A 01 - IoT

Cyber-Phisical systems:

- Have **Sensors** and **actuators** to interact with **phisical** world.
- Are **cyber**: *processing*, *memory*, *connectivity* capability to act in the cyberspace
- IoT is an embodiment of CPS, CPS implement *smart environments* where *smart objects* are both *cyber* and *phisical*.
- **Phisical propetries:**
  1) - *free placement*: small size, different form factories and shells
  2) - *mobility*: wireless communication capability, battery-powered, position-awared
  3) - *exposed to the wild*: redundancy (allowed by low costs), security, low costs.

Smart Environments:

- Can be difined with a varity of charaterisitics based on:
  1) - application they serve
  2) - **interaction with humans**
  3) - pratical system design aspects
  4) - multi-faceted conceptual and algorithmic consideration that allows them to operate **seamlessly** and **unobtrusively**.
- are smart for *management*, *deployment*, *mantenience* and for the *final user* :
  1) - recognise: context, activities, situations
  2) - figure out: user needs at the right time
  3) - provide services

IoT:

- phisical objects embedded with:

  1. Electronic
  2. Software (business logic)
  3. Sensors/actuators
  4. Network connectivity

- each IoT device has:

  1. Sensors/actuators
  2. Microcontroller

3. Sotware (business logic)

- some features: of IoT devices

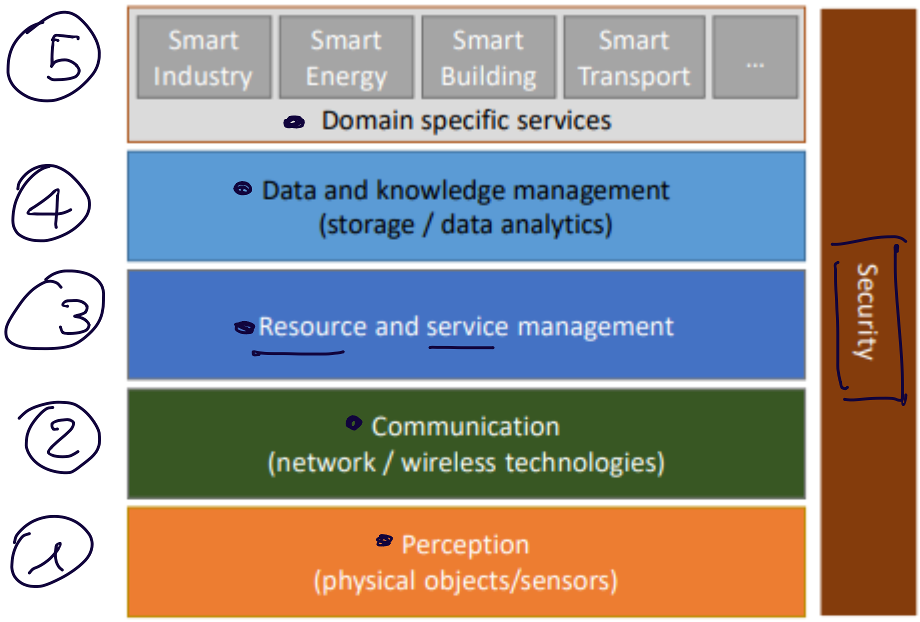
1. interconnect smart devices
2. interconnettivituy through the cloud:
3. communicate with each other and provide data (to the cloud → processing)
4. deliver informations from sensor
5. act on thier environment
6. low-powered, low-bandwith, low energy

IoT - Layered ARCHITECTURE

- a layered architecture:
  4. Domain specific services: *smart industry, city, energy, transport...*
  3. Data and knoweledge mngmnt: *storage / data analytics: data are stored, processed, presented in the cloud*
  2. Communication: *network / wireless technologies*

