BATTERY ENERGY ESTIMATION of DURATION

Q [battery life] = 2 Ah = 2000 mAh

$V = 3{,}3\,V$ NB: J = W·s

P_{OWER} 1 Watt

Energy [J] = Voltage · Intensity

$\Rightarrow E = 3{,}3 \cdot 2 \cdot 3600 = 23760$ J

$$\text{POWER} = \frac{\text{Energy}}{\text{time}}$$

~~= 23760~~

$$\Rightarrow \text{time} = \frac{\text{Energy}}{\text{Power}} = \frac{23760\ W \cdot s}{1\ W}$$

$$= \frac{23760}{1} = 23760\ s$$

time [hours] = 6.6

ENERGY DEFINITION:

Ability of a BODY | ENTITY to do work.

POWER DEFINITION

Ability of a BODY | ENTITY to do work over time.

$$\text{POWER} = \frac{\text{Work}}{\text{time}} = \frac{\text{ENERGY}}{\text{time}}$$

NB: The more ENERGY an entity has, the MORE work it can do.

LIGHT:

~~[illegible]~~ Electromagnetic radiation lying within a certain portion of the electromagnetic spectrum.

↳ VISIBLE LIGHT: 400 – 700 nm.

MOBILE CALLS & ARCHITECTURE

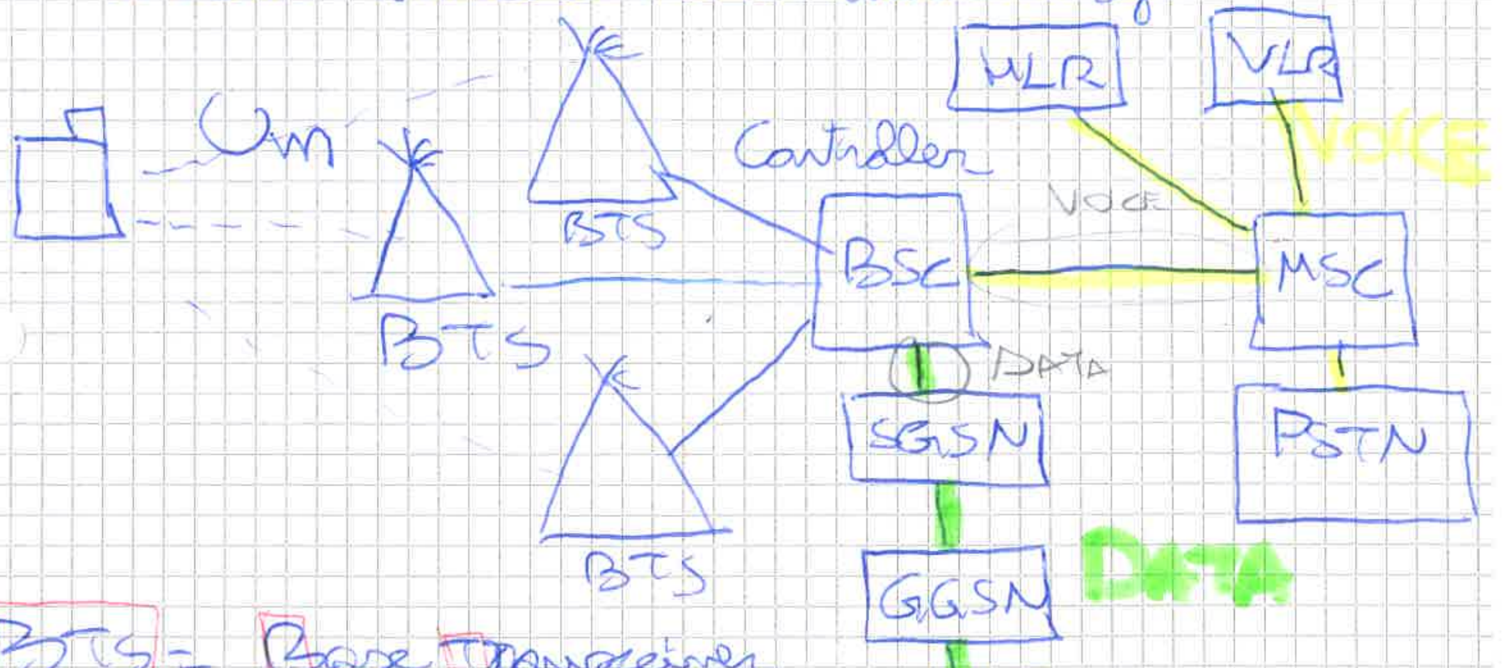
PAGING = Transmission of VOICE / DATA to a destination whose location is not known

IN-BAND SIGNALLING:
Usage of DATA CHANNEL for the transmission of ~~exchanging data~~ CONTROL DATA

OUT-OF-BAND SIGNALLING:
~~......~~ Usage of AD-HOC channels for ~~......~~ the transmission of CONTROL DATA (not transmitted in data channels)
⇒ Approach followed in GSM +

GPRS ARCHITECTURE

Circuit-switched Packet switching



BTS = Base Transceiver Station

BSC = Base Station Controller

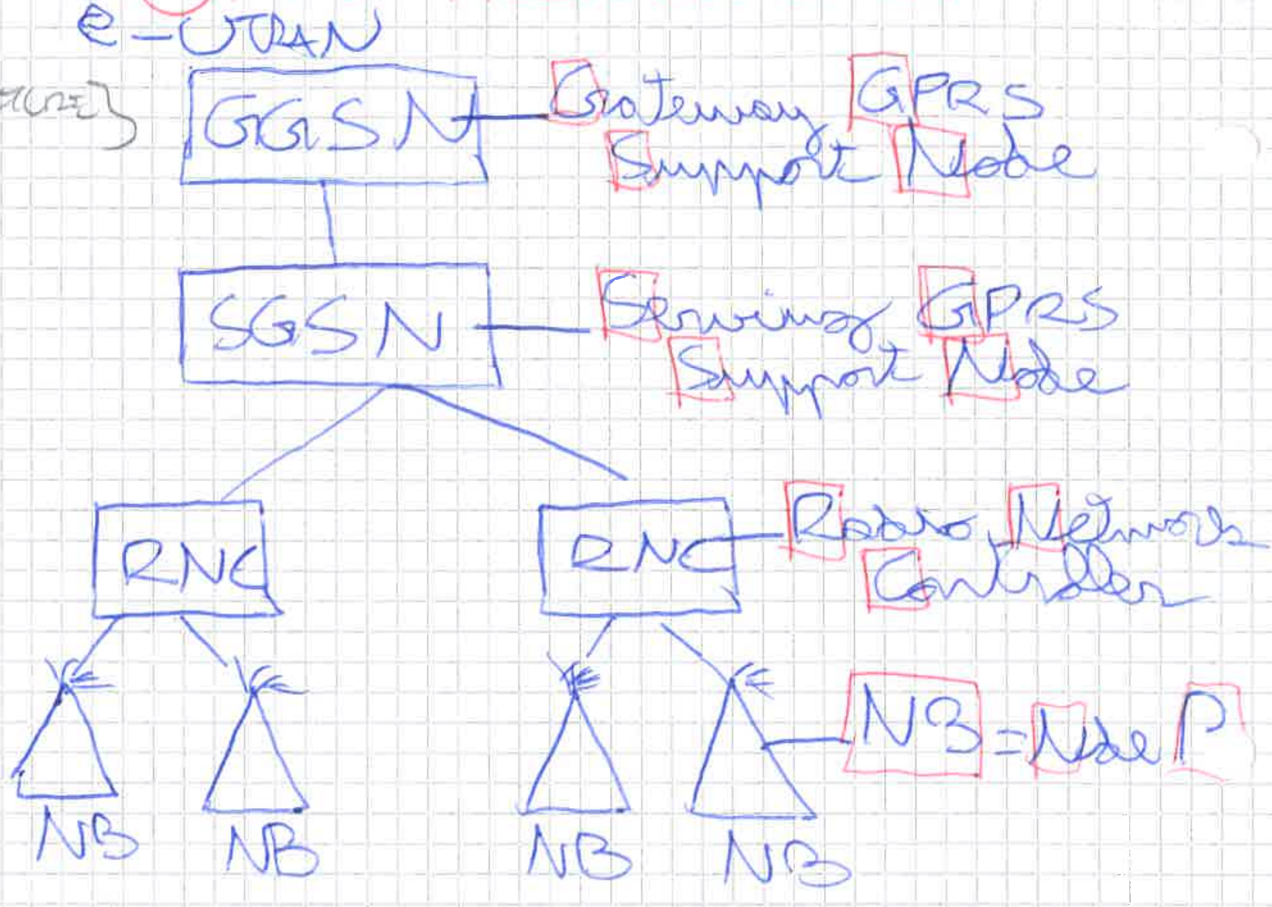
MSC = Mobile Switching Centre

GGSN = Gateway GPRS Support Node

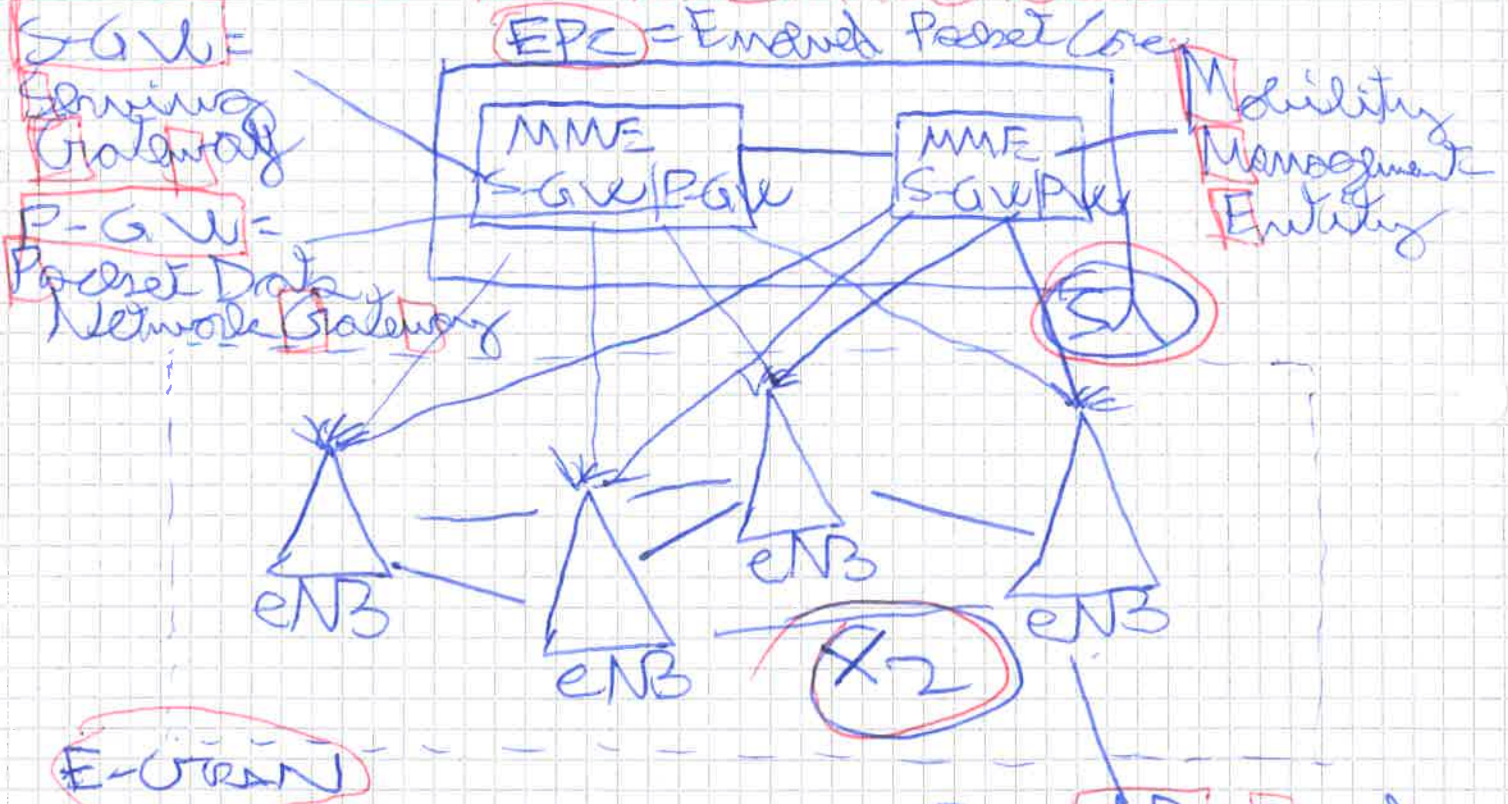
SGSN = Serving GPRS Support Node

UMTS ARCHITECTURE

[UNIFIED PACKET ARCHITECTURE]



LTE ARCHITECTURE

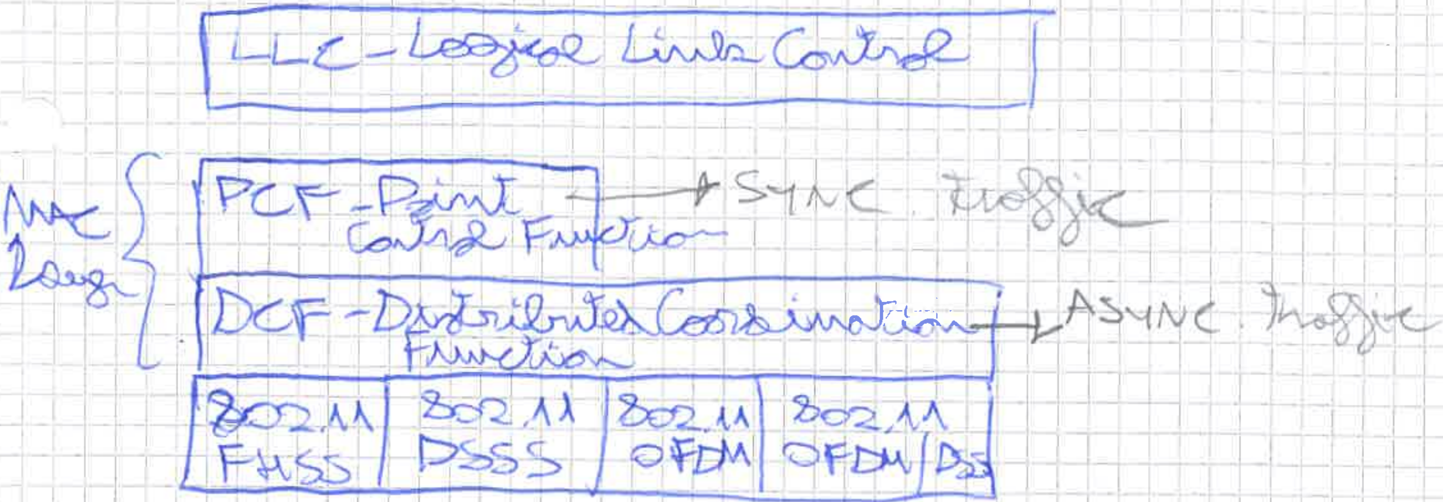


eNB = Evolved Node B

DIFFUSION [SOFT LIGHT]

Spreading of LIGHT over the free space (= NON DIRECT LIGHT DIFFUSION) caused by the light bouncing off many angles of a SURFACE or when it travels through a substance that changes its angles [WRAPS AROUND OBJS]

WIFI'S MAIN FEATURES.



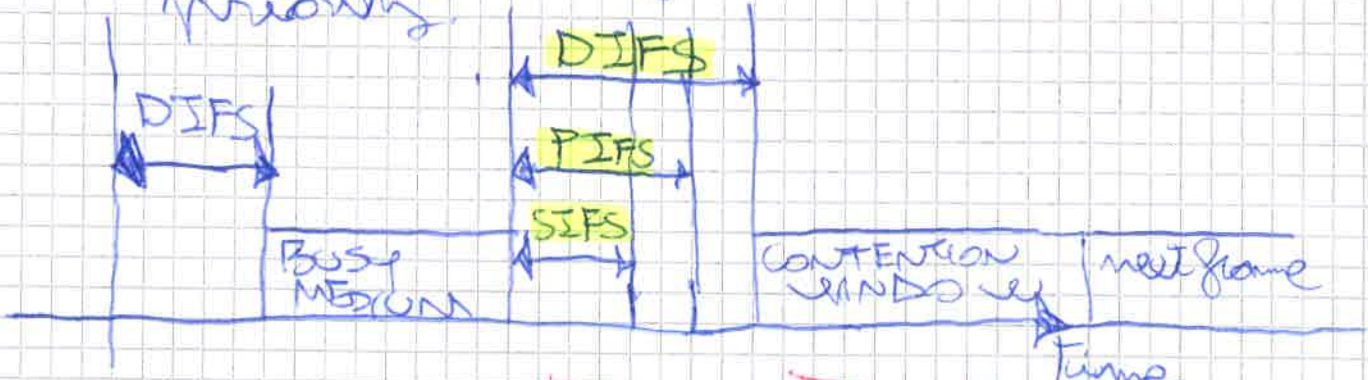
DCF - Distributed Coordination Function:

~~Makes use of CSMA/CA to ensure contention free data transm the delivery of all~~

It uses a CONTENTION ALGORITHM to provide access to the channel for all the TRAFFIC!

=> Based of CSMA/CA & notably used for handling asynchronous data traffic.

=> Uses different IFS for handling different types of TRAFFIC & their priority.



- ~~DIFS~~ SIFS = Short InterFrame Spacing
  Used for immediate traffic (ACKs) & for ~~sync~~ delay-sensitive traffic
  = 28 μs → FHSS, 10 μs → DSSS
- PIFS : Point Coordination Function InterFrame Spacing
  Used in PCF for handling "POLLS".
  = SIFS + 1 slot time

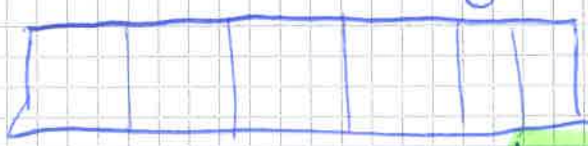
• DIFS: Distributed Coordination Function InterFrame Spacing

= SIFS + 2 slots

Used as min. value for ASYNC. frames contending for access.

CSMA/CA introduces the HIDDEN TERMINAL & EXPOSED TERMINAL problems ⇒ Solved by using RTS/CTS

cause COLLISIONS



leave delays (wait for exposed node to finish transmitting)

NAV (AP/coordinator) ~~slots~~

⇒ RTS (Request-to-Send) & CTS (Clear-to-Send) Used to reserve the BANDWIDTH

IDEA: Multiple stations send a RTS (collisions can occur during this period) & ⇒ AP sets the RTS in the NAV for station avoiding collisions from occurring further.

~~⇒ CTS is send~~

used for SYNCING

RTS: Sent from source to destination for alerting it about its intention to transmit data to it.

CTS: Sent from destination to source to "grant" permission to send data.

PCF - Point Coordination Function:

Works according to a POLLING mechanism

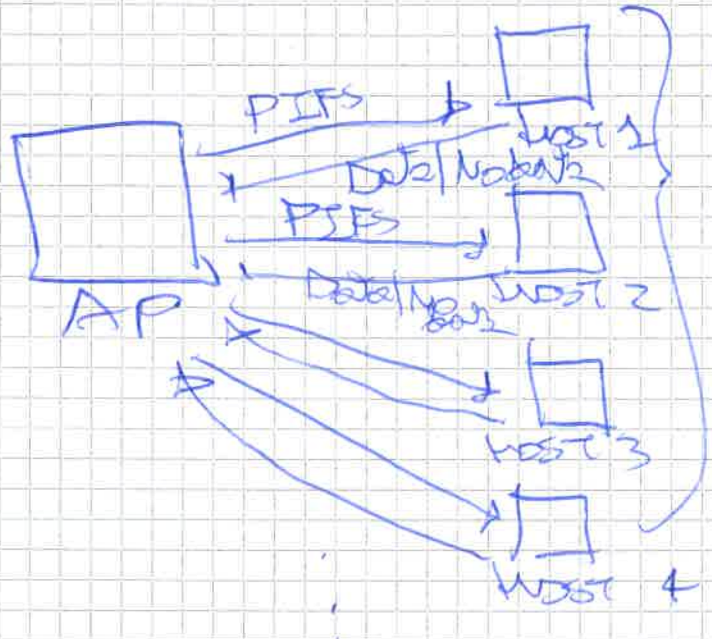
⇒ To ensure fairness, the access time is divided into SUPER-FRAMES comprising:

- CONTENTION-FREE PERIODS
- CONTENTION PERIODS

CONT.-FREE	CONT.-PERIOD

AP - COORDINATOR: Performs POLLING by ~~making~~ allocating a PIFS (smaller than DIFS)

⇒ BLOCK of all the ASYNC. TRAFFIC



Stations with time-sensitive traffic transmit by POLLING (round-robin data sending)
⇒ CONTENTION-FREE PERIOD

ASYNC. TRAFFIC!

