Case Study

Memi Lavi
www.memilavi.com

Case Study

- After learning about implementing Event Driven Architecture let’s put it into practice
- In this case study we’ll design an event-based system for receiving and processing noise telemetry from sensors
- The system will utilize two types of channels
- After designing it – we’ll actually build it ☺

Our System

- Introducing:

NOP

The NOise Processing system

NOP

- A system for receiving and processing noise data from external sensors
- The system should:
  - Receive the telemetry
  - Validate it
  - Notify clients on new data

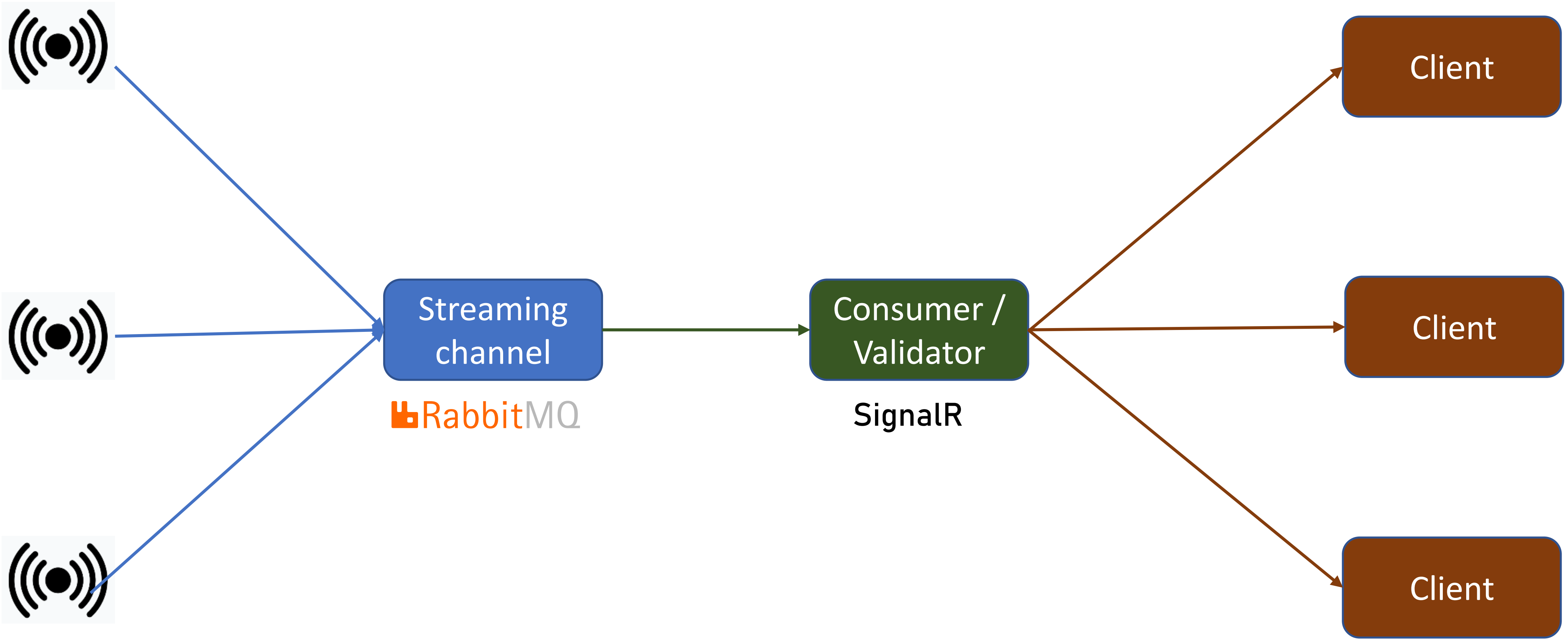
NOP

- The data is a number representing the decibels recorded
- Every sensor sends the data every 30 secs
- …That means that if there are a lot of sensors, there’s quite a lot of data…
- E.g. 1000 sensors => 33 msgs / sec

NOP Design Requirements

Handle load	Streaming engine should be used. Processors pull from the stream when possible
Validate the data	The first thing that should happen after receiving the data
Unknown number of clients	Classic events requirement
No sync users' commands	No synchronous actions required