1. Perception: *sensors and actuators are at the edge of the cloud (phisical objs)*

2. Security: *referred to all the layers above*



PLATFORM for IoT

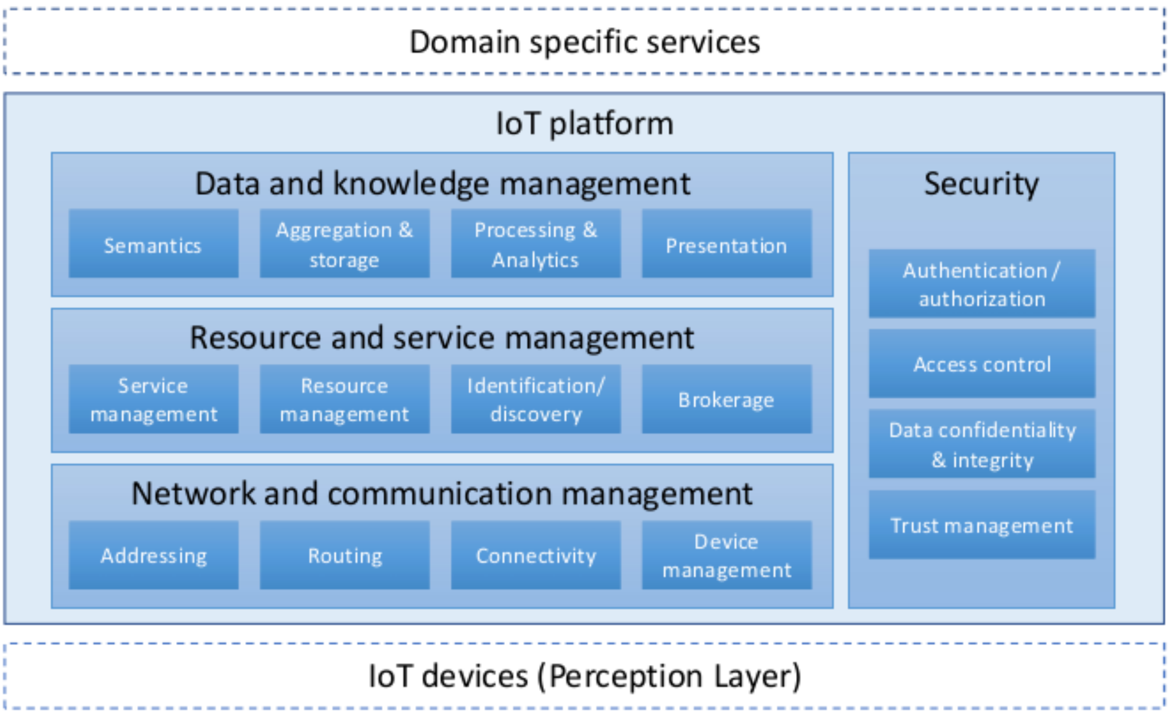
Sw → APPLICATIONS

IoT Platform ← Sw

Devices

- platform for IoT:
  - sw layer between IoT devices and applications
  - their functionalities (may be) distributed between devices themselves, gateways, servers in the cloud
  - *provides*:

1. **identification**: way to identify things in the platform, (IP on internet, URI on web, OID - Object Identifier)
2. **Discovery**: find devices, resources, services to abtain their properties/services/features
3. **device management**:
   - *initial settings*:
     - 1) pairing, security settings, key distribution
     - 2) configuration
     - 3) localisation of devices
   - *managing updates of sw and fw*
   - *device monitoring, logging, auditing*
   - *diagnostic*
   - *remote control*
4. **abstraction/virtualisation**: IoT devices as services
5. **semantics**: provide a way to represent IoT devices and their context
   - enables AI-processing over data
6. **service composition**: builds a composite service interactins services/microservices (like aggregator, data analytics and sons)
7. **Management of data flow**:
   - *sensors → appliction → actuators*
   - support for: *aggregation*, *processing*, *analytics*



- issues in IoT
  - performance
  - energy efficiency

- security
- data analytics/**processing**: informations may be contrasting → brings to make different decisions
- **communication**: how to bring data *producers* (sensors) with *consumers* (actuators/users/app)
- data representation
- **interoperability**: we need standards
- **latency**:
    - IoT is a layered architecture → have to bring data to the cloud, devices are at the edge → **data flowing into the cloud represent same reality from different POV**
    - a sensor provides few data, but they are interconnected → so much data to be processed
- **reliability**: if objects provides for their purpose

* IoT & ~~machine learning~~ AI   IoT & AI

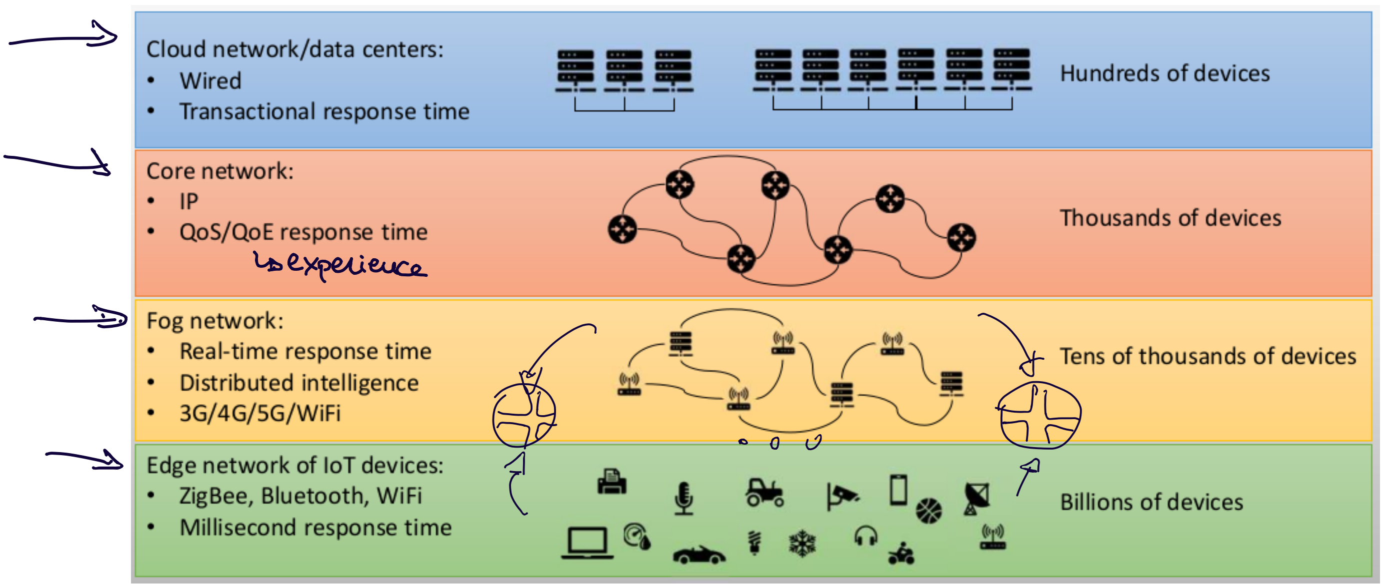
    - curated technology   NON VA BENE perché è ad alto livello per gli IoT
        - represent (inference) knoweledge and make reasons
        - propositional, predicate logics, production rules, semantics ntwrks
        - problems:
            - sensors output (what's goin' on?!): data are noisy, reduntant, missing, fast flowing..
            - can extract windows of data received but unuseful for curated technology

