Microservices 4 real

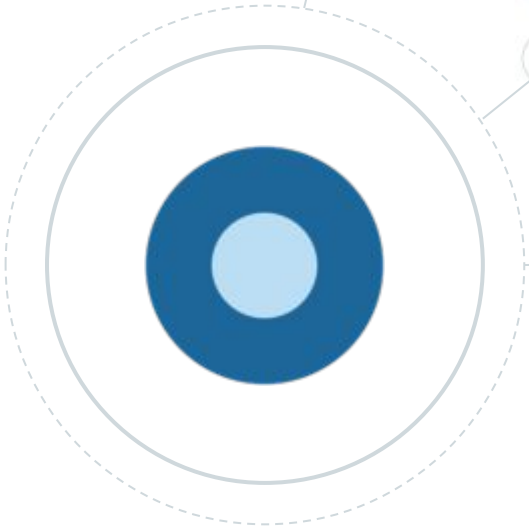
@tinchoz49

ingenious

Agenda

- The context 
- The issue 
- The fix 
- The code 
- Takeaways! 

The Context



The context

I used to work with **PHP** and I was **happy**!

The context

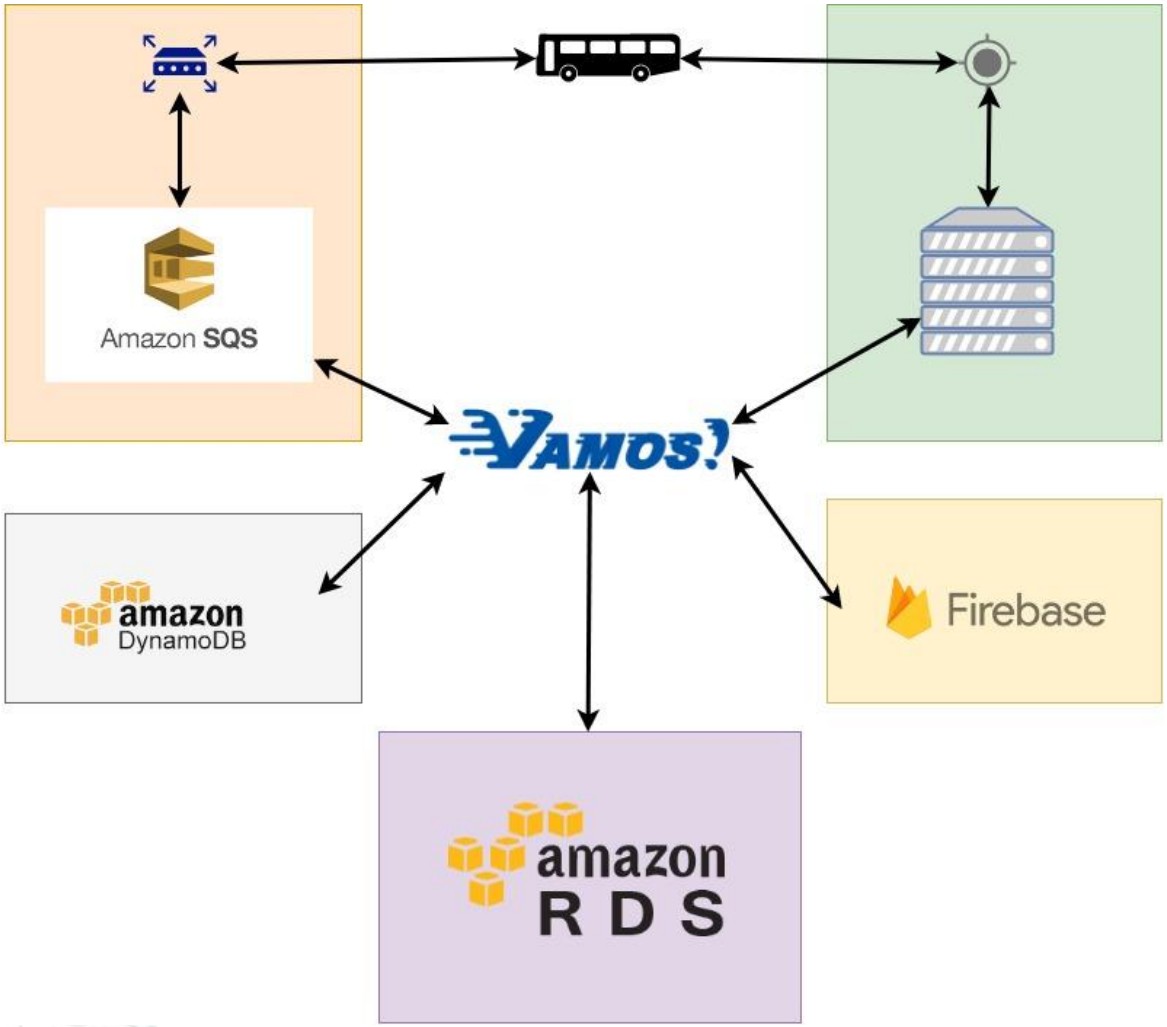
Well, not so much

The Monolithic Path

- # The Root Directory
  - # The `app` Directory
  - # The `bootstrap` Directory
  - # The `config` Directory
  - # The `database` Directory
  - # The `public` Directory
  - # The `resources` Directory
  - # The `routes` Directory
  - # The `storage` Directory
  - # The `tests` Directory
  - # The `vendor` Directory
- # The App Directory
  - # The `Broadcasting` Directory
  - # The `Console` Directory
  - # The `Events` Directory
  - # The `Exceptions` Directory
  - # The `Http` Directory
  - # The `Jobs` Directory
  - # The `Listeners` Directory
  - # The `Mail` Directory
  - # The `Notifications` Directory
  - # The `Policies` Directory
  - # The `Providers` Directory
  - # The `Rules` Directory

The Issue

The issue



The issue

- Support for scale **independently** services
- Support for **independently** deployments
- **Reliability**, fault tolerance
- I didn’t have **time**!!! ⏰

The Fix

Microservices

*The microservice **architectural style** is 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. These services are **built around business capabilities** and **independently deployable**.*

*Microservices are **small**, **autonomous** services that **work together.***

Microservices with Moleculer

*Moleculer is a fast, modern and powerful microservices **framework** for **Node.js**. It helps you to **build** efficient, reliable & scalable services.*

Moleculer provides many features for building and managing your microservices.