After performing the POLLING phase, the AP remains idle for a while, allowing ASYC. traffic to be transmitted. "WiFi available"

SYNCHRONIZATION ⇒

Performed using BEACON FRAMES, which are transmitted periodically to announce a WLAN network & to synchronise the members of the service set to a common clock.

MAIN IoT ATTACKS

INFORMATION LAYER

- JAMMING: Injecting artificial bogus data into the network for the purpose of saturating its capacity
- SPOOFING: Situation in which a person ~~or~~ or a program successfully masquerades as another by falsifying data, to gain an illegitimate advantage.
- EAVESDROPPING: Usage of technique to "overhear" the data being transmitted over a channel.

CONNECTIVITY LAYER

- SYBIL: An attacker creates a huge amount of "BOTS" & uses them to gain a large influence / PSEUDOSPOOFING
  Ex: To "hijack" a P2P network.
- SINKHOLE: Attracts surrounding nodes with misleading route info & performs data forging or ~~deleting~~ forwarding of a part of this data only

(Bugs nel Terreno)

- DDOS: ~~Creation~~ Usage of massive distributed resources for the purpose of ~~[illegible]~~ making a system/infrastructure unavailable

APPLICATION LAYER

- SPEARPHISHING: "pesca ad amo" Popular email scam aimed at making a victim transfer large sums of money / provide their personal information

- **SNIFFING**: Theft or interception of data by an unauthorized party
- **DoS ATTACK**: Same as attack in the CONNECTIVITY LAYER, but at the application layer.

MFSK - MULTIPLE FREQUENCY SHIFT KEYING

More than two frequencies are used for modulating THE SIGNAL.

$$s_i(t) = A \cdot \cos(2\pi f_i t) \qquad 1 \le i \le M$$

$$f_i = f_c + (2i - 1 - M) \cdot f_d$$

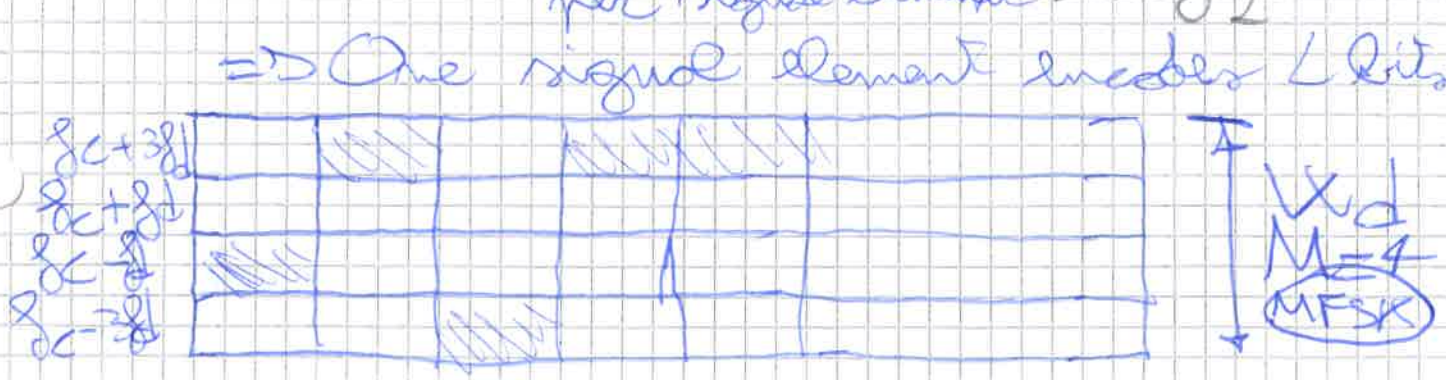
f_c = carrier frequency
f_d = difference frequency
M = # different signal elements $= 2^L$
L = # BITS per signal element

=> In order to match the DATA RATE of the input bit stream, each output signal is held for:

$$\boxed{T_s = L \cdot T_b} \text{ seconds}$$

T_b → Time to transmit a bit
L → # BITS per signal element $= \log_2 M$

=> One signal element encodes L bits.



TOTAL BANDWIDTH REQUIRED

$$\boxed{W_d = 2 \cdot M \cdot f_d}$$

Minimum frequency separation required:

$$2 \cdot f_d = \frac{1}{T_s}$$ [to have ORTHOGONAL FREQUENCY SEPARATION & avoiding interference]

$$\Rightarrow f_d = \frac{1}{2 \cdot T_s}$$

$$\Rightarrow W_d = 2 \cdot M \cdot f_d = \frac{2 \cdot M}{2 \cdot T_s}$$

$$W_d = \frac{M}{T_s}$$

W_d

TIME-VARIANT SYSTEM: [Ex: Shadowing / Amplitude Modulation]

System whose OUTPUT characteristics depend explicitly upon time. Namely, a system in which certain quantities governing the syst. behaviour change with time

$\Rightarrow$ [System will respond to the same input differently at different times]

SPACE-VARIANT SYSTEM: [Ex: Fading]

System whose OUTPUT characteristics depend explicitly upon space. Namely, a system where system whose quantities governing the system behaviour change with time

TIME-INVARIANT SYSTEM: [Ex: Speed of light]

System whose output is NOT a direct function of time.

DIGITAL MODULATION - DATA RATES

T_S = SYMBOL RATE

TECHNIQUE NAME	DATA [Bit] RATE	SYMBOL [Baud] RATE
MPAM	$R = \frac{K}{T_S} = \frac{\log_2 M}{T_S}$	$D = \frac{1}{T_S}$
MPSK	$R = \frac{1}{T_b}$	$D = \frac{R}{K} = \frac{R}{\log_2 M}$
MFSK	~~T_S~~ ~~R~~ $W_d = 2 M \delta = \frac{M}{T_S}$	$L = \log_2 M$ $T_S = L \cdot T_b$ $D = \frac{1}{T_S}$
QAM		$D = \frac{R}{K} = \frac{R}{\log_2 M}$

M = # MODULATION LEVELS

T_S = Symbol rate

ENERGY TYPES

THERMAL ENERGY | THERMAL NOISE | HEAT ENERGY.

~~Energy contained in a system~~

Energy coming from a Rise in Temperature causing atoms & molecules to move faster and collide with each other.

POTENTIAL ENERGY:

Energy held by an object because of its position relative to other objects.

KINETIC ENERGY | MECHANICAL ENERGY

Energy possessed by an object because of its MOTION [Energy possessed by an object in movement]

NUCLEAR ENERGY

$$\boxed{E = m.c^2}$$

Energy trapped inside an ATOM. It can be produced either by FUSION (combining atoms) or FISSION (splitting of atoms).

GRAVITATIONAL ENERGY:

Energy held by an object lying in a gravitational field

ELECTRICAL ENERGY:

Energy derived from electric potential energy.

DOPPLER EFFECT

Change in frequency (f) or wavelength (λ) of a WAVE in relation to an observer who is moving relative to the WAVE SOURCE.

[A moving OBS's λ or f appears to be changing to a fixed observer]

~~PATH LOSS:~~

$$\cancel{\frac{P_r}{P_t} = \frac{\lambda^2}{(4\pi d)^2}}$$

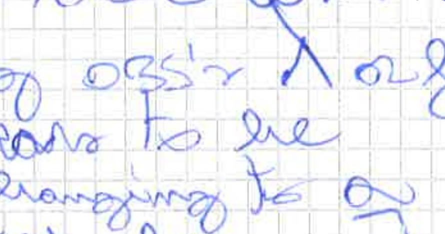
VS

SYMBOL RATE → BIT RATE

If N bits are conveyed per symbol, and the gross bit rate is R, the symbol rate can be computed as:

$$D = \frac{R}{N}$$

Ex:



BIT RATE → SYMBOL RATE

$$R = D \cdot N$$

IEEE 802.11 p - MAC

WITHOUT WAVE BASIC SERVICE SET:
Safety critical / low-latency msgs & control

WITH WAVE BASIC SERVICE SET:
2-WAY TRANSACTIONS, required to use CCH [NOTIFY / ASSOCIATION]
Ex: Tolling, internet access.

DSRC - Dedicated Short Range Comunication
Very HF (60 GHz+) for comunication over short distances.

C-V2X - Cellular V2X
Used for longer Range comunication, over a few Kms.

mm WAVE (ex. 60 GHz) ⇒ huge potential!

BIT ERROR RATE (BER)

SNR per SYMBOL.

$$\gamma_s = \frac{E_s}{N_0} = \frac{\text{ENERGY PER SYMBOL}}{\text{NOISE}}$$

SNR per BIT. ~~BINARY MODULATION~~

$$\gamma_b = \frac{E_b}{N_0} = \frac{E_s}{N_0 \log_2 M} = \frac{\gamma_s}{\log_2 M}$$

$\frac{E_s}{N_0} = \gamma_s$

~~RECEIVED SNR~~ (LESS NOISE in SNR per BIT than in SNR per SYMBOL)

RECEIVED SNR: ~~By BINARY MODULATION~~

$$SNR = \frac{P_r}{N_0 \cdot B} = \frac{E_S}{N_0 \cdot B \cdot T_S} = \frac{E_b}{N_0 \cdot B \cdot T_b} \cdot \log_2 M$$

a certain BANDWIDTH

Because all BITS considered

ERROR PROBABILITY APPROXIMATION IN COHERENT MODULATION:

COHERENT MODULATION: Requires carrier phase information at the RECEIVER, which detects & decides the transmitted data using matched filters.

SYMBOL ERROR RATE $\left\{ P_S(\gamma_S) = \alpha_M \cdot Q\left((\beta_M \cdot \gamma_S)^{\frac{1}{2}}\right) \right.$

BIT ERROR RATE $\left\{ P_B(\gamma_b) = \alpha'_M \cdot Q\left((\beta'_M \gamma_b)^{\frac{1}{2}}\right) \right.$

α_M = # NEIGHBOURS nearest to a constellation at minimum distance

$$\alpha'_M = \frac{\alpha_M}{\log_2 M}$$

β_M = Constant relating minimum distance to AVG. symbol energy

$$\beta'_M = \beta_M \log_2 M$$

Q-FUNCTION:

Probability that a Gaussian random variable X with mean 0 and variance 1 being greater than z:

$$Q(z) = P\{X > z\} = \int_z^{\infty} (2\pi)^{-\frac{1}{2}} e^{-\frac{y^2}{2}} \, dy$$

$$Q(z) = \frac{1}{2} \operatorname{erfc}\left(2^{-\frac{1}{2}} z\right)$$

PHYSICAL LAYER - PHYSICAL PHENOMENA

ENERGY: It is measured in J [Joule]

$$\boxed{E \Rightarrow J = W \cdot s}$$

POWER: It is measured in $\frac{J}{s}$ $\left[\frac{Joule}{s}\right]$

$$\boxed{P \Rightarrow \frac{J}{s} = \frac{W \cdot \cancel{s}}{\cancel{s}} = W}$$

1W ⇒ 1 Watt is the work performed when a current 1A flows through an electrical potential difference of one Volt (1V).

$$\boxed{[1W = 1V \cdot 1A]}$$

By **OHM'S LAW**:

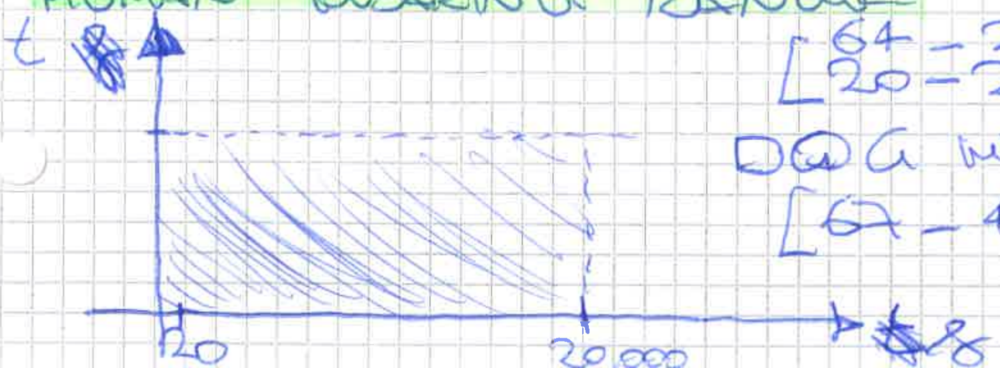
$$\boxed{1W = \frac{1V^2}{\Omega} = 1A^2\,\Omega}$$

RESISTANCE (Ω)

Measure of the electrical resistance of an OBJECT in opposition to the flow of ELECTRIC CURRENT.

$$\boxed{R[\Omega] = \frac{V\,[V]}{I\,[A]}}$$ ⇒ Given by the ratio of Voltage to Intensity

HUMAN HEARING RANGE:



[64 - 23 KHz]

[20 - 20 KHz]

DOG HEARING RANGE:

[67 - 45 KHz]

LIGHT:

Electromagnetic RADIATION within a certain portion of the electromagnetic spectrum.

VISIBLE LIGHT:

~~Vis~~ Light in the visible spectrum of the HUMAN EYE.

430 THz - 750 THz

400 nm - 700 nm

PHYSICAL PROPERTIES

INTENSIVE PHYSICAL PROPERTY [BULK PROPERTY]
Local physical property of a SYSTEM that does NOT depend on the system size or the amount of ~~~~ material in the system.

⇒ Ex:
- T = Temperature
- n = Refractive index
- ρ = Density of an object
- γ = Specific weight
- M = Melting / boiling point

AVG. when "mixed"

[PHYSICAL PHENOMENON'S MAGNITUDE is independent of the size of the system]

EXTENSIVE PHYSICAL PROPERTY: The case physical property of the SYSTEM does depend on the system size or the amount of material in the system.

[The more mass there is, the more intense the physical phenomenon]

EX:
- m Mass,
- V Volume
- W Weight
- E Energy
- C Heat

Sum of the PROPERTIES when "mixed"

LINUX BOOTING: KERNEL

BEFORE DEVICE TREE'S INTRODUCTION:

KERNEL IMAGE ⇒ Contains the entire HARDWARE DESCRIPTION, along with OS

BOOTLOADER ⇒ Loads a single binary (KERNEL IMAGE), and executes it.

⇓

ATAGS passed to the KERNEL through register r2 [info about memory size/location, kernel command line]

AFTER DEVICE TREE'S INTRODUCTION:

KERNEL IMAGE ⇒ Just containing the OS info.

BOOTLOADER ⇒ Loads two binaries: the kernel image & the DTB

DTB (Device Tree Blob) ⇒ Contains a description of HARDWARE.

The ⇒ located in: arch/arm/boot/dts [one per board]

NB: The bootloader passes the DTB address through r2.

CODE RATE: Forward Error Correction

The CODE/CODING RATE of a FEC is the proportion of useful DATA-STREAM to the redundant DATA-STREAM.

$$R = \frac{K}{N}$$

K → useful info

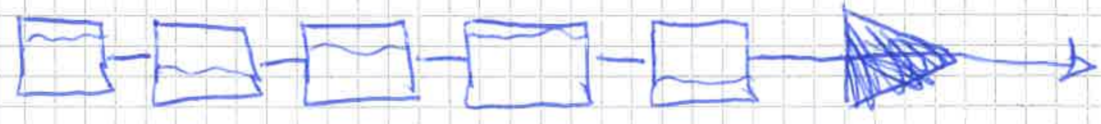
N → redundant info

CCD [Charged - Coupled Device] / ($$$)

Sensor whose goal is to "acquire" an image (i.e.: digitalize what it "sees") & digitalize it onto PIXELS.

⇒ Every pixel's charge is transferred through a limited # OUTPUT NODES to be converted to V (Voltage) ⇒ SEQUENTIAL

CCD LINESCAN



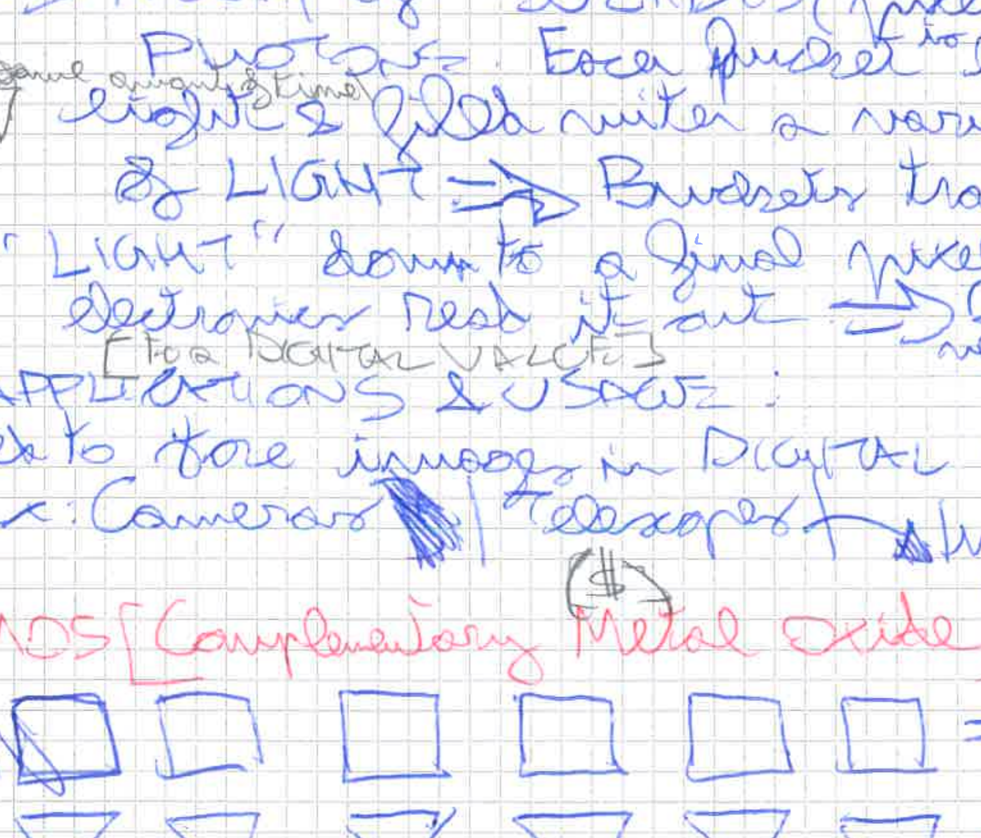
⇒ Array of BUCKETS (pixels) collect PHOTONS. Each bucket is exposed to light (for same amount of time) & filled with a varying amount of LIGHT ⇒ Buckets transfer their "LIGHT" down to a final pixel where the electronics read it out [For DIGITAL VALUE] ⇒ Resulting image very HD

APPLICATIONS & USAGE:

Used to store images in DIGITAL MEMORY [Ex: Cameras | Telescopes + Hubble - space telescope] ($)

CMOS [Complementary Metal Oxide Semiconductor]

(mobile phone)



⇒ PARALLEL

A|DC

Analog-to-Digital Conversion

[No CURRENT required excepting when changing S]

Each pixel has its own charge-to-voltage conversion, which is then digitalized (lower UNIFORMITY w.r.t. CCD)

WIRELESS NETWORKS & EMBEDDED SYSTEMS [LAB]

Dr. Claudio Salvatore

LECTURE 1 – INTRODUCTION

More and more traffic is going MOBILE!

MOBILE DATA: 2G | 3G | 4G — Licensed bands!
CELLULAR NETWORKS

WiFi } unlicensed bands!

=> NETWORKS & IoT devices are more & more pervasive.

EMBEDDED DEVICES represent one of the largest markets
98% CPUs in embedded devices
Mobile traffic (M2M) growing exponentially

IoT NETWORK: Network of devices inter-connected through an internet connection in a fast, scalable, secure & pervasive manner.

More and more mobile | edge devices are interconnected to the internet

OBJECTS (THINGS): Now also equipped with an IP STACK => Internet model exportable to "THINGS", with proper modifications & optimizations

=> PERVASIVE NETWORK
"All around me"
(mobile + embedded)

APPL. LAYER	COAP
NETWORK LAYER	6LOWPAN

6LOWPAN: (L3) Minimal-overhead NETWORK LAYER for IOT devices.

CoAP: (L5) Lightweight HTTP-like protocol for resource-constrained devices

In IOT, M2M, THING 2 THING is the new focus of COMMUNICATION.

=> 5G + M2M is enabling future technology.

More and more computational LOGIC is being moved to the CLOUD & EDGE

MAIN VERTICAL APPLICATION FIELDS:
- Industries & factories [PREDICTIVE MAINTENANCE, INVENTORY MNG.]
- hospitals [REMOTE PATIENT MONITORING]

WIRELESS SENSOR NETWORK: Spatially distributed autonomous devices using SENSORS to cooperatively monitor PHYSICAL OR ENVIRONMENTAL CONDITIONS (temperature, sound, vibration, motion)
[Ex: Cyber-Phy SYSTEMS (NB: NO internet connection)]

GOAL: Acquire info. from the field, to have an augmented knowledge of the world.