    - ML: learn by doing/reciving informations and testing
        - "*automatic systems that can learn from data*"
        - humans provide data,
        - system is fed by examples(*training set*, *testing sets*) to learn how to associate input and output
        - if well trained, output will be (most likely) correct
            - *Unsupervised learning*: analyse data, finds relationships among data points, good to understand the past (not really useful for IoT stuff)

- *SUPERVISED LEARNING*:
  - learn from *past examples*
  - for each example, requires input + desiderate output
  - aims at *predicateing future* or *understenting present*
  - useful when all data are avaliable
- REINFOIRCEMENT LEARNING:
  - learns from examples
  - if output is wrong → provide example → machine self configure itself
  - for each example only one input and reward (e.g: game, win +1, loose -1, otherwise 0)

- ML (supervised, reinfoircement) guarantee for IoT
  1. *Flexibility* (of analysed data)
     - *training phase* infers the ML *classifier* (universal approximation of any function)
  2. *Robusteness* (of analysed data): performance degrates proportionally to the degradation of the input
  3. *Customisability*:
     - improve/ maximise the accuracy given examples
     - reduce memory footprint of the classifier (for ebedding low-power devices)
  4. *Embedding AI into devices*

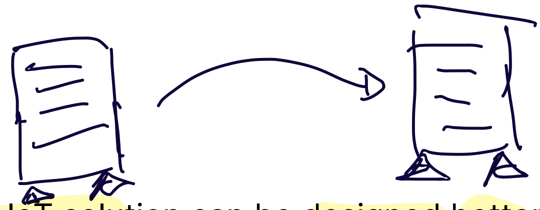
- IoT and Cloud IoT & Cloud
  -



- EDGE
  - At the **edge** is a *network of IoT-enabled* devices consisting of *sensors* and *actuators*
  - devices may *communicate with each other*
  - **cluser of sensors** may *trasmit* thier *data to one device that aggrates* the data to be *collected* by an higher-level entity
  - a **gateway** interconnets IoT enabled devices with the higher level network
- FOG
  - *massive amount of data* ma be generate by a **distributed network of sensors**
  - *Rather than store all data* permanentely in *central storage*, is desirable to do as much *data processing close to the sensors* as possible
  - processing may deal with much data, perform *data transformation* operations, resulting in the storage *of much lower volume of data*
  - fog operations: evaluation, formatting, expanding/decoding, reduction, assestment
  - fog devices are deployed *near the edge* (sensors and other data-generating devices)
  - Cloud vs Fog computing
    - Cloud: centralised storage, processing for a small number of users
    - Fog: distributed processing and storage resources close to massive number of IoT devices
- CORE
  - aka backbone ntwrk, *connects* geographically *dispersed fog ntwrks* providing *access to ntwrk outside the enterprise ntwrk*
  - uses *high performance routers, high capability trasmission lines, multiple interconnected routers* for increased redundancy and capacity
- **BLOCKCHAIN and IoT**
  - a *shared* and *trusted* public ledger for *making transactions*
  - everybody can *inspect* it
  - nobody can *controls* it
  - transaction cannnot be *altered*
  - involves businesses want to record history of transactions
  - BC implies a shif of paradigm fo IoT:
    - from centralised store to a *decentralised* one, in a distributed ledger
    - supports expanding of Iot ecosystem

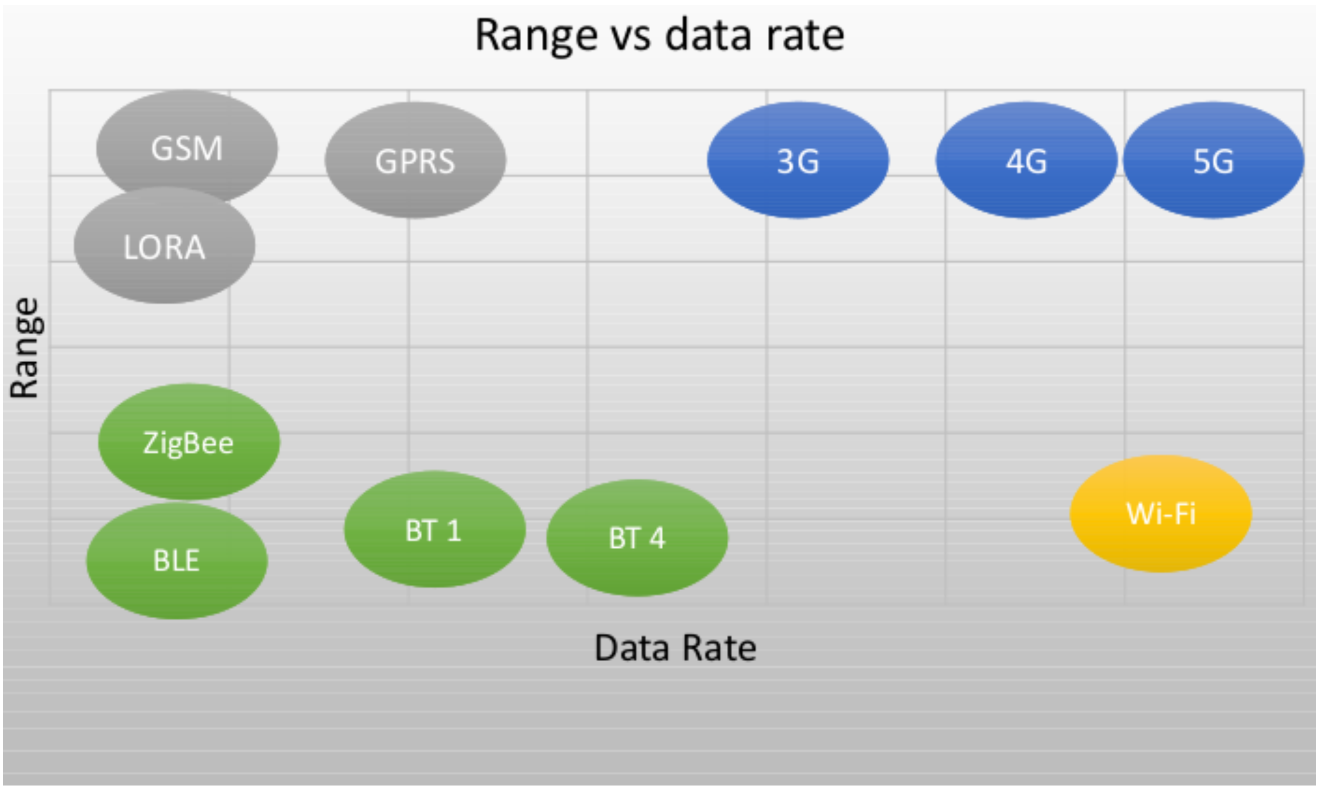
- reduces *maintenance* costs (distributed ledger is public)
- provides *trust* in data produced
- so it certifies steps in business process → BC as shared ledger between companies in supply chain
- *smart contracts to certify intermediate delivery of goods* (steps of business process, state of goods and services)

