UNIVERSITÀ
DEGLI STUDI
DI UDINE

IoT e Sistemi Distribuiti

Architettura

IoT - Caratteristiche

- I dispositivi sono **dinamici**: si adattano automaticamente all'ambiente.
- I dispositivi sono **auto-configuranti**.
- I dispositivi comunicano per mezzo di **protocolli di comunicazione interoperabili**, permettendo lo scambio di informazioni fra **sistemi eterogenei**.
- Ogni **nodo** è distinto da una **propria identità** e da un **identificativo univoco**.
- I dispositivi sono **integrati** nell'infrastruttura delle reti di comunicazione che utilizzano.

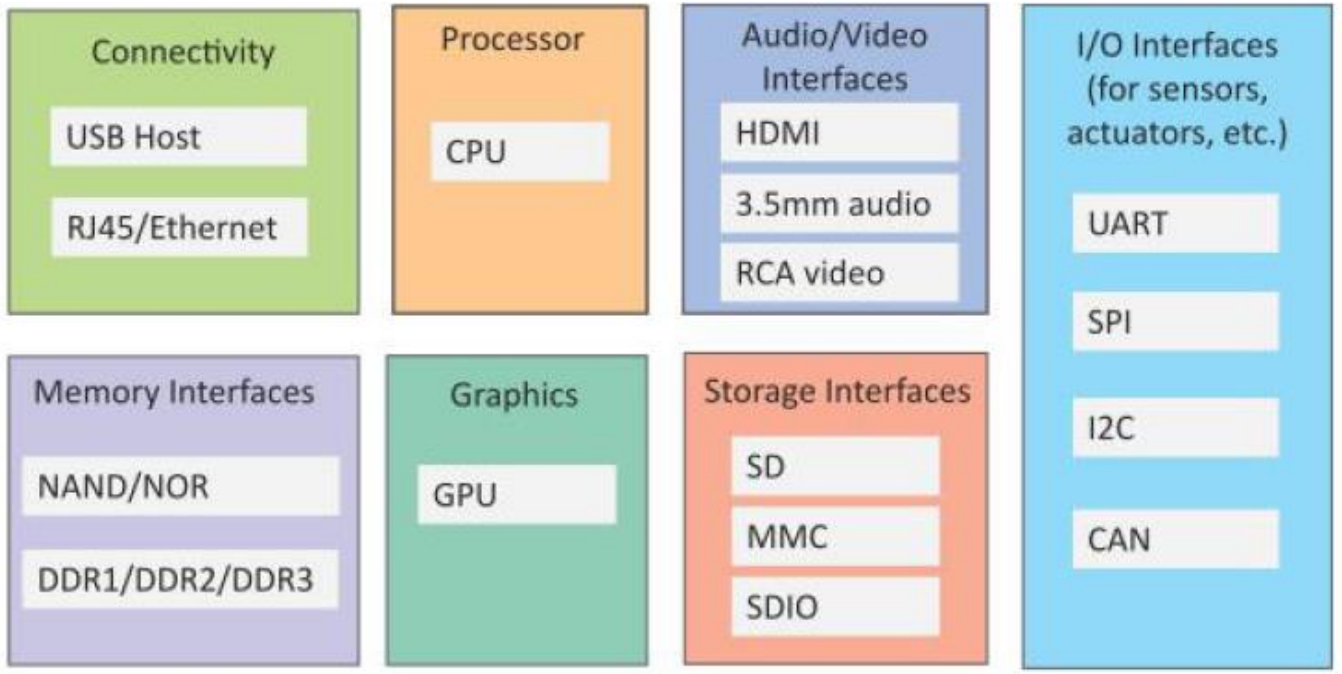
Architettura fisica dell'IoT

- Le "cose" ("Things" in IoT) si riferiscono a **dispositivi** con **identità univoche** in grado di effettuare azioni di **acquisizione** dati da **sensori**, di **azionamento** di **attuatori** e di **monitoraggio** (anche in modalità remota).

- In particolare i dispositivi IoT possono:
  - **scambiare dati** (direttamente o indirettamente) con altri dispositivi connessi od applicazioni;
  - **raccogliere dati** da altri dispositivi ed **elaborarli** localmente od **inviarli** a server centralizzati o back-end di applicazioni cloud;
  - **assolvere** certi **compiti** localmente od entro l'infrastruttura IoT, in base a determinati vincoli spazio-temporali.

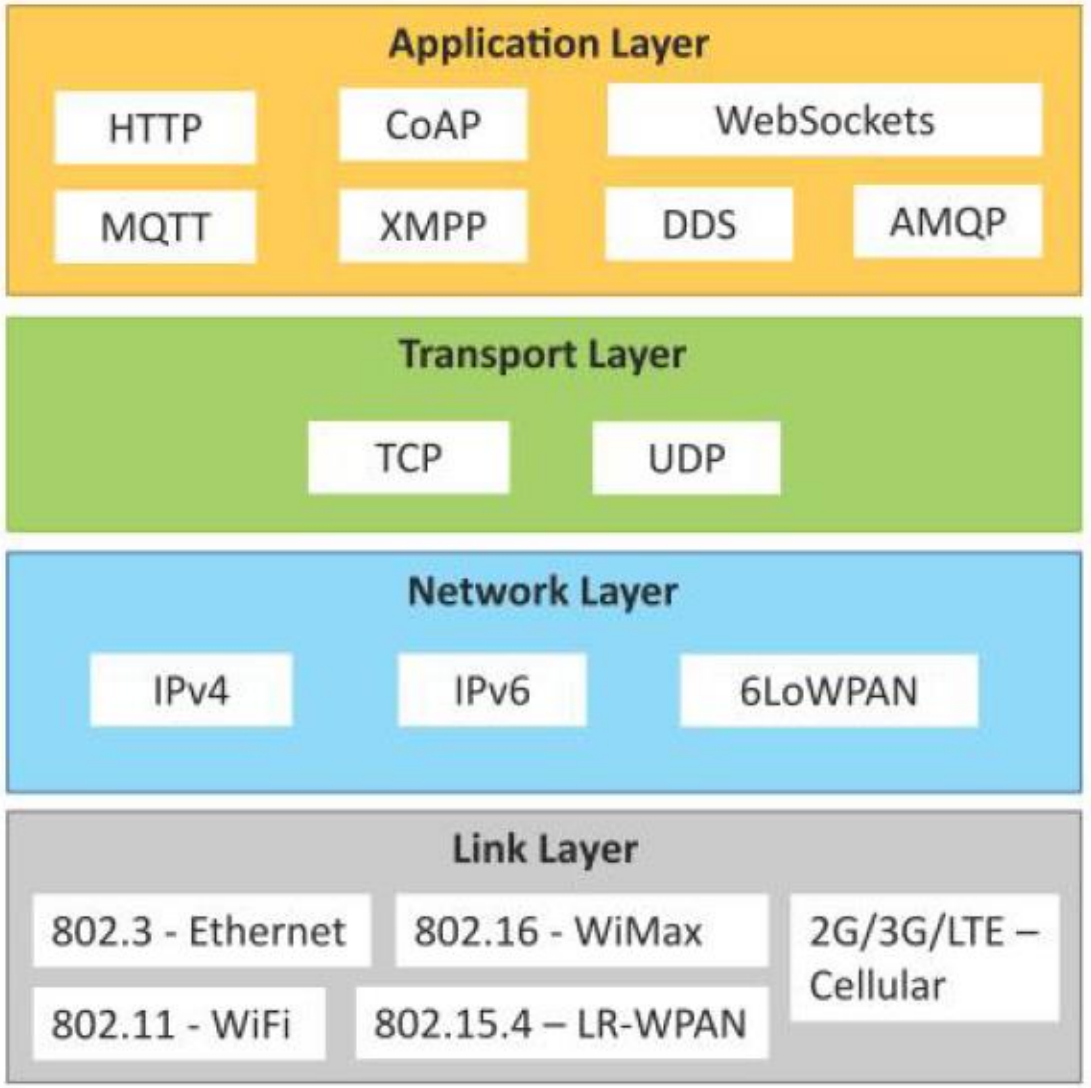
IoT – Diagramma a blocchi (generico)

- Un dispositivo IoT può integrare **varie interfacce** per connettersi ad altri dispositivi (sia in modo **wireless** che **cablato**):
  - interfacce di I/O per i **sensori**;
  - interfacce per la **connessione a Internet**;
  - interfacce per la **memorizzazione/storage**;
  - interfacce **audio/video**.



