1506
UNIVERSITÀ DEGLI STUDI DI URBINO CARLO BO
CORSO DI LAUREA IN INFORMATICA APPLICATA
SCUOLA DI SCIENZE TECNOLOGIE E FILOSOFIA DELL'INFORMAZIONE

Esame di Internet of Things

Stazione di rilevamento temperatura ed umidità

Paride Dominici

Scopo del progetto

- Idea iniziale
  - Contabilizzatore di calore
- Idea definitiva
  - Comparazione efficienza termosifone Vs pompa di calore

Contabilizzatore di calore/1

La contabilizzazione del calore può essere effettuata nel modo seguente:

1. Si fissa una temperatura minima per considerare che il termosifone sia in funzione
2. Si fissa la differenza di tempo per far avanzare il conteggio.
3. Quando la temperatura rilevata supera la temperatura minima vuol dire che il termosifone è acceso e quindi iniziamo a contare.
4. Finché la temperatura rimane al di sopra della temperatura minima ogni delta di tempo facciamo avanzare il numero di elementi contati.

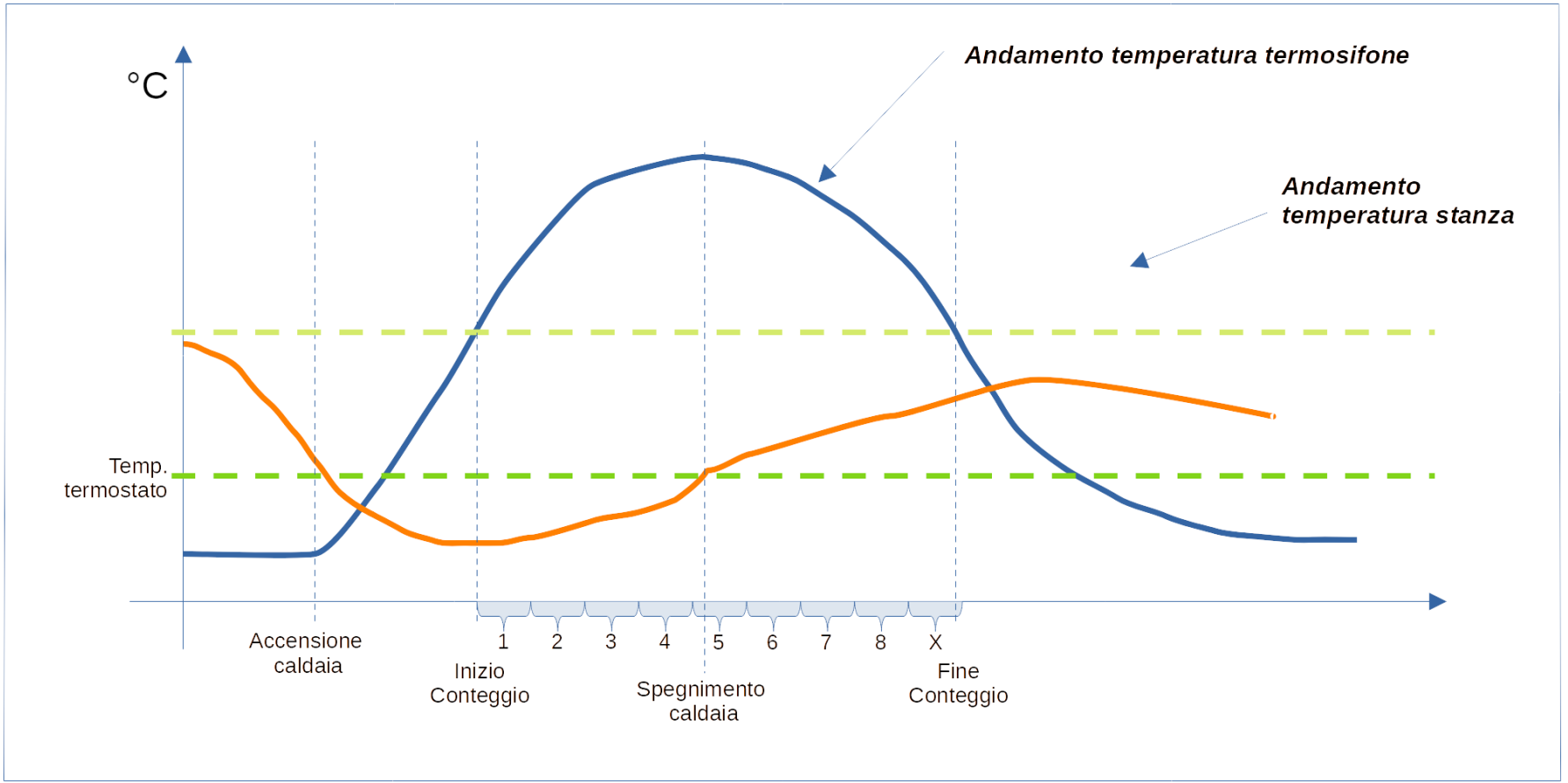
Contabilizzatore di calore/2

Caratteristiche di un contabilizzatore commerciale:

Conteggio

Funzionamento a due sensori e commutazione ad un sensore in presenza di accumulo interno di calore.

ΔT di commutazione:	4,5 K
Temperatura (media di piastra) di inizio conteggio a un sensore:	28°C
Ciclo di conteggio:	2 minuti


°C
Andamento temperatura termosifone
Andamento temperatura stanza
Temp. termostato
Accensione caldaia
Inizio Conteggio
1
2
3
4
5
6
7
8
X
Spegnimento caldaia
Fine Conteggio

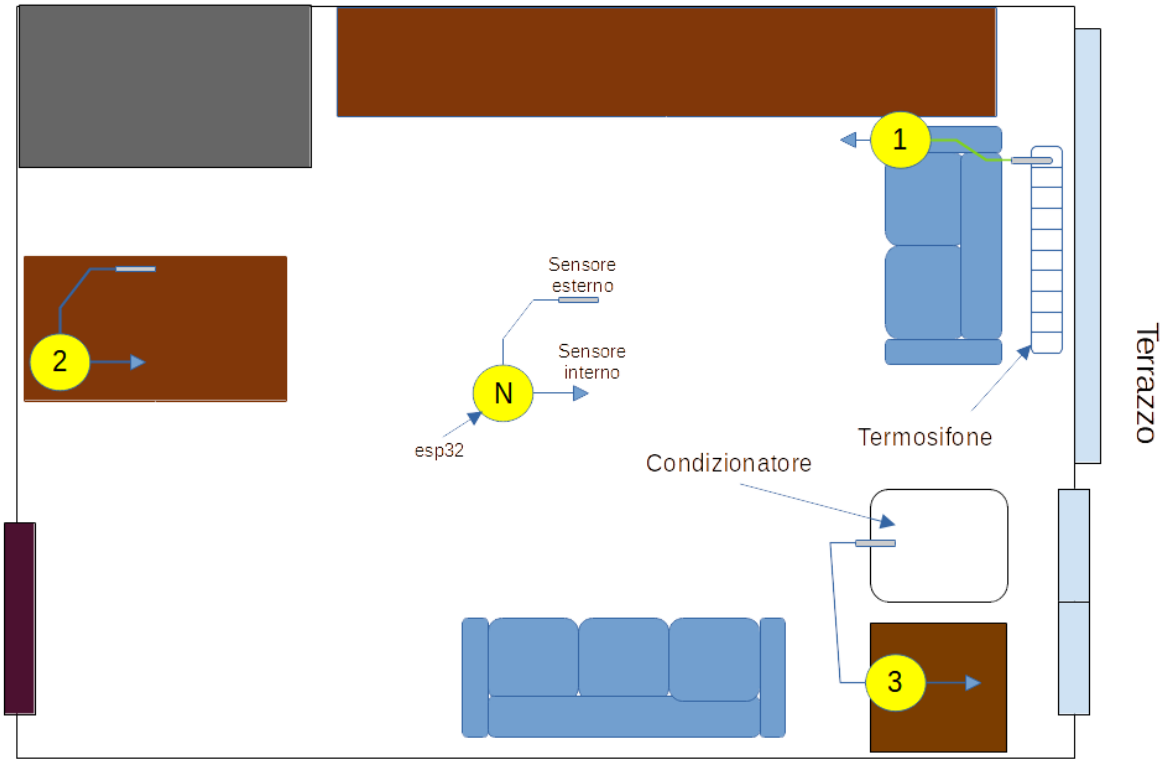
Comparazione efficienza di un termosifone contro efficienza di una pompa di calore