FEATURES of a WSN:
- Sample data from the environment
- Wireless communication
- Autonomously powered (batteries)
[MINIMAL INFRASTRUCTURE, large # UNITS]

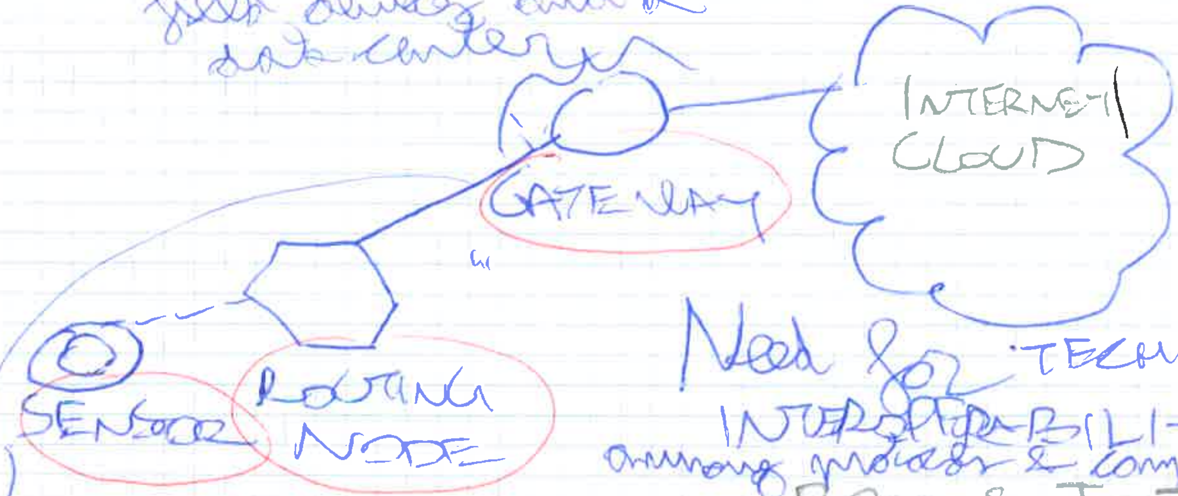
WSN vs IOT:

Set of NODES equipped with low-power radios, smart sensors and embedded CPUs.
⇒ SENSING & MONITORING SERVICES in parallel

Interconnection ~~between~~ among identifiable devices ~~without~~ over the internet (INTERNET connection) for the sensing & monitoring process (EX: Rain in California observation)
↳ NETWORK of devices connected to the INTERNET.
↳ Embedding intelligence into devices to make them "SMARTER" + INTERNET CONNECTION } EDGE COMPUTING

MAIN IOT COMPONENTS.

Proxy between field devices and a data center



Need for TECHNICAL INTEROPERABILITY among protocols & components
Base of IOT.

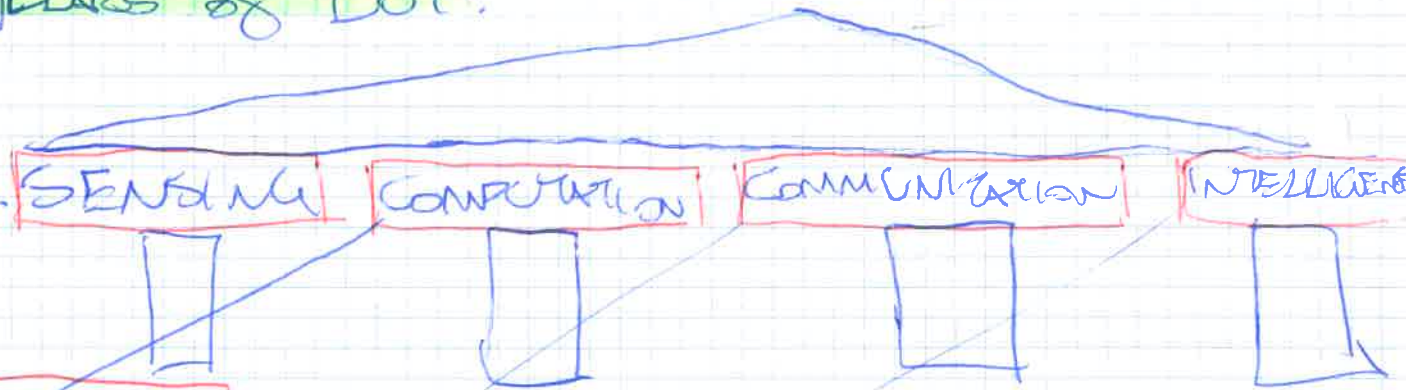
↳ **IOT WIRELESS SENSOR**: Device able to "SENSE" the environment, equipped with ~~a~~ wireless TX capabilities and generally powered by a battery
↳ Limited CPU / main memory / ~~storage~~
↳ Lots of storage (CLOUD / SD)
↳ Low-power radio, limited range and low-power batteries.

↳ **GATEWAY**: Acts as a PROXY (forwarder) of data acquired from the field

=> Performs data transfer from IoT devices to the CLOUD | INTERNET.

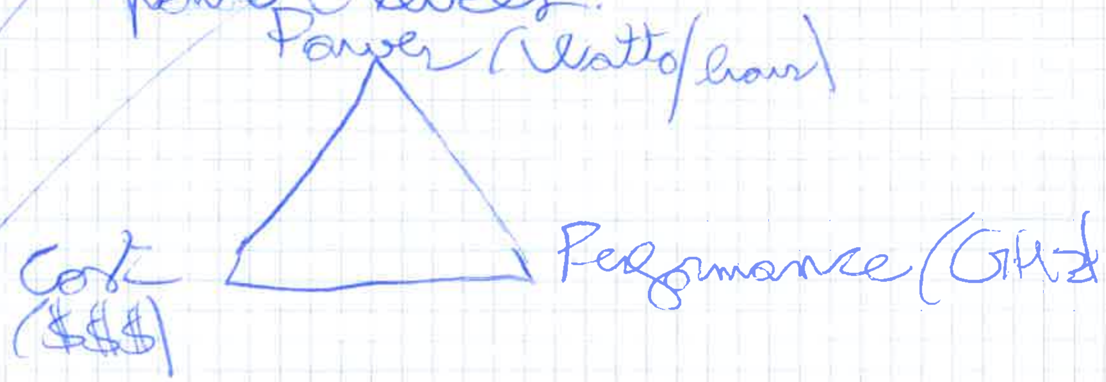
CENTRALIZED SYSTEM (MONOLITHIC) [PAST]	VS	DISTRIBUTED SYSTEMS:
Single POF | POC		No POF | POC
Single process		Multiple (concurrent) processes
Resource sharing		Resource distribution
Homogeneous tech.		Heterogeneous tech.

4 "PILLARS" of IoT:



→ SENSORS
POOR DATA => POOR MODELS => POOR DECISIONS
"The right sensor" is important.

→ MCU: Microcontroller Unit, at ultra-low power levels.



→ COMMUNICATION | Standardization extremely important for INTEROPERABILITY among devices.

→ "EDGE | CLOUD | FOG computing.
Distributed processing: bringing the computation to the EDGE => reduce the data

being sent/stored on CLOUD, & hence being transmitted. [Better for DATA PRIVACY & DATA CONFIDENTIALITY]
⇒ ON-SITE FILTERING of DATA.
[OPTIMIZE in TRANSMISSION / ENERGY CONSUMPTION]

BRIEF HISTORY of WSN:

1980: DSN (Distributed Sensor Networks)
[DARPA] Surveillance & tracking of subsonic low-flying aircrafts.
Sensors (audio) small arrays of microphones
(video) visible wavelength TV cameras
CIA/NSA

⇒ MILITARY, AGRICULTURE, INDUSTRY 4.0, HEALTHCARE, MONITORING, ITS

- MILITARY: "Rembass - II" system
- AGRICULTURE: "Vini Veri" for temperature/humidity sensors
- BUILDING MONITORING: For "Torre Aquila", Trento. Earthquakes
- HEALTHCARE APPLICATIONS: ~~for~~ human body status monitoring.
- SMART HOUSE: accelerometer, temperature ⇒ LIGHT & ENERGY SAVING / reduced costs + heating system
  ↳ DOMOTIC APPLICATIONS
- INDUSTRY 4.0: Augmented safety & security in factories. Widespread deployment of sensors for DEADLY/HARMFUL substances monitoring.
- ~~ID~~ SMART CITY: Improved urban mobility & coordination among vehicles

- SMART FACTORY: ~~Sensors~~ GAS sensors to detect the presence of GAS | warehouse OPTIMISATION.

SENSORS ⇒ Gateway

- IoT for last-mile logistics, grouping trucks load and increasing efficiency.

EMBEDDED DEVICE: Abstract "THING", capable of interoperating horizontally.

"THINGS": Capable of producing information independently from applications.

⇒ INFORMATION FLOW ⇒ Bound to a SERVICE (no matter the technology implying the service)

~~IoT Internet~~

IoT: Interconnection among devices addressable over the internet

"Interconnection of OBJs equipped with SENSING, COMPUTING, COMMUNICATION & limited ANALYTICAL capabilities

LECTURE 2 – DEVICE ARCHITECTURES & COMPONENTS

IOT NODE
→ MCU
→ Sensor modules
→ RADIO TX/RX
→ Energy harvester (solar panels)
→ Energy source (battery)

LIMITED MEMORY | COMPUTATIONAL | N. BANDWIDTH | POWER CONSUMPTION
⇓
Tradeoff among different characteristics (NO SILVER BULLET exists)

↳ Required for any kind of APPLICATION

MCU: MICRO CONTROLLER UNIT: Combination of CPU with other units (such as TIMERS | MEMORY | GPIO FUNCTIONALITY)
↳ System on Chip (SoC)

SENSORS: Provide access to the external world's data. "Sense" the world (5 senses = SENSORS have the same purpose)

SENSE → COMMUNICATE → PROCESS (ANALYZE) + STORE → COMMUNICATE → ACTUATE

COMMUNICATION INTERFACES: USB - based. Used to communicate data to a different location

POWER: Extracted from BATTERIES or by AC/DC conversion with MANIFESTERS

MICROCONTROLLER VS MICROPROCESSOR

CPU, RAM, ROM, I/O, timer are all on a single chip [FIXED RAM/ROM, I/O] ↳ Single-purpose (SoC)

// (easiness & usage VS performance)

Standard CPU running on commodity computers [Flexible RAM/ROM, I/O peripherals] ↳ General-purpose

CPU + MEMORY + PERIPHERALS I/O

PERIPHERALS (accessible via BUS) → I²C, UART, SPI, Ethernet

BUS: access the RAM

- FLASH MEMORY (Re)-writable
- TIMERS, to synchronize events & time measurement
- INTERRUPT MANAGER: manage a peripheral
- (DMA) ⇒ Do not pass through the CPU, write directly into the MEMORY and generate an INTERRUPT when done.
- (PARALLEL PORTS)
- PRINTERS (?)

BUSES

~ typical serial port

- UART: Universal Asynchronous Receive Transmit. Ex: Accelerometer in project

1:1 PROTOCOL, 2/4 WIRES in parallel

Generally not fast (serial signal-based)

Different speeds / modes are possible | wire length

- I²C → Multiple hosts. One master, many slaves. 100 KHZ / 400 KHZ / 1 MHZ ⇒ Sampling rate

N:M NETWORK

2 WIRES — CLOCK — DATA

Low-price sensors

Up to 255 hosts, with an 8-bit address (2^8)

Δ Addressing enables N:M

SPI: Serial Peripheral Interface.
Fast for SHORT WIRES (~10 MHz) but highly subject to EM noise (higher frequency)
Complex, more expensive sensors
- MASTER-SLAVE mode for operation.

NB: higher frequency
⇓
(higher energy consumption)
⇓
Faster response time
(registers faster than DMA!)

DMA allows to transfer directly into/from any memory location
+ into peripherals "mounted" (mapped) in memory. A part of memory is "allocated" to peripherals when they are mounted.

TRANSCEIVERS

Exploit radio frequency spectrum to transmit data. (2,4 GHz, IEEE 802.15.4 Zigbee | 6LOWPAN
↳ Use ENERGY SAVING TECHNIQUES in EMBEDDED SYSTEMS (Ex Standby while IDLE)

SENSORS:
- Accelerometer (I^2C connection) (I^2C | SPI)
- Temperature (analog | digital)
- DIGITAL Compass
- LIGHT (light intensity) tension proportional to LUX level

- GPS | GALILEO: Return the position (high energy consumption)

SENSORS:

ANALOG VS DIGITAL

ANALOG

Output signals are analog & produce continuous analog signals (output ∝ measured value)

Ex: THERMOMETERS, measuring temperature based on the change in resistance of a material.

DIGITAL

Data sensing & transmission take place digitally

⇒ INPUT SIGNAL is converted into digital one.

DISCRETE VALUES (constant values over time)

In base al cambiamento di capacità

MEMS: Micro Electro Mechanical Systems

(1989) Sensors having also electronic and compact structures.

⇒ DIODES, CAVITY, imitating "mechanical" parts ~~(SIMULATED)~~ (REPLACE MECHANICS with ELECTROMAGNETIC PARTS)

Little space taken

⇒ Used in SMARTPHONES for GYROSCOPES, accelerometers, blur correction, ...

[Detect the shift in MASS to make a variable differentiable (capacitor)]

MOVING: capacity changes

STEADY: constant capacity

IMAGE SENSORS: Detects & conveys information making up an image [LUMINANCE LEVEL]

[Convert variable light attenuation into small signal bursts → PHOTOELECTRIC EFFECT]

↳ CCD: (Expensive, high quality)

Charge-Coupled Device

EXCURSUS ABOUT ELECTRICITY:

$\boxed{V = \text{Voltage}}$ Allows ~~the~~ the device to "stand" on its feet & actually operate.

$\boxed{I = \text{Intensity}}$ Actual current flowing through the system.

$$V = \max(V_0, V_1, \ldots, V_{n-1})$$

$$I_R = \sum_i V_{Ci}$$

INTENSITY: We must have that:

$$\boxed{\underbrace{I_{INPUT}}_{\text{Input intensity value provided by the power supplier}} \geq \underbrace{I_R}_{\text{Intensity required for device to work}}}$$

VOLTAGE: $\boxed{V_{INPUT} \approx V_R}$

This condition may occur, based on the APPLIANCE'S TYPE.

- APPLIANCE WITH NO MOTOR (RESISTIVE LOAD): Don't need VOLTAGE STABILISERS. If less voltage supplied, less current will flow through the system. If more voltage supplied, more current will flow through. (If TOO HIGH, may burn the circuits!)
- APPLIANCE WITH MOTOR (INDUCTIVE LOAD) Usually need a voltage range ⇒ Need for a VOLTAGE STABILISER to protect appliances from SPIKES in electricity.

CONVERTER / TRANSFORMER Used to lower the input voltage tension & the electric current signal.

Every pixel's charge converted to Voltage & transmitted as an analog signal.
Complementary Metal Oxide Semiconductor
↳ CMOS: Each pixel has its own charge-to-V conversion. Massively parallel, high
bandwidth.

POWER SUPPLIERS.

Need for correct Voltage / Current to power up devices, even embedded ones [input V too high]
⇒ LINEAR / SWITCHING CONVERTERS: Can be used to bring input V to the right one.

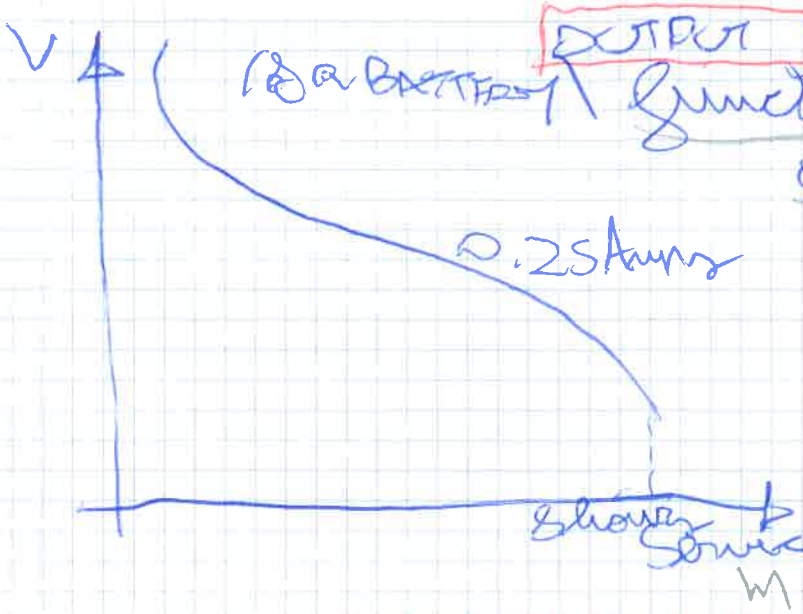
LINEAR CONVERTER: Reduce Voltage value ($, but 5-30% efficiency) the same using voltage partitioning circuits (Voltage dropout) ⇒ Dissipated as heat [Lots of HEAT!]

Efficiency = $\frac{V_{output}}{V_{input}}$ (same as the efficiency of a car)
(car's efficiency 20-30%)

SWITCHING CONVERTERS: Draw small amounts of energy & transfer it step-by-step to the output ⇒ ELECTRONIC SWITCH (Gate-like), controls charge transferred to output
Efficiency ≈ 95%, but more expensive.

BATTERIES.

- NiCD (NICHEL CADMIO): $, simple charge, short lifespan / Rechargeable
- NiMH (NICHEL-METAL IDRURI) $$, high discharge, simple charge.
- LITHIUM ION: $$$, long lifetime (3000 mAh), complex charging
(LIPO)



OUTPUT VOLTAGE of a BATTERY: Non-linear function of the residual charge of the BATTERY

[Low voltage area]

⇓

More current drained

⇒ Hence the reduced power of my electric toothbrush

HARVESTERS:

MOTION ENERGY (DINAMO on bike) | SOLAR (solar panels) | THERMO (~~generate steam~~ thermopower + turbine drives a TURBINE) GENERATOR)

Acquire energy from a potential energy (electrical) in the environment source

DESIGNING AN IOT PRODUCT:

IDEA ⇒ MARKET RESEARCH (Who is going to buy this product?) ⇒ SPECIFICATIONS
Specify functions "WHAT WILL IT DO?"

⇓

FUNCTIONAL & NON-FUNCTIONAL SPECIFICATIONS ⇒ COMPONENTS (HARDWARE) SELECTION ⇒ HW & SW DESIGN

⇓

CASE SELECTION

Energy efficiency concerns

Precision | cost ⟺ Energy efficiency | power consumption

EX: CONTIKI MAC
Periodic wake-ups to listen for packets.

Communication (sleep | awake duty cycles) ⇒ Largest consumer of Energy

~~REQUIRES~~ LoRAWAN: "Sleep until woken up for reception (SLEEP MODE | SWITCH ON | SWITCH OFF)"

HW & EMBEDDED SW DESIGN:

HW BUG FIXING ⇒ HW DESIGN 1. → Schematic design in a standard notation

⇒ HW SAMPLING ~~DESIGN~~ 2. (PCB PRODUCTION) ⇐ Printed Circuit Board

→ GERBER: specify components positions & interconnections

TEST SUITE IMPLEMENTATION (HW & SW tests) 3.

⇖ HW DEBUG 4. (Test based on oscilloscope / logic state analyzer)

PCB: Production of a single OBJ, made up of multiple layers of materials.

↳ PCB ASSEMBLY: solder different components together

INDOOR VS OUTDOOR DEVICE DEPLOYMENT

IK CODE [0-10] Grado di resistenza agli urti ⇒ Protection against mechanical impacts

IP CODE (International Protection): Protection against water & dust, body intrusion / parts

1-6 vs. SOLIDS   1-8 vs. WATER

TASK of a PRODUCTION ENGINEER

LECTURE 3 - EFFECTIVE SOLUTIONS TO DEPLOY SN SENSORS & GATEWAYS.

When designing an IOT ARCHITECTURE, the most important thing to consider is the INTEROPERABILITY among devices.

