

01: INTRODUZIONE

Letteralmente "Internet delle cose", ovvero oggetti in grado di interagire con l'ambiente reale e connessi dalla stessa infrastruttura di rete che abbiamo usato finora per il web.

02: APPLICAZIONI

Le applicazioni riguardano:

- **DISPOSITIVI** (con sensori e ricevitori/trasmettitori).
- **NETWORK BACKBONE** (collega i componenti).
- **APPLICAZIONI SOFTWARE** (elaborano dati e prendono decisioni).

I sensori si dividono in:

- **SENSORI** = input (monitoraggio).
- **ATTUATORI** = output (modificano).

Questi si dividono in **SEMPLICI** e **INTELLIGENTI** (con unità di calcolo e memoria).

Una rete può essere:

- **BIG THREE** (hub and spoke) = rete cellulare, wi-fi e BT.
- **MESH NETWORK** (peer to peer).
- **SOLUZIONI BRUTTATEL/CLOUD** = forse la più usata.

Tutti i dati generati dovranno essere raccolti, memorizzati e analizzati, ed è una grande sfida.

Le aree principali sono:

- **HOME** (Consumer): domotica ...
- **ENTERPRISE**: smart office ...
- **GOVERNMENT**: smart medicine ...

03: GPS TRACKER

Dispositivi USB → dispositivi seriali (adattatori o emulatori).

FORMATO NMEA: National Marine Electronics Association definisce e mantiene gli standard di comunicazione fra i dispositivi elettronici a bordo di un veicolo marino (fasi dei GPS). Aspettare sempre la fine dei segnali.

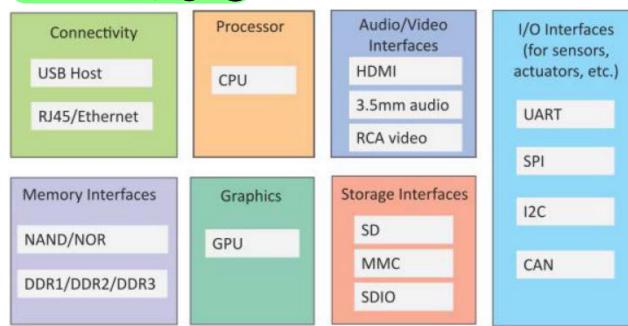
04: PROTOCOLLI E ARCHITETTURA

I dispositivi sono:

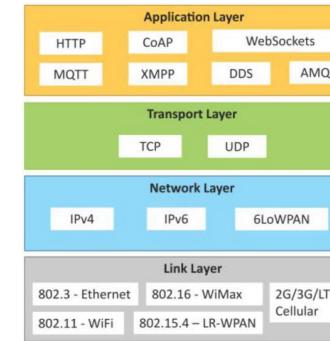
- **DINAMICI** = si adattano all'ambiente
- **AUTO-CONFIGURANTI**
- Usano **PROTOCOLLI DI COMUNICAZIONE INTEROPERABILI** → **SISTEMI ETEROGENEI**
- Ogni **NODO** ha **IDENTITÀ + ID**
- **INTEGRATI**

Le "cose" si riferiscono a dispositivi con identità univoca in grado di effettuare azioni di acquisizione dati da sensori, di azionamento di attuatori e di monitoraggio.

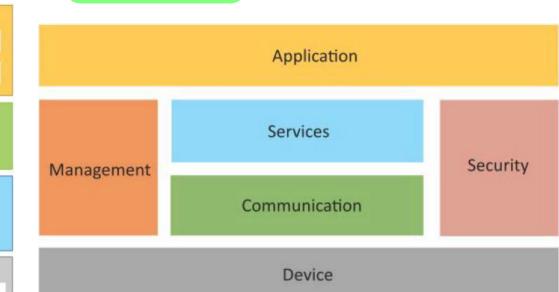
INTERFACCE



PROTOCOLLI



BLOCCHI



I modelli di comunicazione possono essere:

- **REQUEST-RESPONSE** = Client ↔ Server ↔ Resources
- **PUBLISH-SUBSCRIBE** = Publisher → Broker → Customers
- **PUSH-PULL** = Publisher → Queues → Customer
- **EXCLUSIVE PAIR** = Client ↔ Server (\approx WEB SOCKET)
- **REST** =uso di WES Server e WEB API

05: PUBBLICAZIONE E COMUNICAZIONE DEI DATI

Uso delle SOCKET:

- Programmazione troppo a basso livello.

