Low-Rate Wireless Personal Area Networks – IEEE 802.15.4

Fernando Solano – Warsaw University of Technology

fs@tele.pw.edu.pl – EIoT 2020L – 20.04.2018

Quick Review

CoAP, UDP, IPv6

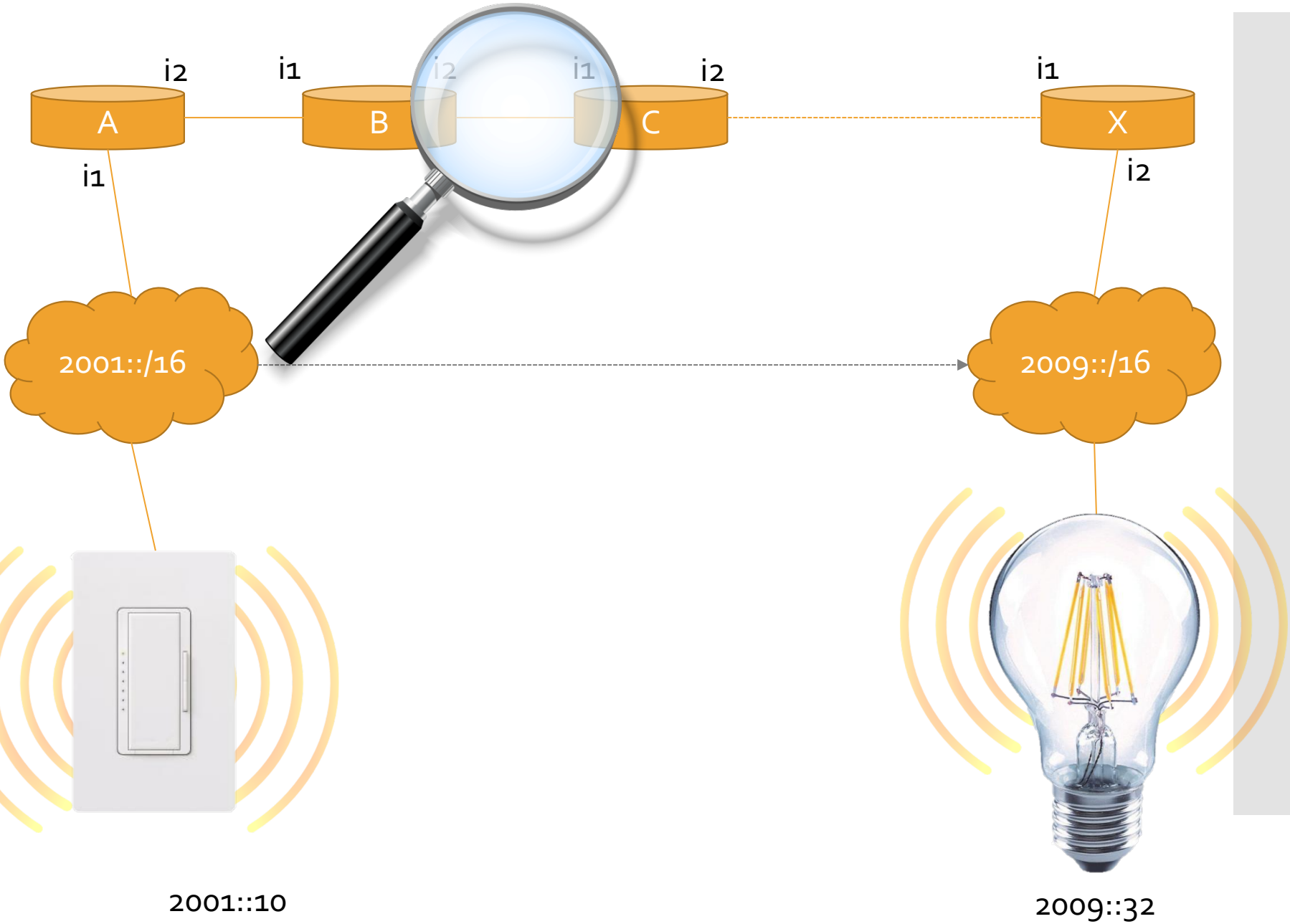
Review of higher layer protocols

Layer	Protocol example	Scope	Typical design question	Example of design answers
Application Layer	**CoAP**, MQTT, HTTP, etc.	Managing Objects	What objects (URI paths) and operations?	GET /.well-known/core GET /sensors/temp PUT /actuators/led1
Transport Layer	**UDP**, TCP	Connecting applications	What (socket) port number? Type of (socket) connection?	Udp port 5683
Network Layer	**IPv6** (~~IPv4~~)	Passing messages across a network of devices	What network addressing scheme and routing protocols to use?	IPv6 address 2000::feo:abcd
Link Layer	?	?	?	?
Physical Layer	?	?	?	?

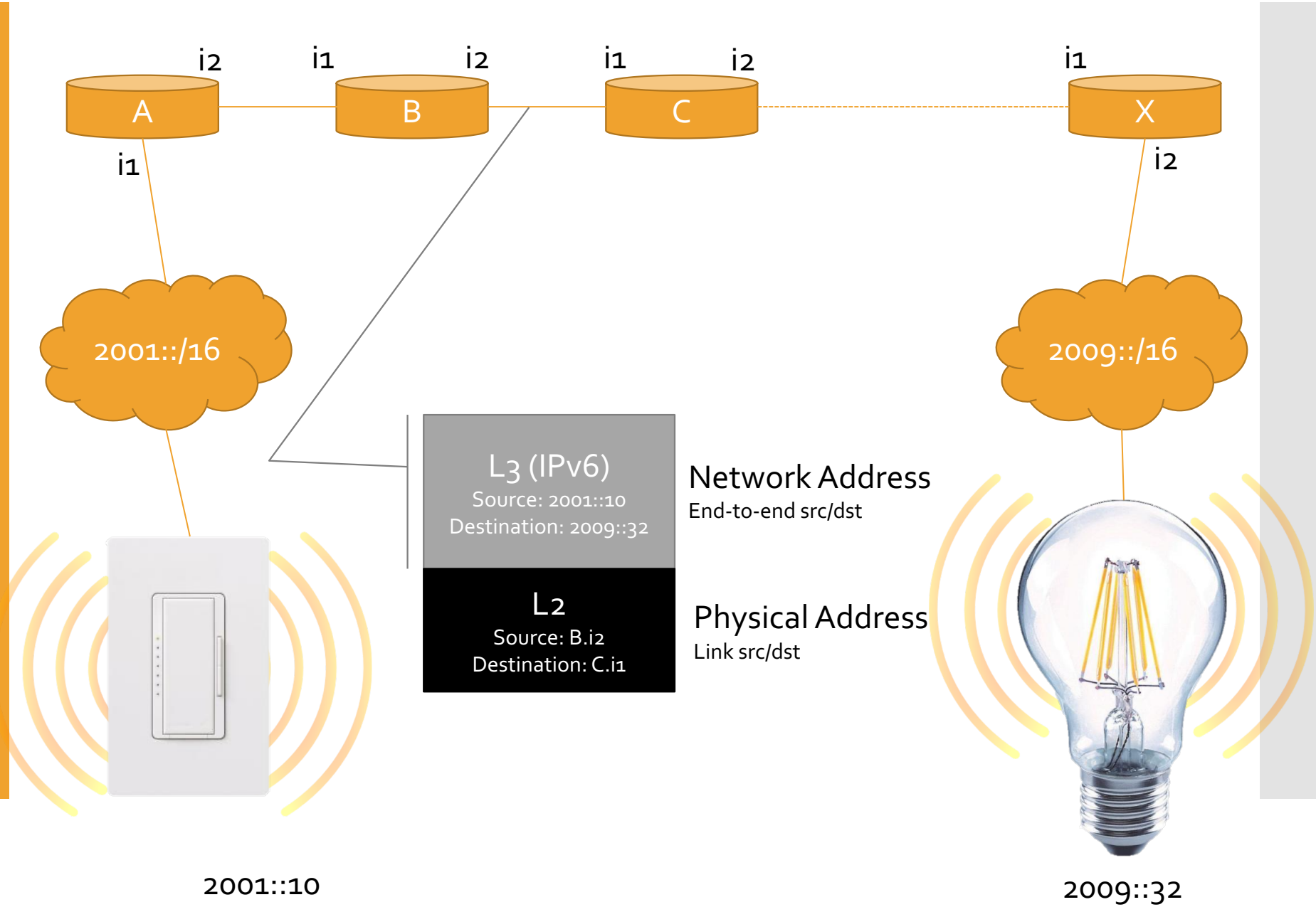
Our link layer protocol

Layer	Protocol example	Scope	Typical design question	Example of design answers
Application Layer	**CoAP**, MQTT, HTTP, etc.	Managing Objects	What objects (URI paths) and operations?	GET /.well-known/core GET /sensors/temp PUT /actuators/led1
Transport Layer	**UDP**, TCP	Connecting applications	What (socket) port number? Type of (socket) connection?	Udp port 5683
Network Layer	**IPv6** (~~IPv4~~)	Passing messages across a network of devices	What network addressing scheme and routing protocols to use?	IPv6 address 2000::feo:abcd
Link Layer	**802.15.4**, Ethernet, WiFi, etc	Passing messages between two neighbouring devices	How to send/receive a message efficiently to/from my neighbour?	Use beacons, non-synchronized

