Microservices with NodeJS

Microservices in 30 minutes or less...

Andrés Rojas

- Project Lead @ WebCreek
- Product Developer
- Tech Speaker
- Programmer

Add me on LinkedIn

https://www.linkedin.com/in/andreserojasi/

Agenda

- Microservices Definition
- When to use Microservices?
- When not to use Microservices?
- What is Moleculer?
- A short example

What are Microservices?

“Microservices are small, autonomous services that work together.”

Sam Newman

“...an approach to developing a single application as a suite of small services, each running in its own process and communicating with lightweight mechanisms, often an HTTP resource API.”

Martin Fowler

What should a microservice be:

- Small in size
- Messaging enabled
- Bounded by contexts
- Autonomously developed
- Independently deployable
- Decentralized
- Built and released with automated processes

When to use Microservices?

Let’s make sure to make the right decision

When you want to:

- Increase your ROI and reduce your TCO (total cost of ownership)
- Have high Fault Tolerance
- Scale easily
- Work with multiple languages/tech (Not Recommended)
- Have faster Time to Market
- Improve the development process in huge apps

When not to use Microservices?

When you become a hammer, you see everything as a nail

It is not for you if:

- Business does not embrace new technologies easily
- There are no Devops in your company
- Product Scope is not clear enough
- Developer’s culture is not mature
- Too much bureaucracy to access resources
- Product is too small

What is Moleculer?

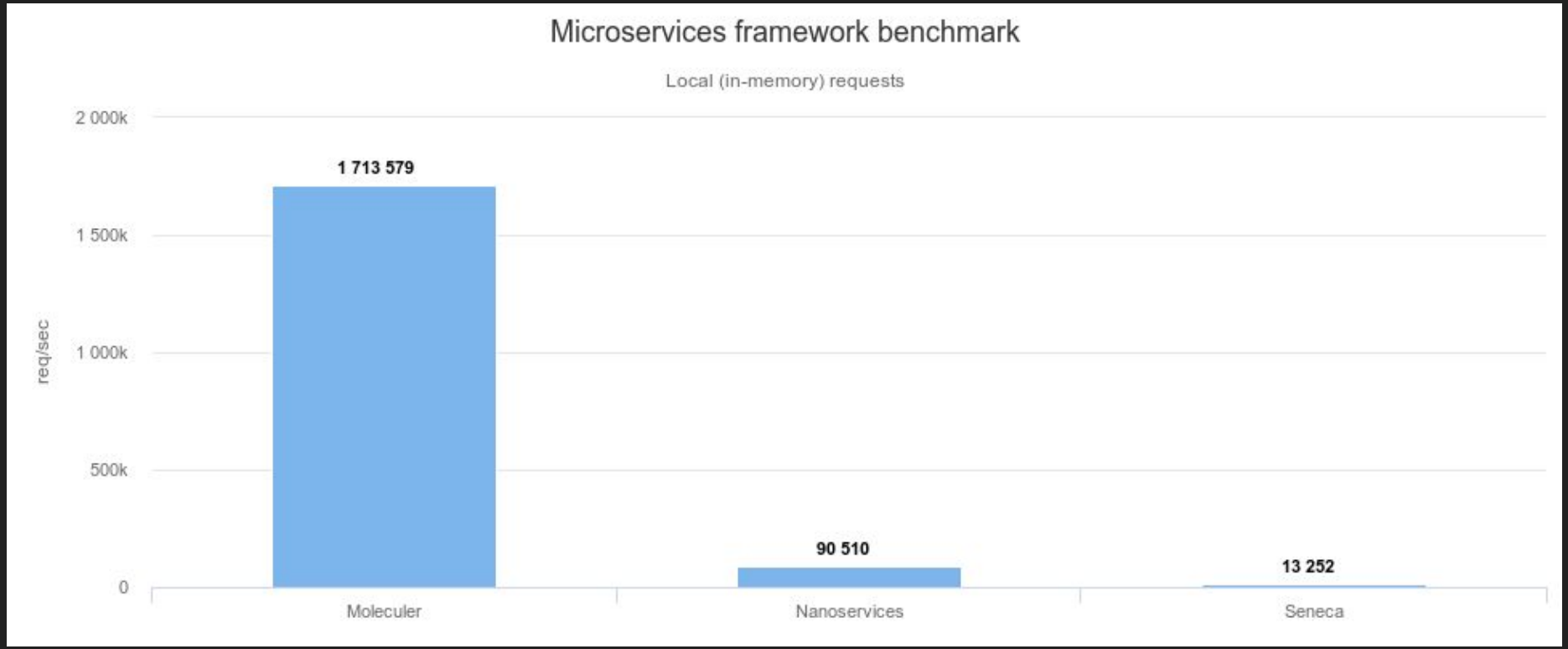
Microservices are easy, and moleculer makes them easier

