And what if microservices are just the beginning?

Micro... frontends?

Aleksander Orchowski

October 10, 2019

Why do we want to decouple everything?

Why do we want to decouple everything?

Or in other words what's wrong with monolithic systems?

Why do we want to decouple everything?

Or in other words what's wrong with monolithic systems?

- We can't understand a big amount of code

Why do we want to decouple everything?

Or in other words what's wrong with monolithic systems?

- We can't understand a big amount of code
- So, we can't easily add new features

Why do we want to decouple everything?

Or in other words what's wrong with monolithic systems?

- We can't understand a big amount of code
- So, we can't easily add new features
- They are complex

Why do we want to decouple everything?

Or in other words what's wrong with monolithic systems?

- We can't understand a big amount of code
- So, we can't easily add new features
- They are complex
- One codebase can be developed effectively by 100 people?

Why do we want to decouple everything?

Or in other words what's wrong with monolithic systems?

- We can't understand a big amount of code
- So, we can't easily add new features
- They are complex
- One codebase can be developed effectively by 100 people?
- Let's skip performance and scalability

Always? It depends. The good architecture also relates to monoliths.

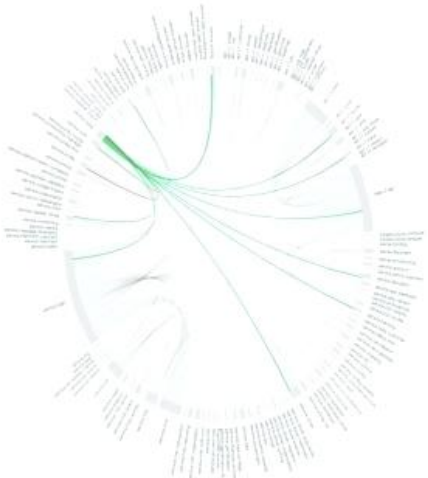
One of the biggest advantages of distributed systems is that knowledge is also distributed through teams. They can focus on a single part of business requirements.
Also, It's possible to easily **replace** each part.

But, everything is loosely coupled...

Architectuure!

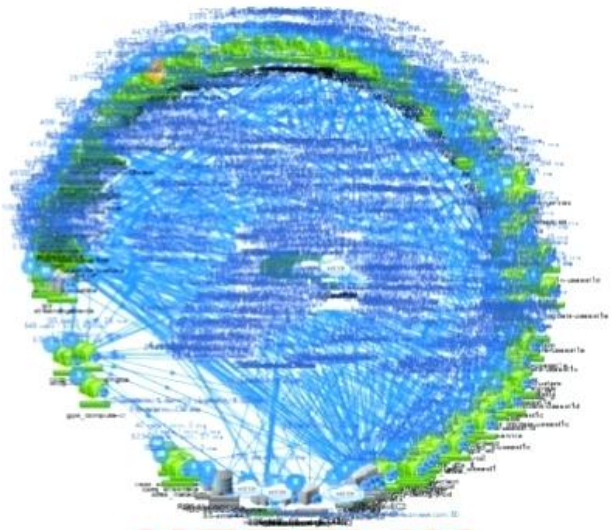
Do microservices solve your problems? ARE YOU SURE?

