protocols,” O’Reilly Open Source Convention, July 2014.
- Hauke Petersen, “Application Layer Protocols and Data Encoding for Constrained Devices,” W3C Web of Things Workshop, Jun 2014
- Peter Egli, “Introduction to MQTT (MQ Telemetry Transport), A Protocol for M2M and IoT Applications,” 2015

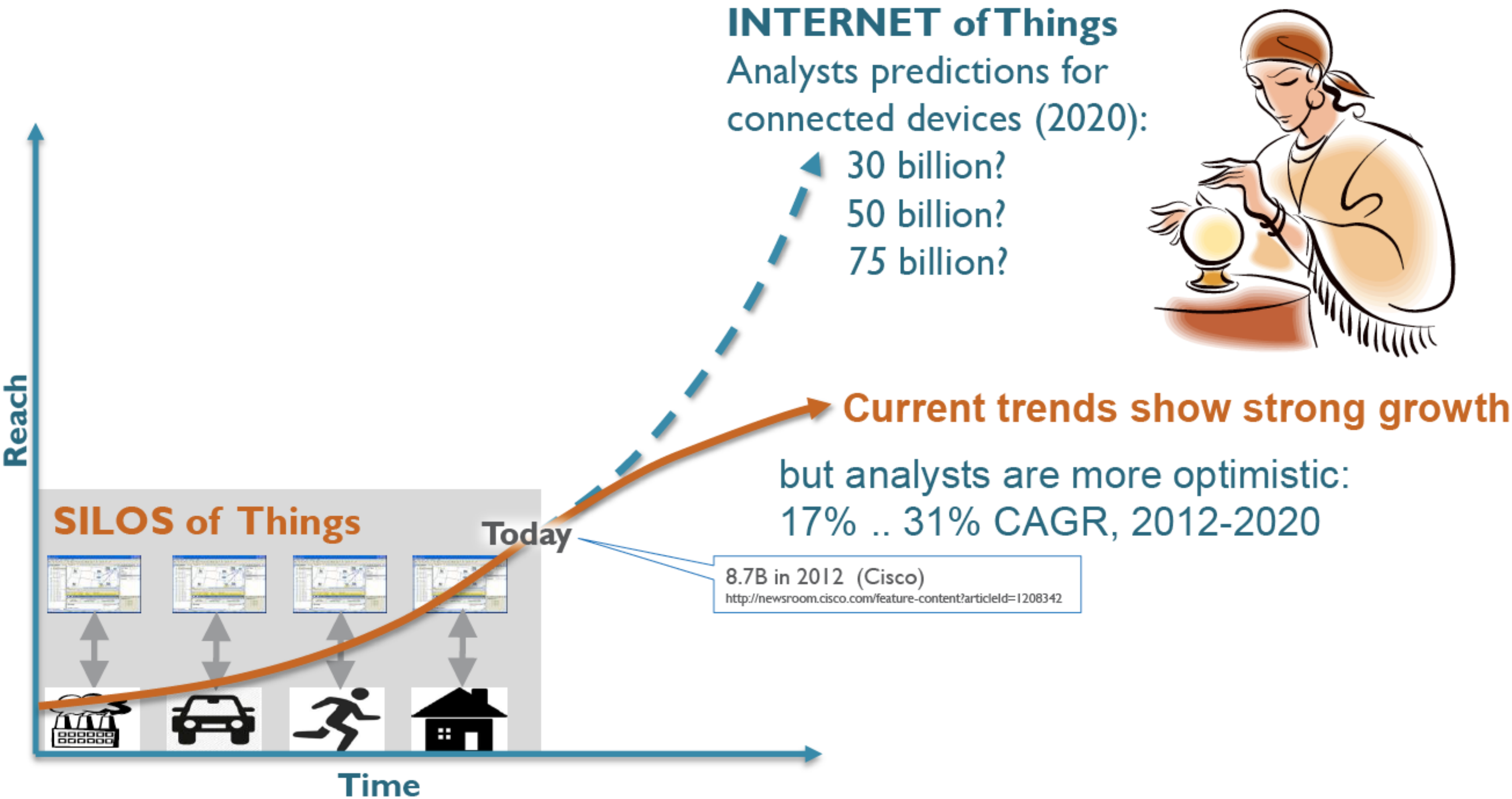
**The copyright of the adapted slides belong to the original owner of the material and are hereby acknowledged.**

# IoT: Not a new Idea

- IoT: Things around us become smart and connected
  - This has been going on for decades
  - By 2010, # of Connected Things > World Population (6.8 Billion)

Weiser, Mark (1991) "the Computer for the 21st Century"
Ubiquitous computing: "The most profound technologies are those that disappear. They weave themselves into the fabric of everyday life until they are indistinguishable from it."

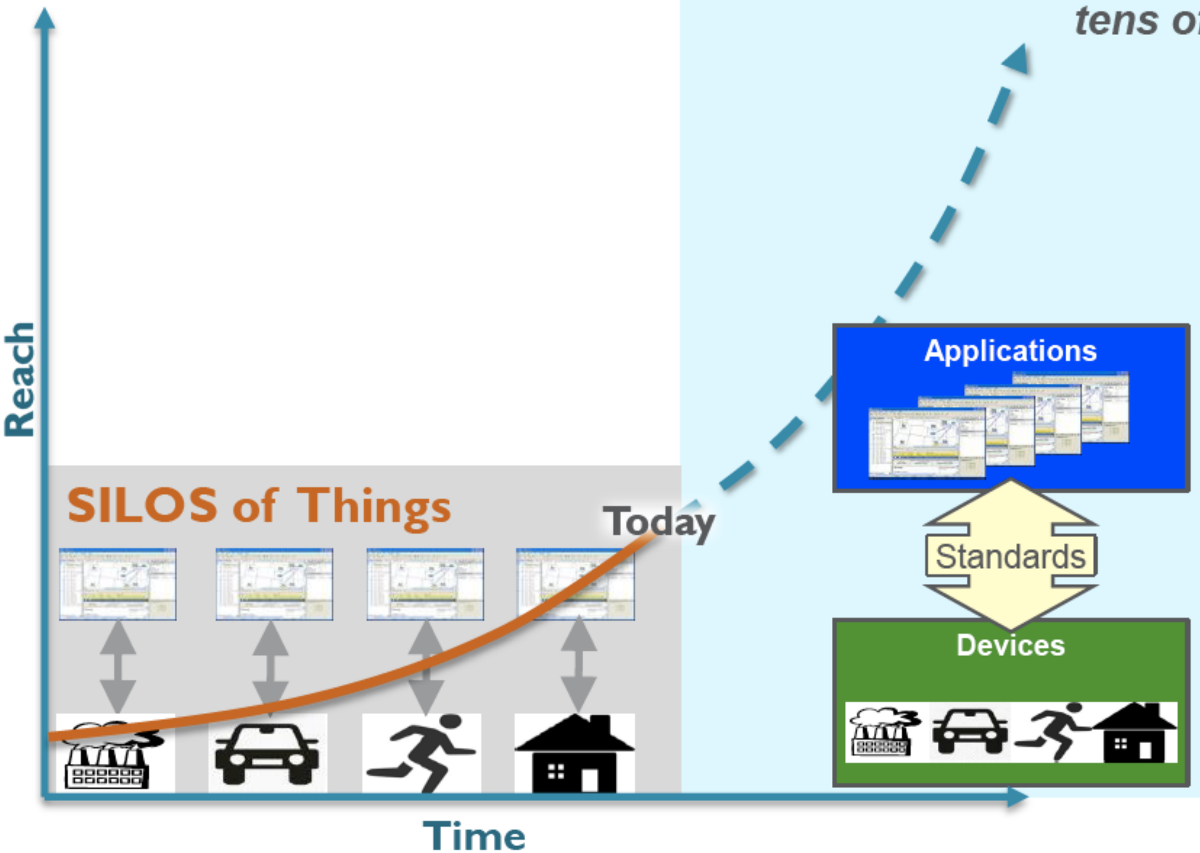
# Accelerating Development of IoT



# Accelerating Development of IoT

## INTERNET of Things

***What will drive demand for many tens of billions more devices?***




### Better IoT Platforms

*… that can "weave themselves into the fabric of everyday life"*

- Integrated wireless
- Right-size processors, memory
- Low cost, low power
- Secure, trustworthy
- Easy software development
- Easy integration into "things"

### Internet-scale IoT ecosystems

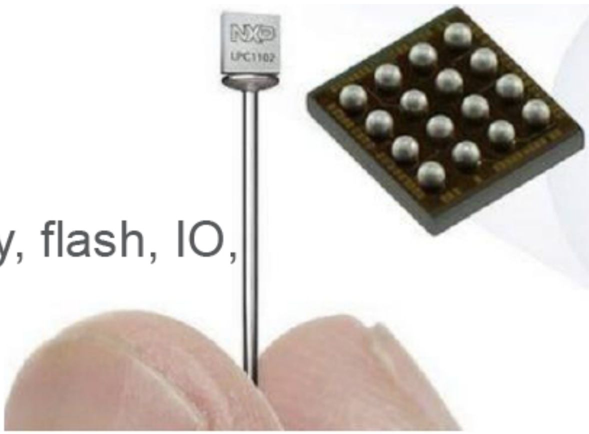
*Bust the silos*

- Standards-based connectivity
- Standards-based provisioning
- Open markets for devices, apps
- End-to-end security

# IoT System-on-Chip (SoC) Platform Evolution

- **Wireless** On-chip radios
  Optimized for IoT bandwidth, power
- **Right size** 32 bit processor with “the right” memory, flash, IO,
- **Low cost** Embeddable, often disposable
- **Low power** No visible power source
  Power managed (off or asleep much of the time)
- **Secure** Trustworthy … *by design*
- **Easy dev’t** Stacks, tools, software
- **Easy integration** into real-world “things”

Industry challenge




Shown: NXP LPC1102 based on ARM Cortex-M0 processor

# IoT Ecosystem Evolution

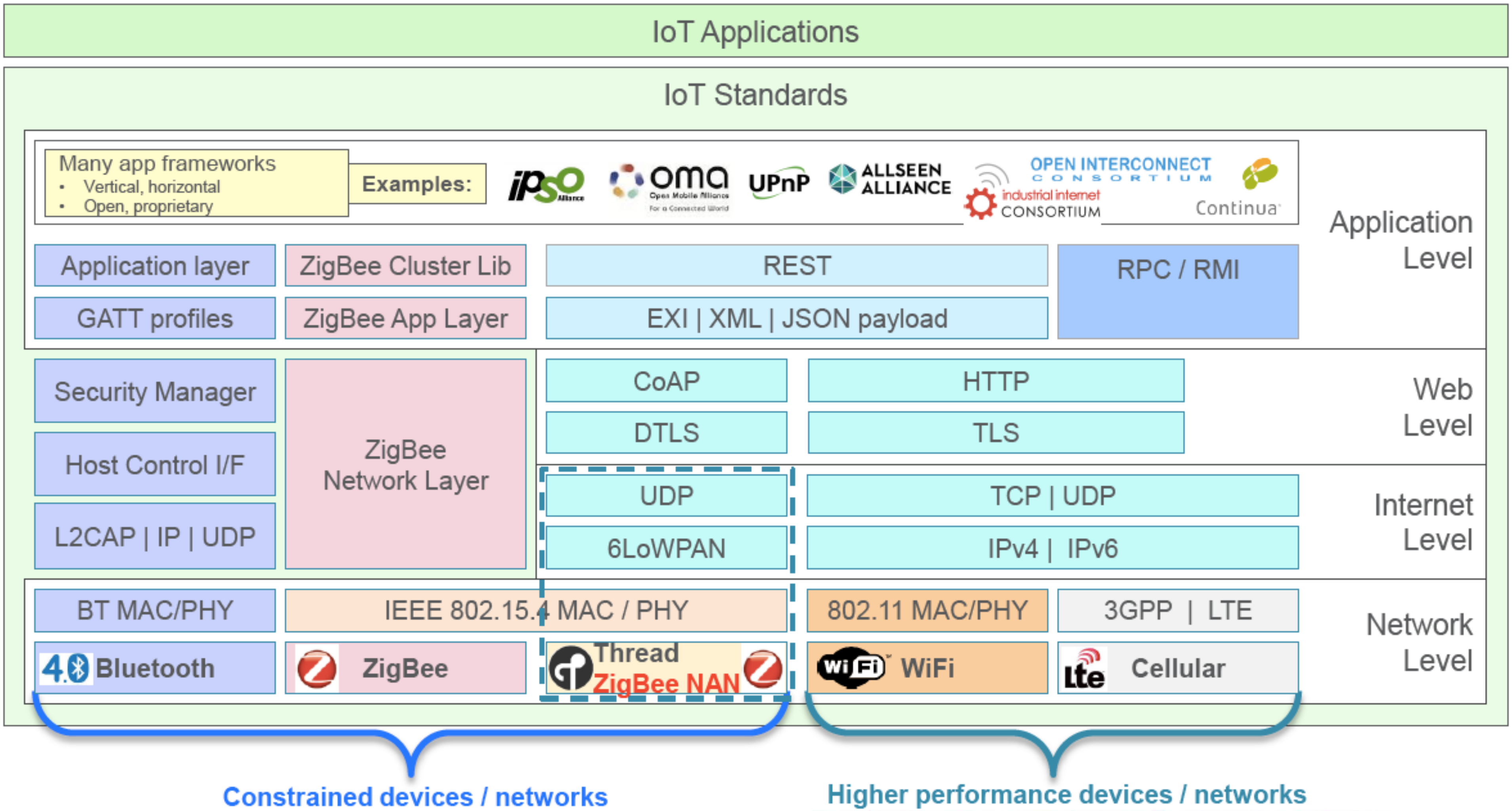
- The problem that we need to solve: Bust the silos!
  - 40 years ago: Internet technologies displaced proprietary interconnects
  - 25 years ago: Web applications – “100% reach”
  - 7 years ago: Mobile revolution: Internet and web in your hand
- Obvious IoT strategy: *Follow the Internet model*
  - Open standards enable independent development of solution components
- However, IoT platforms are ***constrained***
  - Internet / Web standards can’t be used as-is
- “INTERNET” of things is not a new idea
  - >7 years of standards development

> - Low power platforms
> - Limited memory, flash
> - Limited computation
> - Low power wireless
> - Low bandwidth
> - Small packets
> - Sleepy
> - No UI

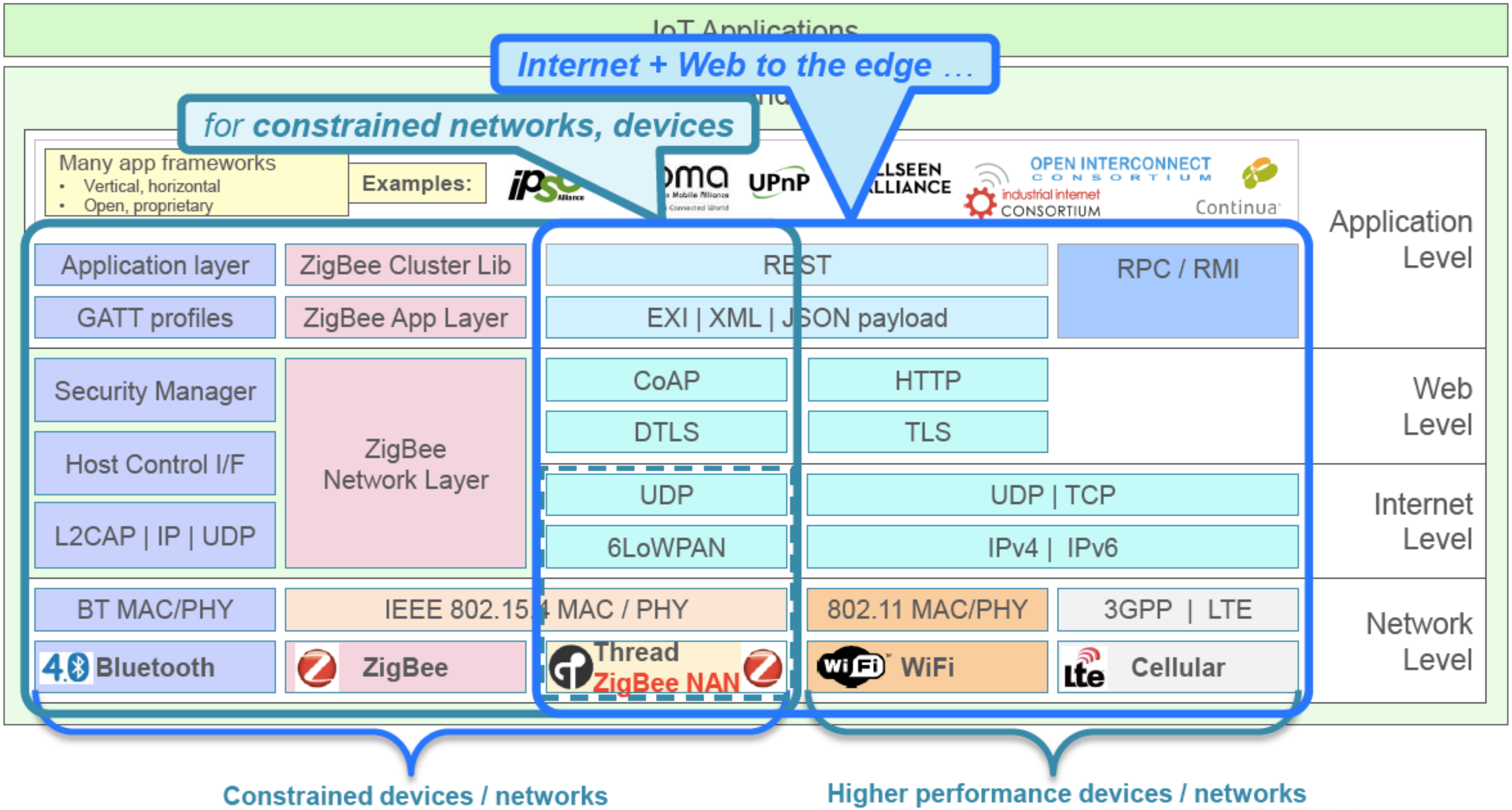
# > 120 relevant IoT/ M2M Standards and counting…

- **Horizontal**
  - 3GPP, 3GPP2, ACM, AHCIET, AIM, AllSeen Alliance, ANCE, Bluetooth SIG, CINTEL, CITEL, Hart Communication Foundation, IETF, IPSO Alliance, MIG, MQTT.org, NFC Forum, ngConnect, NYCE, OASIS, ODVA, OGC, ONVIF, Open Interconnect Consortium, OSGi, PUCC, SD Card, SIM Alliance, TCG, Thread, W3C, WAVE2M, ZigBee Alliance
- **Automotive**
  - AEC-Q100, AUTOSAR, CAR2CAR, CE4A, ERTICO, Global Platform, Icar Support, ITSA, ITS Info-Comms Forum, JASPAR, Mobey Forum, MOST Cooperation, OSPT Alliance, PATA, SAE International, UIC, ATMIA, ISIS, ISO, NACHA, NAMA, SPA
- **Healthcare**
  - AAMI, AdvaMed, American Telemecine Ass'n, ASME, ASTM Int'l, Canadian Telehealth Forum, CDISC, CEN/TC 251, CLSI, Continua Alliance, EHTEL, European Mhealth Alliance, GE1 Healthcare, HIMSS, HITSP, HL7, IHT2, ISO/IEEE 11073, ISO TC215, Joint Commission (JCAHO), mHealth Alliance, MITA, MITA DICOM
- **Home Automation**
  - ASIS Int'l, Aureside, BACnet, Broadband Forum, CABA, EnOcean Alliance, HGI, Home Grid Forum, Home Plug Alliance, KNX, OBIX, PSIA, SIA (security), Z-Wave Alliance
- **Industrial**
  - AIA, Automation Federation, CiA, Industrial Internet Consortium, ISA, M-Bus, Modbus, OCARI Alliance, OMAC, OPC, SMLC
- **Utilities, Smart Grid**
  - AAPA, CIGRE, DLMS, DRSG Coalition, EDSO, EEI, ENTSOE, ESMKIG, Eurelectric, EUTC, Gridwise Alliance, Gridwise Architecture Council, JSCA, NEMA, NIST, T&D Europe, TIA TR-51, UCA, UTC Smart Network Council, UTC
- **Supply Chain**
  - AIM, APICS, CSCMP, GS1, ISM, SCM, XBRL Int'l
- **ITU GSC (Global Standards Collaboration) members**
  - ITU-T, ARIB, ATIS, CCSA, ETSI, ISACC, TIA, TTA, TTC
- **ITU GSC observers**
  - 4G America, AICTO, CDG, GISFI, GSMA, IEC, IEEE, ISO / IEC JCT, OMA, SCTE

# A Glimpse of some IoT Standards

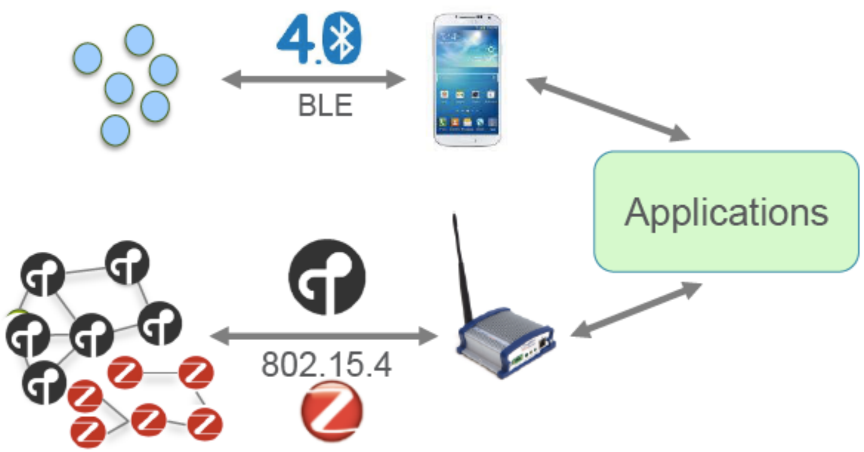


# A Glimpse of some IoT Standards



# Constrained Networks for IoT

- Bluetooth Low Energy (BLE) PAN – hub/spoke topology
  - Widely deployed in phones, tablets
  - Becoming ubiquitous for low-power PAN
  - Smart phone is a "natural" proxy / access point
- 802.15.4 – mesh topology
  - Consumer
    - Thread (2015)
    - ZigBee Pro
  - Industrial
    - ZigBee SE (Smart Energy)
    - ZigBee NAN (neighborhood area)




- Challenges for constrained networks
  - Slow – low data rate "tens to hundreds of k-bits" typical
  - Sleepy – aggressive power management
  - No delivery or in-order guarantee – dropped packets simply drop!
  - Multicast

# Wish List for IoT Consumer/Residential Low-bandwidth Networks

- Mesh-based Architecture for Consumer/ Residential IoT applications

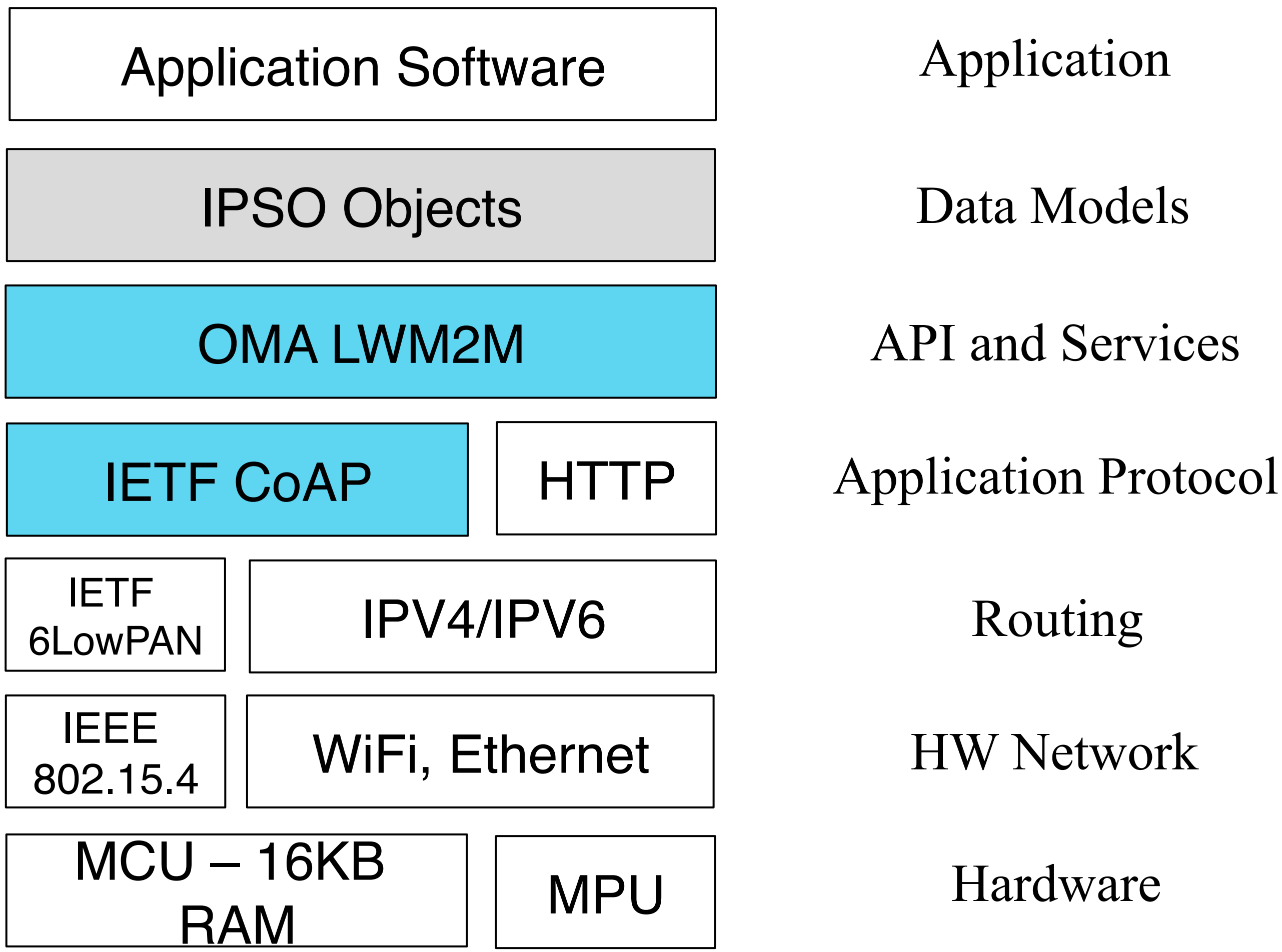
  Why ?
  - Whole-home coverage
  - Enable very-low power radios
  - Coverage increases as Devices are added
- "Open-standard" Protocol
- IP-based (IPv6)
  - Allow End-to-end addressable architecture
- Low Power (sub-10 mW roadmap)
  - e.g. Typical Power consumption of IEEE 802.15.4 ~ 1% of WiFi ; Sleep 99% of the time
- Resilient
  - No Single-Point-of-Failure
- Multi-vendor silicon
- Multi-vendor interoperability
- Secure, Consumer-Friendly, Easy to Install

# Current/ Emerging Options for IoT Networks

- WiFi  (upcoming IEEE 802.11ah expected to be completed by mid 2016)
- ZigBee Pro
- Z-Wave
- Insteon
- Bluetooth / Bluetooth Low Energy (BLE)
- 5G/ LTE M2M
- Thread - yet another IoT-related standards consortium formed in mid 2014

and others …

# One of the Emerging Internet/Web Protocol Stacks for IoT Networks



# Why taking the Internet + Web approach to the Network Edge for IoT ?

- Expect to have "Web-scale" growth for IoT by including Constrained Networks and Devices
- Give every device a Unique address
  - IPv6 is IoT-friendly* thanks to its support of:
    - Huge address space, Auto-configuration, Secure, Mobile, Globally unique end-to-end routing
  - Legacy IPv4 via tunneling
- Enable Web-scale Software/Application/Service development
  - Client/ Server computing paradigm with 100% end-to-end reach
  - Use Web-scale, e.g. W3C Standards, Design Patterns and Tools
  - RESTful, Application/Media-Independent
  - Device and Resource Discovery ; Automated Provisioning

* The Internet of Everything through IPv6: An Analysis of Challenges, Solutions and Opportunities
Antonio J. Jara, Latif Ladid, Antonio Skarmeta - http://ipv6forum.com/iot/images/jowua-v4n3-6.pdf

# BUT… Can Internet and Web protocols scale down to Constrained IoT networks ?

| | |
|---|---|
| ▪ **Addressing** **IPv6**<br>▪ Uniform, unique addressing | ▪ **Inefficient over constrained networks**<br>▪ 40 byte IPv6 header is ~1/3 of an 802.15.4 packet |
| ▪ **Transport** **TCP**<br>▪ Guaranteed in-order packet delivery | ▪ **Impractical with unreliable networks**<br>▪ **Fails on sleepy platforms** |
| ▪ **Application** **HTTP**<br>▪ Any type of message can be exchanged between any nodes | ▪ **Requires reliable, in-order transport (TCP)** |
| ▪ **Security** **TLS**<br>▪ Secure messaging using standards-based protocols | ▪ **Requires reliable, in-order transport (TCP)** |

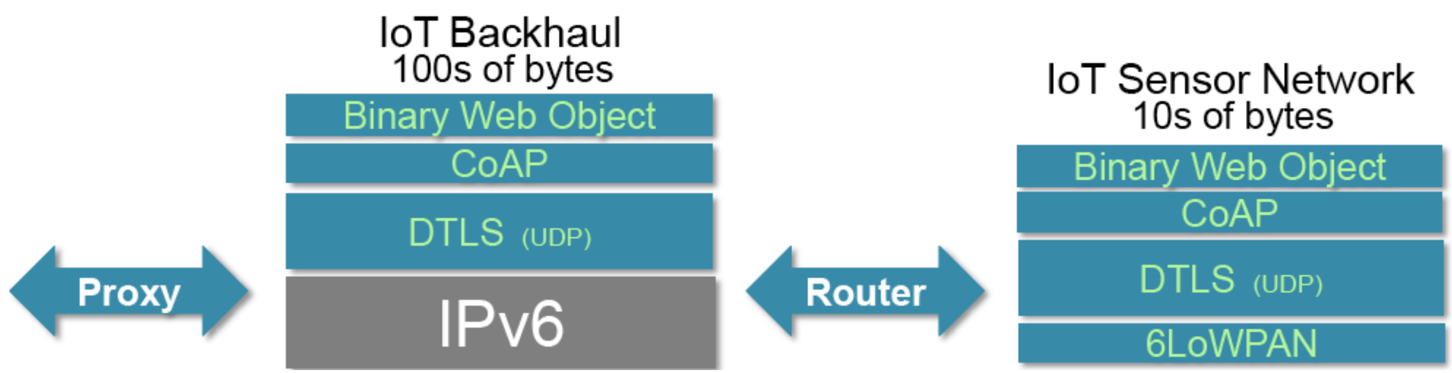
# Internet and Web protocols for Constrained IoT networks ?

IPv6 ➔ **6LoWPAN**
- Header compression on sensor networks

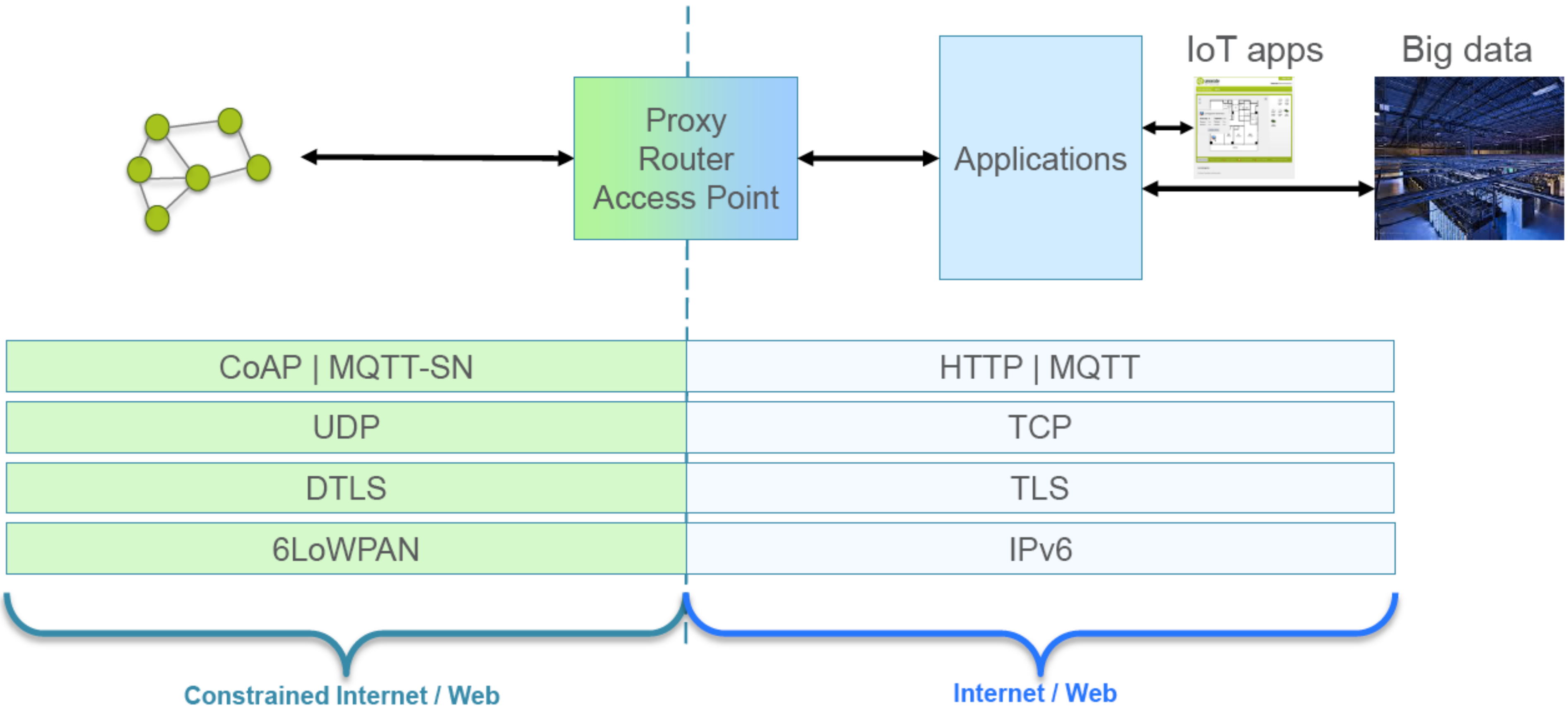
TCP ➔ **UDP**
- No guarantee of packet delivery or order

HTTP ➔ **CoAP**
- HTTP-like (REST) protocol for constrained devices

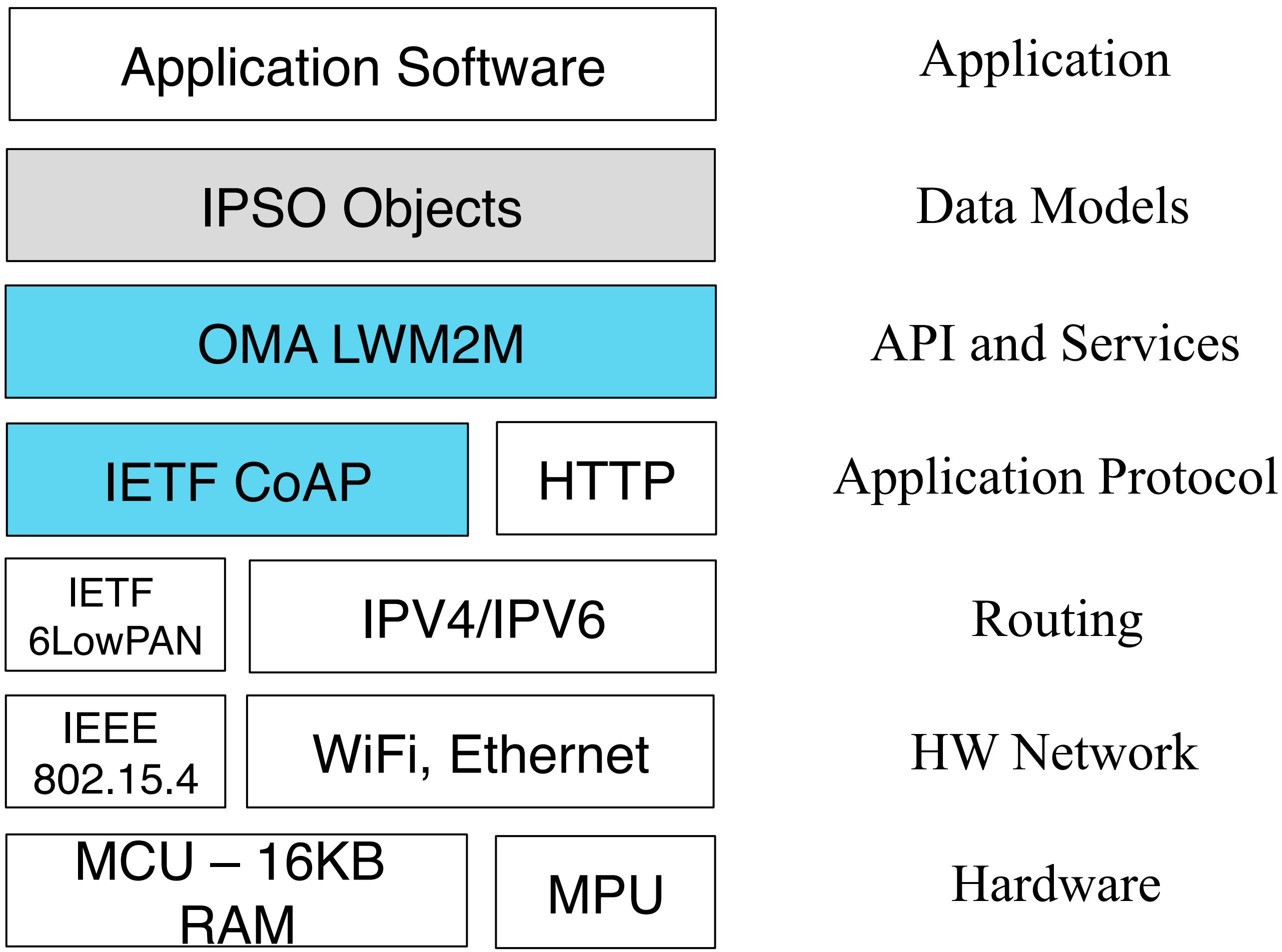
TLS ➔ **DTLS ➔ eDTLS**
- TLS over UDP – stateless – one packet at a time