Usiamo i differenti moduli per misurare le temperature del termosifone (idea iniziale) e dell'ambiente circostante e le confrontiamo con quelle registrate dagli altri due moduli.

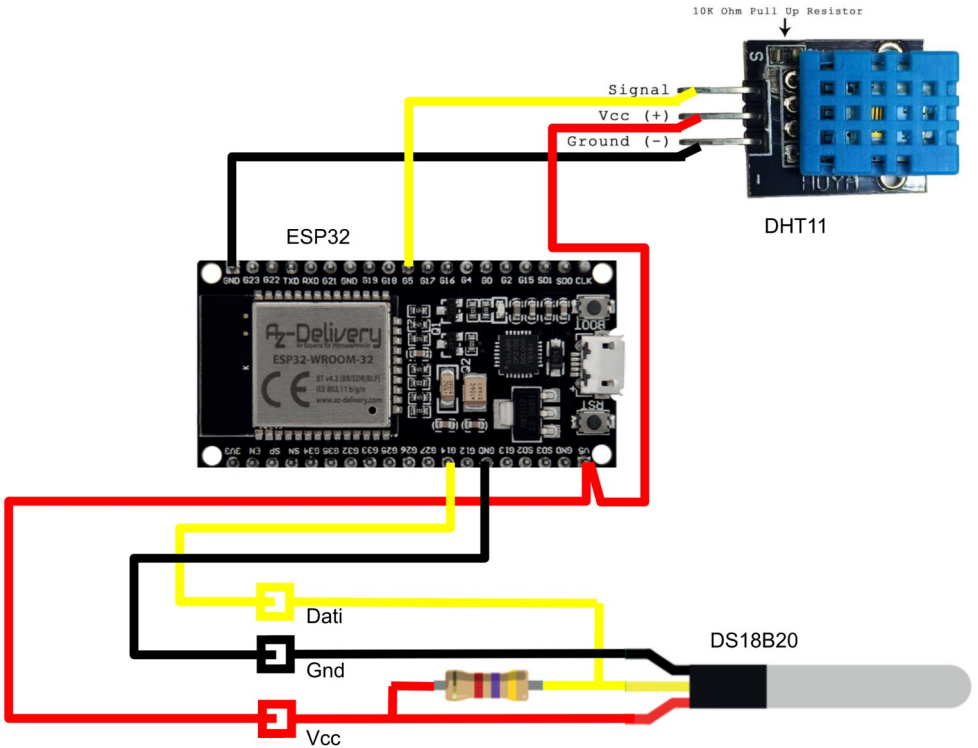
Nello specifico facciamo un confronto tra l'efficienza del termosifone e quella di una pompa di calore.

Il modulo lontano dai primi due serve come riferimento.