450 microservices



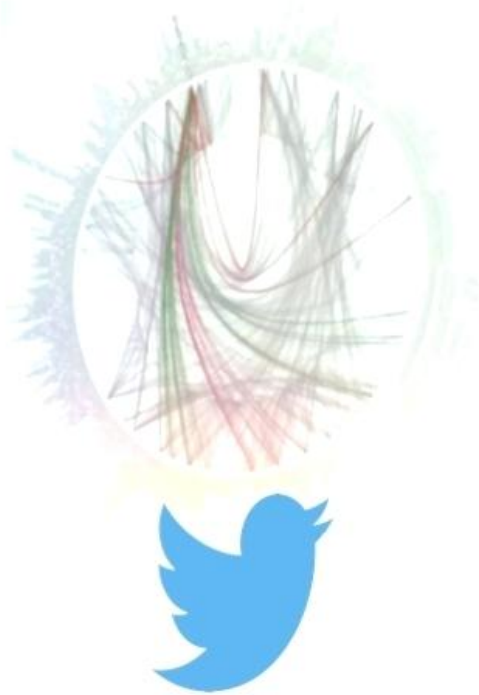
HAIL O

500+ microservices



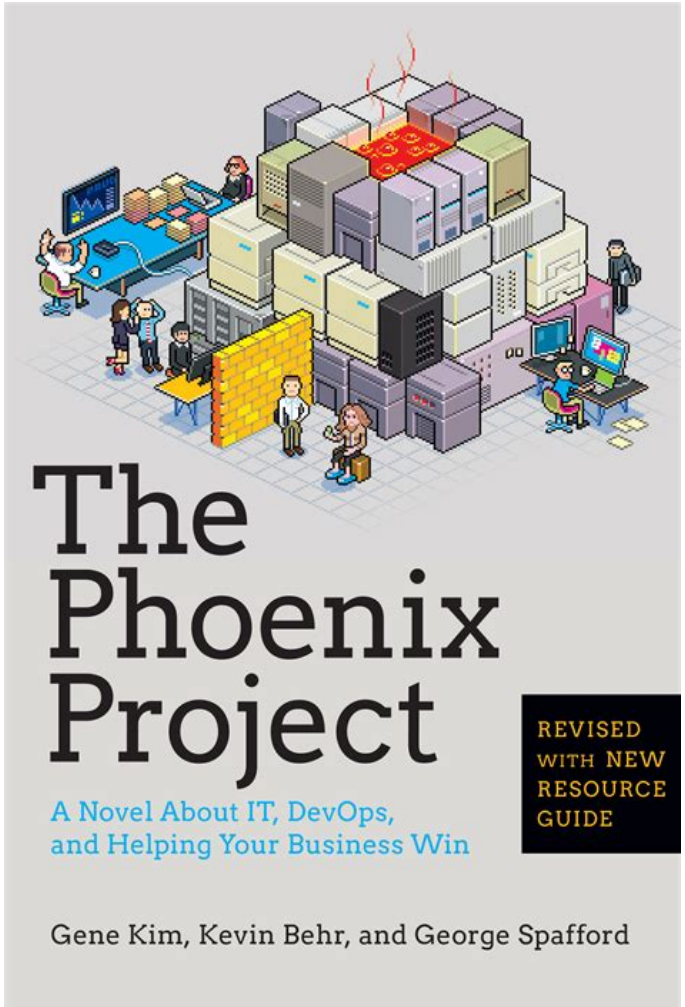
NETFLIX

500+ microservices



Source:
Netflix: http://www.slideshare.net/BruceWong3/the-case-for-chaos
Twitter: https://twitter.com/adrianco/status/441883572618948608
Hail-o: https://sudo.hailoapp.com/services/2015/03/09/journey-into-a-microservice-world-part-3/

For that - DevOps



Okay... 500 Microservices but...

Let's connect everything on UI...

Okay... 500 Microservices but...

Let's connect everything on UI...

Frontend that we know

Now we are going to talk about approaches used at today's systems.

I use the example of the blog app:

- Service which manage users, authentication etc.
- Service for articles (listing, creating, editing, etc.)
- Frontend is SPA/SPA like

Monolith

It's important to understand something about structure of application. So, we have one big block of code:

Pros&Cons

+

- Everything at one place
- Probably - at one pull request we can add new feature
- Deployed once with a frontend. We are sure that both layers work fine together