# Deploying Constrained Internet / Web protocols for IoT: Little Data to Big Data

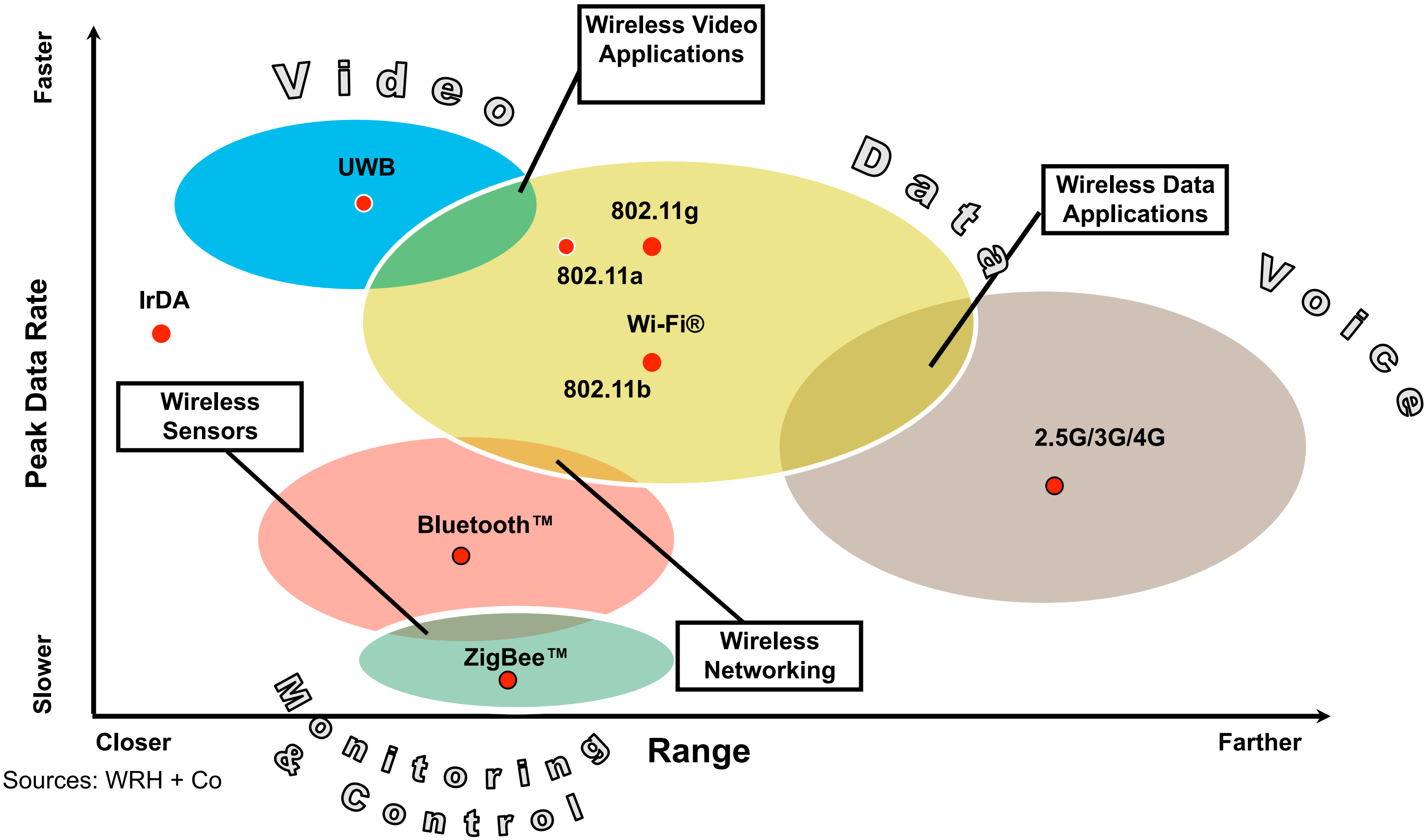


# Layer-by-layer Overview of an Emerging IoT Stack



# Wireless Links for IoT

# Positioning of Different Wireless Link Technologies




Sources: WRH + Co

# IEEE 802.* Wireless Link Standards

| | 802.15.4 | 802.15.1 | 802.15.3 | 802.11 | 802.3 |
|---|---|---|---|---|---|
| Class | WPAN | WPAN | WPAN | WLAN | LAN |
| Lifetime (days) | 100-1000+ | 1-7 | Powered | 0.1-5 | Powered |
| Net Size | 65535 | 7 | 243 | 30 | 1024 |
| BW (kbps) | 20-250 | 720 | 11,000+ | 11,000+ | 100,000+ |
| Range (m) | 1-75+ | 1-10+ | 10 | 1-100 | 185 (wired) |
| Goals | Low Power, Large Scale, Low Cost | Cable Replacement | Cable Replacement | Throughput | Throughput |

# IEEE 802.* Standards and their Application Focus

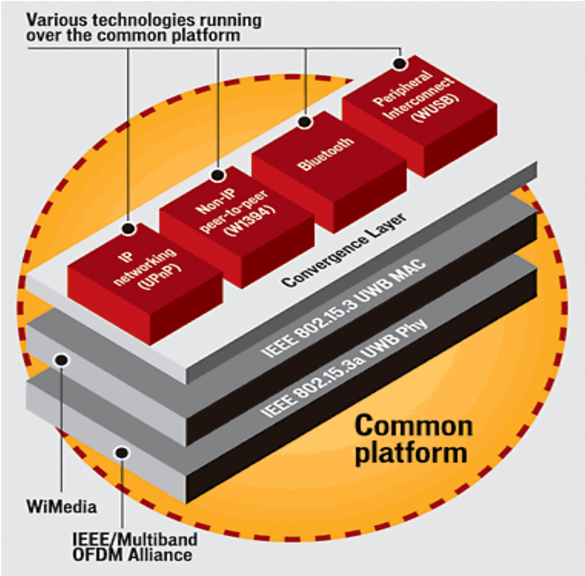
- 802.11* (WiFi): Wireless Ethernet
  - 802.11b:
    - Adequate for highly-compressed video. Non-isochronous MAC requires buffering, network congestion interrupts. Rapidly increasing adoption by IT staff including use in factories & even hospitals. Very long range pt-2-pt links (Wi-Bridges) using outdoor high-gain antennas.
  - 802.11a:
    - Up to 5x rate @ 5.2/5.7 GHz, typically shorter range in practice.
  - 802.11g:
    - 11b vendors competing with 11a data rate at 2.4 GHz.
  - 802.11n:
    - High-throughput extension using MIMO, used in AppleTV etc.
- 802.15.1 (Bluetooth): Short Range Streaming Data & Voice
  - Isochronous support for a range of devices, PC peripherals & headsets.
- 802.15.3 (WiMedia): Streaming Multimedia, Consumer electronics, multiple HDTV channels ; (e.g. may be relevant to Video Surveillance)
  - 802.15.3a: Task group developing alt. UWB PHY, 100-480 Mbps @ 3.1-10.6 GHz ; Players include Intel, Motorola, etc
  - 802.15.3c: 1Gbps range at Microwave frequency ; overlap with 802.11VHT ?
- 802.15.4: Sensor Networks, Home/Industrial Automation, Toys.
  - Low Duty Cycle, Long Battery Life, Highly Scalable Networks

# Other Related Standards

- IEEE P1451
  - Industrial Control applications to support both Monitoring and Actuation using
    - Smart (Networked) Transducers (i.e. Sensors)  and Actuators
    - Network Capable Application Processors (NCAP)
    - Transducer Electronic Data Sheet (TEDS) to realize Self-describing “smart” transducers (Sensors)/actuators
  - Overlap is likely between IEEE P1451.5, ZigBee and Bluetooth which have already defined their full 7-layer protocol stack
- RFIDs are another important class of sensors
  - EPCglobal network Standards: Electronic Product Code (EPC): for RFID applications (www.epcglobalinc.org)
    - Different Classes of RFID tags
      - Passive  tags derive energy from RF radiation from Readers
      - Active tags has their own battery ; may carry sensors on-board.
    - EPC network components
      - Infrastructure including Readers, Middleware
    - So far not using multi-hop relay between tags yet

# Wireless Personal Area Network (WPAN) Standards for IoT

- IEEE 802.15.* Wireless Personal Area Network (WPAN) Standards:
  - 802.15.1 (Bluetooth) (go beyond PHY/MAC)
  - 802.15.3 (UWB=UltraWideband, WiMedia, Wireless USB)




  - 802.15.4: PHY/MAC layer for ZigBee, ISA100.11a, WirelessHART and MiWi
    - Each of the latter specifies additional upper layers for 802.15.4
    - e.g. ZigBee Alliance: Sensor Networking Standard
      - ZigBee also cover Networking layer, Application Framework layer
      - use IEEE 802.15.4 as physical (PHY) and MAC/ data-link layers

# IEEE 802.15.4 WPAN Standard

# IEEE 802.15.4 Basics

- Simple packet data protocol for lightweight wireless networks
  - First released in May 2003
  - Channel Access is via Carrier Sense Multiple Access with collision avoidance and optional time slotting
  - Message acknowledgement and an optional beacon structure
  - Multi-level security
  - Works well for
    - Long battery life, selectable latency for controllers, sensors, remote monitoring and portable electronics
  - Configured for maximum battery life, has the potential to last as long as the shelf life of most batteries

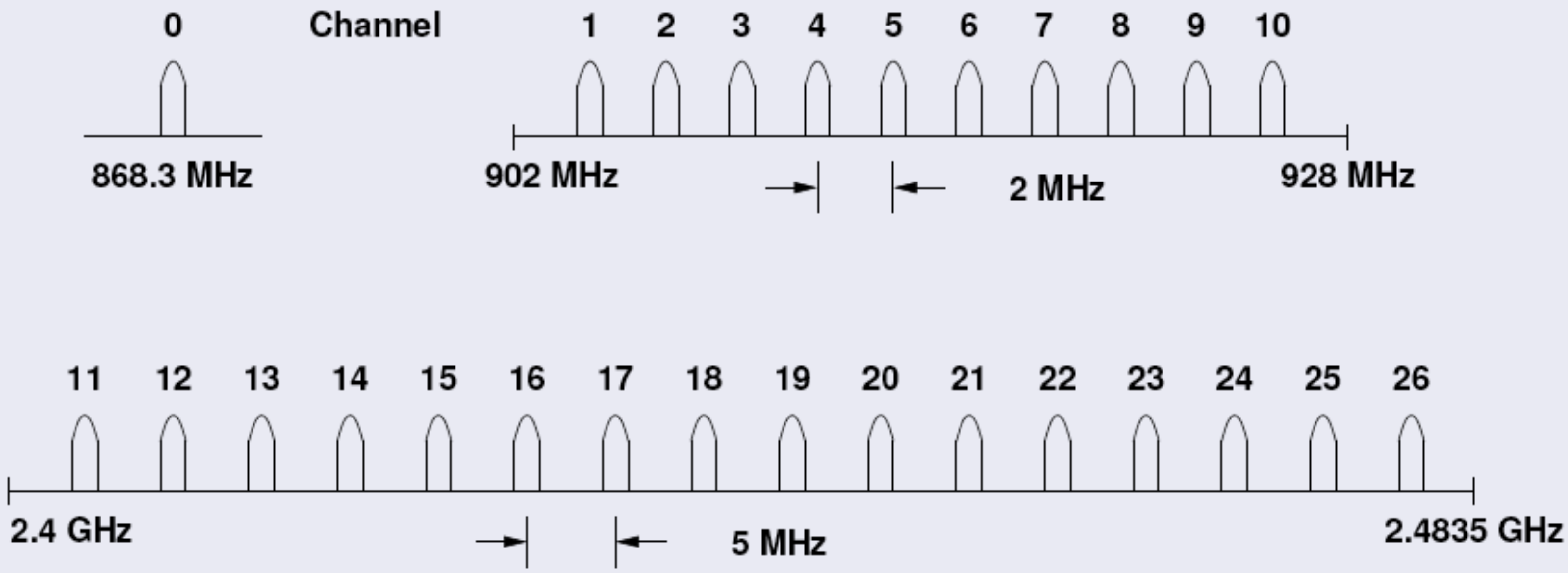
| Frequency Band | License Required? | Geographic Region | Data Rate | Channel Number(s) |
|---|---|---|---|---|
| 868.3 MHz | No | Europe | 20kbps | 0 |
| 902-928 MHz | No | Americas | 40kbps | 1-10 |
| 2405-2480 MHz | No | Worldwide | 250kbps | 11-26 |

# IEEE 802.15.4 Standards

- 802.15.4-2003
  - Original version using Direct Sequence Spread Spectrum (DSSS) with data transfer rates of 20-40 kbps
- 802.15.4-2006
  - Revised version using DSSS with higher data rates and adding Parallel Sequence Spread Spectrum (PSSS)
    - PSSS uses CDMA-principle to send in parallel a superposition of orthogonal sequences with M-ary modulation
  - Up to 250 kbps at a range of 10m
- 802.15.4a-2007
  - Adding Direct Sequence Ultra-wideband (UWB) and Chirp Spread Spectrum (CSS) physical layers to the 2006 version of the standard with ranging support

# Radio Characteristics of IEEE 802.15.4

## Frequencies and Data Rates



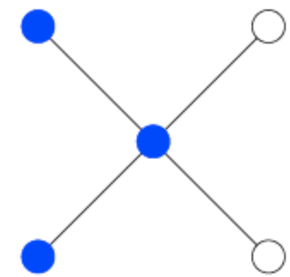

| Frequency | Channels | Region | Data Rate | Baud Rate |
|---|---|---|---|---|
| 868-868.6 MHz | 0 | Europe | 20 kbit/s | 20 kBaud |
| 902-928 MHz | 1-10 | USA | 40 kbit/s | 40 kBaud |
| 2400-2483.5 MHz | 11-26 | global | 250 kbit/s | 62.5 kBaud |

# IEEE 802.15.4 Device Classes and Terminologies

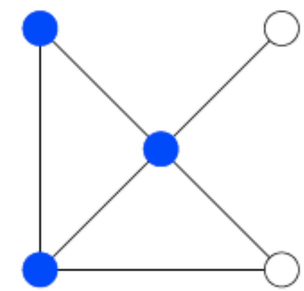
- Full Function Device (FFD)
  - Any topology
  - PAN coordinator capable
  - Talks to any other device
  - Implements complete protocol set
- Reduced Function Device (RFD)
  - Reduced protocol set
  - Very simple implementation
  - Cannot become a PAN coordinator
  - Limited to act as a leaf in more complex topologies
  - Expected to sleep most of the time to conserve energy
- Network Device
  - An RFD or FFD implementation containing an IEEE802.15.4 MAC and PHY interface to the wireless medium
- Coordinator
  - An FFD with Network Device functionality that provides coordination and other services to the network
- PAN Coordinator
  - A Coordinator that is the principal controller of the PAN. Each network has exactly ONE PAN coordinator

# IEEE 802.15.4 Topologies

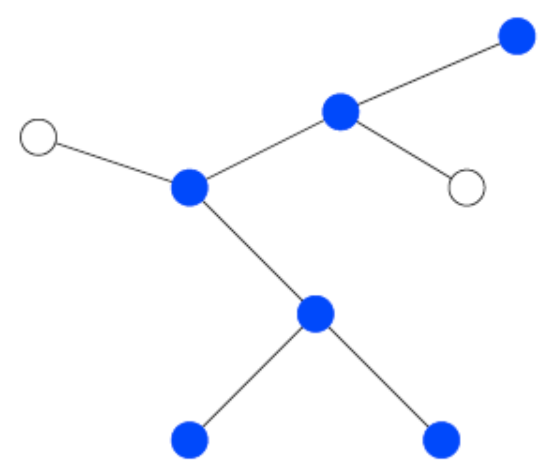
- Star Topology
  - All nodes communicate via the central PAN Coordinator
  - Leafs may be any combination of FFD or RFD
  - PAN coordinator is usually having a reliable power source

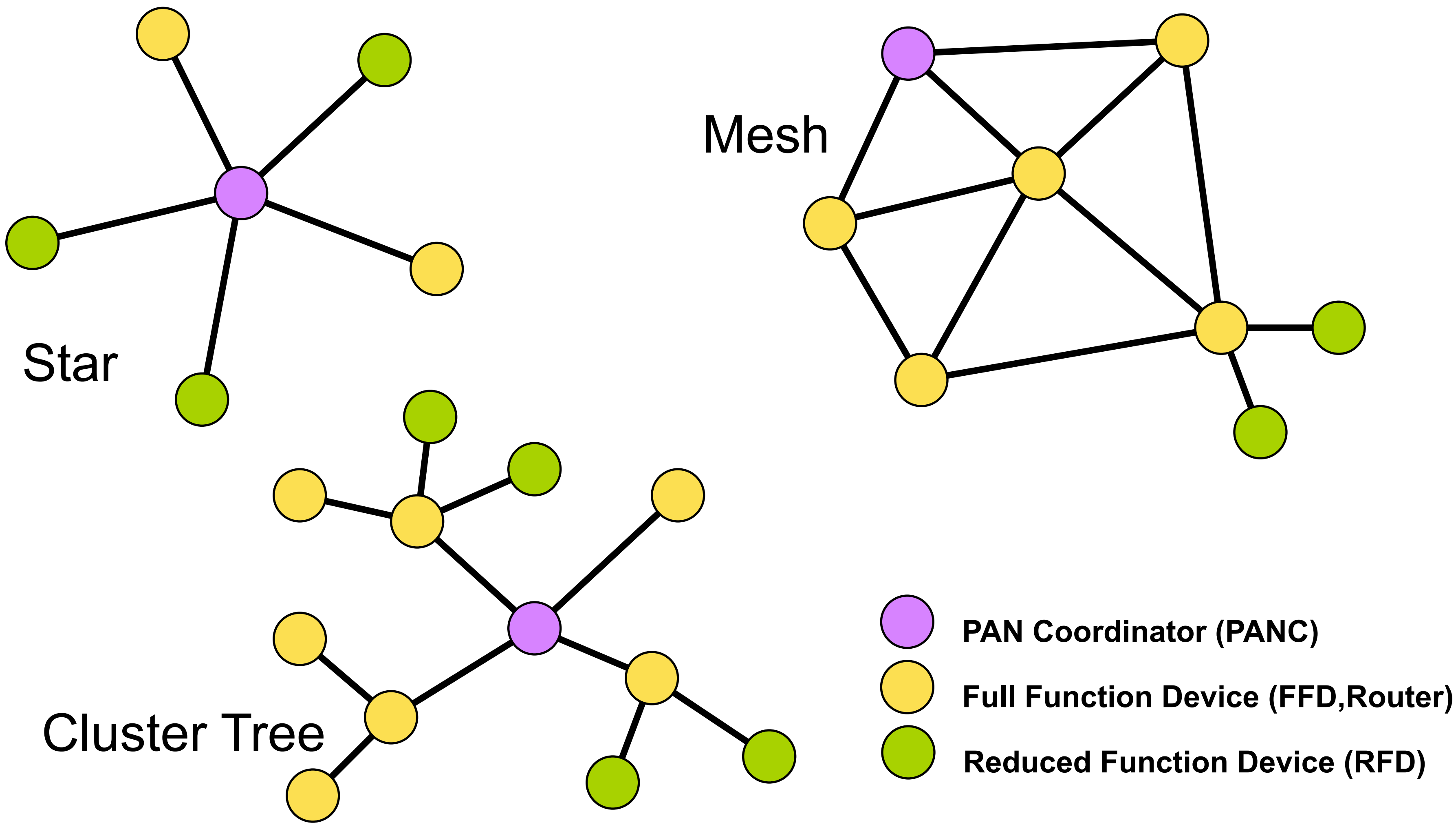


- Peer-to-Peer Topology
  - Extension of the pure star topology
  - Nodes can communicate via the central PAN Coordinator and via additional point-to-point links

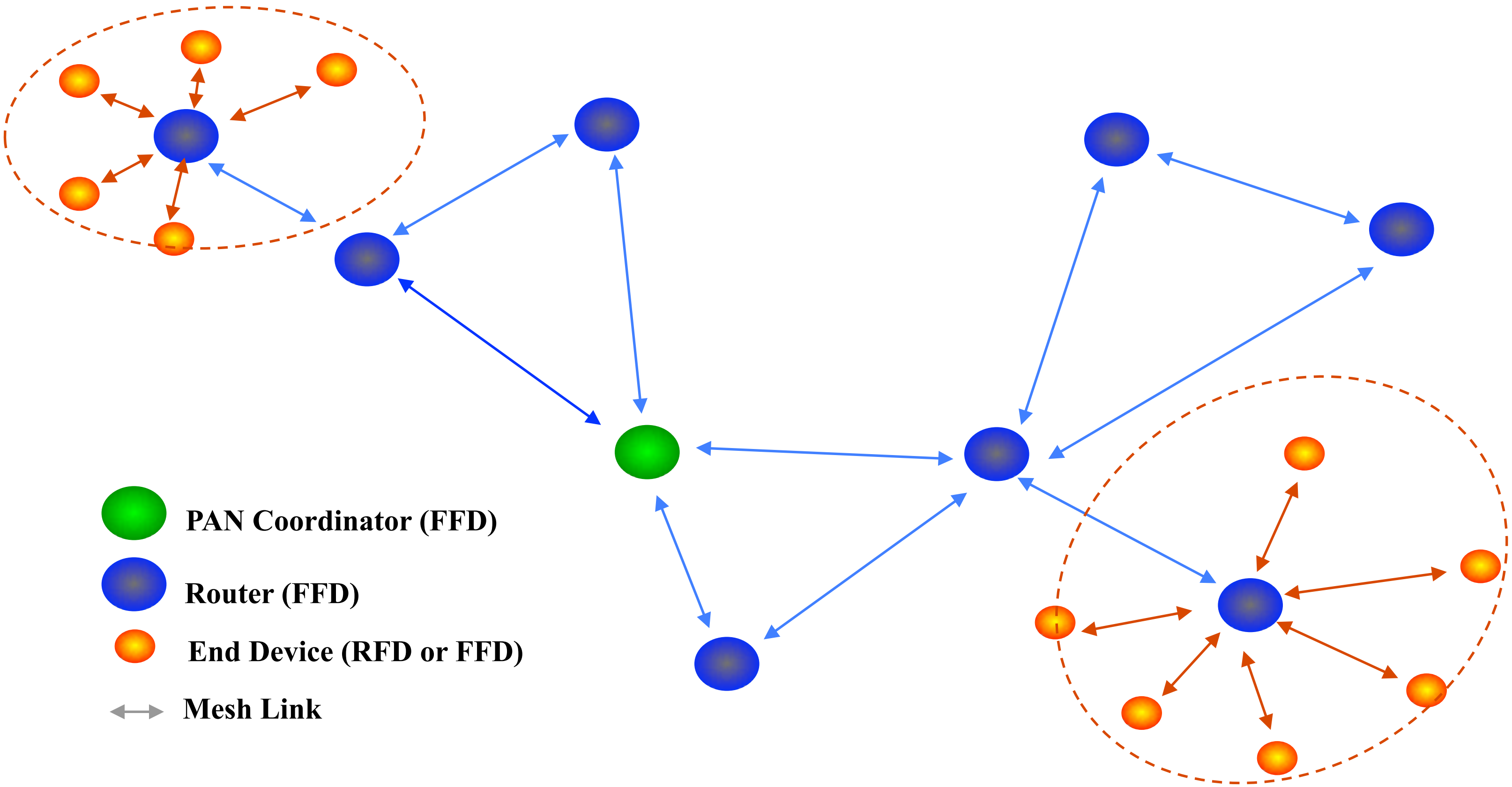


- Cluster Tree Topology
  - Leafs connect to the network of Coordinators (FFDs)
  - One of the coordinators serves as the PAN Coordinator
  - Clustered star topologies are an important case, e.g. each hotel room forms a star in a HVAC system




Star
Mesh
Cluster Tree
PAN Coordinator (PANC)
Full Function Device (FFD,Router)
Reduced Function Device (RFD)

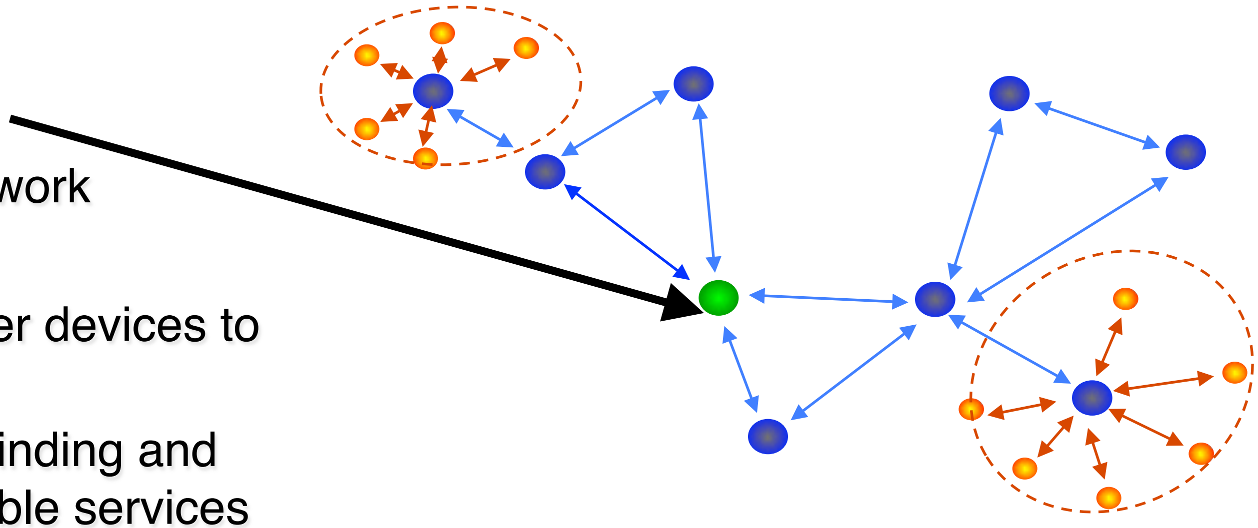
# IEEE 802.15.4 Network Model




- Star networks support a single (PAN) Coordinator with one or more End Devices (up to 65,536 in theory)
- Mesh network routing permits path formation from any source device to any destination device

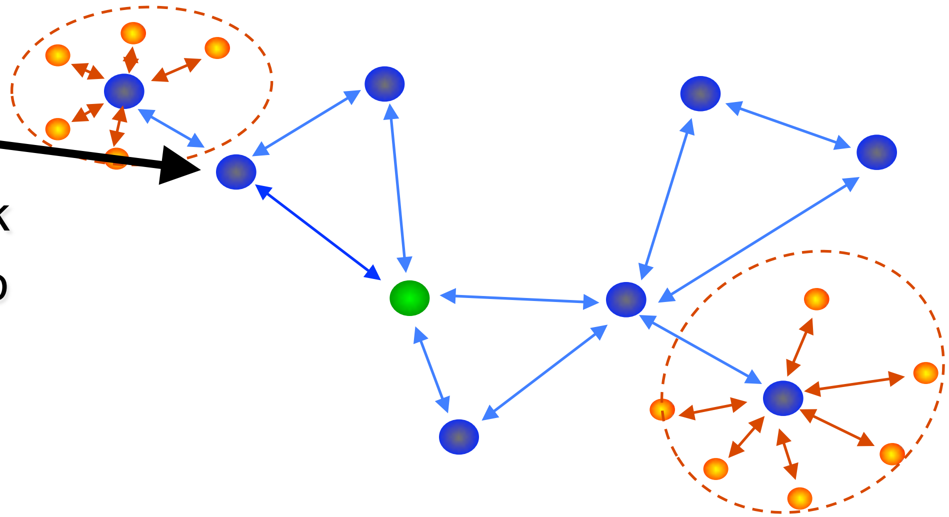
# Network Pieces –PAN Coordinator

- PAN Coordinator
  - "owns" the network
    - Starts it
    - Allows other devices to join it
    - Provides binding and address-table services
    - Saves messages until they can be delivered
    - And more… could also have i/o capability
  - A "full-function device" – FFD
  - Mains powered



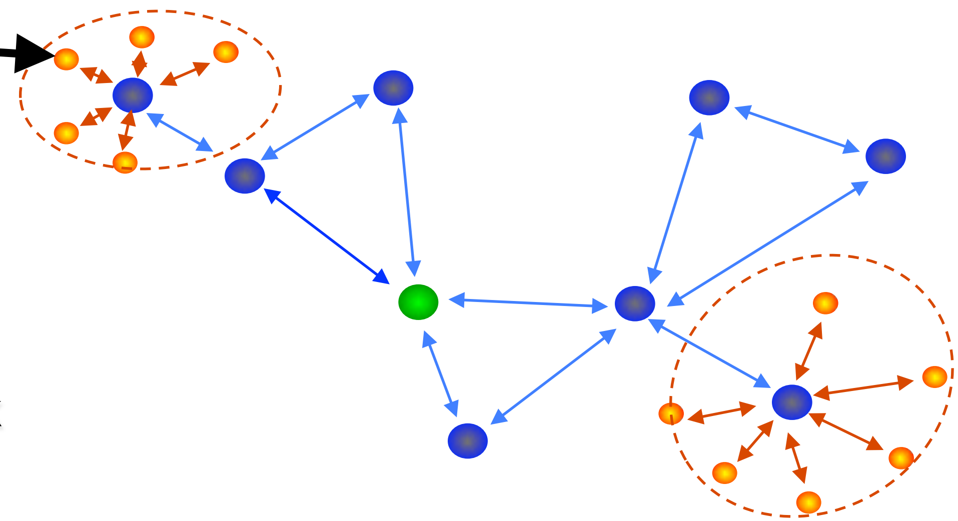
# Network Pieces - Router

- Router
  - Routes messages
  - Does not own or start network
    - Scans to find a network to join
      - Given a block of addresses to assign
  - A "full-function device" – FFD
  - Mains powered depending on topology
  - Could also have i/o capability



# Network Pieces – End Device

- End Device
  - Communicates with a single device
  - Does not own or start network
    - Scans to find a network to join
  - Can be an FFD or RFD (reduced function device)
  - Usually battery powered



# IEEE 802.15.4 Frame Formats

## General Frame Format

| octets: 2 | 1 | 0/2 | 0/2/8 | 0/2 | 0/2/8 | variable | 2 |
|---|---|---|---|---|---|---|---|
| Frame control | Sequence number | Destination PAN identifier | Destination address | Source PAN identifier | Source address | Frame payload | Frame sequence check |

| bits: 0–2 | 3 | 4 | 5 | 6 | 7–9 | 10–11 | 12–13 | 14–15 |
|---|---|---|---|---|---|---|---|---|
| Frame type | Security enabled | Frame pending | Ack. requested | Intra PAN | Reserved | Dst addr mode | Reserved | Src addr mode |

- IEEE 64-bit extended addresses
- 16-bit “short” addresses (unique within a PAN)
- Optional 16-bit source/destination PAN identifiers
- Max. frame size = 127 octets ; Max. frame headers = 25 octets

# IEEE 802.15.4 Frame Formats (cont'd)

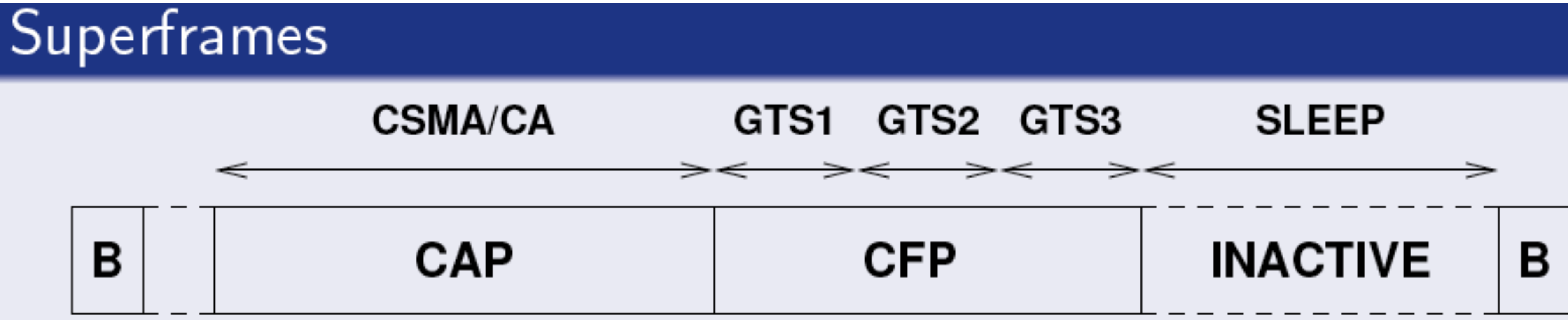
- Beacon Frames
  - Broadcasted by the Coordinator to organize the network
- Command Frames
  - Used for Association, Disassociation, Data and Beacon Requests, Conflict Notification
- Data Frames
  - Carrying User Data
- Acknowledgement Frames
  - Acknowledges successful Data Transmission (if requested)

# IEEE 802.15.4 MAC

- Carrier Sense Multiple Access / Collision Avoidance
  - First wait until the Channel is Idle
  - Once the Channel is free, start sending the data frame after some random backoff interval
  - Receiver acknowledges the correct reception of a data frame
  - If the sender does not receive an acknowledgement, retry the data transmission

- Unslotted Mode:
  - Node -> PAN ; Node -> Node
    - The sender uses CSMA/CA and the receiver sends an ACK if requested by the sender
    - Receiver needs to listen continuously and CANNOT sleep
  - PAN -> Node
    - The receiver polls the PAN whether data is available
    - The PAN sends an ACK followed by a Data Frame
    - Receiving nodes sends an ACK if requested by the sender
    - Coordinator needs to listen continuously and CANNOT sleep

# IEEE 802.15.4 MAC Slotted Mode




- A superframe consists of 3 periods:

  1. During the Contention-Access-Period (CAP), the channel can be accessed using normal CSMA/CD

  2. The Contention-Free-Period (CFP) has Guaranteed Time Slots (GTS) assigned by the PAN to each node

  3. During the Inactive-Period (IP), the channel is not used and all nodes including the Coordinator can sleep

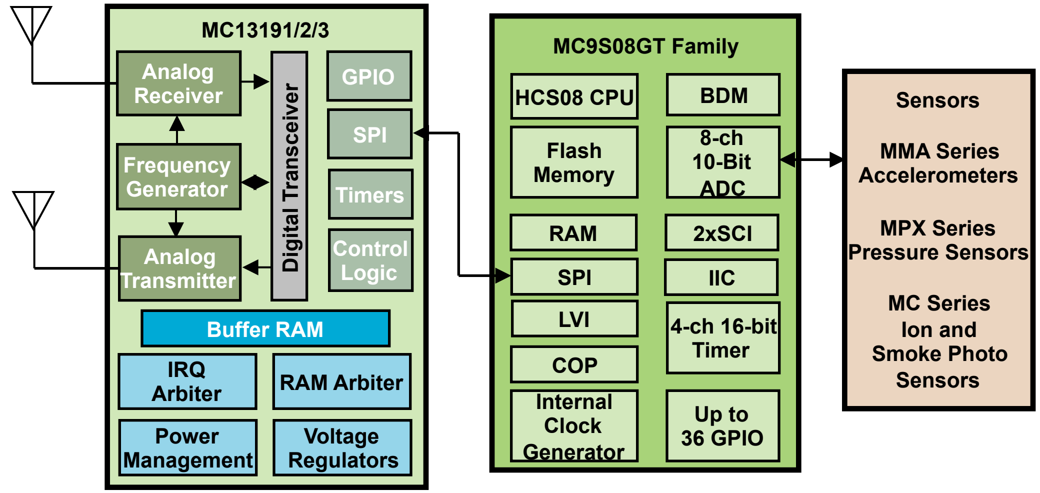
- The PAN delimits superfames using Beacons

# IEEE 802.15.4 Security Services

| Security Suite | Description |
|---|---|
| Null | No security (default) |
| AES-CTR | Encryption only, CTR Mode |
| AES-CBC-MAC-128 | 128 bit MAC |
| AES-CBC-MAC-64 | 64 bit MAC |
| AES-CBC-MAC-32 | 32 bit MAC |
| AES-CCM-128 | Encryption and 128 bit MAC |
| AES-CCM-64 | Encryption and 64 bit MAC |
| AES-CCM-32 | Encryption and 32 bit MAC |

- Key Management must be provided by Higher Layers
- Implementations must support AES-CCM-64 and Null

# 802.15.4 Radio Example from Freescale




- Key Features
  - IEEE® 802.15.4 Compliant
    - 2.4GHz
    - 16 selectable channels
    - 250Kbps Data Rate
    - 250Kbps 0-QPSK DSSS
  - Multiple Power Saving Modes
    - Hibernate 2.3uA
    - Doze 35uA
    - Idle 500uA
  - RF Data Modem
  - Up to 7 GPIO
  - SPI Interface to Micro
  - Internal Timer comparators (reduce MCU resources)
  - -16.6dBm to +3.6dBm output power
    - Software selectable
    - On-chip regulator
  - Up to -92dB Rx sensitivity at 1% PER
  - 2V to 3.4 operating voltage
  - -40˚C to +85˚C operating temperature
  - Low external component count
    - Requires single 16Mhz crystal
  - 5mmx5mm QFN-32
    - Lead-Free

# Reading List for IEEE 802.15.4

IEEE.
IEEE Std 802.15.4-2003.
Technical Report 802.15.4-2003, IEEE, October 2003.

IEEE.
IEEE Std 802.15.4-2006.
Technical Report 802.15.4-2006, IEEE, September 2006.

IEEE.
IEEE Std 802.15.4a-2007.
Technical Report 802.15.4a-2007, IEEE, August 2007.

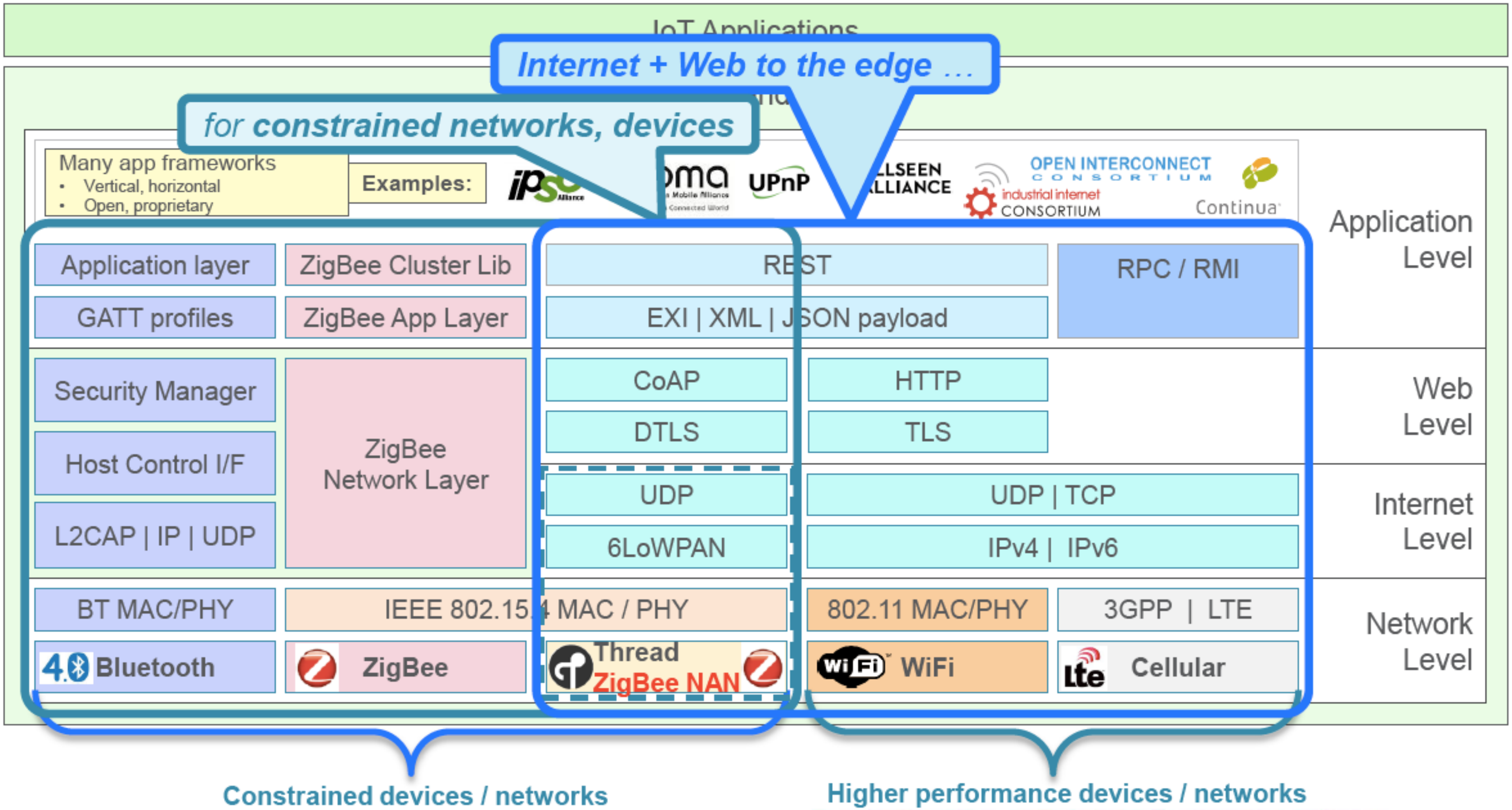
Y. Xiao, H.-H. Chen, B. Sun, R. Wang, and S. Sethi.
MAC Security and Security Overhead Analysis in the IEEE 802.15.4 Wireless Sensor Networks.
*Journal on Wireless Communications and Networking*, 2006:1–12, 2006.

