Acoustic pollution monitoring platform

Fabio Muratori - muratorifabio96@gmail.com

Project goal 1/3

Noise pollution is a pervasive pollutant that adversely affects on the health and well-being of people and wildlife

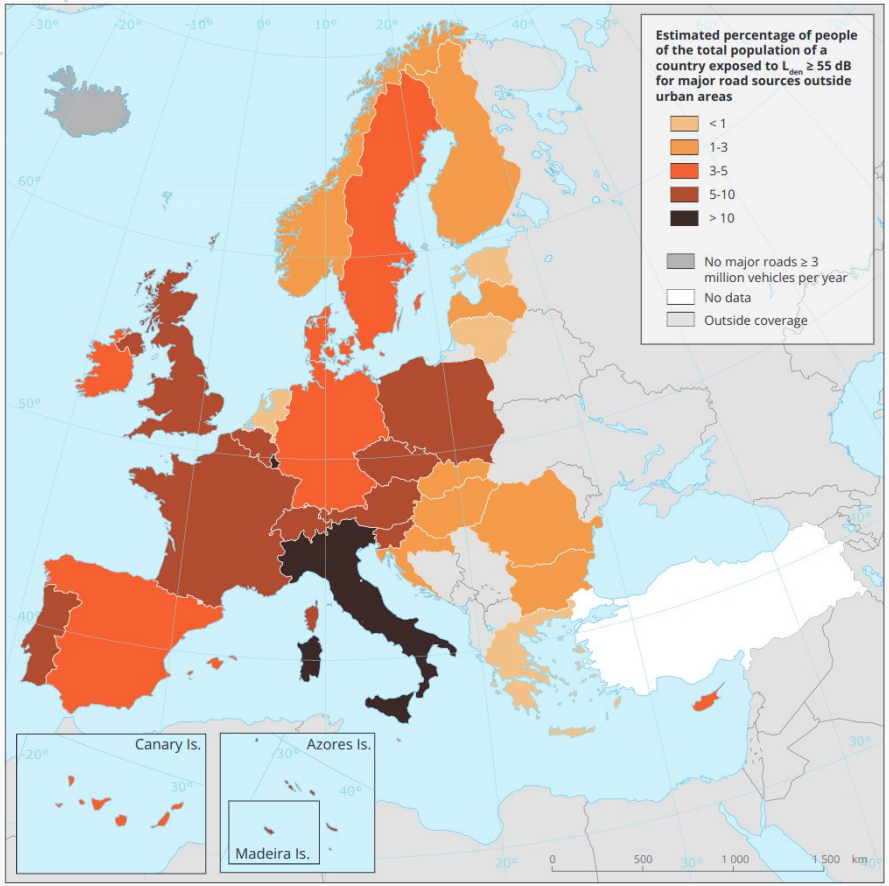
Main sources and causes:

- Human activity
- the most prevalent sources of environmental noise are those related to transportation
  - Air traffic
  - Railways and trains
  - Highways, streets and vehicles

Many health risks [2]:

- Change in auditory sensitivity (loss or hypersensitivity)
- Cardiovascular problems
- Type 2 diabetes
- Sleep problems
- Stress
- ...

Scopo del progetto 2/3



Estimated percentage of population exposed to a noise level >= 55db for major roads outside large population centers [1]

The WHO (World Health Organization) has stated that the maximum noise level should be 65 dB for daytime and 55 dB for nighttime, and that the optimum level should not exceed 45 dB.

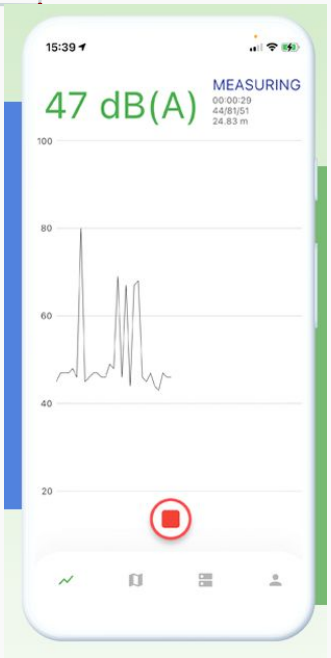
Scopo del progetto 3/3

The goal of this project is to realize a platform dedicated to noise level monitoring in cities through the use of Internet of Things solutions.