Clarifications L2/L3



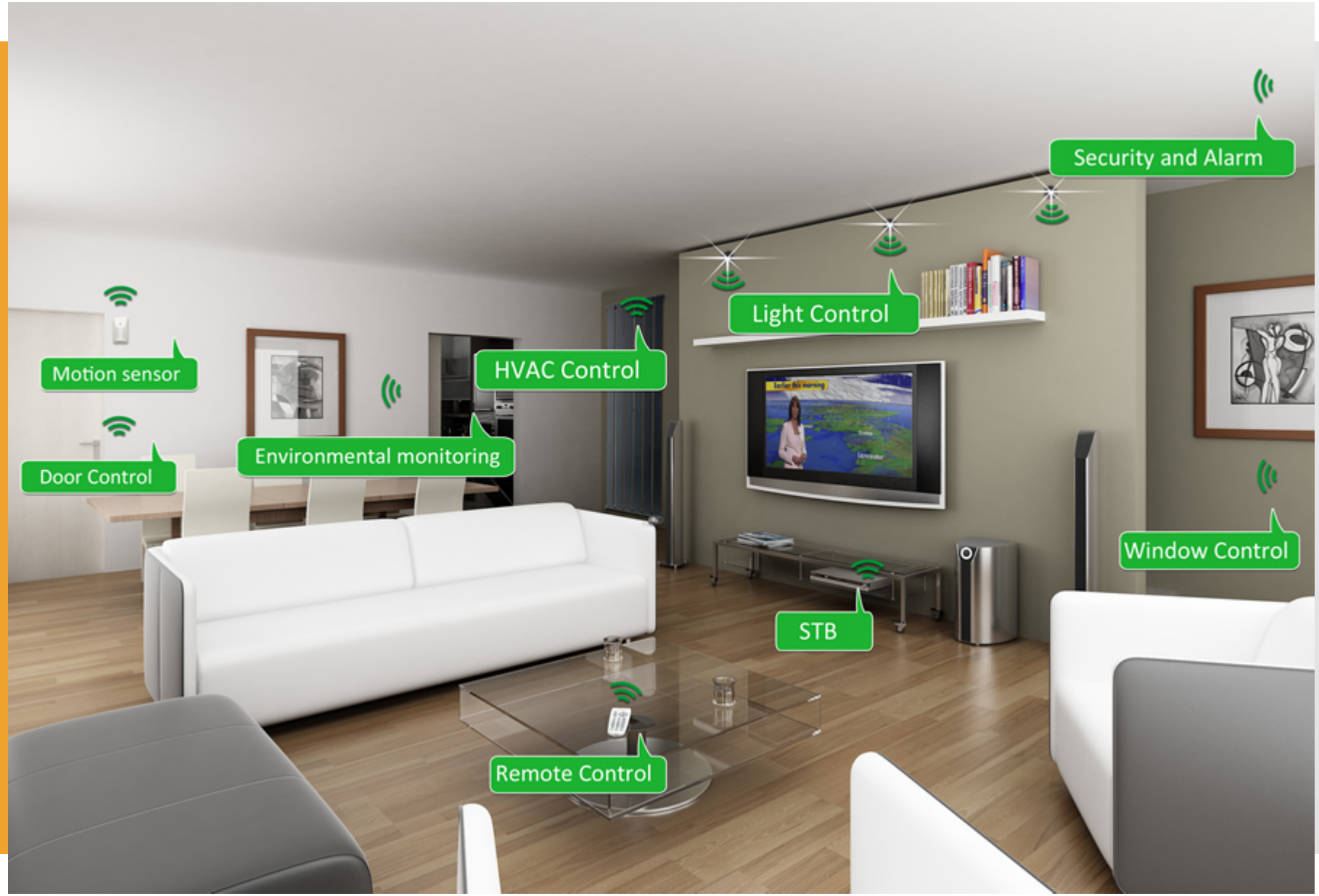
Clarifications L2/L3



Questions to be solved by the Link Layer

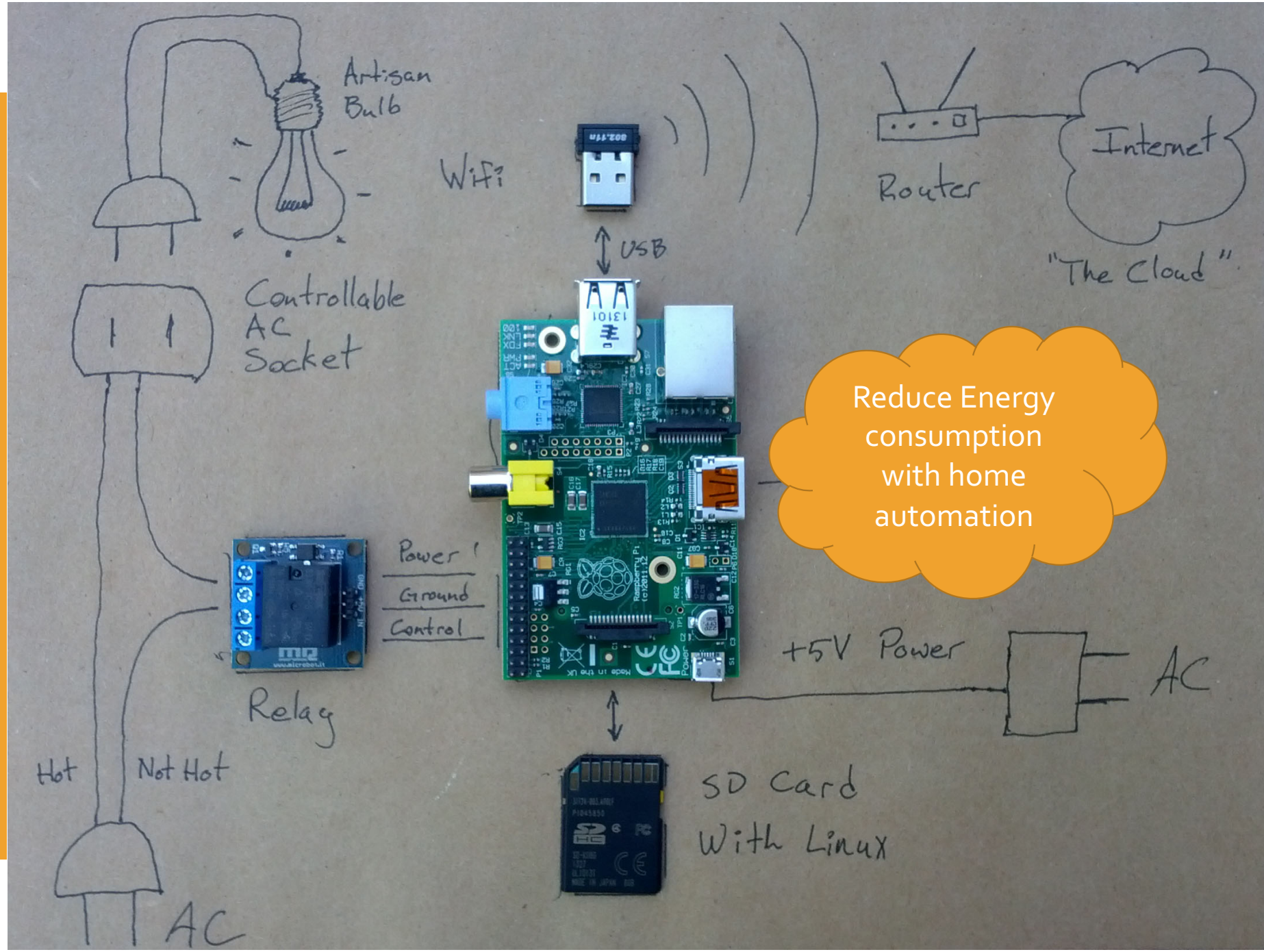
- **How to prevent two (data) frames to be sent at the same time-channel by different devices?**
- **How are physical addresses defined/assigned?**
- How to encode data and to modulate the signal?
- How to detect errors on frame reception?
- How to mark frame start and end?
- What transmission speeds are supported?
- …
- **Specially to IoT:**
  - **How to power-down the transmitter and receiver without loosing data?**

Wireless Sensor Networks and Hardware



A bad example

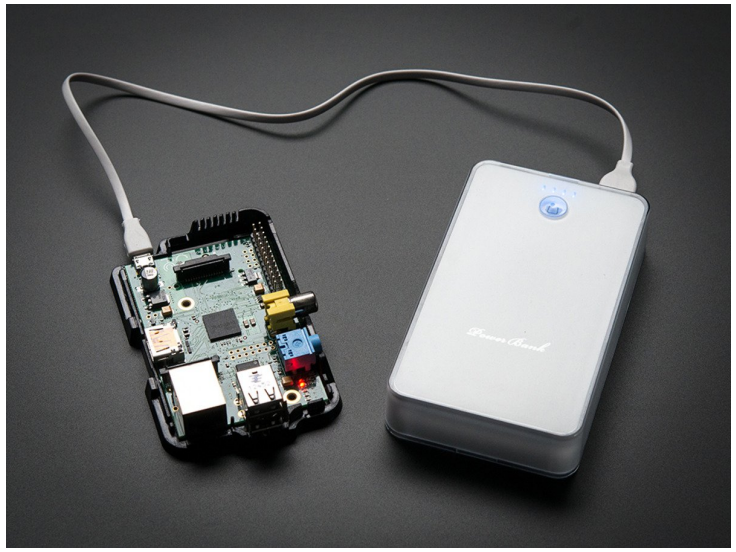
Remote bulb control



A bad example

Remote bulb control

Power Mgte



How often will I need to replace/recharge the battery?

Where can I connect it?
Do I have a plug nearby?

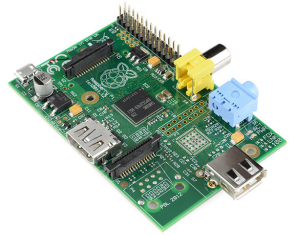
Even if so, what's the point?
Am I saving Energy with this system?

A bad example

Remote bulb control

- **Wireless interface:**
  - **How many bytes do we need to transfer for switching on a bulb?**
  - **What is the WiFi speed?**
  - **What would be the overhead for transfering a small packet?**
  - **Complex modulation (CDMA), complex radio (Direct Sequence Spread-Spectrum)**
- Processor
  - How complex could be our controlling application at the Rpi? (memory/CPU cycles)
  - Can a delay of milliseconds be tolerable? Or is it microseconds justified?
- OS:
  - Can Linux be put into sleep?
  - If not, what's the boot time?
  - Can Linux put to sleep the WiFi independently?
  - Can we reduce the clock speed arbitrarely?

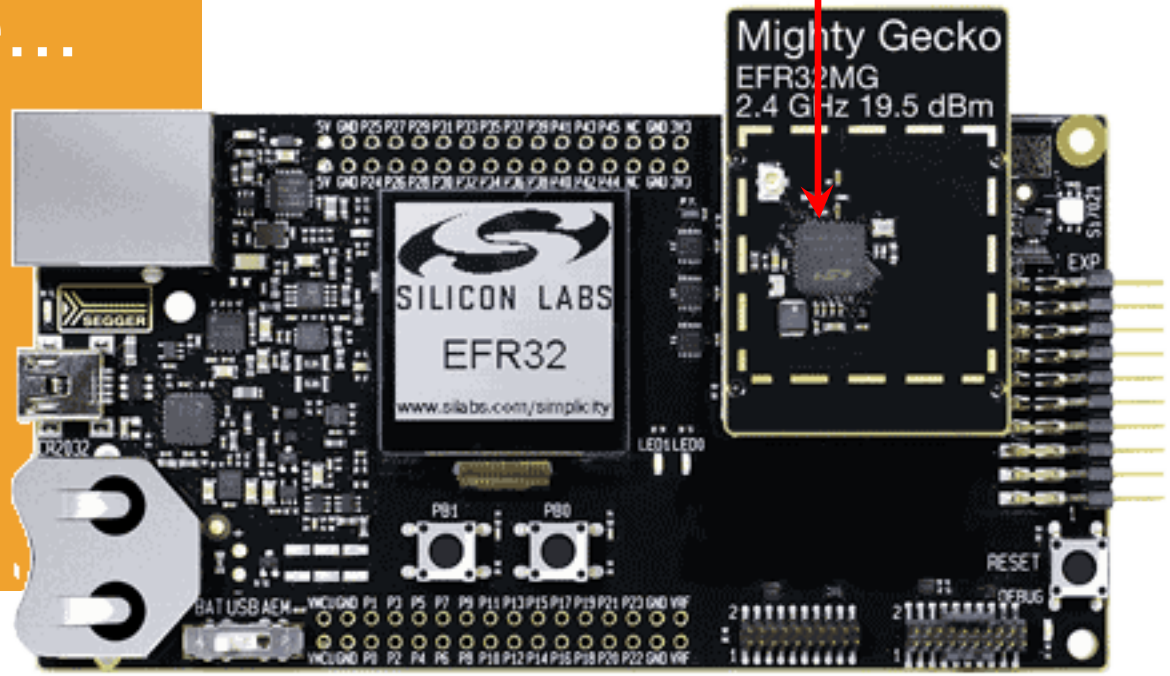
Hardware



	PC	Smartphone	Raspberry Pi
CPU	2.1-3.2 GHz	1.2-1.8GHz	900 MHz
Memory	16-32GB	8-16GB	512 MB
Wireless data rate	~ 1Mbps (BLE) > 11 Mbps (WiFi)		
Power consumption	300 watts	5-12 watts	1 watt (BLE) 2 watts (WiFi)
Battery	-	2000mAh	
Expected battery lifetime	-	1-2 days	4-6 days (if 2000mAh + WiFi)

Assume we deploy a 100 RPI… we'll have to replace 20 batteries a day!

