Drabina Ekspertów

bottega IT SOLUTIONS

Ścisły przewodnik po aspektach miękkich – część II

Enterprise makeover

Making sense of agile requirements

Przejrzysty i testowalny kod na Androidzie?

REST w praktyce - tej dobrej i tej złej

Skalowanie i integracja systemów w asynchronicznym stylu

Do you think you're doing microservice architecture?

CQRS dla każdego

Kiedy, jak i po co migrować na NoSQL

www.bottega.com.pl **facebook.com/BottegaITSolutions**

About us

ŁUKASZ SZCZĘSNY

Systems engineer at 4FINANCE it

Co-organizer of the Warsaw Linux User Group

Fan of automation and DevOps

4FINANCE it

Twitter: @wybczu

Blog: http://wybcz.pl

Homepage: http://wybcz.pl

bottega IT SOLUTIONS

4DEVELOPERS

About us

MARCIN GRZEJSZCZAK

Software Architect at 4FINANCE it

Author of "Mockito Instant", "Mockito Cookbook" books

Co-author of the Groovy core's @Builder AST

Co-founder of the Warsaw Groovy User Group

Co-author of "micro-infra-spring" lib

4FINANCE it

Twitter: @MGrzejszczak

Blog: http://toomuchcoding.blogspot.com

Homepage: http://marcin.grzejszczak.pl

bottega IT SOLUTIONS

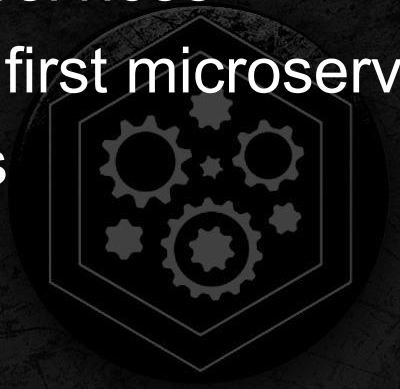
4DEVELOPERS

Agenda

short intro to microservices

how to deploy your first microservice

microservice pitfalls



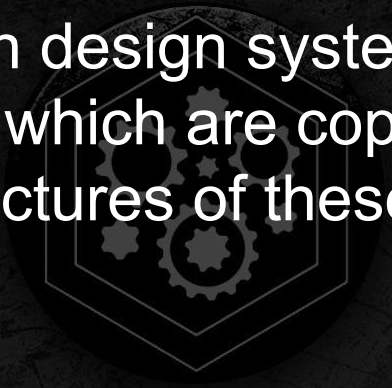
Agenda

short intro to microservices

how to deploy your first microservice

microservice pitfalls

Conway’s Law

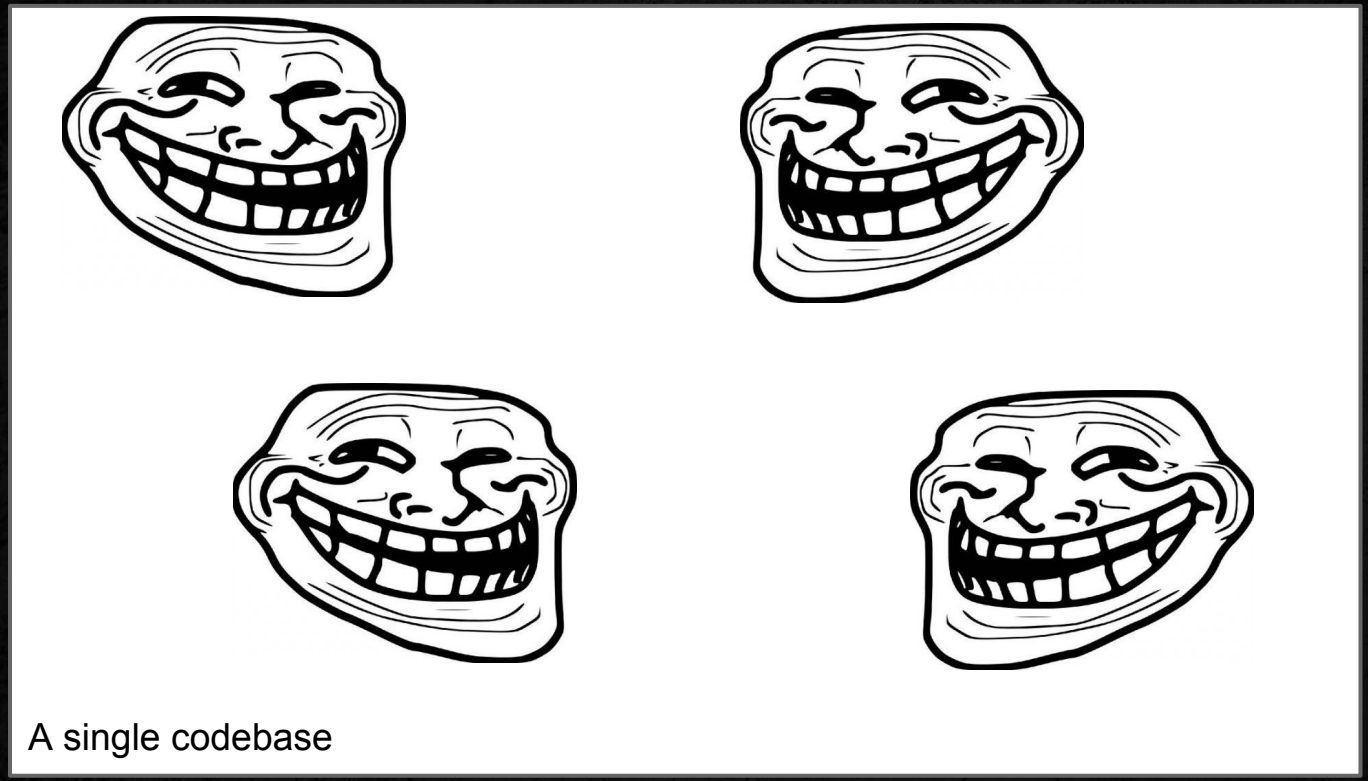


Organizations which design systems ... are constrained to produce designs which are copies of the communication structures of these organizations