Protocolli per l'IoT

- Collegamento dati (Link Layer):
  - 802.3 – Ethernet
  - 802.11 – WiFi
  - 802.16 – WiMax
  - 802.15.4 – LR-WPAN
  - 2G/3G/4G/5G
- Rete (Network/Internet Layer):
  - IPv4
  - IPv6
  - 6LoWPAN
- Trasporto (Transport Layer):
  - TCP
  - UDP
- Applicazione (Application Layer):
  - HTTP
  - CoAP
  - WebSocket
  - MQTT
  - XMPP
  - DDS
  - AMQP



Protocolli a livello data link

- IEEE 802.3 – Ethernet (CSMA/CD, mezzo trasmissivo: cavo coassiale, doppino incrociato, fibra ottica, velocità: 10 Mb/s – 40 Gb/s).
- IEEE 802.11 – Wi-Fi (CSMA/CA: 2,4 GHz/5 GHz, velocità: 1 Mb/s – 6,8 Gb/s).
- IEEE 802.16 – WiMax (velocità: 1,5 Mb/s – 1 Gb/s)
- IEEE 802.15.4 – LR-WPANs (velocità: 40 Kb/s – 250 Kb/s)
- 2G/3G/4G/5G – Mobile communication, cellular networks (velocità: 9,6 Kb/s – 1815 Mb/s)

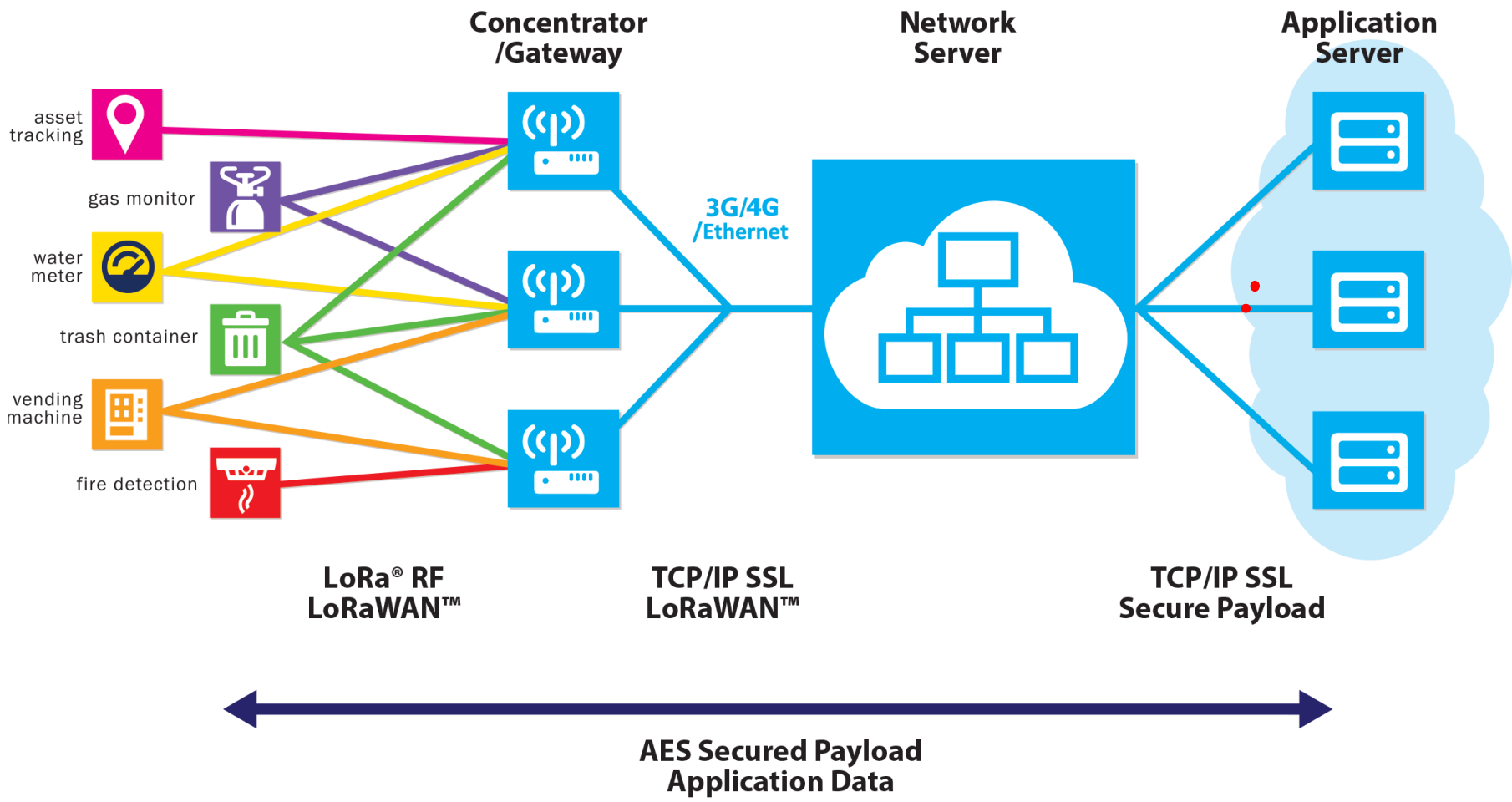
Esempio: LoRaWAN

FREQ. LIBERE

- Long Range WAN basata su bande ISM (Industrial, Scientific and Medical) (EU 863-870MHz, EU 433MHz)

Tre classi di dispositivi:

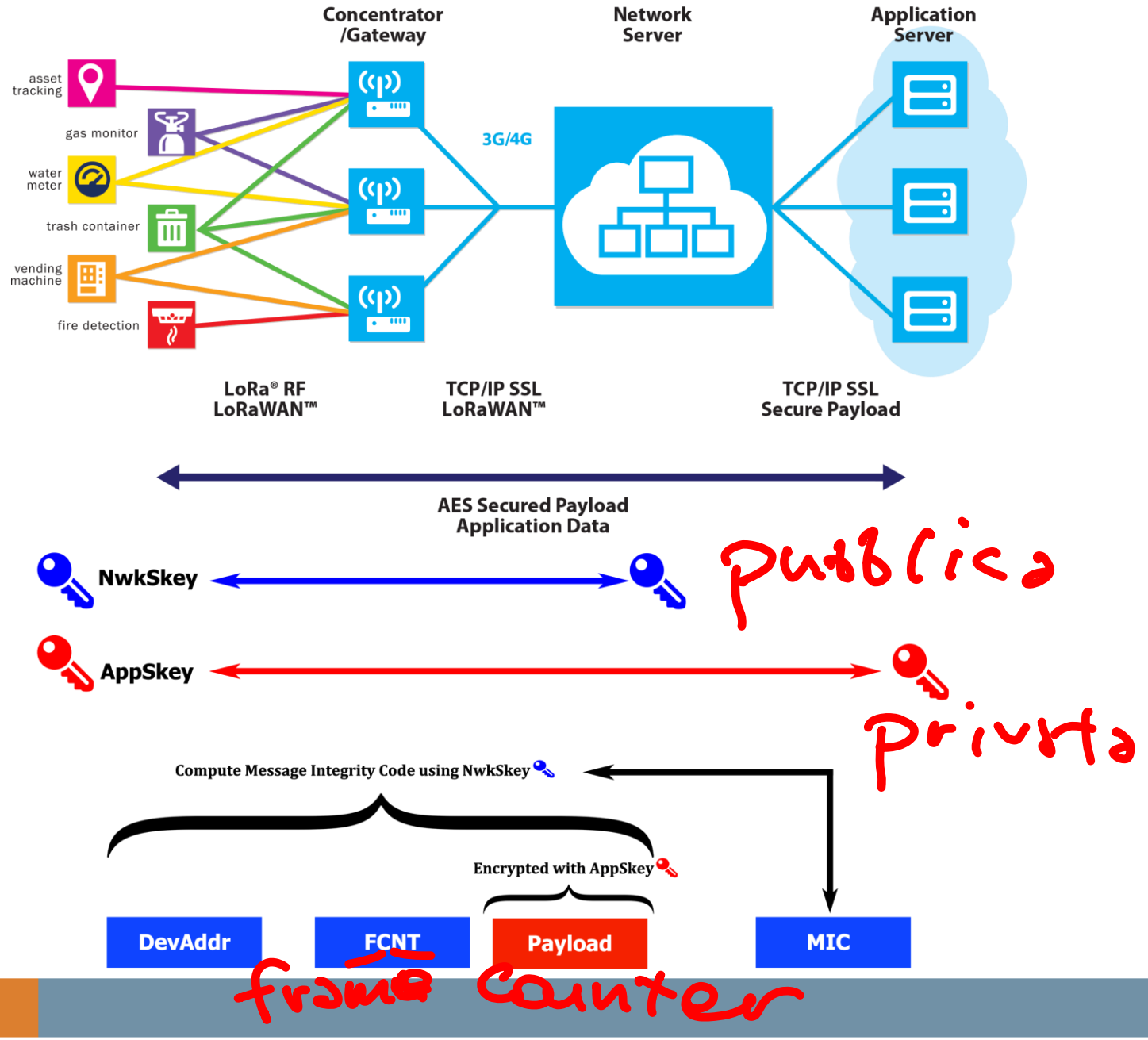
1. Classe A: alimentati a **batteria**. Uplink spedito al server → il device apre due piccole finestre per eventuali comandi.
2. Classe B: alimentati a **batteria**, hanno una finestra extra per i downlink, che viene aperta in momenti pianificati.
3. Classe C: **alimentati elettricamente**, con finestre per i downlink quasi sempre aperte.