An example…



Controler:

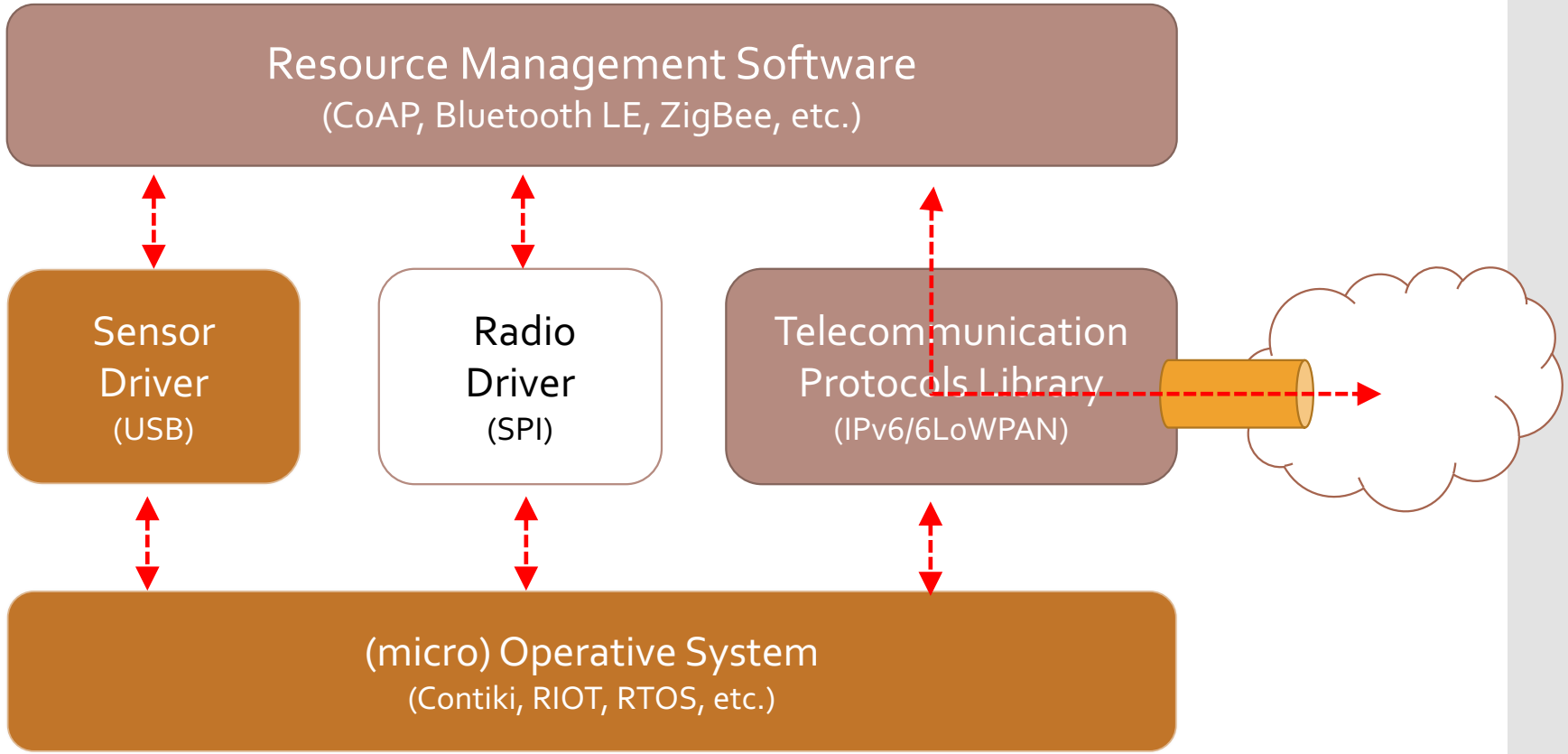
48MHz
256 KB ROM
32KB RAM
< 64 port IO

Radio:

169MHz-2.4GHz
Tx up to +20dBm
Rx of -133 dBm

IoT node

256KB ROM + 32KB RAM



Translated into communications requirements …

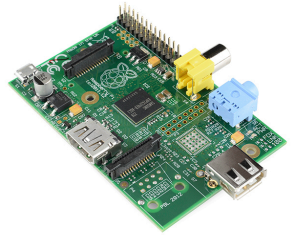
- Simple protocol (small code, small memory)
- Allowing sleeping
- Requiring low-cost hardware

- One possible part of the solution: IEEE 802.15.4

Hardware

802.15.4






	PC	Smartphone	Raspberry Pi	Wireless sensor
CPU	2.1-3.2 GHz	1.2-1.8GHz	900 MHz	**8-48MHz**
Memory	16-32GB	8-16GB	512 MB	**256 KB**
Wireless data rate	> 11 Mbps (802.11)			**20-250 Kbps**
Power consumption	300 watts	5-12 watts	~ 2 watts	**10 mW (on) 10uW (sleep)**
Battery	-	2000mAh		**750mAh**
Expected battery lifetime	-	1-2 days	4-6 days (if 2000mAh)	**4 years @ duty cycle 0.1%**

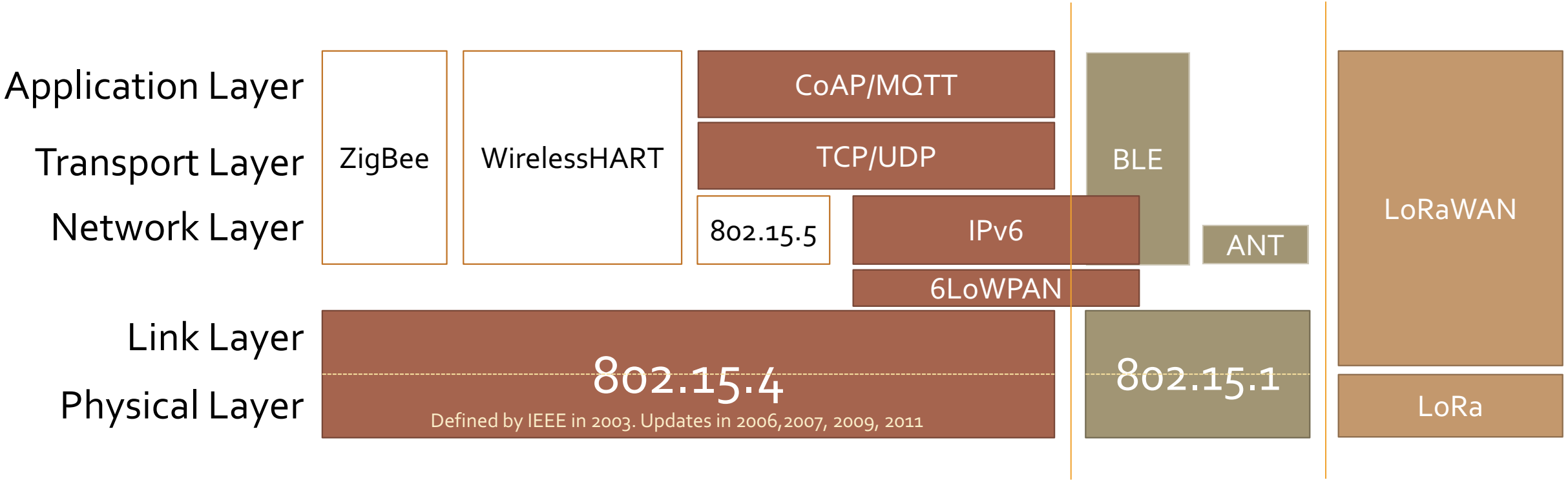
Low-Rate Wireless Personal Area Networks – IEEE 802.15.4

Fernando Solano – Warsaw University of Technology

fs@tele.pw.edu.pl

802.15.4 Multi-purpose

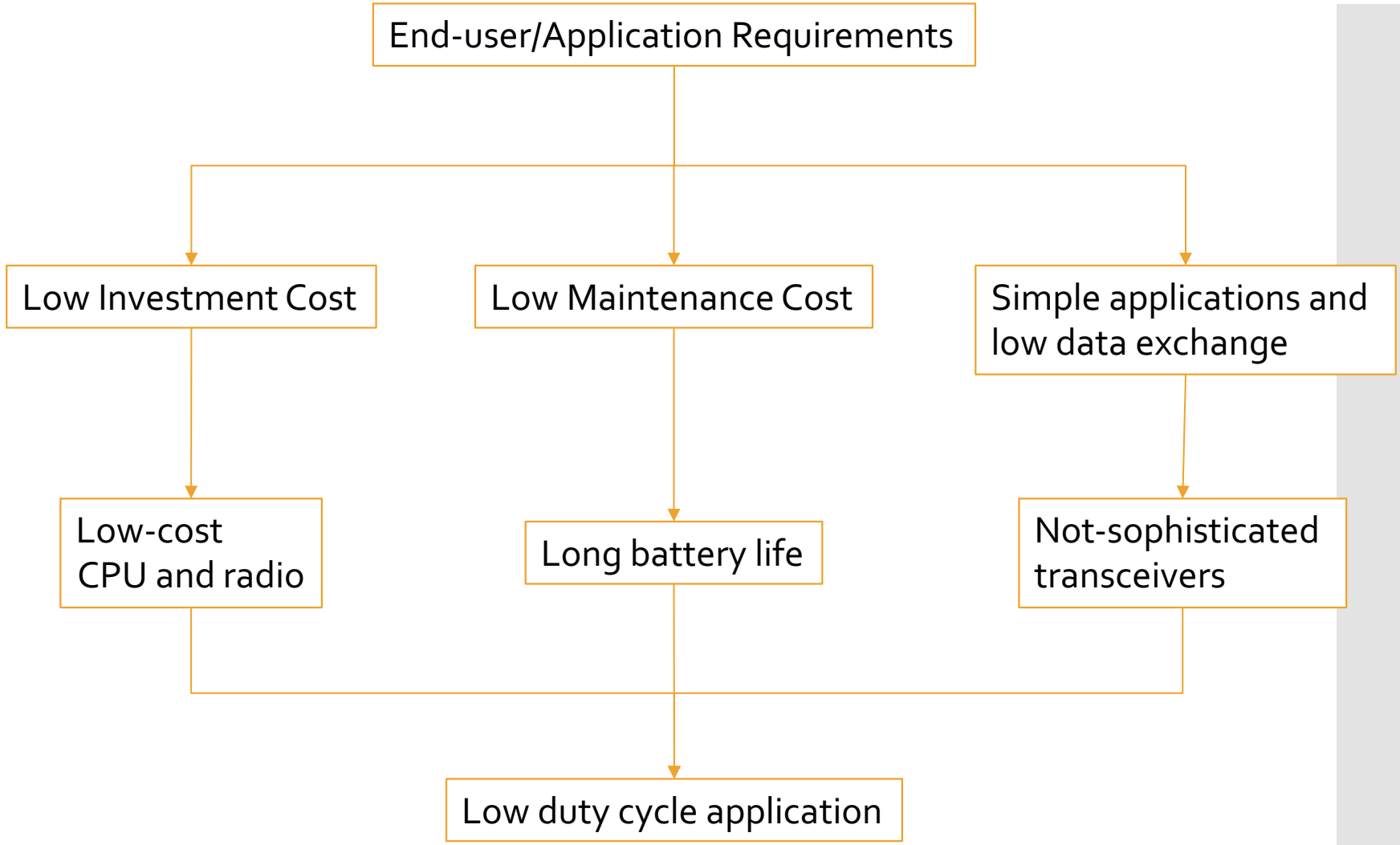
For Wireless Sensor Networks (WSN)



IoT and WSN Applications

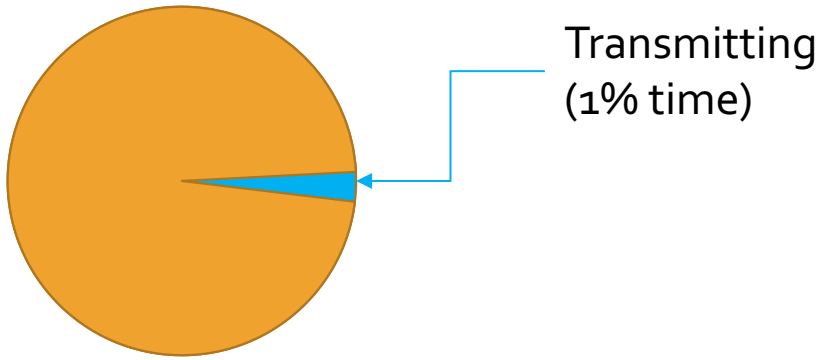
- Industrial and Commerical Control and Monitoring
- Residential Smart Energy
- Home Automation and Networking
  - Consumer Electronics
  - PC Peripherals
  - Home Automation
  - Home Security
  - Personal Healthcare
  - Toys and Games
- Automotive Sensing
- Precision Agriculture
- …

Principles of 802.15.4



Low Cost and Long Battery

- For apps that



- Requirements from hardware/radio
  - Only digital data communications (up to 250 kbps) – modulation
  - Transmitting power of -3 dBm to +20dBm
  - Sensitivity at receiver of -85dBm (for the 2.4GHz band)
- Offering No QoS
  - But timeslots are supported!

