Analyze of architecture of presentation layer in microservices applications

with microfrontend

Aleksander Orchowski

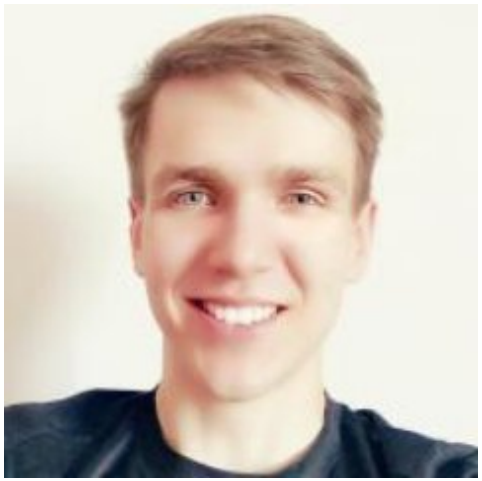
University of Zielona Góra

Promoter: Dr Grzegorz Bazydło

About me

Aleksander Orchowski

- IT Science student
- Software Developer for 3 years
- Java, .NET, NodeJS
- Blogger https://orchowskia.com



Outline

Introduction
- Why do we want to distract everything?
- Architectuure!

Frontend that we know
- Monolith
- Microservices

Microfrontends

Diploma

Why do we want to distract everything?

Or 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 20 people?
- Let's skip performance

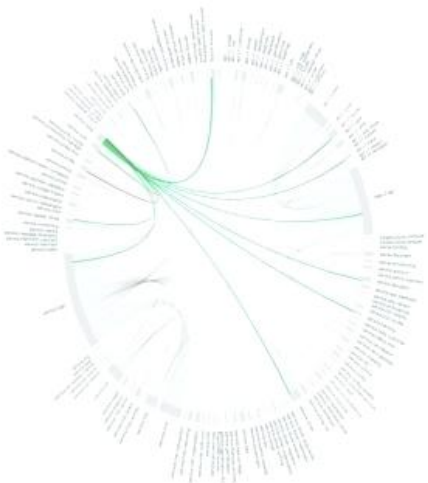
Always? It depends. The good architecture also relates to monoliths.

The biggest advantage of distributed systems is that knowledge is also distributed through teams. Also, we can easily (or should) DELETE each part.

Architectuure!

Do microservices solve your problems? ARE YOU SURE?

450 microservices

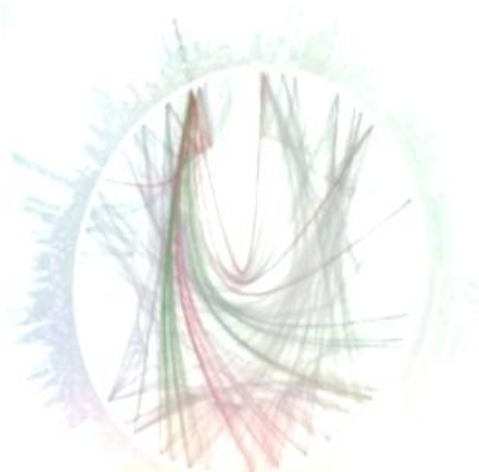


HAIL O

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/

Frontend that we know

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

I use the example about the blog page:

- Microservice which manage users, authentication etc.
- Microservice 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

+

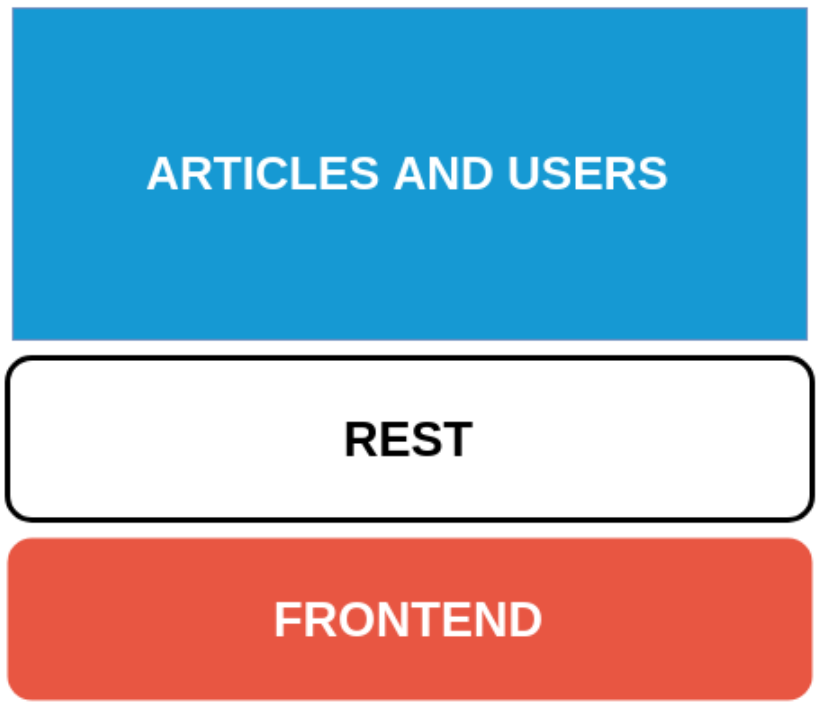
- Everything at one place
- Probably - at one pull request we can add new feature
- Deployed once with a frontend. We are sure that both layers work fine togeth