Allow to "understand" one another | interoperate

- SYNTACTICAL INTEROPERABILITY: (GRAMMAR) JSON | XML: "$\boxed{\text{How}}$ I say it makes sense to both of us"
- SEMANTICAL: "$\boxed{\text{What}}$ I say makes sense to both of us" ~~[illegible]~~

6LOWPAN + COAP: IPV6 - based peers communicating with one another. + Router for internet access (Ex: Mobile phone runs a large gateway)

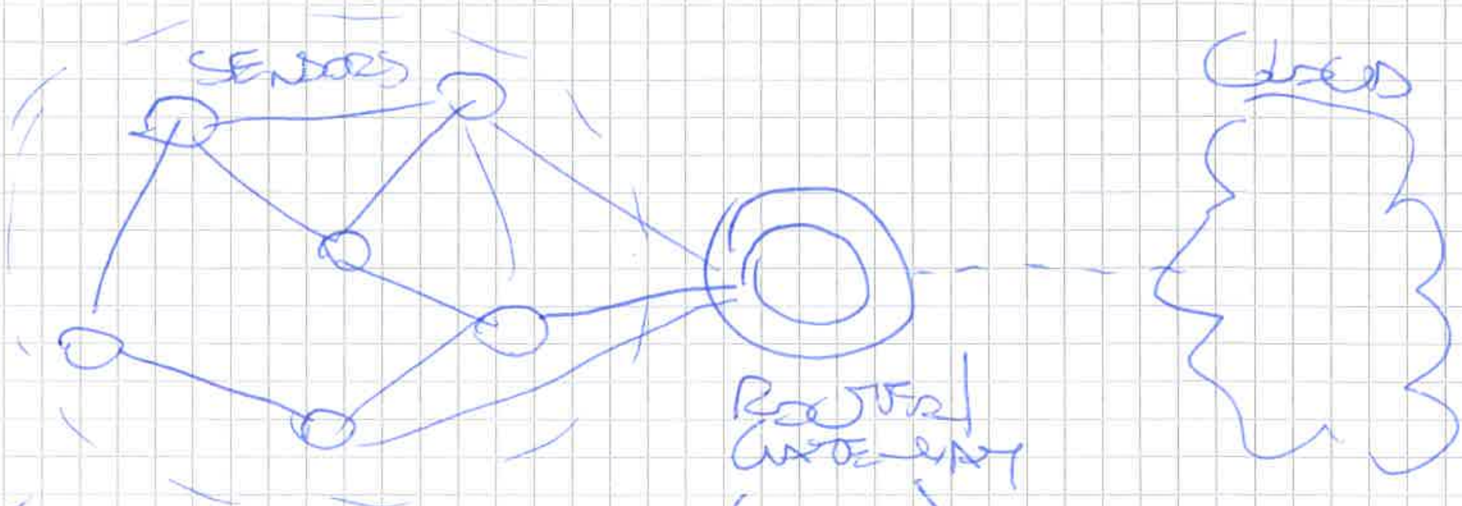
LORAWAN

No IP, network server used for INTEROPERABILITY.

NB - IOT: Make use of a "virtual" gateway, with good interoperability characteristics. Based on pre-existing IOT infrastructure.

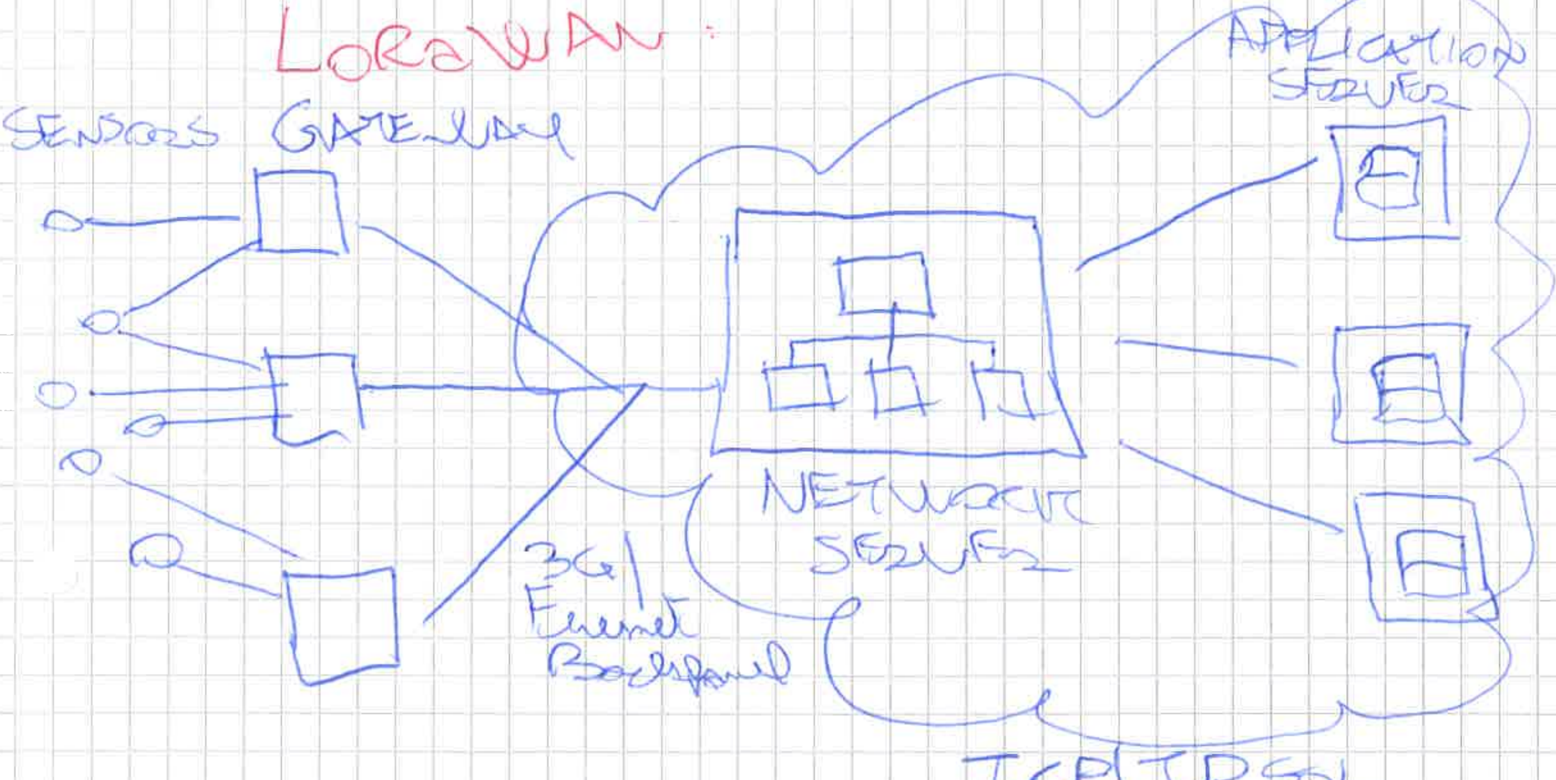
	APPLICATION L.	→ COAP + JSON
	TRANSPORT L.	→ UDP
	NETWORK L.	→ 6LOWPAN
MAC LAYER		→ IEEE 802.15.4
MAC LAYER	DATA LINK LAYER ~~PHYSICAL L.~~	→ IEEE 802.15.1 V4 | V5
	PHYSICAL L.	→ IEEE 802.15.4 (Zigbee) 802.15.1 (Bluetooth)

6LoWPAN & CoAP NETWORK ARCHITECTURE

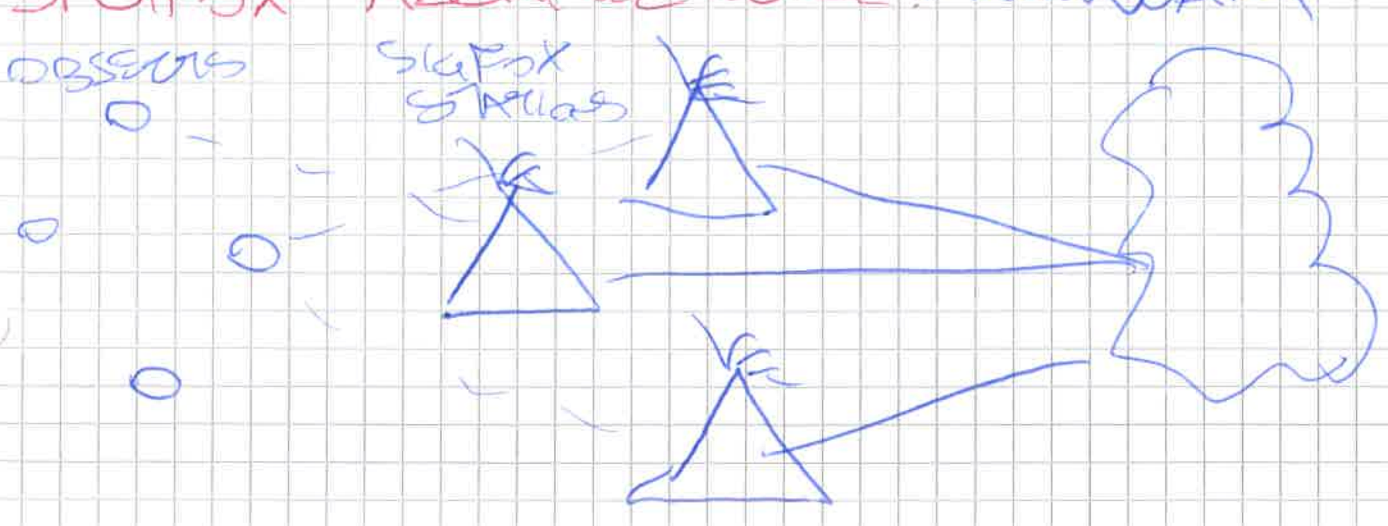


6LoWPAN network (IPv6)

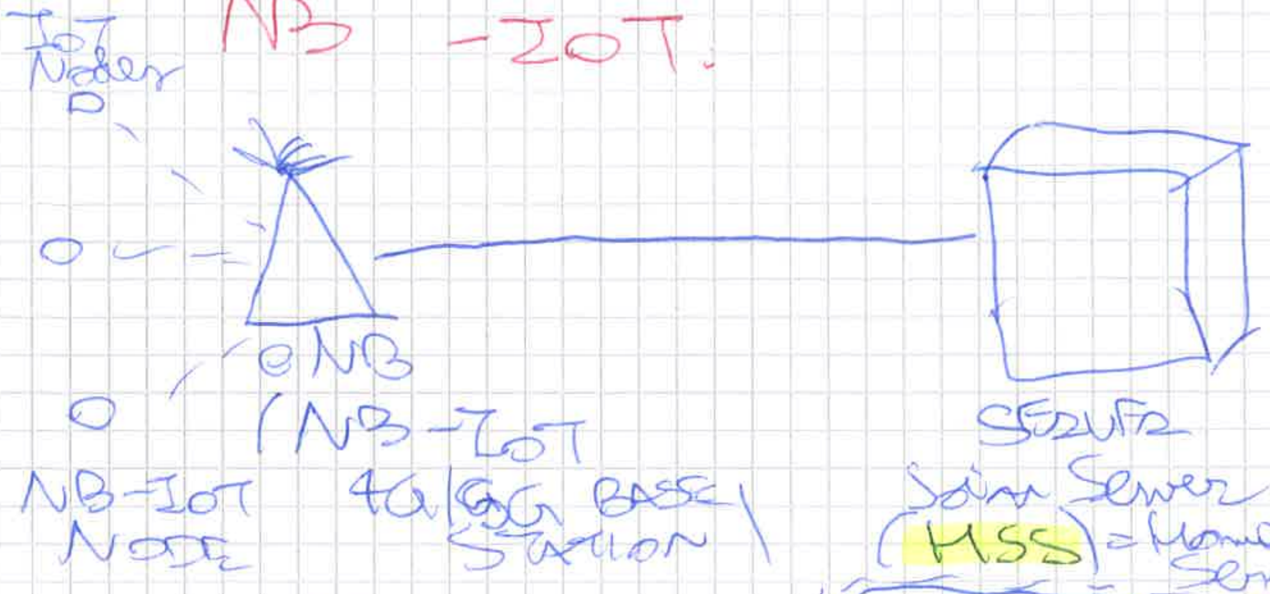
LoRaWAN:



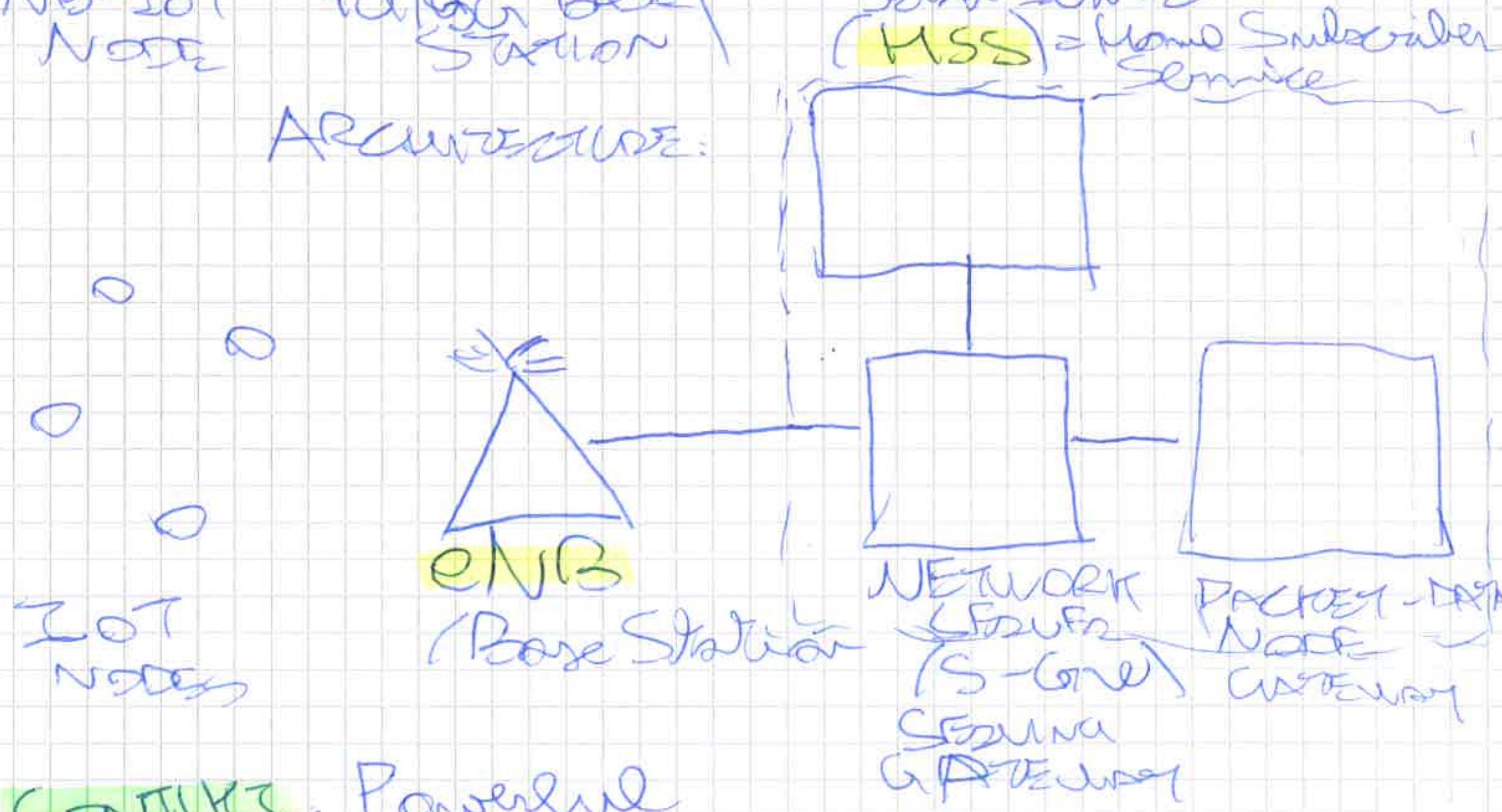
SIGFOX ARCHITECTURE:



NB - IOT:



ARCHITECTURE:



CONTIKI: Powerful low-power internet COMMUNICATION (support for IPv4, IPv6)

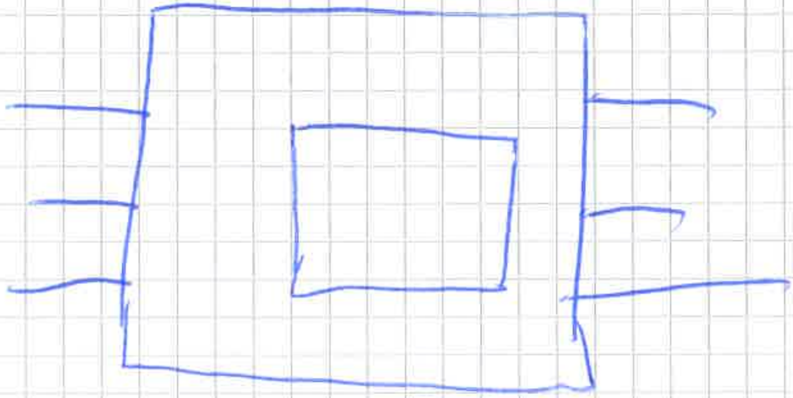
+ CONTIKI MAC (sleeping routers) to operate wireless routers.

⇒ $\boxed{\text{SLIP}}$ (Serial Line IP) to connect IoT network with high-end devices.

GATEWAY: Fundamental entity for INTEROPERABILITY among different network.

MICROCONTROLLERS - SoC

Ad-hoc purpose for a certain task.
⇒ Allows to program close to the HARDWARE.



STEPS (to program a microcontroller)

1. Design of program

2. Implementation of new program (C) interfacing with HAL (Hardware Abstraction Layer) to manage PHY resources

Ex: #DEFINE PIN_8_HI 1

We configure EVENTS in order to be able to "call" them & code abstractions;
CODE for controlling actions

HARDWARE-INDEPENDENT PROGRAMMING would be possible if sufficient abstractions are in place.
(Currently, not done & not possible !)
"Stick to one HW manufacturer"
⇒ technical lock-in.

Each HW type has its own microcontroller & code

=> SCHEDULER can be coded over a MICROCONTROLLER.

=> CONTEXT: current state of the process

=> CONTEXT SWITCH: change the current process being executed

INTERRUPT: higher priority context switch. Blocks whole system until a certain operation is performed fully

```
while (......) {
  // QUEUE of OPERATIONS (FIFO)



}
```

CALLBACK

PROTOCOL - THREADS: "Simulation" of threads in a system.

EVENT - DRIVEN KERNEL: Kernel where the behaviour is given by EVENTS arising & their handling [Similar to Keyboard programming]

DRIVER: Pointer to a function written in C/C++ to interface with a certain H.W. component. => Used to interface

Ex: Contiki OS with a certain DEVICE H.W. component
HARDWARE

OPERATING SYSTEMS in IOT

OS: Abstraction of the physical machine through a simple interface
→ SERVICE: Single part of the OS providing certain functionalities.

An OS is also a RESOURCE MANAGER!

RESOURCES: All physical entities of a computing machine.
→ ABSTRACT RESOURCES are made accessible by the OS.

WSN ⇒ No virtual paging & file system needed (too much overhead)

TRADITIONAL OS	EMBEDDED OS
Large overhead (Many MBytes) Ex: Linux, MAC OS X, Windows	1 KB (OS) + 2 KB (NETWORK) + 1 KB (APPLICATION) → Store as little data as possible & send it out as soon as possible.

POSIX on MCU? MBytes overhead
→ FULL-POSIX compatible ≃ 1 MB. ROM/RAM?
⇒ Profiles for supporting a subset of the STANDARD.
REAL-TIME, MULTI-PURPOSE
→ The more minimal, the fewer functions & the least space required (# PSES1, 55-100 KBs)
→ TOP-DOWN APPROACH to API DESIGN

SoC => BOTTOM-UP approach with a minimal system & API to efficiently describe embedded systems [LIMITED RESOURCES & TEMPORAL REQUIREMENTS]

≈ (1 - 10 KBs) desired for OS (E.g. Contiki, ERIKA) REAL-TIME OS

100 KBs - 1 MBs (POSIX, LINUX REAL-TIME)

CONTIKI OS: Lightweight open-source OS for WSN sensor nodes

- Portable
- RAM: 2 KBs
- Event-driven: an event can trigger kernel => call event handler
- ROM: 40 KBs

NETWORKING: 6LOWPAN + COAP with low-power internet communication.