NOP Event Driven Architecture



Clients subscribers

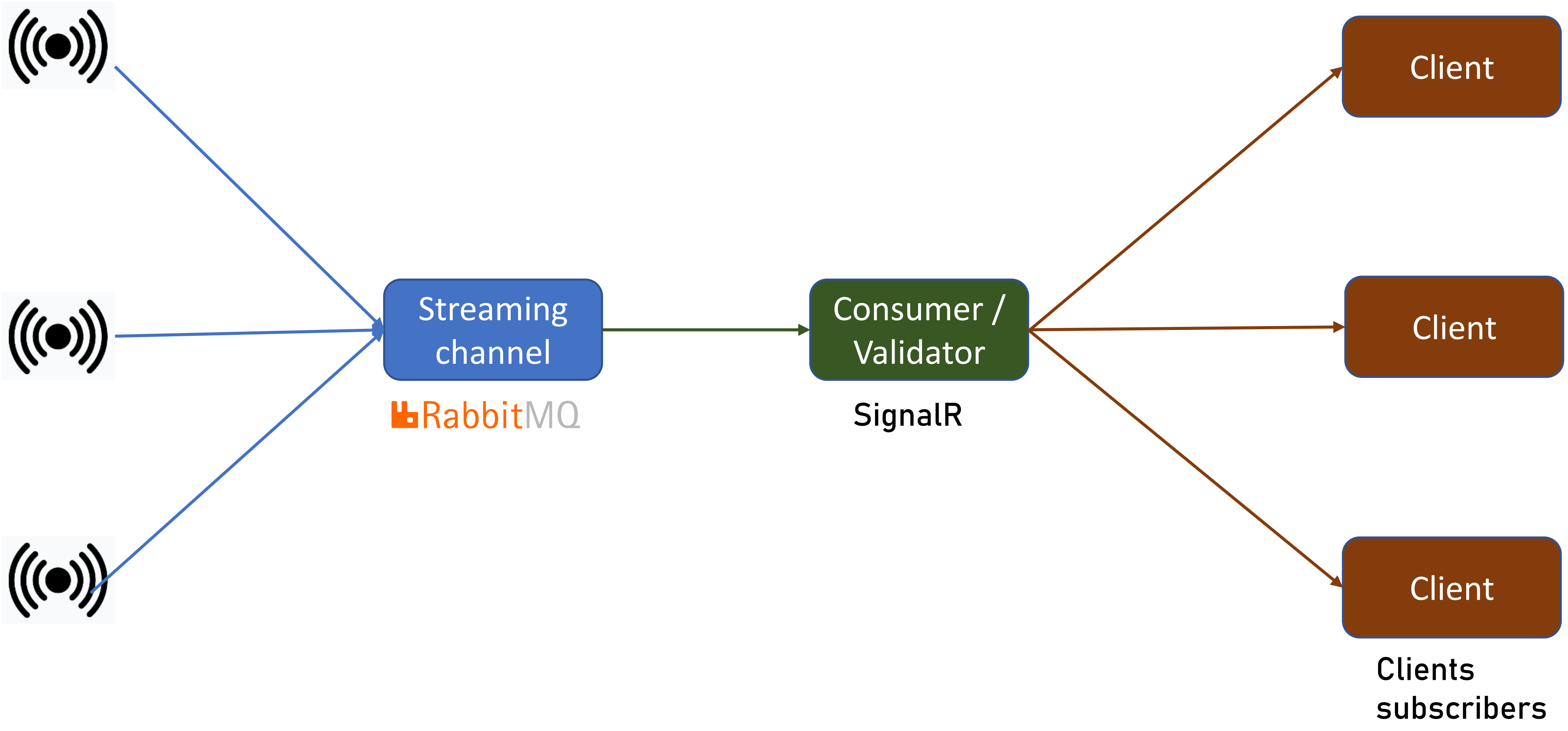
Preparing the Environment

- In order to build the app we need to install some software on our computer.
- We'll install:
  - .NET SDK
  - VS Code with some extensions
  - RabbitMQ

Preparing the Environment

- Note:
    - You don’t have to be a developer in order to build the app
    - We’ll go through all the steps

Summary


Streaming channel
RabbitMQ
Consumer / Validator
SignalR
Client
Client
Client
Clients subscribers

Summary

- We used two types of events mechanism
- No problem with that
- We didn’t implement:
    - Logging
    - Monitoring
    - Security

Summary

- Implementing Event Driven Architecture shouldn’t be too difficult
- The most important part is the design
- Make sure to select the best tools for the tasks
- Always try to use existing tools, do not reinvent the wheel