LoRaWAN

I device hanno due modi per eseguire un join con il network:

1. OTAA (Over-the-Air-Activation): il device e la rete si scambiano una chiave a 128 bit (AppKey). Quando il device spedisce la richiesta per effettuare il join, l'AppKey viene utilizzata per creare un Message Integrity Code (MIC), dopodiché il server controlla il MIC utilizzando l'AppKey. Se il controllo viene passato, il server crea due nuove chiavi a 128 bit, l'App Session Key (AppSkey) e la Network Session Key (NwkSkey). Queste chiavi sono spedite indietro al device, criptate utilizzando l'AppKey come chiave di criptazione. Quando le chiavi vengono ricevute, il device le decripta e le installa.
2. ABP, Activation By Personalization: le chiavi di sessione vengono inserite manualmente dall'utente.



Protocolli a livello di rete

- IPv4 (RFC 791): indirizzi a 32 bit (esauriti nel 2011).
- IPv6 (RFC 2460): indirizzi a 128 bit.
- 6LoWPAN (IPv6 over Low power Wireless Personal Area Networks): opera a 2,4 GHz in combinazione con il protocollo di livello data link 802.15.4 supportando transfer rate di 250 Kb/s (sostanzialmente fornisce algoritmi di compressione per i datagrammi IPv6).

Protocolli a livello di trasporto

request for comment

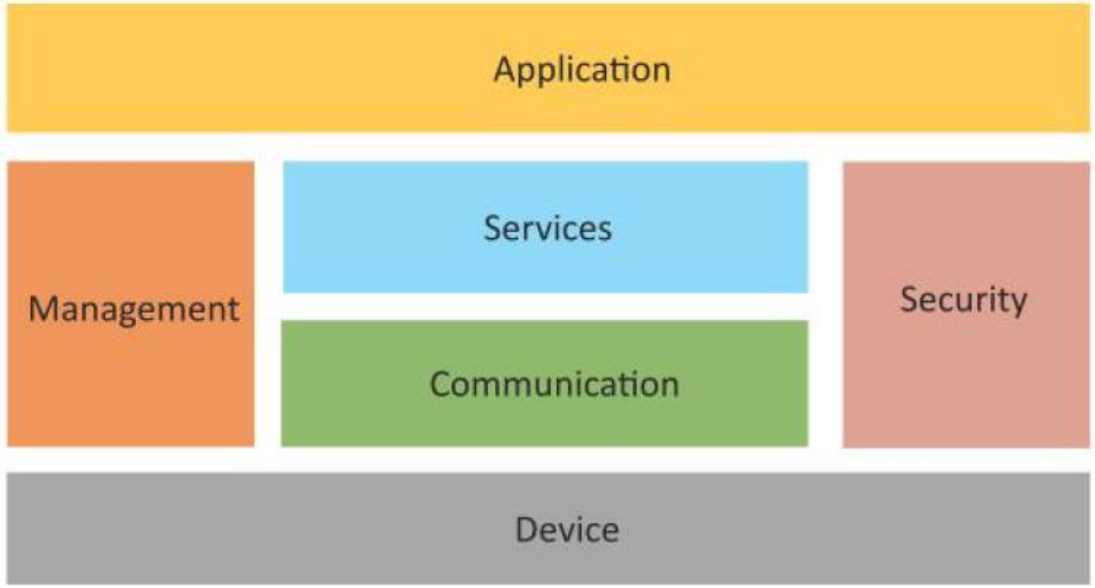
- TCP (RFC 793): orientato alla connessione con stato, garantisce ordinamento dei pacchetti, eliminazione dei duplicati e ritrasmissione dei pacchetti persi. Consente di negoziare la velocità di trasmissione e gestire la congestione del traffico.
- UDP (RFC 768): non orientato alla connessione senza stato, adatto per applicazioni che necessitano comunicazioni veloci, senza la necessità di un setup del canale. Non garantisce ordinamento e non gestisce eventuali errori di comunicazione.

Protocolli a livello di applicazione

- Permettono di codificare i dati a livello applicativo (file) nel formato utile per i protocolli di trasporto.
- I **numeri di porta** definiscono il tipo di applicazione/protocollo (e.g., 80 per HTTP, 1883 per MQTT ecc.):
  - **HTTP** (HyperText Transfer Protocol): stateless, **request-response**, usa URIs per identificare le risorse;
  - **CoAP** (Constrained Application Protocol): **applicazioni M2M** per dispositivi e infrastrutture limitati (constrained), simile a HTTP (ma usa UDP invece di TCP);
  - **WebSocket**: comunicazione client/server full-duplex su TCP (**socket bidirezionale**);
  - **MQTT** (Message Queue Telemetry Transport): client/server basato sul modello **publish/subscribe**;
  - **XMPP** (Extensible Messaging and Presence Protocol): **comunicazione real-time** di messaggi XML (sia client/server che server/server);
  - **DDS** (Data Distribution Service): comunicazione **M2M** con modello **publish/subscribe**;
  - **AMQP** (Advanced Message Queuing Protocol): protocollo per **business messaging** per comunicazioni **point-to-point** o con modello **publish/subscribe**.

Progettazione logica

- La progettazione logica di un Sistema IoT è una rappresentazione astratta delle entità e dei processi coinvolti senza entrare in dettagli di basso livello sull'implementazione.
- Un Sistema IoT è composto da un insieme di **blocchi funzionali** che gli conferiscono le seguenti caratteristiche:
  - identificazione,
  - sensing,
  - attuazione,
  - communicazione,
  - gestione.