E. Callaway, P. Gorday, L. Hester, J. A. Gutierrez, M. Naeve, B. Heile, and V. Bahl.
Home Networking with IEEE 802.15.4: A Developing Standard for Low-Rate Wireless Personal Area Networks.
*IEEE Communications Magazine*, 40(8):70–77, August 2002.

L. D. Nardis and M.-G. Di Benedetto.
Overview of the IEEE 802.15.4/4a standards for low data rate Wireless Personal Data Networks.
In *Proc. of the 4th IEEE Workshop on Positioning, Navigation and Communication 2007 (WPNC'07)*, Hannover, March 2007. IEEE.

S. Labella M. Petrova, J. Riihijarvi, P. Mahonen.
Performance Study of IEEE 802.15.4 Using Measurements and Simulations.
In *Proc. IEEE Wireless Communications and Networking Conference (WCNC 2006)*, pages 487–492, 2006.

# Recall: Some Mainstream IoT Standard Protocol Stacks

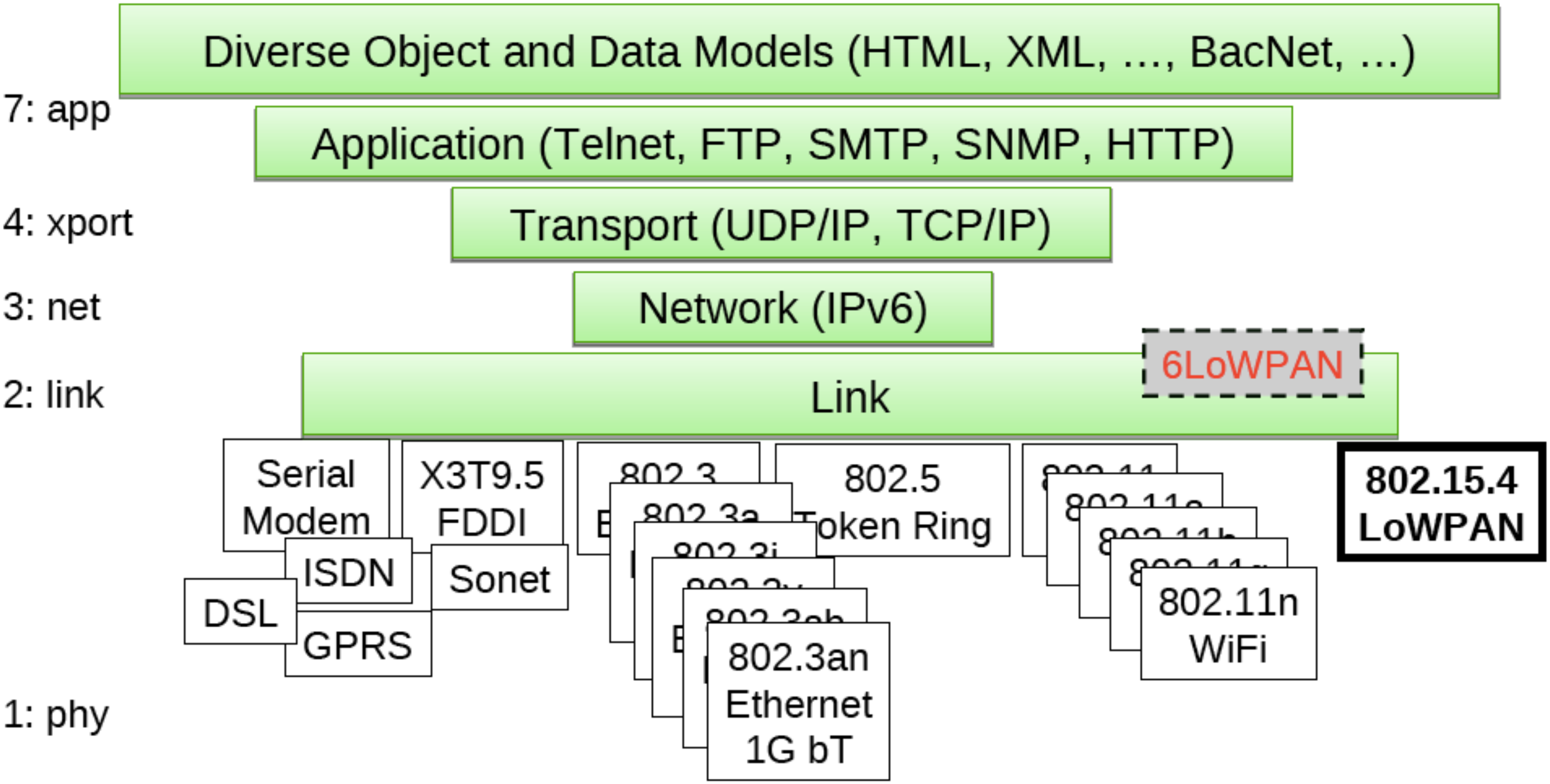


# IPv6 over IEEE 802.15.4 (6LoWPAN)

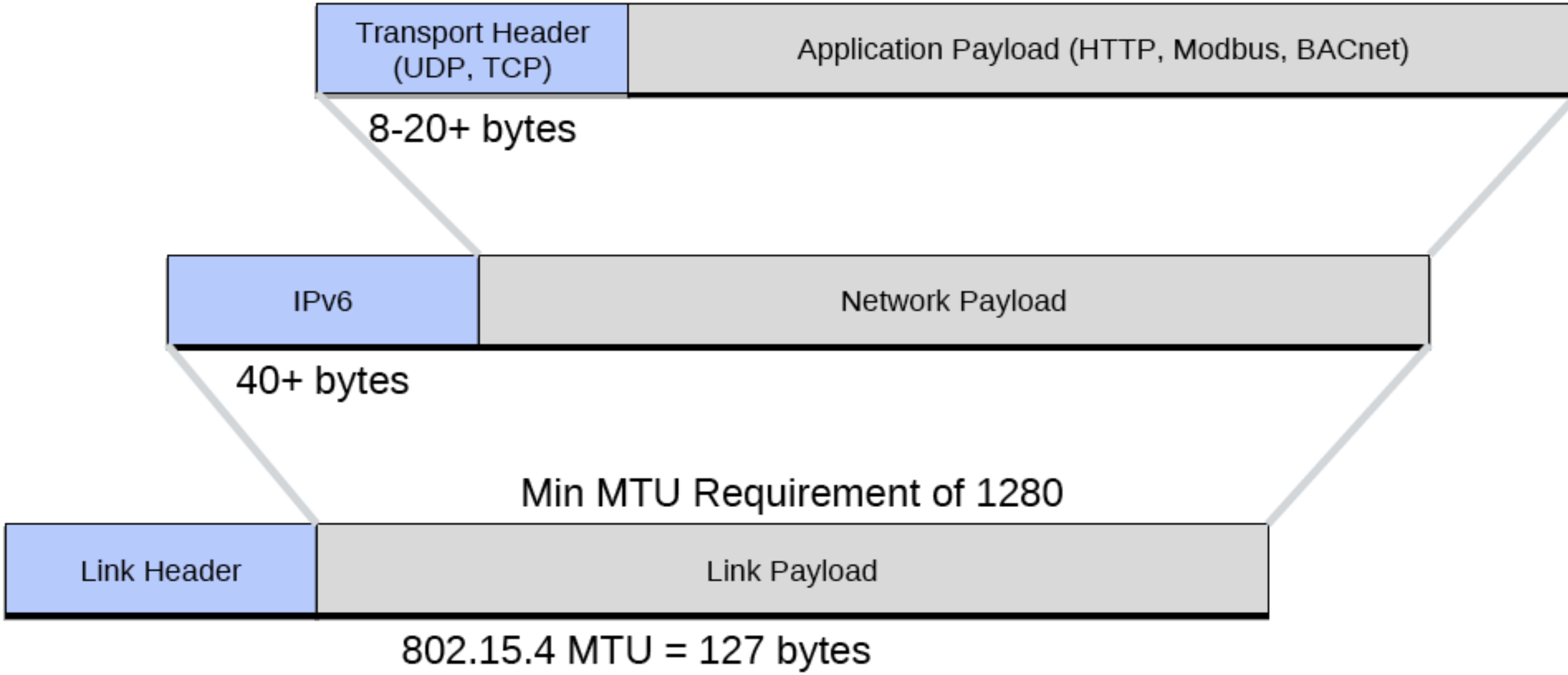
# Motivation/Benefits of IPv6 over 802.15.4

- Let IoT leverage pervasive nature of IP networks
- Open and Freely Available Specifications vs. Proprietary Solutions (i.e. ZigBee)
- Tools for Diagnostics, Management for IP networks already exist
- IP-based devices can be connected readily to other IP-based networks without the need for intermediate entities like Translation Gateways or Proxies (as in the case of the ZigBee approach – at least before ZigBee IP was introduced)
- Due to the expected huge volume of IoT devices, IPv6 is a MUST

# 6LoWPAN (RFC4919): An Adaptation Layer (Layer 2.5)



# 6LoWPAN Adaptation Needs




- IPv6 MTU (1280 octets) >> 802.15.4 MTU (127 octets)
- 48+ bytes UDP+IPv6 Header => Need Header Compression

# Challenges for 6LoWPAN

**High per-packet IPv6/UDP overhead**

- 40-octet IPv6 header and 8-octet UDP header
- 802.15.4 MAC header can be up to 25 octets (Null security) or 25+21 = 46 octets (AES-CCM-128)
- With the 802.15.4 frame size of 127 octets, we ONLY have:
  - 127-25-40-8 = 54 octets (Null security) or
  - 127-46-40-8 = 33 octets (AES-CCM-128)

  of space per packet left for payload, i.e. application data

=> IPv6/UDP Header Compression is needed

**IPv6 MTU Requirements**

- IPv6 requires that links support a Min. MTU of 1280 octets >> MTU of 802.15.4
- Link-layer fragmentation / Reassembly is needed

# Overview of 6LowPAN (RFC4944,6282)

- 6LoWPAN protocol is an adaptation layer allowing to transport IPv6 packets over 802.15.4 networks
- Uses 802.15.4 in Unslotted CSMA/CA mode
  - Strongly suggests Beacons for Link-layer Device Discovery
- Based on IEEE standard 802.15.4-2003/2006
- Fragmentation / Reassembly of IPv6 packets
- Mostly Stateless Compression of IPv6 and UDP/ICMP headers
- Mesh Routing Support (mesh under the multi-hop Layer 2 802.15.4 networks)

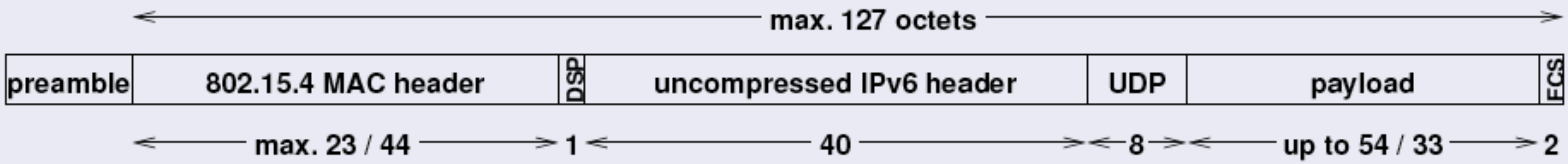
# 6LoWPAN Dispatch Codes

- All 6LowPAN encapsulated datagrams are prefixed by an encapsulation header stack
- Each header in the stack starts with a header type field followed by zero or more header fields
  - Similar to IPv6 "Next-Header" chaining approach

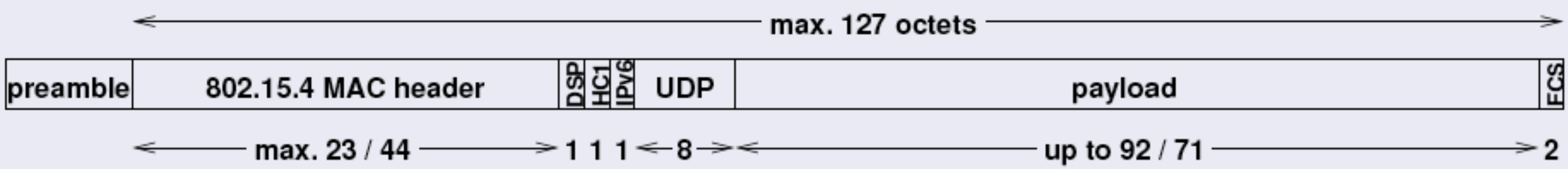
| Bit Pattern | Short Code | Description |
|---|---|---|
| 00 xxxxxx | NALP | Not A LoWPAN Packet |
| 01 000001 | IPv6 | uncompressed IPv6 addresses |
| 01 000010 | LOWPAN_HC1 | HC1 Compressed IPv6 header |
| 01 010000 | LOWPAN_BC0 | BC0 Broadcast header |
| 01 111111 | ESC | Additional Dispatch octet follows |
| 10 xxxxxx | MESH | Mesh routing header |
| 11 000xxx | FRAG1 | Fragmentation header (first) |
| 11 100xxx | FRAGN | Fragmentation header (subsequent) |

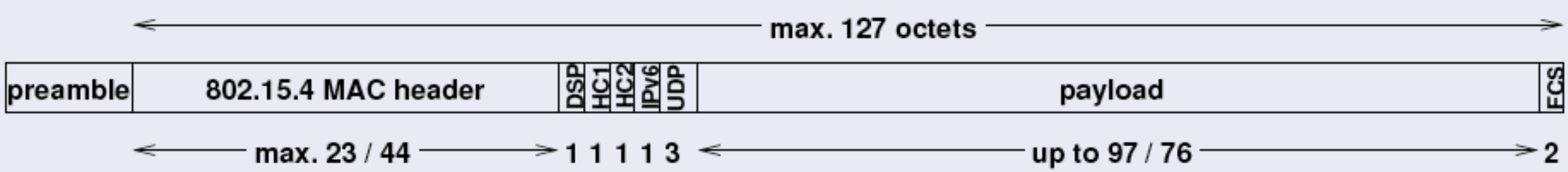
# 6LowPAN Frame Formats

## Uncompressed IPv6/UDP (worst case scenario)



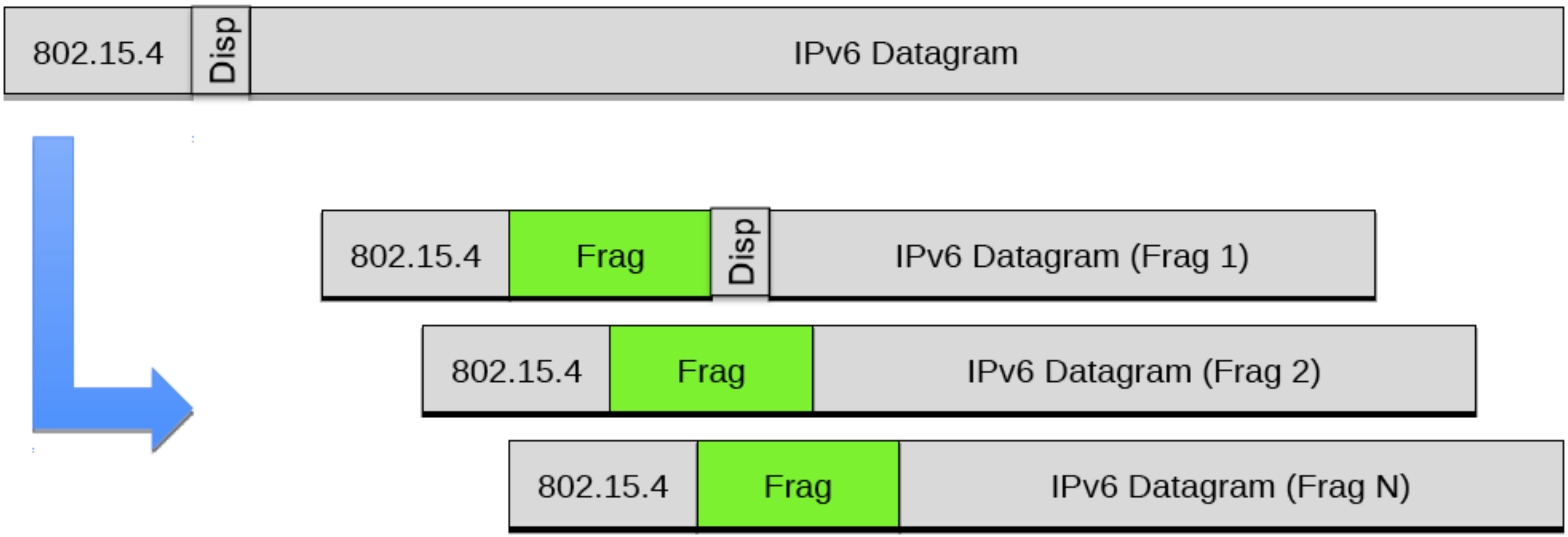

## Compressed Link-local IPv6/UDP (best case scenario)



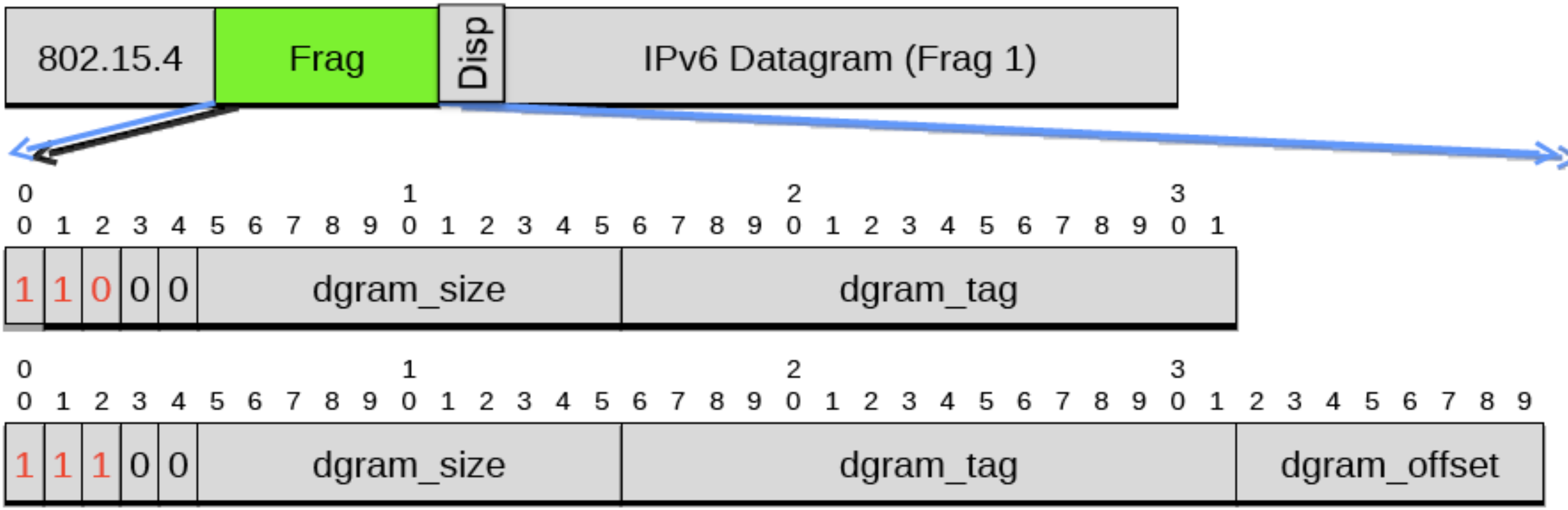




- This shows the max. compression achievable for link-local addresses ; Does not work for Global addresses
- Any non-compressable header fields are carried after HC1 or HC1/HC2 tags (partial compression)

# 6LoWPAN Fragmentation/Reassembly

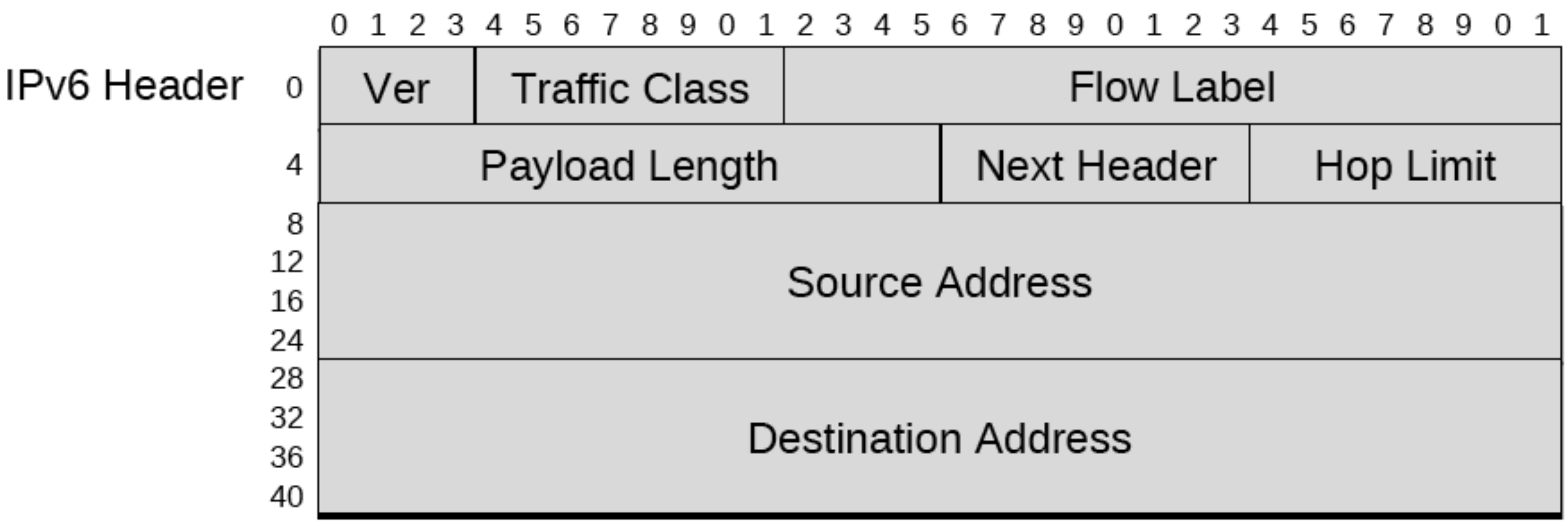


# 6LoWPAN Fragmentation/Reassembly




- dgram_size: Size of datagram in bytes
  - Included in all fragments to simplify buffer allocation
- dgram_tag: Identifies all fragments of a datagram
- dgram_offset: Location of fragments in 8-byte units
  - Omitted in the 1st fragment

# 6LoWPAN Header Compression



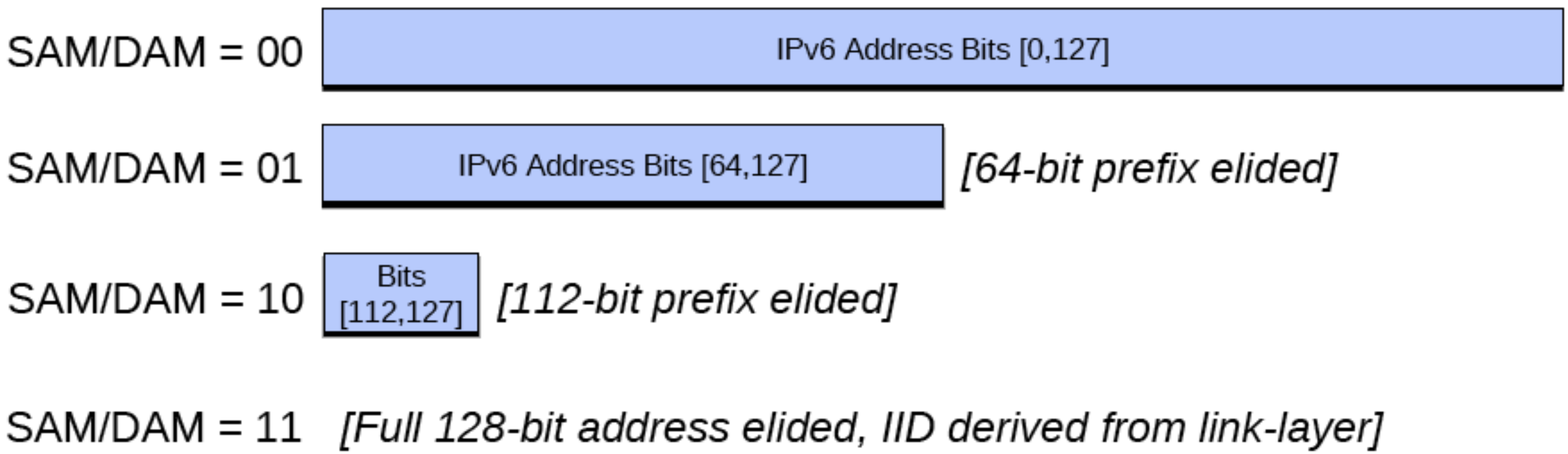

- Omit any header field that can be calculated from the context, send the remaining fields unmodified
- Nodes do not need to (or only maintain very little) compression state (i.e. stateless compression)
- Support (almost) arbitrary combinations of compressed/ uncompressed header fields.
- Common values for IPv6 header fields:
  - Version is always 6
  - Traffic Class and Flow Label are all zeros
  - Payload Length always derived from Layer 2 header
  - Source and Destination Address are link-local ones and derived from L2 addresses

# 6LoWPAN Header Compression for IPv6 Unicast Address


- Prefix
  - Addresses within 6LoWPAN typically contain common prefix
  - Nodes typically communicate with one or few central devices
  - Establish State (i.e. Context) for such prefixes
    - This is the ONLY State Maintenance
    - Support up to 16 contexts
- Interface Identifier
  - Typically derived from Layer 2 address during IPv6 address auto-configuration
  - Omit when Interface Identifier can be derived from L2 address

# 6LoWPAN Header Compression for IPv6 Unicast Address


- Source/Destination Address Mode (SAM/DAM)



# 6LoWPAN Header Compression for Prefix of IPv6 Unicast Address

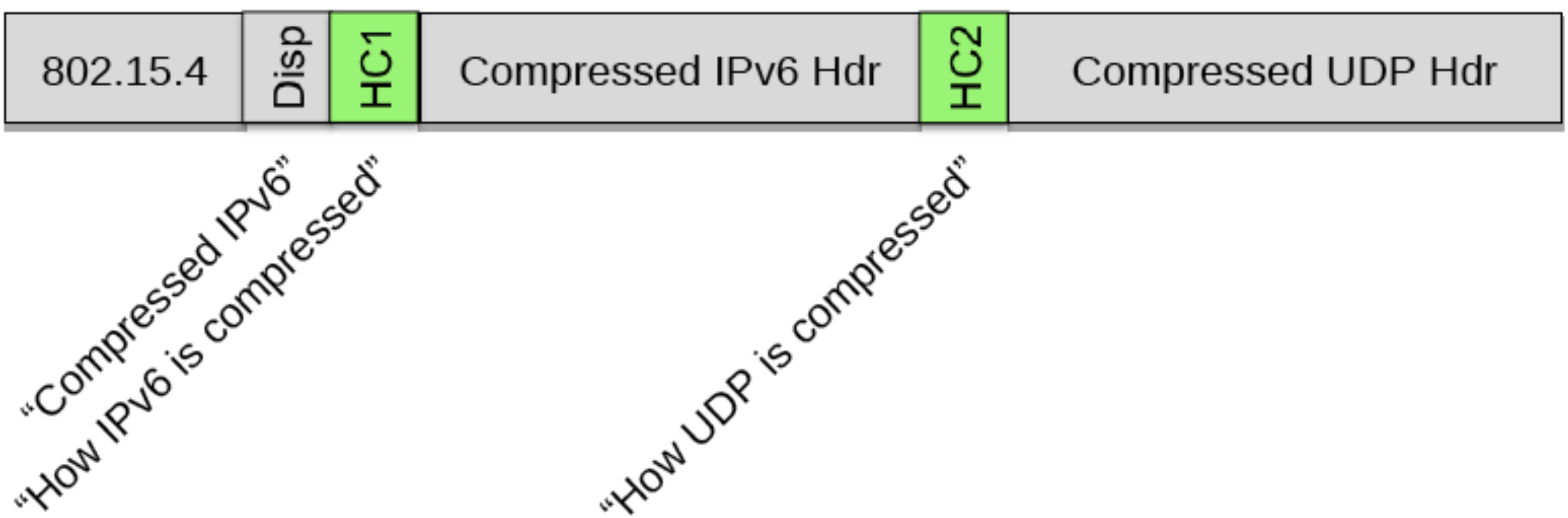

Compression for Link-local or Global IPv6 Prefixes:

- Stateless Mode (SAC/DAC=0)
  - Prefix is link-local (FE80::/10)
- Context-based Mode (SAC/DAC=1)
  - Prefix taken from stored contexts (Up to 16 contexts)
  - CID = 0, use ContextID = 0
  - CID = 1, include 4-bit ContextID for source and destination

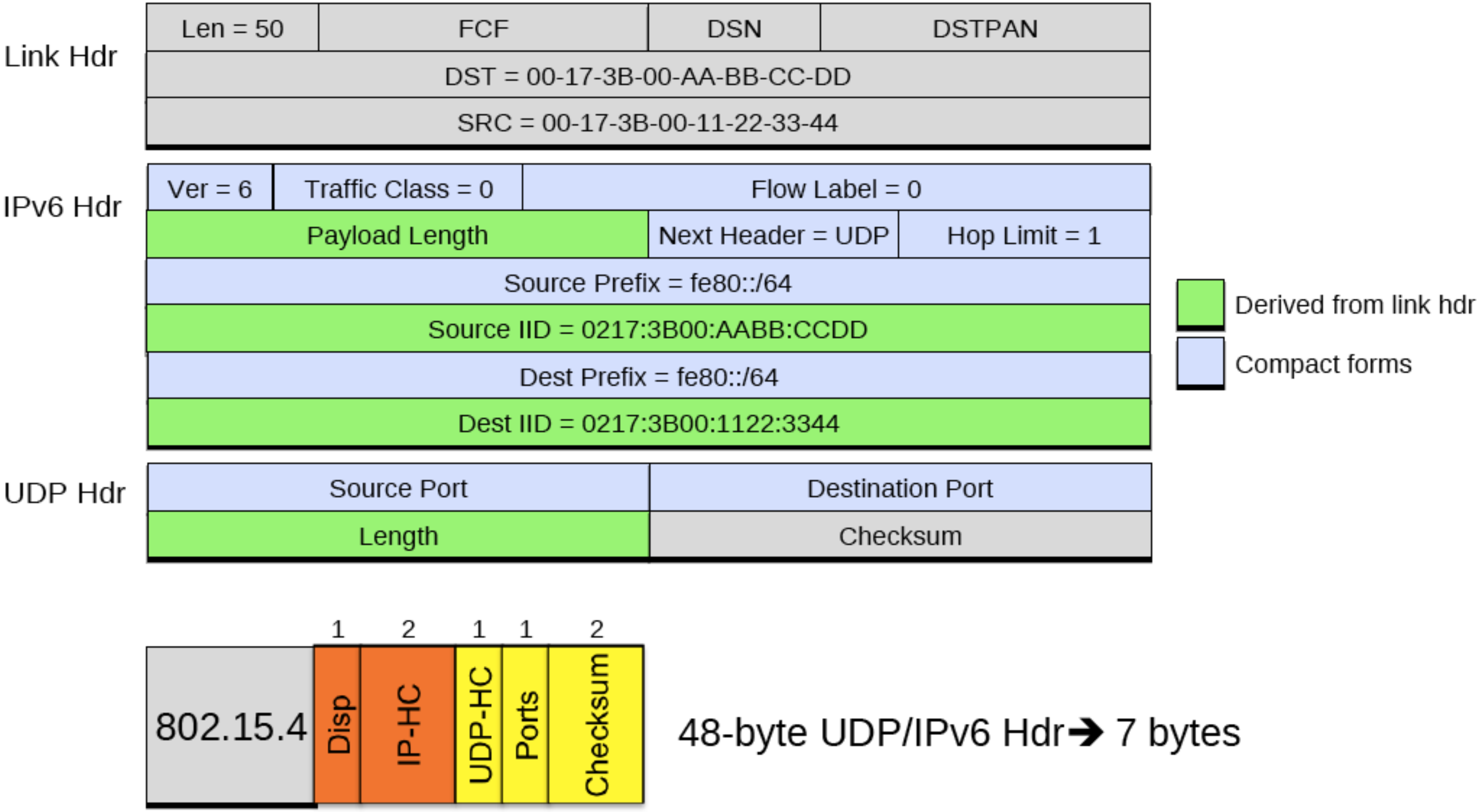
# 6LoWPAN Header Compression

- Each compressed header indicates if the next header is also compressed
- Following control byte(s) include next header identifier