GATEWAY: Network node equipped for interfacing with a network that uses a different protocol (allows interoperability among different networks)

↳ FOR NETWORK INTEROPERABILITY

↳ WEPTEK SAKER: 2.4 GHz | 868/915 MHz

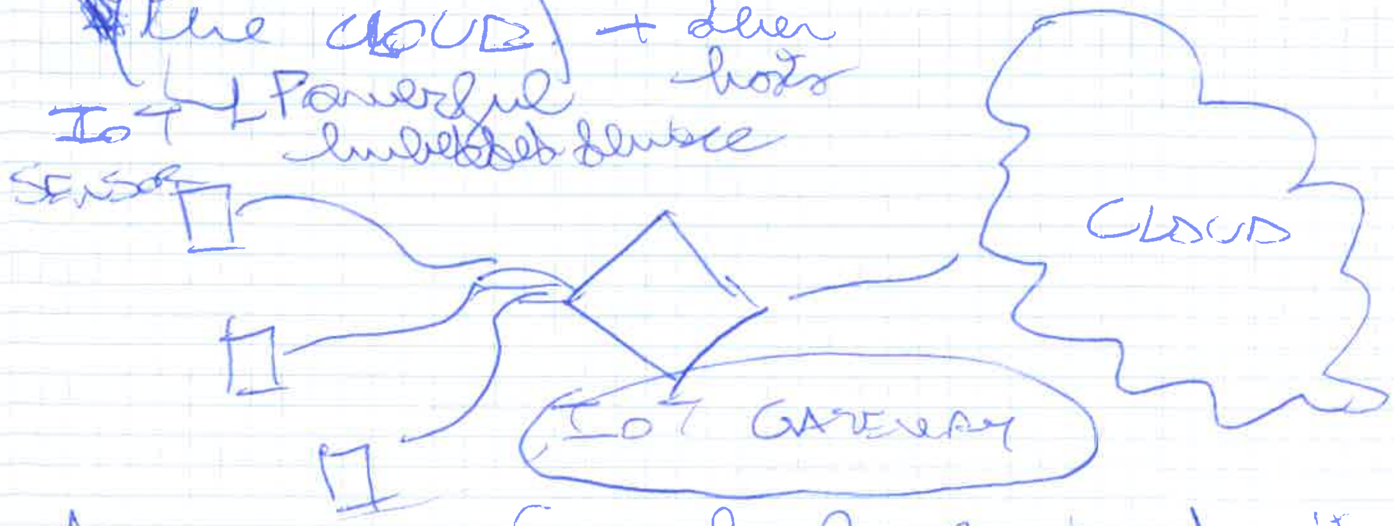
6LOWPAN IOT GATEWAY = Border router in an IOT network (IoT N.) Sensor node with an Ethernet ⇓ Ethernet connection [MANAGEMENT => On Cloud]

It may be smart to move the computation to the EDGE, at the router itself

E.g. Risky! What if cloud company bankrupts where data is collected

IoT GATEWAY "Bridge" between on-field devices & outer internet (the CLOUD) + other hosts

IoT → Powerful embedded device



Modular & extensible SW support ⇐ Can also be equipped with LOCAL PROCESSING & STORAGE CAPABILITIES.

- MODULAR SYSTEM DESIGN
- SW SUPPORT
- ~~SOME COMPUTING CAPABILITIES~~

EX: EMBEDDED GATEWAYS, with some "intelligence" on board

- → RASPBERRY PI
- → BEAGLE BONE BLACK
- → NGS SMART GATEWAY (compatible with BB black)

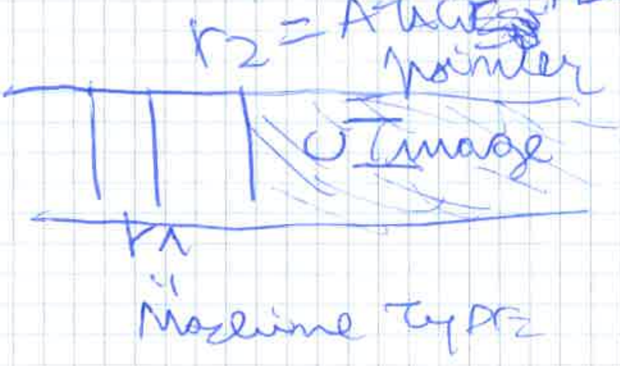
GATEWAY OS:
- Contiki
- Linux distributions
- Debian
- Fedora

(Before the DEVICE TREE)

~~LINUX~~ **KERNEL** & **BOOTLOADER**

Contains the description (as an IMAGE) ⇒ Bootloader loads a single binary (KERNEL IMAGE) & boots it.

ATAGS (extra info): Passed on to the bootloader for booting the system [r1]
⇒ Bootloader tells the kernel on which board it is booting [r2]

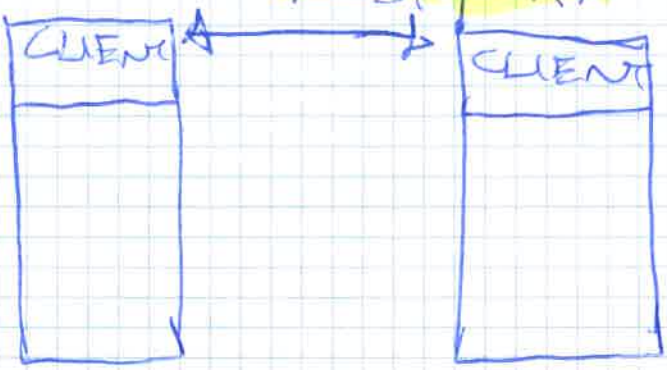


(WITH A DEVICE TREE):
KERNEL & HW description contained in the DTB (Device Tree Blob) — arch/arm/boot/dts

⇒ Bootloader loads into the kernel image & the DTB (Device Tree Blob)

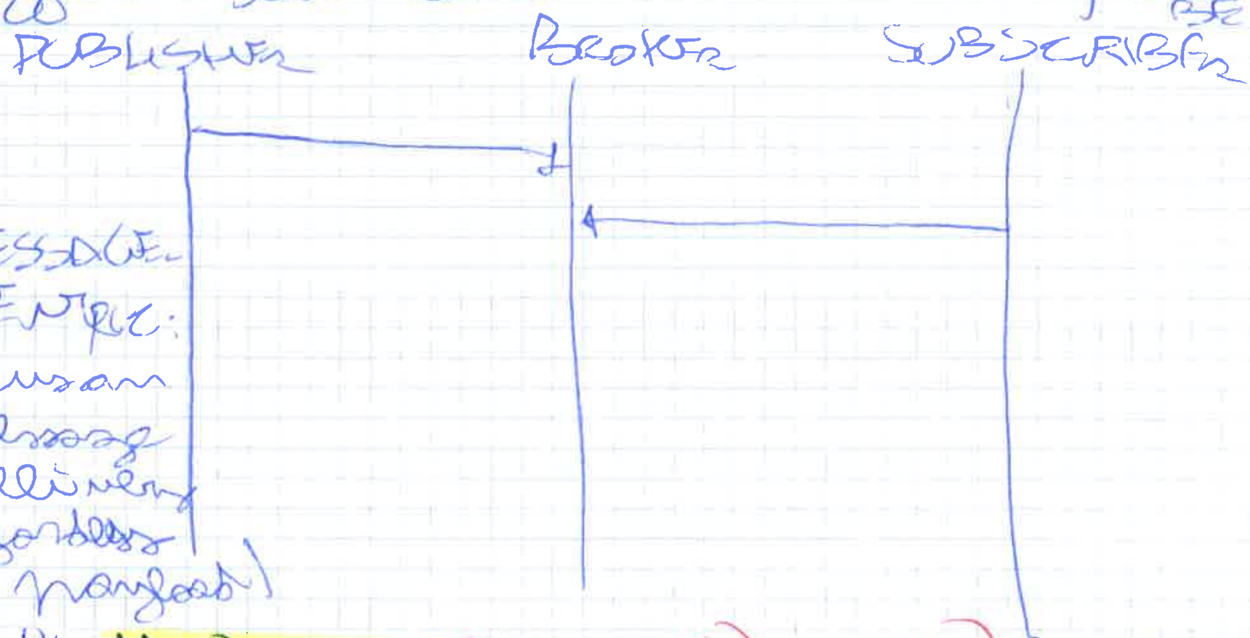
PROTOCOLS for DATA EXCHANGE:

- BUS-BASED: Direct communication between devices/peripherals attached to it
  (Request/response used between end devices to exchange information)
  REST | CoAP



EX: REST | CoAP
⇒
DATA-CENTRIC:
Focus on delivering meaningful/more data

- BROKER-BASED: Centralized BROKER handles the communication between the different entities, based on PUBLISH/SUBSCRIBE.



⇓

MESSAGE CENTRIC: Focus on message delivery (regardless of payload)

EX: MQTT: Message Queue Telemetry Transport

Add one level of INDIRECTION via a QUEUE of msgs

Queue of messages

SoM = System on Module — System running on a single H/W module

SoC = System on Chip — System running on a chip

REAL-TIME SYSTEM: System with strict latency requirements (deadline-driven)

I²C: Used to satisfy deadlines, very much in real-time.

PRU: Programmable Real-Time Unit, used to manage PINs & write directly on memory

of a MICROCONTROLLER

SWITCH (I²C) {
case (N)...

LECTURE 4 – WIRELESS COMMUNICATIONS IN SENSOR NETWORKS

(PHY & MAC LAYERS)

Coverage & data rate increases

↑ Provide

↓ Receive

Separation of concerns !!

Open System Interconnection
!!
ISO (OSI) LAYERS

- APPLICATION LAYER = Actual users' interaction with the final application (HTTP)
- ~~PRESENTATION LAYER~~
- ~~SESSION LAYER~~
- TRANSPORT LAYER = Multi/demultiplexing, end-to-end communication (UDP/TCP)
- NETWORK LAYER = Addressing & routing & transmission data
- MAC — DATA LINK LAYER = Transfer data between network entities (MAC for channel access CSMA/TDMA)
- PHY LAYER = Electrical / physical device specifications
  - IEEE 802.11 WiFi
  - IEEE 802.3 (Ethernet)
  - IEEE 802.16 WiMax
  - IEEE 802.15.4 (Zigbee)
  - IEEE 802.15.1 Bluetooth

NETWORK STACK in LINUX KERNEL:
- 500 KBs minimum footprint
- RAM requirements 4 MBs (not in all devices)

WLAN STANDARDS

WiFi (IEEE 802.11) — LAN
Radius ≈ 100 meters
1999 - IEEE 802.11a - 54 Mbps
2012 - IEEE 802.11ac - 6.93 Gbps

Bluetooth - (IEEE 802.15.1) - PAN

ZigBee - (IEEE 802.15.4) - WPAN

Low rate, low-energy power consumption
⇒ Potentially multi-year battery life.
[Standardization of higher layers by the ZIGBEE CONSORTIUM]

VLC - IEEE 802.15.7

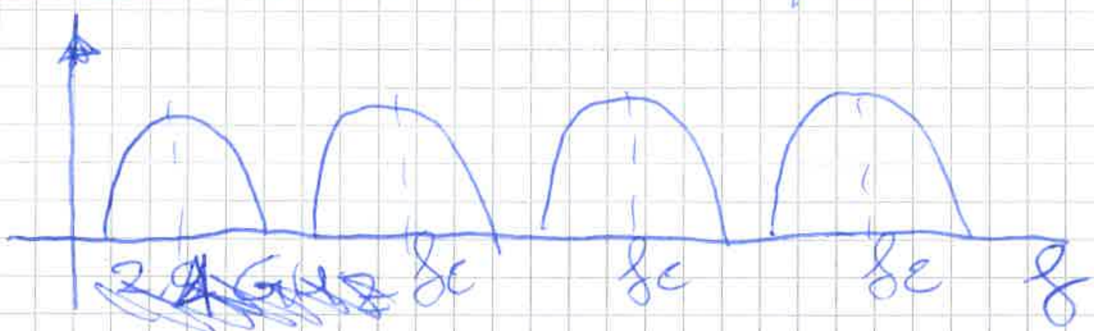
ZIGBEE - IEEE 802.15.4:

Developed for WPAN (Wireless Personal Body Area Networks)

$\boxed{\text{RANGE}}$ 50-150 m

MODULATION: BPSK | ASK | O-QPSK

$\boxed{\text{DATA RATE}}$ ≈ 250 Kb/s

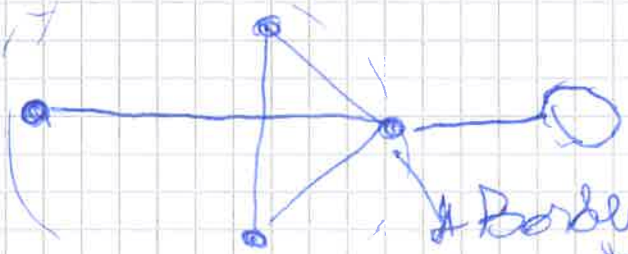


$\boxed{\text{FREQUENCIES}}$ 2.4 GHz | 800/900 MHz

DSSS: (Direct Sequence Spread Spectrum)
Analogously to WiFi, the input signal is XORed with a CHIPPING sequence to increase the BANDWIDTH occupied.
⇒ Immunity to jamming, decreased noise interference, but decreased data rate + ENCRYPTION of signal

MAC IN IEEE 802.15.4:

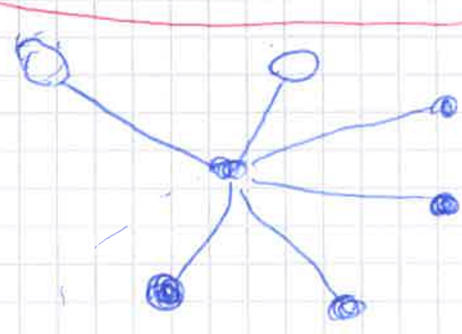
P2P TOPOLOGY:



+ Border router

Direct P2P connection among nodes

STAR TOPOLOGY



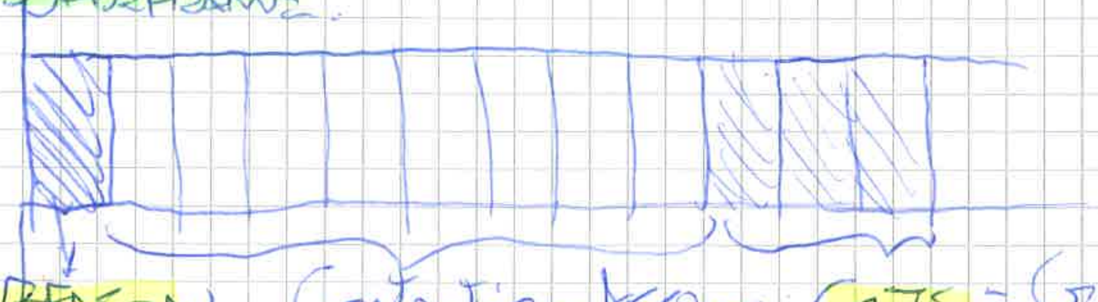
+ PAN COORDINATOR (Wi-Fi AP)
Main controller

NON-BEACON MODE: Sends data according to unslotted CSMA/CA.

BEACON MODE: HYBRID between SLOTTED and CSMA/CA
SCHEDULE-based protocol (GTS mechanism)
⇒ DEGRADABLE, POWER SAVING, LOW OVERHEAD
↳ In such mode (STAR topology), the node waits for the BEACON ⇒ If the node has a GTS allocated, it waits for its Guaranteed Slot
If the node has NO GTS allocated, it performs CSMA/CA in CAP.

CAP = ~~Controlled Access Period~~ Contention Access Period (CAP)

SUPERFRAME



BEACON — Contention Access Period (CAP) — GTS = Guaranteed Time Slot

↳ Slotted CSMA/CA

PHY PDU = 127 bytes

In NON-BEACON MODE, each node is every time in the ACTIVE PERIOD (contention all the time)

SUB-GHZ PROTOCOLS

SUB-GHz waves allow for long-range transmission (as shown by FRIIS' LAW)

$$\frac{P_r}{P_t} = G_t \cdot G_r \cdot \left(\frac{\lambda}{4\pi d}\right)^2$$

FRIIS' LAW

⇒ Fewer base stations & repeaters needed for the same #DEVICES

IEEE 802.15.4g

Sub-GHz dialect of IEEE 802.15.4

[Only the PHY level is modified]

$\boxed{\text{Data rate}}$ 50 Kbps – 300 Kbps

↳ $\boxed{\text{Wi SUN}}$ = Wireless Smart Utility Networks

LORAWAN = Low poweR Wide Area Network

PROPRIETARY standard of LoRa®

Frequency Range: 434, 868 MHz

Bit rate: ≃ 11 Kbps (10937 bps)

Modulation: CSS - Chirp Spread Spectrum using LoRa Technology Modulation

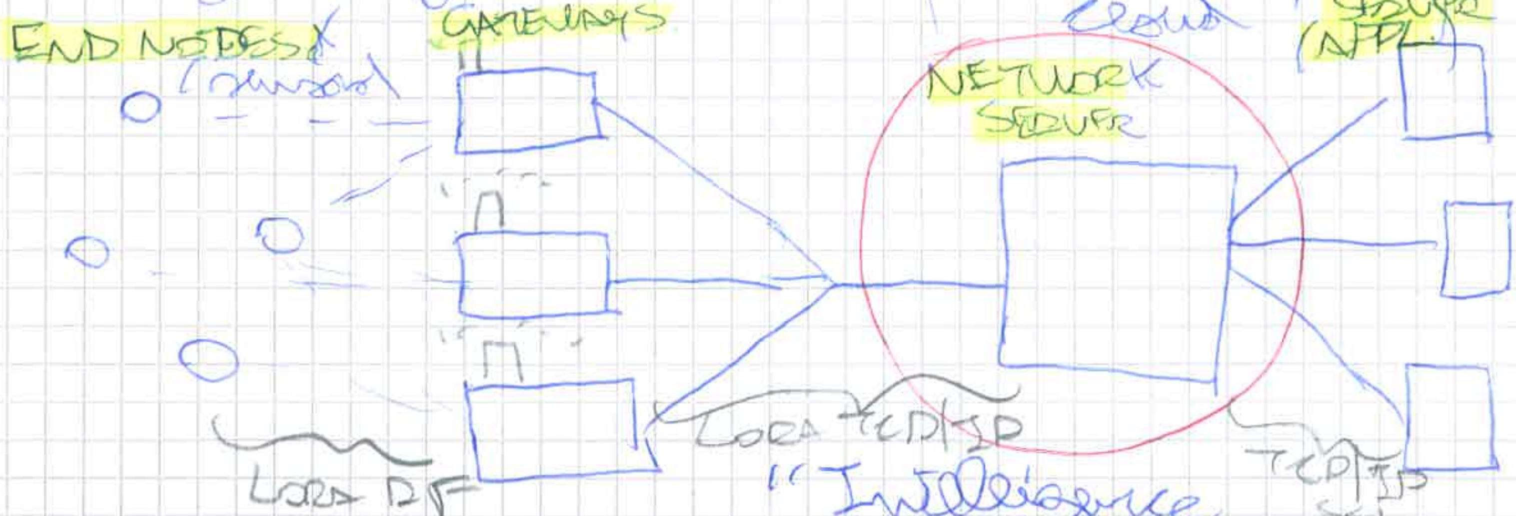
RANGE → Urban: ≃5 Kms
→ Countryside: ≃15 Kms

Bandwidth: 125/250 kHz DOWN/UP

ARCHITECTURE: Star of STARS.

SENSOR NETWORK: Star topology

STAR of GATEWAYS: (Connected to the cloud)



"Intelligence lies on the NETWORK SERVER"

* Filters duplicate packets from gateways, performs security checks, sends ACKs to gateways.

↳ ROUTING of packets to APPL. Servers

LORAWAN NODE ARCHITECTURE.

APPLICATION		
LORAWAN MAC		
CLASS A	CLASS B	CLASS C
LORA MODULATION		
PHY - ISM Band		

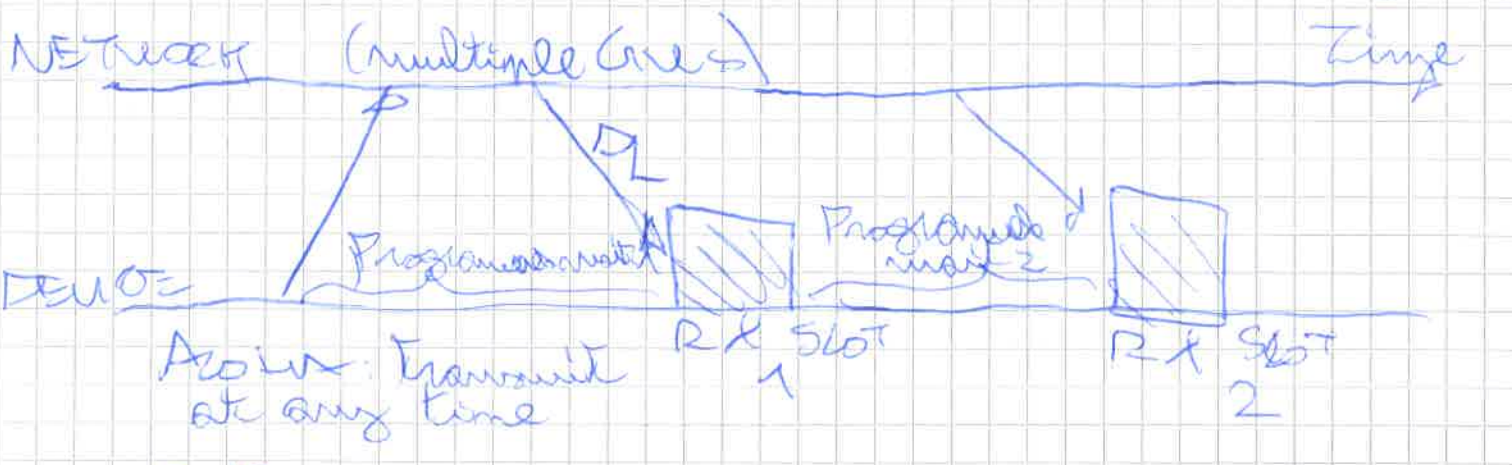
DEVICE TYPES.