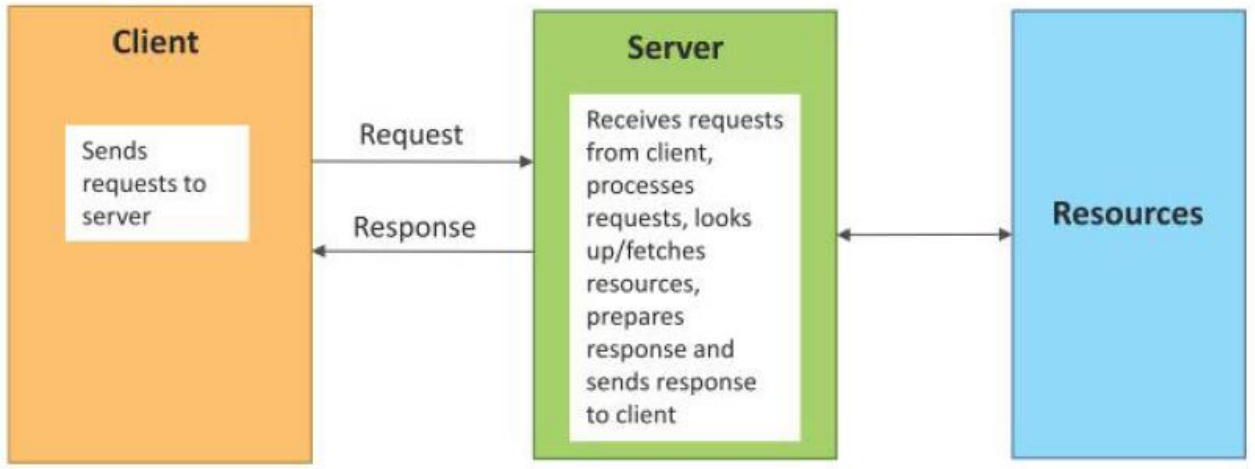


Rivisitazione per l'IoT

Asimmetrico

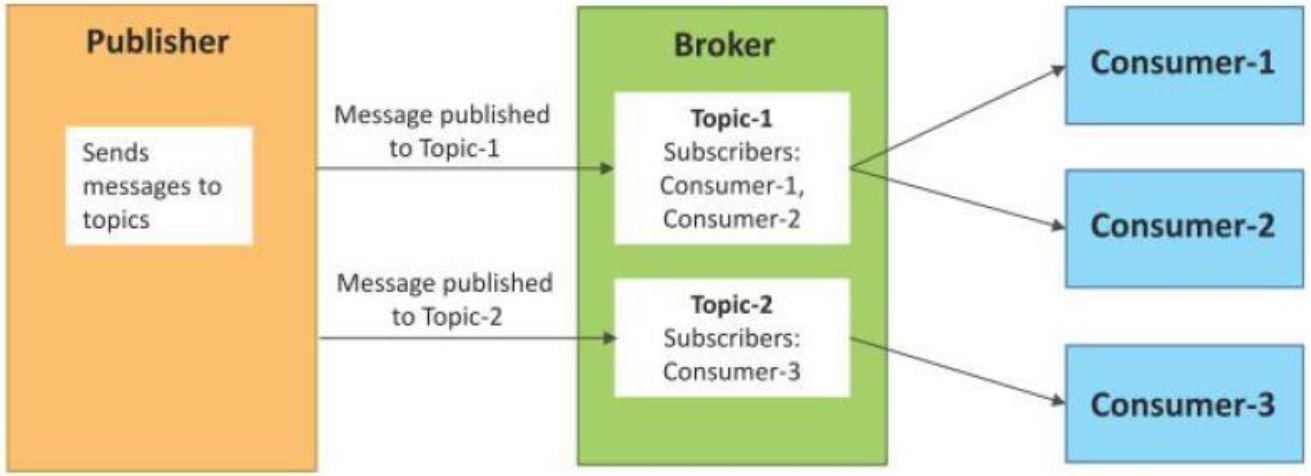
Modelli di comunicazione: Request-Response

- Il modello di comunicazione basato sul paradigma **Request-Response** prevede che i **client** inviino delle richieste al **server** e che quest'ultimo risponda a tali richieste.
- Quando il server riceve una richiesta, effettua le seguenti operazioni:
  - decide come rispondere;
  - recupera i dati e le rappresentazioni delle risorse;
  - genera la risposta e la invia al client.



Modelli di comunicazione: Publish-Subscribe

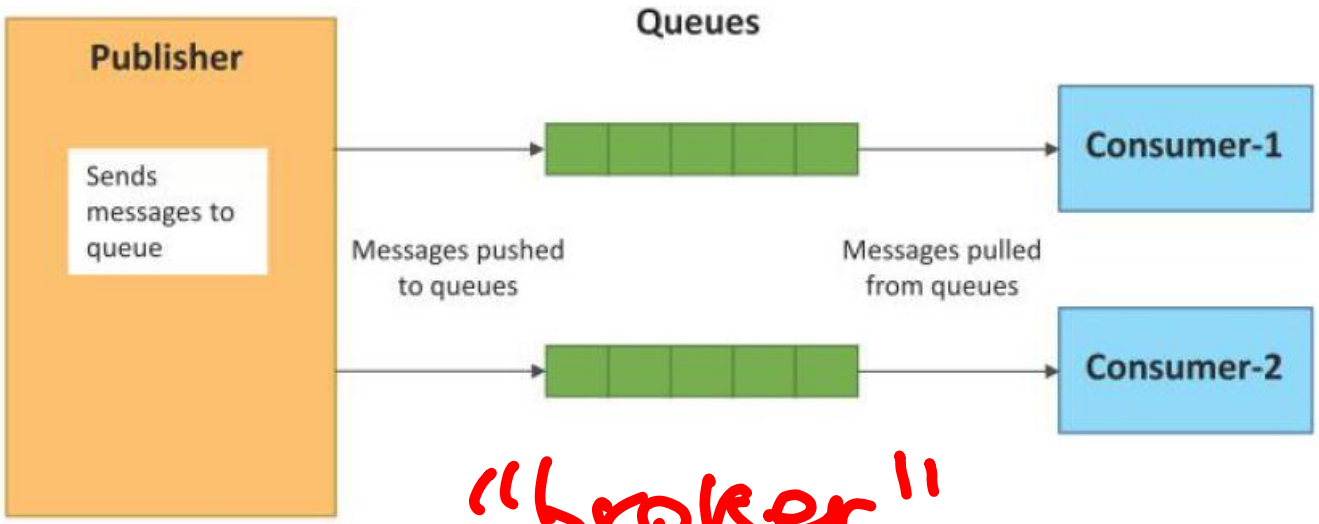
- Il modello di comunicazione basato sul paradigma **Publish-Subscribe** prevede i seguenti ruoli:
  - **publisher** (sorgenti di dati): invia i dati al broker assegnando loro un argomento (topic);
  - **broker**: gestisce gli argomenti ed i dati forniti dai publisher, inviandoli ai consumer in base agli argomenti a cui si sono iscritti;
  - **consumer**: si "iscrivono" agli argomenti gestiti dal broker e ricevono i dati relativi dal broker.



Modelli di comunicazione: Push-Pull

- Il modello di comunicazione basato sul paradigma Push-Pull i **produttori** inviano i dati a delle code.
- I **consumatori** prelevano i dati da tali code.
- I produttori non devono preoccuparsi dei consumatori e viceversa.
- Le code aiutano a disaccoppiare lo scambio di messaggi tra produttori e consumatori.
- Le code fungono anche da **buffer** nelle situazioni in cui produttori e consumatori lavorano con **velocità differenti**.