=> Provide a framework for defining arbitrary Next Header compression methods

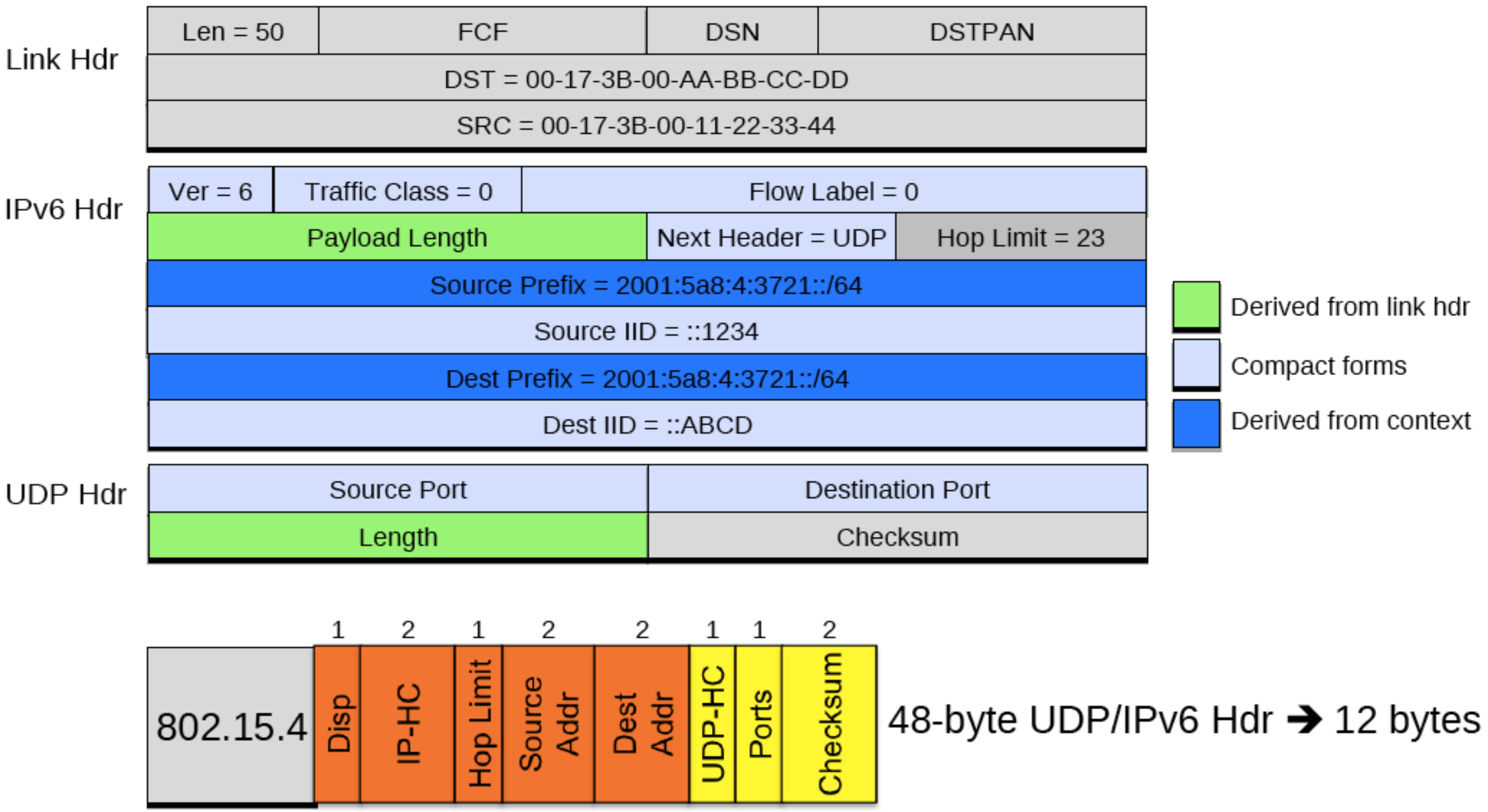


# Example: 6LoWPAN Header Compression for Link-Local Unicast Packet

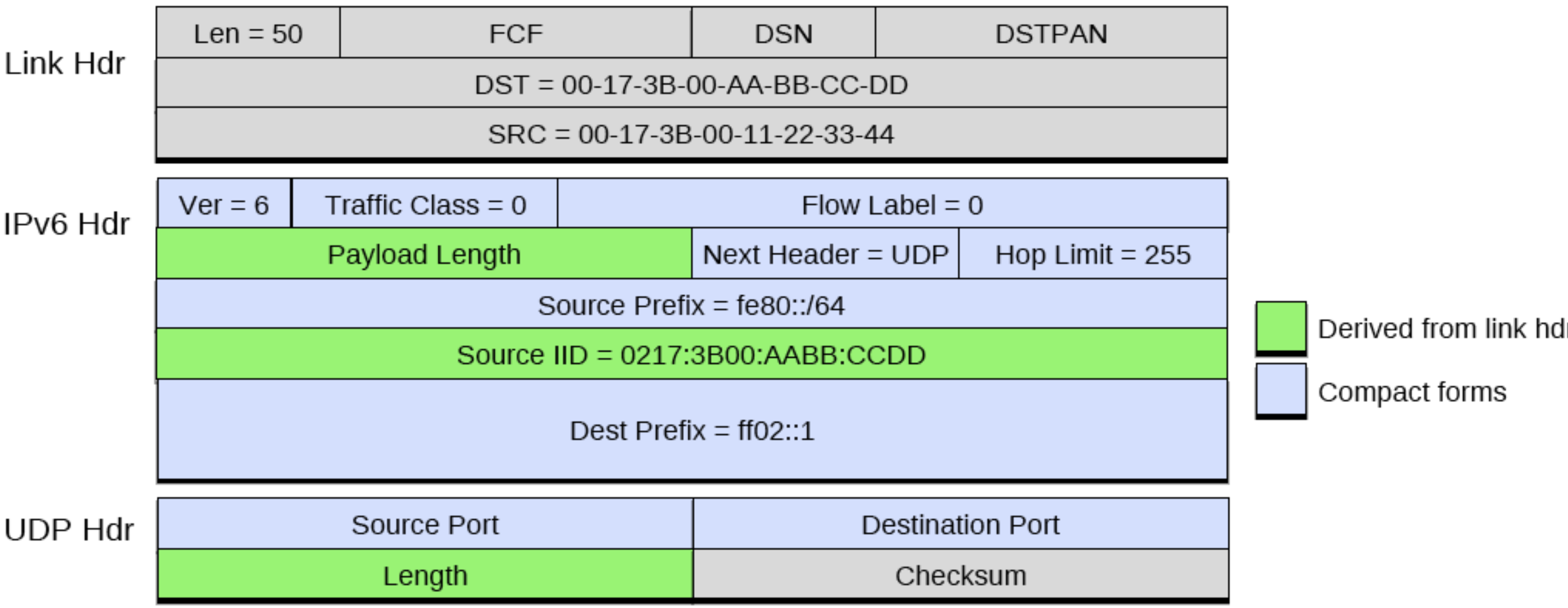


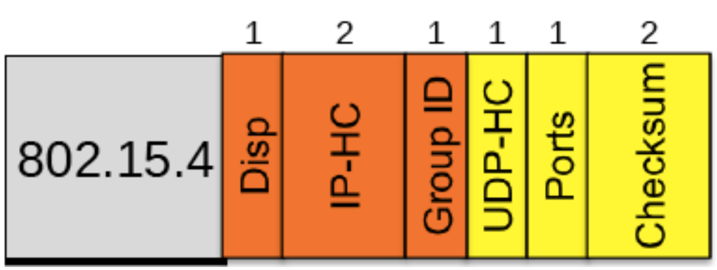

48-byte UDP/IPv6 Hdr → 7 bytes

# Example: 6LoWPAN Header Compression for Global Unicast Packet



# Example: 6LoWPAN Header Compression for Link-Local Multicast Packet



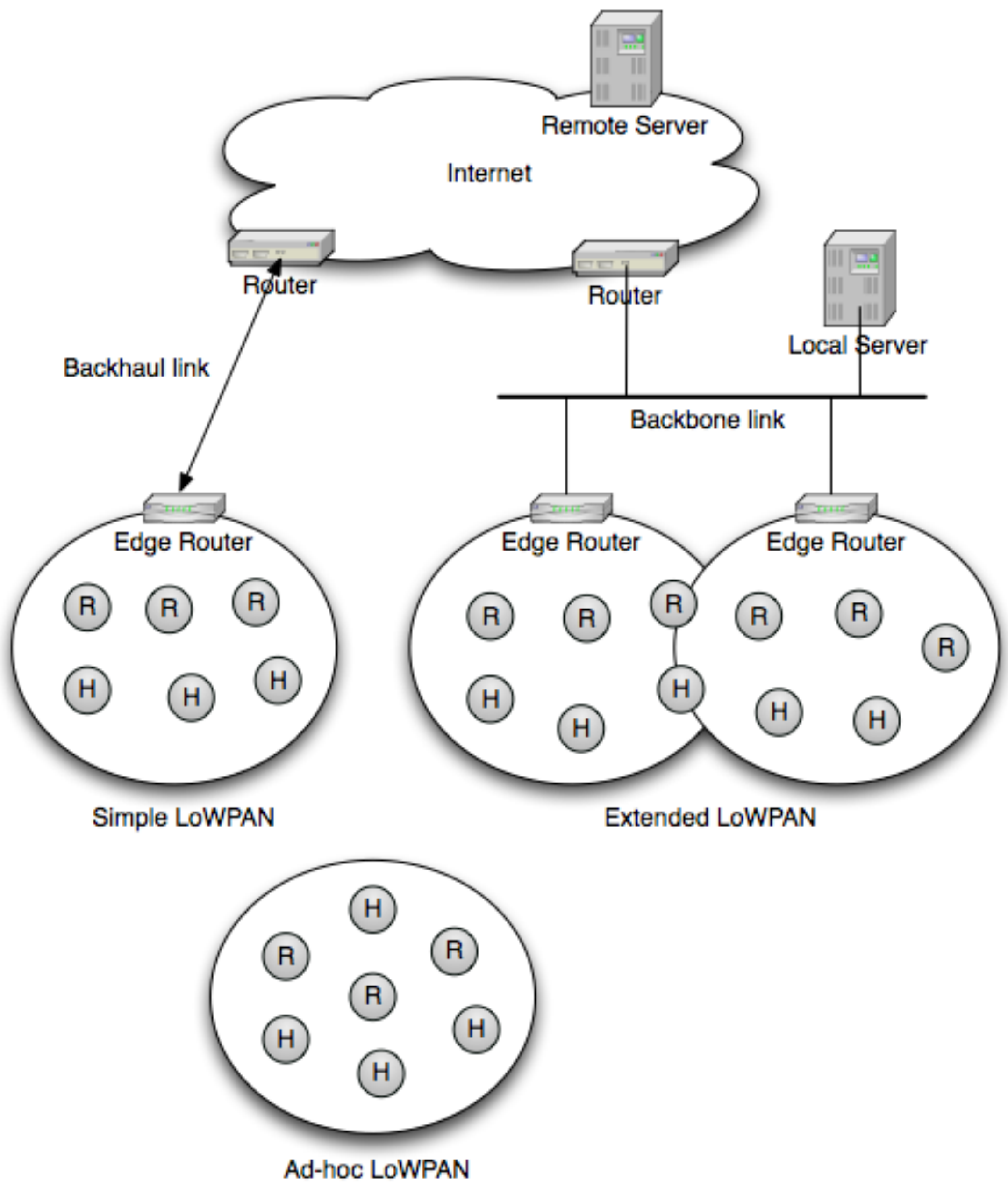




48-byte UDP/IPv6 Hdr → 8 bytes

# Additional Link-Layers (other than 802.15.4) for 6LoWPAN

- Sub-GHz Industrial, Scientific and Medical band radios
  - Typically 10-50 kbps data rates, longer range than 2.4 GHz
  - Usually use CSMA-style medium access control
  - Example: CC1110 from Texas Instruments
- Power-Line Communications
  - Some PLC solutions behave like an 802.15.4 channel
  - Example: A technology from Watteco provides an 802.15.4 emulation mode, allowing the use of 6LoWPAN
- Z-Wave
  - A home-automation low-power radio technology

# Architecture with 6LoWPAN Networks



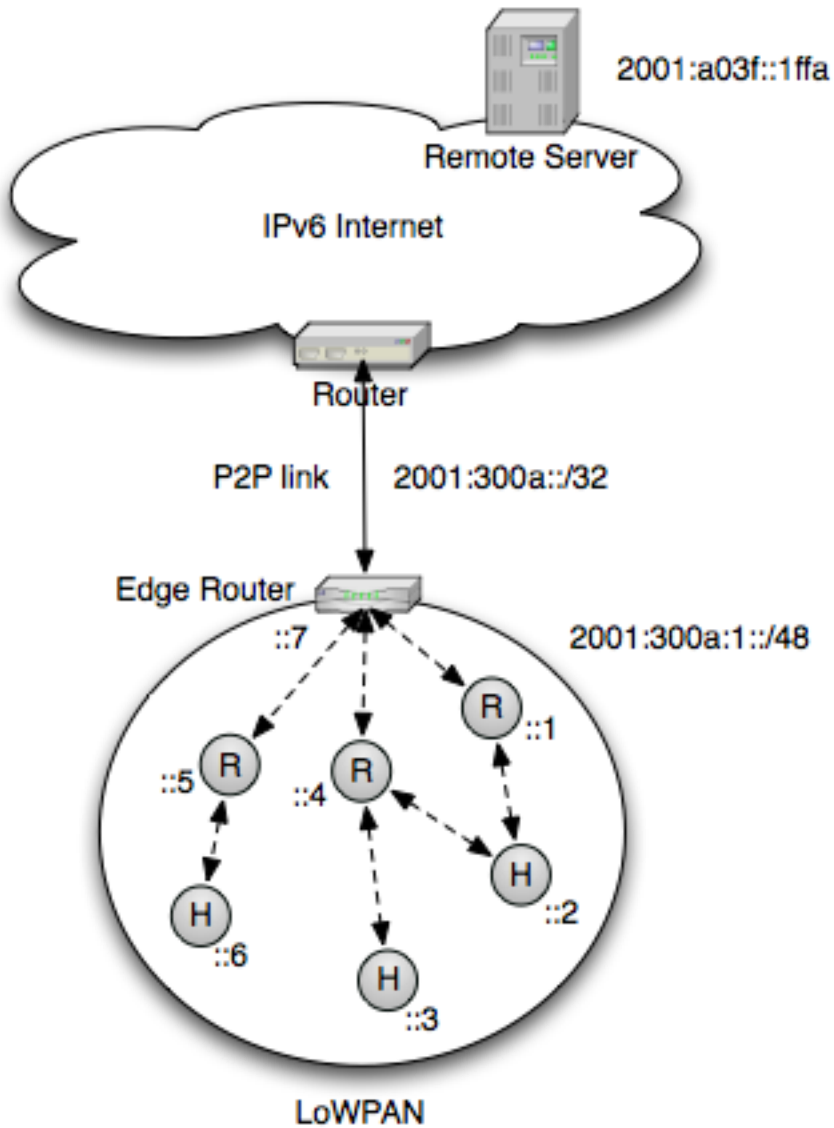
# 6LoWPAN Architecture

- LoWPANs are stub networks
- Simple LoWPAN
  - Single Edge Router
- Extended LoWPAN
  - Multiple Edge Routers with common backbone link
- Ad-hoc LoWPAN
  - No route outside the LoWPAN
- Internet Integration issues
  - Maximum transmission unit
  - Application protocols
  - IPv4 interconnectivity
  - Firewalls and NATs
  - Security

| IPv6 | |
|---|---|
| Ethernet MAC | LoWPAN Adaptation |
| | IEEE 802.15.4 MAC |
| Ethernet PHY | IEEE 802.15.4 PHY |

**IPv6-LoWPAN Router Stack**

# 6LoWPAN Addressing Example



# 6LoWPAN Setup & Operation

- Auto-configuration is important in embedded networks
- In order for a 6LoWPAN network to start functioning:
  - 1. Link-layer connectivity between nodes (commissioning)
  - 2. Network layer address configuration, discovery of neighbors, registrations (bootstrapping)
  - 3. Routing algorithm sets up paths (route initialization)
  - 4. Continuous maintenance of 1-3

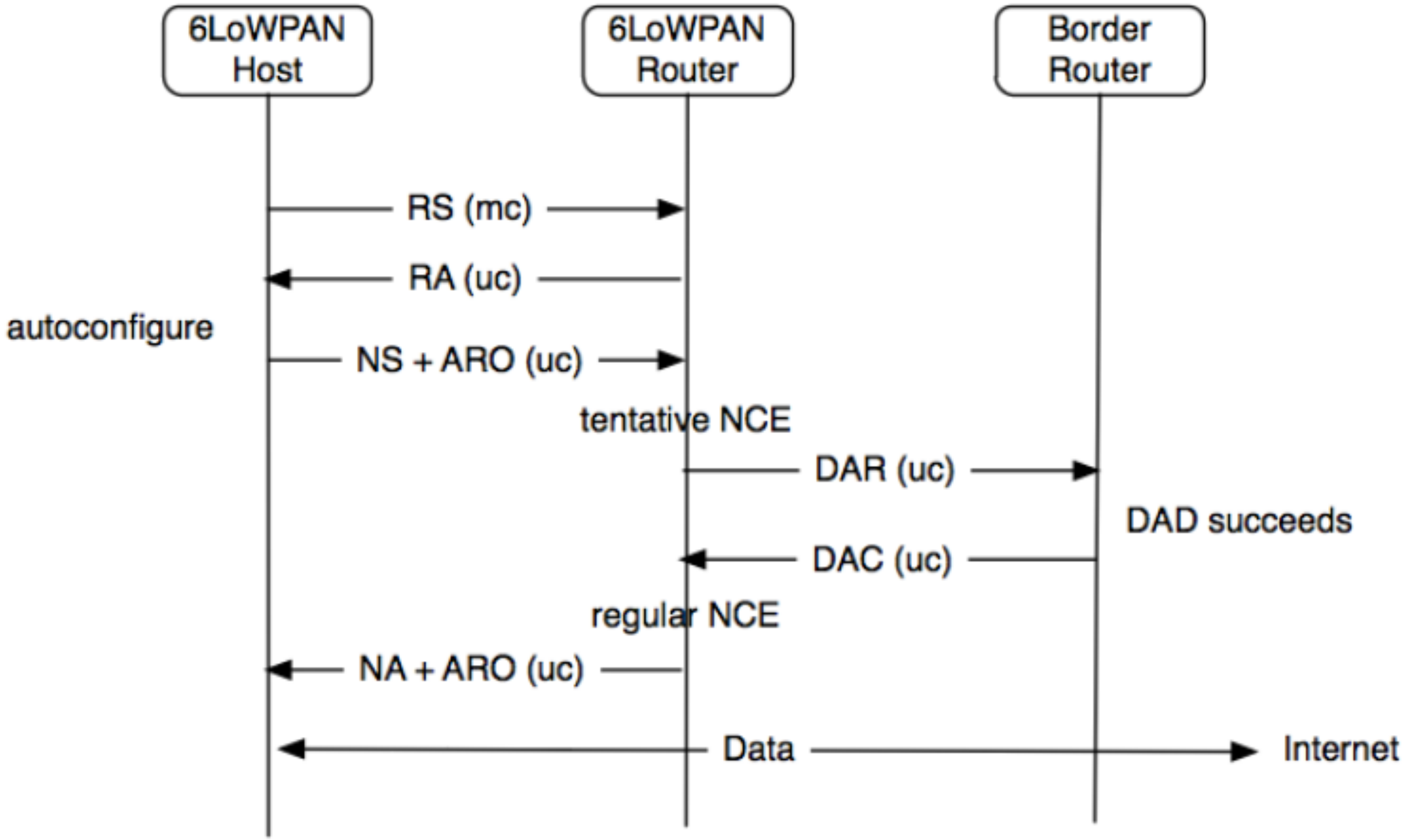
# Link-layer Commissioning

- In order for nodes to communicate with each other, they need to have compatible physical and link-layer settings.
- Example IEEE 802.15.4 settings:
  - Channel, modulation, data-rate (Channels 11-26 at 2.4 GHz)
    - Usually a default channel is used, and channels are scanned to find a router for use by Neighbor Discovery
  - Addressing mode (64-bit or 16-bit)
    - Typically 64-bit is a default, and 16-bit used if address available
  - MAC mode (beaconless or super-frame)
    - Beaconless mode is easiest for commissioning (no settings needed)
  - Security (on or off, encryption key)
    - In order to perform secure commissioning a default key should already be installed in the nodes

# Neighbor Discovery (ND) in 6LoWPAN

- IPv6 Neighbor Discovery (RFC4862) defines:
  - How hosts discover Routers and Prefixes
  - How nodes resolve L2 addresses from IP addresses
  - How nodes perform unreachability detection
- But ND was originally designed for
  - LAN (e.g. Ethernet) connected interfaces
  - Always-on equipment such as PCs
- 6LoWPAN has unique requirements:
  - Need to support BOTH single-hop mesh and multi-hop IP routed networks
  - Lossy and Asymmetric radio environment
  - Frequent multicast traffic is expensive (energy-wise)
  - Address resolution is not required
  - Unique EUI-64 addresses
  - Hosts may be sleeping to preserve energy

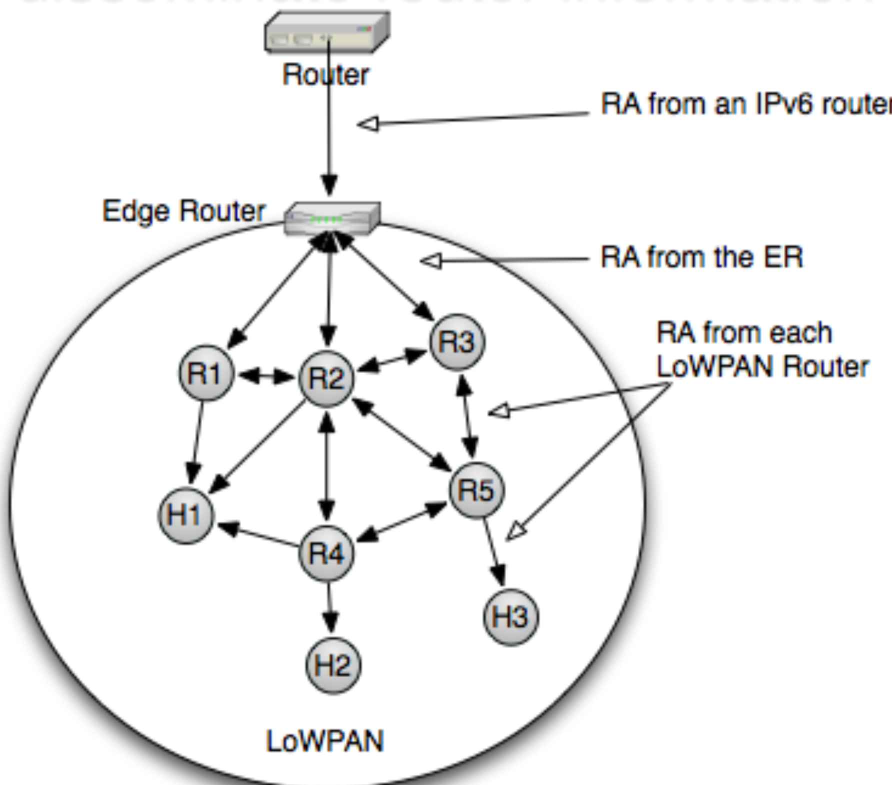
# 6LoWPAN Neighbor Discovery Call-Flow




Legend:
(mc) = Multicast
(uc) = Unicast

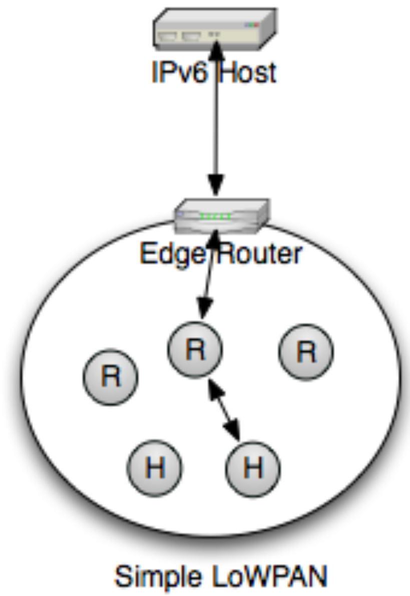
# Prefix Dissemination

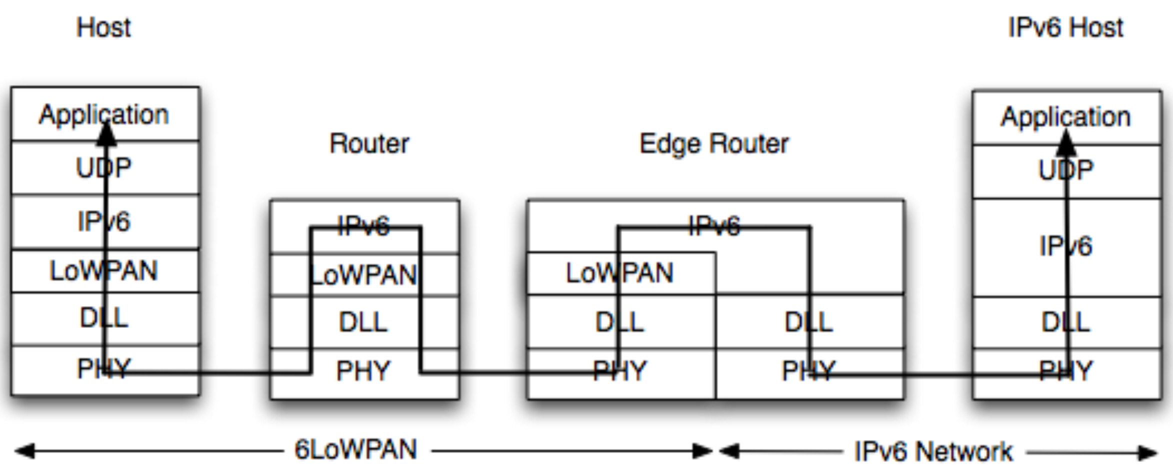
- In normal IPv6 networks RAs are sent to a link based on the information (prefix etc.) configured for that router interface
- In ND for 6LoWPAN RAs are also used to automatically disseminate router information across multiple hops



# 6LoWPAN Routing

- Here we consider IP routing (at Layer 3)
- Routing in a LoWPAN
  - Single-interface routing
  - Flat address space (exact-match)
  - Stub network (no transit routing)



# Routing Protocols for 6LoWPAN

- IP is agnostic to the routing protocol used
  - It forwards based on route table entries
- Thus 6LoWPAN is routing protocol agnostic
- Special consideration for routing over LoWPANs
  - Single interface routing, flat topology
  - Low-power and Lossy wireless technologies
  - Specific data flows for embedded applications
- MANET protocols useful in some ad-hoc cases
  - e.g. AODV, DYMO
- New IETF Working Group formed
  - Routing over low-power and lossy networks (ROLL)
  - Developed specifically for embedded applications

# IETF ROLL Working Group (WG)

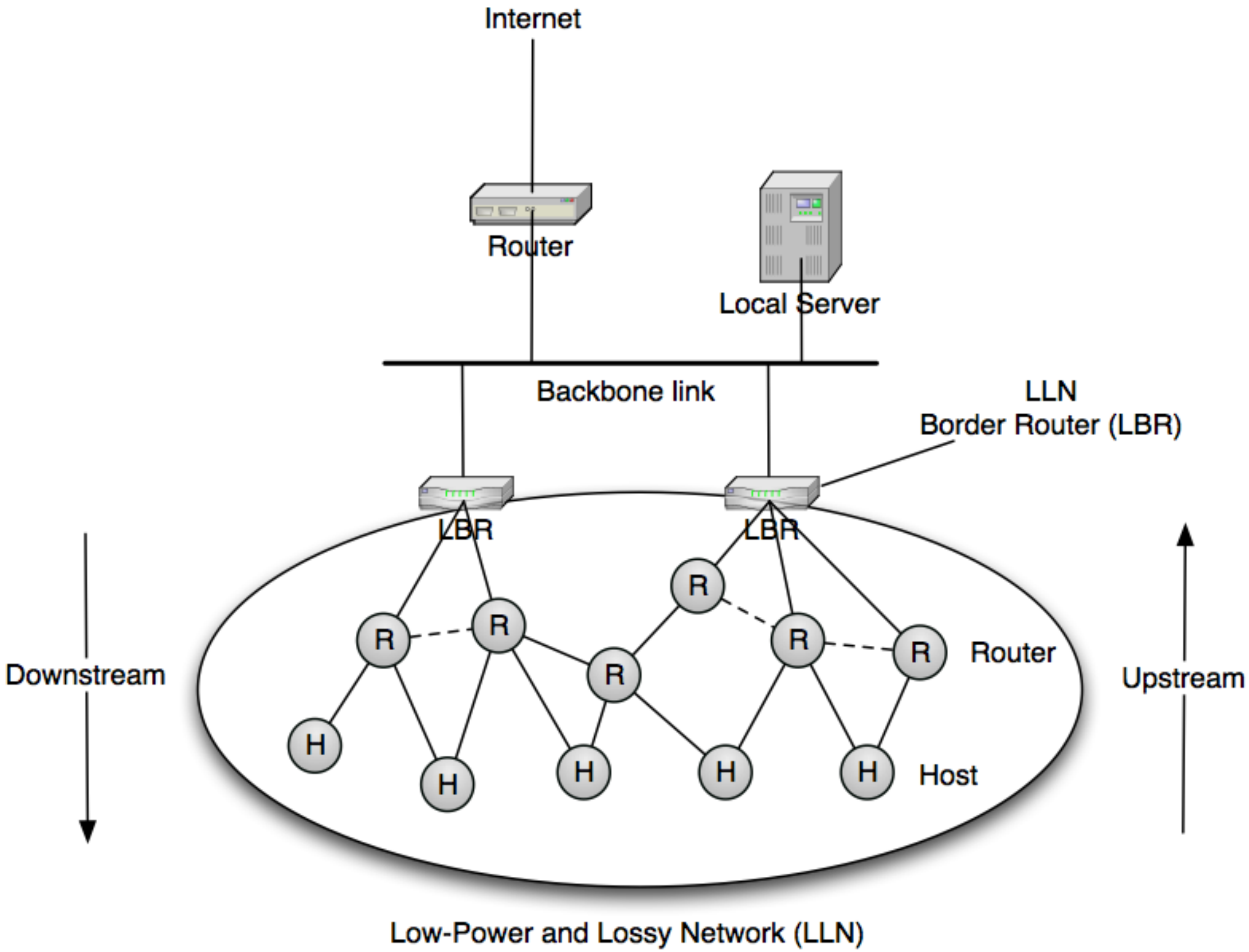
- Routing Over Low power and Lossy networks (ROLL)
  - Working group at the IETF
- Standardizing a routing algorithm for embedded apps
- Application specific requirements
  - Home automation
  - Commercial building automation
  - Industrial automation
  - Urban environments
- Analyzed all existing protocols
- Solution must work over IPv6 and 6LoWPAN
- The work results in RFC6550-6553: RPL (pronounced as "Ripple")

# RPL from IETF ROLL

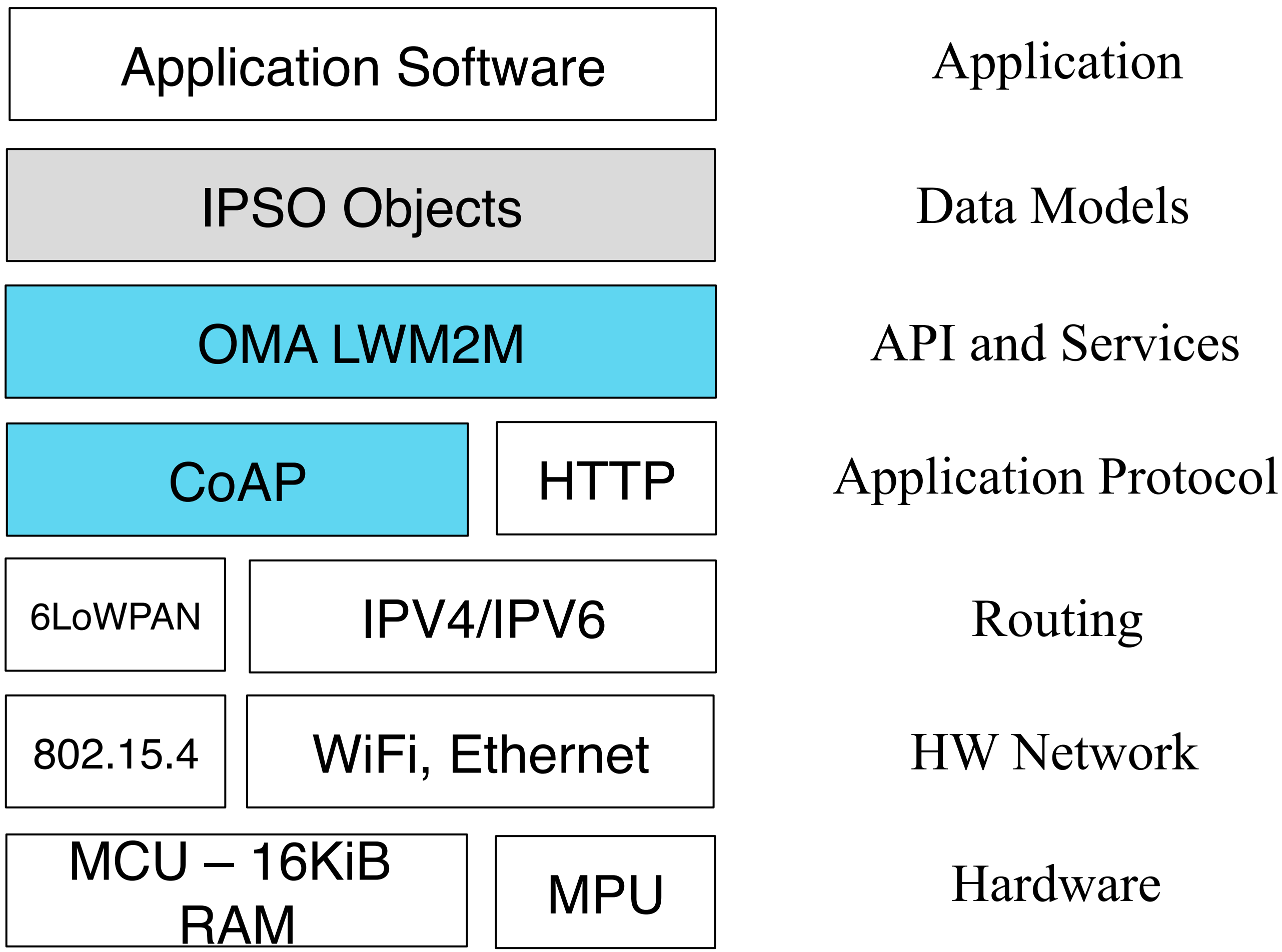
RPL (pronounced as “Ripple”) RFC6550-6553:

- Proactive Distance-Vector approach
- Approach is to build a Colored Destination-Oriented Directed Acyclic Graph (DODAG) comprised of 6LoWPAN routers to a Border Router (DODAG root)
  - Data flow implicitly to the Root
  - Use DAG instead of Trees for route redundancy/ resiliency
  - Multiple logical (colored) DAGs can co-exist in/ share the same physical network => Even more choices of paths for Traffic Engineering

# IETF ROLL RPL "Ripple"



# Recall: One of the IoT Standards "Stack"



# CoAP, OMA LWM2M, and IPSO Smart Objects

## Service and Application level Interoperability for IoT

# CoAP-> OMA LWM2M->IPSO

**IPSO** – smart objects built on top of LWM2M

- Application objects using LWM2M object model
- Composable – complex objects can be built up from simple ones
- Extensible – easy to add new resources and object types

**LWM2M** – built on top of CoAP

- Server profile for IoT middleware
- Simple, re-usable object model
- Device management objects
- API for onboarding, management

**CoAP**

- Device abstraction API and data compatibility layer
- Designed for constrained networks and devices
- HTTP proxy through standard web APIs
- Resource directory for scalable discovery and linking

# The Constrained Application Protocol (CoAP)

# Constrained Environment and Device Classes

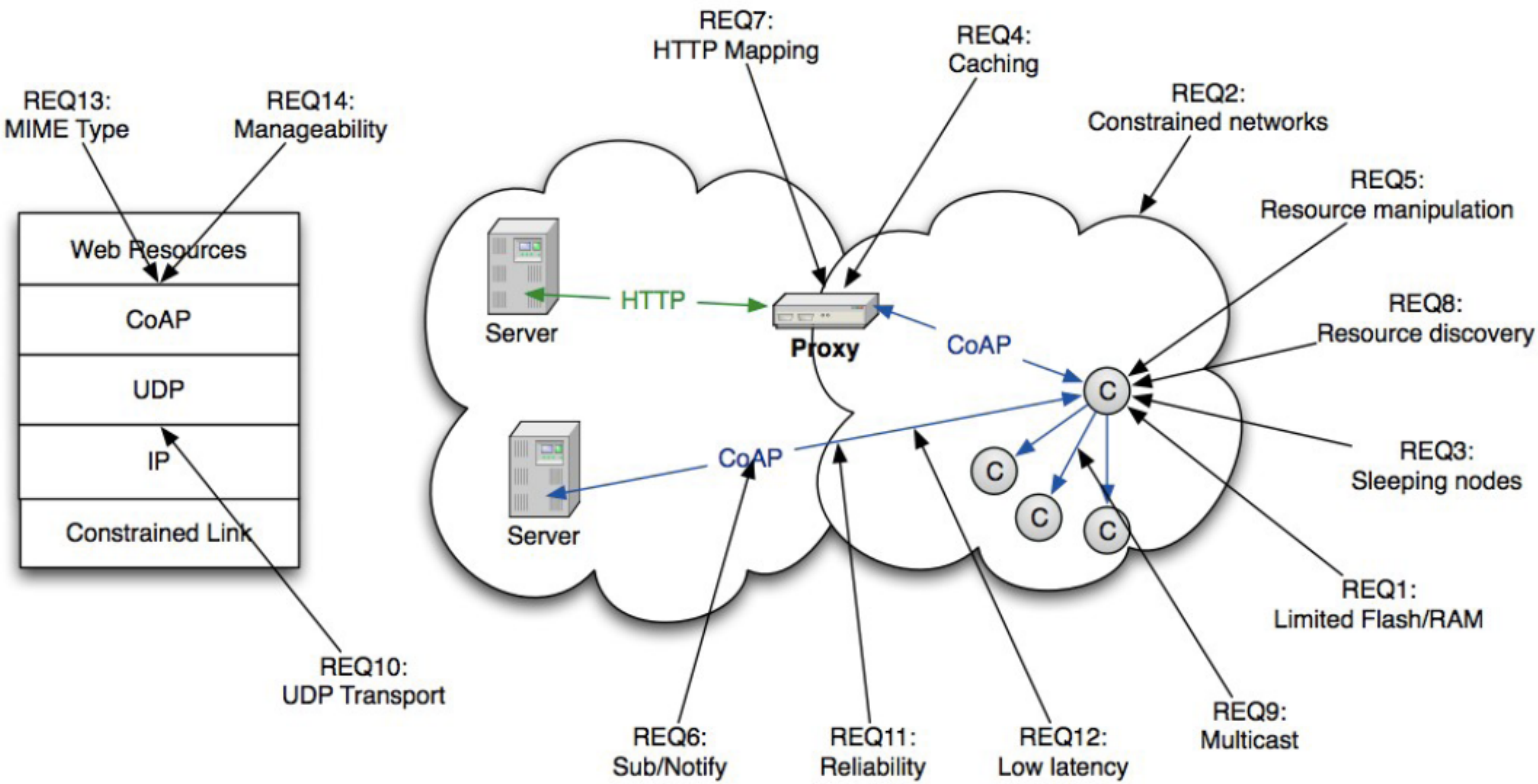
Constrained Environment:

- Low Cost
- Limited Processing
- Battery to last many years
- Varying Network Availability
- Often Low Data Rate

e.g. 10KB RAM, 100KB Flash, 40MHz MCU

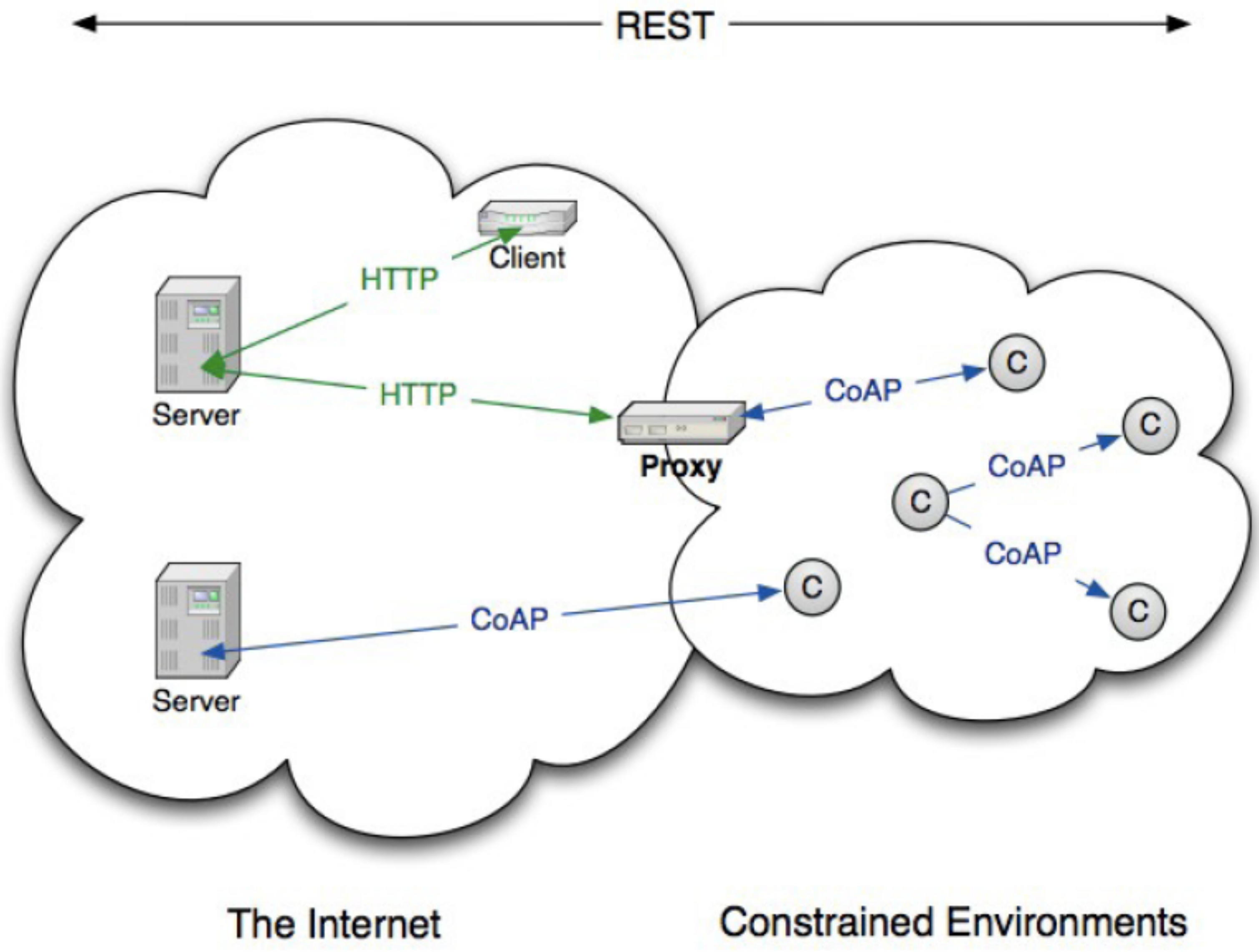
| Class | Rough Translation |
|---|---|
| 0 | Can’t run IP stack securely. |
| 1 | Integrated security but can’t employ full stack using HTTP over TLS. |
| 2 | Small but benefit from efficient protocols that free resources for application or reduce operational costs. |

# CoAP Design Requirements




See draft-shelby-core-coap-req

# CoAP Architecture



# The RESTful design for Web Applications
# REST: Representational State Transfer

It is the HTTP Client-Server programming style:

- W3C Technical Architecture Group – It's how the Web works
- Roy Fielding's UC Irvine dissertation, 2000
- Simple Methods:
  - Get, Put, Post, Delete (and a few others)

## Key Concepts

- Resources – Anything that can be named
  - Transparent connections – Applications just need the URI
- Interfaces – Simple basic Client-Server communications
  - Nothing App-specific: It's just Get, Put, Post, Delete, etc
- Representational – Current or Intended state of the Resource
  - Standard formats: HTML, JSON, EXI (Efficient XML Interchange), XML
- Hypermedia-driven Applications
  - REST applications can discover how to interact with Resources

# REST for IoT: CoAP

- CoAP – Constrained Application Protocol
- REST for IoT
  - **Implementation** ~~HTTP~~ **CoAP**
  - **Resources** Uniform identifiers (anything with a name)
  - **Interfaces** GET, PUT, POST, DELETE (not app-specific)
  - **Representations** ~~Page description~~ **Binary objects**