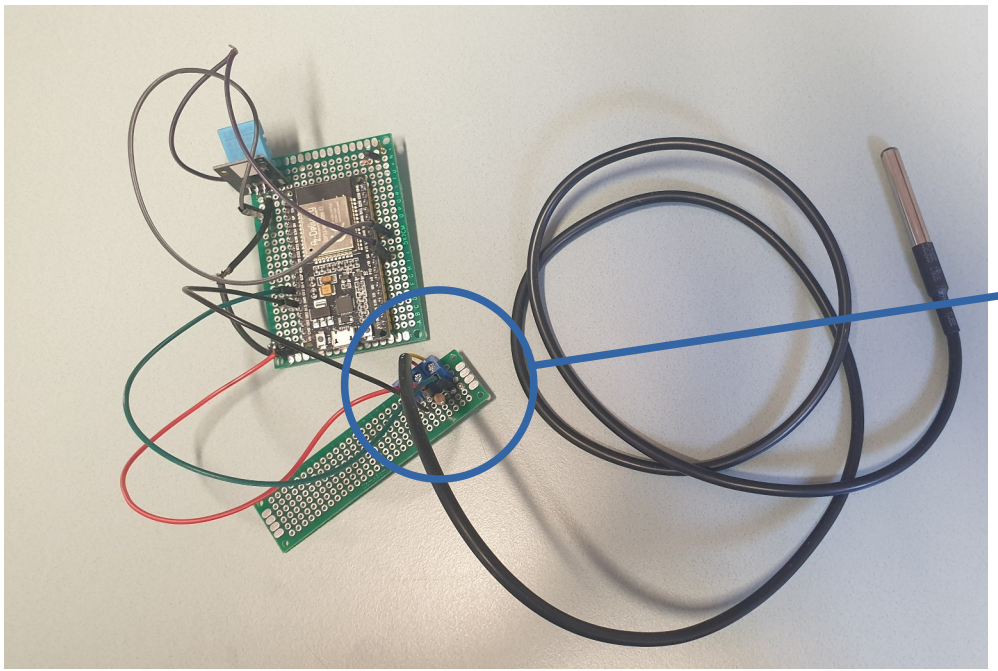
Ambiente del test

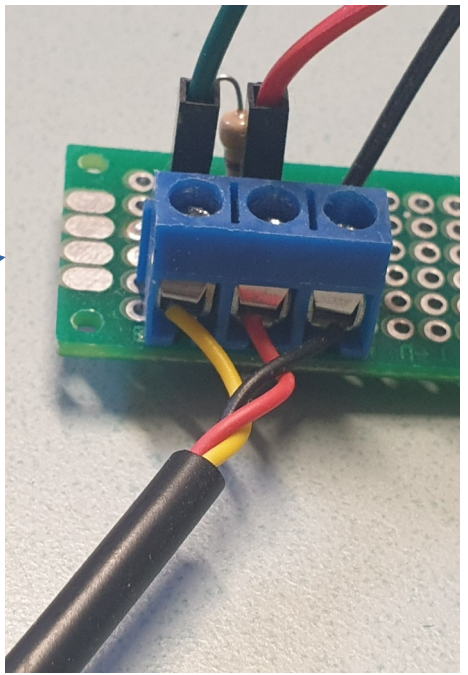


Schema di collegamento dei moduli



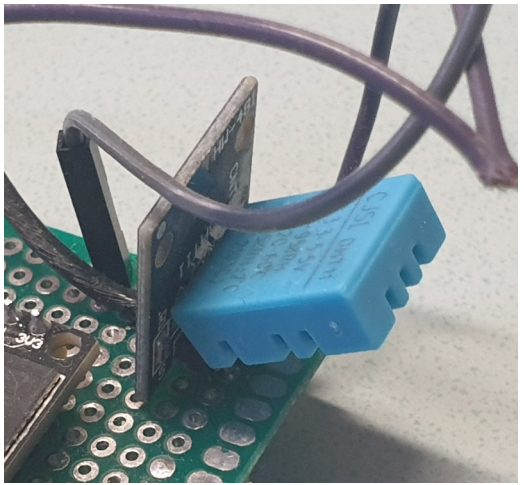
Moduli/1





Moduli/2

DS18B20



DH11

Modulo di ricezione dei dati/1

- Access point WiFi
  - Servizio hostapd => Accetta le connessioni in entrata
  - Servizio dnsmasq => distribuisce gli indirizzi IP in modo dinamico con DHCP
  - Per collegarci usiamo ssh e vnc

Modulo di ricezione dei dati/2

- Servizi di ricezione dati dai moduli
  - Tentativo di installazione InfluxDB
    (su architettura arm solo versione < 2.0)
  - Installazione broker MQTT (mosquitto)
  - Creazione del subscriber che raccoglie i dati (python)
    - Libreria python **paho-mqtt**
  - Analisi dei dati
    - Utilizzo di Pandas e Matplotlib

Modulo di trasmissione dei dati/1

- Usiamo delle esp32
  - Tentativo di installazione InfluxDB
    (solo versione < 2.0 su architettura arm)
  - Installazione broker MQTT (mosquitto)
  - Creazione dei subscriber (python)
    - Libreria python **paho-mqtt**
  - Analisi dei dati
    - Utilizzo di Pandas e Matplotlib

Modulo di trasmissione dei dati/2

```
#define DHTTYPE DHT11   // DHT 11
#define dht_dpin 5
DHT dht(dht_dpin, DHTTYPE);

#include <ArduinoMqttClient.h>
#include "arduino_secrets.h"

// costanti
char ssid[]  = SECRET_SSID;   // your network SSID (name)
char pass[]  = SECRET_PASS;   // your network password

float h_stanza = 0.0;
float t_stanza = 0.0;

float t_termosifone = 0.0;

DS18B20 ds(14);

char clientID[]  = "iot_paride_1";

WiFiClient wifiClient;
MqttClient mqttClient(wifiClient);

const char broker[]      = "192.168.0.10";
IPAddress broker_ip(192, 168, 0, 10);
int        port          = 1883;
const char topic[]       = "iot/message";
const char topic_sync[]  = "iot/sync";
const char listenTopic[] = "iot/led";

int count = 0;
int count_wifi_connections = 0;
int count_mqtt_connections = 0;

```

Subscriber MQTT/1

```
7  import paho.mqtt.client as mqttclient
8  import time
9  import datetime
10
11 import mysql.connector
12
13 def GetDataArray(line):
14   array_singolo_elemento = []
15   elements = line.split(";")
16   #print(len(elements))
17   if len (elements) >= 8:
18     array_singolo_elemento = []
19     for elemento in elements:
20       dati = elemento.split("=")
21       array_singolo_elemento.append(dati[-1])
22   return array_singolo_elemento
23
24 def writeOnDatabase(timestamp, line):
25   mydb = mysql.connector.connect(
26     host="localhost",
27     user="paride",
28     password="12345",
29     database="iot"
30   )
31   dati = GetDataArray(line)
32   mycursor = mydb.cursor()
33
34   sql = "INSERT INTO sensori (time, sender_ip, clientID, h_stanza, t_stanza, t_termosifone, count, count_wifi_c, count_mqtt_c) VALUES
     (%s, %s, %s, %s, %s, %s, %s, %s, %s, %s)"
35   val = (str(timestamp)[:19].replace("-","").replace(".","").replace(":","").replace(" ",""), dati[0], dati[1], dati[2], dati[3], dati[4
     ], dati[5], dati[6], dati[7])
36
37   mycursor.execute(sql, val)
38
39   mydb.commit()
40
41
42 def on_connect(client, usedata, flags, rc):
```

Subscriber MQTT/2

```
42 def on_connect(client, usedata, flags, rc):
43   if (rc == 0):
44     print("client is connected")
45     global connected
46     connected = True
47   else:
48     print("connection failed")
49
50 def on_message(client, userdata, message):
51   # global Messagereceived
52   Messagereceived = True
53   #print ("Message received %s" % (str(message.payload.decode("utf-8"))))
54   #print ("Message topic=%s" %  (message.topic))
55   print ("%s;%s" % (datetime.datetime.now(), str(message.payload.decode("utf-8"))))
56   f = open("registrazioni.txt", "a")
57   f.write("%s;%s\n" % (datetime.datetime.now(), str(message.payload.decode("utf-8"))))
58   f.close()
59
60   # scrivo anche nel DB
61   writeOnDatabase(datetime.datetime.now(), str(message.payload.decode("utf-8")))
62
63 connected=False
64 Messagereceived= False
65 broker_address="192.168.0.10"
66 port=1883
67 user="mqtt_user"
68 password="123456"
69
70 client = mqttclient.Client("MQTT")
71 #client.username_pw_set(user, password=password)
72 client.on_connect = on_connect
73 client.on_message = on_message
74 client.connect(broker_address, port=port)
75 client.loop_start()
76 #client.subscribe("mqtt/firstcode")
77 client.subscribe("iot/message")
78 while connected != True:
79   time.sleep(0.2)
80 while Messagereceived != True:
81   time.sleep(0.2)
82 client.loop_stop()
83
```