Moleculer is:

“A fast, modern and powerful microservices framework for Node.js. It helps you to build efficient, reliable & scalable services.”

https://moleculer.services/

Why Moleculer? Because it is fast...



Why Moleculer? Because it is easy...

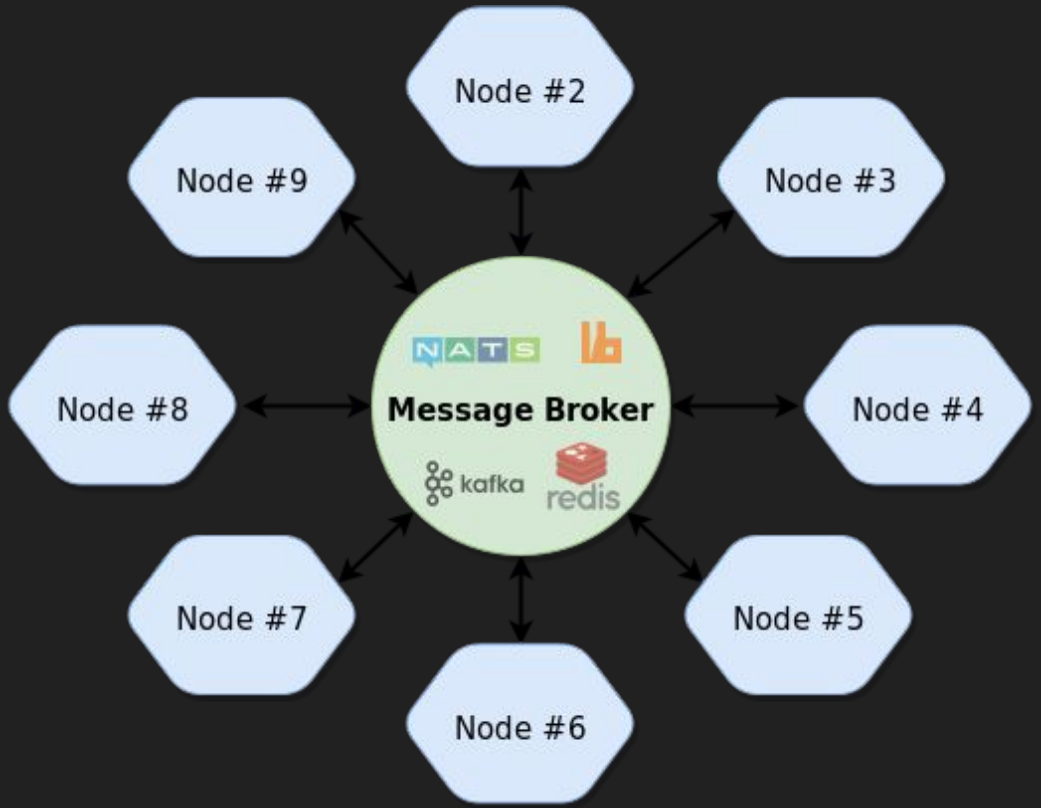
```
const { ServiceBroker } = require("moleculer");

// Create boker
const broker = new ServiceBroker();

// Create service
broker.createService({
    name: "math",
    actions: {
        add(ctx) {
            return Number(ctx.params.a) + Number(ctx.params.b);
        }
    }
});

// Start broker
broker.start()
    // Call service
    .then(() => broker.call("math.add", { a: 5, b: 3 }))
    .then(res => console.log("5 + 3 =", res));
```

And more important… It talks in different ways



A short example...

Let’s build a simple app using microservices...

We are building...

An API for a Library that will look for a book information and also return information about the Author.

Thanks!