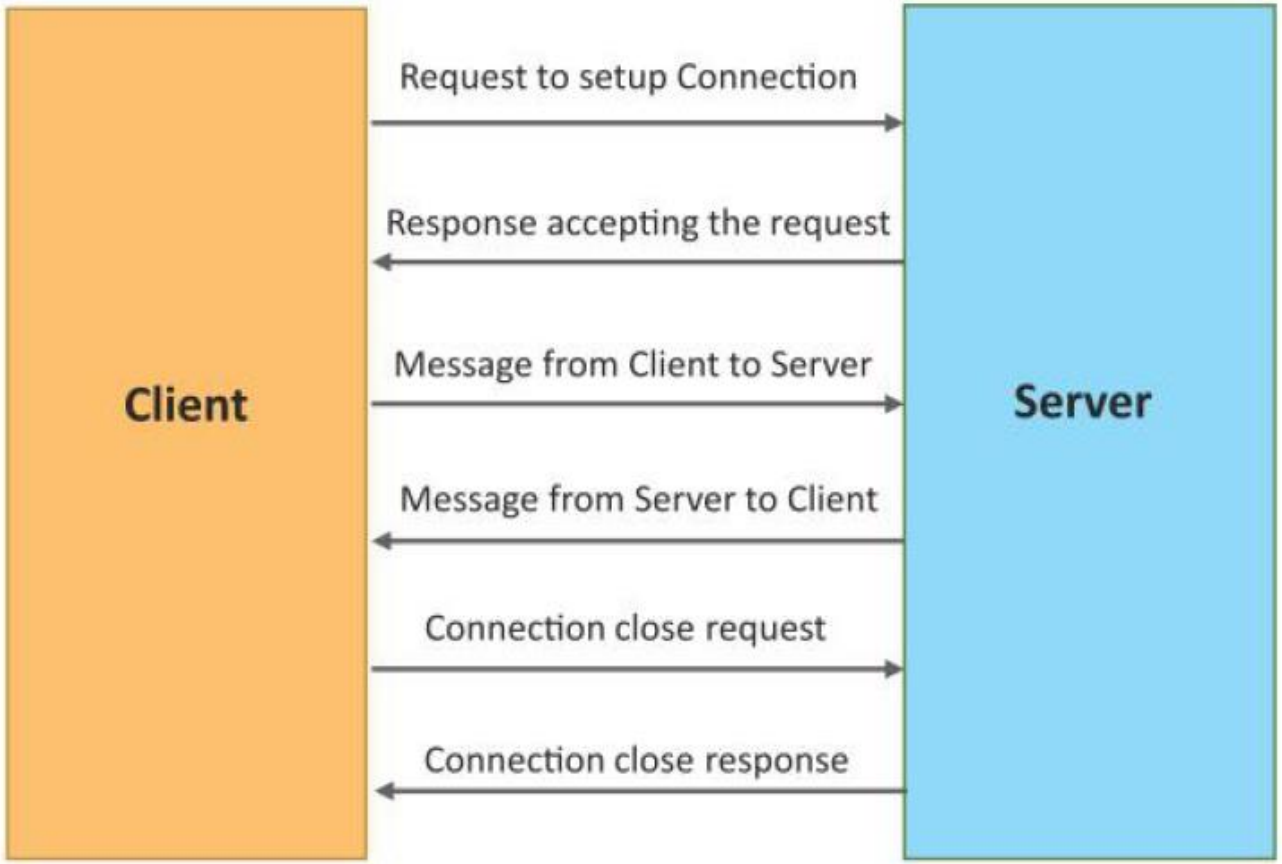
nodo con sensore



"broker"
1 topic solo

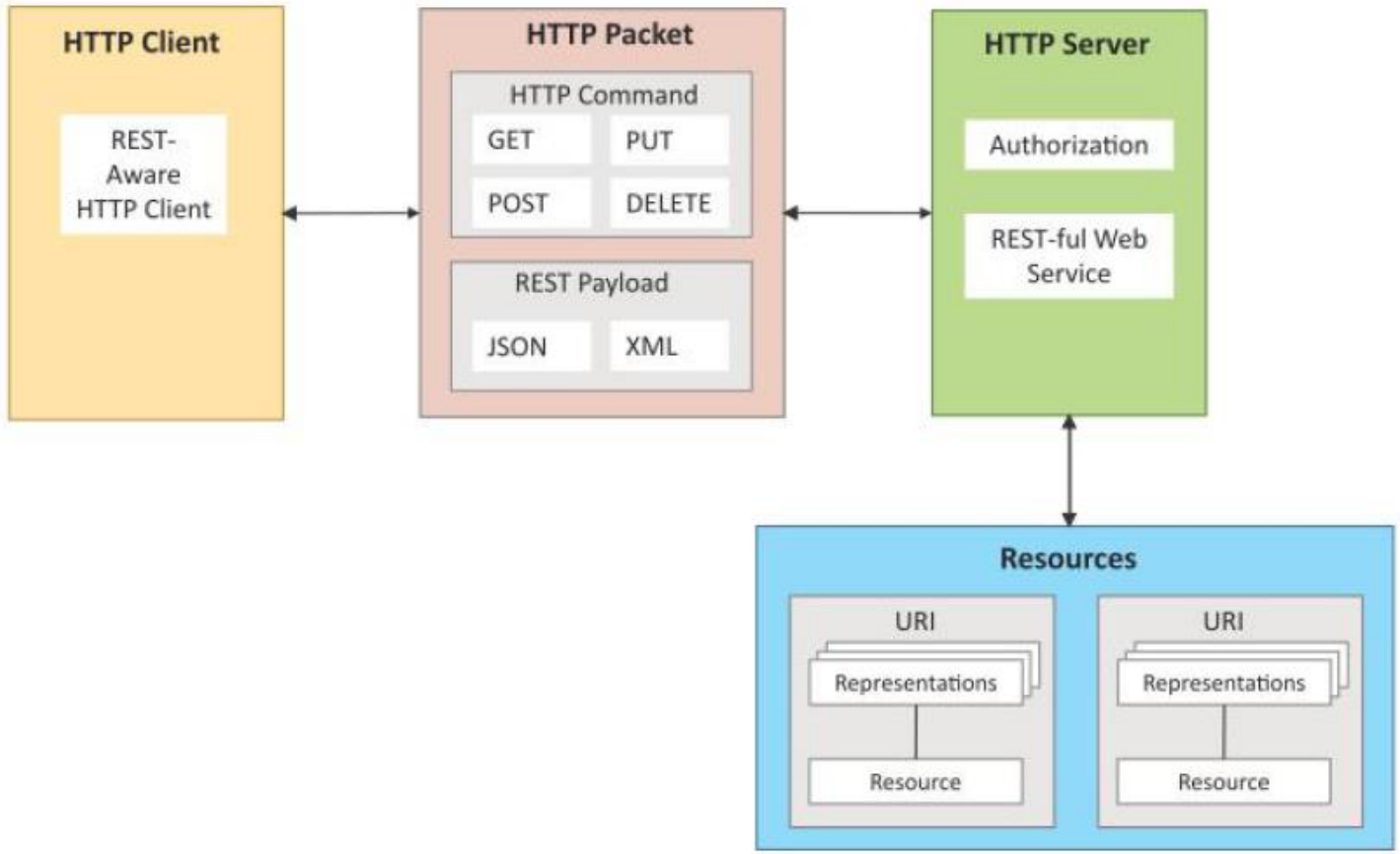
Modelli di comunicazione: Exclusive Pair

- Il modello di comunicazione basato sul paradigma Exclusive Pair è **bidirezionale** e **full duplex** ed utilizza una connessione stabile fra client e server.
- Una volta che la connessione viene stabilita, resta aperta fintanto che il client non invia una richiesta di chiusura.
- Il client ed il server possono scambiare messaggi dopo l'apertura della connessione.