The project consists of:

- IoT devices capable of measuring noise levels
- Backend platform to manage measurements
- Prototype interfaces for remote noise level monitoring
- Off-the-shelf components for scalable and resilient services

State of the art

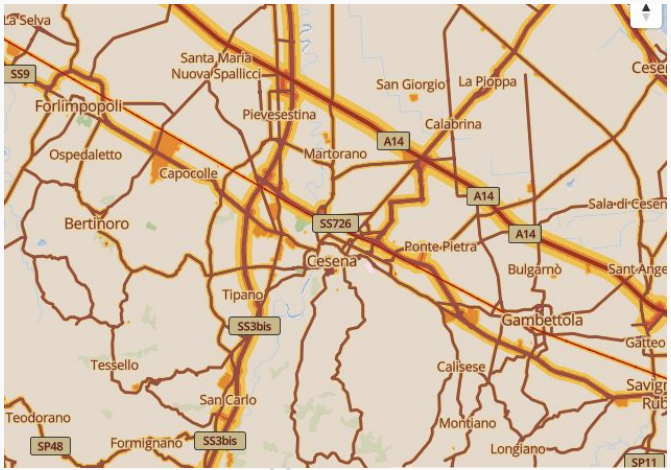


NoiseTube [6]

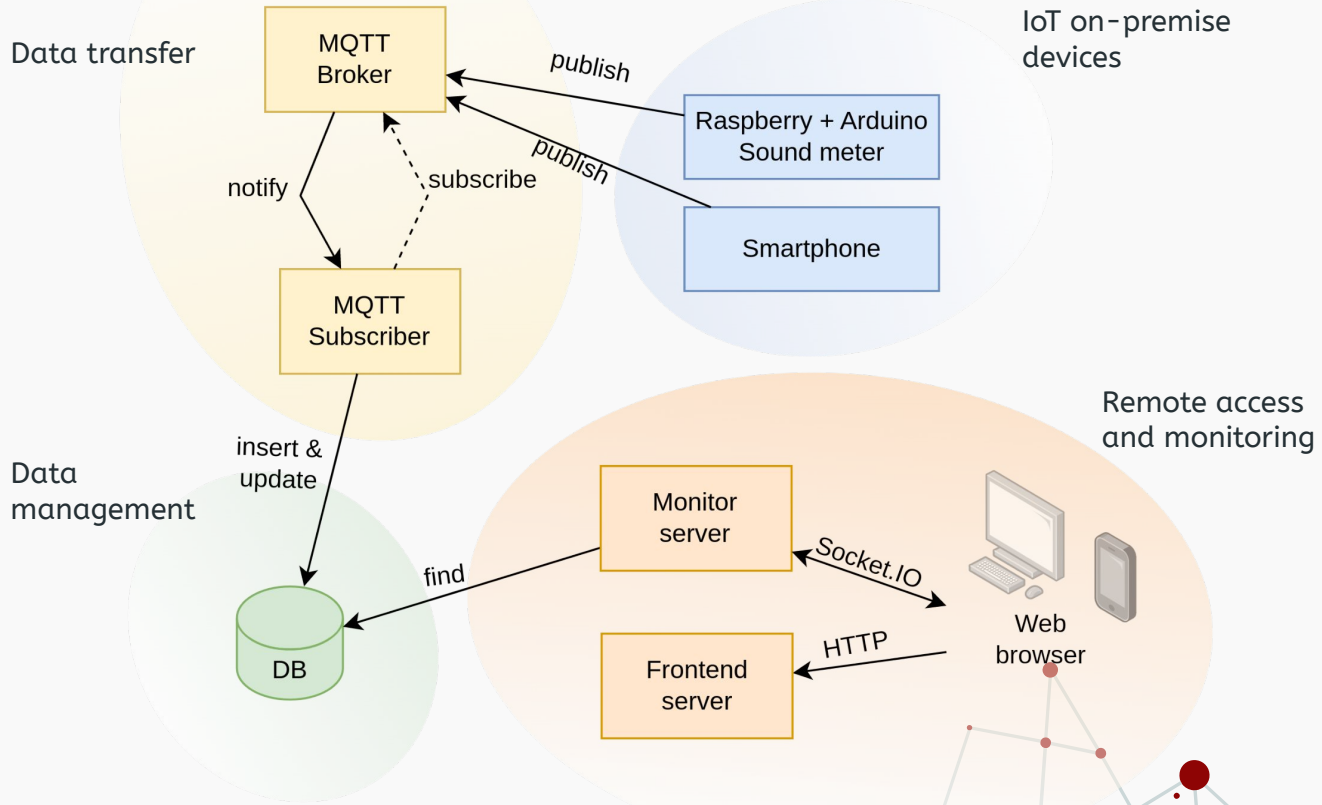
- Smartphone app
- Users provide noise data
- Collaborative approach