Pros&Cons

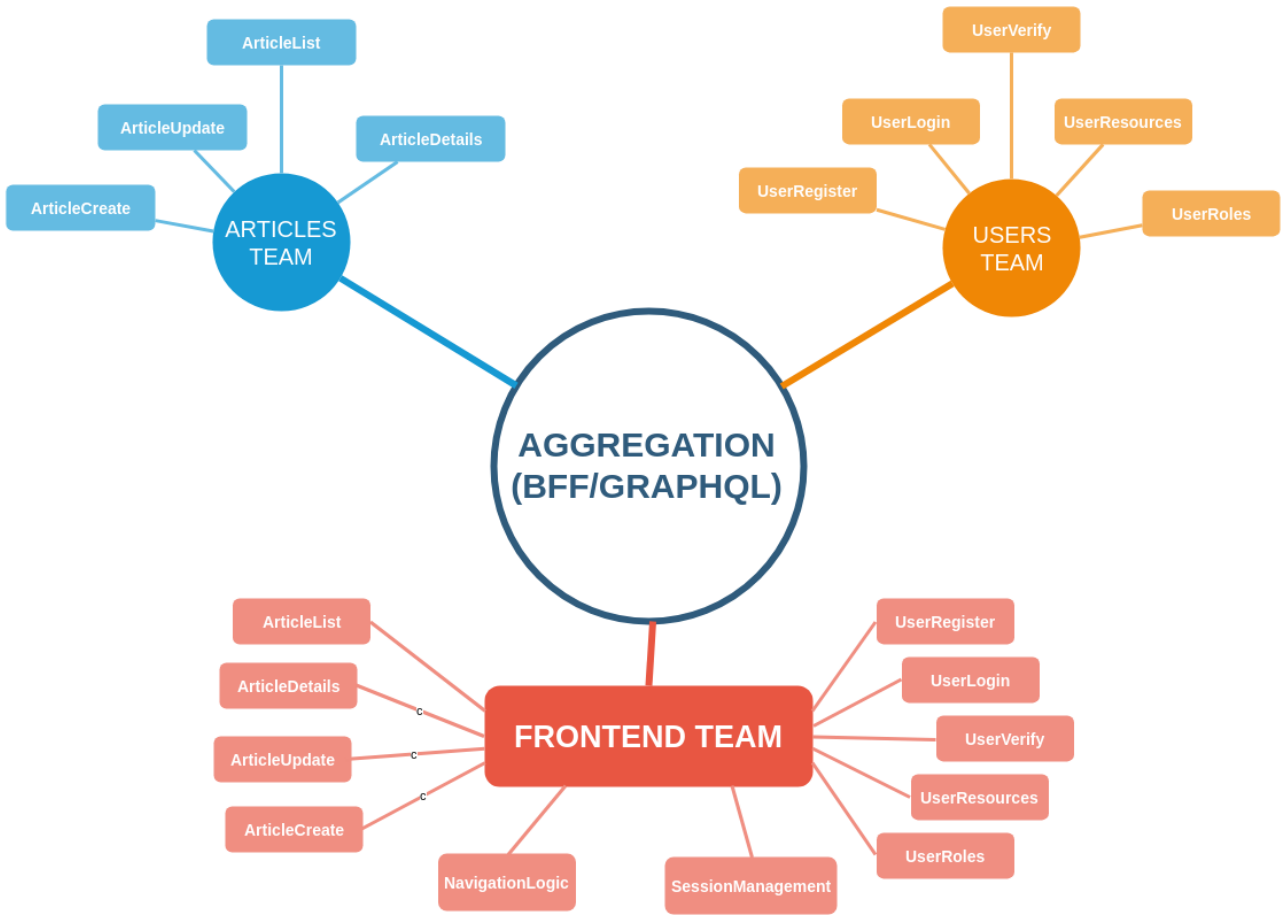
+

- Everything at one place
- Probably - at one pull request we can add new feature
- Deployed once with a frontend. We are sure that both layers work fine together

−

- Vertical scalability
- Complex inside
- A lot of legacy code can exist
- High entry threshold

Microservices



Pros&Cons

+

- Working parallel

Pros&Cons

+

- Working parallel
- Many smaller codebases

Pros&Cons

+

- Working parallel
- Many smaller codebases
- This codebases are really small

Pros&Cons

+

- Working parallel
- Many smaller codebases
- This codebases are really small
- Microservices are trendy now

Pros&Cons

+

- Working parallel
- Many smaller codebases
- This codebases are really small
- Microservices are trendy now
- We can use a lot of cool tools