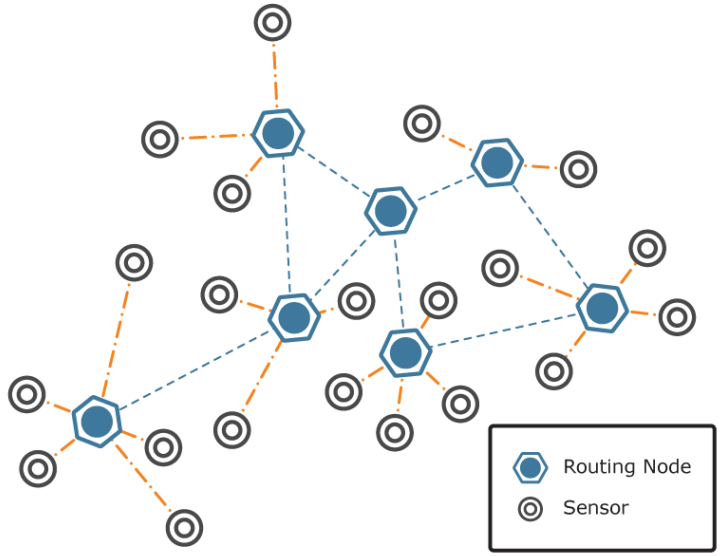
802.15.4 Hardware Device Types

Reduced Function Device (RFD):

- Minimal resources (processing and memory)
- Low cost
- Battery operated
- Examples: light-switches, stick-on sensors, etc.

Fully Function Device (FFD):

- Battery operated/Permanent power
- may forward traffic on behave of others



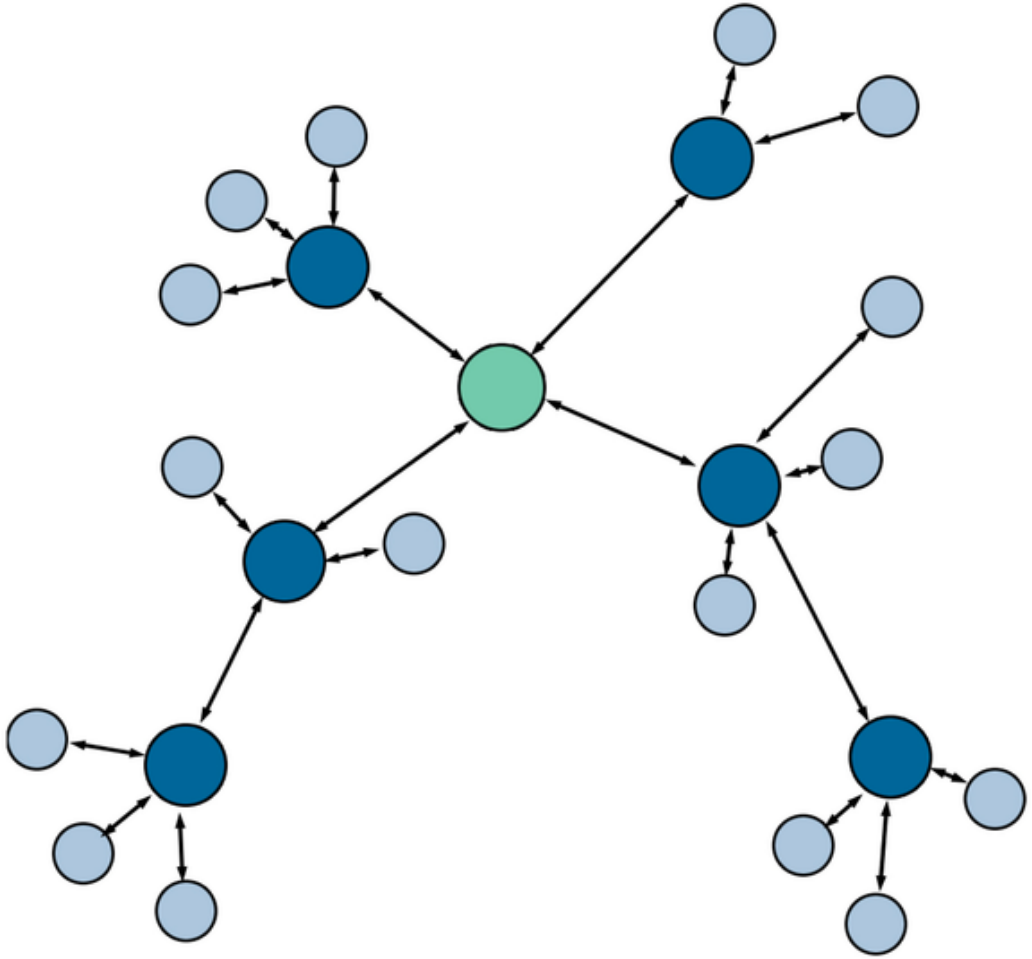
802.15.4 Network Components

- PAN Coordinator (only one in the PAN):
  - Establish the network
  - Defines structure and operating mode
  - Receives requests for joining the PAN network
- Coordinator (o or more):
  - a proxy for the PAN coordinator to other nodes that are no in range
- Device (at least one in the PAN)
  - Implements a small part of the standard

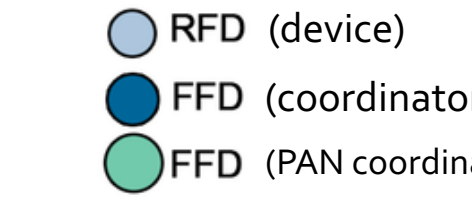
Device Types vs. Network Components

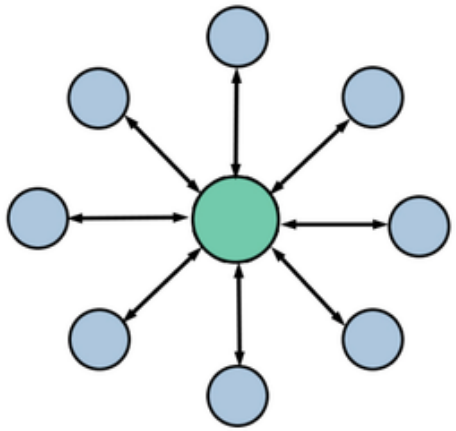
	Reduced Functional Device (RFD)	Fully Functional Device (FFD)
Device or Node	YES	YES
Coordinator		YES
PAN Coordinator		YES

Personal Area Network (PAN) and supported topologies



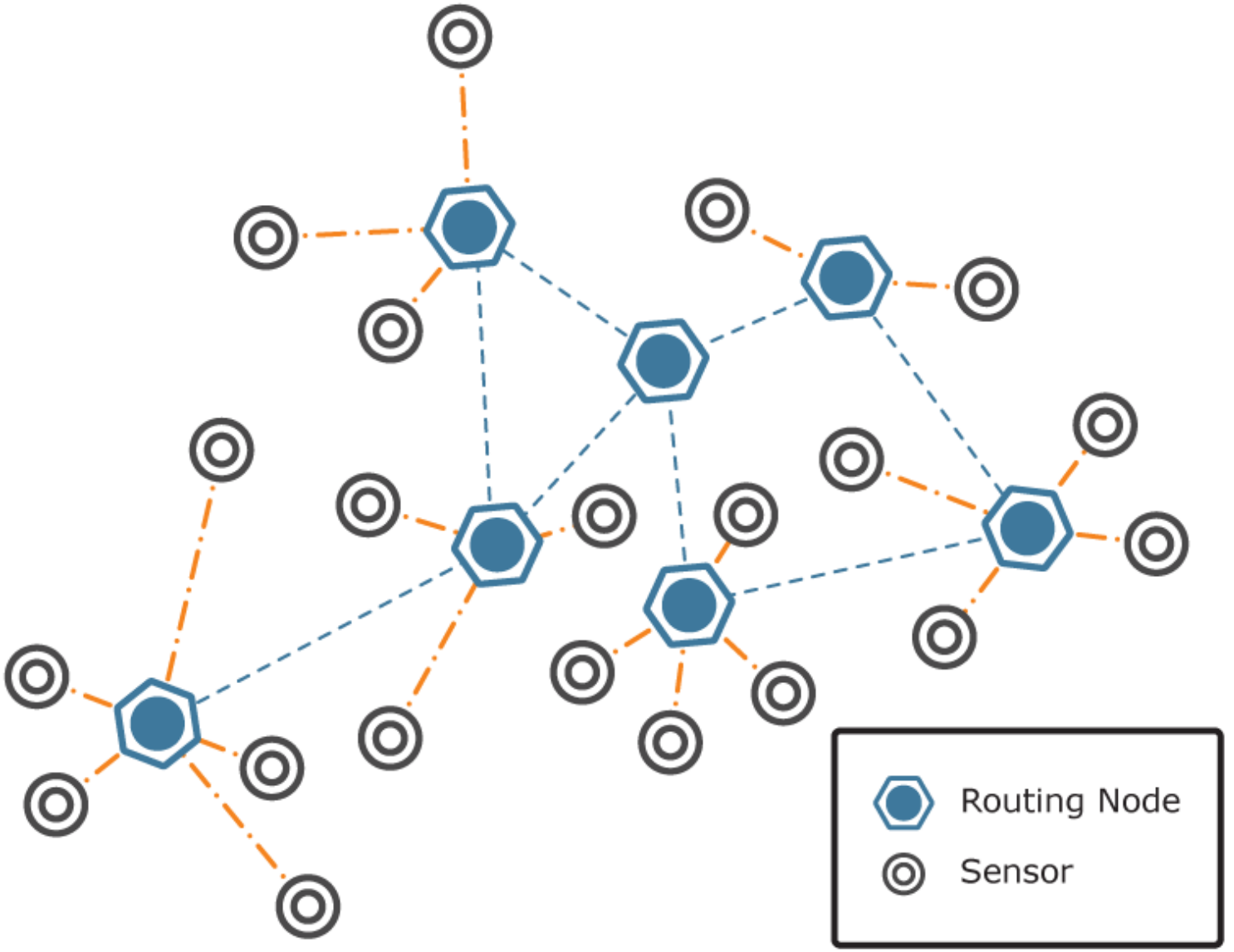
Cluster-Tree topology





Star topology

Support for mesh-networks

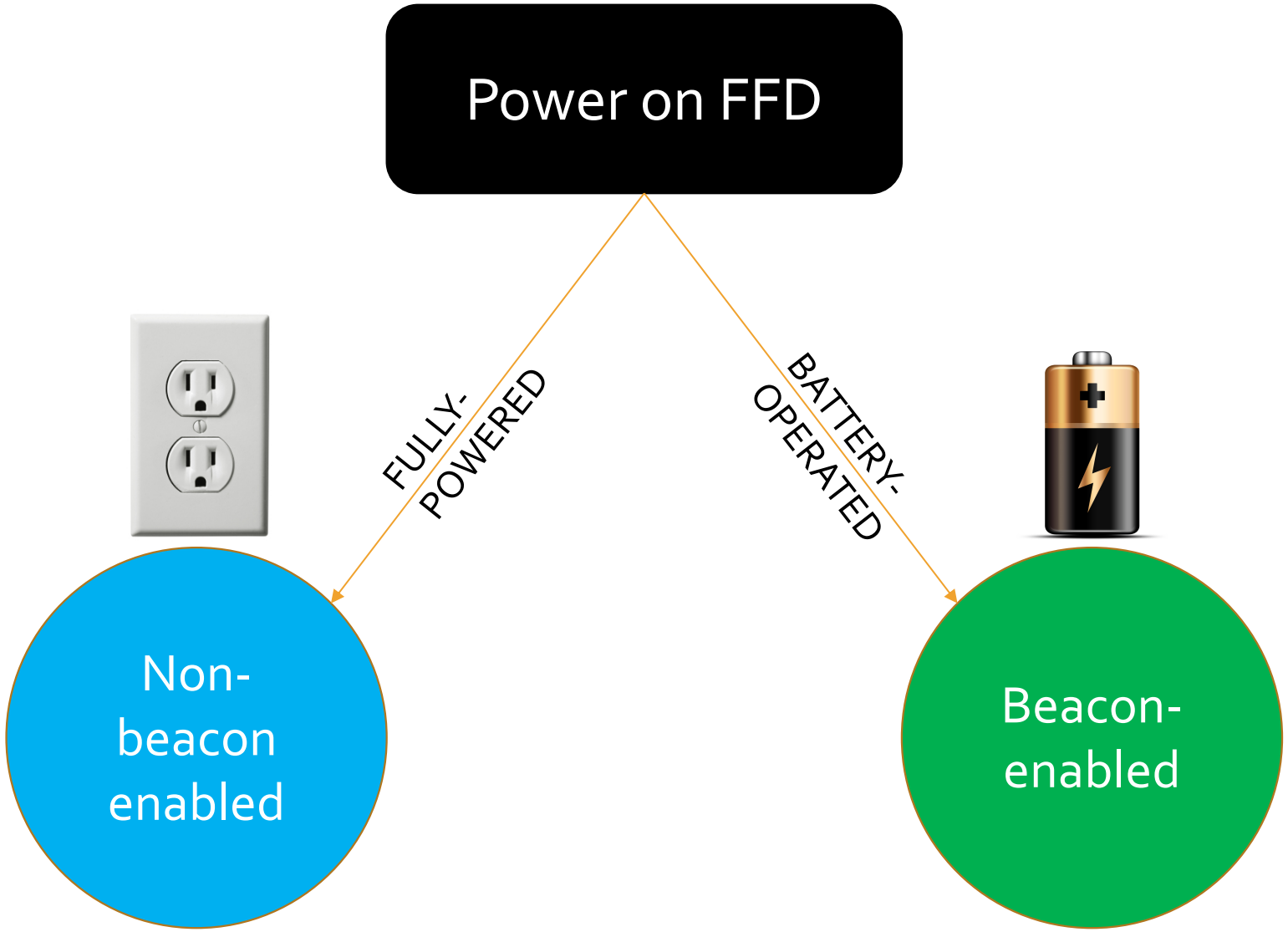


… And Mesh? Allowed, but not standardized (depends on higher layer implementations)