- Difficilmente espandibile e adattabile.

Protocollo MQTT (Message Queuing Telemetry Transport):

- Protocollo leggero di tipo publish/subscribe.

- Connessione bidirezionale, ordinata e senza perdita di dati.

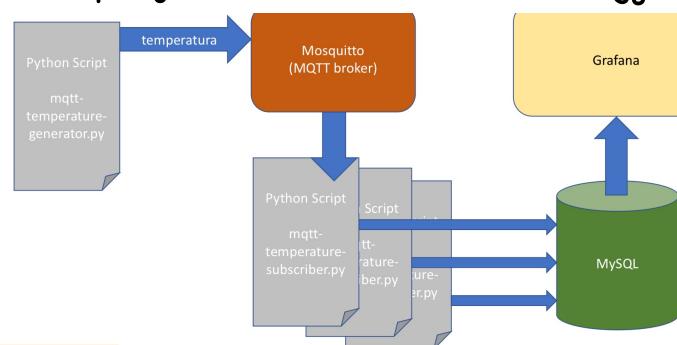
- Le entità sono: Broker, Publisher, Subscriber.

- I messaggi sono: Connect, Disconnect, Publish, Subscribe.

- La misura della qualità (QoS): At most once, At least once, Exactly once.

06: PRESENTAZIONE DEI DATI

GRAFANA: soluzione open source per gestire l'analisi ed il monitoraggio dei dati provenienti da un ampio spettro di database.



07: ARDUINO E SENSORI

Arduino: piattaforma open source di prototipazione elettronica.

Per l'utilizzo con Arduino bisogna convertire i segnali continui in segnali digitali tramite l'ADC.

Led: si serve della capacità di alcuni semiconduttori di produrre fotoni attraverso un fenomeno di emissione spontanea.

Resistori: componente elettrico che oppone una specifica resistenza.

Fotoresistore: componente la cui resistenza è inversamente proporzionale alla quantità di luce che lo colpisce.

Pulsante: converte la pressione esercitata in segnale elettrico.

Potenziometro: portatore di tensione regolabile, permette di ottenere una resistenza variabile.

Servomotore: motore che può ruotare di 180 gradi e gli impulsi ricevuti dicono al motore quale posizione assumere.

Motore a CC: converte energia elettrica in energia meccanica.

08: ARDUINO E PYTHON

FIRMATA

Protocollo generico per la comunicazione su porta seriale fra microcontrollori e computer host. Può funzionare con i più diffusi linguaggi di programmazione per quanto riguarda l'host. Consente di controllare il microcontrollore da un computer, mediante un firmware comune.

INFLUXDB

E' un TSDB, ovvero un Time Series Data Base. Ottimizzato per la gestione efficiente di dati organizzati come serie temporali. E' un database NoSQL e memorizza i dati come punti.

MISURAZIONE: Es. Temperatura

TAG: Indicano come i dati vengono indirizzati e ricercati (coppia chiave / valore).

CAMPIONI: indicano i dati che vengono memorizzati.