File di testo con i dati ricevuti

```
386640 2022-05-23 07:19:14.964873;sender_ip=192.168.0.16;clientID=iot_paride_2;h_stanza=25.00;t_stanza=24.50;t_termosifone=24.12;count=40686;count_wifi_c=19;count_mqtt_c=85
386641 2022-05-23 07:19:30.126143;sender_ip=192.168.0.23;clientID=iot_paride_3;h_stanza=23.00;t_stanza=24.70;t_termosifone=23.94;count=2292;count_wifi_c=1;count_mqtt_c=3
386642 2022-05-23 07:19:45.918845;sender_ip=192.168.0.14;clientID=iot_paride_1;h_stanza=21.00;t_stanza=25.50;t_termosifone=23.62;count=125188;count_wifi_c=5;count_mqtt_c=46
386643 2022-05-23 07:19:47.655787;sender_ip=192.168.0.16;clientID=iot_paride_2;h_stanza=25.00;t_stanza=24.50;t_termosifone=24.12;count=40687;count_wifi_c=19;count_mqtt_c=85
386644 2022-05-23 07:20:02.819623;sender_ip=192.168.0.23;clientID=iot_paride_3;h_stanza=23.00;t_stanza=24.70;t_termosifone=23.94;count=2293;count_wifi_c=1;count_mqtt_c=3
386645 2022-05-23 07:20:18.642556;sender_ip=192.168.0.14;clientID=iot_paride_1;h_stanza=21.00;t_stanza=25.60;t_termosifone=23.62;count=125189;count_wifi_c=5;count_mqtt_c=46
386646 2022-05-23 07:20:20.397896;sender_ip=192.168.0.16;clientID=iot_paride_2;h_stanza=25.00;t_stanza=24.50;t_termosifone=24.12;count=40688;count_wifi_c=19;count_mqtt_c=85
386647 2022-05-23 07:20:35.514759;sender_ip=192.168.0.23;clientID=iot_paride_3;h_stanza=23.00;t_stanza=24.70;t_termosifone=23.94;count=2294;count_wifi_c=1;count_mqtt_c=3
386648 2022-05-23 07:20:51.374530;sender_ip=192.168.0.14;clientID=iot_paride_1;h_stanza=21.00;t_stanza=25.60;t_termosifone=23.62;count=125190;count_wifi_c=5;count_mqtt_c=46
386649 2022-05-23 07:20:53.142305;sender_ip=192.168.0.16;clientID=iot_paride_2;h_stanza=25.00;t_stanza=24.50;t_termosifone=24.12;count=40689;count_wifi_c=19;count_mqtt_c=85
386650 2022-05-23 07:21:08.215416;sender_ip=192.168.0.23;clientID=iot_paride_3;h_stanza=23.00;t_stanza=24.70;t_termosifone=23.94;count=2295;count_wifi_c=1;count_mqtt_c=3
386651 2022-05-23 07:21:24.093693;sender_ip=192.168.0.14;clientID=iot_paride_1;h_stanza=21.00;t_stanza=25.60;t_termosifone=23.62;count=125191;count_wifi_c=5;count_mqtt_c=46
386652 2022-05-23 07:21:25.887609;sender_ip=192.168.0.16;clientID=iot_paride_2;h_stanza=25.00;t_stanza=24.50;t_termosifone=24.12;count=40690;count_wifi_c=19;count_mqtt_c=85
386653 2022-05-23 07:21:40.908893;sender_ip=192.168.0.23;clientID=iot_paride_3;h_stanza=23.00;t_stanza=24.70;t_termosifone=23.94;count=2296;count_wifi_c=1;count_mqtt_c=3
386654 2022-05-23 07:25:13.288107;sender_ip=192.168.0.14;clientID=iot_paride_1;h_stanza=21.00;t_stanza=25.60;t_termosifone=23.62;count=125198;count_wifi_c=5;count_mqtt_c=46
386655 2022-05-23 07:25:15.170611;sender_ip=192.168.0.16;clientID=iot_paride_2;h_stanza=25.00;t_stanza=24.50;t_termosifone=24.12;count=40697;count_wifi_c=19;count_mqtt_c=85
386656 2022-05-23 07:25:29.781144;sender_ip=192.168.0.23;clientID=iot_paride_3;h_stanza=23.00;t_stanza=24.70;t_termosifone=23.94;count=2303;count_wifi_c=1;count_mqtt_c=3
386657 2022-05-23 07:25:45.896452;sender_ip=192.168.0.14;clientID=iot_paride_1;h_stanza=21.00;t_stanza=25.60;t_termosifone=23.62;count=125199;count_wifi_c=5;count_mqtt_c=46
386658 2022-05-23 07:25:47.831117;sender_ip=192.168.0.16;clientID=iot_paride_2;h_stanza=25.00;t_stanza=24.50;t_termosifone=24.12;count=40698;count_wifi_c=19;count_mqtt_c=85
386659 2022-05-23 07:26:02.477296;sender_ip=192.168.0.23;clientID=iot_paride_3;h_stanza=23.00;t_stanza=24.70;t_termosifone=23.94;count=2304;count_wifi_c=1;count_mqtt_c=3
386660 2022-05-23 07:26:18.619555;sender_ip=192.168.0.14;clientID=iot_paride_1;h_stanza=21.00;t_stanza=25.60;t_termosifone=23.62;count=125200;count_wifi_c=5;count_mqtt_c=46
386661 2022-05-23 07:26:20.573672;sender_ip=192.168.0.16;clientID=iot_paride_2;h_stanza=25.00;t_stanza=24.50;t_termosifone=24.12;count=40699;count_wifi_c=19;count_mqtt_c=85
386662 2022-05-23 07:26:35.173583;sender_ip=192.168.0.23;clientID=iot_paride_3;h_stanza=23.00;t_stanza=24.70;t_termosifone=23.94;count=2305;count_wifi_c=1;count_mqtt_c=3
386663 2022-05-23 07:26:53.319268;sender_ip=192.168.0.16;clientID=iot_paride_2;h_stanza=25.00;t_stanza=24.50;t_termosifone=24.12;count=40700;count_wifi_c=19;count_mqtt_c=85
386664 2022-05-23 07:28:13.285812;sender_ip=192.168.0.23;clientID=iot_paride_3;h_stanza=24.00;t_stanza=24.70;t_termosifone=23.94;count=2308;count_wifi_c=1;count_mqtt_c=3
386665 2022-05-23 07:34:29.509682;sender_ip=192.168.0.14;clientID=iot_paride_1;h_stanza=21.00;t_stanza=25.60;t_termosifone=23.69;count=125215;count_wifi_c=5;count_mqtt_c=46
386666 2022-05-23 07:35:04.470626;sender_ip=192.168.0.16;clientID=iot_paride_2;h_stanza=26.00;t_stanza=24.60;t_termosifone=24.19;count=40715;count_wifi_c=19;count_mqtt_c=85
386667 2022-05-23 07:35:51.007270;sender_ip=192.168.0.23;clientID=iot_paride_3;h_stanza=24.00;t_stanza=24.80;t_termosifone=23.94;count=2322;count_wifi_c=1;count_mqtt_c=3
386668 2022-05-23 07:36:40.595429;sender_ip=192.168.0.14;clientID=iot_paride_1;h_stanza=21.00;t_stanza=25.60;t_termosifone=23.69;count=125219;count_wifi_c=5;count_mqtt_c=46
386669 2022-05-23 07:37:45.856184;sender_ip=192.168.0.14;clientID=iot_paride_1;h_stanza=21.00;t_stanza=25.60;t_termosifone=23.62;count=125221;count_wifi_c=5;count_mqtt_c=46
386670 2022-05-23 07:38:34.492435;sender_ip=192.168.0.23;clientID=iot_paride_3;h_stanza=24.00;t_stanza=24.80;t_termosifone=23.94;count=2327;count_wifi_c=1;count_mqtt_c=3
386671 2022-05-23 07:39:56.751258;sender_ip=192.168.0.14;clientID=iot_paride_1;h_stanza=21.00;t_stanza=25.60;t_termosifone=23.69;count=125225;count_wifi_c=5;count_mqtt_c=46
386672 2022-05-23 07:41:34.928328;sender_ip=192.168.0.14;clientID=iot_paride_1;h_stanza=21.00;t_stanza=25.60;t_termosifone=23.69;count=125228;count_wifi_c=5;count_mqtt_c=46
386673 2022-05-23 07:47:34.907356;sender_ip=192.168.0.14;clientID=iot_paride_1;h_stanza=21.00;t_stanza=25.70;t_termosifone=23.75;count=125239;count_wifi_c=5;count_mqtt_c=46
386674 2022-05-23 07:48:07.636403;sender_ip=192.168.0.14;clientID=iot_paride_1;h_stanza=21.00;t_stanza=25.70;t_termosifone=23.75;count=125240;count_wifi_c=5;count_mqtt_c=46
386675 2022-05-23 07:49:45.807829;sender_ip=192.168.0.14;clientID=iot_paride_1;h_stanza=21.00;t_stanza=25.80;t_termosifone=23.75;count=125243;count_wifi_c=5;count_mqtt_c=46
386676 2022-05-23 07:49:48.539745;sender_ip=192.168.0.16;clientID=iot_paride_2;h_stanza=26.00;t_stanza=24.70;t_termosifone=24.31;count=40742;count_wifi_c=19;count_mqtt_c=85
386677 2022-05-23 07:50:01.104035;sender_ip=192.168.0.23;clientID=iot_paride_3;h_stanza=25.00;t_stanza=24.80;t_termosifone=24.06;count=2348;count_wifi_c=1;count_mqtt_c=3
386678 2022-05-23 07:50:18.532851;sender_ip=192.168.0.14;clientID=iot_paride_1;h_stanza=21.00;t_stanza=25.90;t_termosifone=23.75;count=125244;count_wifi_c=5;count_mqtt_c=46
386679 2022-05-23 07:50:21.327006;sender_ip=192.168.0.16;clientID=iot_paride_2;h_stanza=26.00;t_stanza=24.70;t_termosifone=24.31;count=40743;count_wifi_c=19;count_mqtt_c=85
386680 2022-05-23 07:50:33.801126;sender_ip=192.168.0.23;clientID=iot_paride_3;h_stanza=25.00;t_stanza=24.90;t_termosifone=24.06;count=2349;count_wifi_c=1;count_mqtt_c=3
386681 2022-05-23 07:50:51.257423;sender_ip=192.168.0.14;clientID=iot_paride_1;h_stanza=21.00;t_stanza=25.80;t_termosifone=23.75;count=125245;count_wifi_c=5;count_mqtt_c=46
386682 2022-05-23 07:50:54.031964;sender_ip=192.168.0.16;clientID=iot_paride_2;h_stanza=26.00;t_stanza=24.70;t_termosifone=24.37;count=40744;count_wifi_c=19;count_mqtt_c=85
386683 2022-05-23 07:51:06.496624;sender_ip=192.168.0.23;clientID=iot_paride_3;h_stanza=25.00;t_stanza=24.90;t_termosifone=24.06;count=2350;count_wifi_c=1;count_mqtt_c=3
386684
```