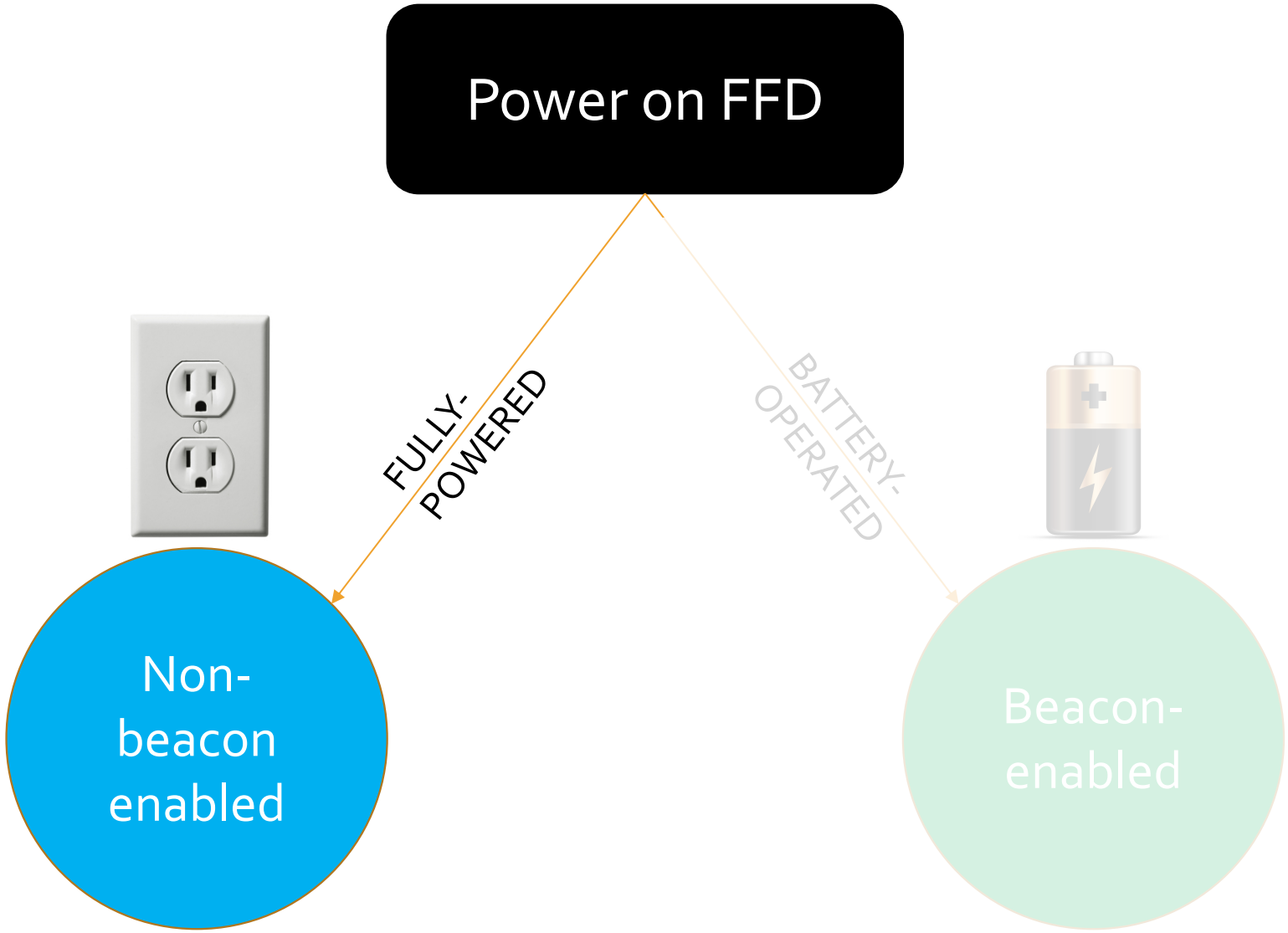
Error Control

- Two mechanisms:
  - Simple full handshake:
    - Acknowledgements on requested non-broadcast data frames
  - Cyclic Redundancy Check (CRC) over the whole frame using 16-bits

Types of Systems

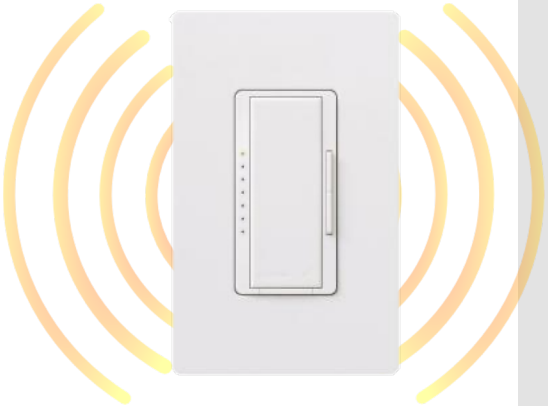


Types of Applications



Communication with a FFD

PAN coordinator
FFD – fully powered



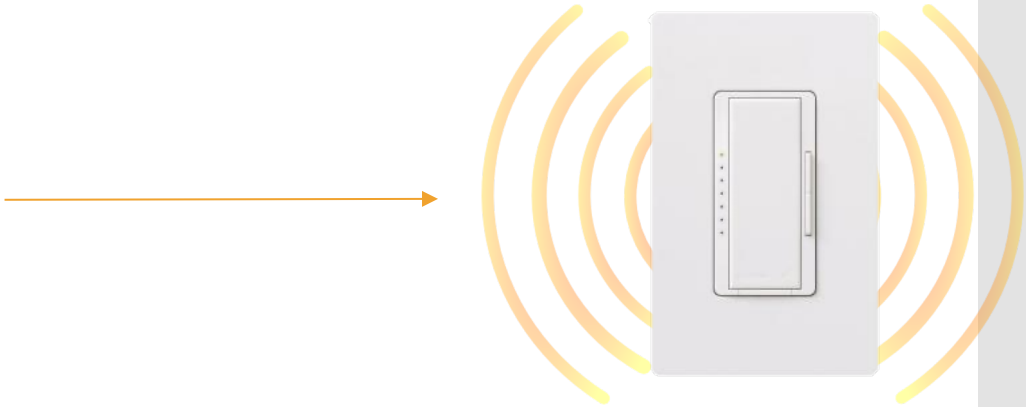
Device
RFD - battery powered

- The Device can send any time to the PAN coordinator, since the PAN Coordinator is always listening (ON).

Sleeping problems…

1% of duty cycle

PAN coordinator
FFD – fully powered



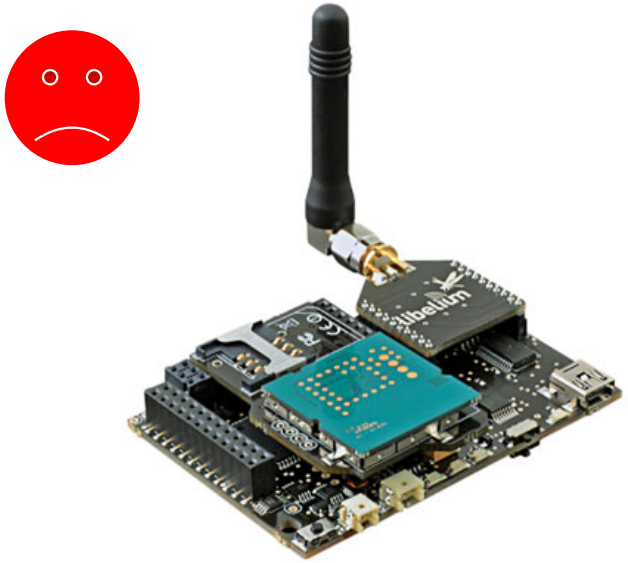
Device
RFD - battery powered

- How to assure that receiver (device) is awake when the transmitter (PAN coordinator) wants to communicate?

