[[info - Internet of Things]] [[progetto - Internet of Things]]

Lezione 1 - Intro

Origini

Termine coniato da Kevin Ashton nel 1999. Usa il termine per descrivere il collegamento di oggetti a internet tramite RFID In realtà nasce ancora prima. Tra le prime invenzioni ci sono: - Codice a barre - Wearable computer - Head mounted display (bike simulator)

Nel 1969 nasce ArpaNet, precursore di internet. 1973 nasce RFID passivo, leggibile e scrivibile.

> [!abstract]+ **Definizione** IoT is a system of interrelated computing devices, mechanical and digital machines, objects, animals or people that are provided with unique ids (UIDs) and the ability to transfer data over a network without requiring human-to-human or human to computer interaction.

“Things” Caratteristiche: 1. Devono avere un ricevitore / trasmettitore 2. Devono possedere un IP univoco 3. Devono essere dotati di un sensore o di HW che svolge qualche compito 4. Deve avere la capacità di Store-and-forward (mem interna) 5. Low-power e low-bandwith Quindi possono essere anche entità animate dotate di impianti.

Lezione 2 - GPS

Sensore: ricevitore GPS Procedura: - Connessione al sensore tramite porta USB (seriale) oppure simulazione software del sensore - Lettura dati in formato **NMEA 0183** - Parsing dei dati per estrazione delle informazioni di interesse - Logging dei dati su filesystem - Creazione di un **socket** server multi thread per la diffusione delle informazioni di interesse in formato JSON - Utilizzo delle API di Google Maps per visualizzare su mappa la posizione rilevata.

Lettura dati Usando Python con modulo pySerial si accede alla porta seriale (USB) e si leggono i dati. #### Emulazione ![[Pasted image 20250318161542.png#invert|center|500]]

Formato NMEA

National Marine Electronics Association. Solitamente usato da ricevitori GPS (in versione 0183 o 2000). Standard di comunicazione fra dispositivi a bordo di veicoli marini

Ci interessano due tipi di dati: GNRMP: Presenza simultanea di più costellazioni GNSS. GPRMC: Usa solo una connessione a GPS. #### Salvataggio dati Conviene implementare il salvataggio in un thread separato, entra in funzione

periodicamente ogni x secondi prendendo i dati da scrivere dalla lista (struttura dati)

API di Google Maps

Con Xampp si costruisce una piccola webapp che si connetta alla socket per recuperare le coordinate dell'oggetto.

Lezione 3 - Applicazioni

Dispositivi (smart devices): con sensori, ricevitori/trasmettitori, raccolgono e inviano dati. **Network Backbone** collega i componenti **Applicazione software** elabora i dati e prende decisioni

Sensori vs Attuatori

Sensori → input, monitora - Semplici o composti Attuatori → Output, modifica ![[Pasted image 20250318163048.png]]

Lezione 4 - Pubblicazione (comunicazione) dati

I dati vengono scritti in un socket server, tramite UDP o TCP. Le socket sono usate come meccanismo di comunicazione tra sensori e attuatori per inviare i dati al server. Metodo di basso livello difficilmente espandibile.

In Python, occorre importare `socket`.

MQTT Message Queueing Telemetry Transport

Protocollo leggero che si fonda su TCP/IP. Entità coinvolte sono: - **Broker** - Intermediario centralizzato - Server che gestisce il flusso di messaggi. Riceve messaggi dai publisher e li inoltra ai subscriber - **Publisher** - Invia messaggi al broker - Client che invia (pubblica) i messaggi a uno o più topic. Non conosce i destinatari (subscriber) ma invia semplicemente i dati al broker. - **Subscriber** - Ricevono dal broker solo i messaggi rilevanti. - Client che si iscrive a uno o più topic per ricevere messaggi di suo interesse. Quando il broker riceve un messaggio su un topic a cui il subscriber è iscritto, glielo inoltra.

I *topic* possono essere semplici (temperatura) o strutturati (casa/cucina/forno/temperatura)

Ogni connessione al broker può specificare una QoS: - at most once - messaggio inviato una volta e non ci sono conferme - At least once - Si riprova a inviare il messaggio fino quando non viene ricevuto correttamente - Exactly once - C'è garanzia che il messaggio venga ricevuto una sola volta.