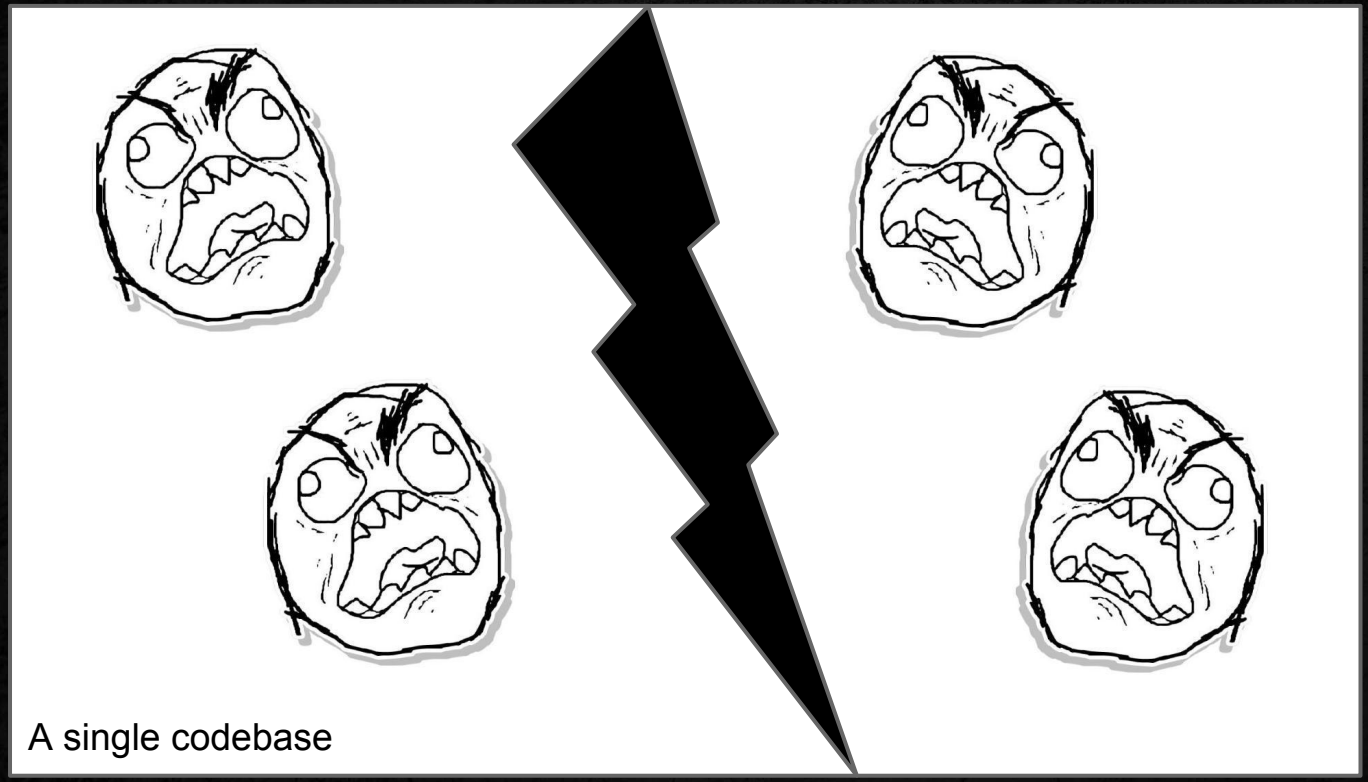
— M. Conway

Conway, Melvin E. (April 1968), How do Committees Invent?, Datamation 14 (5): 28–31, retrieved 2009-04-05

Conway’s Law in practice



Conway’s Law in practice



Conway’s Law in practice

Concept:

one team

two countries

one codebase

Conway’s Law in practice

Reality:

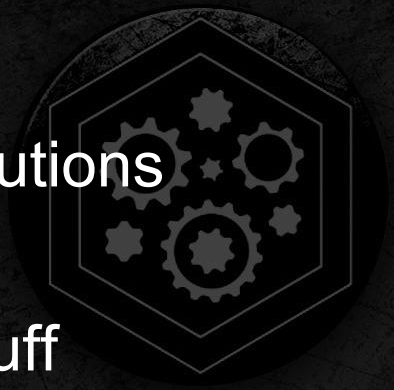
two teams

two countries

one codebase

Conway’s Law in practice

Effect:

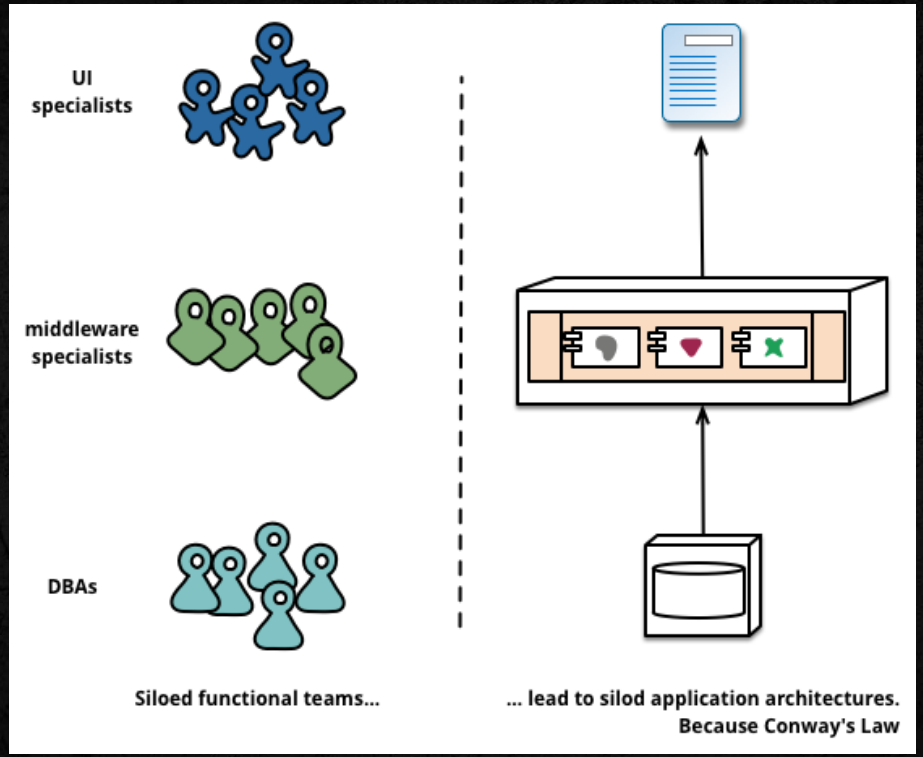


two different solutions

solving same stuff

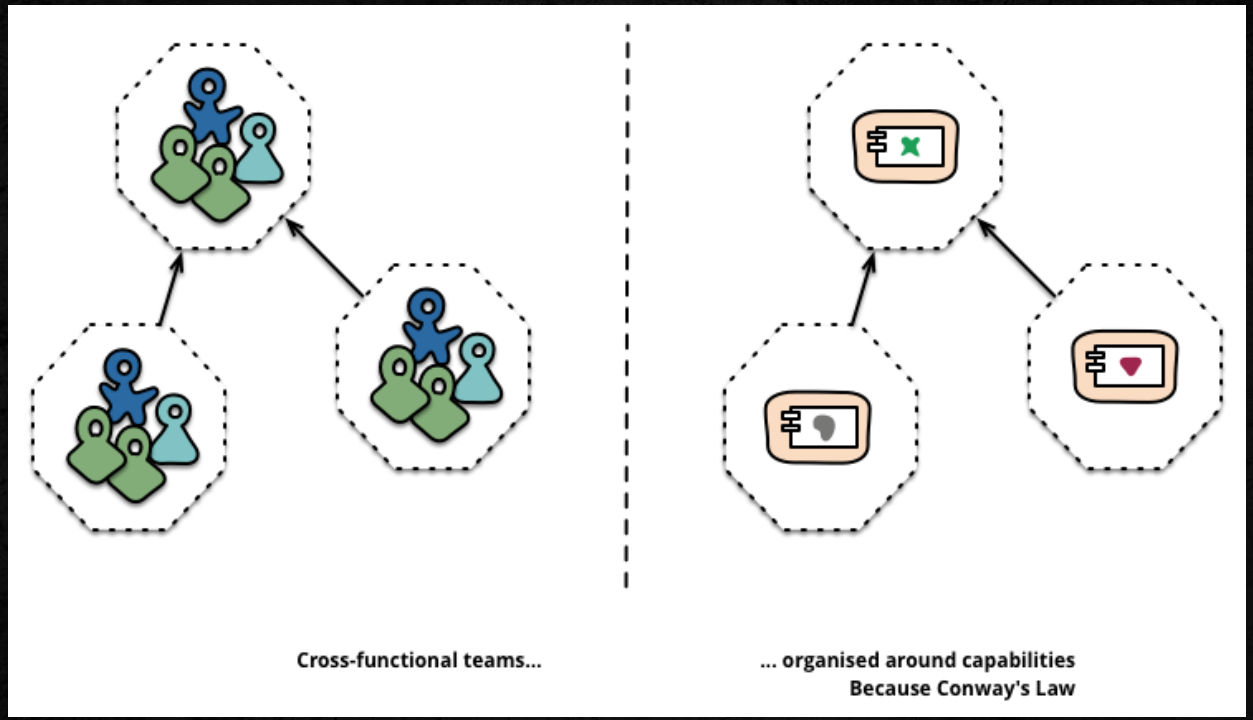
one codebase

Conway’s Law - siloed teams



extract from http://martinfowler.com/articles/microservices.html

Conway’s Law - cross functional teams



extract from http://martinfowler.com/articles/microservices.html

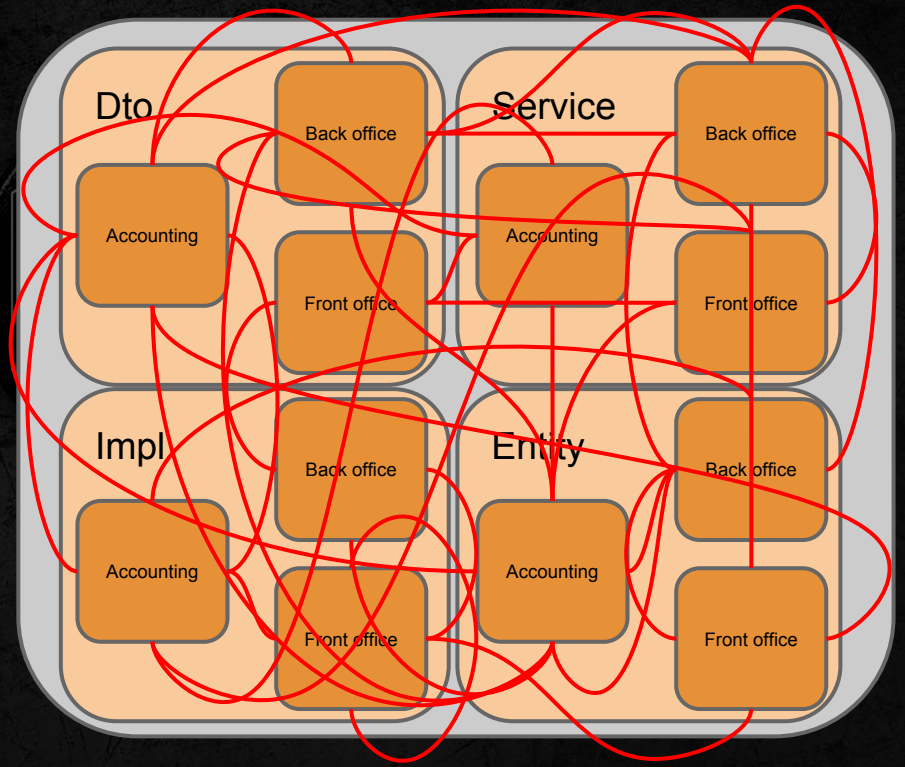
Business flow

Common problematic code flow

monolith

many programmers

big organization

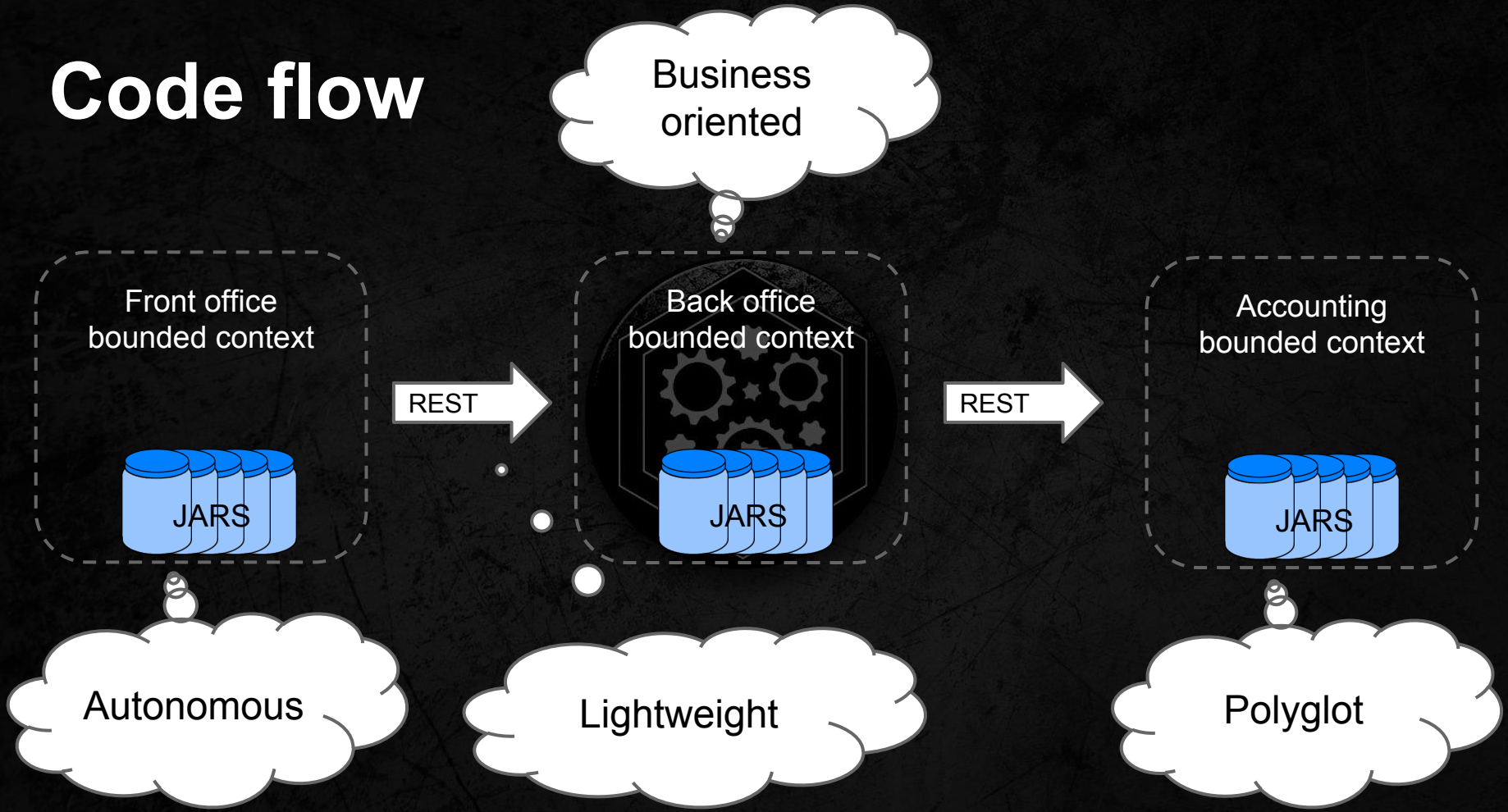


Common problematic code flow

Looks familiar?

http://www.foodnetwork.com/topics/spaghetti-recipes.html

Code flow


Business oriented
Front office bounded context
JARS
REST
Back office bounded context
JARS
REST
Accounting bounded context
JARS
Autonomous
Lightweight
Polyglot

Microservices vs ESB

http://www.banzaj.pl/pictures/sport/boks/Haye_Walujew/haye_vs.walujew_2.jpg

Microservices vs ESB

Enterprise Service Bus

intelligent communication layer between services

provides routing, transformations etc

Microservices vs ESB

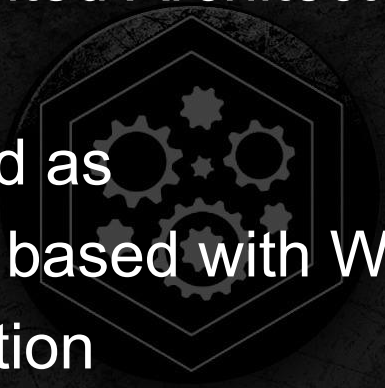
Microservices approach favors

smart endpoints (services)

dumb pipes (means of communication)

Microservices vs SOA

SOA - Service Oriented Architecture - a very broad topic



Typically understood as

- XML and SOAP based with WSDL
- ESB based solution

Microservice may be called “more thoroughly described SOA”