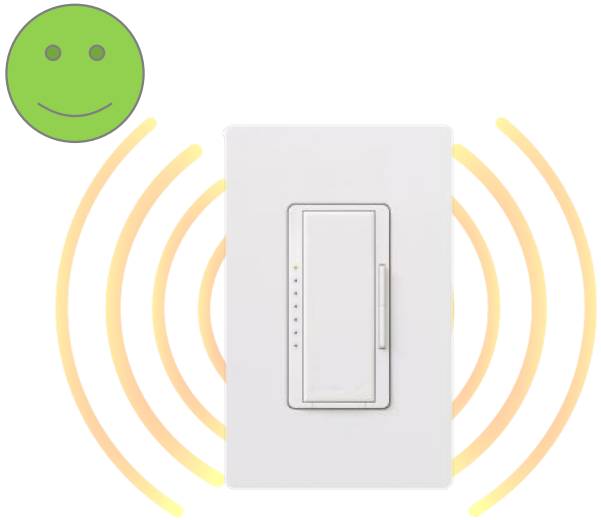
INDIRECT DATA TRANSFER

Sleeping problems…

First solution



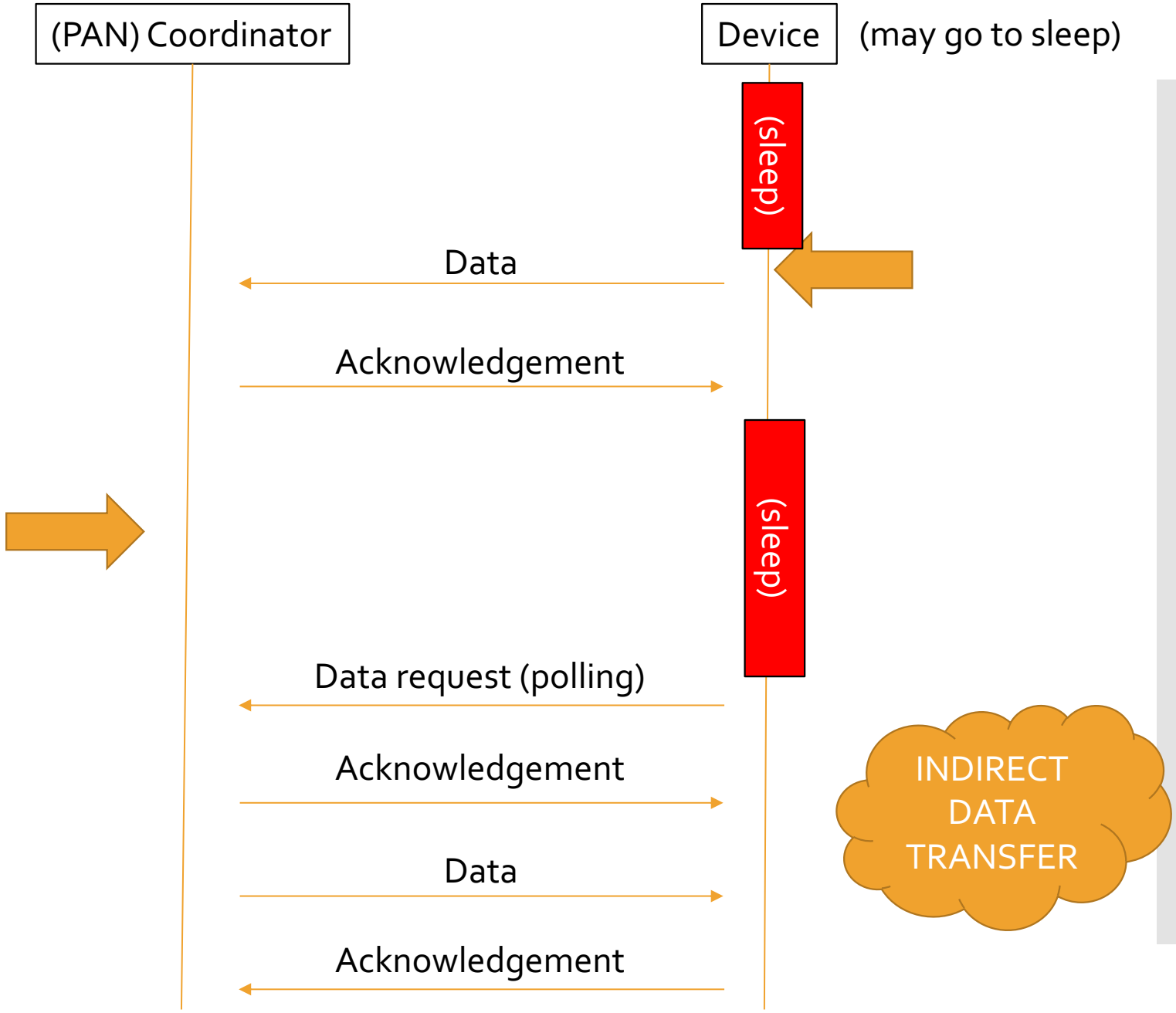
PAN coordinator
FFD – fully powered



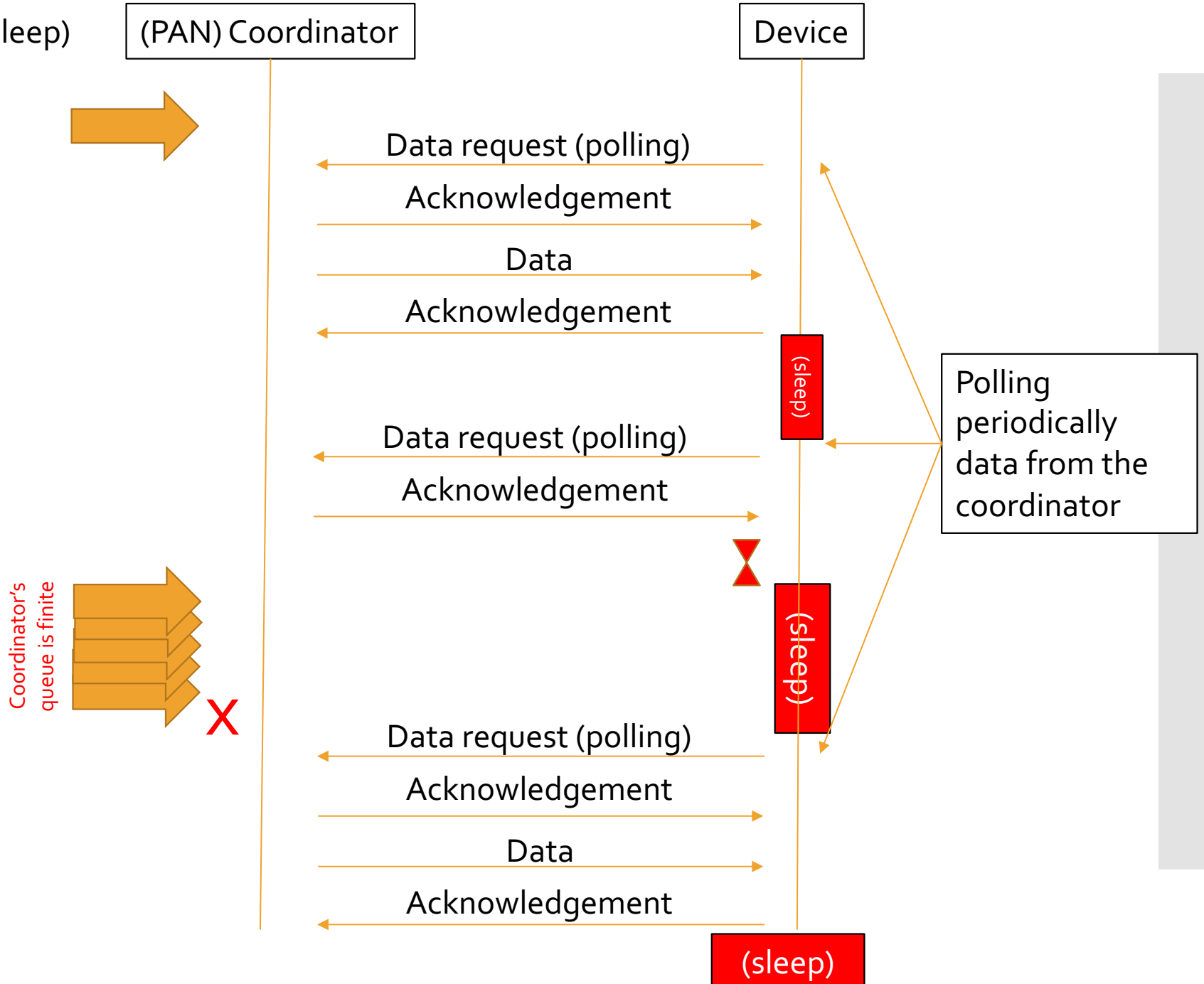
Device
RFD - battery powered

- How to assure that receiver (device) is awake when the transmitter (PAN coordinator) wants to communicate?
  - Device polls data from the PAN coordinator (when the device is awake)

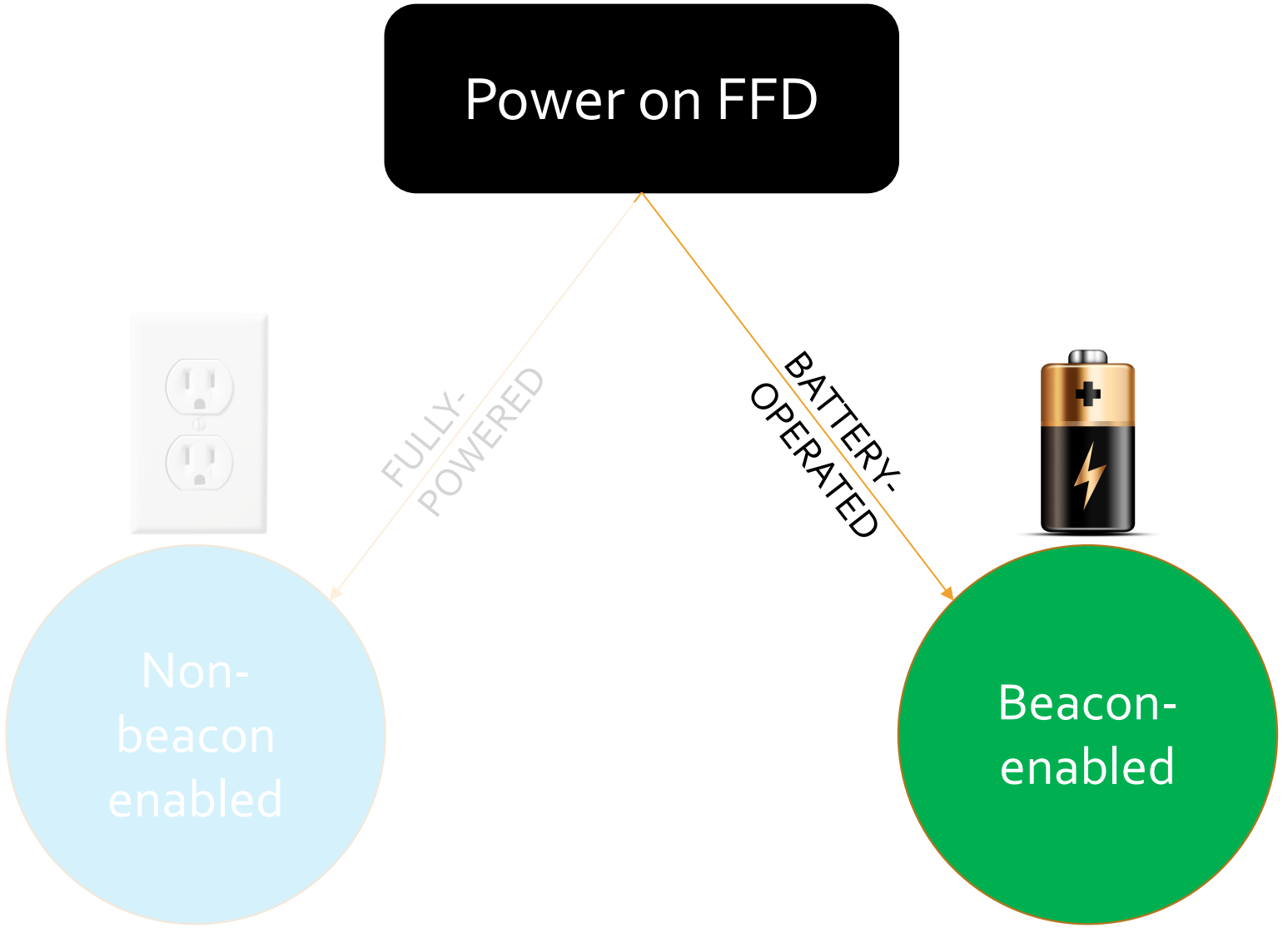
Data TRx – Non-Beacon



Non-Beacon Drawbacks

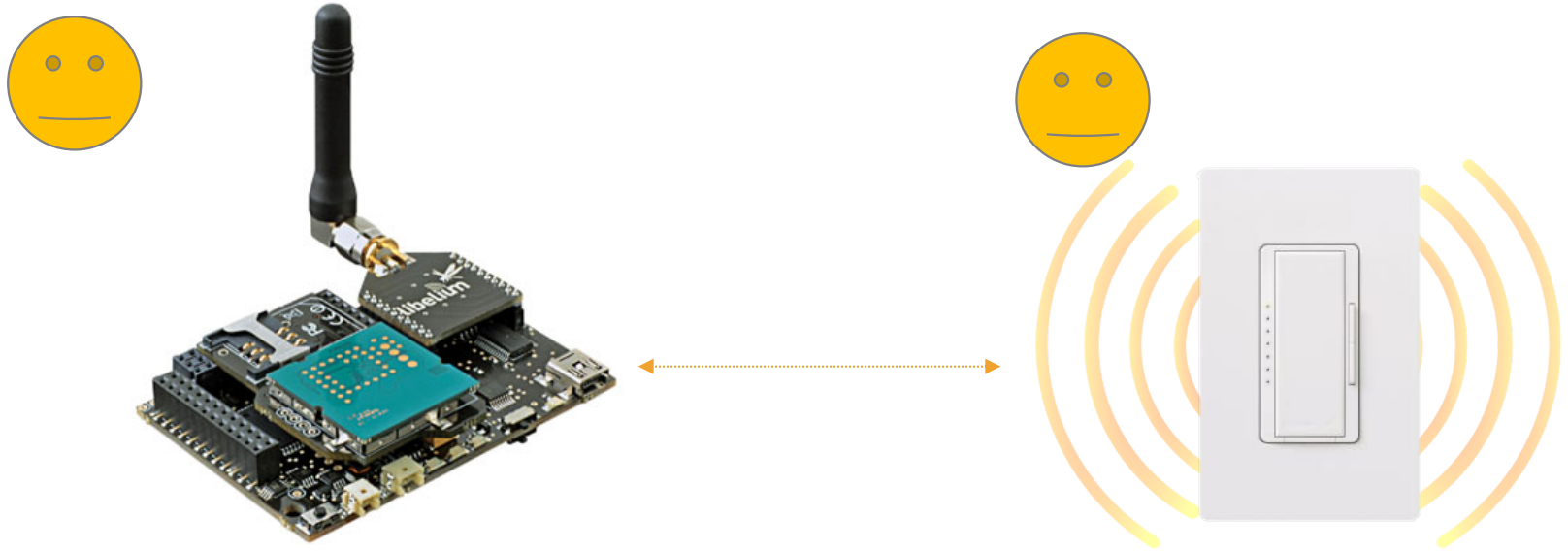


Types of Applications



Sleeping problems…

Second solution



PAN coordinator
FFD – battery powered

Device
RFD - battery powered

- How to assure that both (PAN) Coordinator and Device are awake when either of them wants to communicate?
  - How does the Device know when to transmit?
  - If the Device knows when to transmit, then the Device can poll data from the PAN coordinator (same as before)