Pros&Cons

+

- Working parallel
- Many smaller codebases
- This codebases are really small
- Microservices are trendy now
- We can use a lot of cool tools
- We are still skipping performance and scalability

Pros&Cons

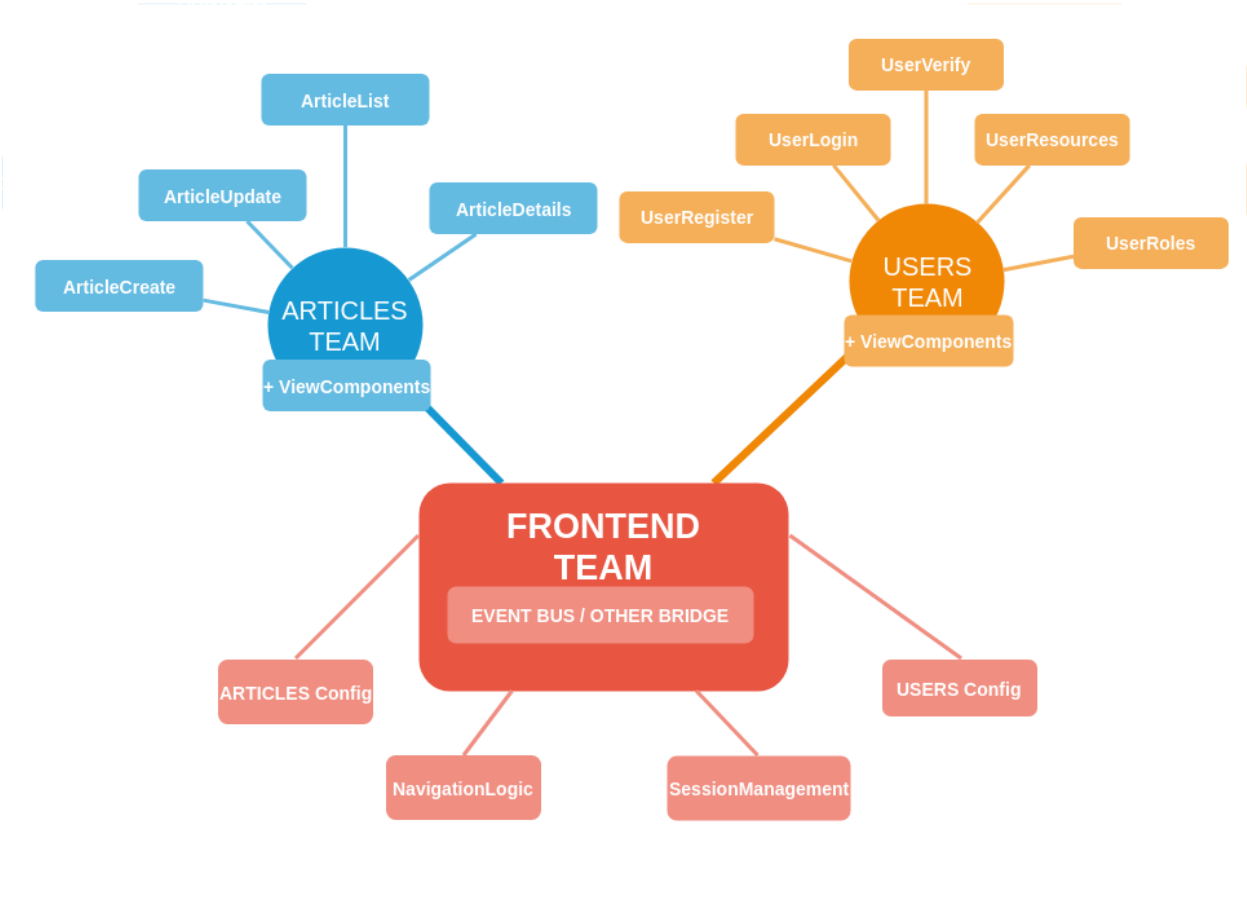
+

- Working parallel
- Many smaller codebases
- This codebases are really small
- Microservices are trendy now
- We can use a lot of cool tools
- We are still skipping performance and scalability