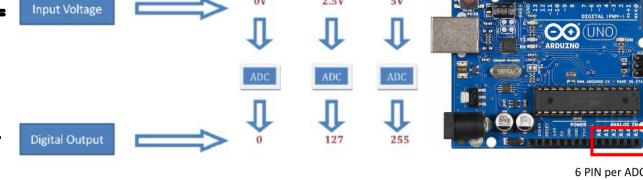
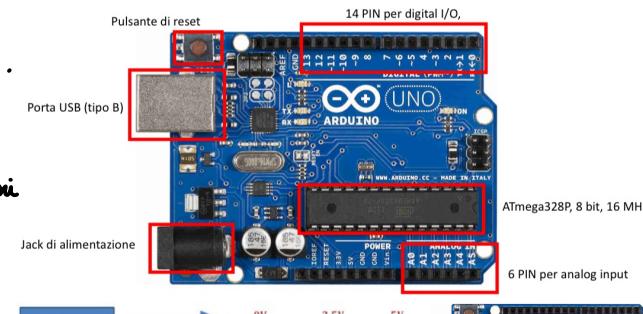
POLITICA DI CONSERVAZIONE: per quanto tempo e come conservare i dati.

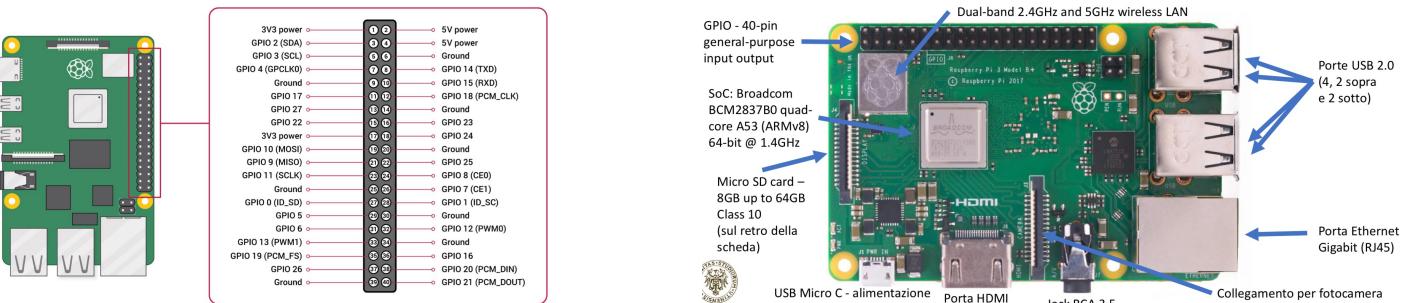
SERIE: Raccolta di dati che condividono una misurazione, un tag set e una chiave di campo.

INSIEME: Campioni con timestamp simboli e dati di campo arbitrari.

09: RASPBERRY

E' un single board computer che permette di utilizzare un computer completo in praticamente qualunque situazione, dati gli ingombri ridotti ed i bassi consumi.





10: DISTRIBUZIONE DEL SOFTWARE IOT

DOCKER

Piattaforma che consente la distribuzione del software e dell'ambiente utile al suo funzionamento in unità chiamate container. Questi contenitori consentono agli sviluppatori di "imballare" un'applicazione con tutte le parti di cui ha bisogno, come librerie e altre dipendenze in un unico pacchetto. Consente di isolare le applicazioni l'una dall'altra.

VIRTUALIZZAZIONE: astrarre le componenti hardware (macchina virtuale).

CONTAINERIZZAZIONE: virtualizzazione a livello di sistema operativo, si riferisce a una funzionalità del sistema operativo in cui il kernel consente l'esistenza di più istanze isolate dello spazio utente.

CONTAINER VS VIRTUAL MACHINE



DOCKER CONTAINER



IMMAGINE: Modello di sola lettura contenente le specifiche per la creazione di un contenitore.

ENGINE: Applicazione client-server.

REGISTRY: Registro Docker che memorizza le immagini Docker.

12: ARCHITETTURA E PROGRAMMAZIONE 2

Sistemi IoT: garantire un supporto affidabile e continuo di applicazioni aziendali. Focus sui dati generati dai sensori e il modo in cui essi vengono comunicati, raccolti, analizzati e usati.