−

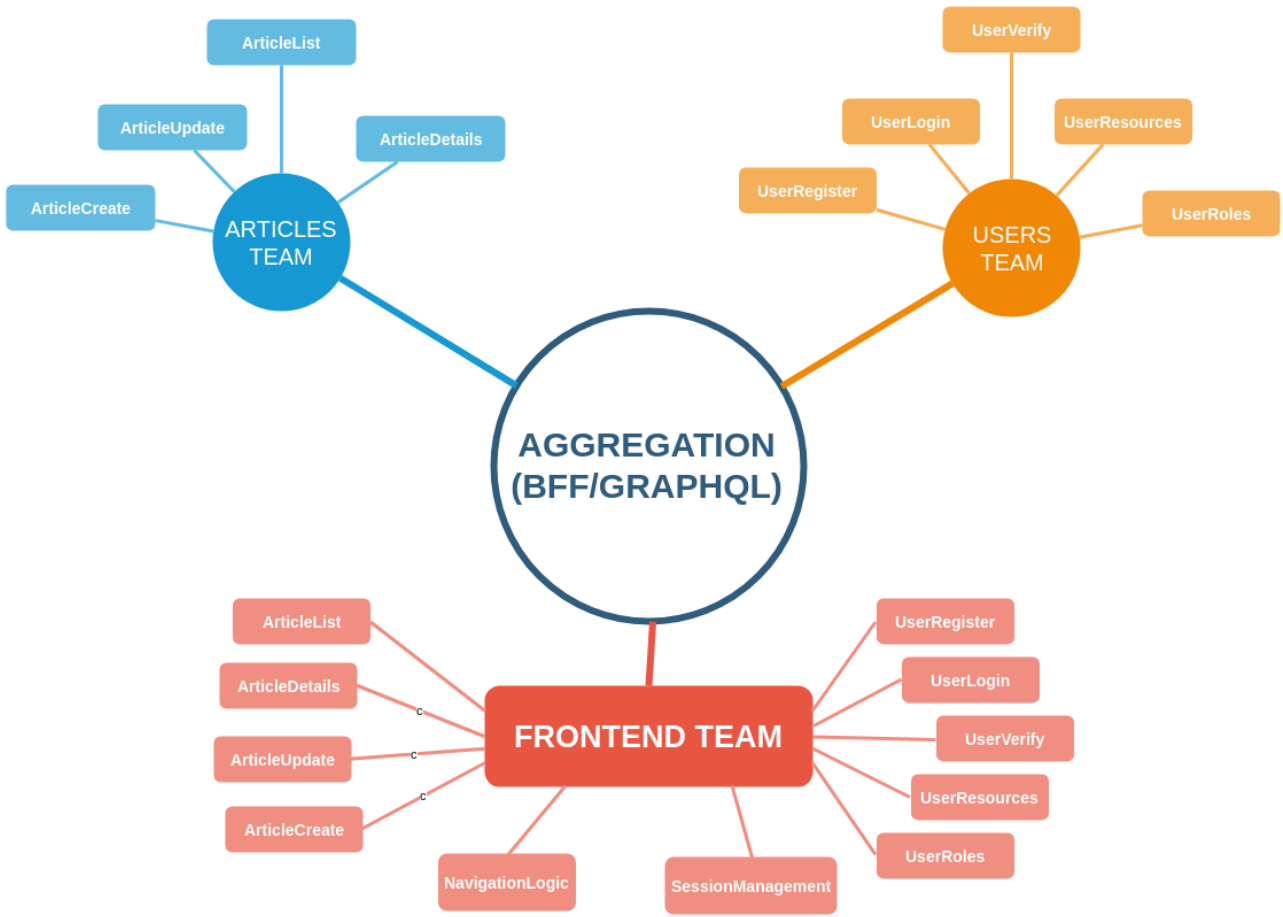
- Scalability
- Complex inside
- A lot of legacy code can exist