Interoperabilty and standards



- Interoperability force to have standards
- Verical silos: a straight implementation of an IoT solution can be designed bottom-up (from phisical to application layers) → vertical silos.
    - idea: keep content inside → no visibility from outside
    - one layer stacked solution: sw architectures have layers (of IoT) like silos
    - your solution will only *work alone*:
        - only your devices
        - any change/update rquires intervention of IoT fabric
        - other vendors *cannot interfere*
    - vendor lock-in: clients entrapped
    - no components from other vendors
    - force *high cost to migrate to another vendor*
        - POV of customer is no change vendor (move from one silos to another)

Wireless standards



- **IEEE 802.11 (Wi-Fi)**
  - **frequency**: 2.4 Ghz
  - **bit rate**: 1,2 Mbps
  - **trasmission range**: ~ 100 meters (2Mbps) - 130 meters (1 Mbps)
- **IEEE 802.11A, B, ... , AC, ...** (Evolution)
  - **frequency**: 5 Ghz
  - **bit rate**: up to 400 Mbps
  - **transmission range**: increased (**G**)(???magari avere qualche dato sarebbe bello)
  - **introduced technologies**:
    - QoS (**E**)
    - directional antennas (**N**)
    - roaming between access poins (**F**)
- **IEEE 802.15.4 (low power antennas) and ZigBee**
  - IEEE 802.15.4 defines both phisical and MAC layers (1,2)
  - ZigBee: industrial consortium promoting development of low-power sensor networks
    - defines also network(3) and application (5 || 7) level
  - **low power***:
    - low throughput (up to 115 kbps)
    - low duty cycle (around 1%)
  - **supports multi-hop deployments** → enables coverage of large areas
- **BLUETOOTH**
  - no authentication
  - higher data rate than ZigBee (?!?! servono dati)
  - personal and multimedia communicatio (audio, low quality video)
  - bluetooth2 increases
    - **throughput**: up to 10 Mbps

Standards in IoT

- require common *interests* and *agreements* among different stakeholderds

- motivated by reduction of costs (related to technology development)

- *coopetiotion* among different stakeholders

- happens when a technology is mature

- big revenues are somwhere else
- no interest put big money in developing technology

- but problem is not solved:
  - competing consortium for standards
  - developement is a competiotion for upper layers
    [x] domain specific services smart city / smart industry) smart transport...
    [x] data and knoweledge management data storage / data analytics
    [x] resource and service management
    [x] communication wireless capabilities) networks
    [ ] perception (sensors/phisical objects)
  - (thus standardisation) is moving up ap middleware/application level

- if too many standards:
  - not only vertical silos interoperability problem but problem with different standards
  - *Interoperability* problem: competing alliances befines own stntandars → competition
  - devolopement is a competition moves to upper layers (where data are processed, collected, stored) → never ending process
  - solution: introduce ***IoT application/integration gateways***:
    - are at application layer
    - don't translate only low-level protocols
    - also map one into the other different application-level behaviors
    - transition from one protocol to another
    - gateways work also at: session(5), presentation(6), application(7) layer

• shift from INTEROPERABILITY of vertical silos to interoperability of protocols standard
- HARD: semantics of standards are ≠ ⇒ TRANSLATION of protocols implies semantics like Bluetooth ZigBee