- CoAP key features
  - RESTful HTTP-like response/request
  - Easy to interface with RESTful web applications
  - UDP based (asynchronous messaging)
  - Compensates for transient / unreliable characteristics of IoT networks
  - Resource discovery and linking (RFC6690)
  - Simple web-compatible proxy and cache options

# CoAP vs. HTTP

| | CoAP | HTTP |
|---|---|---|
| **Transport** | UDP | TCP |
| **Message confirmation** | Optional – confirmable, non-confirmable | All messages acknowledged |
| **Message order** | Not ordered | Ordered |
| **Requests/responses** | Asynchronous | Uses established connection |
| **Encoding** | Can be binary | Plain text (usually) |

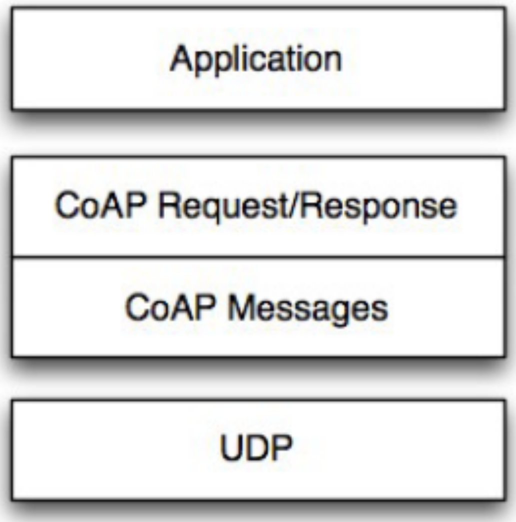
# CoAP Feature Highlights

- Embedded web transfer protocol ( coap:// )
- Support both Synchronous and Asynchronous Transaction models
- UDP binding with reliability and multicast support
- GET, POST, PUT, DELETE methods
- URI support
- Small, simple 4-byte binary header
- DTLS-based PSK, Public key and Certificate security
- Subset of MIME types and HTTP response codes
- Built-in Discovery
- Optional Observation and Block Transfer support

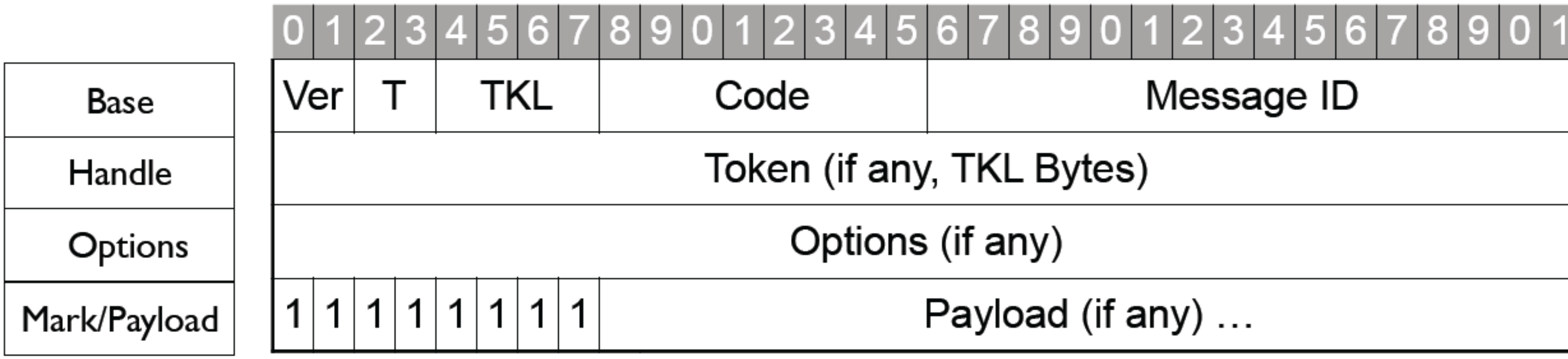
| Message | Code |
|---|---|
| GET | 0.01 |
| POST | 0.02 |
| PUT | 0.03 |
| DELETE | 0.04 |

# The Transaction model for CoAP

- Transport
  - UDP binding with DTLS security
  - CoAP over SMS or TCP possible
- Base Messaging
  - Simple message exchange between endpoints
  - Confirmable or Non-Confirmable Messages
  - Acknowledgement or Reset Message
- REST Semantics
  - REST Request/Response piggybacked on CoAP Messages
  - Method, Response Code and Options (URI, content-type, etc)

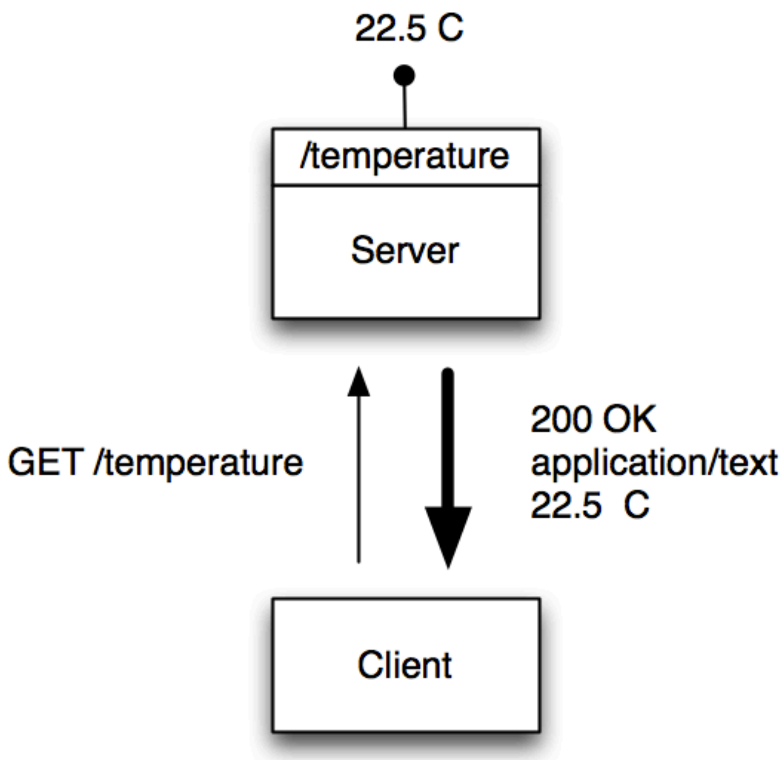


# CoAP Message




| | |
|---|---|
| **Ver** | Version |
| **T** | Transaction Type |
| **TKL** | Token Length |
| **Code** | Request Code |
| **Message ID** | Identifier |

# CoAP Protocol



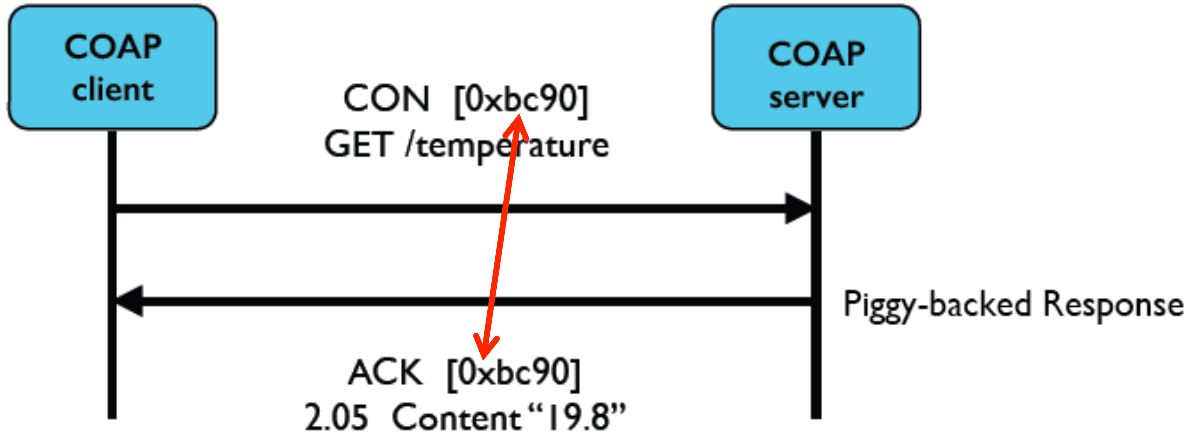

- Makes each IoT device a lightweight server that exposes a REST API
- A CoAP endpoint can be both client and server
- Roles can be reversed and the sensor, as a client, can also interact with a REST API at another endpoint or server node
- Peer to Peer interaction is based on a duplex client-server pattern

# CoAP URI

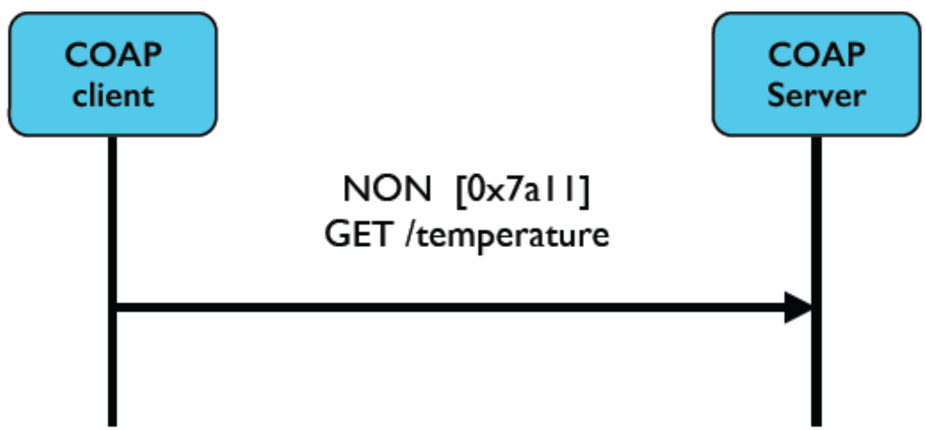
```
coap+sms://+441234567/garden/peas/water

coap://building.uk:5633/~room/occ.xml
```

# CoAP Request/ Response

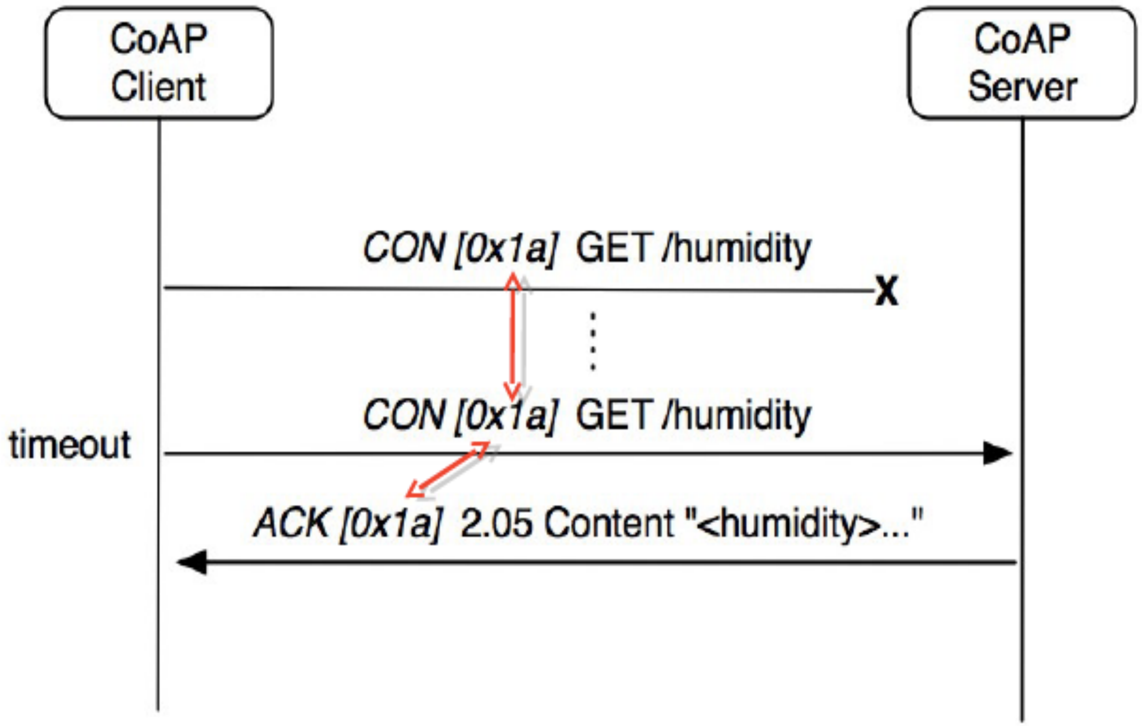
Confirmed Request



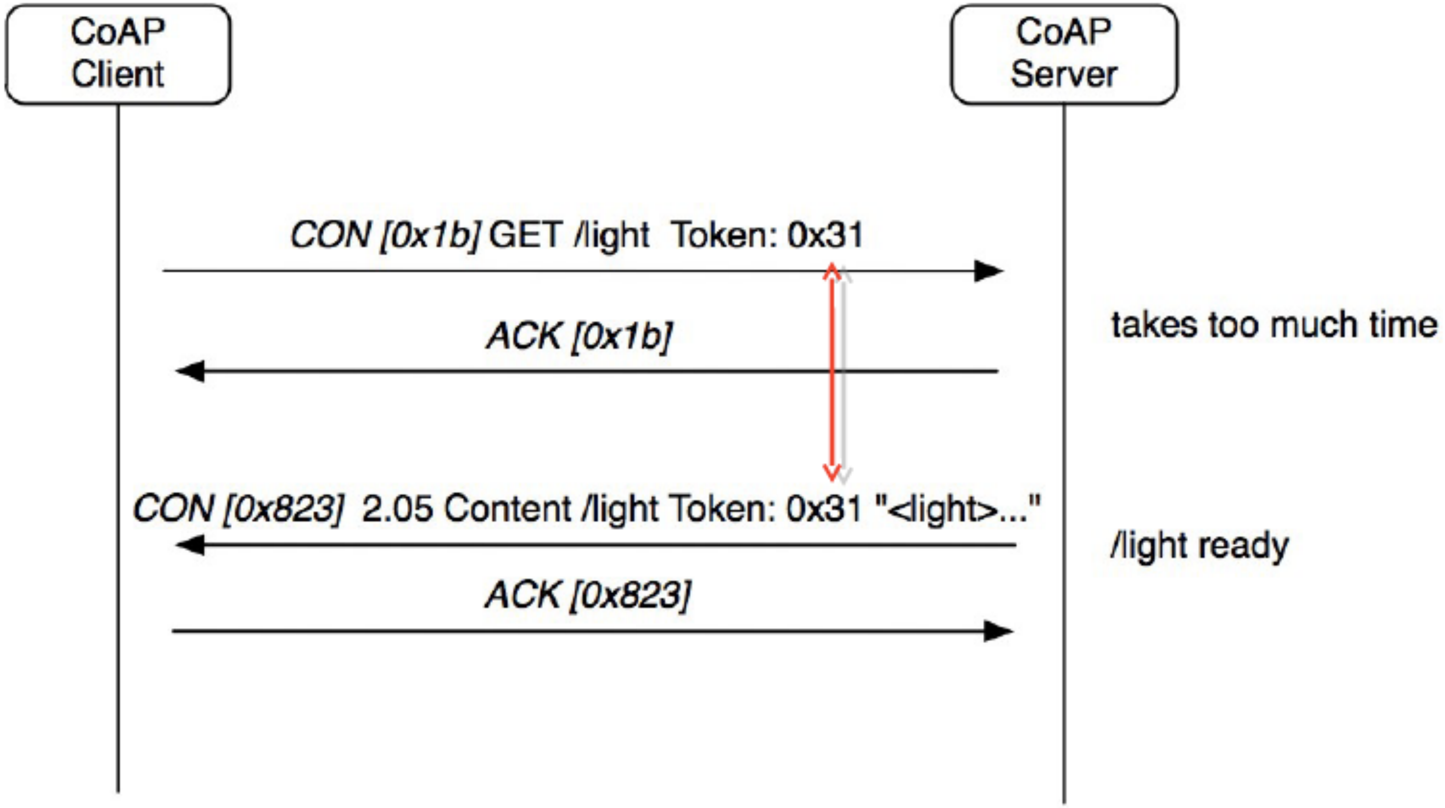

Non Confirmed Request



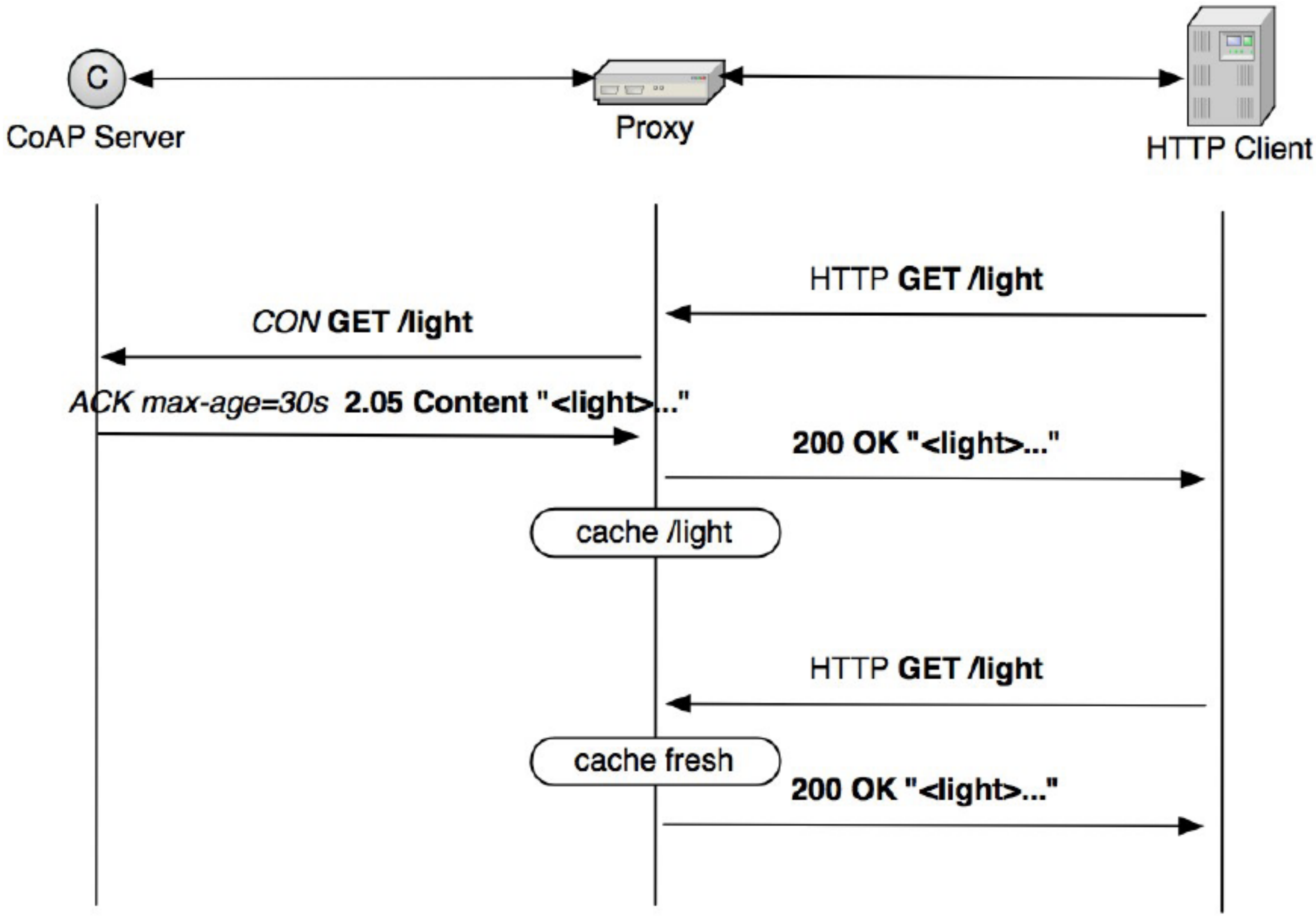
# CoAP Dealing with Packet Loss



# CoAP Separate Response



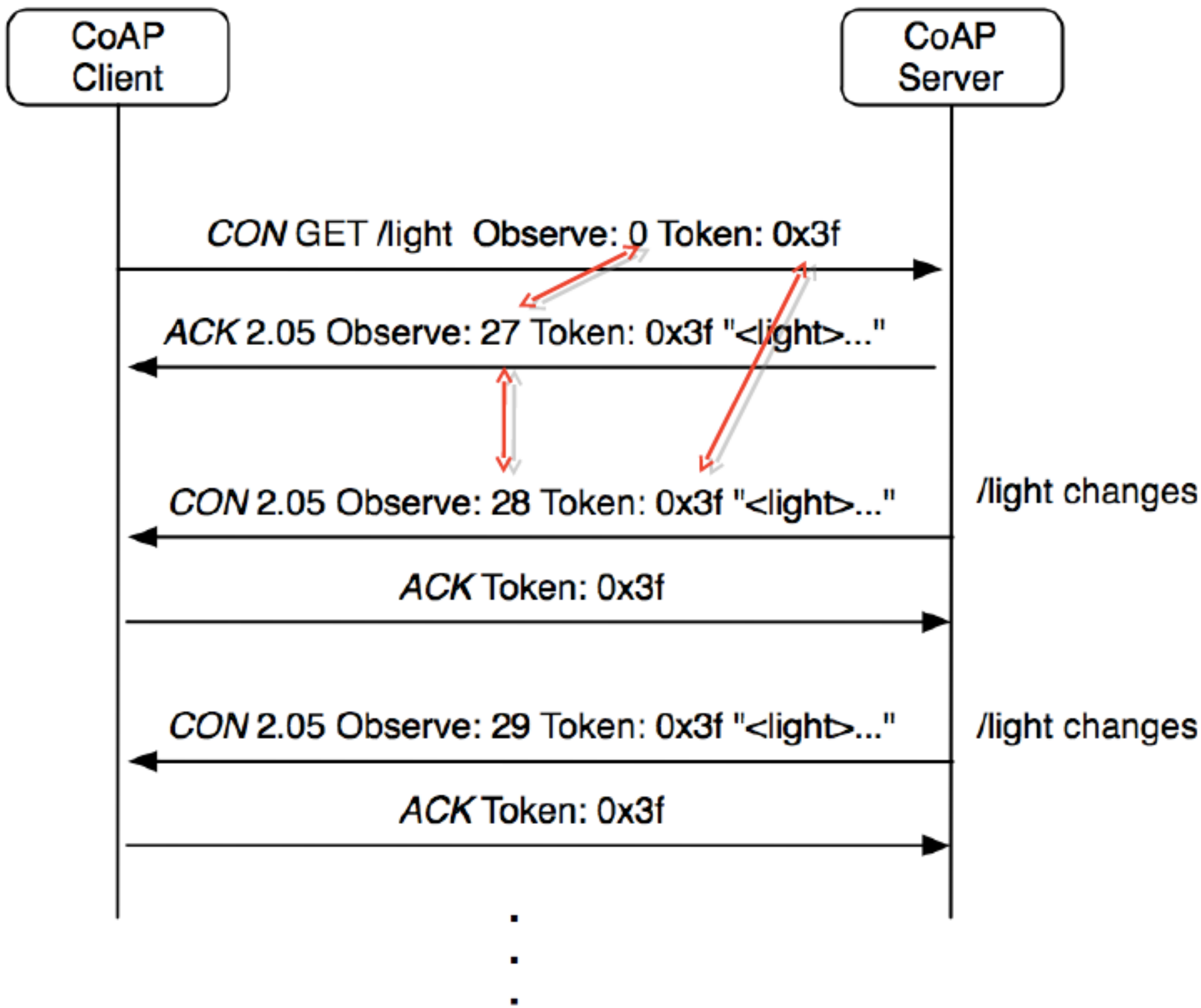
# CoAP Proxy and Caching



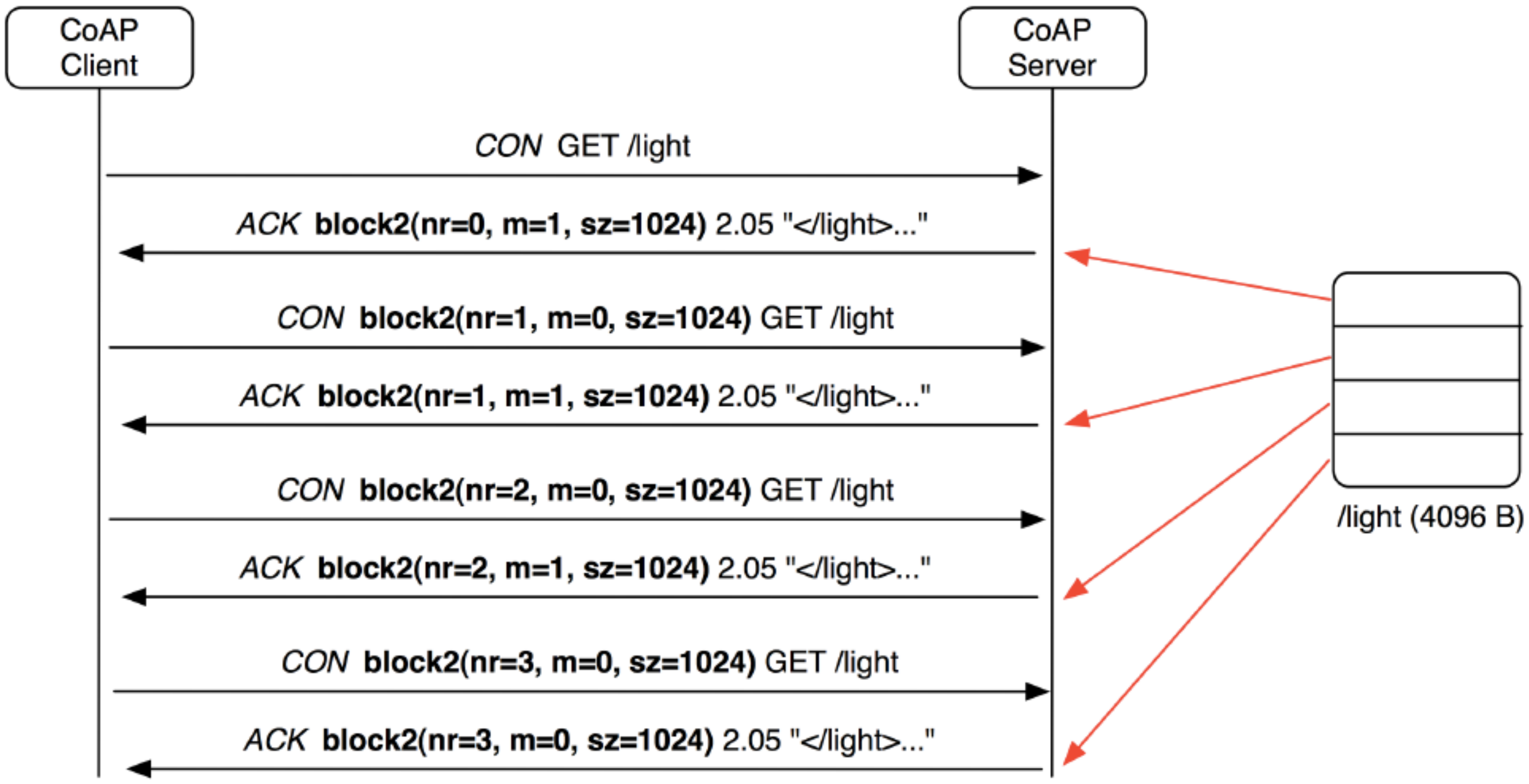
# CoAP Caching Support

CoAP includes a simple caching model
  - Cacheability determined by response code
  - An option number mask determines if it is a cache key
- Freshness model
  - Max-Age option indicates Cache Lifetime
- Validation model
  - Validty checked using the Etag Option
- A Proxy often supports caching
  - Usually on behalf of a constrained node, or
  - A Sleeping node, or
  - To reduce network load

# Support of “Observation” mode

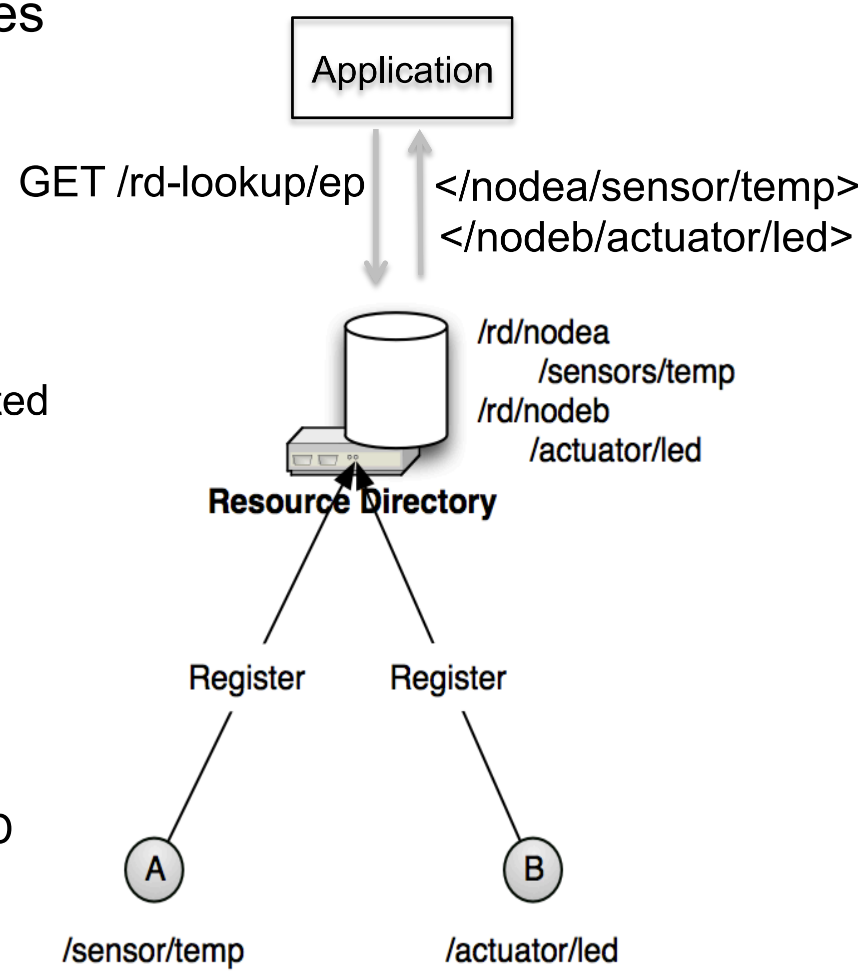


# Support of Block Transfer



# CoAP Resource Discovery

- RFC 6690 CoRE Link Format defines
  - The link format media type
  - Peer-to-peer discovery
- A Directory approach is also useful
  - Supports sleeping nodes
  - No multicast traffic, longer battery life
  - Remote lookup, hierarchical and federated distribution
- CoRE Link Format is used in Resource Directories
  - Nodes register their resource links to an RD
  - Nodes refresh the RD periodically
  - Nodes may unregister (remove) their RD entry



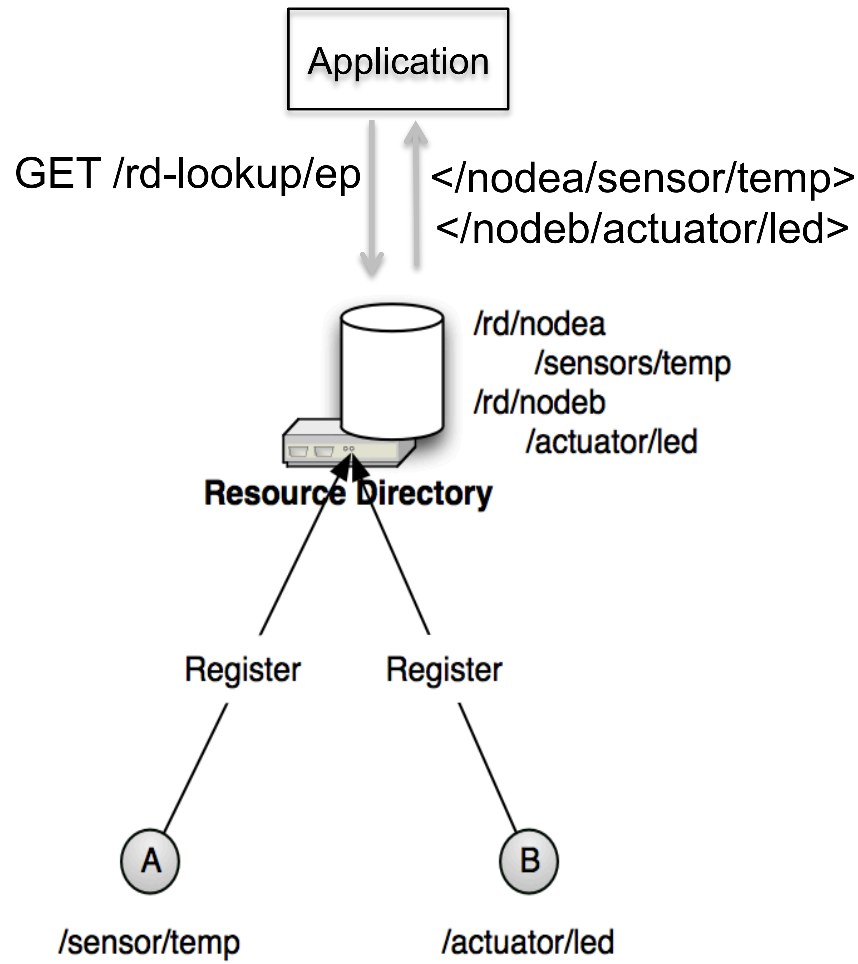
# CoAP Resource Discovery

```
GET /.well-known/core
```

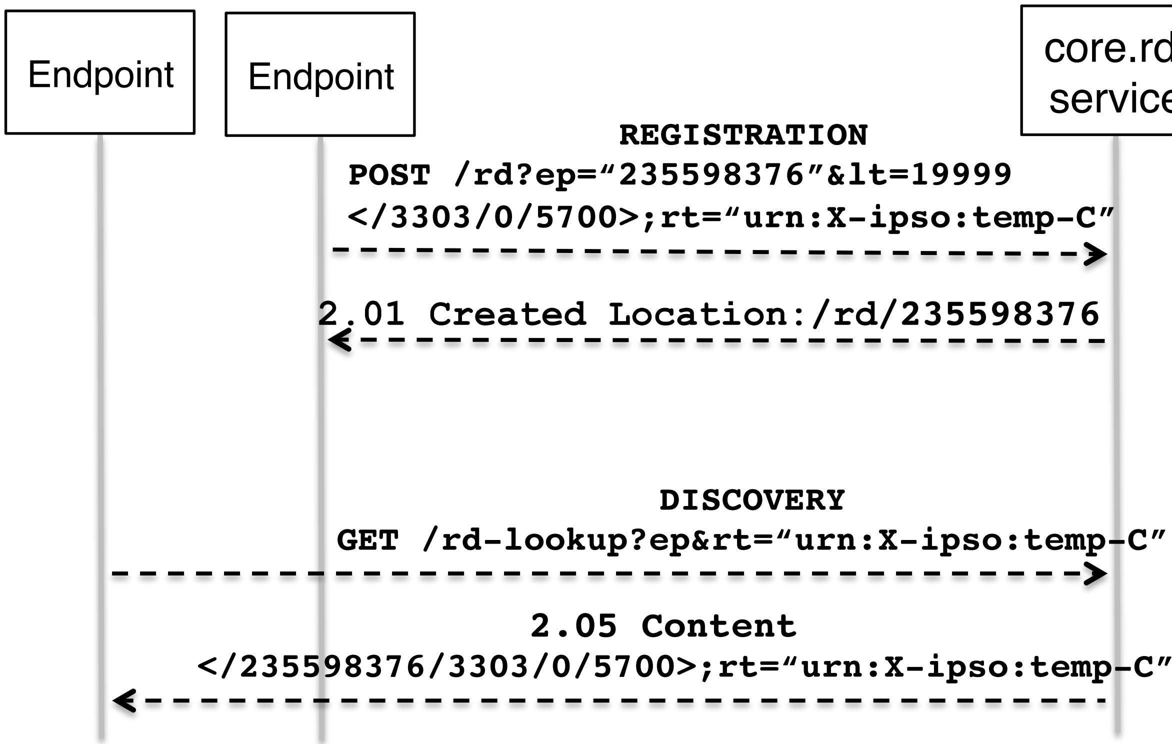
```
rt
ct
if
sz
```

```
GET /.well-known/core?rt=radiation
```

```
2.05 Content
</sensors/radiation>;rt="geiger";if="sensor"
```



# Resource Discovery Example Flow



# Realization/ Implementation of CoAP in Practice

- Many Open Source Implementation available:
  - Java CoAP Library Californium
  - C CoAP Library Erbium
  - libCoAP C Library
  - jCoAP Java Library
  - OpenCoAP C Library
  - UCB TinyOS and Contiki include CoAP support
- Some commercial products/ systems:
  - Sensinode NanoService (acquired by ARM in 2013)
  - RTX 4100 WiFi module
- Firefox has a CoAP plugin called Copper
- Wireshark has CoAP dissector support

AMQP

HTTP

MQTT

# Alternatives to CoAP ?

XMPP

DDS

STOMP

# Standardization Activities

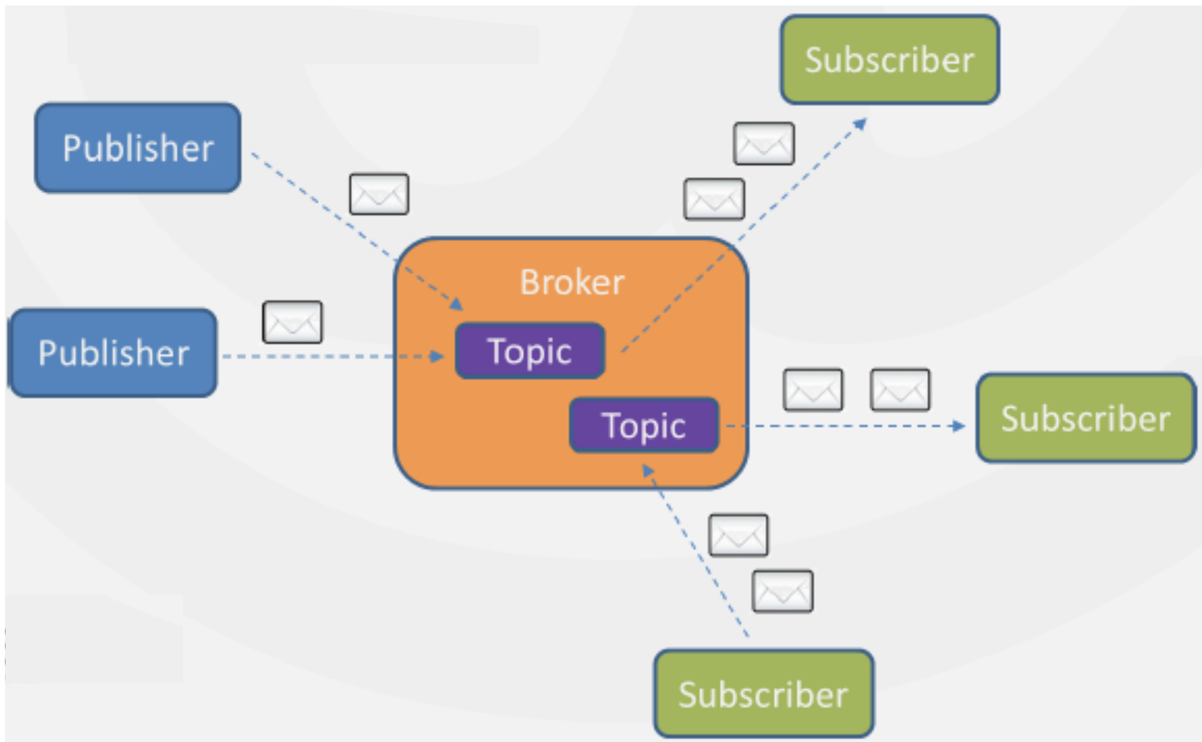
- HTTP
  - IETF standard (RFC 2616 is HTTP/1.1)
- CoAP
  - IETF standard (RFC 7252), June 2014
- Message Queuing Telemetry Transport (MQTT)
  - MQTT v3.1.1, OASIS standard, Nov. 2014
- Advanced Message Queuing Protocol (AMQP)
  - AMQP v1.0, OASIS and ISO 19464 standard, Oct 2012