Monolith with REST

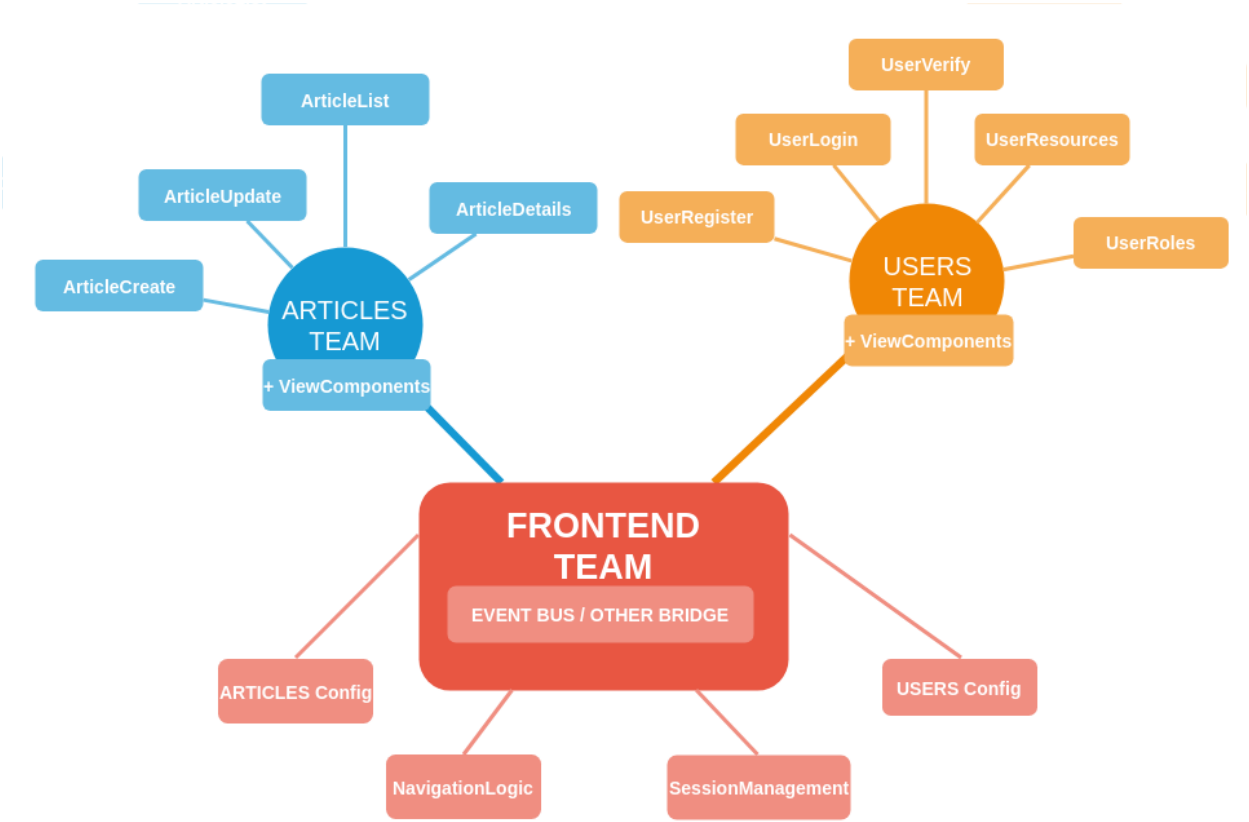
We(developers) like to have layers and separations. So a lot of systems have REST interface(or similar). We can work more parallel. Also, we have two smaller codebases.