Noise Pollution Map [7]

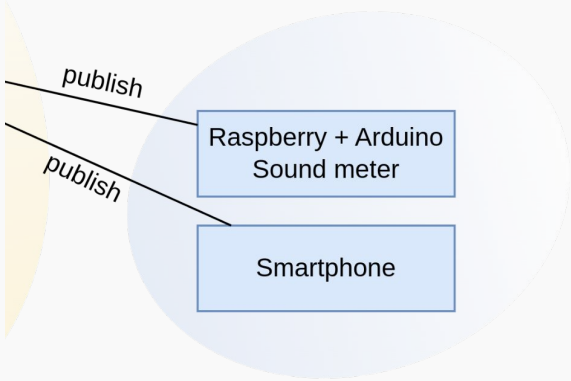
- Noise estimation based on geographical area and human activities (roads, businesses, buildings)
- No sensors are used



Platform architecture overview



Architecture - devices



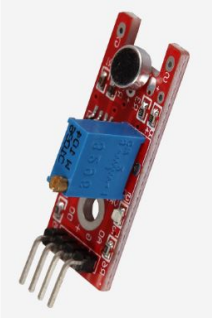
First solution for noise measurement with Raspberry PI platform and Arduino:

1. Analog sensor KY-038 [5] performs the measurement;
2. Arduino manages the sensor and sends the data to Raspberry PI via serial channel;
3. RaspberryPI receives the data at Arduino and communicates it to an MQTT platform

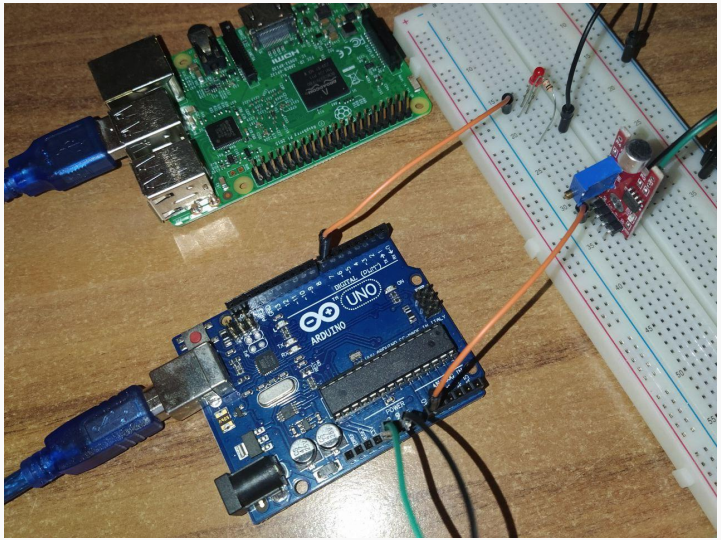
Second solution with Android smartphone:

- The device's microphone can be used to measure the noise level;
- The Foreground Service framework allows long-duration and background tasks to run;
- The application constantly retrieves data from the microphone and communicates it to an MQTT platform;

Architecture - devices example



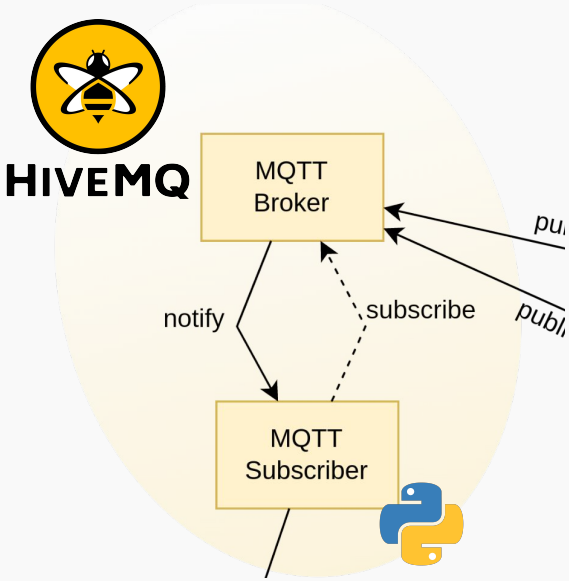
Sound level
Meter KY-038



Dispositivo assemblato. Nell'immagine:

- Raspberry PI *above*
- Arduino *below*
- Meters *on the right*

Architecture - data transfer



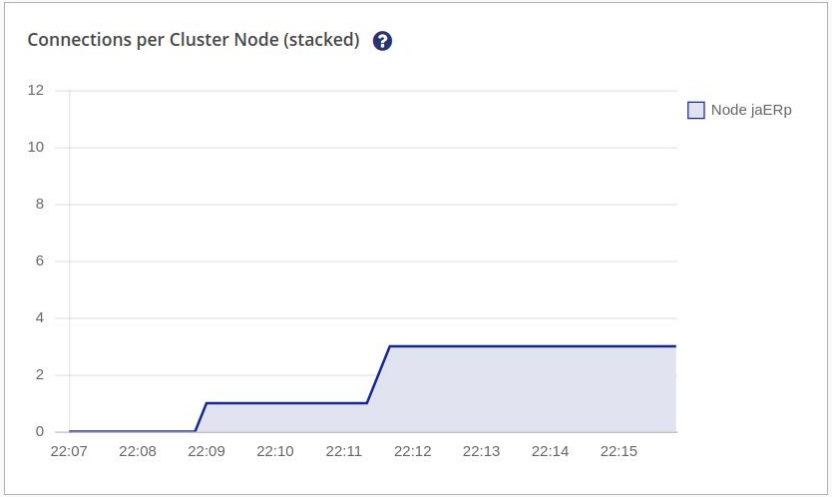
Through an MQTT platform, data is transferred from a source to the database

1. Client Subscribers register to wait for data
2. Client Publisher publishes the data
3. Server Broker receives data from Publishers and sends it to all Subscribers who have registered

HiveMQ[3] as a highly scalable MQTT platform (Broker Cluster, Load balancing).

- The framework provides a Server Broker
- The measurement devices are the Client Publishers
- A Python script is the Client Subscriber that receives data from the Broker
    - And inserts it into the database

Architecture - data transfer example



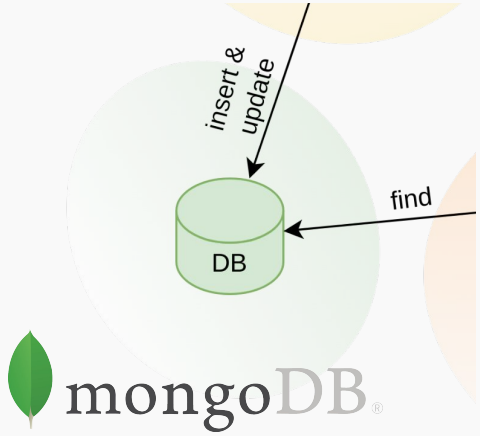
Part of the HiveMQ dashboard showing the number of active connections

android_meter_001	192.168.0.123
arduino_meter_002	192.168.0.106
sound_pollution_subscriber	127.0.0.1

MQTT clients connected to HiveMQ.

3 connected devices are shown: a Subscriber process and 2 Publisher processes (Arduino and Android)

Architecture - data management



Used MongoDB[4]

- Document-based No-SQL database
- Supports horizontal scalability
  - data replication on multiple nodes
  - geographic distribution
  - data sharding
- Each device that sends data is automatically registered as a document.

A device document contains:

- Device name
- Geographic coordinates
- Date of activation
- Last date of receiving measurements
- List of measurements