802.15.4 SUPERFRAME

Contains info about:

- How often a beacon is TX
- Awake/sleep period
- Reserved timeslots
- Whether there is data pending for a RFD

Cases for Beacon-enabled LR-WPAN

Data exchange frequency

Very rarely

Rarely

Periodically

Non-synchronized

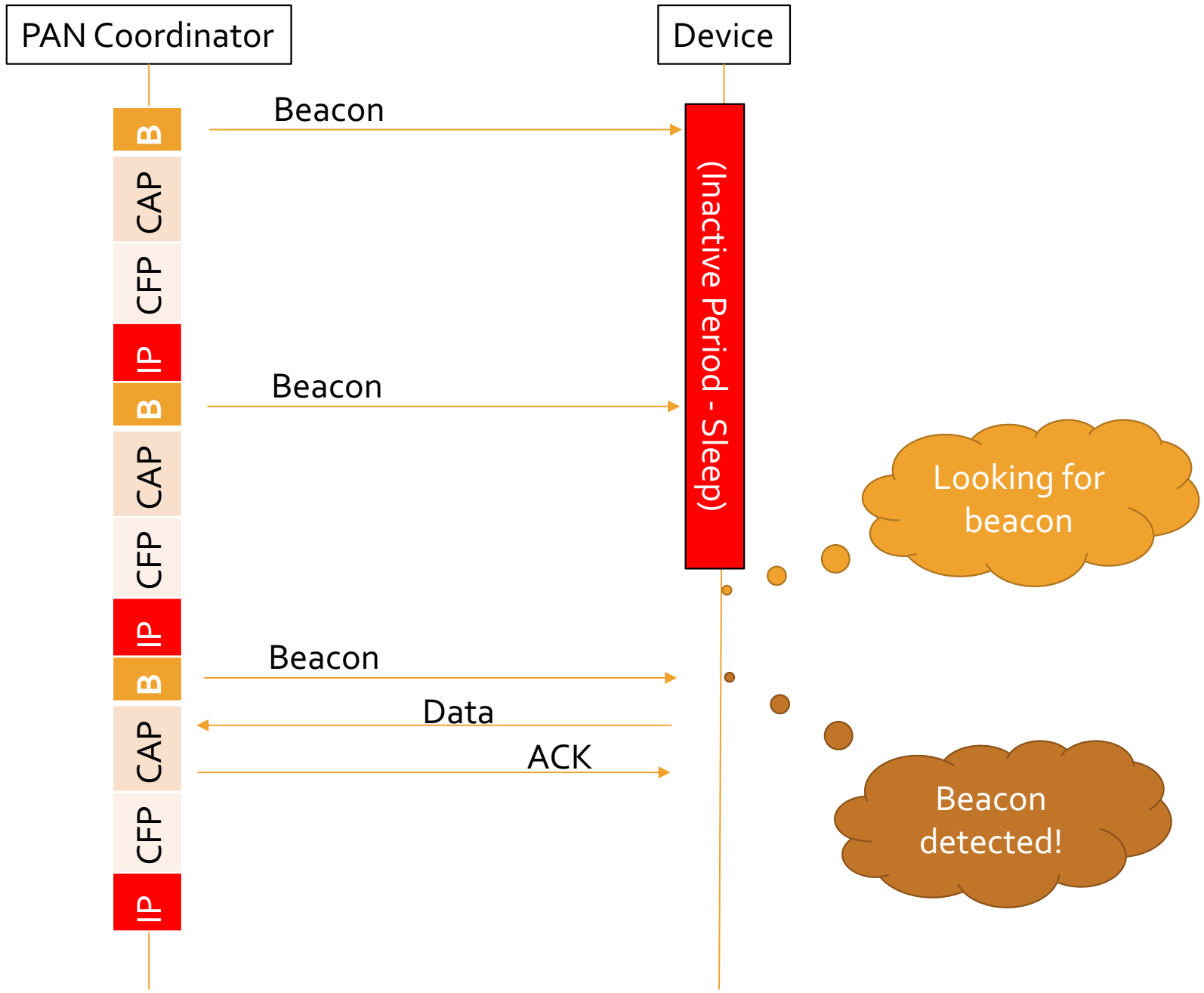
Synchronized

Guaranteed Time Slots

Beacon-enabled

Non-synchronized

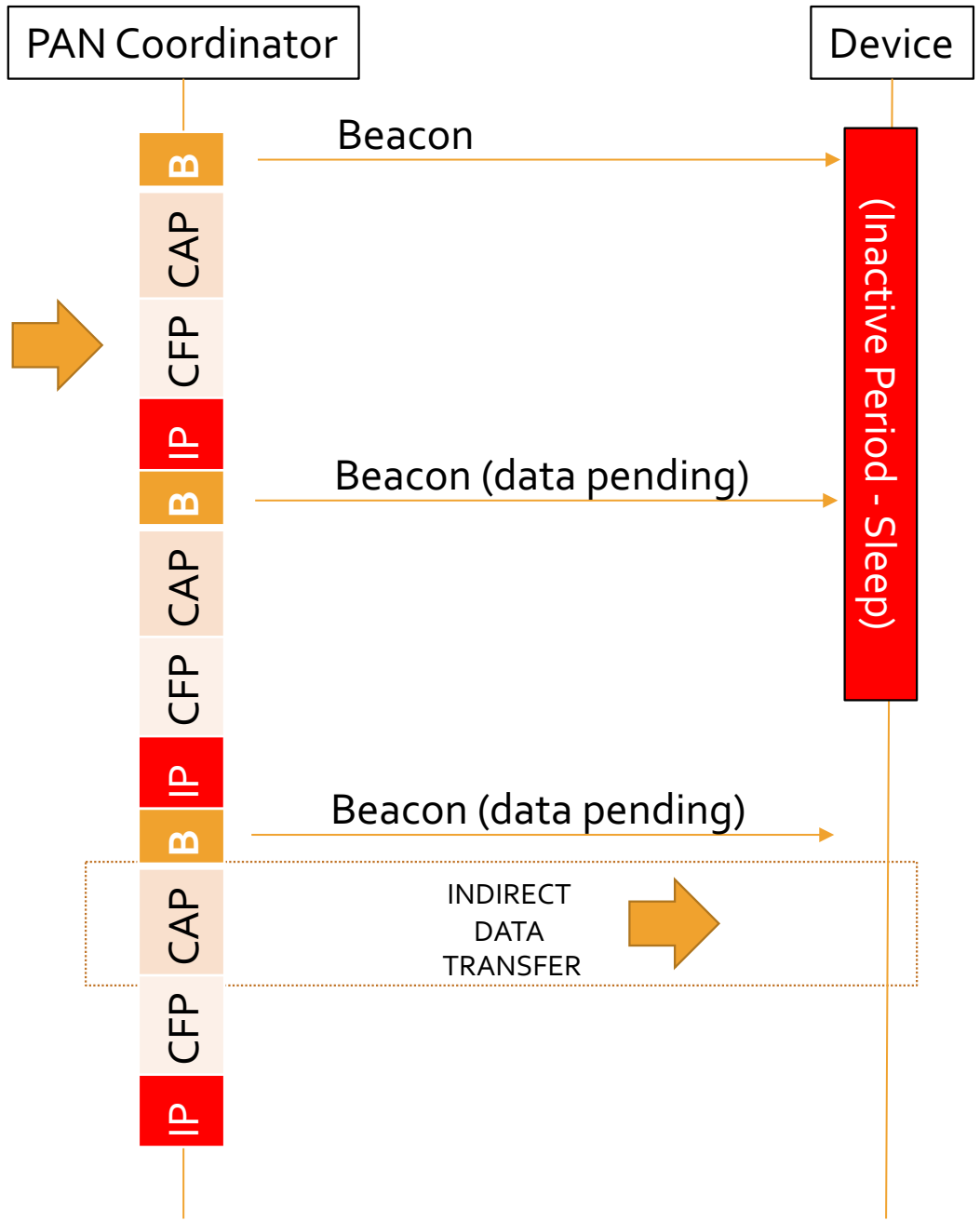
Device => Coord



Beacon-enabled

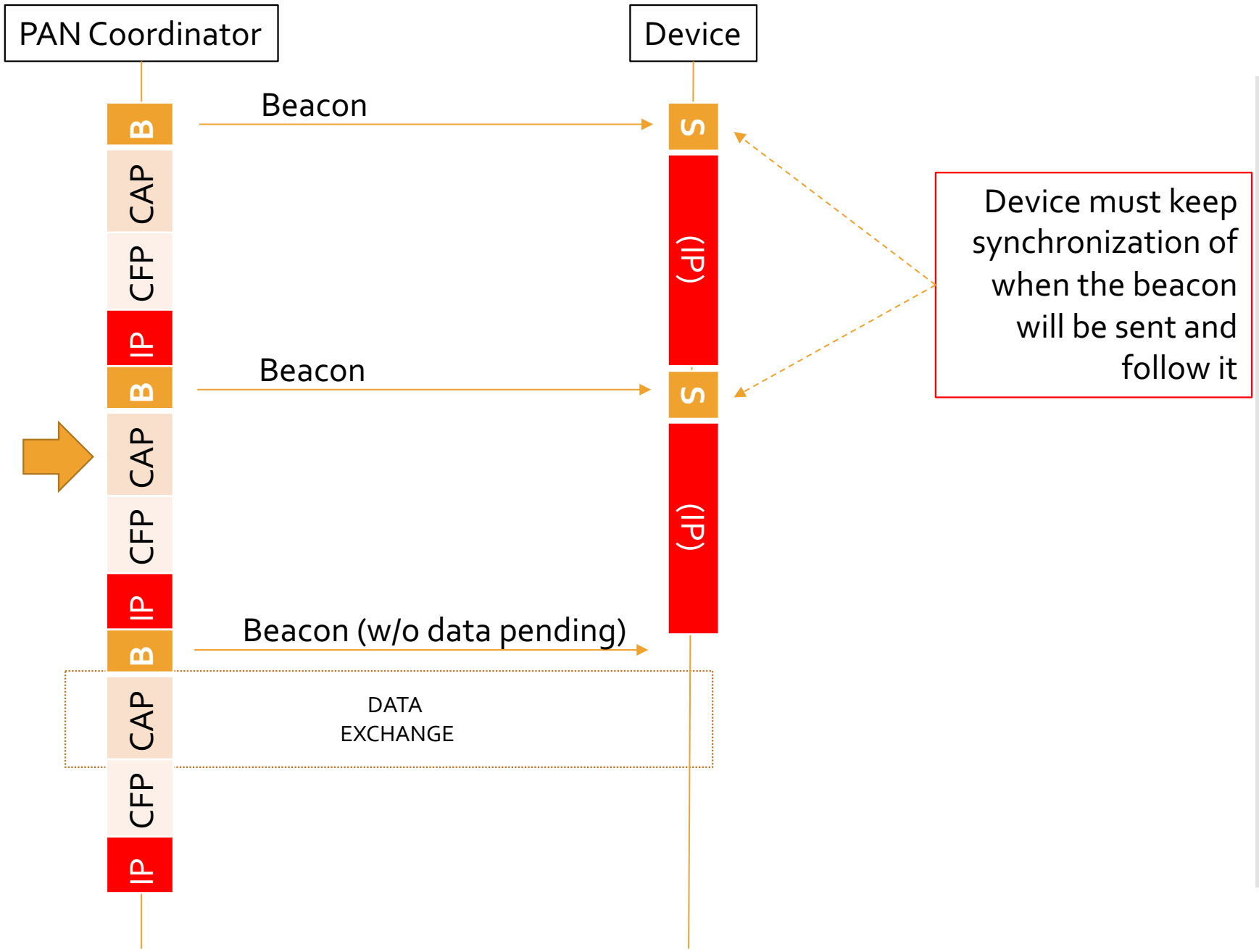
Non-synchronized

Coord => Device



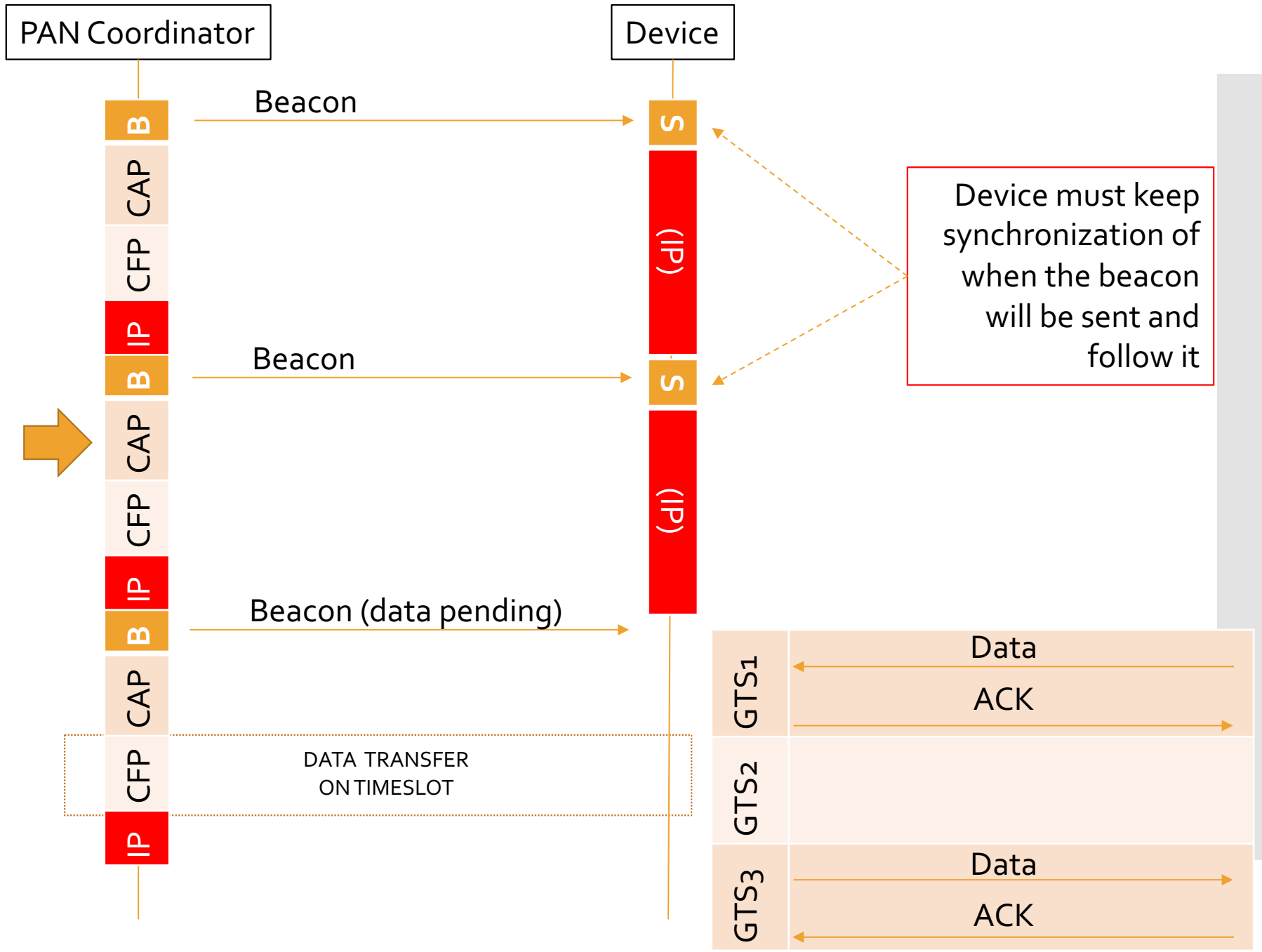
Beacon-enabled

Synchronized



Beacon-enabled

Guaranteed Time-Slots (GTS)



Data TRx Summary

	Non-beacon	Beacon w/o synchronization	Beacon with GTS
From device to PAN coordinator	Send in CAP	Send in CAP after beacon	Send in GTS
From PAN coordinator to device	Device polls in CAP, coordinator sends in CAP	Coordinator notifies in beacon, device polls in CAP, coordinator sends in CAP	Send in GTS

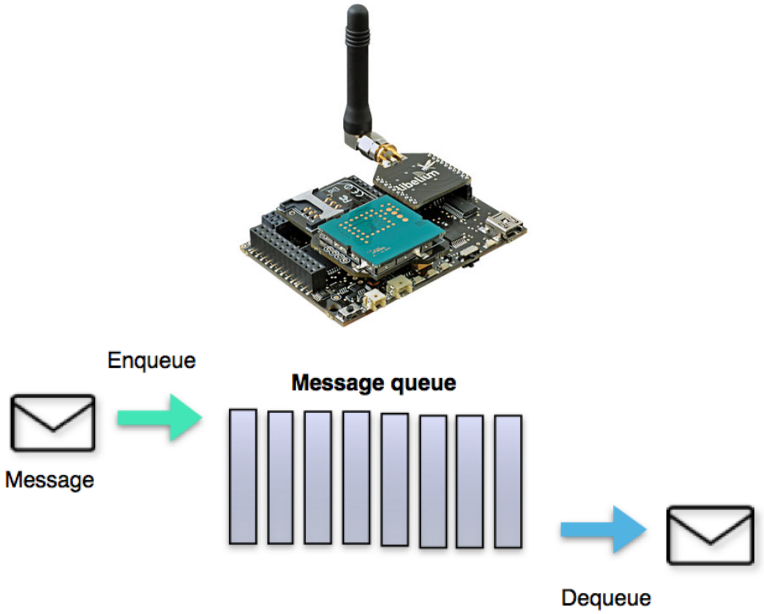
Efficiency

Non-beacon enabled

Polling frequency

MEMORY @ FFD

Coordinators must store frames that will be eventually pulled by devices



- Polling frequency must be fast-enough to prevent frames drops in Coordinators queue

POWER @ RFD

Devices must guess when data is pending at the coordinators

- Polling frequency of RFD must be slow-enough as to not waste energy when there are no frames pending at the coordinator

Beacon-Enabled

MEMORY @ FFD

Coordinators must store frames that will be eventually pulled by devices

- Polling frequency must be fast-enough to prevent frames drops in Coordinators queue

POWER @ FFD