Convertitore del file per Pandas

```
sep = "\t"
# sep = ","

with open('registrazioni.txt') as f:
  lines = f.readlines()
print (len(lines))

array_elements= []
intestazione_array = []
for line in lines:
  elements = line.split(";")
  if len (elements) >= 9:
    if elements[0][0:4].isnumeric(): # salto le righe con eventuali dati spuri
      if len(array_elements) == 0:
        intestazione_array.append("time")
      array_singolo_elemento = []
      for elemento in elements:
        dati = elemento.split("=")
        if len(array_elements) == 0:
          if len(dati) > 1:
            if dati[0] == "count":
              intestazione_array.append("count_sent")
            else:
              intestazione_array.append(dati[0])
        if len(dati) == 1:
          array_singolo_elemento.append(dati[-1][:19].replace("-","").replace(".","").replace(":","").replace(" ",""))
        else:
          array_singolo_elemento.append(dati[-1])
      #
      if not "nan" in array_singolo_elemento:
        array_elements.append(sep.join(array_singolo_elemento))



f1 = open("registrazioni_new.txt", "w")
f1.write(sep.join(intestazione_array))
f1.write("\n")
for riga in array_elements:
  f1.write(riga)
f1.close()
```

File dei dati pronto per essere usato da Pandas

Tab

```
time→sender_ip→clientID→h_stanza→t_stanza→t_termosifone→count_sent→count_wifi_c→count_mqtt_cCRLF
20220404212124→192.168.0.23→iot_paride_3→30.00→21.90→20.25→0→1→1CRLF
20220404212204→192.168.0.23→iot_paride_3→34.00→22.40→20.25→0→1→1CRLF
20220404212237→192.168.0.23→iot_paride_3→32.00→22.50→20.25→1→1→1CRLF
20220404212310→192.168.0.23→iot_paride_3→30.00→22.50→20.25→2→1→1CRLF
20220404212342→192.168.0.23→iot_paride_3→29.00→22.40→20.31→3→1→1CRLF
20220404212415→192.168.0.23→iot_paride_3→29.00→22.20→20.31→4→1→1CRLF
20220404212448→192.168.0.23→iot_paride_3→28.00→22.00→20.31→5→1→1CRLF
20220404212521→192.168.0.23→iot_paride_3→28.00→21.80→20.37→6→1→1CRLF
20220404212553→192.168.0.23→iot_paride_3→28.00→21.70→20.37→7→1→1CRLF
20220404212626→192.168.0.23→iot_paride_3→28.00→21.50→20.44→8→1→1CRLF
20220404212659→192.168.0.23→iot_paride_3→29.00→21.40→20.44→9→1→1CRLF
20220404212731→192.168.0.23→iot_paride_3→29.00→21.40→20.50→10→1→1CRLF
20220404212804→192.168.0.23→iot_paride_3→30.00→21.30→20.56→11→1→1CRLF
20220404212837→192.168.0.23→iot_paride_3→30.00→21.30→20.62→12→1→1CRLF
20220404212909→192.168.0.23→iot_paride_3→30.00→21.20→20.69→13→1→1CRLF
20220404212942→192.168.0.23→iot_paride_3→30.00→21.20→20.75→14→1→1CRLF
20220404213015→192.168.0.23→iot_paride_3→30.00→21.20→20.75→15→1→1CRLF
20220404213048→192.168.0.23→iot_paride_3→30.00→21.20→20.81→16→1→1CRLF
20220404213120→192.168.0.23→iot_paride_3→30.00→21.20→20.87→17→1→1CRLF
20220404213153→192.168.0.23→iot_paride_3→30.00→21.20→20.94→18→1→1CRLF
20220404213226→192.168.0.23→iot_paride_3→30.00→21.20→20.94→19→1→1CRLF
20220404213258→192.168.0.23→iot_paride_3→30.00→21.30→21.00→20→1→1CRLF
20220404213331→192.168.0.23→iot_paride_3→30.00→21.30→21.06→21→1→1CRLF
20220404213404→192.168.0.23→iot_paride_3→30.00→21.30→21.12→22→1→1CRLF
20220404213436→192.168.0.23→iot_paride_3→30.00→21.30→21.12→23→1→1CRLF
20220404213509→192.168.0.23→iot_paride_3→30.00→21.40→21.19→24→1→1CRLF
20220404213542→192.168.0.23→iot_paride_3→30.00→21.40→21.25→25→1→1CRLF
20220404213614→192.168.0.23→iot_paride_3→30.00→21.40→21.25→26→1→1CRLF
20220404213647→192.168.0.23→iot_paride_3→30.00→21.40→21.31→27→1→1CRLF
```