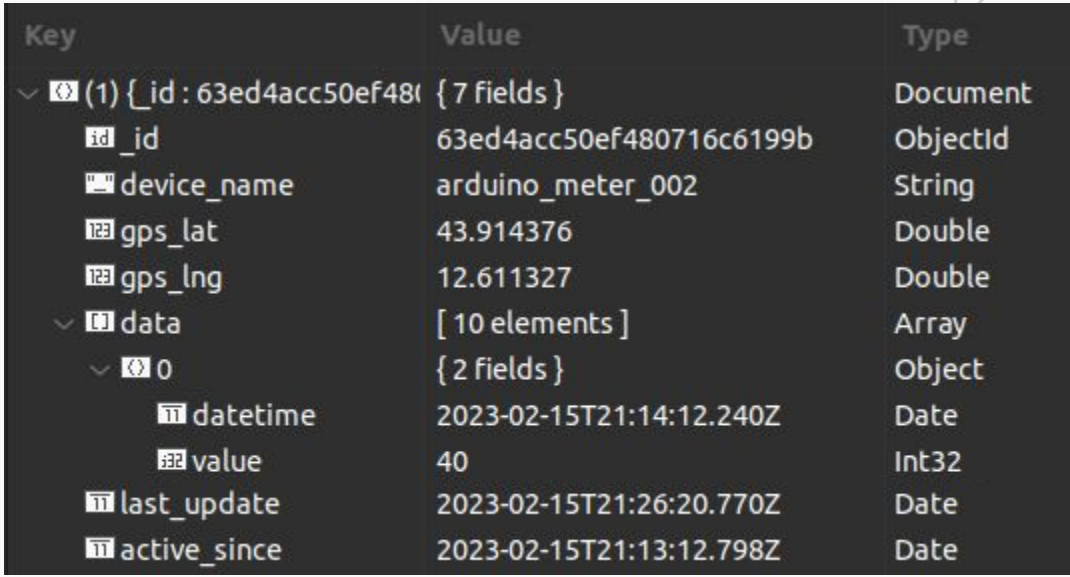
Architecture - data management example

Queries performed by the MQTT Subscriber are:

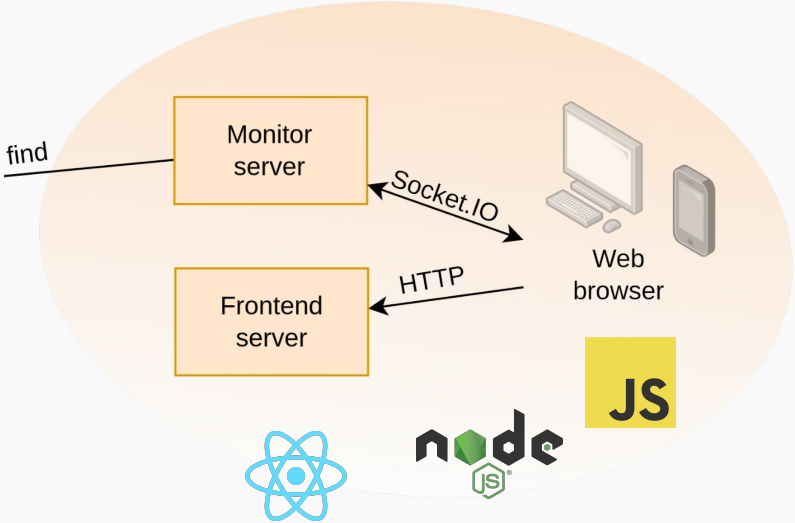
1. Creating a document for a new device
2. Adding measurements for an existing device

Queries performed by the Server Monitor are:

1. Retrieving the list of devices existing
2. Select data for a device and the latest measurements



Architecture - remote access



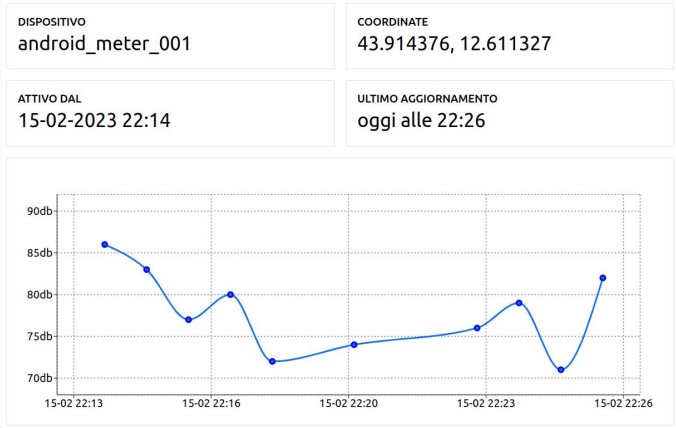
Monitor server handles all monitoring requests that come in from users

- A device is monitored only once
- Efficiency in relation to connected users

Frontend server provides a web app to users
Operating system independence

- A two-way channel between browser and monitor server
- Client requests monitoring
- Server communicates updates

Architecture - remote access example



Bibliographical references

[1] European Environment Agency's 2020 report on "Environmental Ruomore in Europe."
https://www.eea.europa.eu/publications/environmental-noise-in-europe

[2] Health effects of noise
https://hms.harvard.edu/magazine/viral-world/effects-noise-health

[3] HiveMQ communications platform MQTT
https://www.hivemq.com/

[4] MongoDB database no-SQL
https://www.mongodb.com/

[5] Noise Sensor KY-038
https://datasheetspdf.com/datasheet/KY-038.html

[6] NoiseTube android app
https://www.noisetube.net/

[7] Global noise pollution map
http://lukasmartinelli.ch/gis/2016/04/03/openstreetmap-noise-pollution-map.html