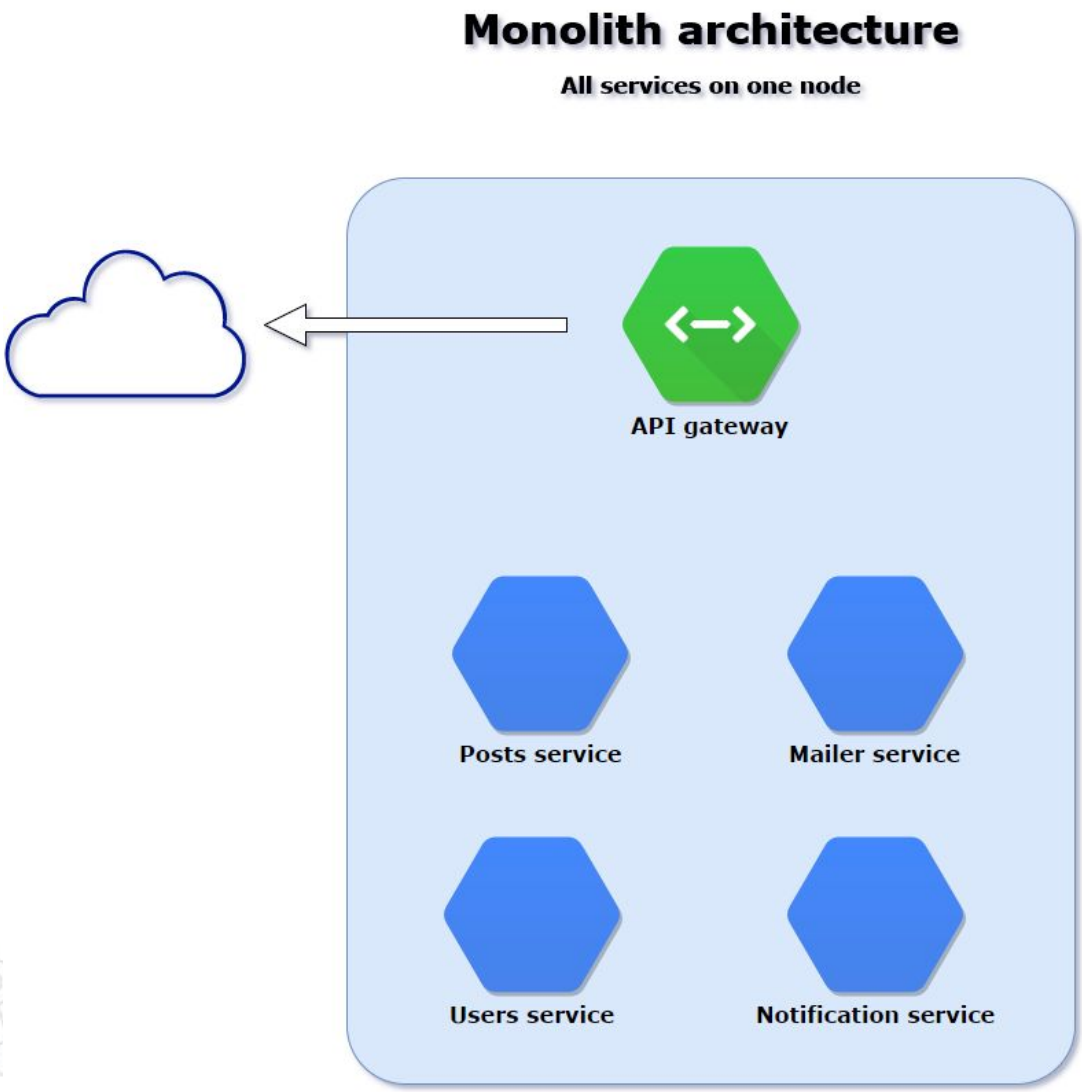
Moleculer - features

- Fault tolerance built-in service registry & auto discovery
- Support event driven architecture with balancing
- Load balanced requests & events (round-robin, random, custom)
- Pluggable transporters (NATS, MQTT, Redis)
- Built-in caching solution (memory, Redis)
- Health monitoring, metrics & statistics
- Promise-based solution (using bluebird)
- Request-reply concept