CLASS A: Energy-efficient, for battery-powered devices.

CLASS B: Scheduled receive slots (BEACONS from GW for SYNC)

CLASS C: Maximum receive slots

CLASS A: (only close RX when doing TX)



=> For each UPLINK, 2 DLINK slots lim

BUSINESS MODEL: PROPRIETARY Protocol € Semtech (©)
RADIOS Only manufactured by SEMTECH

"Open ECOSYSTEM, but closed ELEMENTS"

NOISE MITIGATION

SIGFOX — Tiny slice of SPECTRUM

Bandwidth: NB | UNB (Ultra-Narrow Band)

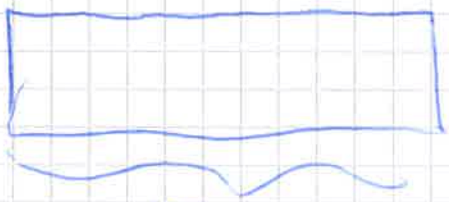
Modulation: BPSK (little spectrum used)

Frequency: 868 MHz [EU]

Payload: 12 bytes

Throughput: 100 bps

} Very energy-efficient

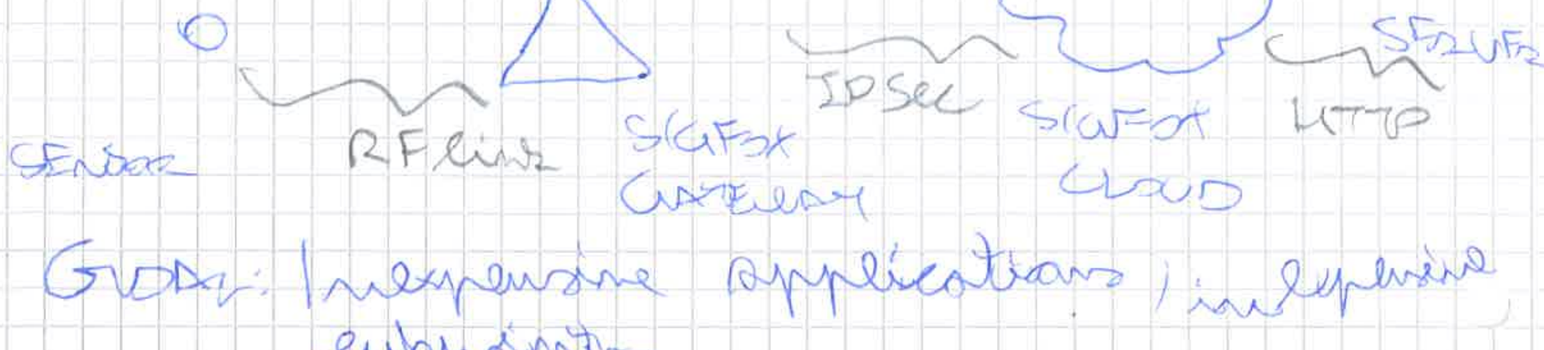
PACKET: 

PAYLOAD - 12 bytes

RANGE: 30-50 kms [RURAL] (long-lasting batteries)

3-10 kms [URBAN]

ARCHITECTURE



GOAL: Inexpensive applications, inexpensive endpoints.

BUSINESS MODEL: "Open ecosystem, open elements"

Any manufacturer can produce devices, as long as LICENSE TERMS respected.

+ ROYALTIES from network operators to sell TECH. STACK to customers

NB-IOT: Narrow Band IOT

Fully LTE-based (LTE Cat NB1)

LPWA - Low-Power Wide Area TECH. that connects devices on pre-existing LTE networks.

INFREQUENT 2-way data.

[2016 - 3GPP standardization in Release 13]

NB: LTE reuses existing LTE infrastructure!

Frequency Bands: (LTE ones): 700 | 800 | 900 MHz - 1.8 | 2.1 | 2.6 GHz

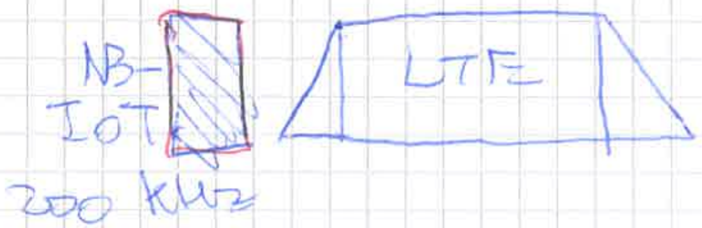
Bandwidth: 180 KHz.

DRATE: 200 Kbps UPRATE: 144 Kbps

Battery life: up to 15 years!
(Ex: Smart meter)

3 DEPLOYMENT SCENARIOS:

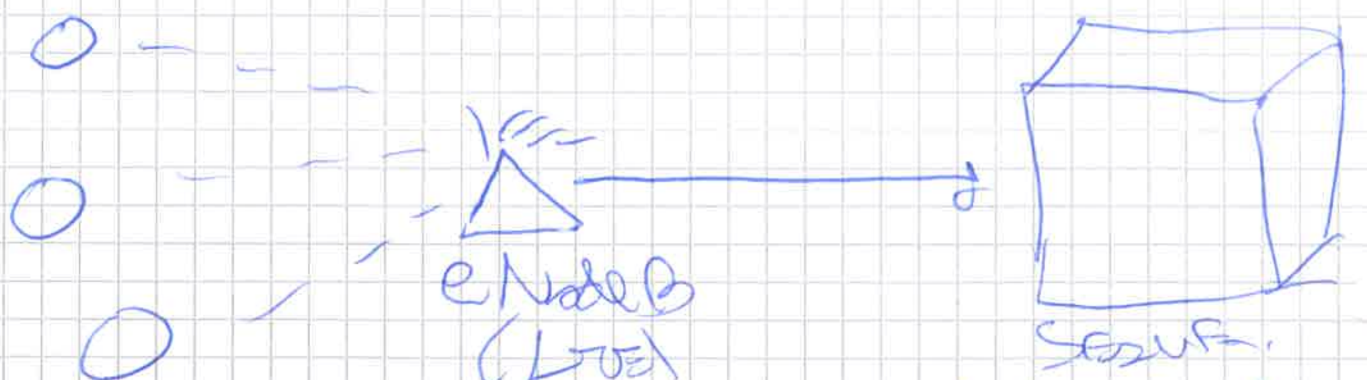
1 - STANDALONE: ~~RE~~USE unused ~~existing~~ GSM channels [200 KHz]



2 - GUARD-BAND: Use FOSSE LTE Resource Blocks, not used in data traffic.

3 - IN-BAND: Use internal LTE resource blocks for data traffic

ARCHITECTURE:



Since 2016 => Standardization [Infrequent data exchange]

LECTURE 5 - WIRELESS COMMUNICATION NETWORKS

NETWORK LAYER

IPv6 with compressed header

INTRODUCTION TO 6LOWPAN:

Part 8

6LOWPAN: Standardizing STACK for IoT

CoAP	APPLICATION	IETF
UDP	TRANSPORT	IETF
IPv6 LOWPAN	NETWORK	→ Reliability, flow control
" "	MAC	IEEE → Addressing & data transfer
IEEE 802.15.4	PHY	IEEE

IPv0-3: Development & experimenting

IPv4: 32-BITS. Used to identify a computer on a network.

$$\underbrace{xxxx\ xxxx}_{8\ BITS} . \underbrace{xxxx\ xxxx}_{8\ BITS} . \underbrace{xxxx\ xxxx}_{8\ BITS} . \underbrace{xxxx\ xxxx}_{8\ BITS}$$

$2^{32} \simeq 4.3$ B different addresses

ISSUE: $\boxed{\text{IPs' EXHAUSTION!}}$

⇓

In the IoT world, each device (things) is going to have its own address, they addressable from anywhere in the world ⇒ Too many devices (IoT devices' explosion) for just 4.3 B addresses!

IPv5: Development / experimental version.

IPv6: Next-generation Internet Protocol.

⇒ 128-BITS for identifying hosts

Simple routing & addressing

STANDARDISED NEIGHBOR DISCOVERY

IPV6 ADDRESSES $2^{128} \approx$ More than you could ever imagine!

NETWORK ID	INTERFACE ID

64-bit prefix (HIERARCHICAL) NETWORK ADDRESS

64-bit interface ID

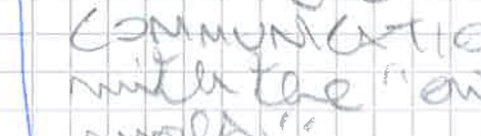
⇓

IID: uniquely identify a net. interface in a network

2^{64} HOSTS in a network

IPV6 ADDRESSING:

- UNICAST (1:1)



  - GLOBAL UNICAST
    communicate with a single host on a different link
  - LINK LOCAL (ADDRESS)
    communicate with devices on the same link (not in ROUTING TABLE)
  - ROUTER'S LINK LOCAL ADDRESS used by hosts as default gateway address

Used for COMMUNICATION with the "outer world"

- MULTICAST 1:N

(Ex. communicate with multiple Netflix servers)

- ANYCAST: One to any of a set of destination nodes

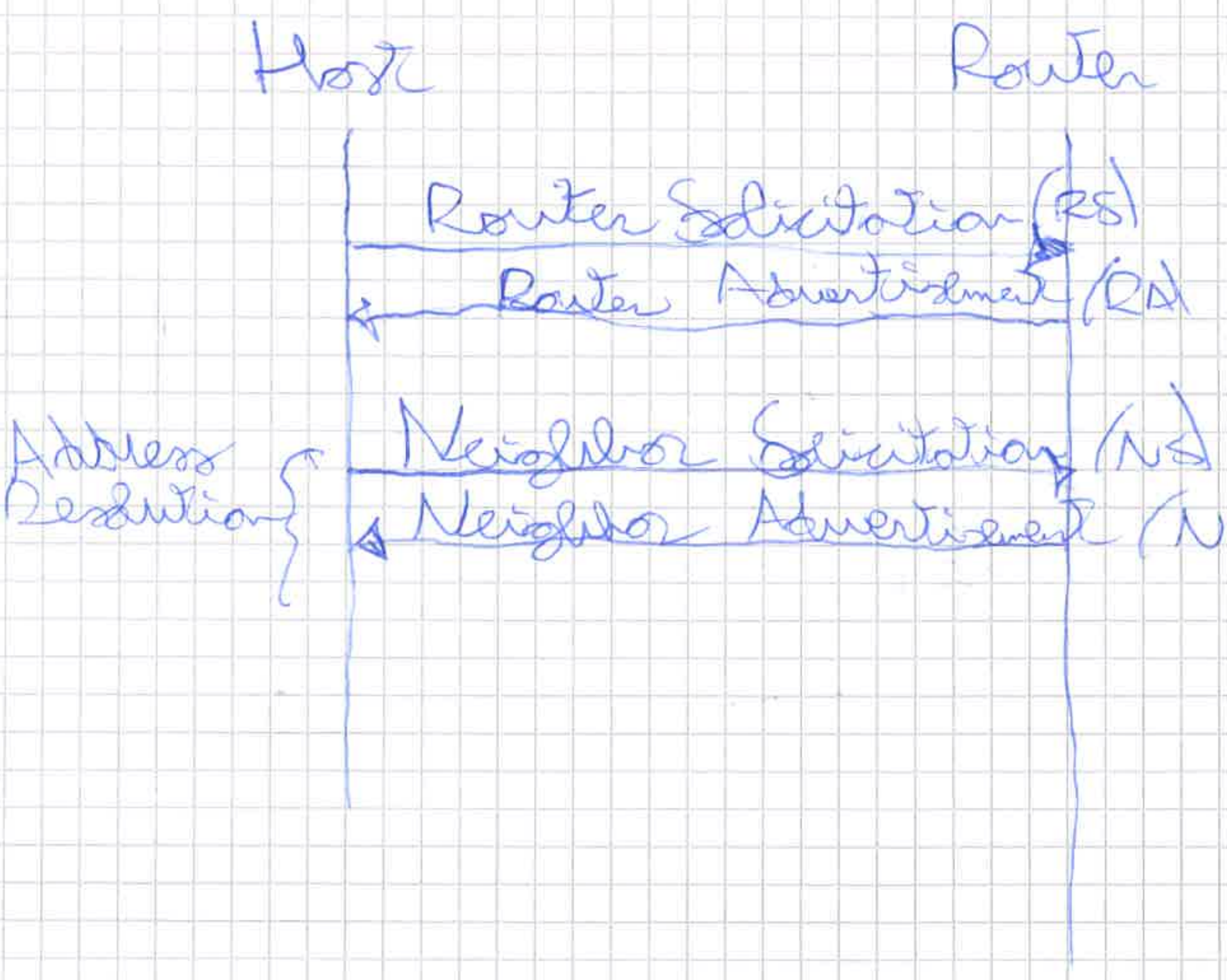
IPv6 NEIGHBOR DISCOVERY (ND)

Used to define the INTERFACE among neighbors. It is usable for:

- FINDING NEIGHBORS: Via Neighbor Solicitation (NS) | Neighbor ~~Acknowledgment~~ Advertisement (NA)
- FINDING ROUTERS: Via Router Solicitation (RS) | Router ~~Acknowledgment~~ Advertisement (RA)
- ADDRESS RESOLUTION: Detect duplicate address | neighbors' unreachability

⇒ $\underbrace{\text{DHCPv6 + ND}}_{\text{They can also be combined}}$

PROCEDURE OVERVIEW:



ISSUE: Most Internet services still run IPv4.

→ Allow IPv6 and IPv4 to co-exist via:

- TUNNELING (IPv6 packets placed inside IPv4)
- DUAL STACK ROUTERS (IPv4 + IPv6)
- NAT PROTOCOL TRANSLATION (IPv6 address ↳ converted to IPv4 address)

~~IPv6 over IEEE 802.15.4.~~

IPv6 PACKET SIZE OVER IEEE 802.15.4.

↳ Perform IPv6 ADDRESS ELISION & COMPRESS other FIELDS.

EDGE ROUTER: Performs 6LOWPAN → IPv6 Station

(based on different heuristics, such as distance, cost, least congested one)

ANYCAST can be used

6to4 IPv4 → IPv6 TRANSITIONING: Provide IPv6 compatibility via ANYCAST ADDRESSING IPv4 router

↳ DEFAULT GATEWAY used

IPv6 NEIGHBOUR DISCOVERY (ND)

IPv6 = format; ND = brain to discover peers

Defines interface between neighbours for:

- FINDING NEIGHBOURS (NS | NA)
- FINDING ROUTERS (RS | RA)
- ADDRESS RESOLUTION using NS|NA
  ⇒ Neighbour unreachability | duplicate address detection.

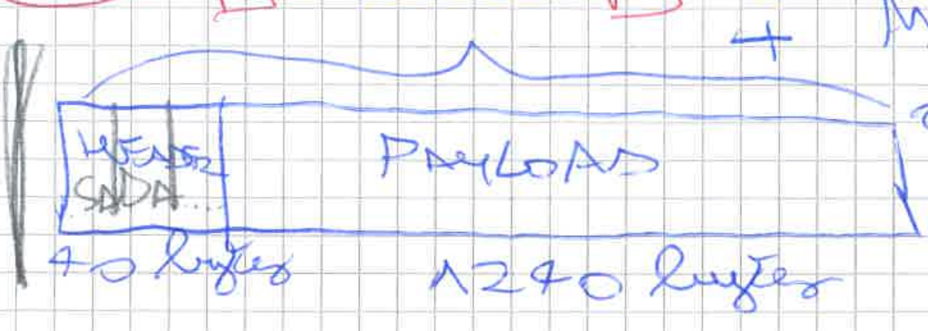
DHCPv6 + ND } can be jointly used

Still, low adoption of IPv6 by legacy devices (Tunneling routers | dual stack | NAT protocol translation)

IPv6 PACKET: Fixed 40-byte length HEADER (not a variable size)

+ Max 1280 bytes of PAYLOAD

MTU = Maximum Transmission Unit



~~IP~~ 6LoWPAN (Low-power IPv6)

IPv6 over low-power devices

Every single thing nowadays uniquely addressable via its own IPv6
⇒ MINIMAL MEMORY, CODE, POWER REQUIRED
⇒ IPv6 over LOW-POWER DEVICES = 6LoWPAN.

APPLICATION
UDP / ICMP
IPv6 + LoWPAN
IEEE 802.15.4 MAC
IEEE 802.15.4 PHY

ROUTER (first 3 layers)

- Minimal code / memory
- End-to-end internet integration
- Unicast / multicast / broadcast support

DIFFERENT 6LoWPAN ARCHITECTURES

They are STUB NETWORKS!

- SIMPLE LoWPAN (Single Border Router to go out)
- EXTENDED LoWPAN (common backbone links / multiple Edge Routers to go out)
- AD-HOC LoWPAN (No route outside the LoWPAN) → P2P

ISSUES of 6LoWPAN:

- Support for 64-bit and 16-bit 802.15.4 addressing.
- NETWORK auto-config
- FRAGMENTATION (standard) — (127 BYTES for 802.15.4 frames) [1280 BYTES for IPv6 MTU]

Very unoptimized IPv6 over IEEE 802.15.4.

IPv6 HEADER	UDP HEADER	PAYLOAD
0	40 48	127

} DEFAULT EFFICIENCY with IEEE 802.15.4

≈ 42% throughput

IP HEADER COMPRESSION ($\overbrace{\text{IPHC}}^{\text{2 OCTETS}}$)

Stateless & flow-independent compression

⇒ COMPRESS IPv6 HEADER & COMPRESSED HEADER / MULTIHOP communication

$\boxed{\text{40 BYTES}}$ ⇒ $\boxed{\text{5 BYTES}}$

UNCOMPRESSED IPv6 HEADER

- 1 octet DISPATCH
- 1 octet LoWPAN IPHC
- 2 octets SOURCE ADDRESS
- 2 octets DESTINATION ADDRESS

FLAT ADDRESS SPACES, with unique MAC ADDRESS

⇒ Address is unique already at the MAC level (64-bits IEEE 802.15.4 MAC address) in a network

IPv6 PREFIX ELISION

If all nodes in a network know the GLOBAL NETWORK ID, then the PREFIX can be elided

(Ex: nodes lie in the same network area)

~~NETWORK ID~~	IID
	64 bits

⇓

Can be further compressed, like multicast addresses.

~~MINIMAL OVERHEAD in 6LOWPAN:~~

~~108 BYTES~~

~~127 BYTES~~

MINIMAL OVERHEAD: No IPv6 HEADER, compression & 16-bit addressing [108 BYTES payload, 19 BYTES header]

ROUTER: translate 6LOWPAN into IPV6

NETWORK AUTO-CONFIGURATION in 6LOWPAN

4 STEPS for 6LOWPAN'S SETUP:

(LL) MAC LAYER INIT

1. LL COMMISSIONING: setting up (LL) link-layer connectivity between nodes => CHANNEL, MAC MODE for IEEE 802.15.4 [MAC & PHY]

NETWORK LAYER INIT

2. BOOTSTRAPPING: network layer (NL) address configuration / neighbors' discovery (No standard ND, but low-power wireless sensor nodes)
3. ROUTE INITIALIZATION: Set up paths in routing algorithms
4. GO TO 1.

FRAGMENTATION: IPV6 ⇄ 6LOWPAN

IPV6: MTU ≥ 1280 bytes!