Agenda

short intro to microservices

how to deploy your first microservice

microservice pitfalls

Write code

As a developer

I want my microservice codebase to be small

I want to be fully responsible for supporting that service

I don’t want people from outside my team to push changes to my codebase

Write code

introduce code review / working via Pull Requests

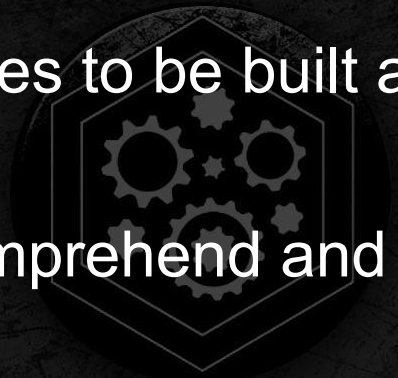
dev team responsible for CD pipeline

dev team receives all alerts

Build it

As a developer

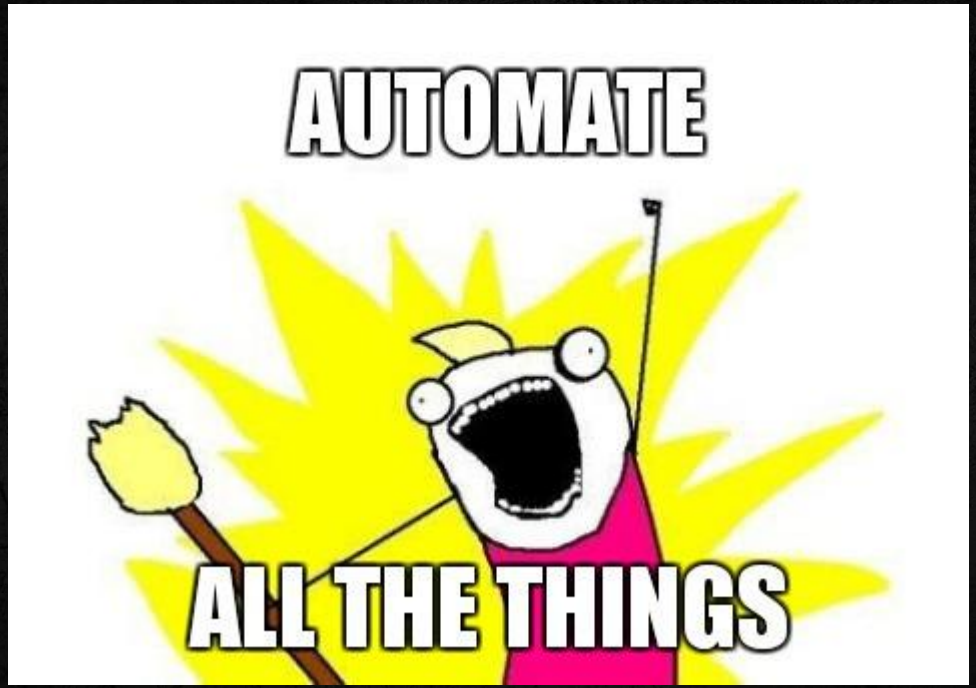
I’d like all services to be built alike



it’s easier to comprehend and support

I’d like to have fast feedback if my code works

Build it



Build it

Jenkins as a Code

Jenkins master and slaves deployment

Jenkins’ jobs creation

one CD pipeline template to rule them all

Build it

```
def project = 'quidryan/aws-sdk-test'
def branchApi = new URL("https://api.github.com/repos/${project}/branches")
def branches = new groovy.json.JsonSlurper().parse(branchApi.newReader())
branches.each {
    def branchName = it.name
    job {
        name "${project}-${branchName}".replaceAll('/','-')
        scm {
            git("git://github.com/${project}.git", branchName)
        }
        steps {
            maven("test -Dproject.name=${project}/${branchName}")
        }
    }
}
```

Test it

As a developer

I don’t want to hardcode service’s IPs and ports

I don’t want to set up whole environment for tests

I’d like to test my application in isolation

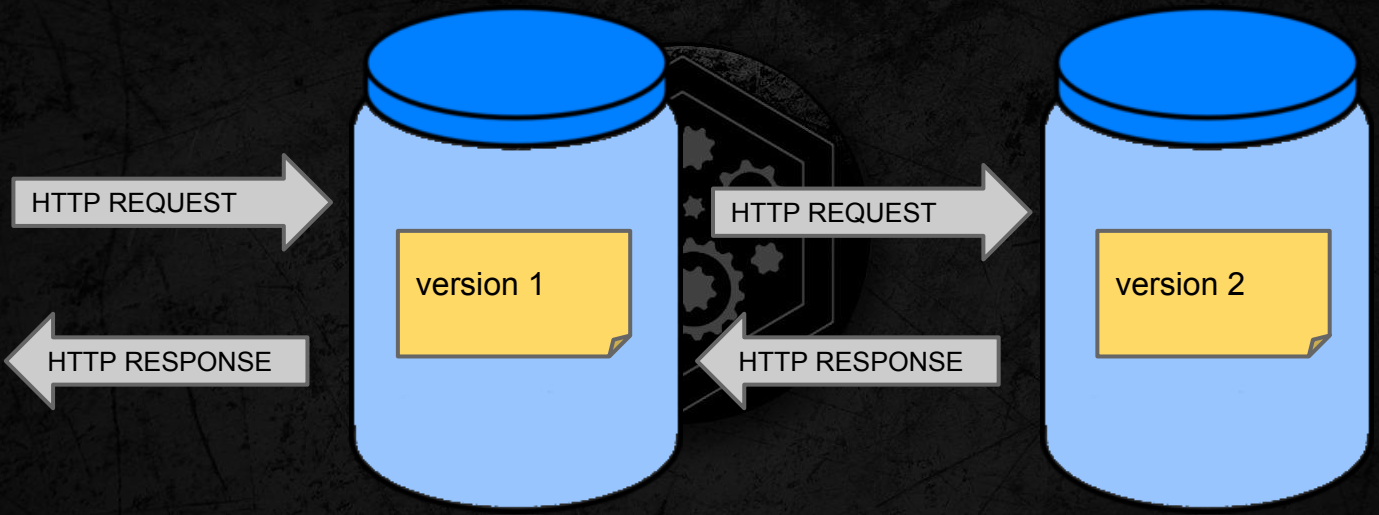
I’d like to ensure that others can talk to my service