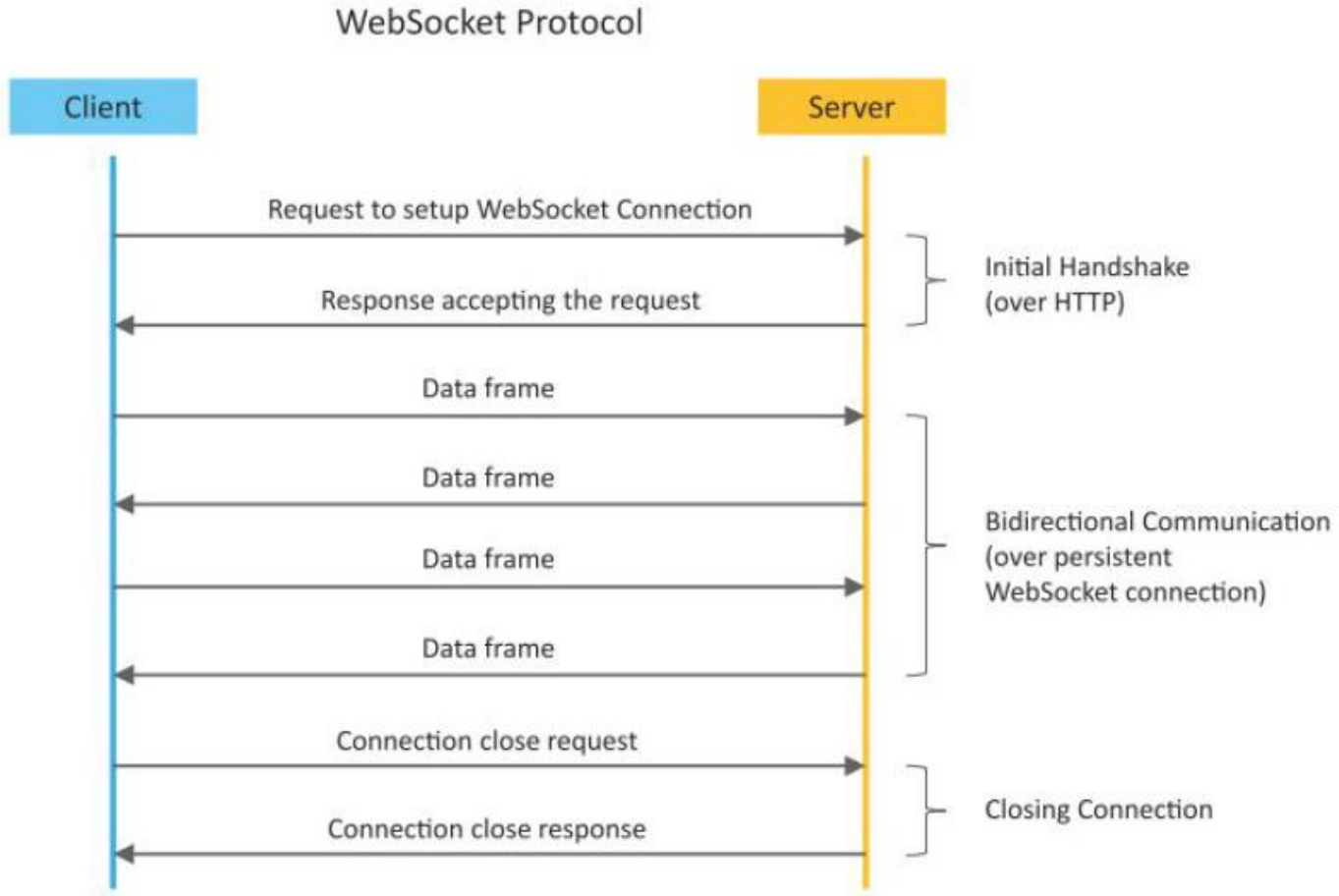
API di comunicazione basate su REST

- REpresentational State Transfer (REST) è basato su un insieme di principi architetturali per cui si possono progettare web service e web API che concentrano l'attenzione del programmatore sulle **Risorse del Sistema** e su come gli **stati di tali Risorse** vengano rappresentanti e comunicati.
- La API di tipo REST seguono il modello di comunicazione **Request-Response**.
- I vincoli dell'architettura REST si applicano a componenti, connettori ed elementi di dato, all'interno di un sistema ipermediale distribuito.



API di comunicazione basate su WebSocket

- La API di tipo WebSocket consentono una comunicazione bidirezionale, full duplex fra client e server.
- La API di tipo WebSocket seguono lo standard del modello di comunicazione Exclusive Pair.



Livelli dell'IoT e template di deployment (I)

Un sistema IoT è format dale seguenti componenti:

- **Dispositivi**: un dispositivo IoT consente l'identificazione, l'acquisizione dati, l'invio di comandi edil monitoraggio in modo remoto.
- **Risorse**: componenti software dei dispositivi IoT per l'accesso, l'elaborazione e la memorizzazione dei dati provenienti dai sensori, per il controllo degli attuatori e per l'accesso alla rete.
- **Servizi di controllo**: servizi nativi che girano sul dispositivo ed interagiscono con altri servizi di rete. Ad esempio, nel caso dei web service, i servizi di controllo inviano i dati dal dispositivo ad un web service e ricevono i comandi per il controllo del dispositivo da un'applicazione tramite un altro web service.

Livelli dell'IoT e template di deployment (II)

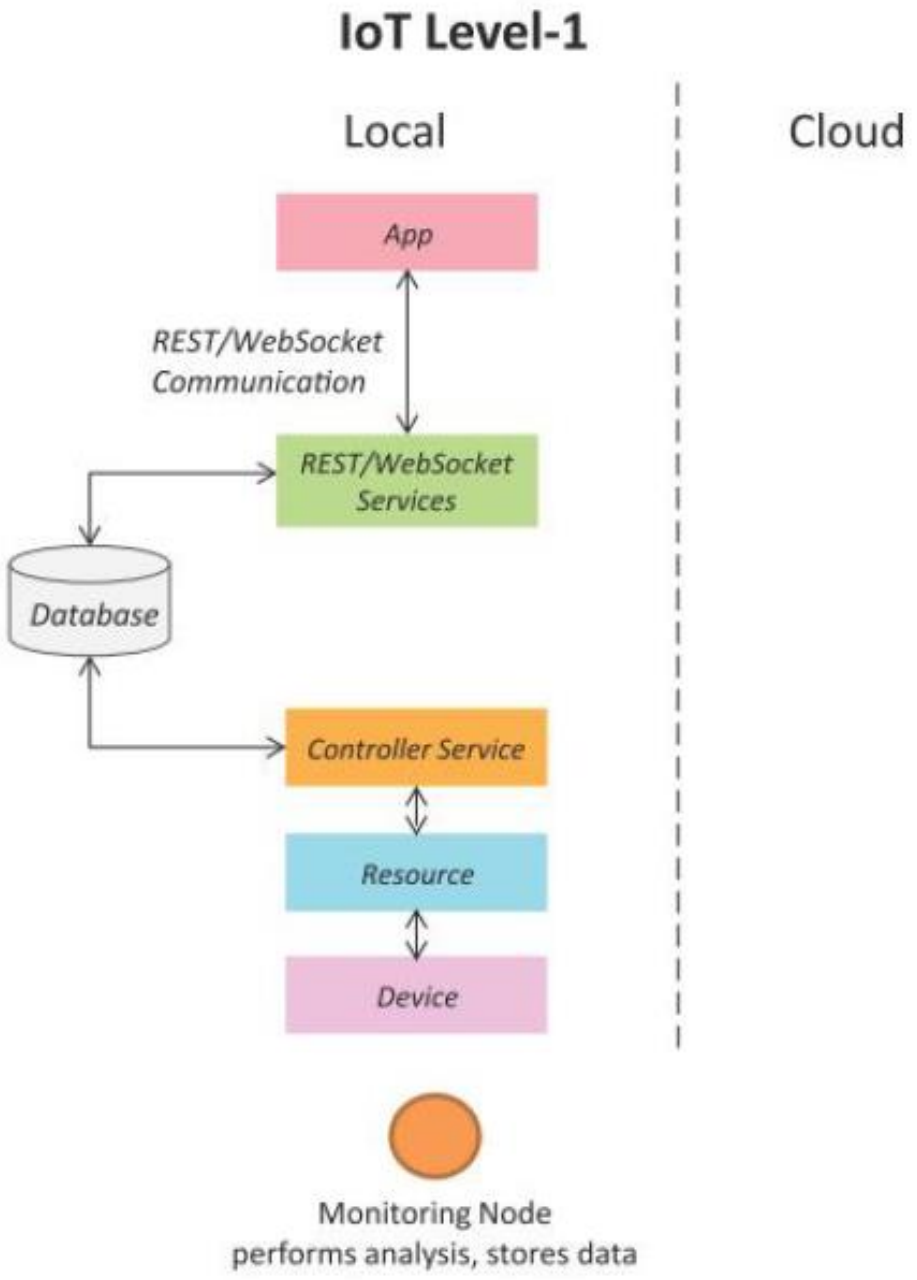
NoSQL: chiave-valore, lo schema muta

- **Database**: possono essere locali o distribuiti nel cloud e servono a memorizzare i dati generate dai dispositive IoT.
- **Web Service**: fungono da tramite fra i dispositive IoT, le applicazioni, i database ed i componenti di analisi. Possono essere implementati usando HTTP e REST oppure il servizio di WebSocket.
- **Componenti di analisi**: sono responsabili dell'elaborazione dei dati IoT e della generazione di risultati che siano comprensibili ed utili agli utenti.
- **Applicazioni**: forniscono un'interfaccia che gli utenti possono utilizzare per mantenere sotto controllo un sistema IoT, per visualizzarne lo stato e i risultati dell'elaborazione dei dati.