−

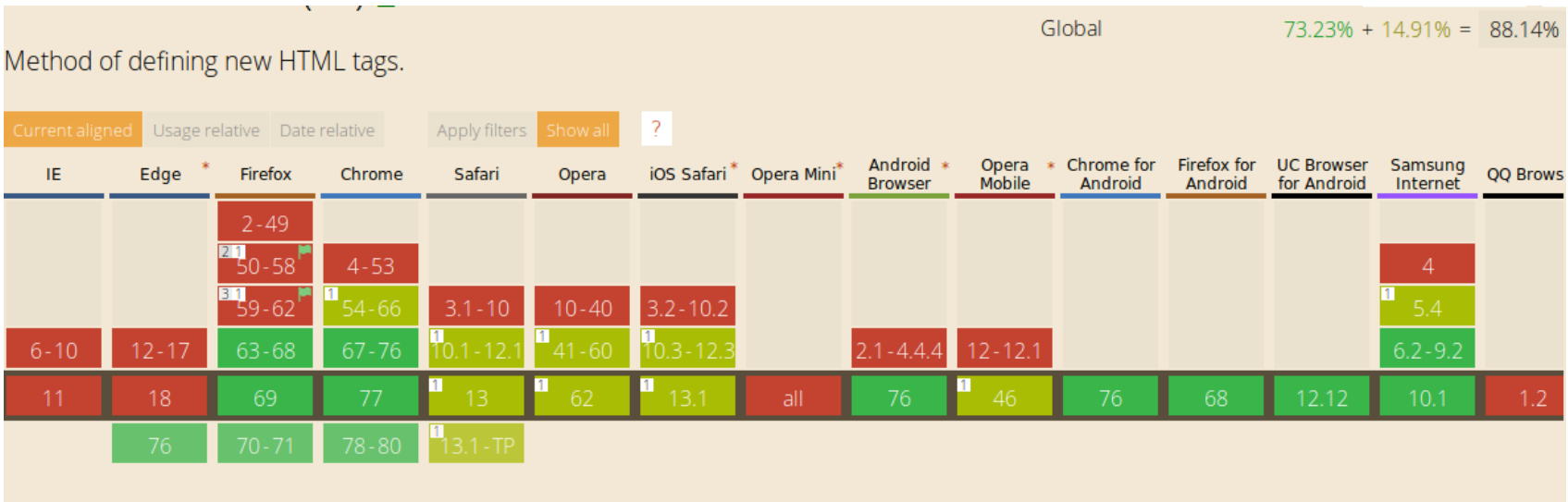
- Many codebases - additional automation costs
- Integration needed (+ more testing etc.)
- Infrastructural costs
- Eventual consistency
- ...
- Just a lot of work more