Esempio di analisi dei dati/1

Tabulazione per separare i campi

Aggiungo i campi che mi servono

Creo una figura con tre grafici

Calcolo il numero di dati ricevuti ogni giorno

Calcolo media, massima e minima delle temperature giornaliere per ogni sensore dht11

```
1  import pandas as pd
2  import matplotlib.pyplot as plt
3
4  df = pd.read_csv('registrazioni_new.txt', sep="\t", header=0)
5
6  def converti_data_ora(x):
7    ora = ("%06d" % (x.time % 1000000))
8    return (x.time % 1000000)
9
10 #aggiungo un campo nel quale è presente solo l'orario
11 df['time2']  = df.apply(lambda x: (x.time % 1000000) , axis = 1)
12
13 #aggiungo un campo nel quale è presente solo la data
14 df['date']  = df.apply(lambda x: str(x.time / 1000000)[:8] , axis = 1)
15
16 # plt.style.use('ggplot')
17 fig, axs = plt.subplots(nrows = 3, ncols = 1)
18 fig.set_facecolor('lightsteelblue')
19 fig.tight_layout()
20
21 # print(df.head())
22 conteggio_dati = df.groupby(["date"])["date"].count()
23 #ragguppo i valori per data e faccio la media per ogni giorno
24 # medie_1 = df[(df.clientID=='iot_paride_1')].groupby(["date"])["t_stanza"].mean()
25
26 # date = df[(df.clientID=='iot_paride_1')].groupby(["date"])
27
28 # print(date["date"])
29
30 media_temperature = df[(df.clientID=='iot_paride_2')].groupby(["date"])["t_stanza"].mean()
31 minima_temperatura = df[(df.clientID=='iot_paride_2')].groupby(["date"])["t_stanza"].min()
32 massima_temperatura = df[(df.clientID=='iot_paride_2')].groupby(["date"])["t_stanza"].max()
```