Service Discovery

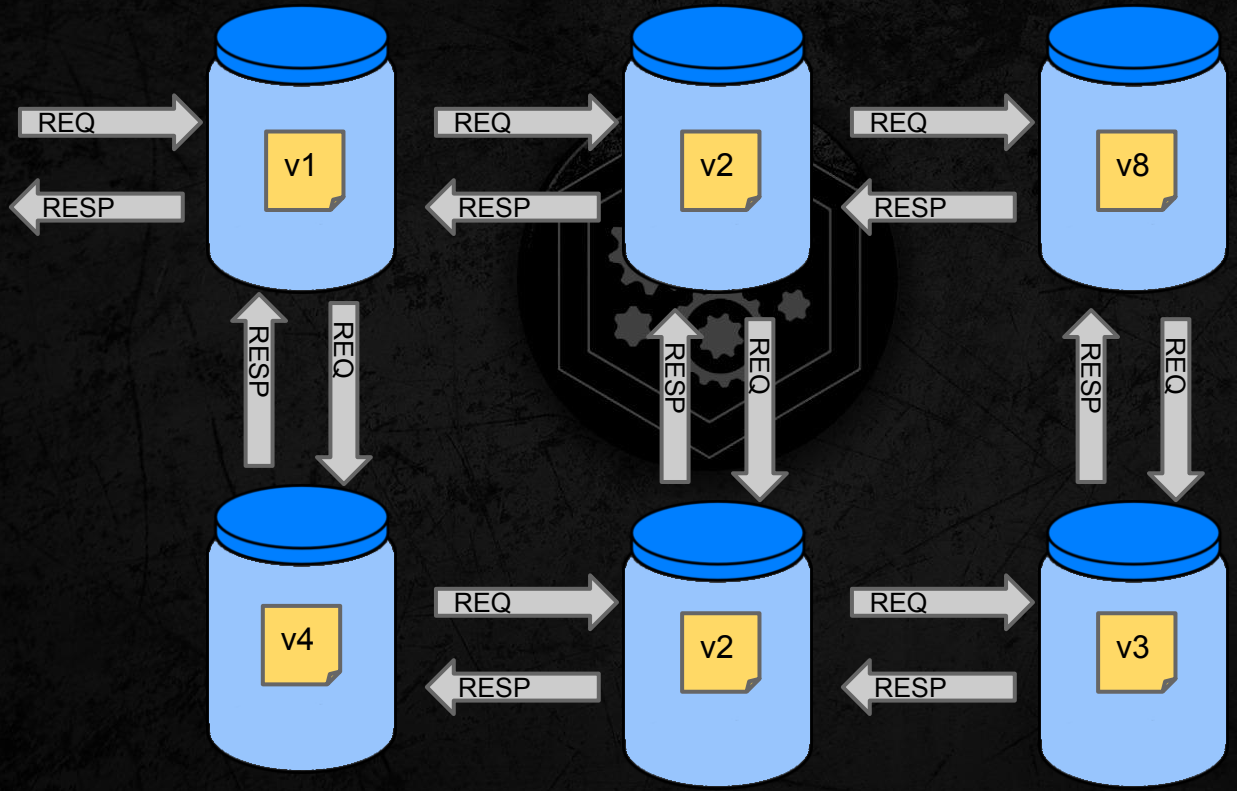
Find your collaborator’s address and port with

- Zookeeper
- Consul
- Eureka
- Etcd
- ...

