Edge#DA

Data Analytics course project **2020/2021**

Luca Cervioni - Dmitry Mingazov

Aim of the project

The aim of the project was to understand how an open-source messaging framework like ***ZeroMQ*** could be used to design a pub-sub communication system, with the purpose of moving data computation from the ***cloud*** to the ***edge***, the closest level to where the data is generated.

This has been done to reduce possible latencies in communication that could make the system less reactive.

Programming language and libraries used

All the code has been written in *C* due to the low level scope and the low computational capability of the devices.

Main libraries used within the project:

- *ZeroMQ*

  (to allow communication between nodes)

- *SQlite3*

  (to store data at the end of the workflow)

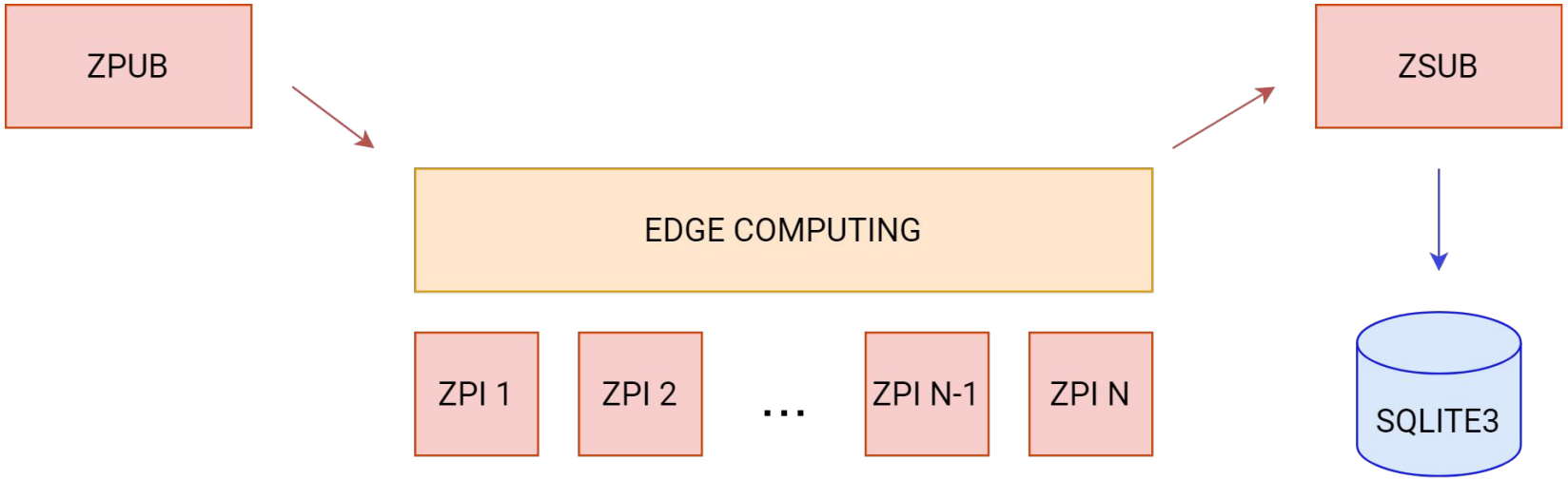
ZeroMQ

It is an open-source high-performance asynchronous **messaging framework**, aimed at use in distributed or concurrent applications, that gives the possibility to make communication between devices **fast and simple**.

It mostly works on sockets that carry atomic messages and it is able to create sockets N-to-N with patterns like fan-out, pub-sub, task distribution and request-reply.

source: https://zeromq.org/

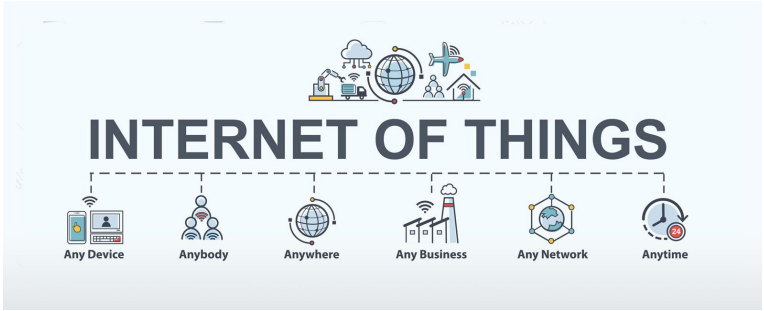
System architecture



System architecture: ZPUB

Simulates an endless stream of data (like several **IoT** devices), generating random *JSON* messages.