IoT – Livello 1

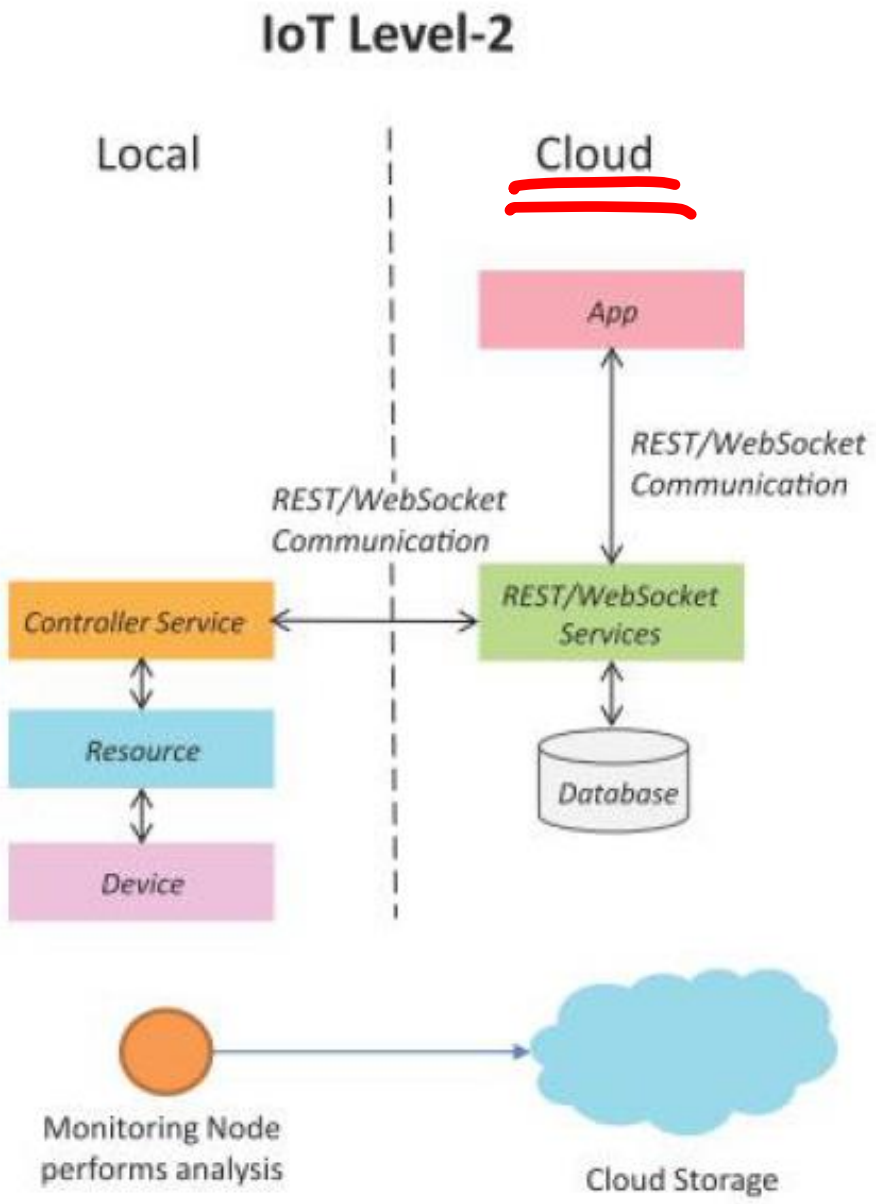
schema generale

- Un sistema IoT di livello 1 ha un **singolo nodo** (dispositivo) che acquisisce dati e/o comanda gli attuatori, memorizza i dati, li elabora e fa girare l'applicazione.
- I sistemi di Livello 1 sono adatti per soluzioni a **basso costo e complessità** in cui i dati coinvolti sono relativamente pochi e le attività di analisi non sono pesanti dal punto di vista computazionale.



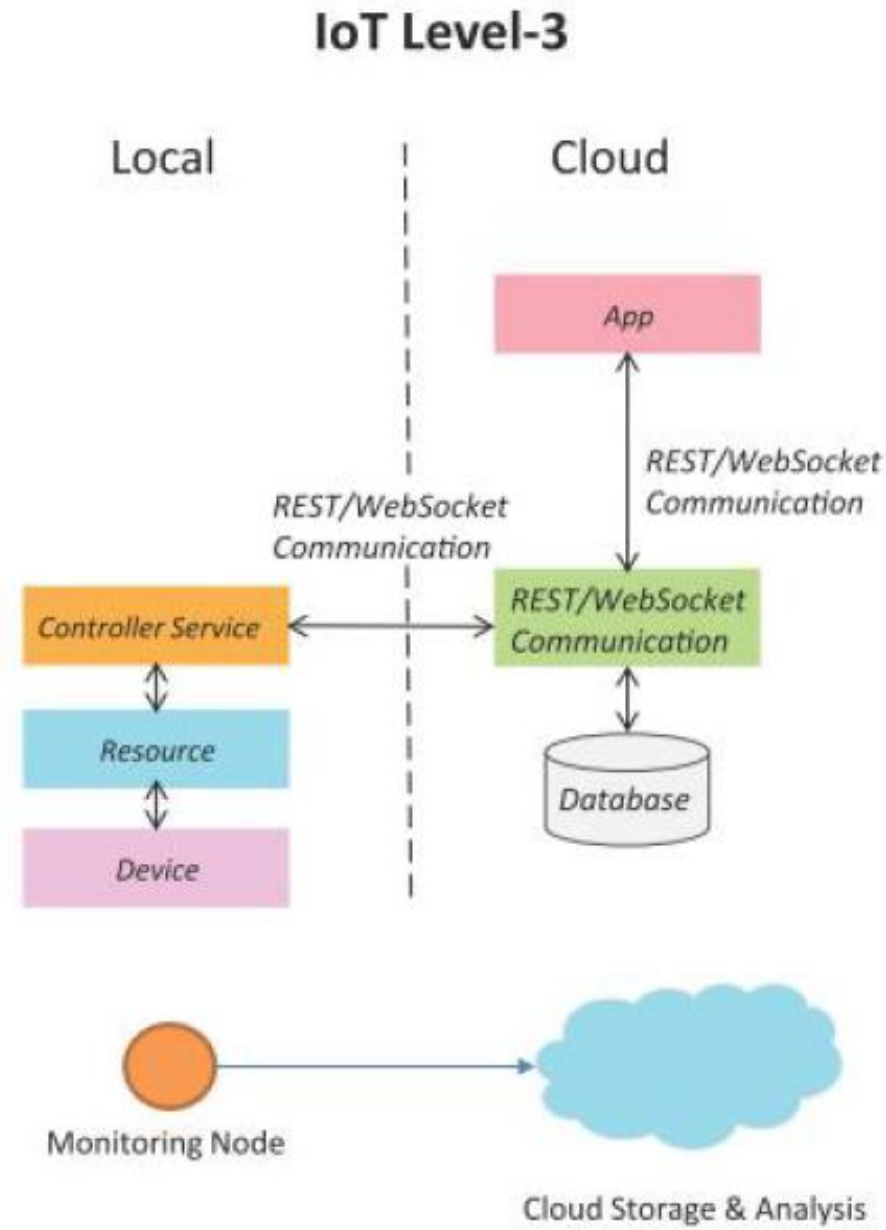
IoT – Livello 2

- Un sistema IoT di livello 2 ha un **singolo nodo** (dispositivo) che acquisisce dati e/o comanda gli attuatori ed elabora i dati.
- I dati vengono memorizzati nel cloud e **l'applicazione è cloud-based**.
- I sistemi di Livello 2 sono adatti per **soluzioni con molti dati**, ma in cui **l'attività di analisi** preliminare **non è pesante** dal punto di vista computazionale e quindi può essere espletata localmente.