Il broker usato è Eclipse Mosquitto, ideale per piccoli progetti e prototipi. mosquitto.org/download

Lezione 5 - Architettura e Protocolli

Le "cose" si riferiscono a dispositivi con ID univochi in grado di acquisire dati da sensori, azionare attuatori e monitorare. **Sense** $\longrightarrow$ **Infer** $\longrightarrow$ **Act**

Diagramma a blocchi

Dispositivi IoT integrano varie interfacce per collegarsi ad altri dispositivi: - I/O per sensori - Connessione a Internet - Memorizzazione / storage - Audio / video

`UART` = Universal Async Receiver - Transmitter `SPI` = Serial Peripheral Interface `I2C` = Inter-Integrated Circuit `CAN` = Controller Area Network

Protocolli per l'IoT

 A livello data link: - IEEE 802.3 = Ethernet con cavo coassiale o duplex o fibra ottica - IEEE 802.11 = WiFi - IEEE 802.16 = WiMax - 2G/3G/4G/5G = Mobile communications

LoRaWAN Long Range WAN basata su bande ISM (Industrial, Scientific, Medical) = bande libere quindi diverse da 2.4GHz, 5GHz, 433MHz...

Usato per comunicazioni a lungo raggio e basso consumo

Una rete LoRaWAN è costituita da: - *End nodes*: i dispositivi IoT - *Gateway*: Antenne intermediarie che ricevono e inviano i dati da End Nodes a Network Server. - *Network Server*: Gestisce la rete - *Application Server*: le applicazioni finali che processano i dati.

Esistono 3 classi di dispositivi: - **Classe A (All)** - Alimentati a batteria, basso consumo. - Dopo l'invio dei dati al gateway viene aperta una piccola finestra per eventuale ricezione di comandi da parte del server. - Es: Sensori ambientali - **Classe B (Beacon)** - Alimentati a batteria, offre finestre sincronizzate per la ricezione di comandi - **Classe C (Continuous)** - Sempre in ascolto - Alimentati da rete elettrica - ES: sistema di allarme

Protocolli a livello di rete IPv4: 32 bit IPv6: 128bit 6LoWPAN: 2.4GHz in combinazione con il protocollo di livello data link (=algoritmi di compressione per IPv6)

Protocolli a livello di trasporto **TCP:** orientato alla connessione con stato, garantisce ordinamento dei pacchetti eliminazione dei duplicati e ritrasmissione di pacchetti persi. **UDP:** Adatto per applicazioni che necessitano di comunicazioni veloci, senza necessità di setup del canale.

InfluxDB

è un TSDB, Time Series DataBase, ottimizzato per la gestione efficiente di dati organizzati come serie temporali, ideale per op. IoT. è un DB **NoSQL** e memorizza i dati come punti. Ogni punto è un insieme di coppie chiave-valore e un timestamp. *serie* = insieme di punti raggruppati in base a un target (= insieme di coppie chiave-valore). Sono raggruppate da un identificatore di misurazione.

- Misurazione
  - simile al concetto di tabella nei DB classici. Contenitore per tag e campi.
- Tag
  - Indicano come i dati vengono indicizzati e ricercati
  - `source = 'Arduino Uno'`
- Field
  - Indicano i dati che vengono memorizzati
- Retention Policy
  - Per quanto tempo conservare i dati e cosa farne quando invecchiano.
  - es: "on_day_by_hour": durata di 1 giorno con frammento di 1h.
- Series
  - Raccolta di dati che condividono una misurazione, un tag set e una chiave di campo.
- Set
  - Contiene campioni con timestamp univoci e dati di campo arbitrari.

Installazione InfluxDB

Seguire step nelle slide 8. Per accedere alla webapp: `localhost:8086`

API Token:

`NNptjHbOItkq5VTQSQd7HEx8rQPulbOWxxXj-mOdz4aRcXDcNcIw7qflY2rsgmcoZ42bJSUnTC1vjjydKIkwwQ==`

API Token clonato:

`-C-HVISKeMNEdqjOGLcgvp5bn5yoKpPz3fQKxUHn3vl_kw1mqzwUhRCa6K4HSLFfA3EaSntvgEIqIHmhNrb66A==`

> [!bug]+ **Bug** Il file weather-monitor.py non compila perchè non riconosce la librerie `serial_port`. Provare a usare pyserial e riscrivere il codice

RaspberryPI

Versione usata: `Raspberry Pi 3 Mod. B+` Utente: `progetto` Psw: `progetto`

Setup

1. Preparazione micro-SD con Raspberry Pi Imager. Selezionare il modello di Raspberry e il tipo di OS. Poi impostare nome utente, psw, hotspot WiFi, abilitare SSH.
2. Setup Software del sistema:
    1. GUI
        1. Avendo a disposizione connessione HDMI e periferiche
    2. Headless
        1. Inserire SD, collegare alimentazione, connettersi via SSH
        2. Primo setup
            1. Trovare IP del Rasp. guardando i dispositivi connessi nel dispositivo dell'hotspot.
            2. `ssh pi@<IP-Raspberry_pi>`
            3. Al primo accesso con utente `pi` e pws `raspberry` verrà eseguito uno script di setup.