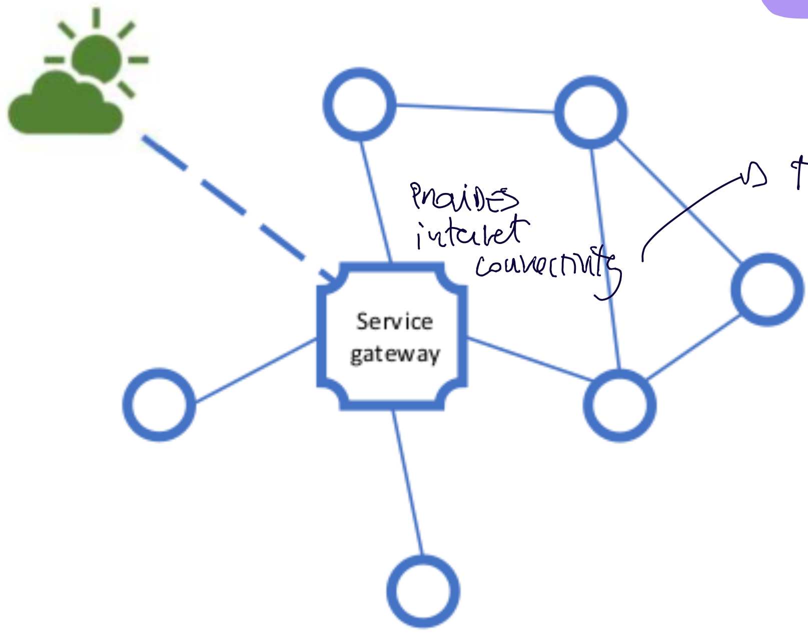
-

- **gateway configurations**

- *type A configuration*

Host: service interworking gateway
L7 application layer: data
L6 Presentation layer: data
L5 Session layer: data
L4 transport layer: frames
Medium:
L3 Network layer: packets
L2 Data-link layer: frames
L1 phisical layer: bits

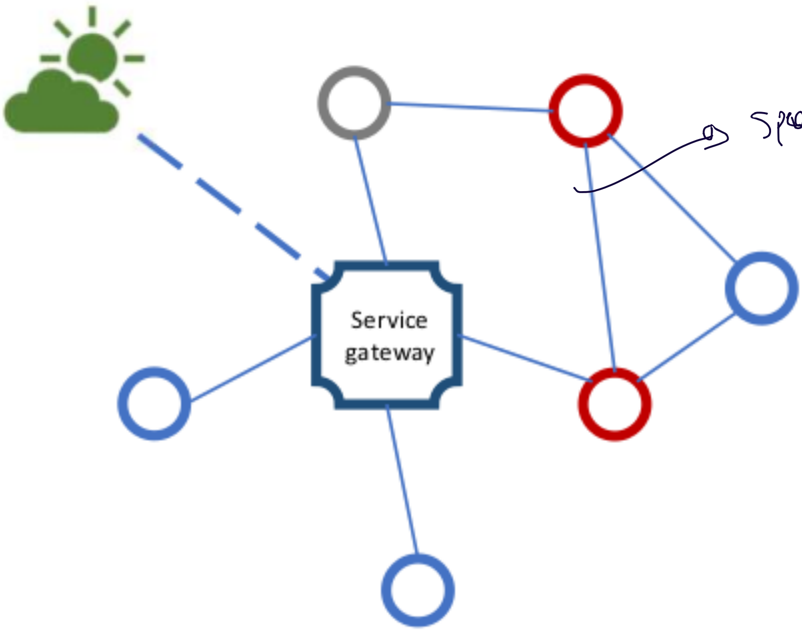
TYPE A



Provides internet connectivity

→ translating an internet protocol to something can go on the elsewhere (internet)

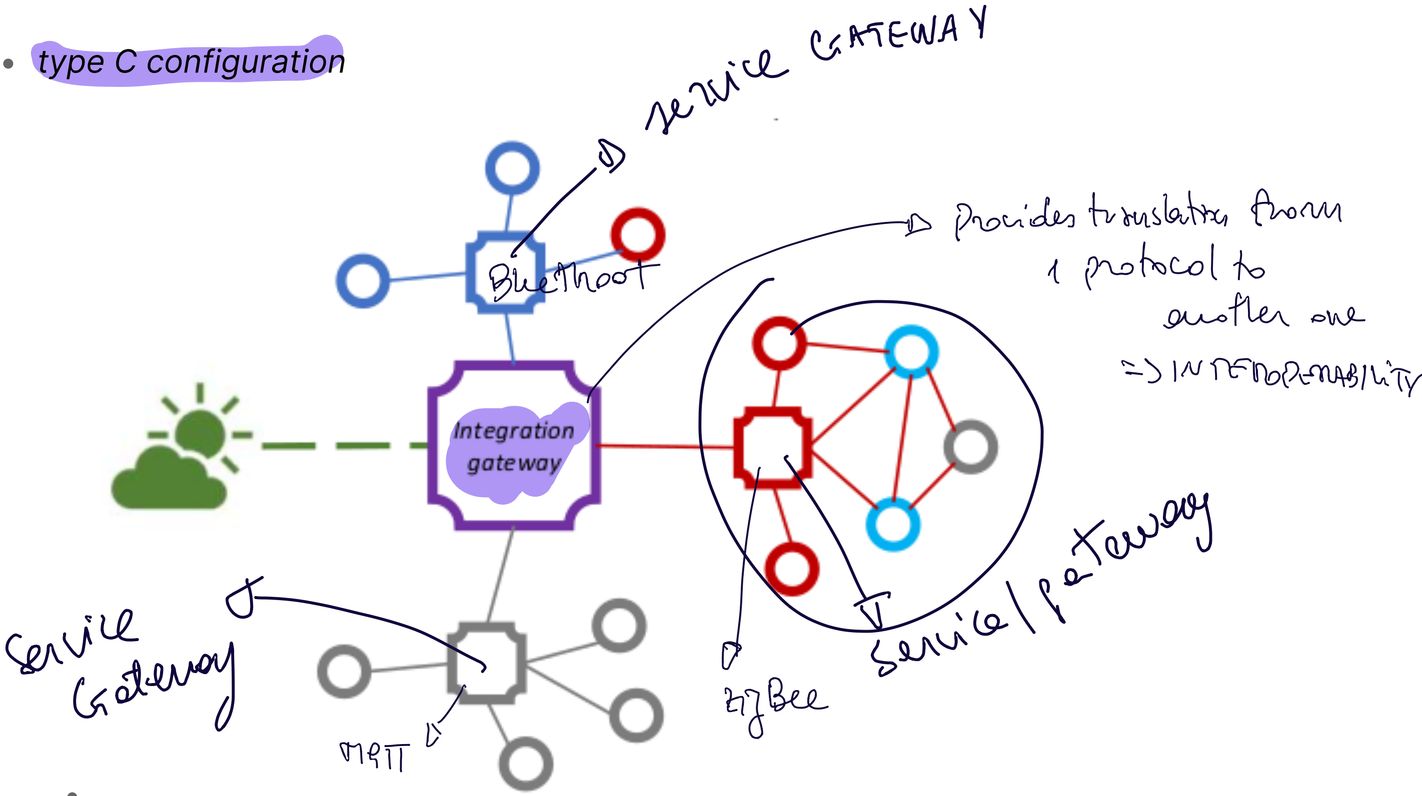
- Same vendor and same protocols
  - protocols to communicate at same level, devicese may be not standardised (also vertical silos solution), devices need acess to the world(internet) →
  - **service gateway** enables communication of devices to the worlds
- type B configuration