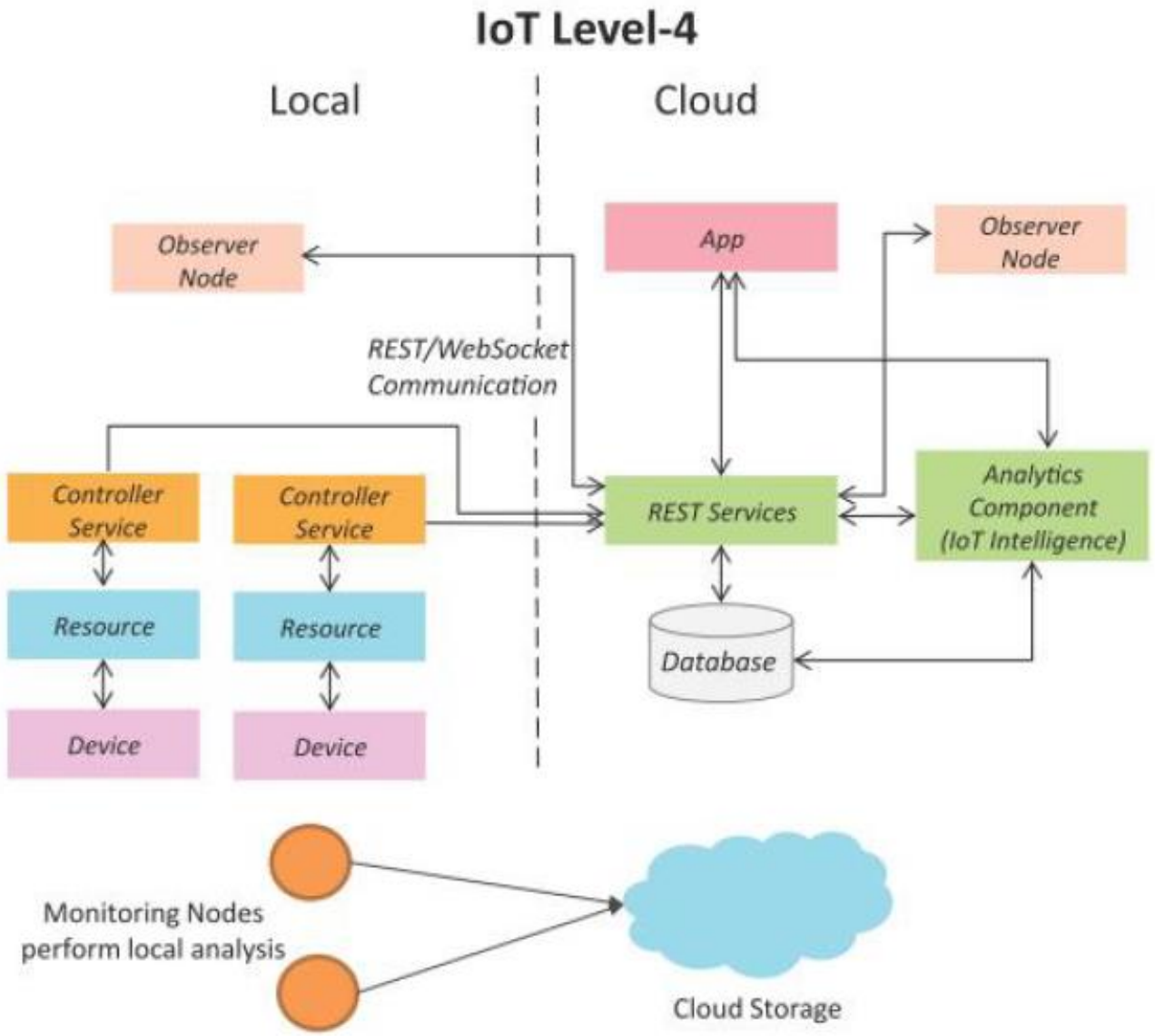
IoT – Livello 3

- Un sistema IoT di livello 3 ha un **singolo nodo** (dispositivo) che acquisisce dati e/o comanda gli attuatori.
- I dati vengono memorizzati ed analizzati nel cloud e **l'applicazione è cloud-based**.
- I sistemi di Livello 3 sono adatti per soluzioni con **molti dati** ed in cui **l'attività di analisi** è computazionalmente **pesante**.



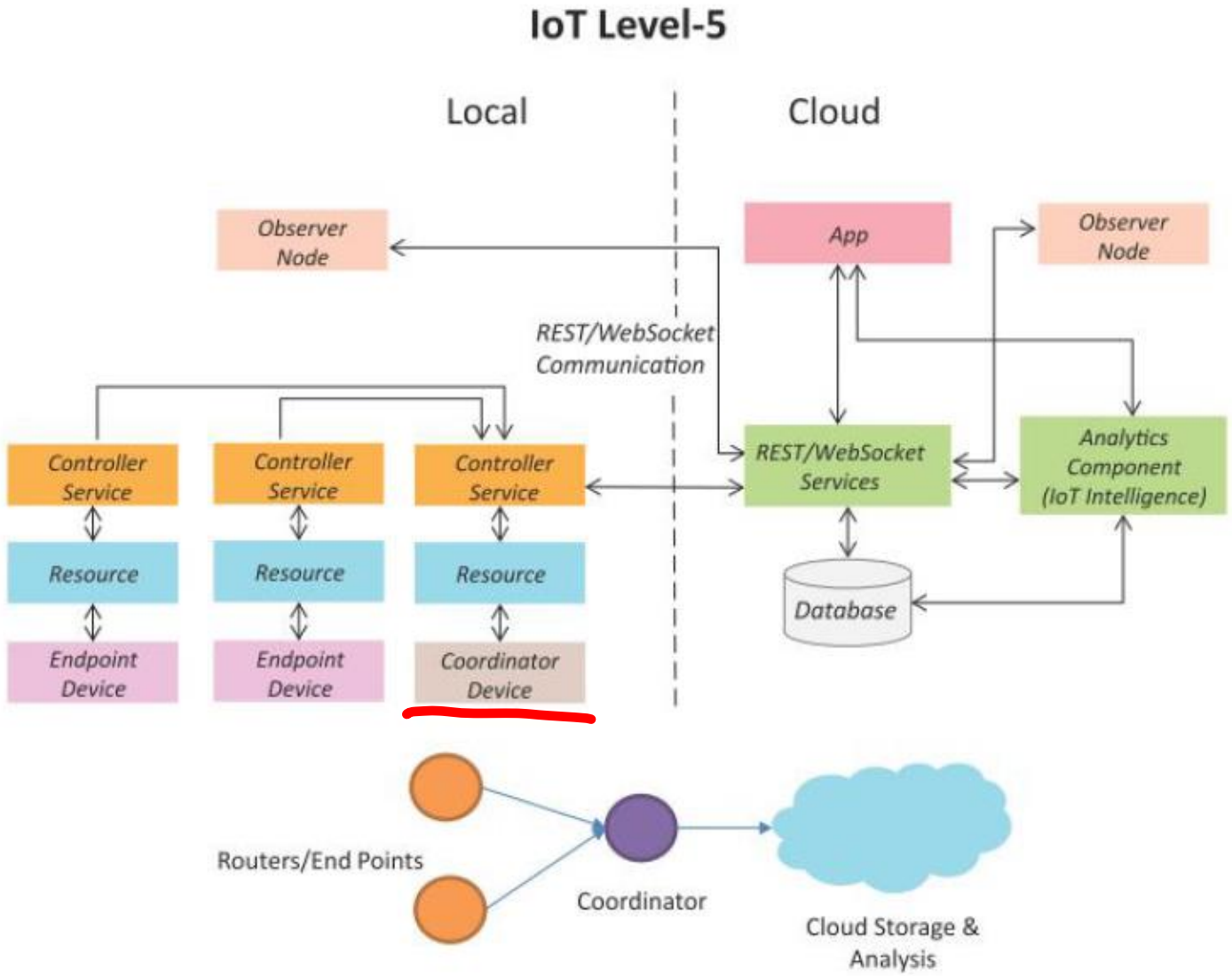
IoT – Livello 4

- Un sistema IoT di livello 4 ha **diversi nodi** (dispositivi) che acquisiscono dati e/o comandano gli attuatori e compiono attività di analisi dei dati.
- I dati vengono memorizzati nel cloud e **l'applicazione è cloud-based**.
- Un sistema di Livello 4 ha anche dei **nodi osservatori** (sia locali che nel cloud): essi possono iscriversi e ricevere informazioni gestite nel cloud a partire dai dati forniti dai dispositivi IoT.
- I sistemi di Livello 4 sono adatti per soluzioni che richiedono **molti nodi**, con **molti dati** ed in cui **l'attività di analisi** è computazionalmente **pesante**.



IoT – Livello 5

- Un sistema IoT di livello 5 ha **diversi nodi** (dispositivi) che acquisiscono dati e/o comandano gli attuatori ed un nodo coordinatore.
- Il nodo coordinator raccoglie i dati dagli altri nodi e li invia nel cloud.
- I dati vengono memorizzati nel cloud e **l'applicazione è cloud-based**.
- Un sistema di Livello 5 ha anche dei **nodi osservatori** (sia locali che nel cloud): essi possono iscriversi e ricevere informazioni gestite nel cloud a partire dai dati forniti dai dispositivi IoT.
- I sistemi di Livello 4 sono adatti per soluzioni basate su **reti di sensori wireless** con **molti dati** ed in cui **l'attività di analisi** è computazionalmente **pesante**.



Aws

IoT – Livello 6

- Un sistema IoT di livello 5 ha **diversi nodi** (dispositivi) indipendenti che acquisiscono dati e/o comandano gli attuatori ed inviano i dati direttamente nel cloud.
- I dati vengono memorizzati nel cloud e **l'applicazione è cloud-based**.
- I componenti di analisi elaborano i dati e memorizzano i risultati nel database distribuito nel cloud.
- I risultati vengono visualizzati tramite l'applicazione cloud.
- Il **controllore centralizzato** conosce in tempo reale lo stato di tutti i nodi ed invia loro i comandi di controllo.