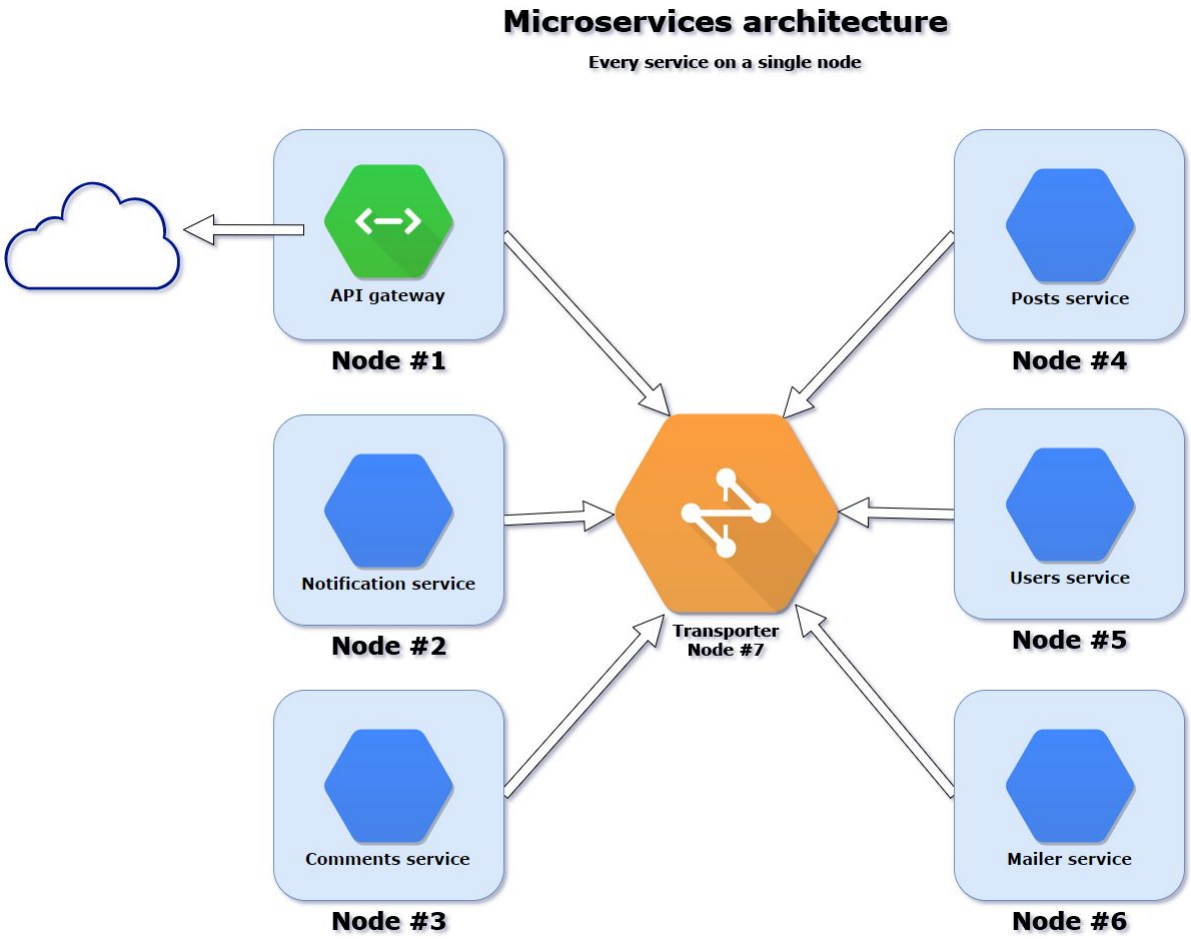
Moleculer - concepts

- Service 🔵
- Node 👻
- Service Broker 🏭
- Transporter 🔌
- Gateway 🚪

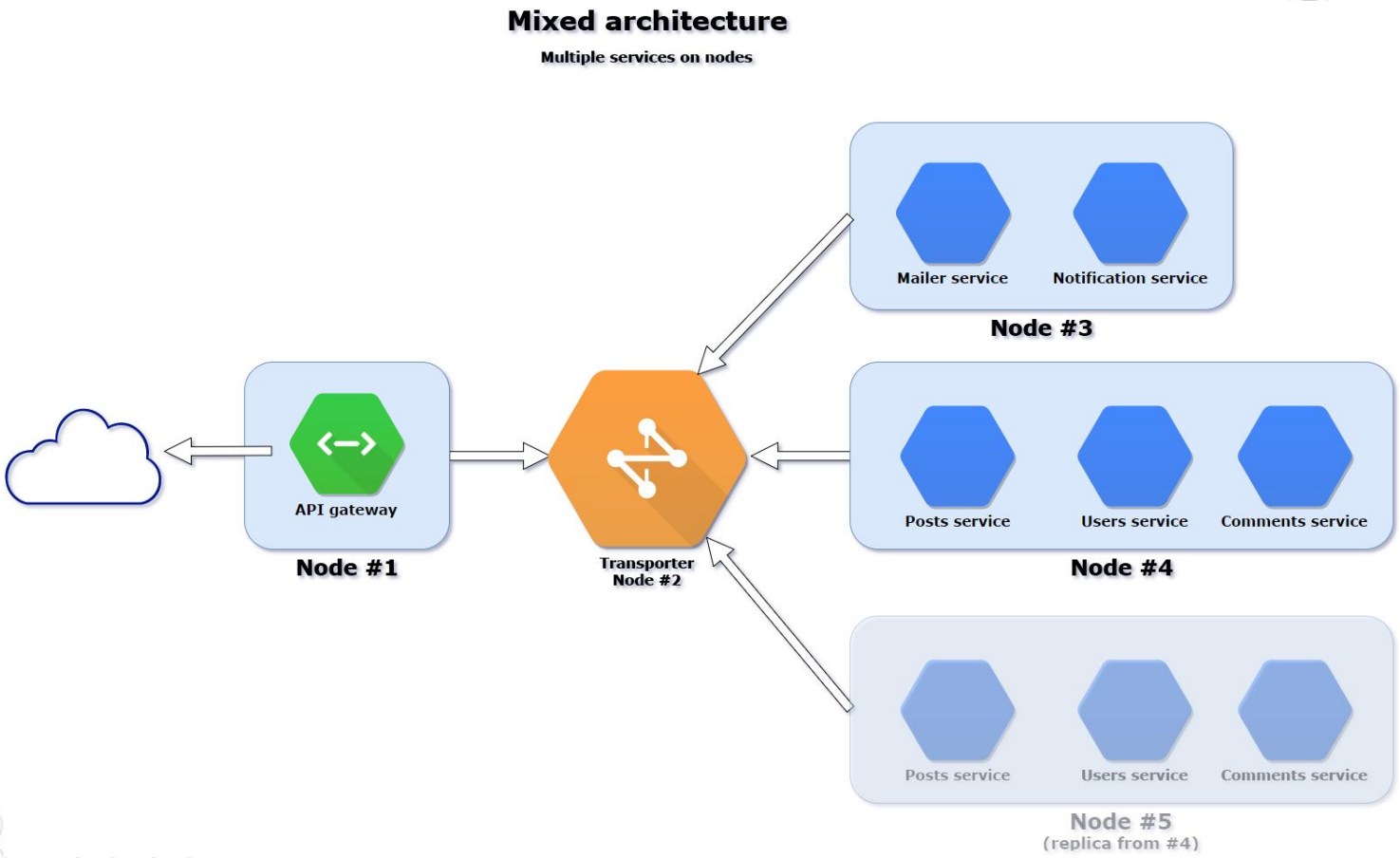
Moleculer - architecture



Moleculer - architecture



Moleculer - architecture



The code

Take-aways!

Takeaways!

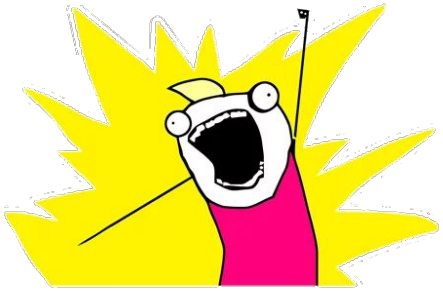
- We'll always have concepts like: Service, Node, Service Broker, Transporter and Gateway
- A microservice can be just a **module**
- The **frontier** of our architecture matter
- fault tolerant => easy deploy
- **time is precious**, try to focus on your problem and the solution

1.