Per vedere il pinout del Raspberry: `pinout` Gli script si scrivono in Python e vengono eseguiti direttamente dalla shell via SSH: `python file.py`.

```
import RPI.GPIO as GPIO
GPIO.setmode(GPIO.BCM) # Broadcom SOC Channel Number, ci si riferisce ai pin con quei numer
GPIO.setup(18, GPIO.OUT)
GPIO.output(18, True) # accende il led
GPIO.output(18, False) # spegne il led
GPIO.cleanup()
```

![[Pasted image 20250408164717.png#invert|left|300]] ### PWM I pin utilizzabili sono GPIO12/GPIO18 o GPIO13/GPIO19.

Docker

Piattaforma che consente la distribuzione di software e dell'ambiente utile ad esso in unità dette *container*.

Virtualizzazione

> possibilità di astrarre le componenti hardware degli elaboratori al fine di renderle disponibili al software in forma di risorsa virtuale.

Quindi è possibile installare sistemi operativi su hardware virtuale —→ macchine virtuali.

Per gestire le macchine virtuali serve un HyperVisor (VMware, Virtual Box...) Ogni VM richiede tante risorse ed è lenta ad avviarsi. ### Containerization > virtualizzazione a livello di sistema operativo. Il kernel consente l'esistenza di più istanze isolate dello spazio utente (i container).

Questi container, partizioni, ambienti virtuali, jail, possono apparire come veri computer dal punto di vista dei programmi in esecuzione. I programmi in esecuzione dentro un container possono vedere solo i contenuti e i dispositivi assegnati ad esso.

I container: - Condividono lo stesso OS dell'host. - Sono leggeri, occupano poche risorse e si avviano velocemente.

Docker Image

Immagine = modello di sola lettura contenente le specifiche per la *creazione* di un container. (template) Di solito si crea con un Dockerfile, un file di testo tipo:

```
FROM python:3.11-slim
COPY app.py /app/app.py
CMD ["python", "/app/app.py"]
```

Da cui si crea il container con:

```
docker build -t immagine .
```

Docker Engine

Composto da 3 parti: #### Docker Daemon Esegue i container, gestisce le immagini, reti, volumi. è un *processo* che in background ascolta le richieste. #### Docker CLI Interfaccia da terminale #### Rest API Il modo in cui la CLI comunicano con il daemon. Usata da strumenti esterni per controllare Docker via codice.

Docker Registry

Dove vengono memorizzate le immagini Docker.
https://hub.docker.com/ è quello usato di default nei comandi tipo `docker pull | docker run`

Esempio:

`docker run ubuntu echo Hello World` → "hello world" Altri comandi utili: `docker images` `docker ps -a`

Node-RED

https://nodered.org

Framework per collegare flusso informativo di dispositivi e servizi software riducendo la quantità di codice necessario usando un editor node based che gira su browser web. Incluso di default nelle immagini di Raspbian.

è possibile collegare Arduino al Raspberry o al PC per usarlo con Node-RED, usando la libreria StandardFirmata `npm install node-red-node-arduino`

Sistemi Autonomi e Droni

- Beccheggio = Pitch - Rollio = Roll - Imbardata = Yaw

Struttura di un Drone

- Telaio
  - Alcuni hanno una PDB (Power Distribution Board)
- Motori
  - Brushless, hanno un verso di rotazione.
- ESC
  - Electronic Speed Controller
  - Da collegare alla PDB e alla centralina di volo. Fornisce ai motori la giusta potenza regolando V e A
- Eliche
  - Passo = Pitch = distanza percorsa con una singola rotazione
  - Eliche lunghe con passo maggiore spostano più aria → sono più efficienti ma generano più attrito
  - Eliche corte → motori ad alto RPM
- MCU (centralina)
- Power Module
- Batteria
  - Solitamenete LiPo
  - Numero di celle
    - ogni cella da 3V a 4.2V
    - Tensione per stoccaggio: 3.8V
  - Capacità in mAh
  - Fattore di scarica (30-40C) = moltiplicatore massimo della capacità con cui è possibile scaricare la batteria in sicurezza.
- Radiocomando / ricevitore
- Sensori / accessori
  - Barometro
  - IMU
  - Bussola
  - GPS
  - LIDAR
  - Camera + gimbal

Software di controllo

MAVLink

Protocollo di comunicazione

Pilotare un drone tramite software

![[Pasted image 20250508135400.png]] I comandi python vengono tradotti in messaggi MAVLink e inviati a MAVProxy che li inoltra al drone.