→ speak same language

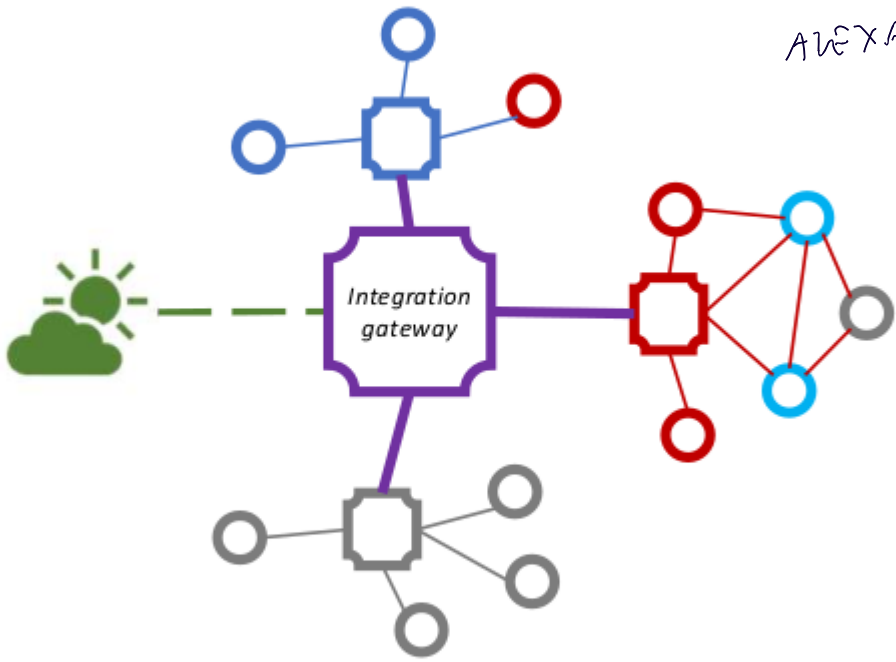
  - *two different vendors*, *same protocol*
  - devices manufactured from different industries
  - IoT devices respect a standard (for communication)
  - still need gateway to connect devices to the world (internet)
  - different networks and need go to the cloud → introduce service gateway
  - case of zigbee networks: 1 device provides as service gateway

- *type C configuration*



  -
  - *different vendors and different protocols*
  - devices are different by groups (proprietary), each have its own service gateway
    - different networks, devices communicate by different protocol (→ service gateway)
  - they neeed to map their messages to other proprietary → integration gateway: enable talking service gateway (n map 1 protocol) that impement differt behaviours
  - integration gateway is complex: a (integration) gateway that talks to other (service) gateway → map one protocol to another inside protocol
    - enforce reliability (1 gateway is down, there are others)
  - e.g.: temperature sensor, wi-fi connected, at application level need data to send somewhere, this transition may be implemented:
    - Push: send every data when is available
    - Pop: data are send on request
  - '#' of mapping from one protocol to another (at same level) the integration gateway manages n^2 mappings (every protocol to another)

- *type C/II configuration*



ALEXA/GOOGLE HOME:
- they open the market to who is able to speak their protocol
=> forces other single vertical silos