These messages will be then sent to the *Edge Computing* layer.



System data format

```
{
   "t":595211951,
   "time":"Sun Sep 12 00:20:17 2021",
   "ref":"jzp://edv#211b.0000",
   "uuid":"d682d80d-22d2-4a1c-9908-9a63a4869de2",
   "type":"Environmental",
   "m":[
      {
         "k":"environmental_temperature",
         "t":595211951,
         "tz":"Sun Sep 12 00:20:17 2021",
         "v":6.867828,
         "u":"°C",
         "ref":"jpz://edv#211b.0000/environmental_temperature"
      },
      {
         "k":"relative_humidity",
         "t":595211951,
         "tz":"Sun Sep 12 00:20:17 2021",
         "v":12,
         "u":"%",
         "ref":"jpz://edv#211b.0000/relative_humidity"
      },
      {
         "k":"battery_level",
         "t":595211951,
         "tz":"Sun Sep 12 00:20:17 2021",
         "v":19,
         "u":"%",
         "ref":"jpz://edv#211b.0000/battery_level"
      }
   ]
}
```

t: timestamp in seconds.

time/tz: formatted timestamp.

ref: physical device identifier.

k: type of measure.

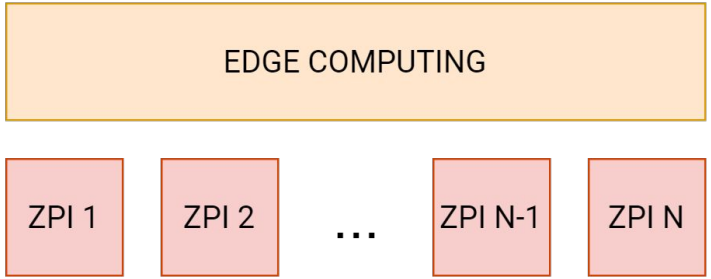
v: value of measure.

u: unit of measure.

System architecture: Edge Computing

The *Edge Computing* layer is a cluster of *ZPIs*. Its purpose is to receive the messages sent by *ZPUB* and perform a “job” on top of them.

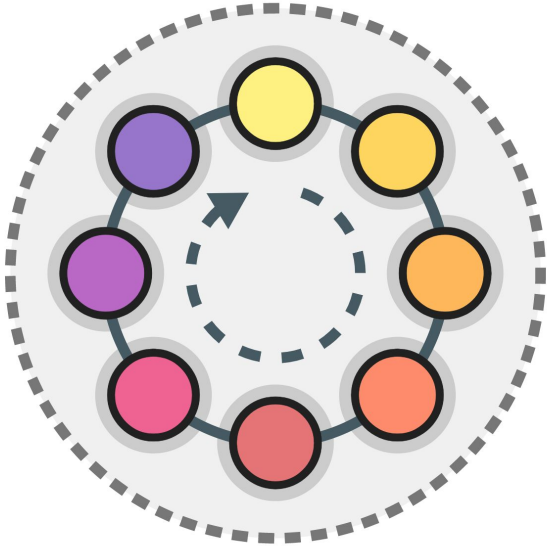
In this project case, the job consists of processing the data (filtering out some of the values) and sending it to *ZSUB*.



ZPIs cluster

ZeroMQ push-pull sockets were used to obtain a cluster of nodes. *ZPI* connects to a push socket (*ZPUB*) and starts receiving data from it.

If more than one *ZPI* is running, the data sent by *ZPUB* is split among them following a *Round Robin* scheduling. Furthermore, a *ZPI* could join or leave the cluster without any consequences.



System architecture: ZSUB

It receives processed data from the *Edge Computing* layer and proceeds to store it.

In the project case, *SQlite* was used as storage solution.

In a real world scenario, this part of the system could send the data to a more complex storage solution (e.g. in a cloud server in order to be analyzed later).

Conclusions

ZeroMQ turned out to be an excellent choice for the purpose of the project. It was reliable, fast and easy to work on.

We were able to transmit volumes of data far higher than the amount required in a normal use case, without any relevant loss.

The project itself can be found on *Github*, at the following URL: https://github.com/dmitry-mingazov/edge-da

Future works

The next step of the project would be testing it on devices with limited resources and see how it actually goes. In fact, the target devices would be boards with microprocessors designed specifically for the **IoT** world, such as the *STM32MP1* microprocessor series.