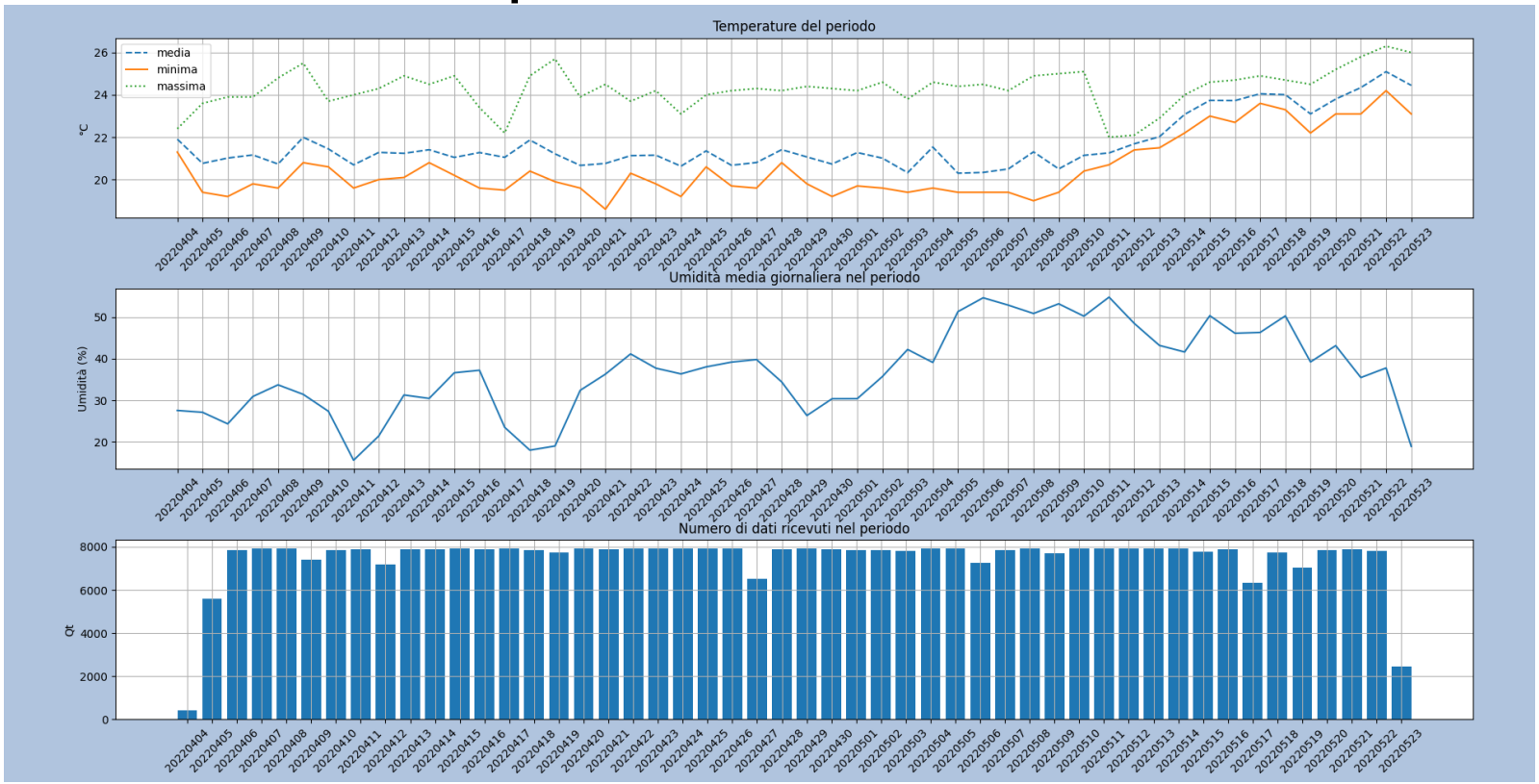
Esempio di analisi dei dati/2

Grafico delle temperature

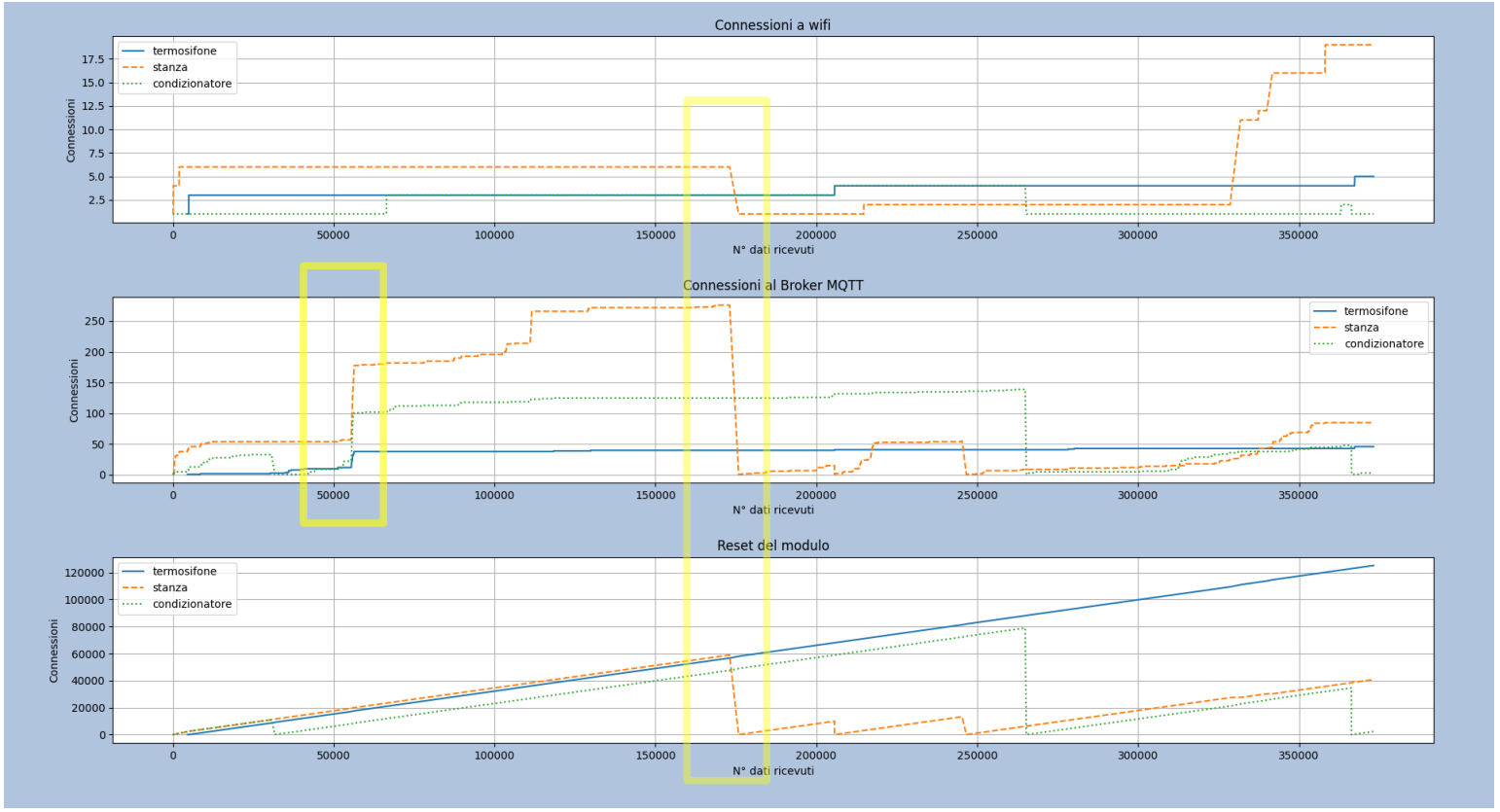
Grafico dell'umidità

```
33
34  # calcolo l'umidità media usando i dati del modulo numero 2
35  media_umidita = df[(df.clientID=='iot_paride_2')].groupby(["date"])["h_stanza"].mean()
36
37  solo_date = []
38  # creo un array con solo le date da usare come asse delle x
39  for d in df[(df.clientID=='iot_paride_2')].groupby(["date"]):
40    solo_date.append(d[0])
41
42  row = 0
43  # grafico con i dati delle temperature
44  axs[row].set_title("Temperature del periodo")
45  axs[row].plot(solo_date, media_temperature, label='media', linestyle = '--')
46  axs[row].plot(solo_date, minima_temperatura, label='minima', linestyle = '-')
47  axs[row].plot(solo_date, massima_temperatura, label='massima', linestyle = ':')
48  axs[row].set_ylabel('°C')
49
50  for label in axs[row].xaxis.get_ticklabels():
51    label.set_rotation(45)
52
53  axs[row].legend()
54  axs[row].grid(True)
55
56  solo_date = []
57  for d in df.groupby(["date"]):
58    solo_date.append(d[0])
59
60  row = 1
61  # grafico con l'umidità media giornaliera
62  axs[row].set_title("Umidità media giornaliera nel periodo")
63  axs[row].plot(solo_date, media_umidita, label='umidità (%)', linestyle = '-')
64  axs[row].set_ylabel('Umidità (%)')
65  axs[row].grid(True)
66  for label in axs[row].xaxis.get_ticklabels():
```