Consumer Driven Contracts

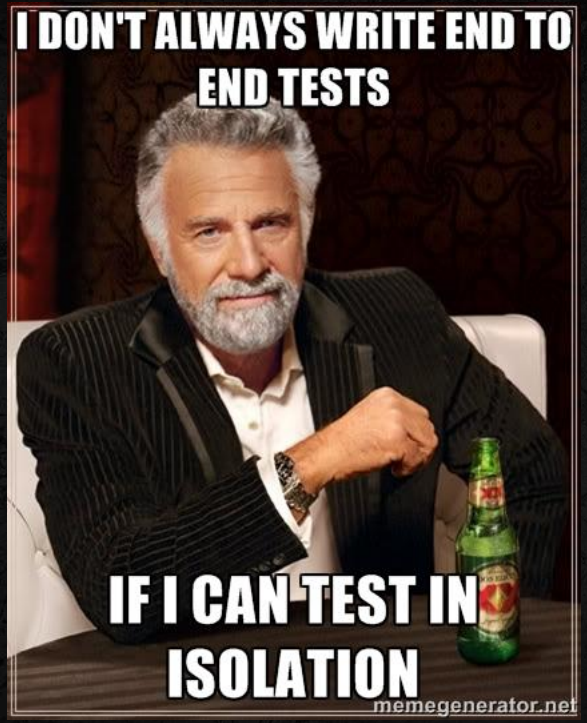


Consumer Driven Contracts

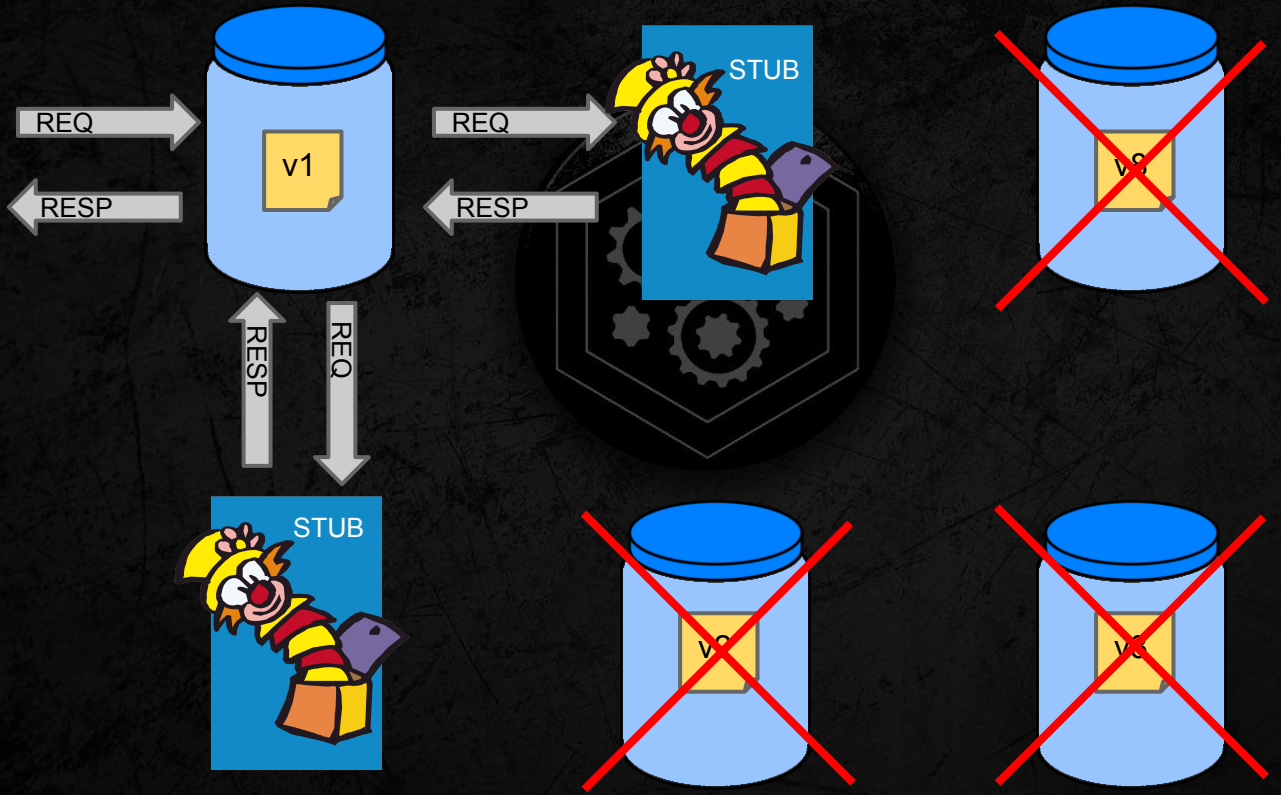


Consumer Driven Contracts

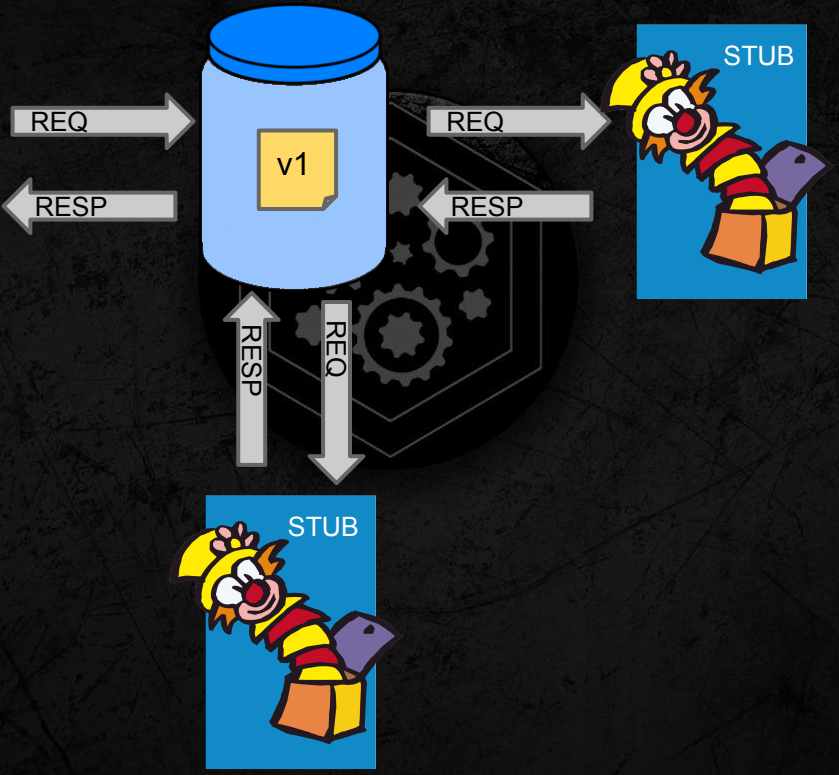
Consumer Driven Contracts



Consumer Driven Contracts

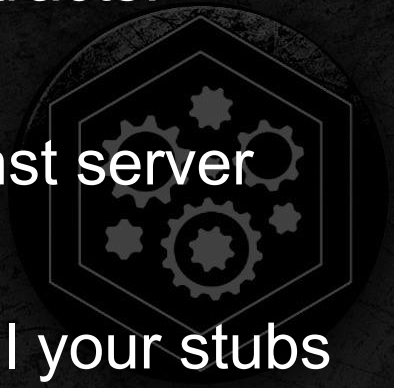


Consumer Driven Contracts



Consumer Driven Contracts

Consumer Driven Contracts:



test your stub against server

your consumers call your stubs

Deploy it

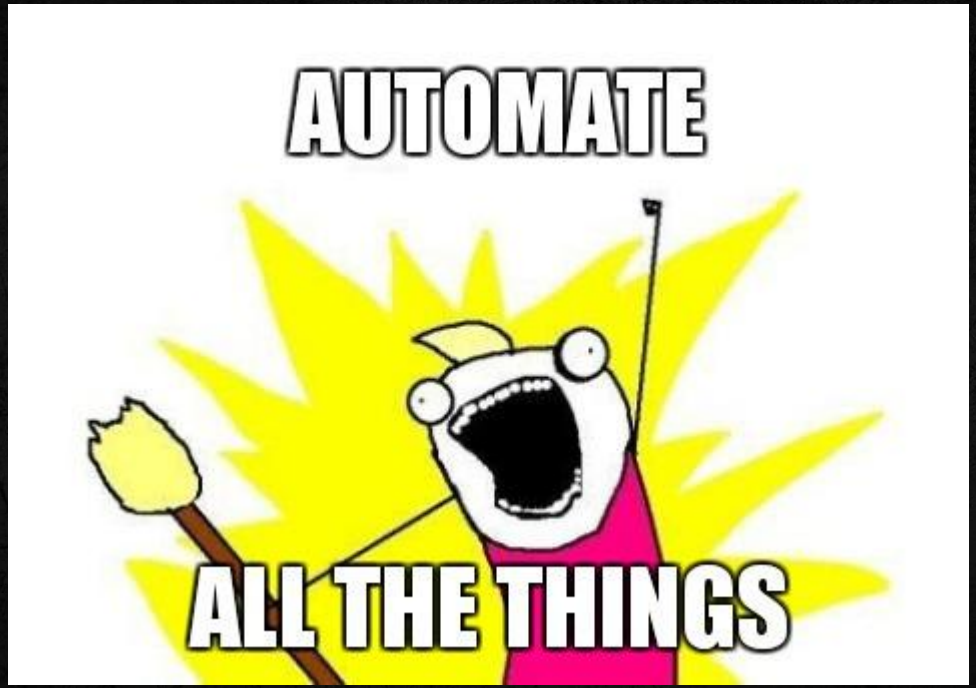
As a developer

I’d like my feature to be on production ASAP

I’d like to have application properties

- in one place
- auditable
- secure

Deploy it



Deploy it

Environment provisioning

- Puppet
- Chef
- Salt
- Ansible
- ...