# MQTT Overview

- Background
  - Previously Message Queuing Telemetry Transport
    - Created by IBM & Eurotech
  - Now: MQ Telemetry Transport … no Queue
    - Donated to Eclipse Foundation and OASIS standard
- Key Features
  - Lightweight – smallest packet size = 2 bytes (header)
  - TCP-based – socket connection oriented
    - Use SSL/TLS to encrypt payload
  - Reliable
    - Three QoS levels: "At Most Once", "At Least Once", "Exactly Once"
    - Avoid packet loss on Client disconnection
  - Publish/ Subscribe model – Decouple Producers and Consumers
  - Payload Agnostic:
    - No data types
    - No metadata
    - Any data format (Text, Binary, JSON, XML, BSON, ProtoBuf, etc)

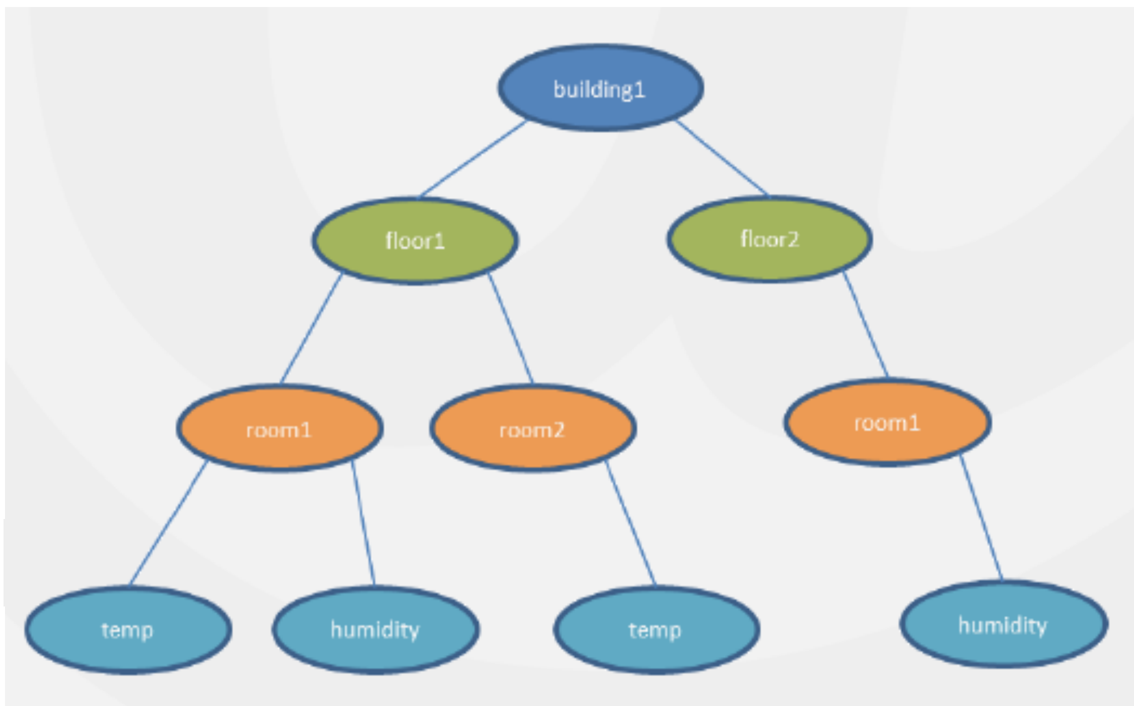
# MQTT Publish/Subscribe model

- Broker and Connected Clients
  - Broker receives subscriptions from Clients on Topics
  - Broker receives messages and forward them
  - Clients subscrbe/ publish on Topics

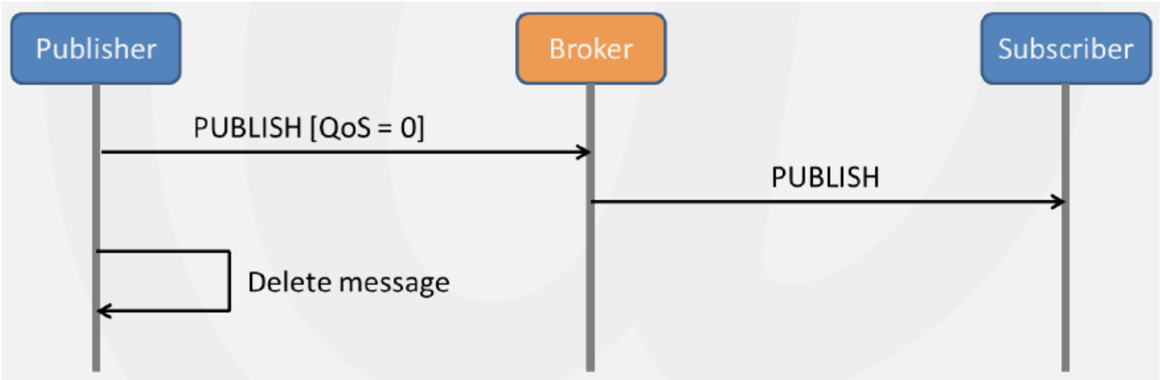


# MQTT Hierarchical Topics

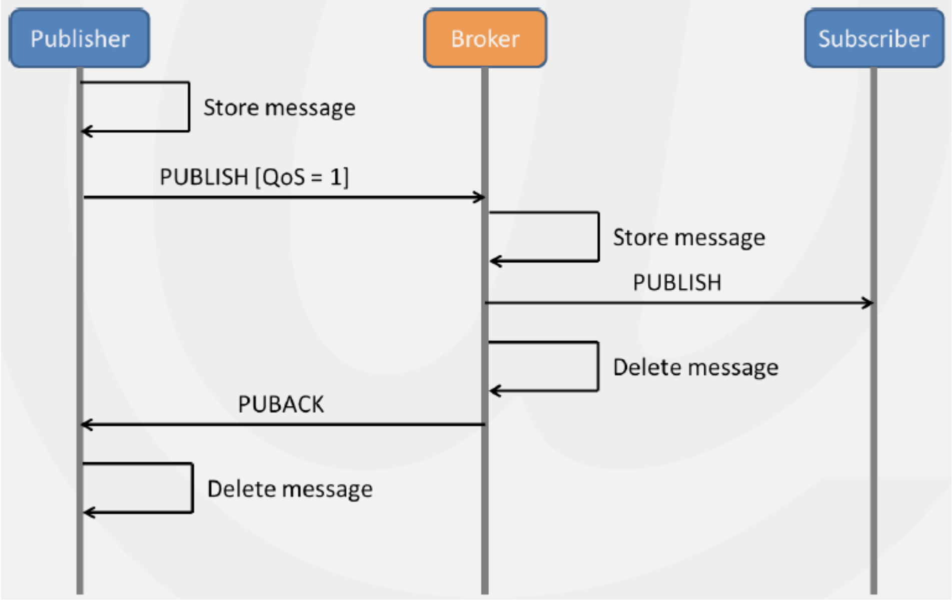
- Topics for Publish and Subscribe
  - Hierarchical
  - Supporting Wildcards ( # and + )
    - e.g., building1/+/room1, building1/floor1/room1/#



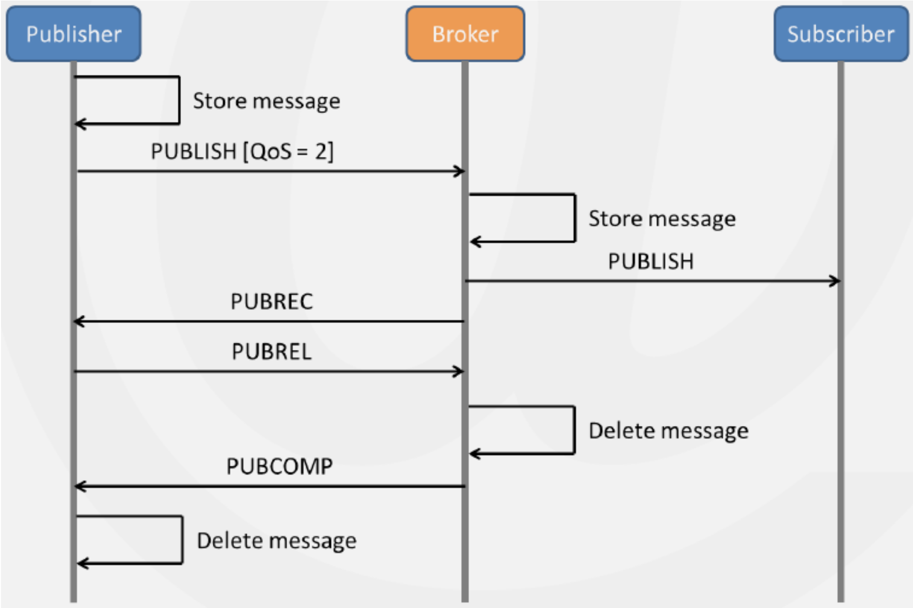
# MQTT Quality of Service (QoS)




QoS 0: “At Most Once” (Fire and Forget)




QoS 1: “At Least Once”



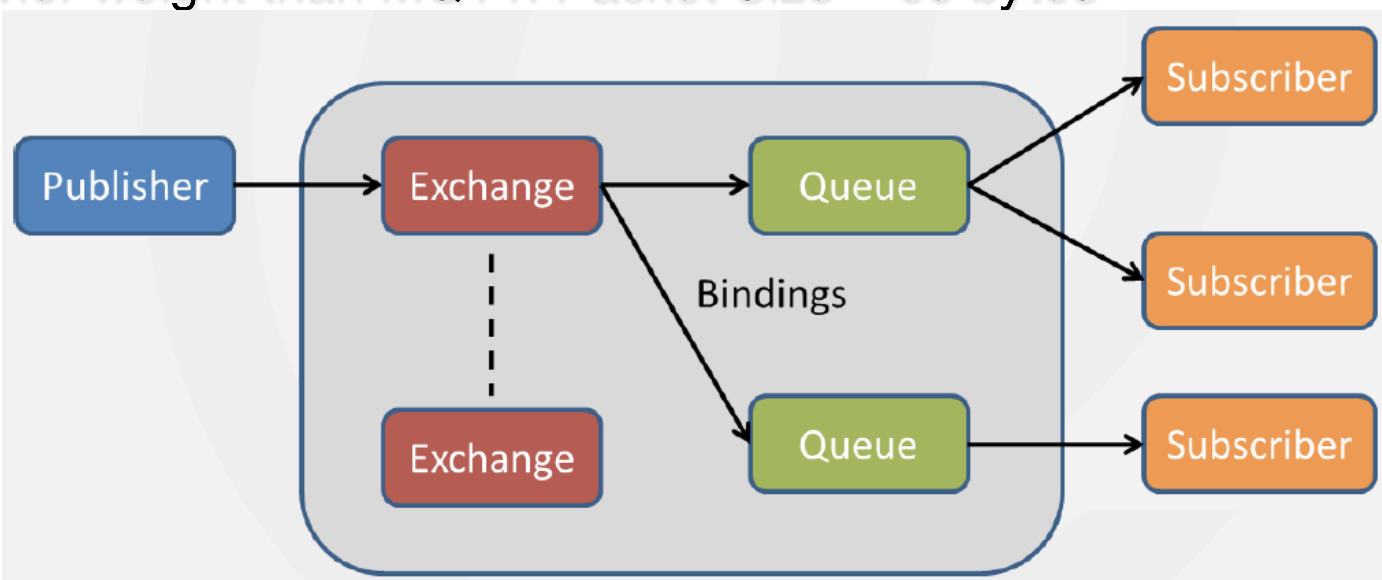

QoS 2: “Exactly Once”

# MQTT Additional Key Features

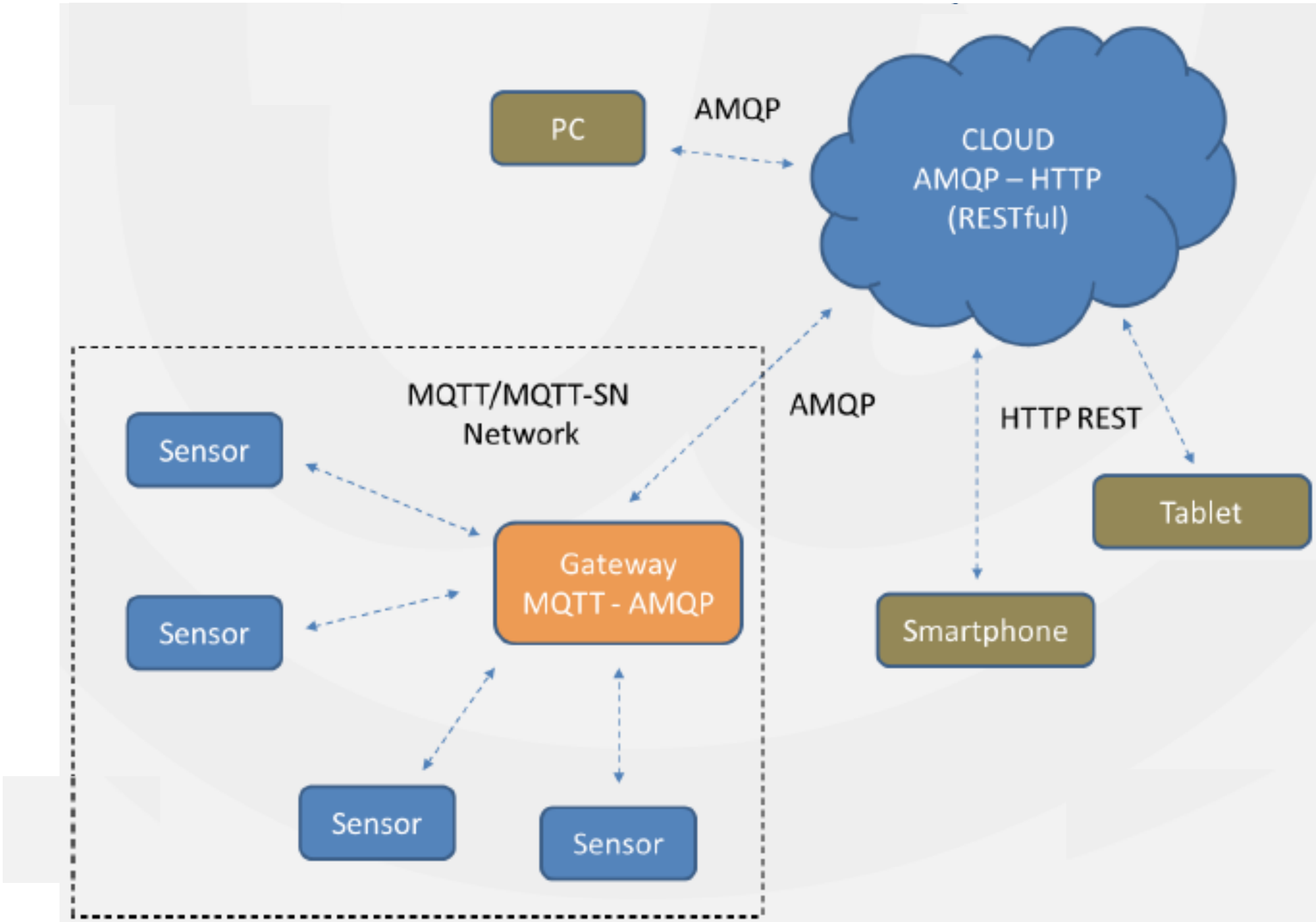
- Keep-Alive message
  - PINGREQ/PINGRESP message
  - Broker can detect Client Disconnection
- Will & Testament
  - Make a "Will" message with QoS and Topic on connection
  - Broker sends on unexpected Client disconnection
- Retain message
  - Published message is kept on the Broker ;
  - A new subscriber on Topic receives the "last known" good message
- Clean session
  - On Client Disconnection, all subscriptions are kept
  - No need to re-subscribe on client re-connection
  - Receive all messages published during Offline

# Advanced Message Queuing Protocol (AMQP)

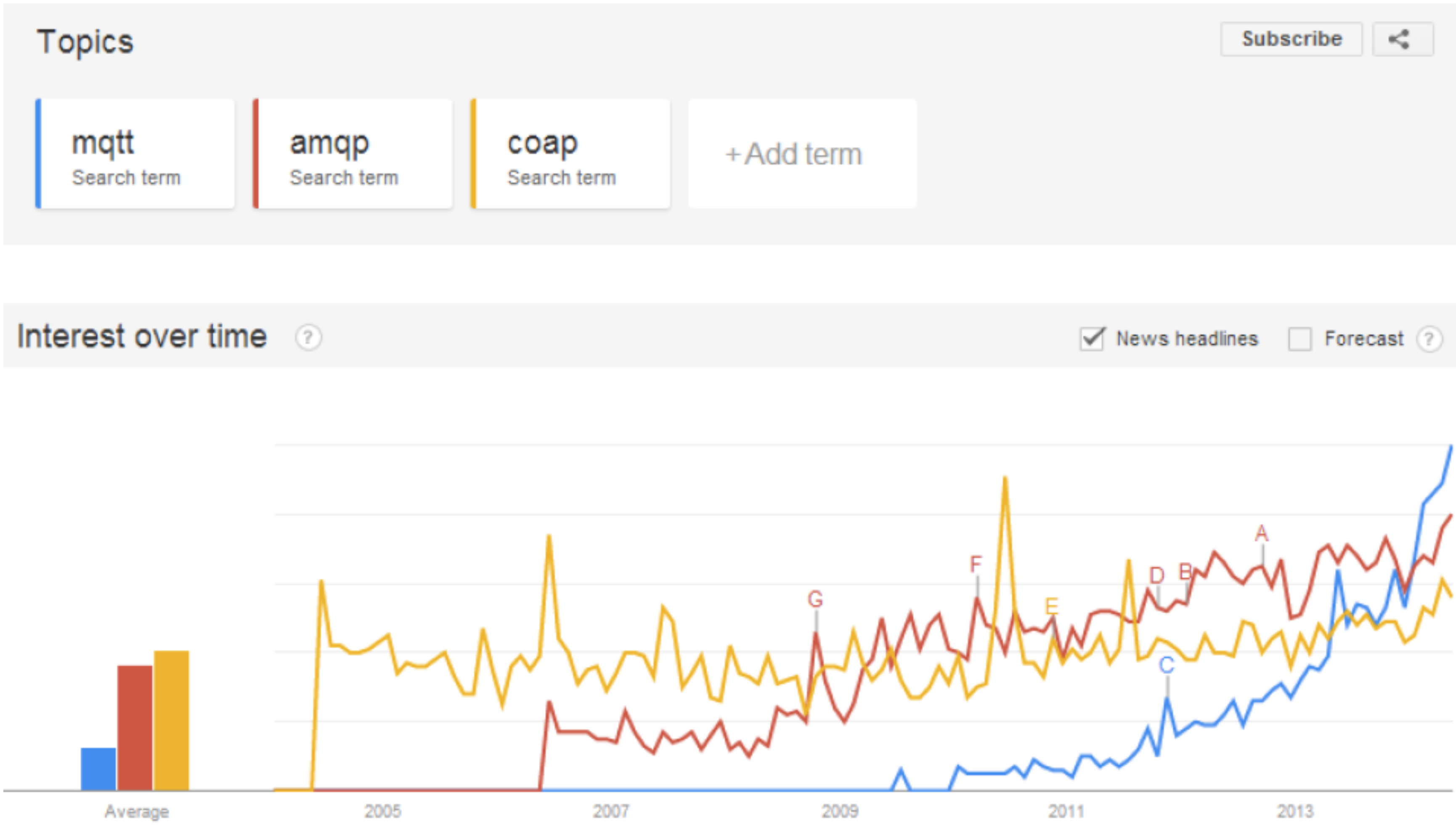
- Also follows the Publish-Subscribe model
- An Exchange module to receive messages and apply routing
- Support Binding to define rules to bind exchange to Queue
- Queue for storing messages
- Binary connection-oriented
- Support credit-based Flow Control
- SSL/TLS and SASL for security
- Heavier weight than MQTT: Packet Size ~ 60 bytes



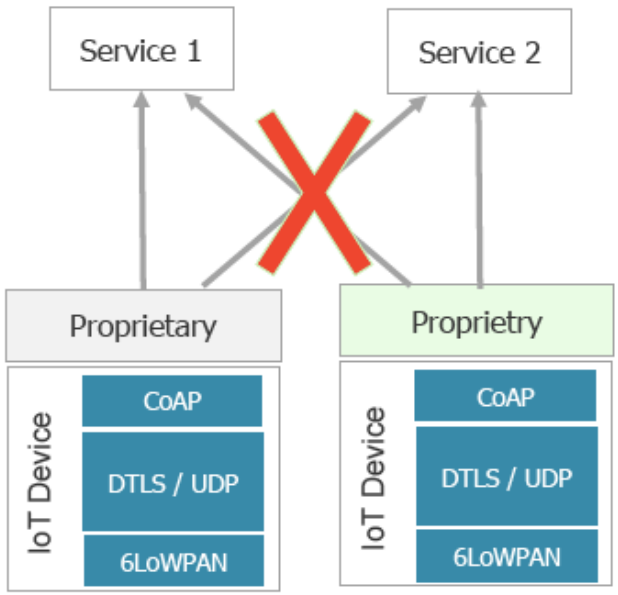
# Sample Deployment Scenario for MQTT/AMQP



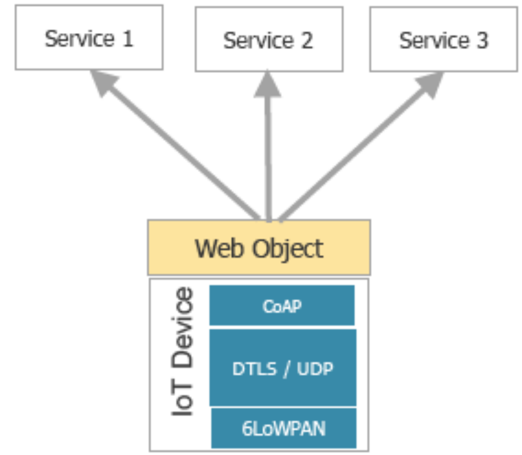
# IoT protocols Trend



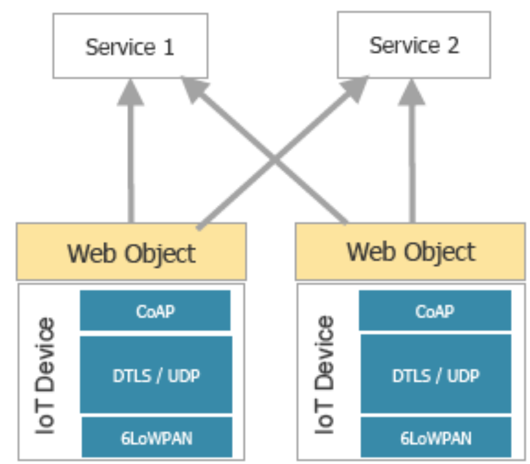
# The Need of Standard Web Object Definition for Service/Device Interoperability




*Non-interoperable devices & Services*




*Interoperable Services*

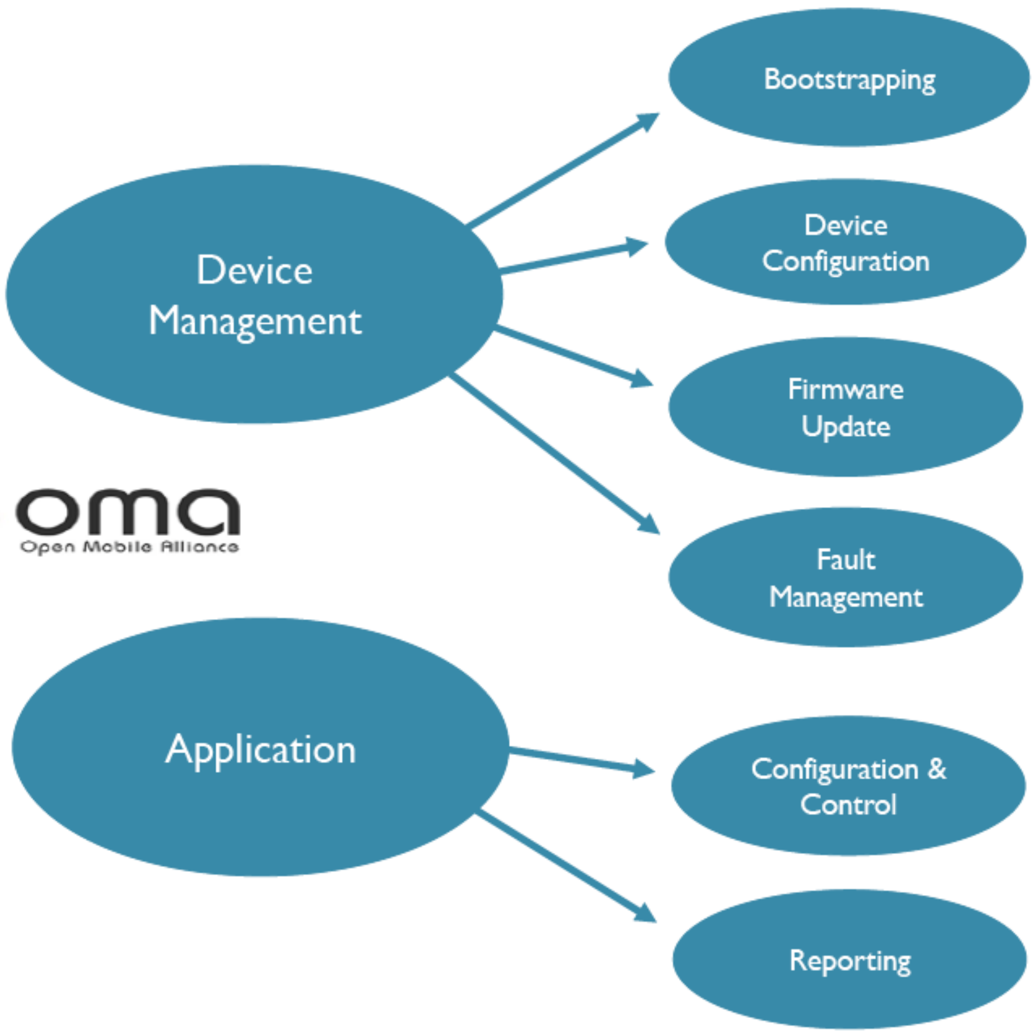



*Interoperable Devices & Services*

In addition to data communication we need standard web objects for Interoperability

# Device Management (DM) for IoT
# via
## The Open Mobile Alliance Light-Weight Machine-to-Machine Protocol
## (OMA LWM2M)

# Light Weight Device Management




- Key management
- Service provisioning

- Changes to settings
- Changes to parameters of the device

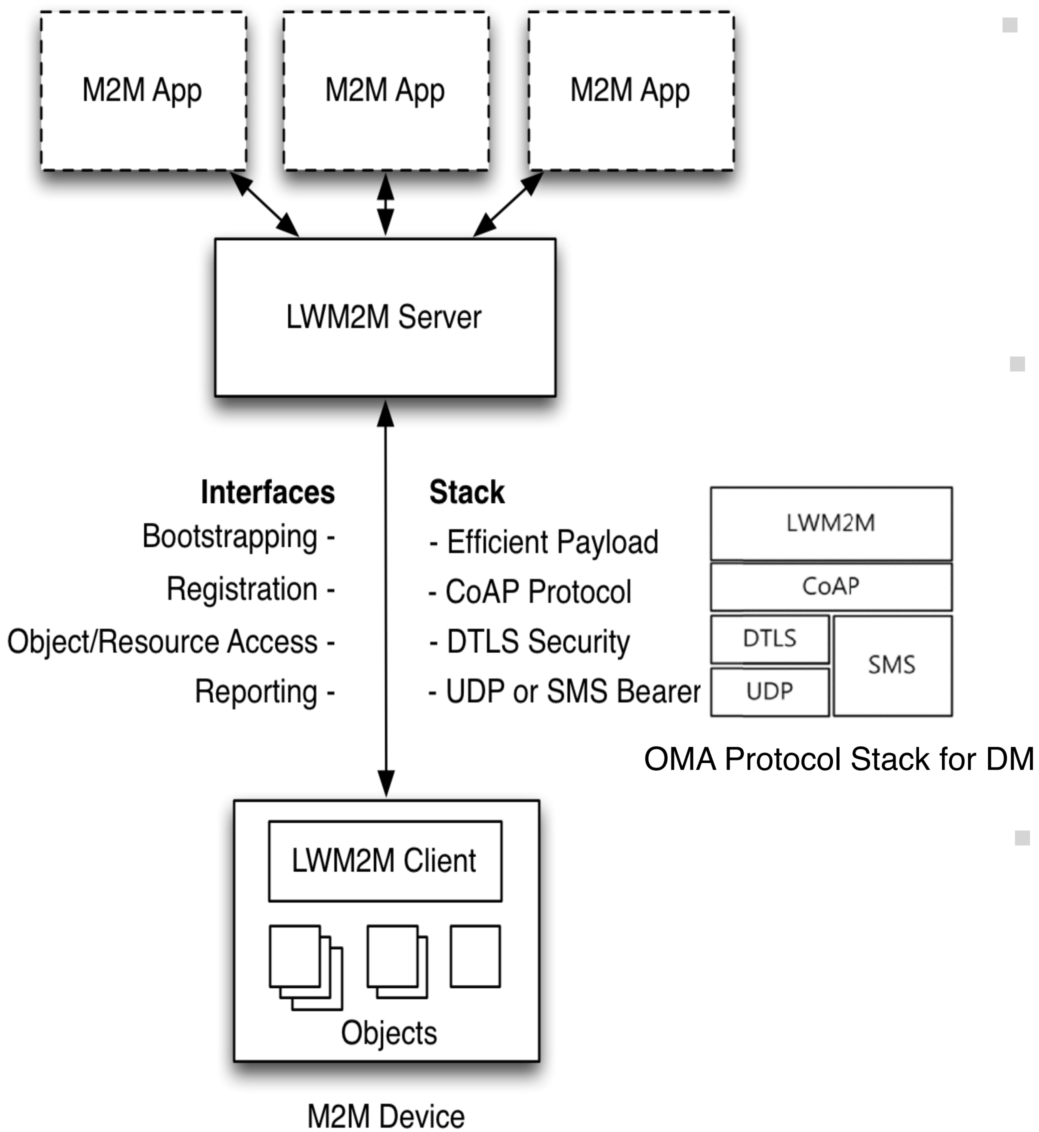
- Update application and system software
- Bug fixes

- Report Errors from devices
- Query about status of devices

- Configure settings of the application
- Send control commands

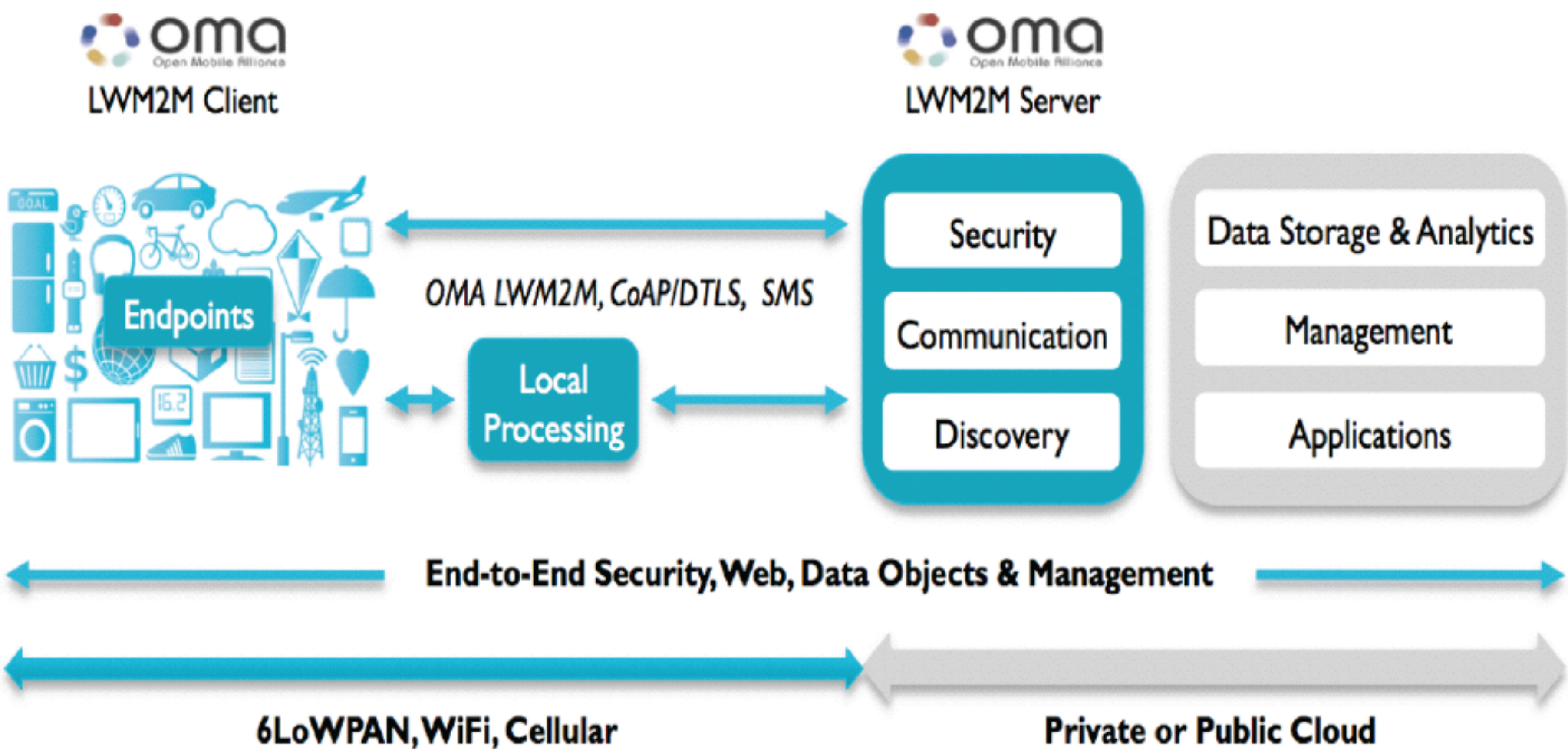
- Notify changes in sensor values
- Notify alarms and events

# OMA LWM2M Reference Architecture




- Web (M2M) Applications
  - Application abstraction through HTTP/RESTful API
  - Resource Discovery and Linking
- LWM2M Server
  - CoAP Protocol
  - Supports HTTP Caching Proxy
  - Resource Directory
  - Gateway and Cloud deployable
- LWM2M Clients are Devices
  - Device abstraction through CoAP
  - LWM2M Clients are CoAP Servers
  - Any IP network connection

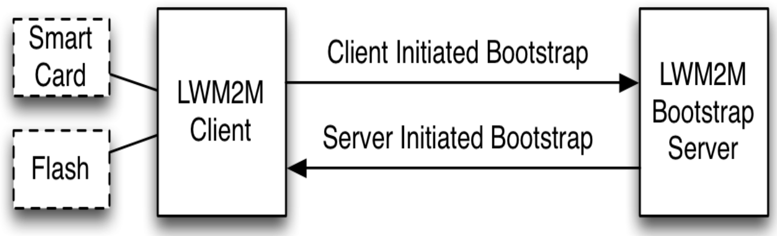
# LWM2M DM Deployment Scenario



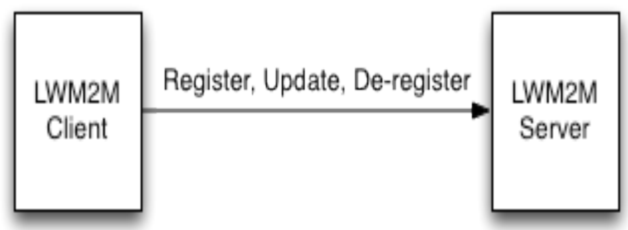

from Whitepaper Feb 2014, Vodafone, ARM, Ericsson

# LWM2M Interfaces

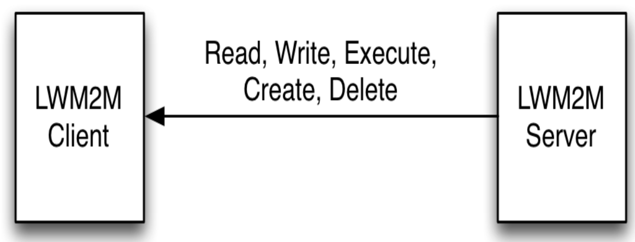
- Bootstrap Interface
  - Configure Servers & Keying
  - Pre-Configured, Smart Card, or Server Initiated Bootstrap
  - CoAP REST API