Scalabilità: milioni di dispositivi nella stessa rete.

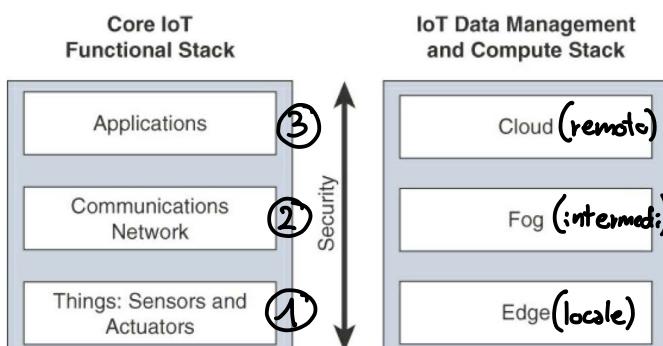
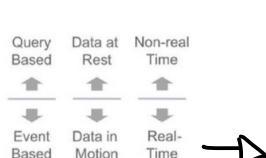
Sicurezza: organizzati in una rete di sensori wireless e quindi fisicamente esposti ad attacchi / intercettazioni.

Limitazioni: potenza computazionale molto limitata e specializzata, connettività spesso inaffidabile con basse velocità di trasmissione dati.

Volume di dati: quantità di dati elevata che causa problemi alla rete.

Supporto legacy: i sistemi sono composti da vecchi dispositivi legacy affiancati a quelli moderni.

ARCHITETTURA IoTWF



L'architettura IoTWF viene poi semplificata in modo da comprendere i principi chiave di progettazione dei casi d'uso specifici del settore.

Vantaggi: separa chiaramente due componenti che possono tranquillamente funzionare in parallelo. Modella esplicitamente il fatto che il calcolo sia distribuito tra diversi livelli. Estende la sicurezza a tutti i livelli.

Stack Funzionale: considera come i dati vengono acquisiti, trasmessi e analizzati.

Stack di gestione ed elaborazione: focalizzata sui dati e come vengono elaborati (su 3 livelli).

LIVELLO 1

I sensori/attuatori sono classificati in base alla fonte di alimentazione (cavo o batteria), dalla mobilità, dalla frequenza di invio dei dati, dal tipo di dati, dalla portata, dalla densità dell'oggetto per cella o dall'affidabilità.

LIVELLO 2

E' composto da 4 sottolivelli:

- Accesso alla rete: aspetti di basso livello delle connessioni.
- Gateway e backhaul: il gateway è il dispositivo che raccoglie i dati dagli smart object nell'ambiente. Inoltre i suoi dati su un supporto (la rete di backhaul), per raggiungere il punto di elaborazione principale.
- Trasporto: protocolli per garantire la consegna a destinazione dei pacchetti di dati.
- Gestione di alto livello: protocolli di alto livello di tipo applicativo.

LIVELLO 3

Controlla le applicazioni e analizza i dati e la rete.

13: NODE-RED

Node-RED: framework che consente di collegare in un flusso informativo dispositivi e servizi software, riducendo considerabilmente la quantità di codice necessario. Queste azioni possono essere compiute in modo visuale con un editor che gira all'interno di un browser web. Utile per lo sviluppo di prototipi.

14: SISTEMI AUTONOMI E DRONI

SISTEMI AUTONOMI

Si definisce autonomo un sistema in grado di eseguire automaticamente in dato compito, eventualmente adattando il proprio comportamento mediante autocapprendimento.

DRONI

Termino associato a veicoli aerei senza pilota (VPR: veicoli a pilotaggio remoto).

MAVLink: protocollo di comunicazione.

MAVProxy: software a livello di comando che funge da Ground Control Station (GCS).

ArduPilot: software open source per il controllo remoto di veicoli.

Mission Planner: software che svolge funzioni di GCS, configuratore, calibratore... Si appoggia a MAV.