IEEE 802.15.4: 80-100 bytes payload

Leads to poor PERFORMANCE over low-power wireless mesh networks.

$$P_L = 1 - (1-p)^n$$

$p = P\{\text{Packet is lost}\}$ (fragment)

$n = \#$ FRAGMENTS

If a fragment is lost, the whole packet needs to be re-transmitted

=> Reduce # FRAGMENTS as much as possible at the ~~network~~ ~~application level~~ application level

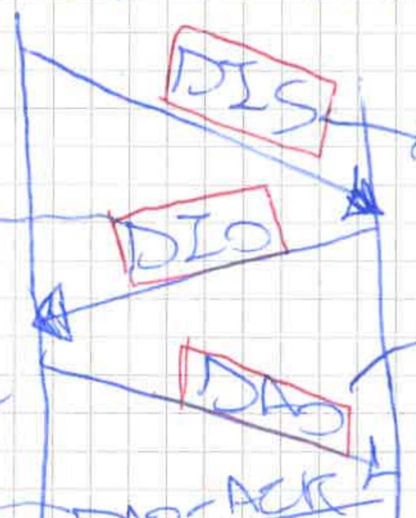
(LOW-POWER)

RPL - ROUTING {IPv6 Routing Protocol for LLNs}

DODAG = Destination Oriented Directed Acyclic Graph formed in a NETWORK

Allows generation of ROUTES according to an OBS. FUNCTION (Ex. Least [illegible])

(DODAGS maintained by the SINK NODE



DODAG Information Solicitation: Allows other nodes to discover an RPL instance & join it (request for DIO)

DODAG Information Object

Link-local multicast => Allows nodes to join it

Destination Advertisement Object

"Can I join you as a child on DODAG #X?"

YES / NOT response

UDP TRANSPORT LAYER

"Best effort" | fast & minimal overhead

Unreliable, connectionless, low processing

Few messages exchanged & minimal overhead required!

[Good for Low computational capabilities & battery-powered devices]

UDP HEADER COMPRESSION: Only 61616 – 61632 Ports (4 bits)

Checksum may also "go".

4 BYTES for the whole compressed UDP header.

1	1	2
UDP-HC	PORTS	CHECKSUM

NON-COMPRESSED UDP HEADER
20 BYTES

6LOWPAN – NEIGHBOUR DISCOVERY:

~~6LO~~ IPv6's ND (Node Discovery) is NOT optimized for 6LoWPAN

⇒ (• Always ON nodes /
• Massive MULTICAST traffic

⇒ Own OPTIMISED ND (Neighbour Discovery) in 6LoWPAN for low-power wireless devices (Minimised control traffic for ENERGY SAVING)

LECTURE 6 - WIRELESS COMMUNICATION NETWORKS

6

APPLICATION LAYER & INTEROPERABILITY

PHYSICAL INTERNET: Application of packet - switched internet principles to the field of LOGISTICS (Ex: Water monitoring) (Ex: tap monitoring)

PLATFORM for CONTAINERS' MONITORING (Procter & Gamble), powered by IoT devices that provide information about temperature, position or other data.

⇒ DATA needs to be distributed to all entities involved.

⇓

Need for INTEROPERABILITY:

NAME: Possibility to connect systems in a SEAMLESS & TRANSPARENT manner to achieve a common goal.

LEVELS of INTEROPERABILITY:

- TECHNICAL: Low - level communication among multiple devices (entities)
- SYNTACTICAL: How data is formatted
- SEMANTICAL: What data lies in the message exchanged [UNIVERSAL DATA MEANING]

ONTOLOGY: Representation of knowledge.

- ORGANIZATIONAL INTEROPERABILITY:
multiple organizations need to manage to communicate with one another
(Ex: BLOCKCHAIN for logistics, ensure data integrity by multiple parties)

- PLATFORM INTEROPERABILITY: Delegation of technical, syntactical, semantical interoperability to a ~~f~~ different platform.
platform needs to interoperate with other platforms

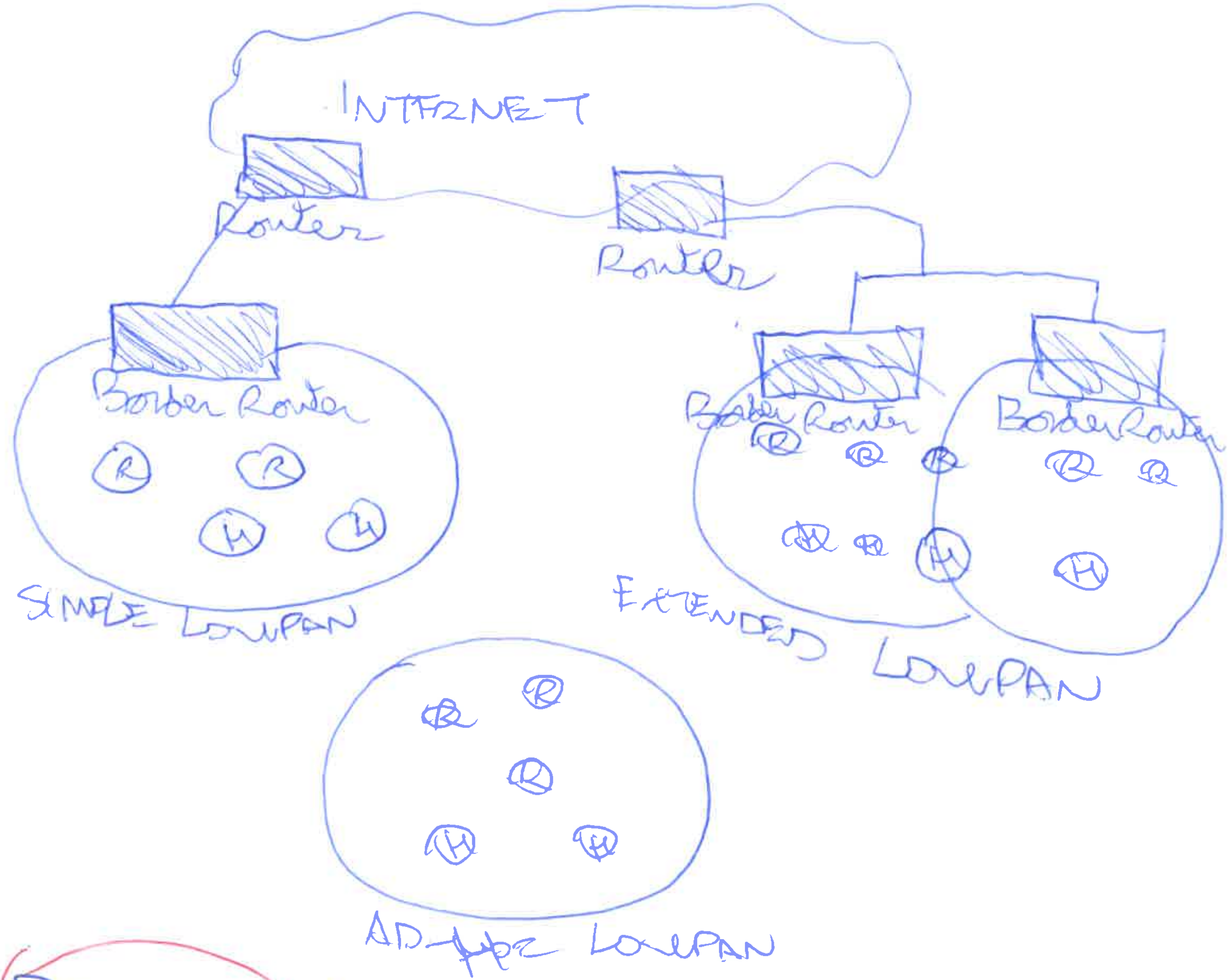
REST - APPLICATION LAYER
Allows to "surf" among devices in a network or multiple networks
⇒ Everything is a RESOURCE, devices can be accessed via HTTP.
⇒ Retrieve sensor data
⇒ Control actuators
⇒ Set and retrieve info. about a node.
(Ex: HTTP/COAP)

WEB SERVER:
Software designed to serve web pages/ web sites/ web services
(Ex: Apache2)

WEB SERVICE:
Method to achieve interoperable M2M interaction over a network
(set of web APIs)

LOWPANS: Consist of STUB NETWORKS, having either:

- Multiple Border Routers with a common backbone link (EXTENDED)
- One single Border Router (SIMPLE)
- No route outside the LOWPAN (AD-HOC)



DODAGs allows for the generation of routes according to an "OBJECTIVE FUNCTION"

OF1: Reduce energy consumption (minimize #HOPS)

OF2: Reduce the packet loss

(maximize #hops)

⇒ Multiple DODAGs allow for the establishment & support for redundant PATHS.

~~REST~~: Running on top of HTTP

REST (2000) Representational State Transfer

- GET
- POST
- PUT
- DELETE
- OBSERVE

Fits very well the IoT world, however not real-time functioning, but ⇒ BEST EFFORT.

OBSERVE: GET updates from publisher [PUSH MODEL]

Uniform interface between clients and servers

→ RESOURCE = Temporally varying membership function $M_R(t)$, which maps to a set of entities at time t.

NB: The mapping's semantics distinguishes a resource from another one pro it needs to be static (Ex: IMAGE is one area kind of mapping)

STATE of a RESOURCE = Current value of a resource at time t.

$S(t)$ State at time t.

(SYNTACTIC + SEMANTIC representation of a resource)

REST is based on 4 PRINCIPLES:

1. IDENTIFICATION of RESOURCES
2. MANIPULATION of RESOURCES through REPRESENTATIONS
3. SELF-DESCRIPTIVE MESSAGES
4. HYPERMEDIA as engine of APPLICATION STATE

URI: Way to UNIQUELY identify a resource in a network.

MANIPULATION through ~~RESOURCES~~ REPRESENTATIONS:

Representation ≠ resource

⇒ We send representations from a client to a server to manipulate the representation of a RESOURCE.

RESTFUL verbs are used to achieve the INTERACTIVITY of HYPERMEDIA & web services.

POST: Stateful interaction.

GET: Non-stateful interaction.

CHARACTERISTICS: (Stateless) Code "on demand".

- SCALABLE
- INDEPENDENT DEPLOYMENT
- INDEPENDENT COMPONENTS' DEPLOYMENT
- PROXIES to REDUCE LATENCY: Gateway

COAP: Constrained Application Protocol

resource-constrained

Allows different systems in the same network to interact with one another.

APPLICATION LAYER for IOT DEVICES
RESTful

"Lightweight HTTP model for constrained devices" ⇒ UDP-BASED

DATAGRAM COMMUNICATION

GOAL of COAP: Interoperability among different devices in the same NETWORK.

"Take a node & turn it into a RESOURCE PROVIDER" based on UDP

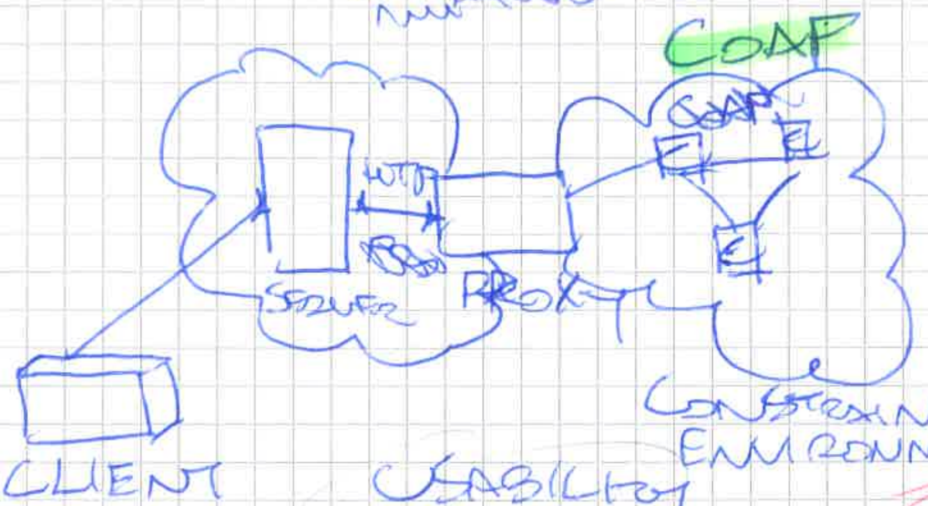
CoAP is an alternative to HTTP on resource constrained devices

It Provides the PUSH MODEL [PUBLISHER + SUBSCRIBER] (Send sensed data over)

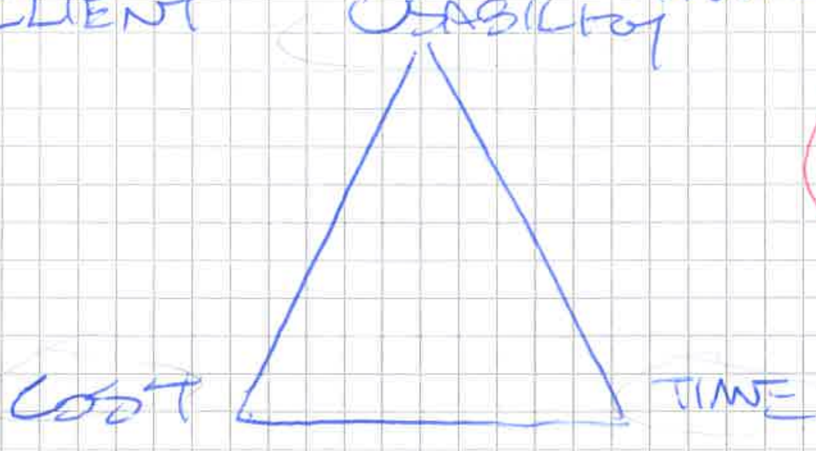
CoAP $\xrightarrow{\text{mapping}}$ HTTP

CoAP defines a simple (+ with queries) MAPPING for realizing the same API over HTTP for CoAP & proxying between CoAP & HTTP.

→ BUS-BASED PROTOCOL for interoperability with the web with internet [DIRECT COMMUNICATION, NO BROKER]



PROJECT'S TRIANGLE



MESSAGES' FORMAT EXCHANGED:

~~XML~~ ~~EMS~~

Efficient XML Interchange = EXI

ETSI 4 standard

Alternative to JSON and XML, as recommended by ETSI for M2M TECHNICAL SPECIFICATIONS.

- [OPTIONAL] schema information
- Avoid cost of parsing (XML-less)

~~XML~~ EXI $\overset{\text{better}}{>}$ XML !

97% COMPRESSION RATE wrt. XML.

EXI
CoAP

Competitor of CoAP

MQTT: Message Queueing Telemetry Transport

(IBM)

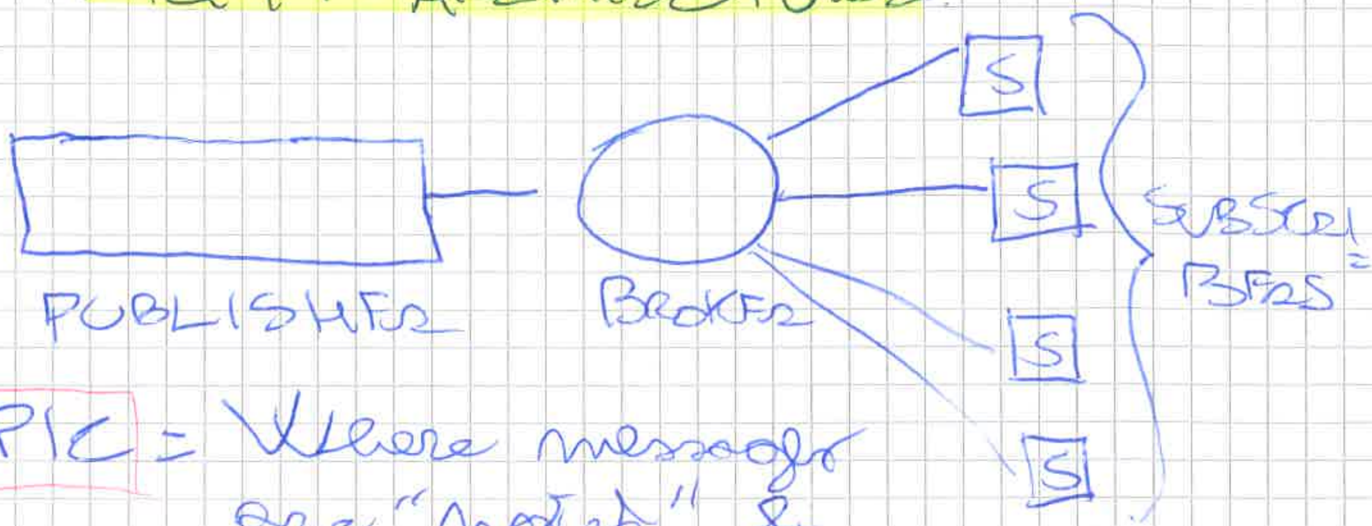
⇒ Alternative to CoAP

~~Broker~~ Publish | Subscribe messaging protocol for low-computing capacities devices over unreliable narrow-bandwidth networks.

TCP-based protocol & message-oriented

↳ LOTS of ENERGY CONSUMED

MQTT ARCHITECTURE:



TOPIC = Where messages are "posted" & accessible by clients subscribing to it. (long strings, which may be an overhead for IOT PROTOCOLS' payloads)

IMPRACTICAL for IEEE 802.15.4

TCP / ~~IP~~ ≠ No good fit for the IOT WORLD

~~MQ~~ MQTT for IOT:

- Run MQTT over UDP (perform a mapping)
- Broker support for indexing long topic names

Generally, lives in a SN GATEWAY
SN = Sensor Network

[performs MQTT-SN translation to MQTT]

Gateway is also accessible via a FORWARDER (in case no direct network connection)

MQTT vs CoAP

MQTT	CoAP
[PUBLISHER-SUBSCRIBER MODEL]	[STATE TRANSFER]
N : 1	1 : 1
+ Decouple producer from the consumer	+ "OBSERVE" method support
+ Support for persistence (LIVE DATA)	+ State transfer model
+ BROKER for one extra level of INDIRECTION	(NO BROKER direct connection)
[TCP-based for connecting to a BROKER]	[UDP-based] integration with HTTP
+ MESSAGE FORMAT must be known up front	+ SUPPORT for DEVICE DISCOVERY

Which one? "IT depends"!

RESEARCH in IoT:

- USER CONNECTED 2 A USER lront "Responsiveness"
- TRADE-OFF among "liveness" and "efficiency" & ~~data~~ data sent.
- ENERGY-EFFICIENT~~CY~~ in battery-powered IoT devices (SENSOR-side, GATEWAY-side, CLOUD-side → LoRA
  multi-MAC
  multi-MAC
- INTEROPERABILITY among different standards
  sensor & devices (where?)
- COMPLEX & DISTRIBUTED LOGIC for efficient processing

[EDGE COMPUTING & IN-NETWORK DATA PROCESSING]

= LAYER-3 SECURITY [IP SEC] → AES-128

LECTURE 7 - CONTIKI OS

A WSN (Wireless Sensor Network) is made up of a set of SENSOR NODES

EMBEDDED ELECTRONIC DEVICE, each one having sensors, a microcontroller, limited memory, a battery, RX/TX antennas.

OS for WSN:

A sensor node is also equipped with an OS.

OS: Interface between HARDWARE and PROGRAMMER, hiding low-level implementation details.

RESPONSIBILITIES of AN OS:

It manages components' functioning of the underlying HARDWARE (non-functional characteristics) such as:

- DRIVERS for ANTENNAS & SENSORS
- SCHEDULING of PROCESSES / THREADS
- NETWORK STACK
- POWER MANAGEMENT

(EXAMPLE OSs for EMBEDDED SYSTEMS: Contiki, Tiny OS, Mantis OS, Nano-RK)

CONTIKI OS

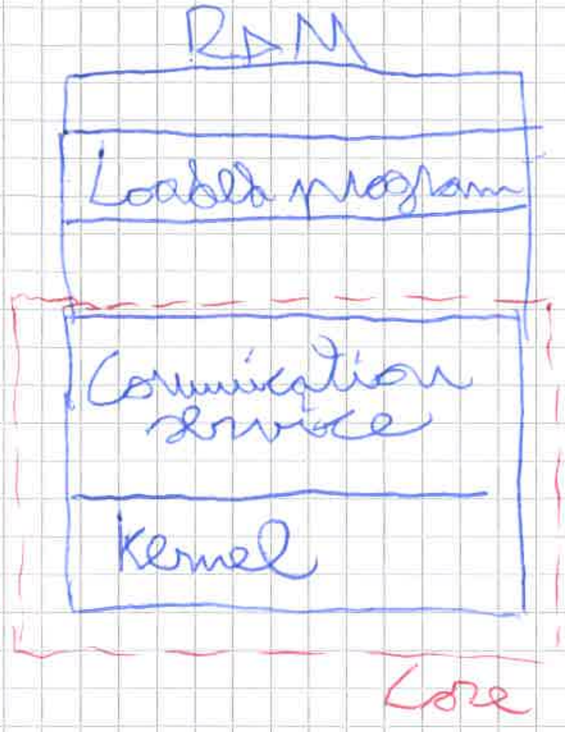
WHAT: Dynamic OS for networks of embedded systems.