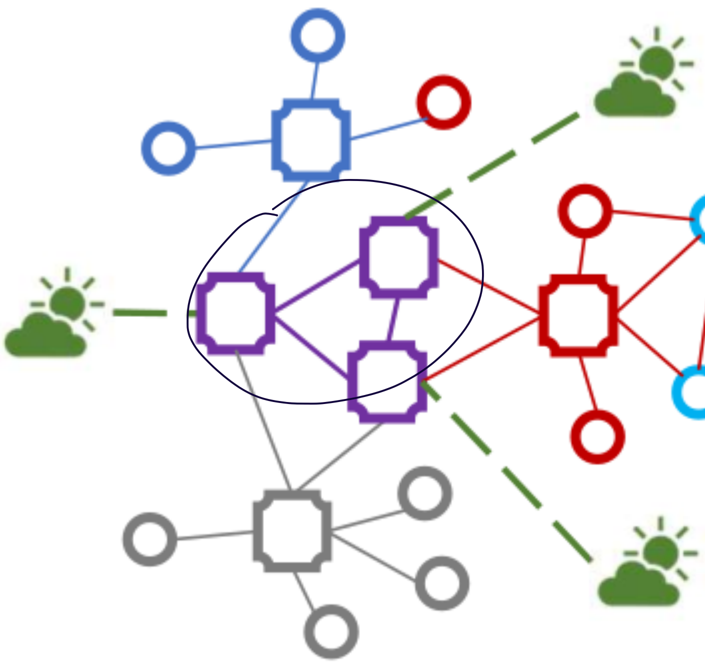
-
- *Integration gateway is a smart device e.g.: google home, alexa*
- is a service gateway
- big players force using their own solution: protocols, interfaces
    - to connect different devices you have to speak big palyers' protocols
- need to translate behaviour to big players' device and Iot provides integration

• *type D configuration*

• Since integration gateway has to map 1 protocol to others (type C) square $O(n^2)$ => SOL.
• An integration gateway which is distributed.

1) reduce complexity (linear)

2) these integration gateways talk an internal language
=> so 1 int. gateway has to translate the vertical silos to ~~their~~ internal protocol

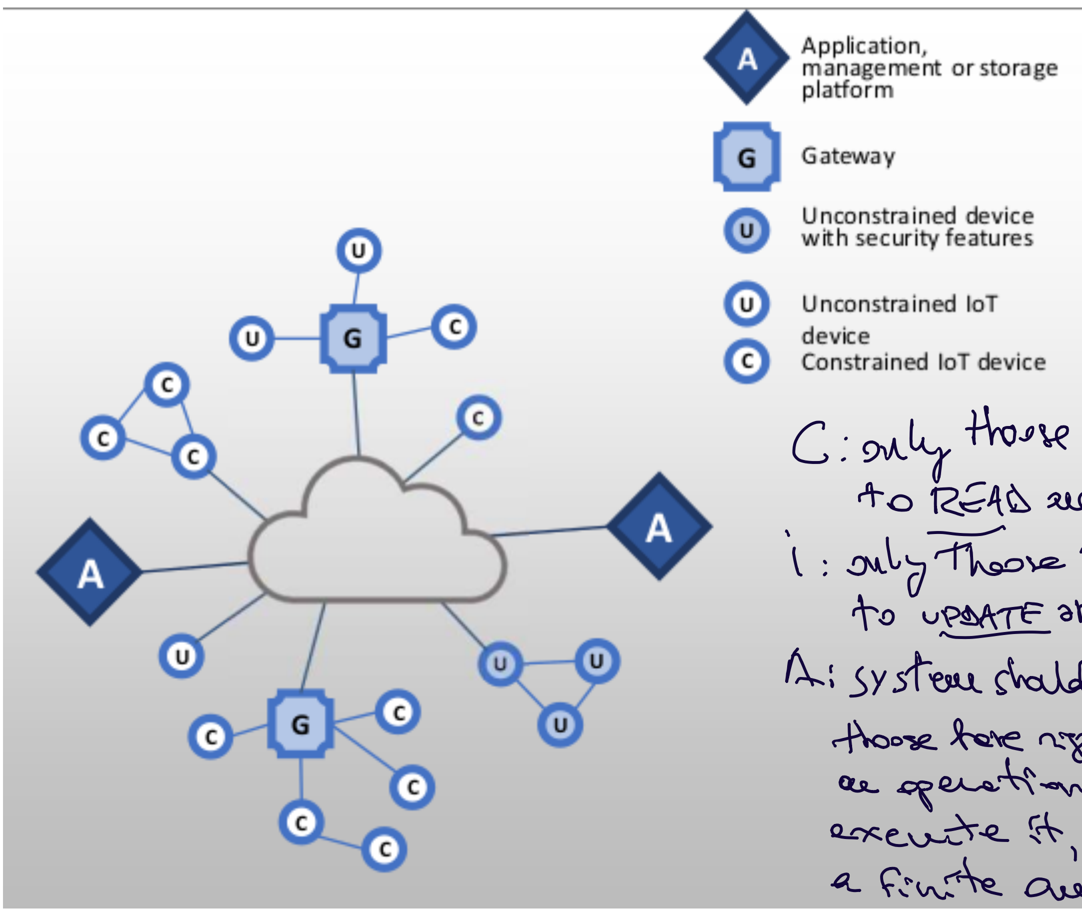


4) More reliability $O(n)$

-
- different vendors, different protocols, distributed integration gateways
- '#' of mapping from one protocol to another (at same level) the integration gateway manages n mappings(linear, n standard → n transiotion → 1 mapping)

- 1 solo passaggio tra i gateway e redundancy (a gateway goes down, no problem)