Microfrontends

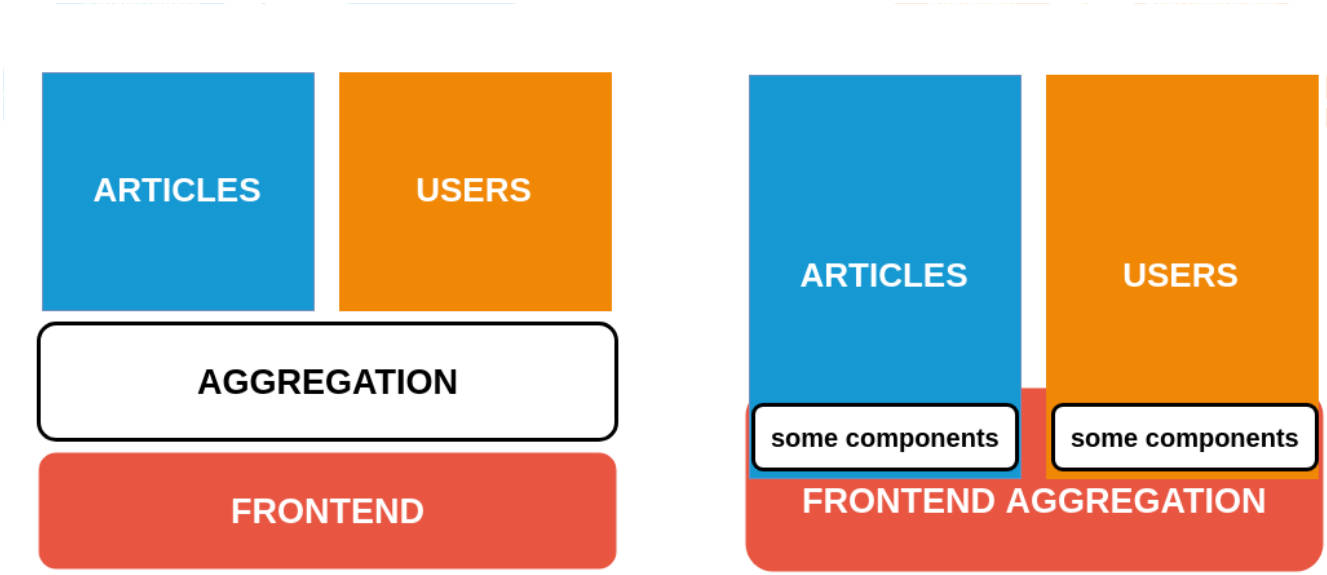


Webcomponents

Additional HTML tags defined in java script files, which can be shared between a lot of pages.