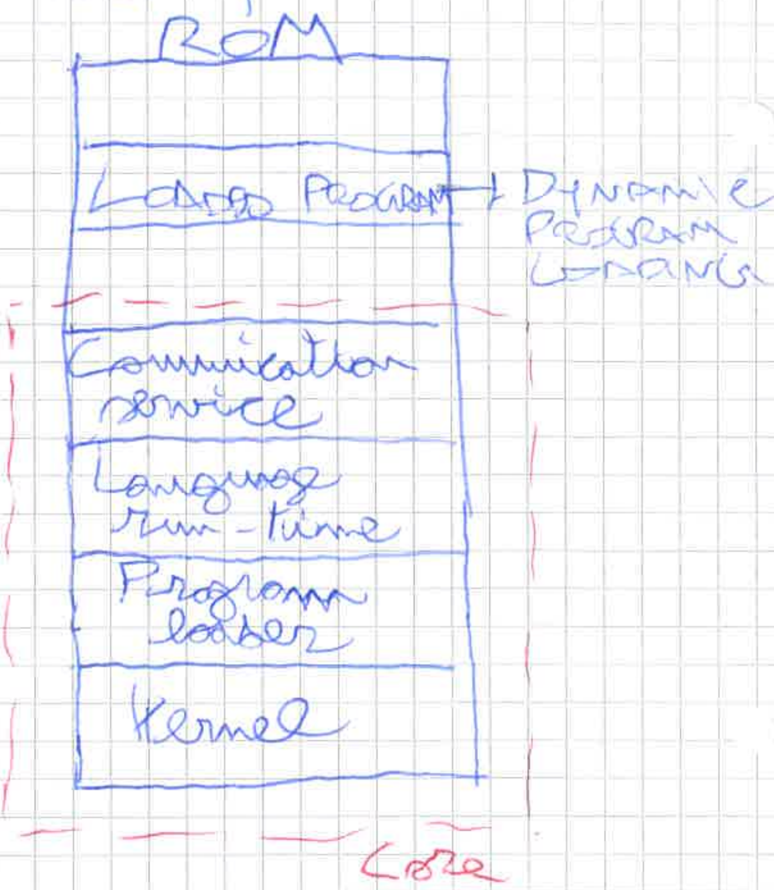
CHARACTERISTICS: (C-written)

- RAM / ROM: Small footprint
- NETWORK: TCP/IP [6LowPAN] IPv4 & IPv6 ⇒ uIP
- PORTABLE
- EVENT-DRIVEN KERNEL
- PROTOTHREADS
- IPSEC (On-node DB)

(COOJA TOOL: Simulate WSN)

CONTIKI SYSTEM OVERVIEW:





PROTOTHREADS: (THREAD-like functioning)

PROGRAM is made up of a set of EVENT HANDLERS.

EXTERNAL EVENT
⇓
EVENT HANDLER
(perform an action, then return to the caller)

} COMPLEX PROGRAM FLOW & COMPLEX STATE MACHINES

(EX: Sensors & GUIs' triggering events, handled by specific event handlers)

PROTOTHREAD-VIEW: A protothread is a SET of macros to simplify the management of state machines
(THREAD-STYLE PROGRAMMING)
sequential program flow

EXAMPLE PROTOTHREAD:

```
PROCESS_THREAD_BEGIN();
do {
  .
  .
  .
} while (!ack_received());
PROCESS_THREAD_END();
```

(Ex: MAC PROTOCOL, create an FSM [FINITE STATE MACHINE] that alternates "sleep" periods to "awake" periods ⇒ COMPLEX MODELLING

& LOCAL CONTINUATION (Line where function stopped) (Ex: Markov Chain)

Protothread blocked (busy waiting)

⇒ Maintain its status, but not its CONTEXT.

It can be set in a specific function to capture the state of the function. LOCAL VARIABLES

(RESUMABLE later and to restore the state of the function)

⇒ Sort of "CONTEXT SWITCH" among functions to save current THREAD'S content (but NOT local variables in a function)

PROCESSES.

What:

A PROCESS is a section of code, executed by Contiki & invoked by an event-driven scheduler.

A PROCESS STRUCTURE consists of 2 PARTS:

1. PCB - Process Control Block. Literally a "struct", containing information about the process & stored in RAM.

1) **PCB**

```
struct process {
    struct process *next;
    const char *name;
    :
    :
}
```

Declared as:

```
PROCESS{hello_world_process, "Hello world process"}
```

PCB
↳ Used by the kernel for SCHEDULING processes

2) **PROCESS THREAD**: Contains actual code executed when process is executed.

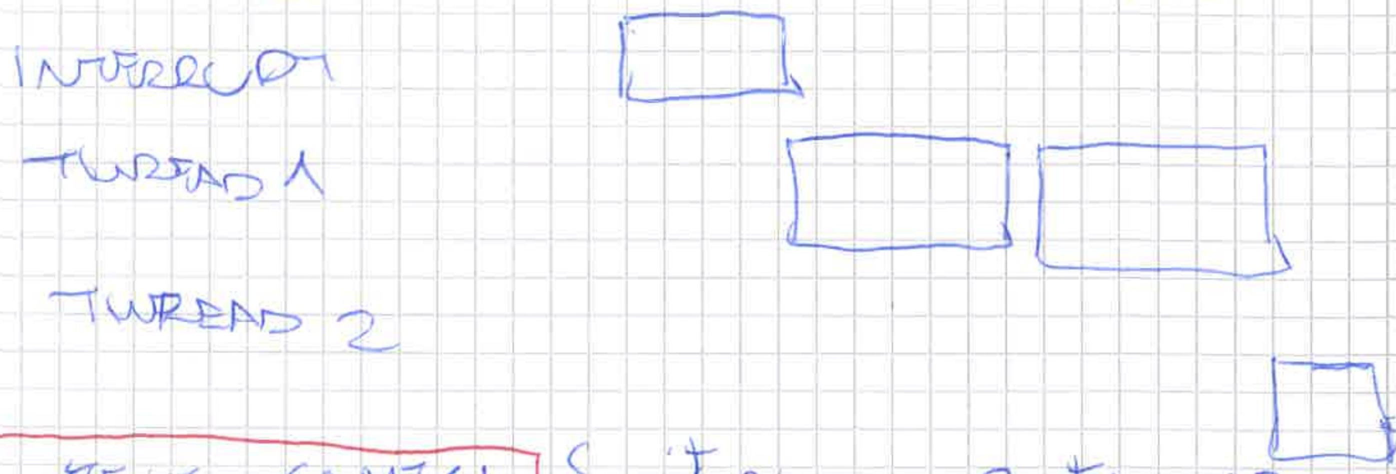
```
PROCESS_THREAD(hello_world_process, ev, data)
{
    PROCESS_BEGIN();
    printf("hello world \n");
    PROCESS_END();
}
```

SCHEDULING:

- **SEQUENTIAL SCHEDULING**: Single thread, no deadlines!

- **EVENT-DRIVEN SCHEDULING**: Handlers for external events.

• MULTITHREADED SCHEDULING: Implements scheduling and CONTEXT SWITCHING



CONTEXT SWITCH: Switching of the CPU from one thread to another one (according to a certain policy)
[All previous INFO. is saved] as "THREAD CONTEXT"

SCHEDULING: Activity of process manager, handling the removal of the running process from the CPU & selection of another process for execution (based on a certain strategy. Ex: LSJF)

In CONTIKI, we have two ways to execute THREADS:

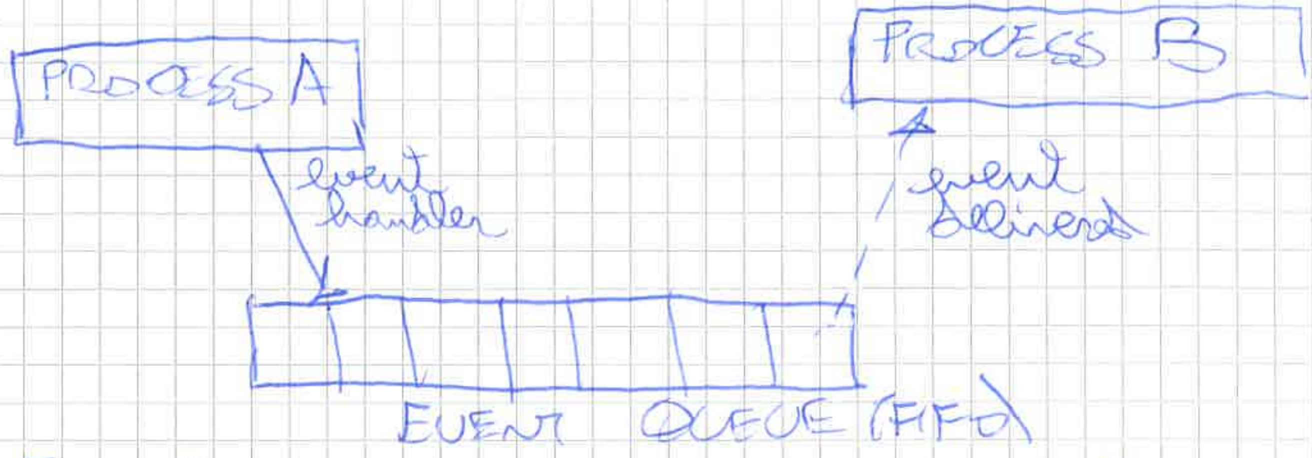
- COOPERATIVE: "Continuous" manner to execute threads
  [NO PREEMPTION UNTIL PROCESS IS COMPLETED]
- PREEMPTIVE: Stop execution of process in case an I/O interrupt occurs / timer.

  ⇓ REAL-TIME TIMER: Abstraction of the HW timer

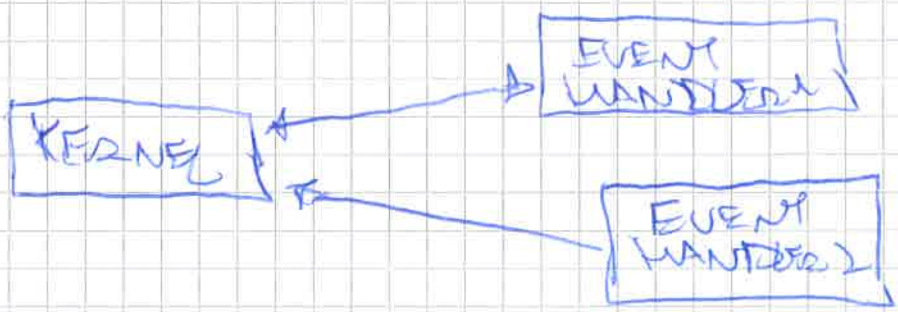
  ≠

  HW SCHEDULED.

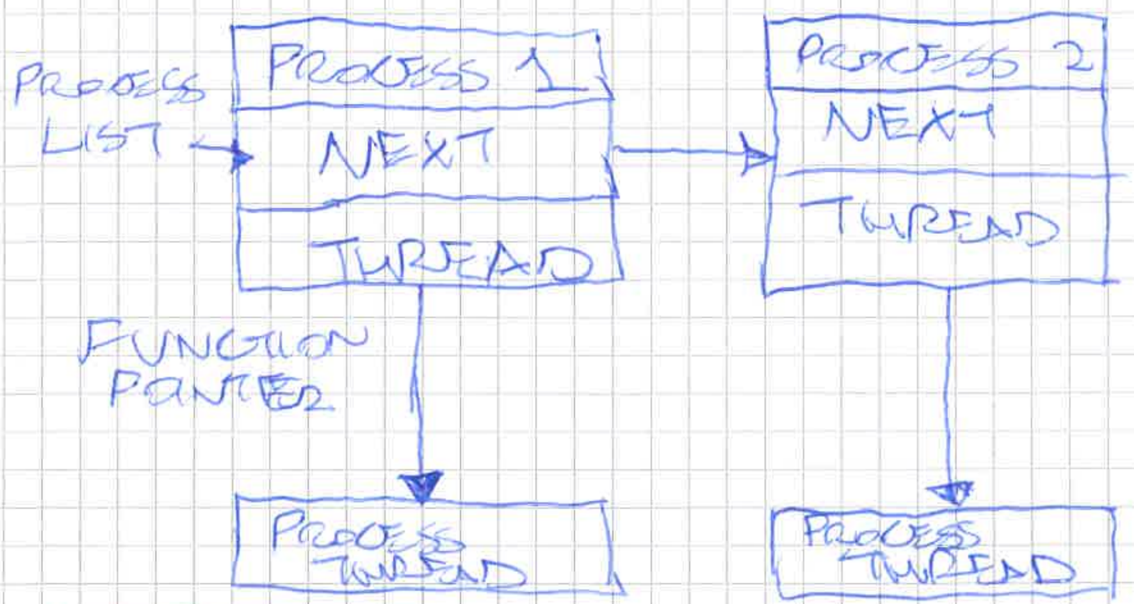
EVENT-DRIVEN MODEL FOR COOPERATIVE SCHEDULING



Events are placed in a QUEUE, from which they are extracted & delivered to a certain process.
(Run to completion)



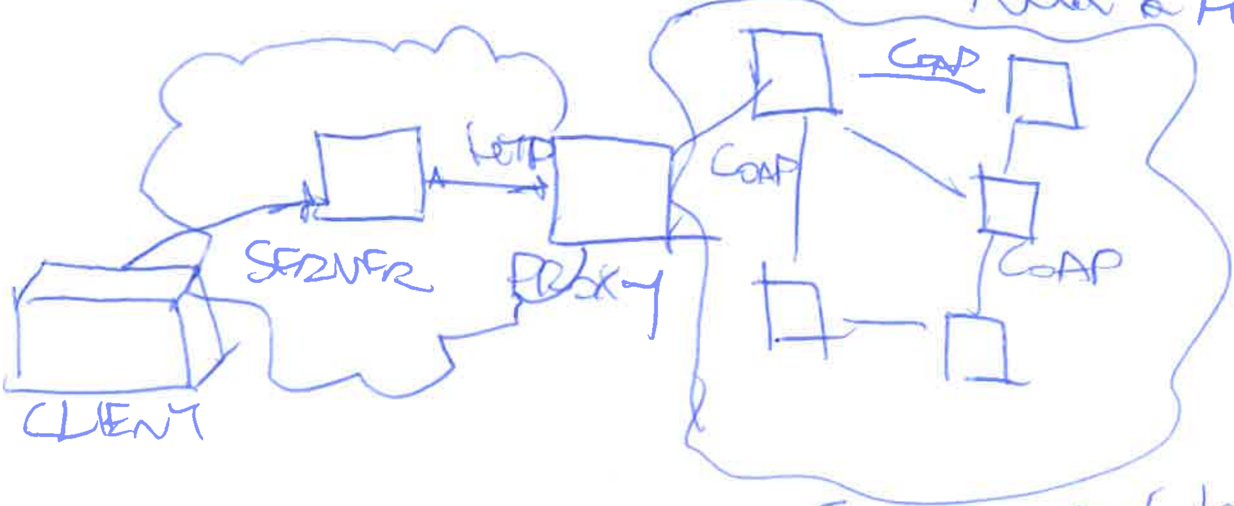
EXAMPLE EVENT-DRIVEN MODEL



EVENT TIMER Generate timed events to a certain process, when the event timer expires.
(Ex: SPADE / Hardware programming)

COAP NETWORK:

Analogously to 6LoWPAN, it operates within a "STUB" network to connect to the outer world via a PROXY.



COAP => Used in CONSTRAINED NETWORK

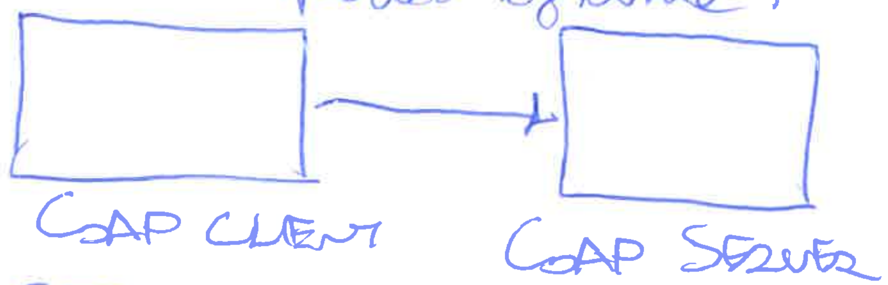
MQTT-SN: If a NAT is present, some initialization operation need to be carried out.

HTTP	COAP
TCP-based	UDP-based
4 VERBS	4 VERBS + 1 OBSERVE

$$\text{COAP} \xrightarrow[\text{mapped}]{\text{can be}} \text{HTTP}$$

=> Usage for M2M / IoT devices

OBSERVE KEY WORD: Used to OBSERVE a RESOURCE ~~CE~~ from an IoT (CoAP) client to a CoAP server & keep the representation updated by the server over a period of time.



BEST-EFFORT approach for receiving UPDATES

WSN: Set of SENSOR NODES that transmit the data measured by sensors to a base station, in a SINGLE-HOP or MULTI-HOP manner.

DUTY CYCLE: Fraction of one period in which a SIGNAL or SYSTEM is active.
(Can also be used to describe the percentage in which a system is active)

Ex: <1% DUTY CYCLE device means that the device is active only 1% of the time.

$D = \frac{T_{UP}}{T_{UP} + T_{down}} = \frac{T_{UP}}{T_{ALL}}$ ⟹ Only when up for TX

LOCAL CONTINUATION: "Mark" the line at which a function stopped for RESUMING execution later on.

⟹ Set in a specific function to capture the STATE of it.

After the local continuation has been set, it can be resumed in order to RESTORE the state of the function at the LINE where it stopped.

LC_BEGIN

LC_END

NETWORKING.

Contiki's NETSTACK supports both IPv4 and IPv6 NETWORKING.

7 ISO/OSI LAYERS are fully "covered".

APPLICATION	→ WEBSOCKETS | HTTP, CoAP
PRESENTATION	
SESSION	
TRANSPORT	→ UDP | TCP SOCKETS
ROUTING NETWORK	→ LOWPAN / (uIP TCP/IP)
MAC | RDC	→ CSMA DUTY CYCLING
PHYSICAL	→ RADIO LAYER

RDC - RADIO DUTY CYCLE (MAC).

Contiki MAC, Duty Cycle Radio for ASYNC COMMUNICATION.

SLEEPY RADIO, in which you "sleep" and wake up periodically for transmitting/receiving [ENERGY SAVING]

→ PHASE-LOCK: to understand when to wake up & reduce delay, we can "start slightly earlier than wake-up" ~~and~~ by LEARNING

CoAP: - APPL LAYER

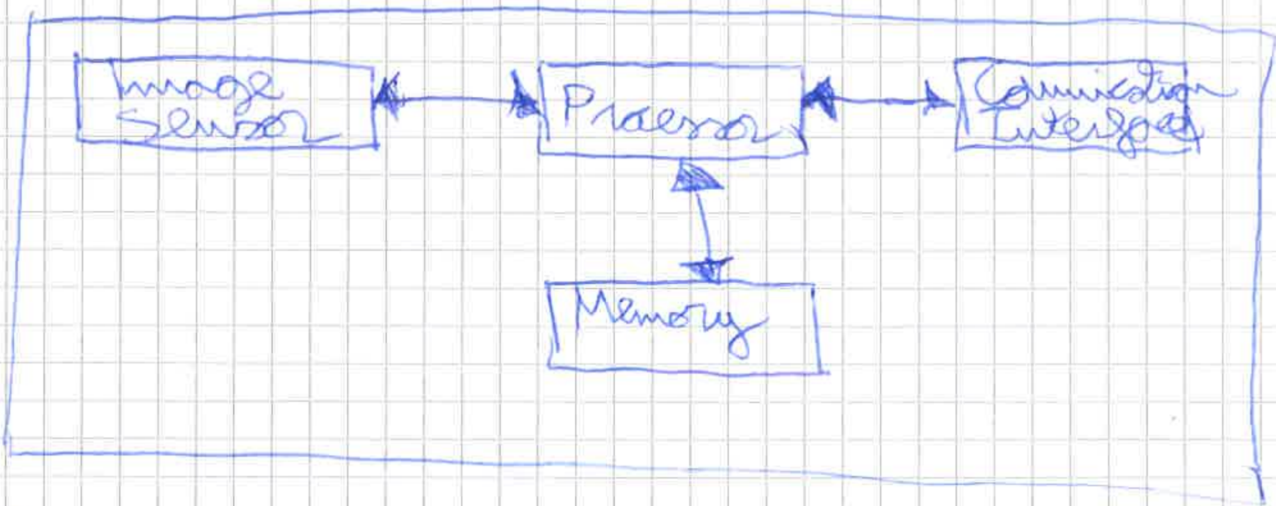
$\boxed{\text{RESOURCE}}$ --> URI PATH associated with resource handler.

Dinamic data → PERIODIC_HANDLER, Manage gathered from Emessage sent periodically.

EVENT RESOURCE: Activated when an event occurs (TRIGGERED) & EVENT HANDLER is invoked [Ex: mouseOn in Javascript]

PERIODIC RESOURCE: Used to dispatch data gathered from sensors in a PERIODIC manner.

SMART CAMERA ARCHITECTURE



Ex: NGS Smart Camera
⇒ FPGA for pixel-wise algorithms' processing

MICROCONTROLLER used for COMBINATORY LOGIC.