Coordinator must always generate beacons, even if there is no data exchange

- If synchronized or GTS, RFD must keep track of beacons even if there is no data exchange

MAC Services

What can 802.15.4 provide to higher-layers?

Scanning Radio Channels

ENERGY SCAN DETECTION

By the PAN Coordinator to find an "empty" channel for starting a PAN.

Measures energy levels on channels

ACTIVE CHANNEL SCAN (poll)

By devices to find available coordinators to join in beacon and non-beacon PANs

Sends a beacon request message

PASIVE CHANNEL SCAN

By devices to find available coordinators to join in beacon PANs

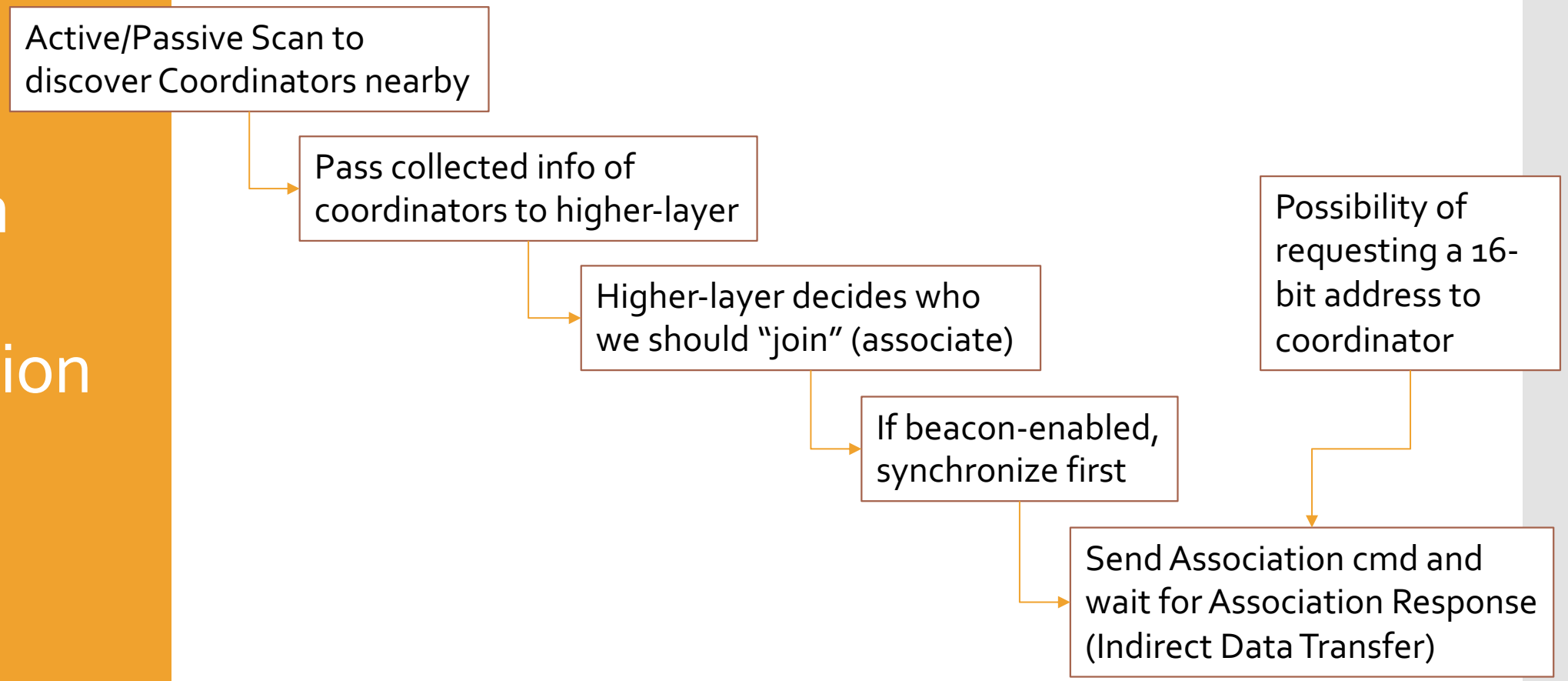
Waits for a beacon message

ORPHAN CHANNEL SCAN

By devices that lost the coordinator (over specific logical channels)

?

Association and Disassociation Control



Beacon Management

- On a FFD allows from higher layers:
  - Configuring the device as a coordinator or PAN coordinator
  - Selecting logical channel
  - Establishing beacon periodicity
  - Setting up superframe characteristics
- On a devices receiving beacons:
  - Notify higher layers

Addressing

Layer			
IPv6	Network address	128-bits written in hex format in eight groups of 16-bits separated by ::	FE80::ABCD:1234:E6A8
802.15.4	Physical (MAC) address	Long: 64-bits (8 bytes) written in hex format, each byte separated by -	02-AB-4F-C9-00-AA-DE-AD
		Short: 16-bits (2 bytes) written in hex format, each byte separated by -	03-45
802.1 (Ethernet)	Physical (MAC) address	48-bits written in hex format, each byte seprated by :	ac:de:48:00:11:22

Bibliography

- IEEE 802.15.4-2003
- IEEE 802.15.4-2006
- IEEE 802.15.4-2007
- IEEE 802.15.4-2009
- IEEE 802.15.4-2011