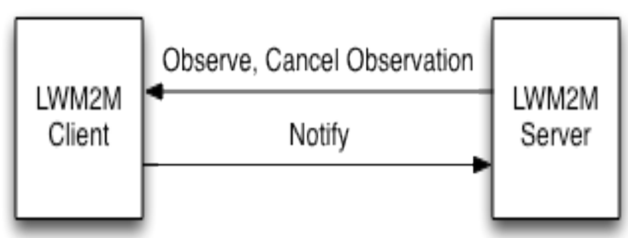

- Registration Interface
  - RFC6690 and Resource Directory



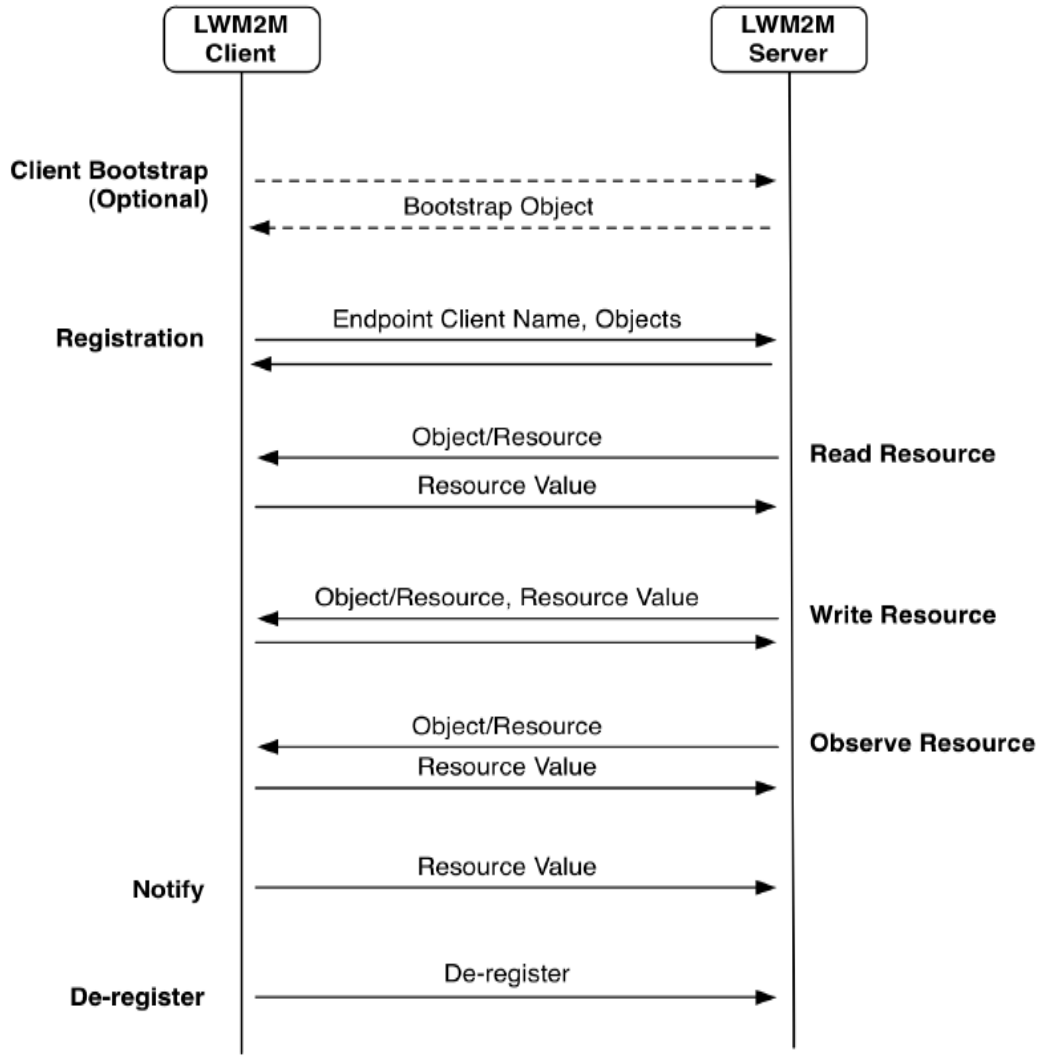

- Management Interface Using Objects
  - Management Objects and Resources
  - CoAP REST API




- Reporting Interface
  - Object Instances and Resources Report
  - Asynchronous notification using CoAP Observe



# LWM2M Interface Call-Flow



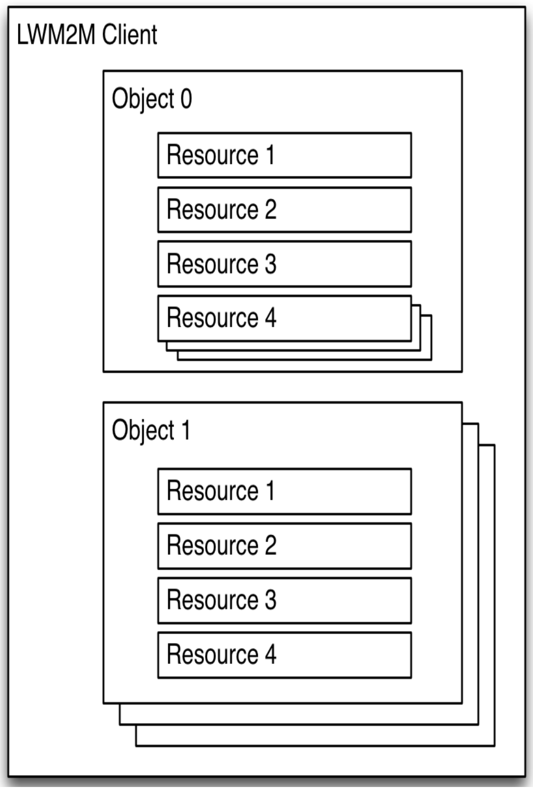
# LWM2M Object Model

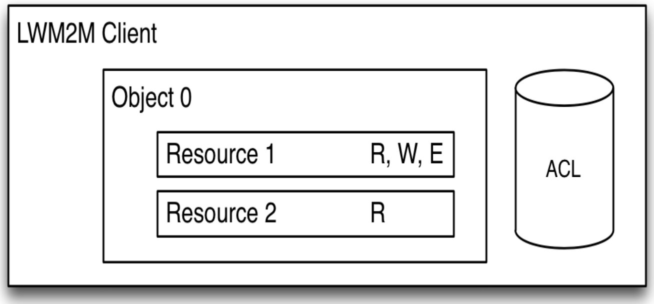
- A Client has one or more Object Instances
- An Object is a collection of Resources
- A Resource is an atomic piece of information that can be
  - Read, Written or Executed
- Objects can have multiple instances
- Objects and Resources are identified by a 16-bit Integer, Instances by an 8-bit Integer
- Objects/Resources are accessed with simple URIs:

  /{Object ID}/{Object Instance}/{Resource ID}

  Example:

  **/3/0/1** - Object Type=3 (Device), Instance=0, Resource Type = 1 (Device Mfg.)



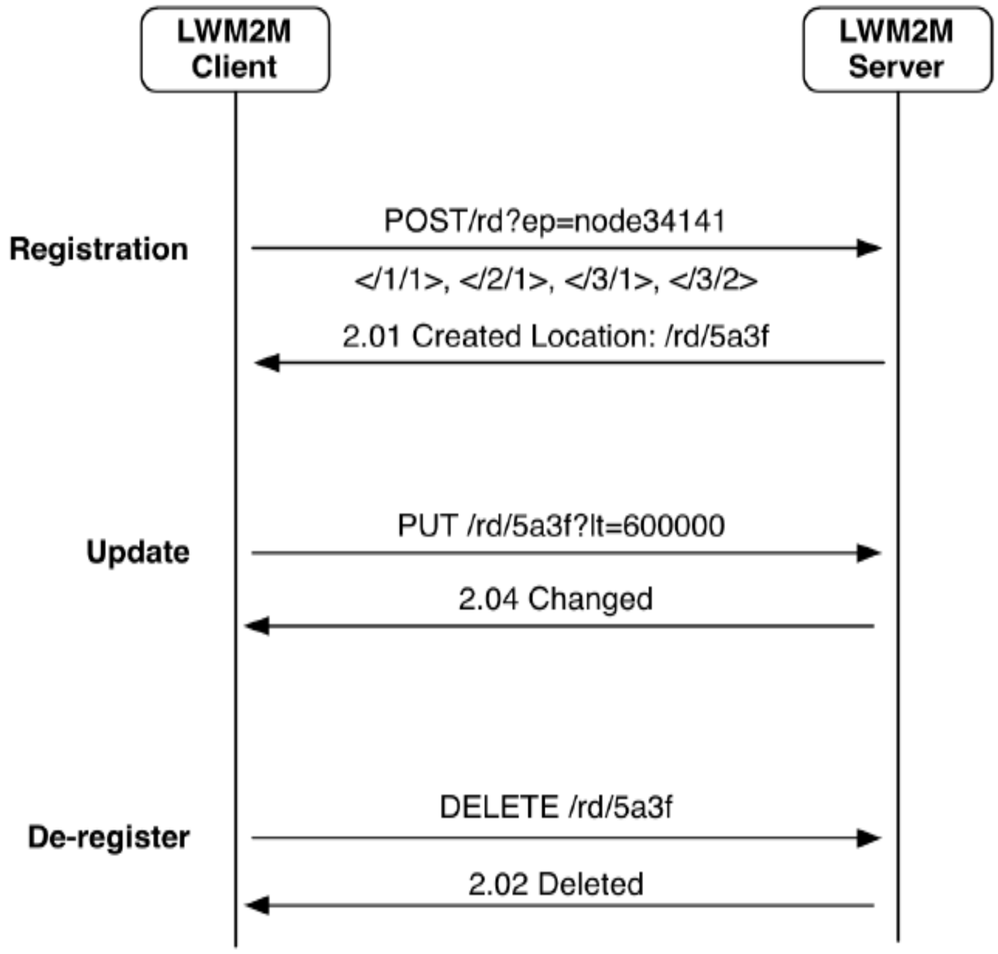
# LWM2M Management Objects

| Object | Object ID |
|---|---|
| LWM2M Security | 0 |
| LWM2M Server | 1 |
| Access Control | 2 |
| Device | 3 |
| Connectivity Monitoring | 4 |
| Firmware | 5 |
| Location | 6 |
| Connectivity Statistics | 7 |

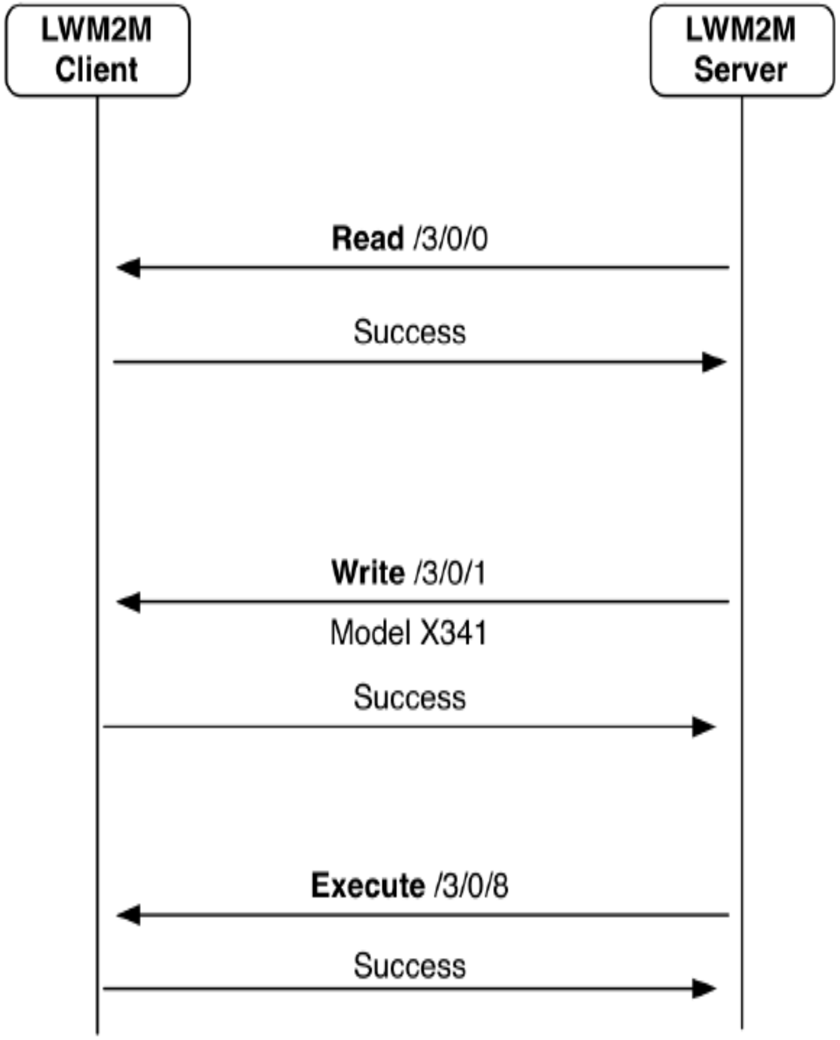
# LWM2M Position Object Example, OMA Template

| Resource Name | ID | Access Type | Multiple Instances? | Type | Range | Units | Descriptions |
|---|---|---|---|---|---|---|---|
| Latitude | 0 | R | No | Decimal | | Deg | The decimal notation of latitude, e.g. -43.5723 [World Geodetic System 1984] |
| Longitude | 1 | R | No | Decimal | | Deg | The decimal notation of longitude, e.g. 153.21760 [World Geodetic System 1984] |
| Altitude | 2 | R | No | Decimal | | m | The decimal notation of altitude in meters above sea level. |
| Uncertainty | 3 | R | No | Decimal | | m | The accuracy of the position in meters. |
| Velocity | 4 | R | No | Refers to 3GPP GAD specs | | Refers to 3GPP GAD specs | The velocity of the device as defined in 3GPP 23.032 GAD specification. This set of values may not be available if the device is static. |
| Timestamp | 5 | R | No | Time | | | The timestamp of when the location measurement was performed. |

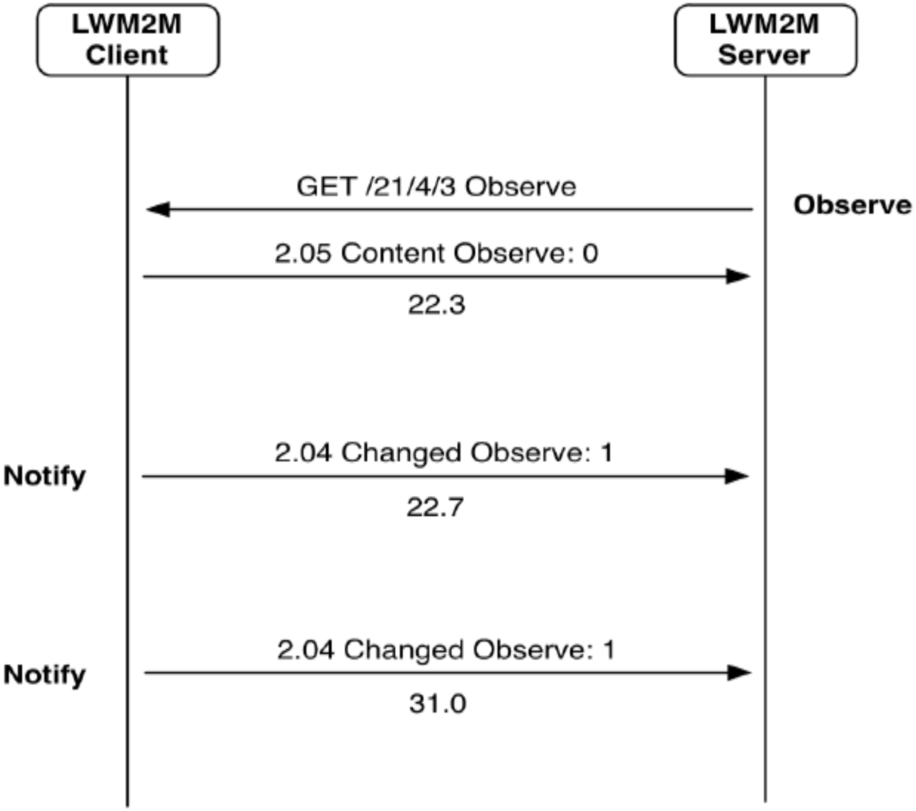
# LWM2M Registration



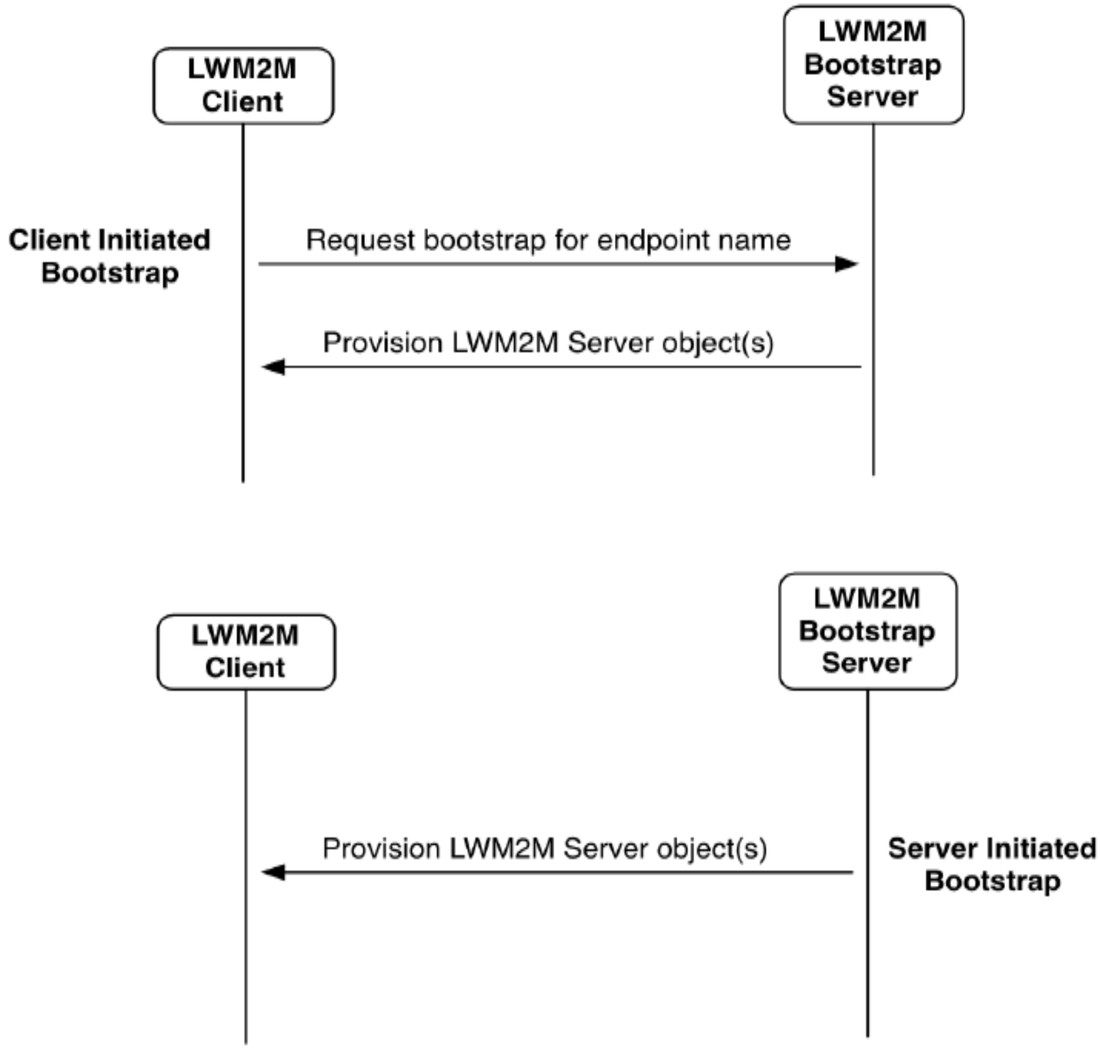
# LWM2M Object Access



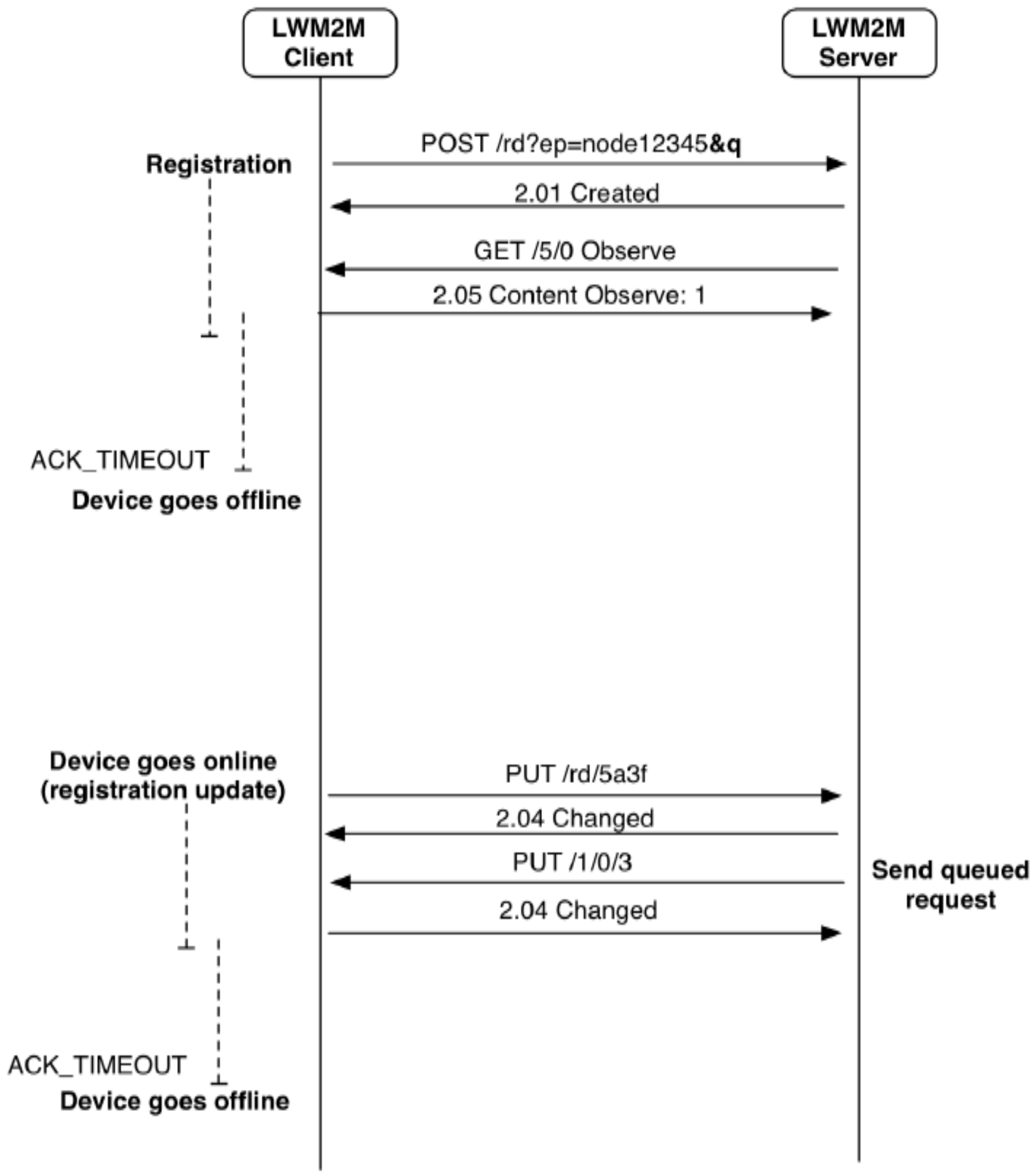
# LWM2M Notification



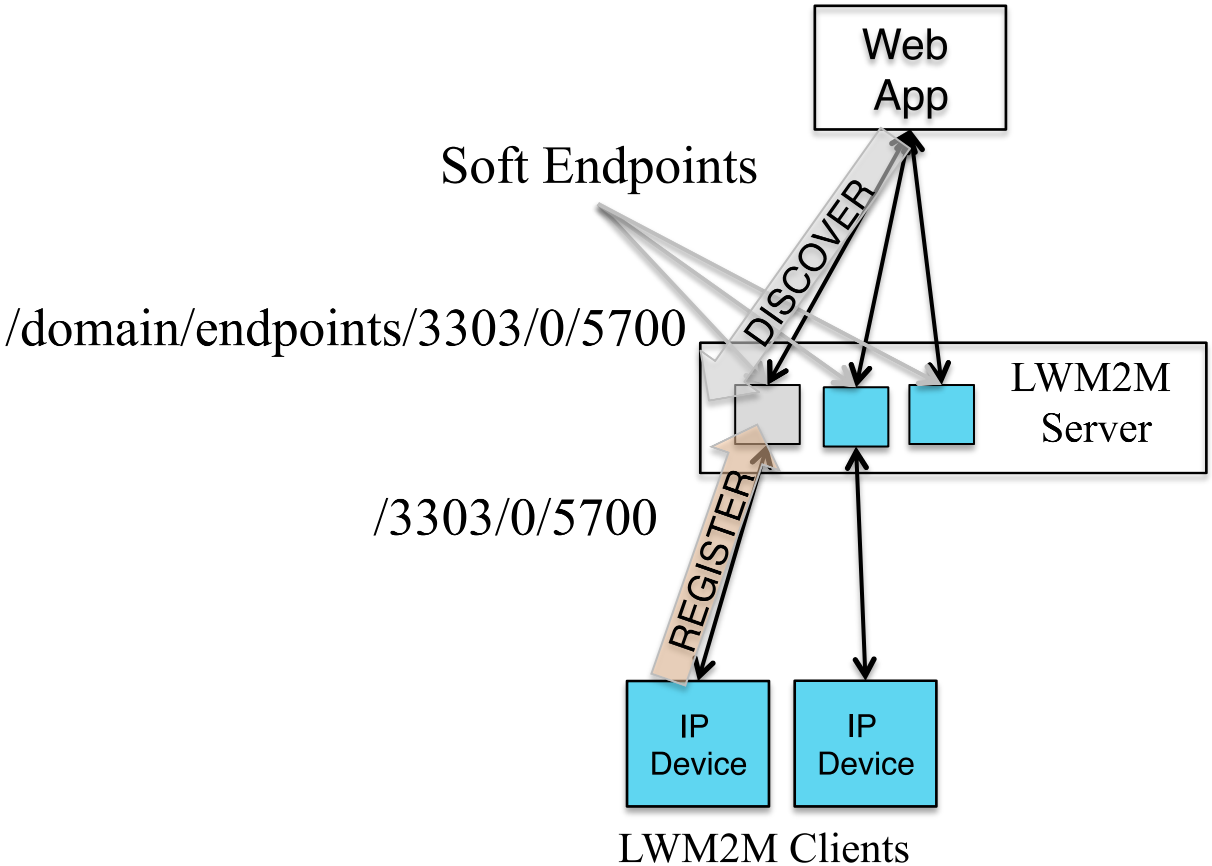
# LWM2M Bootstrapping



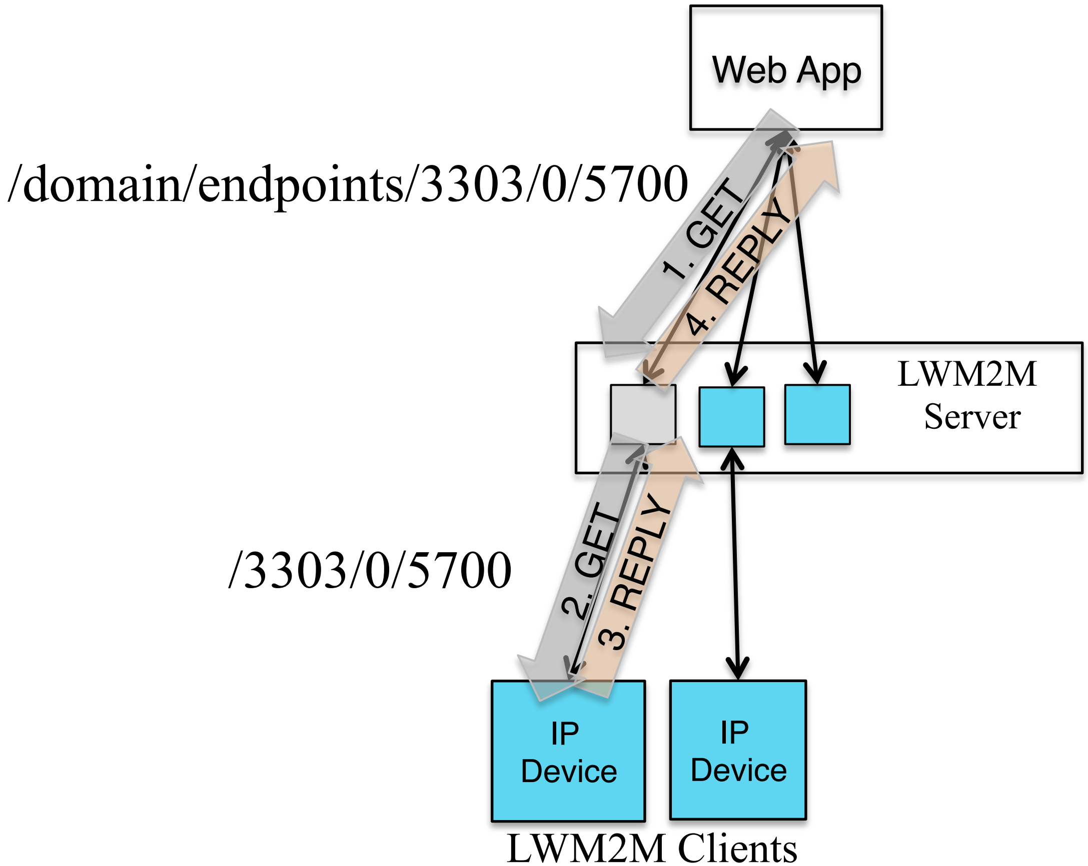
# LWM2M Queue Mode (Sleeping Devices)



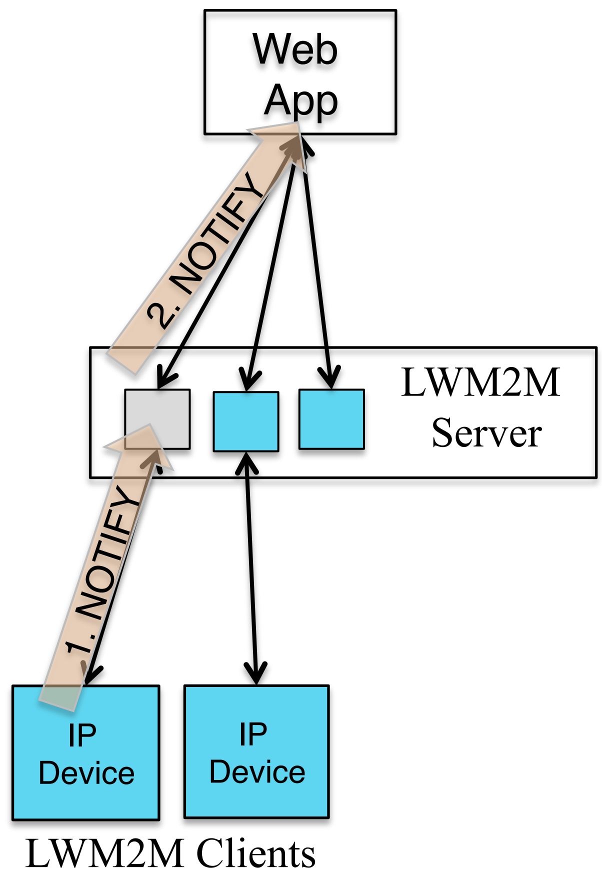
# LWM2M Application Server



# LWM2M Application Server

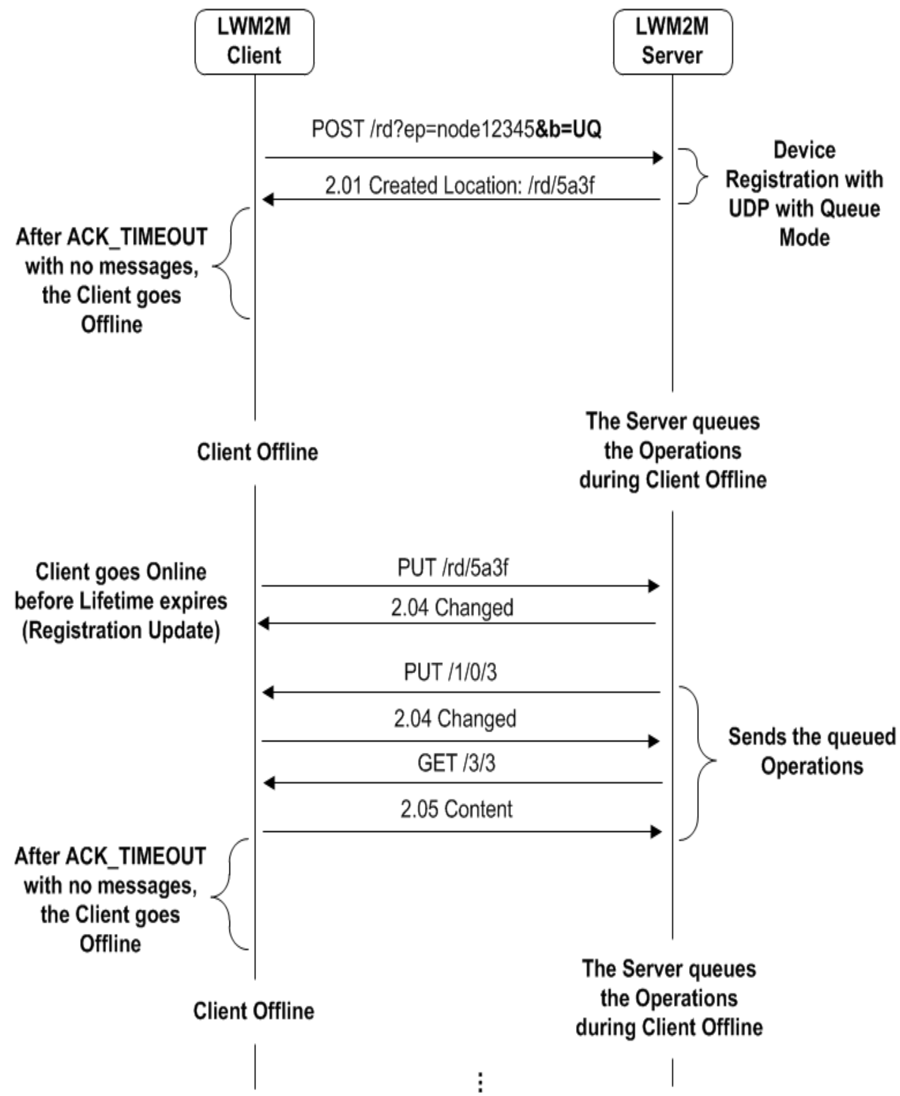


# LWM2M Application Server



# LWM2M Supports Sleeping Endpoints “b=UQ”

- Client uses the registration refresh to inform LWM2M server that it is awake, and listens for any queued operations



# LWM2M support Parameter Observations

- LWM2M provides a mechanism to control Observation
- "Write Attributes" Interface using query parameters to set observe attributes:
  - Pmin – minimum observation quiet period, to limit notification frequency
  - Pmax – maximum observation quiet period, to guarantee notifications
  - Lt – low limit measurement notification, like low alarm, in engineering units
  - Gt – high limit measurement notification, like a high alarm, in engineering units
  - Step – Minimum delta change required to notify, in engineering units

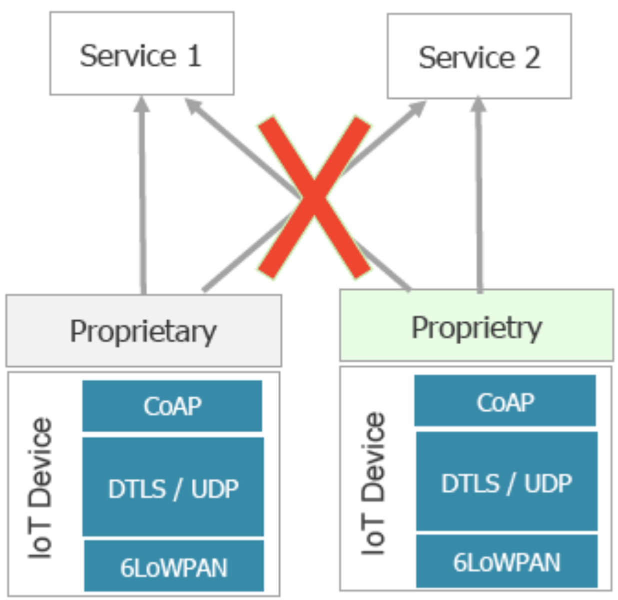
# LWM2M Bulk Read

- Returns TLV or JSON based on requested content-format
- Linked Objects are supported

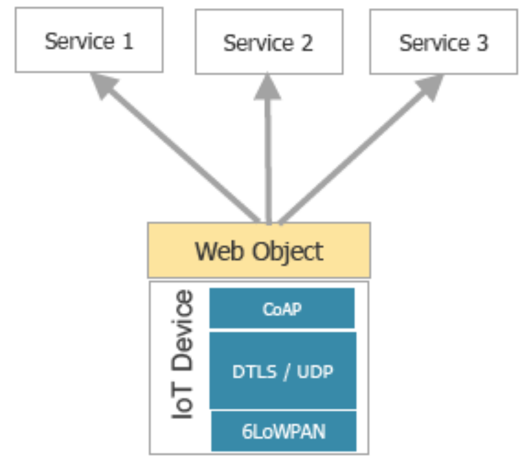

{“e”:[
 {"n":"0","sv":"Open Mobile Alliance"},
 {"n":"1","sv":"Lightweight M2M Client"},
 {"n":"2","sv":"345000123"},
{"n":"3","sv":"1.0"},
{"n":"6/0","v":"1"},
 {"n":"6/1","v":"5"},
 {"n":"7/0","v":"3800"},
 {"n":"7/1","v":"5000"},
 {"n":"8/0","v":"125"},
 {"n":"8/1","v":"900"},
 {"n":"9","v":"100"},
 {"n":"10","v":"15"},
 {"n":"11/0","v":"0"},
 {"n":"13","v":"1367491215"},
 {"n":"14","sv":"+02:00"},
{"n":"15","sv":"U"}]
}

# IPSO Smart Objects

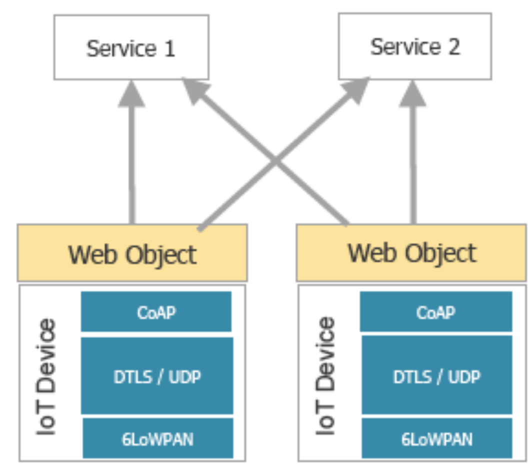
# The Need of Standard Web Object Definition for Service/Device Interoperability




*Non-interoperable devices & Services*




*Interoperable Services*




*Interoperable Devices & Services*

In addition to data communication we need standard web objects for Interoperability

# IPSO Web Objects

- The IPSO Alliance promotes the Internet Protocol for Smart Objects
- We need semantics to build Web of Things
- Web Objects exposes the STATE and BEHAVIOR of a device
- IPSO defines Web Object guidelines

# IPSO Smart Object Example: Temperature Sensor

## Purpose : Define *state* and *behavior of a device*.

Example
Temperature sensor: This IPSO object should be used over a temperature sensor to report a remote temperature measurement. It also provides resources for minimum/maximum measured values and the minimum/maximum range that can be measured by the temperature sensor. The unit used here is Celsius degree.