Microservices

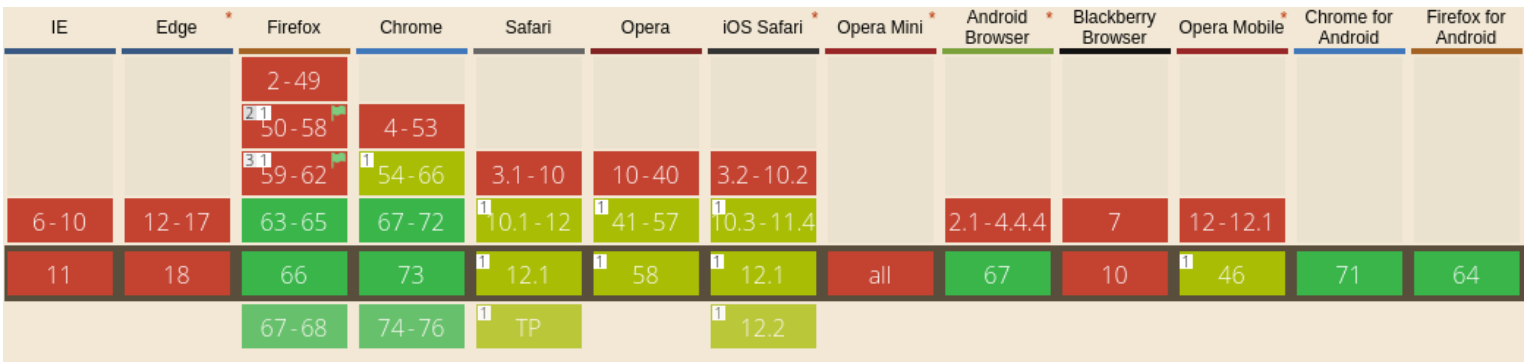


Microfrontends

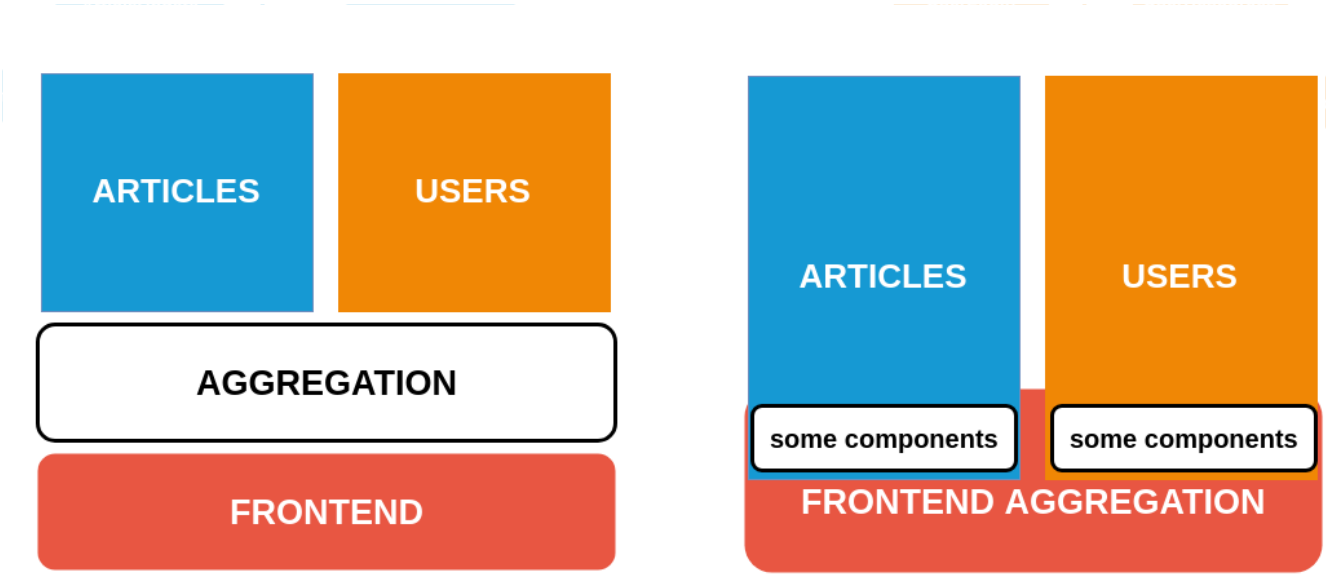


Webcomponents

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



Comparison


ARTICLES
USERS
AGGREGATION
FRONTEND
ARTICLES
USERS
some components
some components
FRONTEND AGGREGATION

Diploma work

Project area

- Microfrontends 100%
- Microservices + standard frontend 100%

Theory area

- Reading needed literature : 70%
- Diagrams are prepared

Questions?