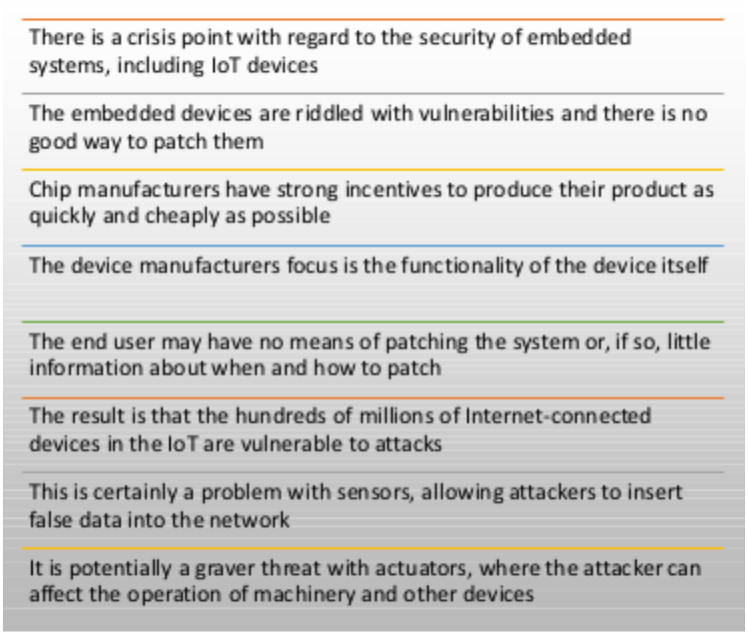
security in IoT SECURITY in IoT

- problems:
  - in IoT the number of devices connected to internet are heterogenously in the way they are connected
    - heterogenity is a problem in security: cuz security threat case by case → specific security features
  - IoT devices don't come with security features (or at least are insufficient)
- Ways to connect IoT devices:
  - IoT devices can be directly connected to the internet or via gateway
    - why? → for oroviding data to the cloud (processed, collected, analysed)
  - devices can be:
    - uncostrained with security feature (rare)
    - uncostrained IoT (e.g.: smart camera that process big amount of data)
    - costrained IoT device: limitation is connecting, managing them
- 

C: only those have rigth to READ an info, can read it
I: only Those that have right to UPDATE an info, can update it
A: system should guarantee that those have rigth to execute an operation and want to execute it, can do this in a finite amount of time

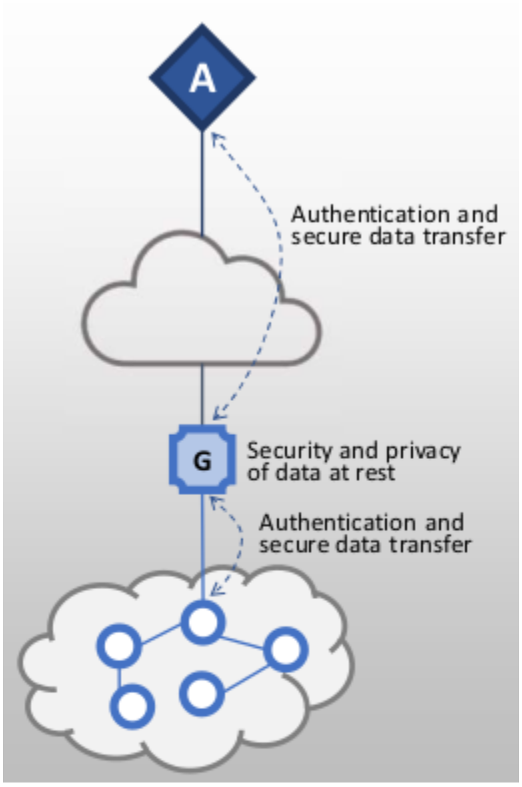
  - **patching vulnerability** (problems):

- when vendors develop a product → design functionalities and (hastily:frettolosamente) want to reach the market before competitors
- standards don't provide security features
- cuz they need to be fast → devices are full of bugs, software/hw, (can became vulnerabilities)
- general devices has an OS stuctured and get patches to update sw/fw
  - in IoT OS are very light with no (or limited) security features
    - OS is shitty, impossible installing updates
- when IoT devices when are already into the market is impossible to patch
- even if manufactures want, no way to fix them (apart redefing/redeploy them)
- problem of sensors:
  - related to confidentiality (e.g.: wearable sensors → health informations)
  - related to authenticity of data: false informations tampered that sensor get/send (e.g.: temperature in nuclear power plants)
- problem of actuators:
  - they act (take actions, e.g.: robot arm), fake commands have much more effect

- 

- **IoT privacy /security requirements**
  - IUT-T stndrds reccomandation Y.2066 includes list of security requirements for the IoT during; capturing, storing, transferring, aggregating and processing the date of thing:
  - **security audit**: logging all relevant events of a system
    - → investigation or feed sensors to detect attacks

- **communication security**: from communication layer, confidentilaty and integrity during data transfers or trasmission
- **data management security**: confidentiality and integiry of data when storing or processing data
- **service provision security**: deny anouthorased accesses to service and protect privacy informations ralate IoT
- **Mutual authentication** : Iot that belong to a network communicate each other → service gateway, devices authentication must be done at any layer you need → different layers (application layer authentication or locally at the gataway)
  - must be on the gateway and device (before a device can access the IoT)
  - possible implemet a fake gatway → device need to trust the gateway
- **integration of security policies and techniques:** ability to do that, ensures security vontrol over variety of devices and user network

- **IoT gateway security functions:**
  - 

  - Security specifications particuraly concerns gateways
    - security needed from gateway to the cloud
    - security needed from device to gateway
      - → data flows through gateway be protected
    - some case access points or built network between IoT devices
    - gateway is more powerful device
      - much energy
      - powerful machines

    - most of the work performed by gateways → device implement what is needed
- as soon IoT devices are locate into the wild, someone can tampere them
    - uploading new firmware or software
    - so authentication needed on phisical layer (perception)
    - but also needed at application layer (cuz decive → gateway → cloyd)
- gateways are used to update, check, control device and auto configuration or configuration by applications
    - deployment phase is critical:
    - devices don't know to which gateway are connected
        - configuration by hand || want be performed by gateway in a secure way