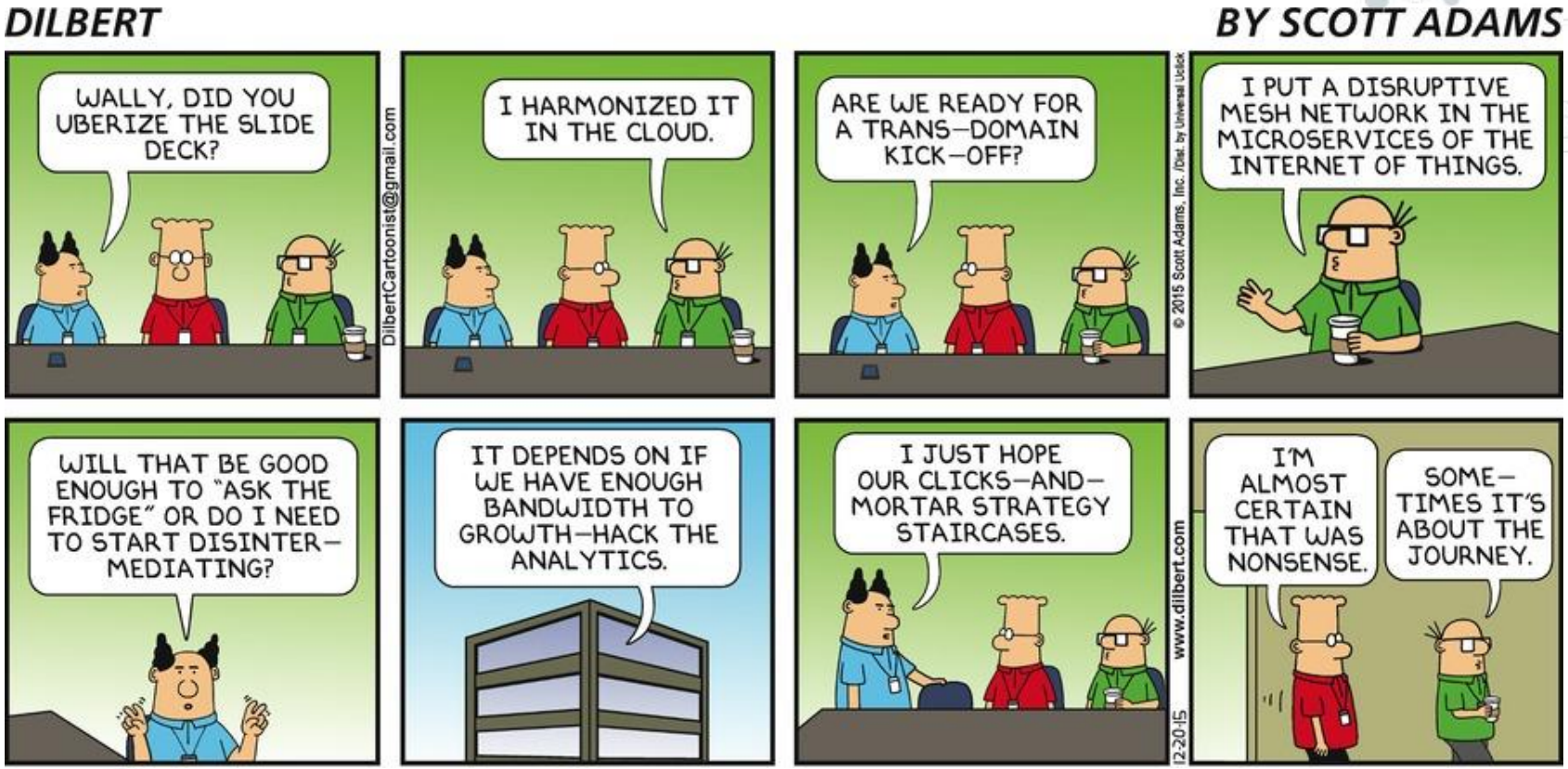
You don’t need microservices to modularized your app

2.

The future is decentralized



Gracias!


DILBERT
BY SCOTT ADAMS
WALLY, DID YOU UBERIZE THE SLIDE DECK?
I HARMONIZED IT IN THE CLOUD.
ARE WE READY FOR A TRANS-DOMAIN KICK-OFF?
I PUT A DISRUPTIVE MESH NETWORK IN THE MICROSERVICES OF THE INTERNET OF THINGS.
WILL THAT BE GOOD ENOUGH TO "ASK THE FRIDGE" OR DO I NEED TO START DISINTER-MEDIATING?
IT DEPENDS ON IF WE HAVE ENOUGH BANDWIDTH TO GROWTH-HACK THE ANALYTICS.
I JUST HOPE OUR CLICKS-AND-MORTAR STRATEGY STAIRCASES.
I'M ALMOST CERTAIN THAT WAS NONSENSE.
SOME-TIMES IT'S ABOUT THE JOURNEY.
DilbertCartoonist@gmail.com
© 2015 Scott Adams, Inc. /Dist. by Universal Uclick
www.dilbert.com
12-20-15

Contact

You can find me at:

twitter.com/tinchoz49

github.com/tinchoz49