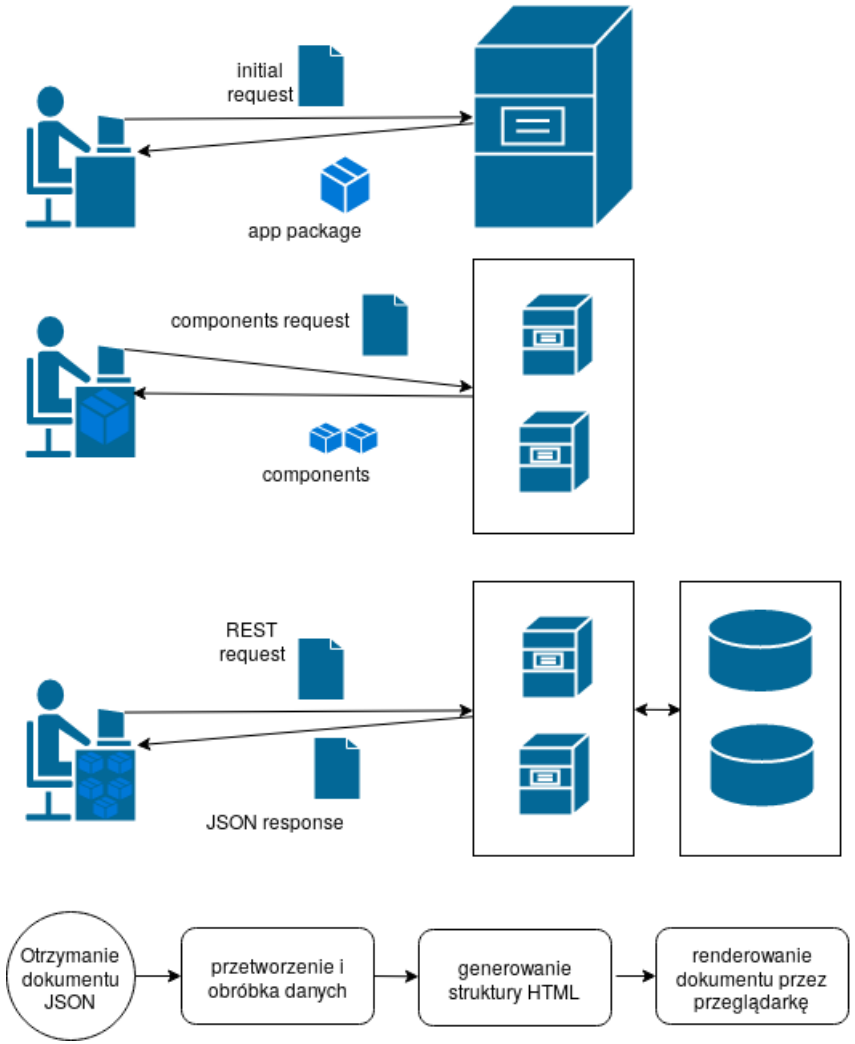
Comparison



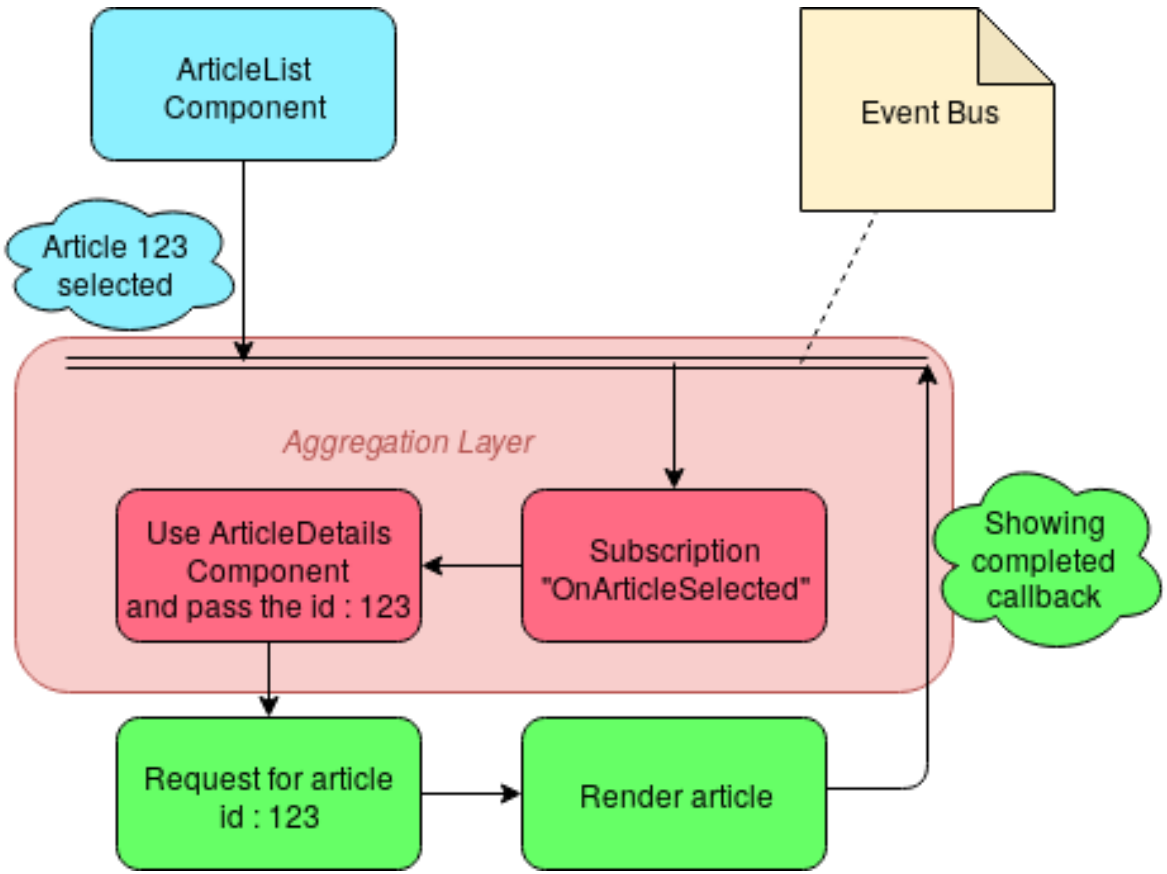
g

DEMO TIME

Component loading



Components messaging



Dev & Ops

DevOps culture is mandatory at the microservices environment, but is it sufficient with microfrontends?

Cross functional teams/Cross functional people

Backend
Frontend
Operations

Questions?

- My blog page: https://orchowskia.com
- Twitter: @orchowski