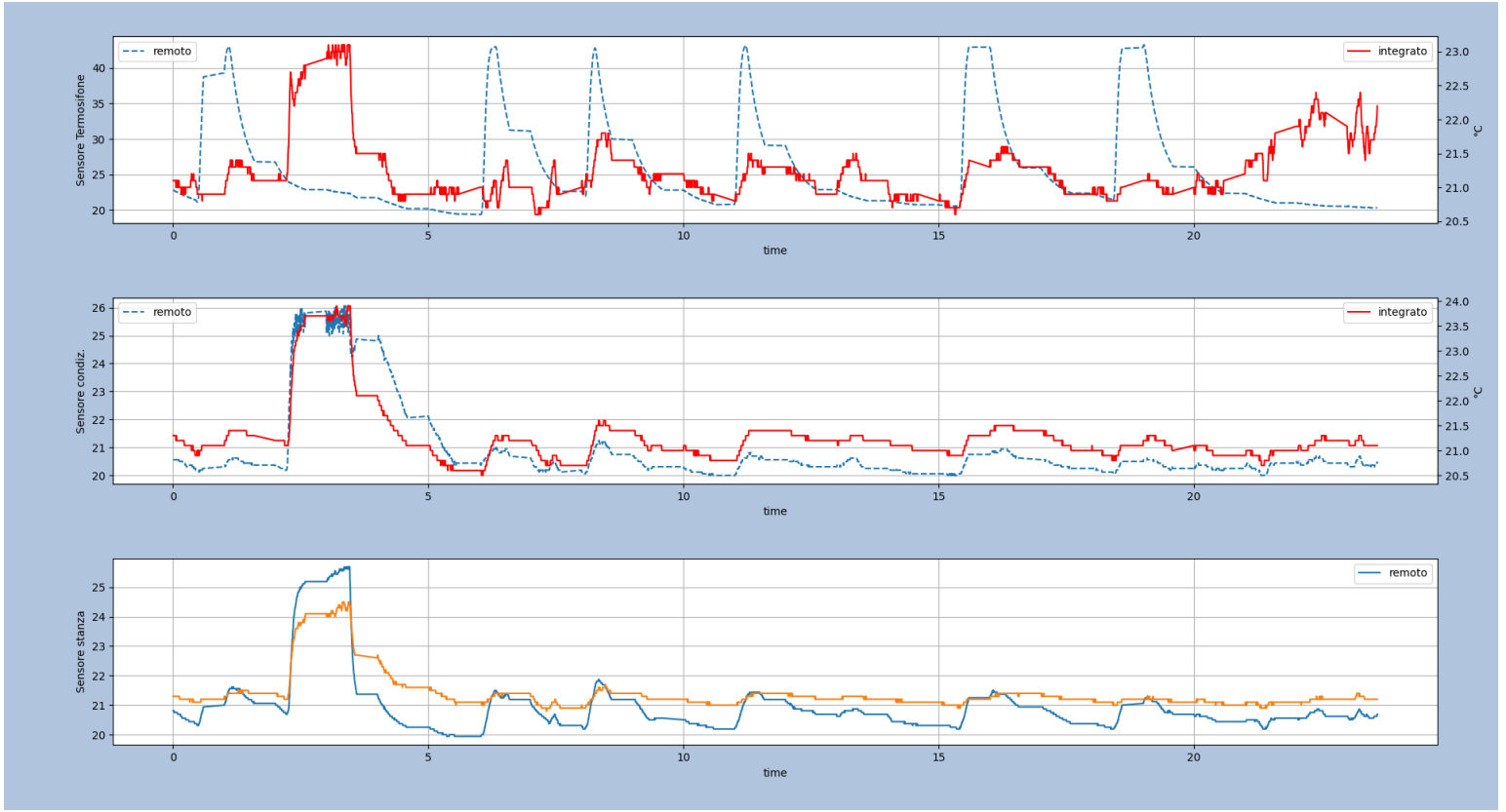
Esempio di analisi dei dati/3



Problemi con la connessione



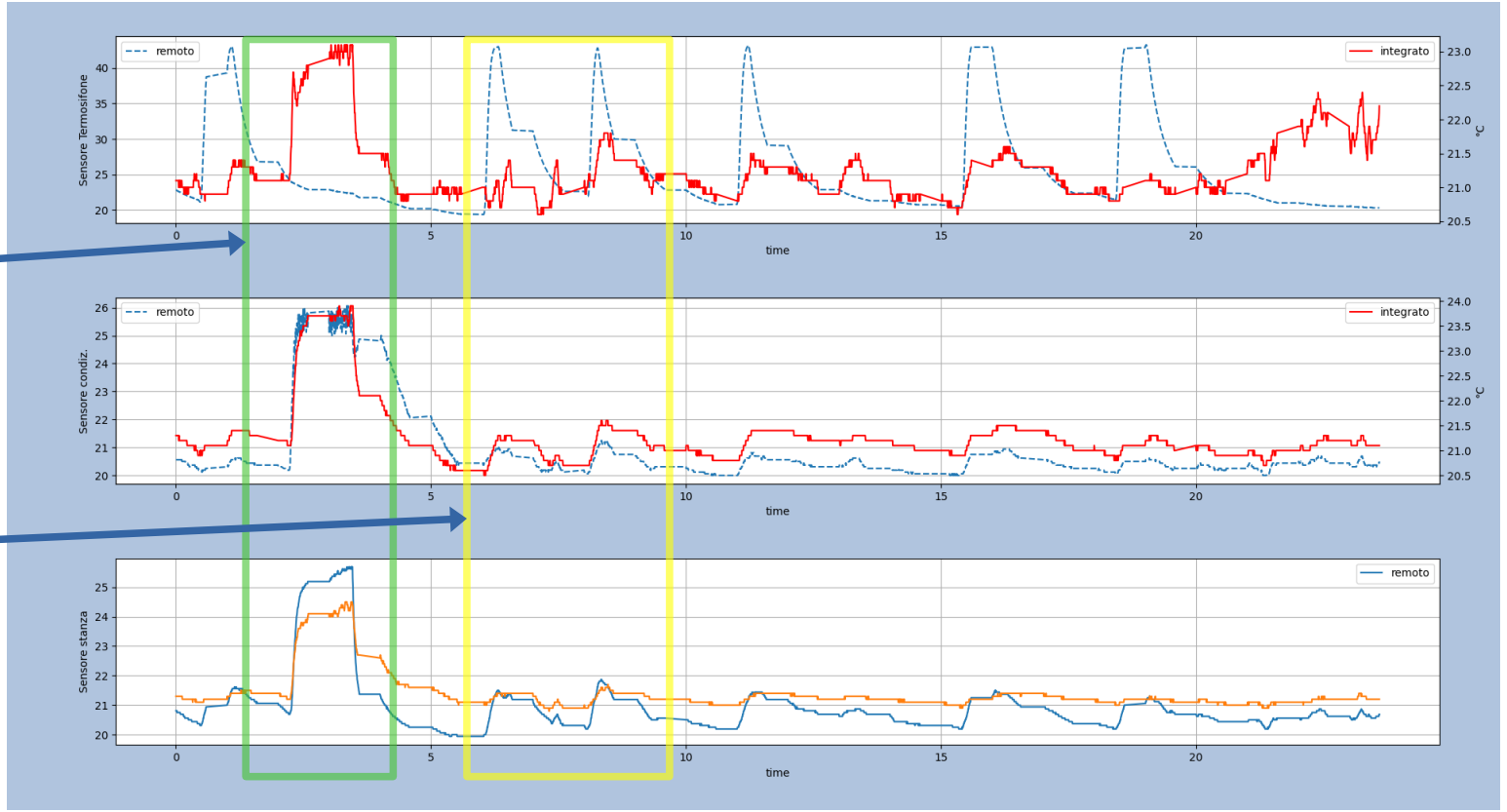
Confronto efficienza



Confronto efficienza

Accensione pompa di calore

Accensione termosifone



Conclusioni

Risorse

E’ possibile scaricare tutti i file del progetto dal repository

https://github.com/pdomi2001/IOT_project

I file presenti sono:
- **mqtt_subscriber.py**: Questo script resta in ascolto sul server MQTT e scrive le rilevazioni
- **converti_dati.py**: Converte i dati ricevuti in un formato che sia leggibile da Pandas
- **analisi_dati_connessioni.py**: Analizza i dati relativi alle connessioni effettuate dai moduli e crea una serie di grafici che indicano le riconnessioni nell'intero periodo.
- **analisi_dati_periodo_intero.py**: Analizza i dati relativi alle temperature medie giornaliere e l'umidità media giornaliera.
- **analisi_dati_termico.py**: Analizza i dati relativi ad un dato giorno (nel nostro caso il 14 aprile 2022) e mostra gli andamenti, durante la giornata delle temperature registrate dai sensori interni e dalle sonde dei moduli.

- **progetto_esp.ino**: Progetto per Arduino da caricare sulla ESP32
- **arduino_secrets.h**: File che contiene le informazioni per far connettere alla rete WIFI la ESP32

Più i file accessori dei grafici generati e degli schemi presenti in questo documento.