*Object info*

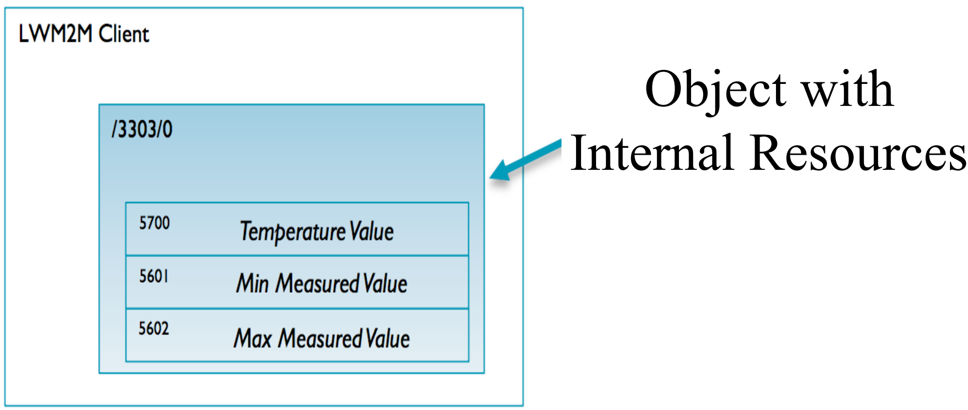
| Object | Object ID | Object URN | Multiple Instances? |
|---|---|---|---|
| IPSO Temperature | 3303 | urn:oma:lwm2m:ext:3303 | Yes |

*Resource Info*

| Resource Name | Resource ID | Access Type | Multiple Instances? | Type | Units | Descriptions |
|---|---|---|---|---|---|---|
| Sensor Value | 5700 | R | No | Decimal | Cel | This resource type returns the Temperature Value in °C |
| Min Measured Value | 5601 | R | No | Decimal | Cel | The minimum value measured by the sensor since it is ON |
| Max Measured Value | 5602 | R | No | Decimal | Cel | The maximum value measured by the sensor since it is ON |

Accessing the Resources

- *Temperature Value* — */3303/0/5700*
- *Min Measured Value* — */3303/0/5601*
- *Max Measured Value* — */3303/0/5602*



# IPSO Smart Objects Use the OMA LWM2M Object Model

- REST API with a URI template
  - Objects
  - Object Instances
  - Resources
  - (Resource Instances)
- Reusable resource and object IDs
  - Common definitions for concepts
  - Map to semantic terms e.g. temperature, currentValue
  - IDs are registered with the OMNA
- Can be embedded in a path hierarchy on the server
  - /home/weather/3303/0/5700

`3303/0/5700`

Object ID, defines object type

Object Instance, one or more

Resource ID, defines resource type

# IPSO Smart Object Starter Pack

**Table 1 Smart Objects defined by this Technical Guideline**

| Object | Object ID | Multiple Instances? |
|---|---|---|
| **IPSO Digital Input** | 3200 | Yes |
| **IPSO Digital Output** | 3201 | Yes |
| **IPSO Analogue Input** | 3202 | Yes |
| **IPSO Analogue Output** | 3203 | Yes |
| **IPSO Generic Sensor** | 3300 | Yes |
| **IPSO Illuminance Sensor** | 3301 | Yes |
| **IPSO Presence Sensor** | 3302 | Yes |
| **IPSO Temperature Sensor** | 3303 | Yes |
| **IPSO Humidity Sensor** | 3304 | Yes |
| **IPSO Power Measurement** | 3305 | Yes |
| **IPSO Actuation** | 3306 | Yes |
| **IPSO Set Point** | 3308 | Yes |
| **IPSO Load Control** | 3310 | Yes |
| **IPSO Light Control** | 3311 | Yes |
| **IPSO Power Control** | 3312 | Yes |
| **IPSO Accelerometer** | 3313 | Yes |
| **IPSO Magnetometer** | 3314 | Yes |
| **IPSO Barometer** | 3315 | Yes |

# Ad-Hoc IPSO Smart Object Example: BLE Heart Rate Sensor Profile

Object info:

| Object | Object ID | Object URN | Multiple Instances? | Description |
|---|---|---|---|---|
| **Heart Rate** | 12200 | urn:oma:lwm2m:x:12200 | Yes | Heart Rate Monitor |

Resource Info:

| Resource Name | Resource ID | Access Type | Multiple Instances? | Mandatory | Type | Range or Enumeration | Units | Descriptions |
|---|---|---|---|---|---|---|---|---|
| **Sensor Value** | 5700 | R | No | Mandatory | Float | | BPM | Heart Rate Measurement Value |
| **Digital Input State** | 5500 | R | No | Optional | Boolean | | | Sensor contact status 0=no contact, 1= contact |
| **Total Energy** | 5950 | R | No | Optional | Float | | kJ | Energy Expended |
| **Reset Cumulative Energy** | 5822 | E | No | Optional | Opaque | | | Reset 5950 Energy Expended to zero |
| **Body Sensor Location** | 5951 | R,W | No | Optional | String | | | Intended sensing location on the body |
| **R-R Interval** | 5952 | R | No | Optional | String | | | Sequence of R-wave intervals |

# Ad-Hoc IPSO Smart Object Example - A Smart Thermostat

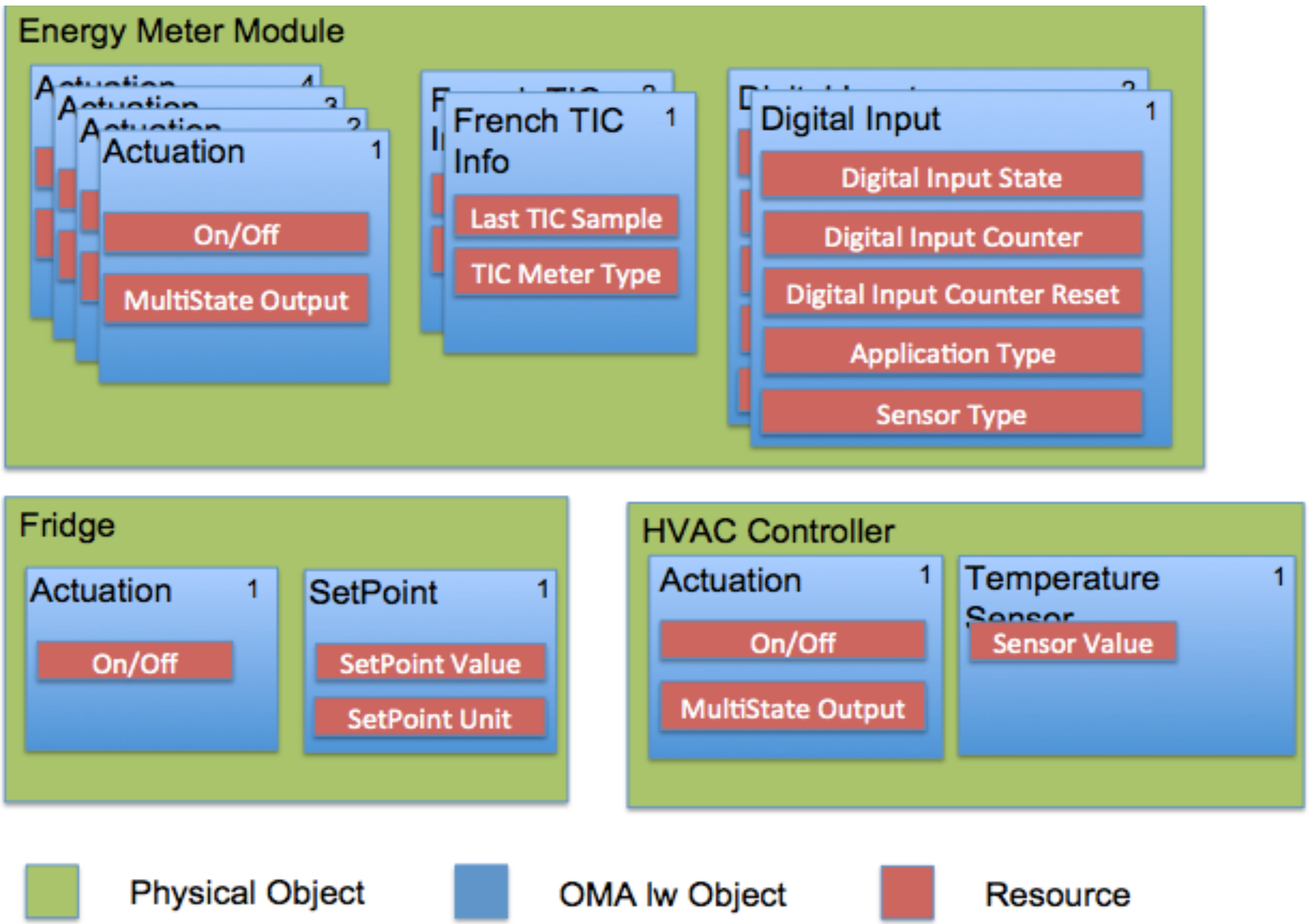
Object info:

| Object | Object ID | Object URN | Multiple Instances? | Description |
|---|---|---|---|---|
| **Smart Thermostat** | 12300 | urn:oma:lwm2m:x:12300 | Yes | Smart Thermostat with multiple settings |

Resource Info:

| Resource Name | Resource ID | Access Type | Multiple Instances ? | Mandatory | Type | Range or Enumeration | Units | Descriptions |
|---|---|---|---|---|---|---|---|---|
| **Sensor Value** | 5700 | R | No | Mandatory | Float | | Per Units resource | Temperature measurement |
| **Units** | 5500 | R,W | No | Mandatory | String | ucum:degF, ucum:degC | | Units for 5700 |
| **Application Type** | 5750 | R,W | No | Optional | String | | | Name, e.g. “Hall Thermostat” |
| **Cooling** | 5200 | R | No | Optional | Boolean | | | 1=cooling |
| **Heating** | 5201 | R | No | Optional | Boolean | | | 1=heating |
| **Heat Source** | 5203 | R | No | Optional | String | “Emergency” , “Normal” | | Indicates heat source |
| **Fan Timer Active** | 5204 | R,W | No | Optional | Boolean | | | 1=running |
| **Fan Timeout** | 5205 | R,W | No | Optional | String | | UTS | Time for fan to stop |
| **Energy Save Mode** | 5206 | R,W | No | Optional | Boolean | | | 1= Energy Save mode |
| **Away Mode** | 5207 | R,W | No | Optional | Boolean | | | 0=Home, 1=Away |
| **Setpoint** | 5208 | R | No | Optional | Float | | | Desired Temperature |
| **HVAC Mode** | 5209 | R,W | No | Optional | String | “Heat”, “Cool”, “Heat-Cool” | | System Mode |
| **High Setpoint** | 5210 | R,W | No | Optional | Float | | | Highest desired temperature |
| **Low Setpoint** | 5211 | R,W | No | Optional | Float | | | Lowest desired temperature |
| **High Away Setpoint** | 5212 | R,W | No | Optional | Float | | | Highest away mode temperature |
| **Low Away Setpoint** | 5213 | R,W | No | Optional | Float | | | Lowest away mode temperature |

# Composite IPSO Smart Objects



# IPSO Smart Object Development

- Smart Objects are Easy to Modify and Customize
  - Based on Consistent Design Patterns and Reusable Resource Definitions
  - Object Sets can be Forked and Modified
  - Expecting Domain-Specific Object Sets to be Created by Collaborative Vertical Working Groups
  - New Object Sets can be Released as new Smart Object Guidelines
  - Objects in Released Smart Object Guidelines are Registered with the OMA, Use Standard OMA DDF (XML) File Format Object Descriptors

# IPSO Smart Objects Future Work Examples

## Linked Composite Objects

- Gateway Management Objects – Mapping of TR-069 to REST
- Behavioral Objects – Smart Objects to represent embedded Timers, Sequencers, Controllers and bindings to resources
- Mapping and Binding of Smart Objects to Zigbee Application Clusters (OnOff Cluster Example)
- Mapping and Binding of Smart Objects to Bluetooth Application Profiles (Heart Monitor Example)
- Advanced Lighting Objects

# IPSO/LWM2M Uses CoRE RD Resource Links (RFC 6690)

```
<4001/0/9002>;rt=“oma.lwm2m”;ct=50;obs=1
```

Resource Type

Content Type

Observable

- Links are uploaded during registration to inform the LWM2M server about resources on the endpoint
- Links are discovered using GET with content type “application/link-format”
- JSON representation using content type “application/link-format+json”

# Summary

**Application Software**

- Not tied to specific device or protocol
- Any Programming Language
- Runs on devices, gateways & services

**IPSO Smart Objects**

- Application Level Interoperability
- Reusable Device to Application API
- Not tied to any specific protocol

**OMA LWM2M**

- Service Layer Specification
- Device Management over CoAP
- Object Model for DM and Applications

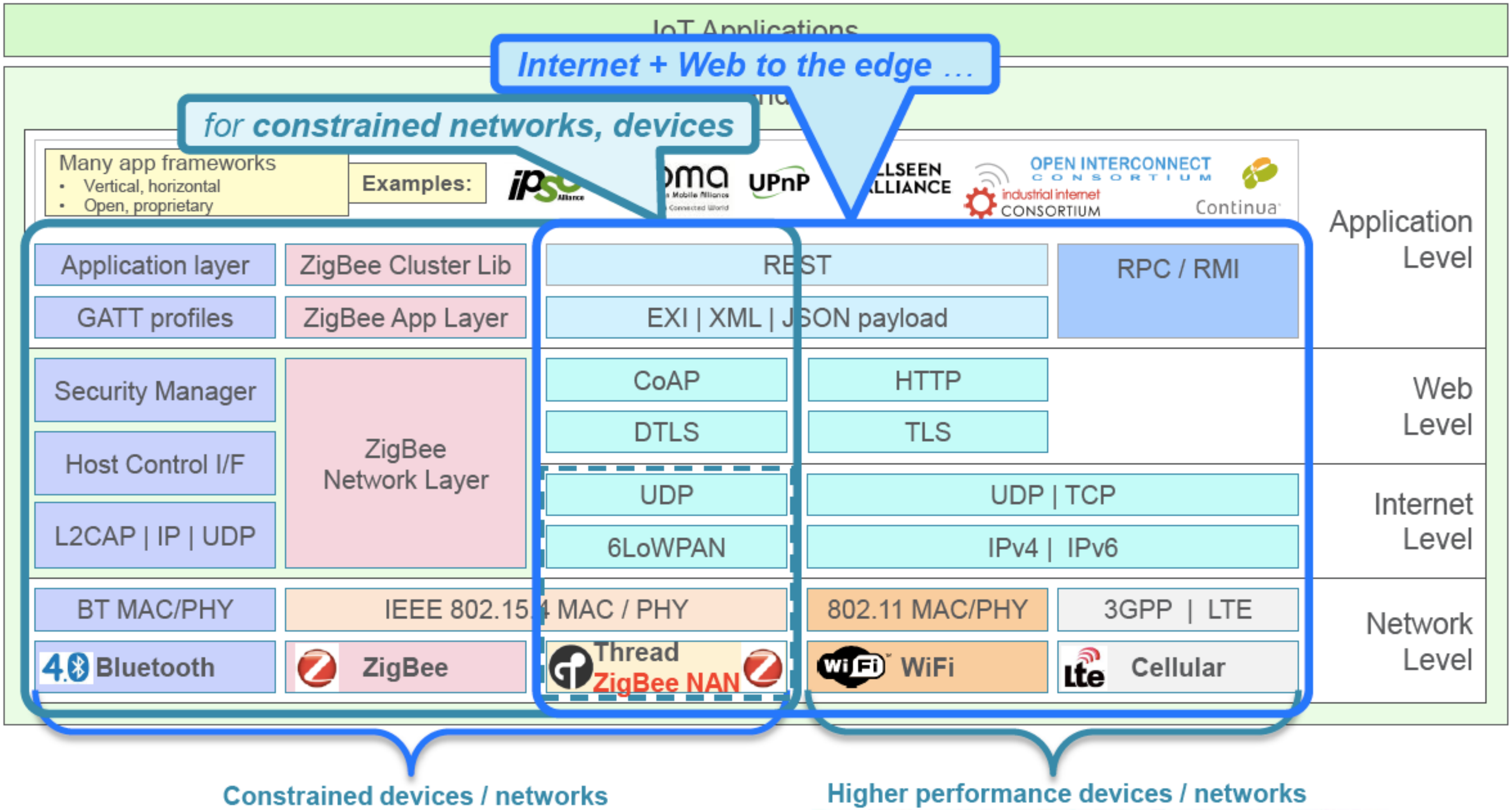
**CoAP**

- REST protocol for constrained devices
- IETF Standard (RFC 7252)
- Uses TCP or UDP, any IP connection
- Discovery using IP Multicast or Directory

# References

- IPSO Smart Object Guideline
  http://www.ipso-alliance.org/smart-object-guidelines
- OMA LWM2M Specification
  http://openmobilealliance.hs-sites.com/lightweight-m2m-specification-from-oma
- IETF CoAP and Related Specifications
  CoAP (RFC 7252):
  http://tools.ietf.org/html/rfc7252
  CoRE Link-Format (RFC 6690):
  http://tools.ietf.org/html/rfc6690
  CoRE Resource Directory:
  http://tools.ietf.org/html/draft-ietf-core-resource-directory-01
- CoAP Community Site
  http://coap.technology/

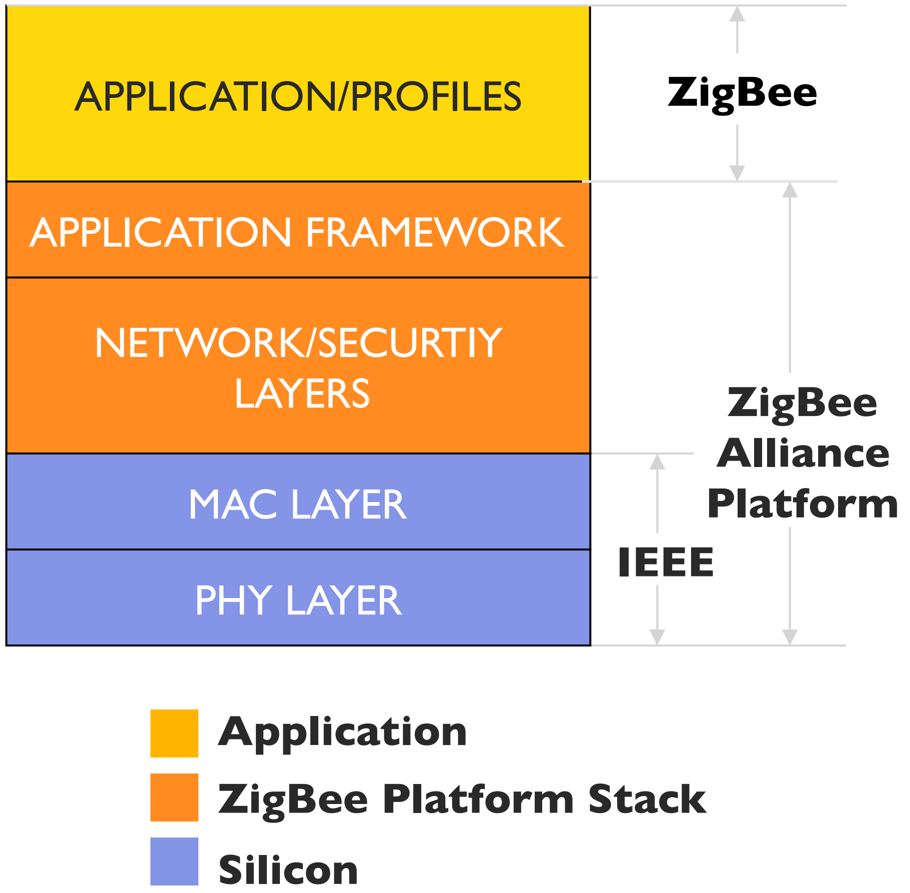
# Recall: Other mainstream IoT Protocol Stacks



# Overview of ZigBee:
# An Alternative Protocol Stack for IoT

# ZigBee Protocol Stack

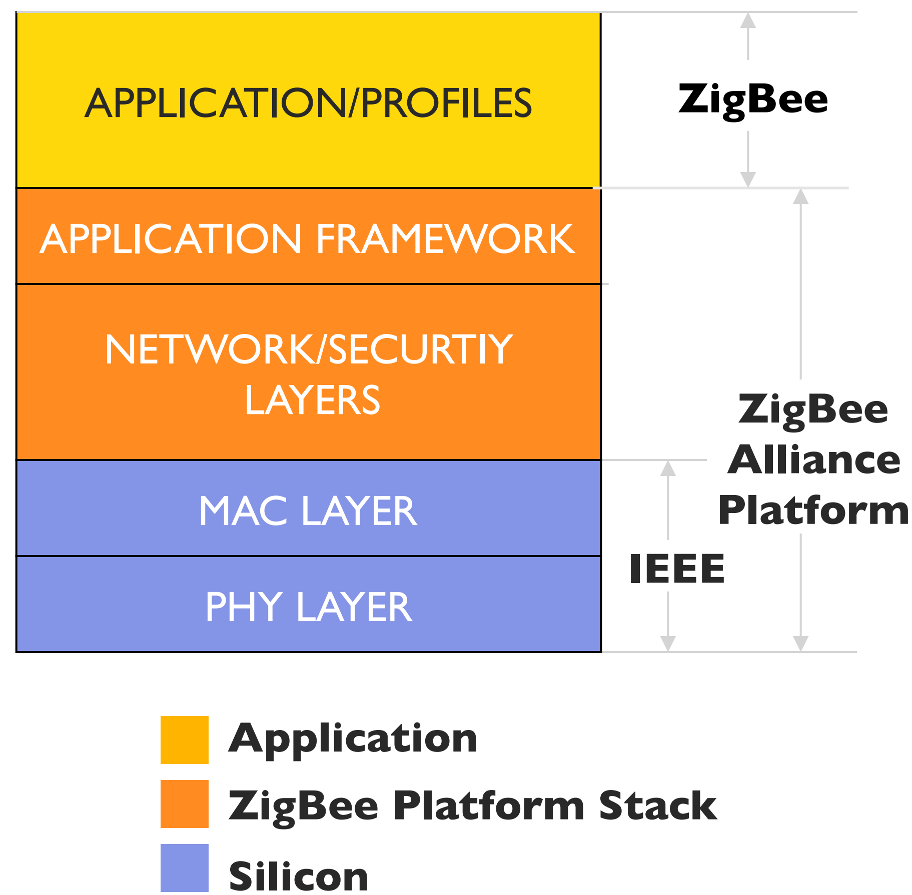
- ZigBee
  - Based upon the international IEEE 802.15.4 standard for PHY/MAC
- IEEE STD 802.15.4®
  - Designed by Motorola, Philips and other companies to supply the radio and protocol, allowing the designer to concentrate on the application and their customers' needs



# The ZigBee Protocol Stack

- ZigBee aims to address the needs of most remote monitoring/ control and sensor network applications

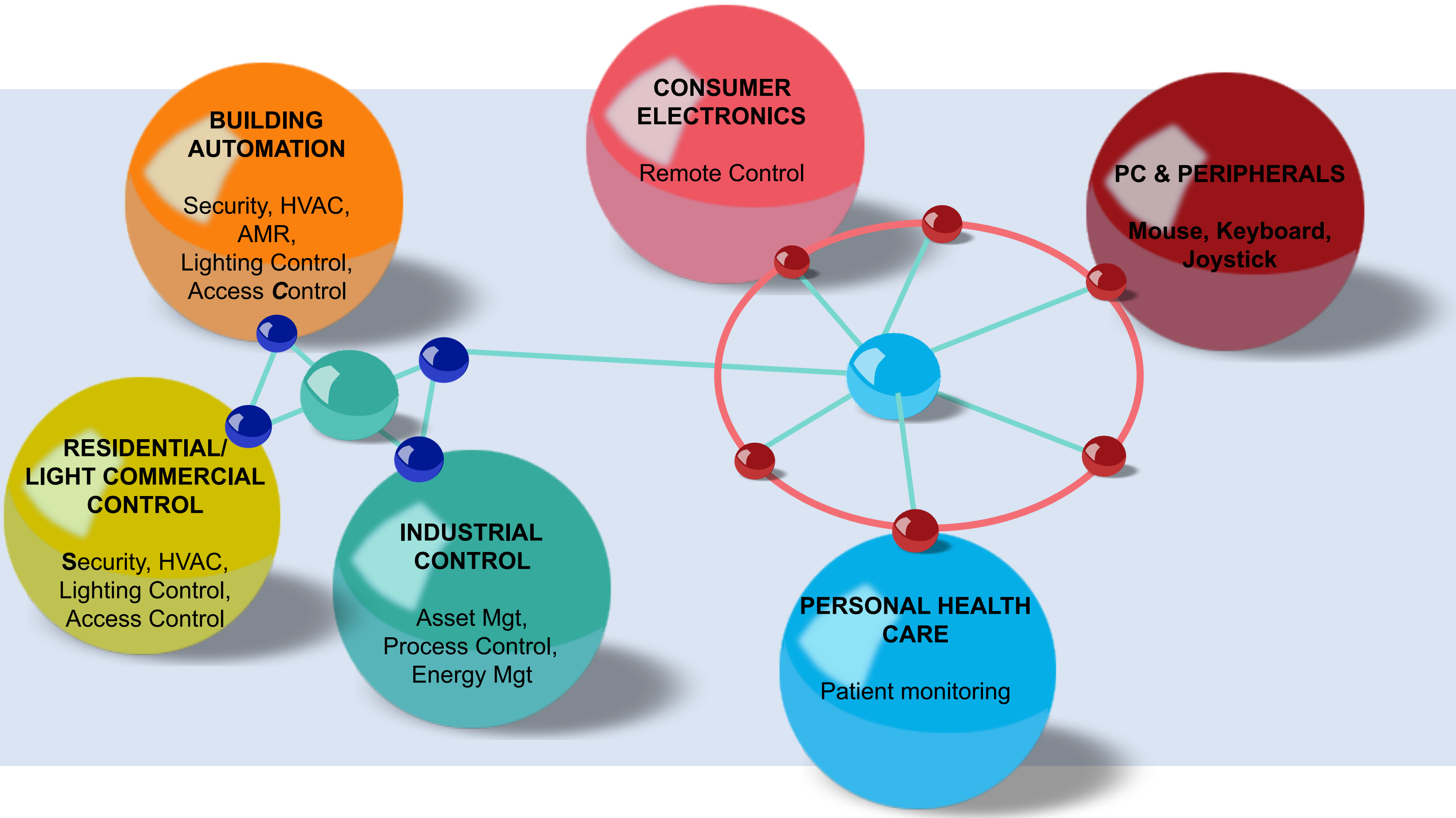
- Relationship between ZigBee and IEEE 802.15.4
  - ZigBee takes full advantage of a powerful physical radio specified by IEEE 802.15.4
  - ZigBee adds logical network (NWK), security and application framework and profiles



# ZigBee Features

- ZigBee is designed to be a low power, low cost, low data rate, wireless solution.
- ZigBee relies upon the robust IEEE 802.15.4 PHY/MAC to provide reliable data transfer in noisy, interference-rich environments
- ZigBee layers on top of 15.4 with Mesh Networking, Security, and Applications control
- ZigBee Value Propositions
  - Addresses the unique needs of most remote monitoring and control network applications
    - Infrequent, low rate and small packet data
  - Enables the broad-based deployment of wireless networks with low cost & low power solutions
    - Example: Lighting, security, HVAC,
    - Supports peer-to-peer, star and mesh networks
  - Monitor and sensor applications that need to have a battery life of years on alkaline batteries
    - Example – security systems, smoke alarms

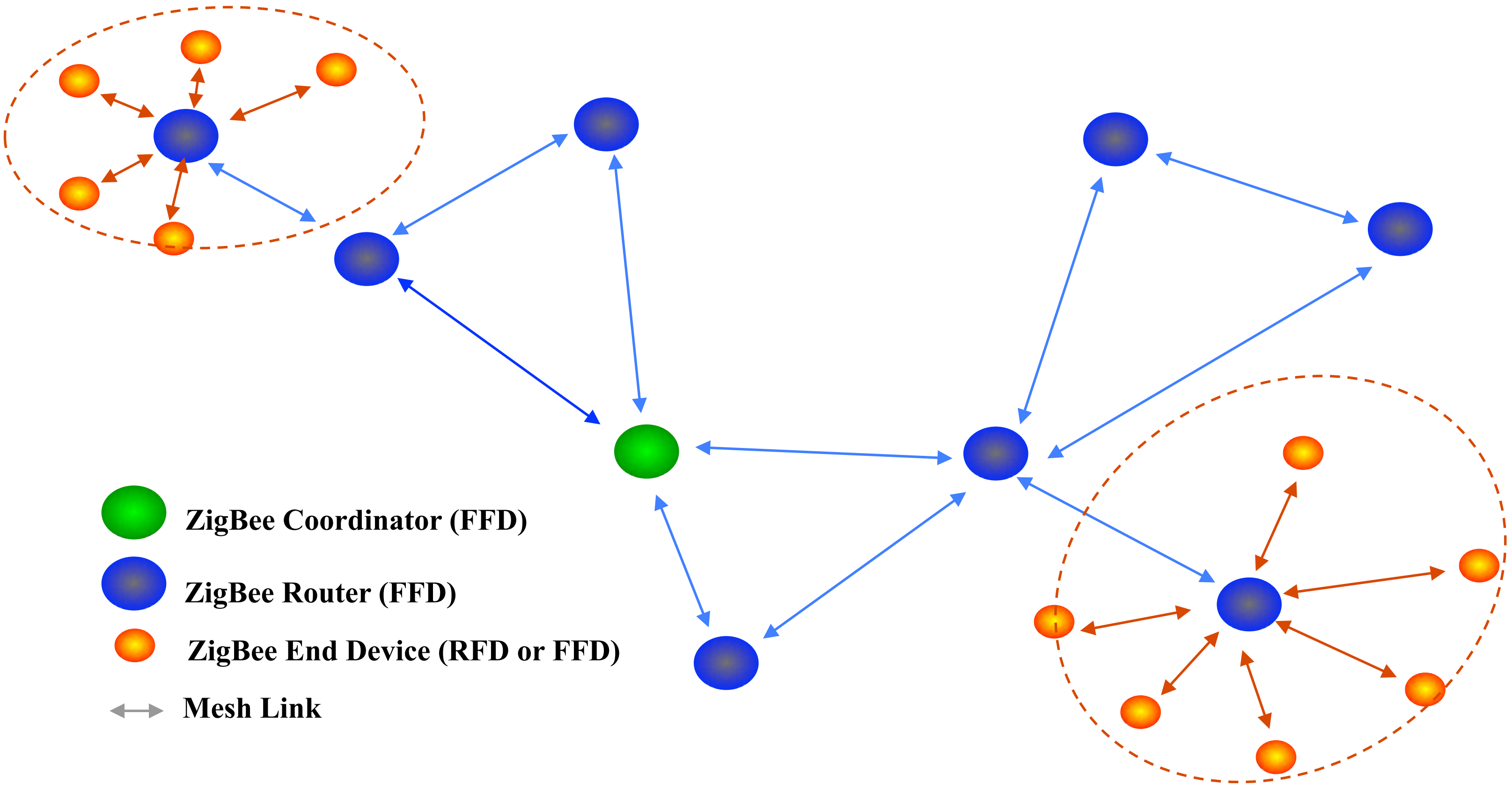
# ZigBee Wireless Markets and Applications



# ZigBee Feature Set

- ZigBee V1.0
  - Ad-hoc self forming networks
    - Mesh, Cluster Tree and Star
  - Logical Device Types
    - Coordinator, Router and End Device
  - Applications
    - Device and Service Discovery
    - Messaging with optional responses
    - Home Controls Lighting Profile
    - General mechanism to define private Profiles
  - Security
    - Symmetric Key with AES-128
    - Authentication and Encryption at MAC, NWK and Application levels
    - Master Keys, Network Keys and Link Keys
  - Qualification
    - Conformance Certification (Platform and Profile)
    - Interoperability Events

# ZigBee Network Model




- Star networks support a single ZigBee coordinator with one or more ZigBee End Devices (up to 65,536 in theory)
- Mesh network routing permits path formation from any source device to any destination device

# ZigBee Stack Architecture Basics

- Addressing
  - Every device has a unique 64 bit MAC address
  - Upon association, every device receives a unique 16 bit network address
    - A ZigBee Network DOES NOT use IP-addressing !!
  - Only the 16 bit network address is used to route packets within the network
  - Devices retain their 16 bit address if they disconnect from the network, however, if the LEAVE the network, the 16 bit address is re-assigned
  - Network-wide (NWK), i.e. multi-hop broadcast implemented above the MAC:
    - Network (NWK) address 0xFFFF is the broadcast address
    - Special algorithm in ZigBee Network Layer to propagate the message
    - “Best Effort” or “Guaranteed Delivery” options
    - Radius Limited Broadcast feature

# How A ZigBee Network Forms

- Devices are pre-programmed for their network function
  - Coordinator scans to find an unused channel to start a network
  - Router (mesh device within a network) scans to find an active channel to join, then permits other devices to join
  - End Device will always try to join an existing network
- Devices discover other devices in the network providing complementary services
  - Service Discovery can be initiated from any device within the network
- Devices can be bound to other devices offering complementary services
  - Binding provides a command and control feature for specially identified sets of devices

# Detail Steps to form a ZigBee Network

- Network Scan
  - Device scans the 16 channels to determine the best channel to occupy.
- Creating/Joining a PAN
  - Device can create a network (coordinator) on a free channel or join an existing network
- Device Discovery
  - Device queries the network to discover the identity of devices on active channels
- Service Discovery
  - Device scans for supported services on devices within the network
- Binding
  - Devices communicate via command/control messaging

# Comparing ZigBee with other Technologies

(source: Freescale)

| Feature(s) | IEEE 802.11b | Bluetooth | IEEE 802.15.4 |
|---|---|---|---|
| Power Profile | Hours | 1 Week | 1Year+ |
| BOM | $9 | $6 | $3 |
| Complexity | Complex | Very Complex | Simple |
| Nodes/Master | 32 | 7 | 64000 |
| Latency | Enumeration upto 3 seconds | Enumeration upto 10 seconds | Enumeration 30ms |
| Range | 100 m | 10m | 70m |
| Extendability | Roaming possible | No | YES |
| Data Rate | 11Mbps | 1Mbps | 250Kbps |
| Security | Authentication Service Set ID (SSID) | 64 bit, 128 bit | 128 bit AES and Application Layer user defined |

# ZigBee Stack Evolution

- The ZigBee stack specification is defined in a document with ZigBee reference base 053474
- ZigBee 2004
  - 053474r06
- ZigBee 2006
  - 053474r13
- ZigBee PRO (aka ZigBee 2007)
  - Released in 2007
  - 053474r18
  - Basis for ZigBee SE (Smart Energy) v1.0
- ZigBee IP (under the effort of ZigBee SE v2.0 )
  - A Completely DIFFERENT Stack !!
  - More later …

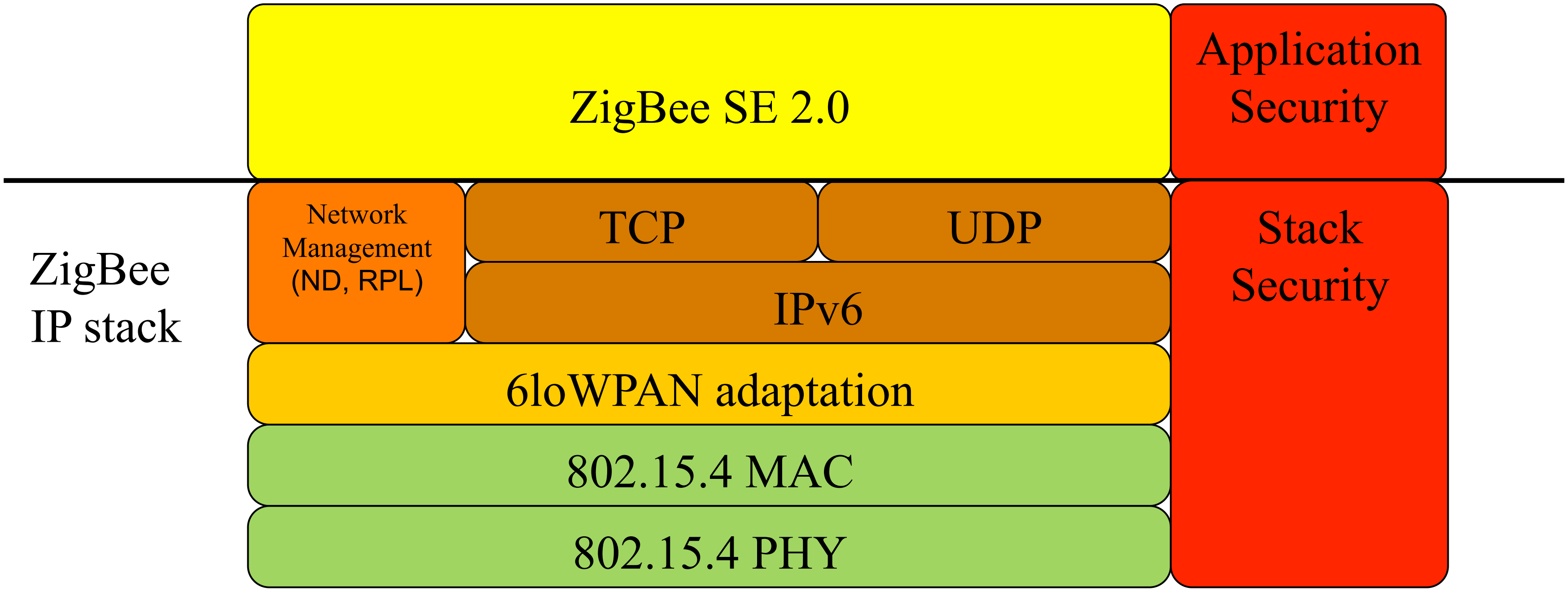
# Why a new, different ZigBee stack ?

- Enable to use multiple MAC/PHYs

  => Split into SE (Smart Energy) 2.0 Application Layer and Underlying stack

  - SE 2.0 Application Layer is Stack Agnostic as it is based on TCP
- ZigBee IP stack is aimed at 802.15.4 networks
  - Leveraging IETF 6loWPAN adaptation layer for IPv6 over 802.15.4
- ZigBee is also developing guidelines for interfacing SE2.0 to HomePlug powerline and other IEEE-based stacks, e.g. Ethernet, 802.11

# What is the ZigBee IP stack ?




- A collection of independent standard specifications, e.g. RFCs, does not produce a standards-based stack which is interoperable across products from different manufacturers
- ZigBee IP specification is a “super-specification” which
  - Uses other standard specifications as its basis
  - Identifies required standard specifications
  - Clarifies modes of Operation to enhance:
    - Interoperability and Streamlining

# ZigBee IP stack Highlights

- IEEE 802.15.4-2006 MAC/PHY
- IETF RFC4944, 6282: 6LowPAN Header Compression Adaptation layer
- IETF RF6775:  6LowPAN Neighbor Discovery
- IPv6 Network Layer
  - RH4 Routing Header
  - Hop-by-hop Header RPL option
- TCP/UDP Transport Layer
- IETF ROLL (Routing Over Low power and Lossy links) Working Group RPL routing protocol RFC6550
  - Non-storing mode
- IETF PANA (RFC5191) /EAP/EAP-TTLSv0/TLS security
  - Public key (ECC and RSA) and PSK cipher suites
- IETF RFC6762 Multicast DNS (mDNS) / DNS-SD Service Discovery support

# Outlook for IoT “Standards”

Many key players are in the running to create THE IoT Standard Architecture/Protocol

- IPSO Alliance, founded 2008
  - ARM, Ericsson, Atmel, Cisco, Google
- The AllSeen Alliance, founded by Qualcomm, Cisco, Microsoft in 2011
  - Released the AllJoyn Open-Source Software Framework with Linux Foundation since Dec 2013, latest stable release in Dec 2016 ;
  - Announced to merge AllJoyn with IoTivity in Oct 2016 (now Apache licensed).
- The Industrial Internet Consortium (IIC), founded March 2014
  - IBM, Intel, Microsoft, Cisco, AT&T
- The Open Interconnect Consortium (OIC), announced July 2014
  - Broadcom, Intel, Atmel, Samsung
  - Renamed to Open Connectivity Foundation (OCF) in Feb 2016 and added Microsoft, Qualcomm, Cisco, GE etc to its membership.
  - Produce IoTivity – an open source reference implementation for OCF specification
- The Thread Group, announced July 2014 to standard a secure wireless mesh protocol stack
  - Nest Labs (Google), ARM, Freescale, Samsung
  - Announced collaboration agreement with ZigBee Alliance, Apr 2015
  - Released OpenThread, an open-source implementation in May 2016
  - Thread announced a Liaison Agreement with OCF in July 2016

See: State of IoT Standards (circa Sept 2016):

https://www.cloudtp.com/doppler/state-iot-standards-2016/