ROS: framework per agevolare la programmazione di robot.

PROGRAMMAZIONE

Col modulo python Dronekit i comandi vengono tradotti in messaggi MAVLink e inviati a MAVProxy.

ASPECTI NORMATIVI E PRATICI

Drone che pesa meno di 250 grammi: nessuna burocrazia (a meno che non possa ottenere dati personali).

Drone che pesa più di 250 grammi: necessaria registrazione (D-Flight) e superore in esame.

Assicurazione obbligatoria.

CLASSI DI DRONI

C0: 250g, 50m, 24V

C1: 900g, 120m, 70km/h, max 5MP, 24V, 80dB.

C2: 4kg, 24V, 120m, geofencing

C3: 25Kg, 48V

C4: Altri tipi, 25kg

15/16: ROBOT OPERATING SYSTEM

Framework per agevolare la programmazione di robot. E' una raccolta di strumenti, librerie e convenzioni. Un robot è un dispositivo dotato di sensori, motori e unità computazionale.

Caratteristiche: Offre servizi simili a quelli di un S.O.

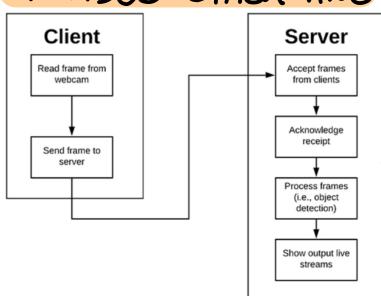
ROS equation: PLUMBING = message passing interface.

Comunicazione tra processi: All'inizio comunicano al ROS Master tutti i dettagli (messaggi ROS) sui canali ROS topic. Il ROS Master funge da broker, smista i messaggi con il ROS topic desiderato dai publisher ai subscriber.

File system: packages, meta packages, package manifest, repositories, message and service type descriptions.

- Concetti computazionali:**
- **nodes**: processi.
 - **master**: processo di intermediazione tra nodi.
 - **parameter server**: processo che affianca il master.
 - **topics**: bus di comunicazione tra nodi.
 - **message**: messaggi.
 - **Service**: meccanismo request / reply
 - **bags**: metodo per memorizzare ed imprimere nuovamente ROS topics e per logging dei dati generati.

17: VIDEO STREAMING



Client: video streamer in formato OpenCV. I suoi compiti saranno di catturare i fotogrammi dalla fotocamera e inviare i frame al server sulla rete tramite la libreria Image2MQ.

Server: accetta i frame in arrivo dal client, applica una routine di rilevamento di oggetti ad ogni frame, mantiene un conteggio degli oggetti per ciascuno dei frame e visualizza i frame a video.

Framework Caffe: Framework per il deep learning disponibile nei moduli di OpenCV.

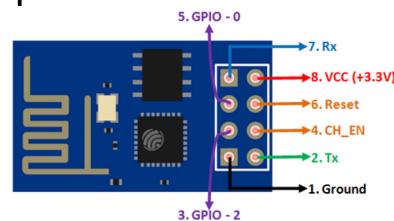
18-19: SISTEMI A BASSO CONSUMO-ESP266

Microcontrollore con Wi-Fi integrato e supporto completo del TCP/IP.

Ha un firmware che fornisce di default un interprete di comandi AT, per usare il prodotto come Access Point o Client in una rete.

Per la comunicazione seriale con Arduino non si possono usare i pin TX-RX, bisogna prima impostare una velocità più bassa (9600 bps).

L'utilizzo di codice specifico risulta molto più veloce, e le librerie di supporto sono molte e ad ampio spettro.



20: ARDUINO E LORAWAN

Lorawan: permette la comunicazione verso un network server tramite gateway.

Gli **scenari applicativi** sono: l'**agricoltura**, la **comunicazione punto-punto**, **caveat**, **sincronizzare uno sciame di droni**.