Deploy it

Application deployment

- Rundeck
- Capistrano
- Fabric
- Ansible
- Freight
- ...

Deploy it

Application configuration

- Version it!
- Encrypt it!

Spring Cloud Config Server

micro-infra-spring-config

Monitor it

As a developer

I don’t want to grep my logs from different servers

I’d like to have application data in one place

- logs
- metrics
- health status

Monitor it

Logs

Unify logging patterns!

Collect logs in one place

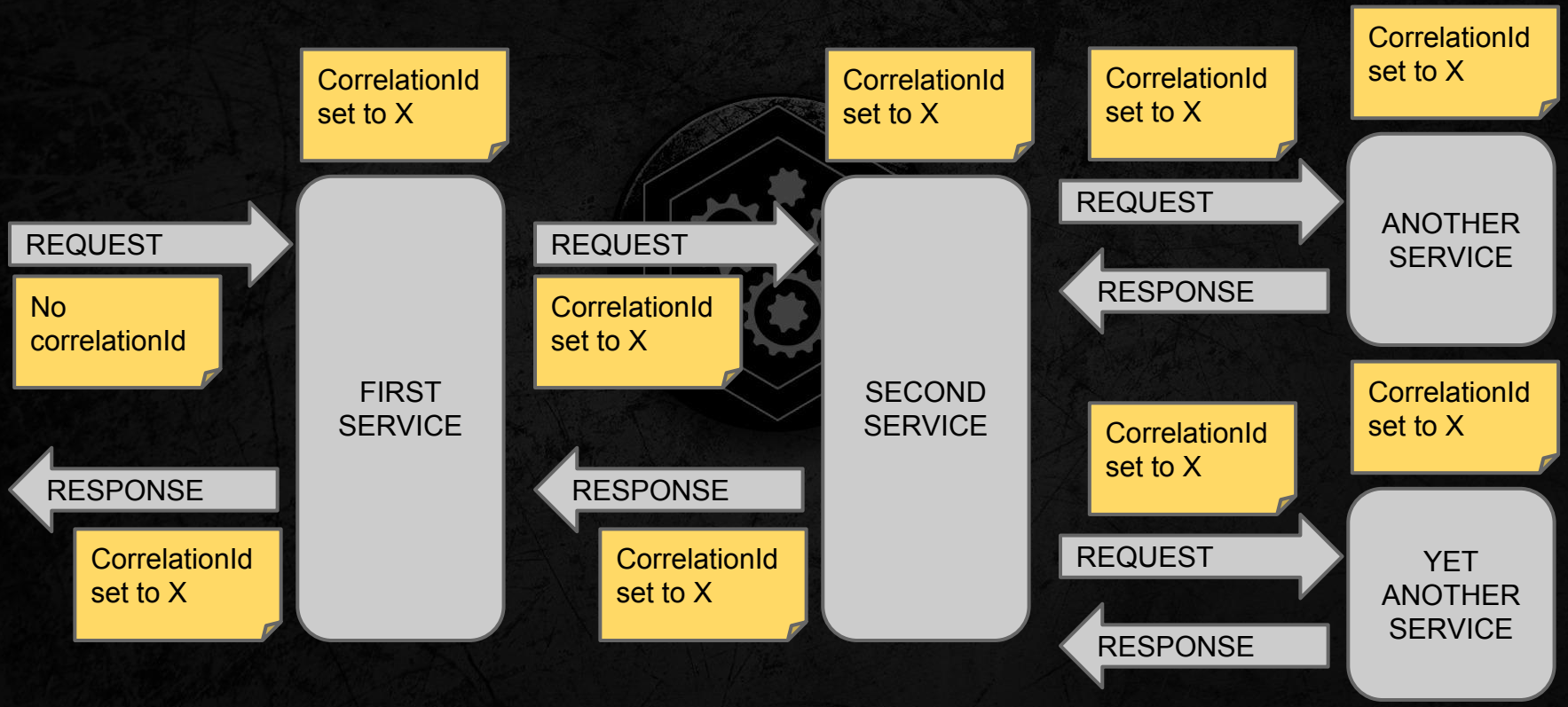
syslog,

ELK stack, graylog2,

Splunk, Loggly

...

CorrelationID



CorrelationID

0 to **7** of 7 available for paging

correlationId	@timestamp	severity	shortmessage	app_name
0009120f-487d-4f37-a542-169730039160	2015-04-15T23:25:41.919+02:00	WARN	Request method 'HEAD' not supported	fat-web-
0009120f-487d-4f37-a542-169730039160	2015-04-15T23:25:41.920+02:00	WARN	Request method 'HEAD' not supported	fat-web-
0009120f-487d-4f37-a542-169730039160	2015-04-15T23:25:56.722+02:00	INFO	Caching product configuration	fat-web-
0009120f-487d-4f37-a542-169730039160	2015-04-15T23:25:56.895+02:00	INFO	Received a request to consume client registration ...	fraud-service
0009120f-487d-4f37-a542-169730039160	2015-04-15T23:25:56.912+02:00	INFO	Found cookie affiliate data AffiliateData{provider...	fat-web-
0009120f-487d-4f37-a542-169730039160	2015-04-15T23:25:56.920+02:00	WARN		fat-web-
0009120f-487d-4f37-a542-169730039160	2015-04-15T23:25:57.146+02:00	INFO	Found cookie affiliate data AffiliateData{provider...	fat-web-

0 to **7** of 7 available for paging

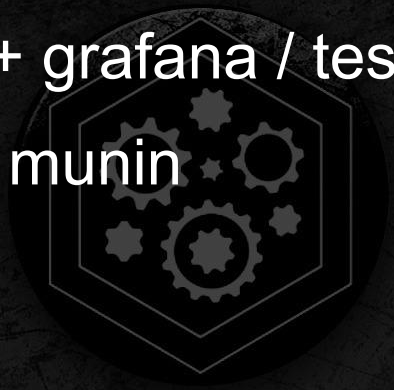
Monitor it

Metrics

graphite + grafana / tessera

collectd / munin

statsd



Monitor it

Alters

nagios / zabbix

cabot

logstash!

Agenda

short intro to microservices

how to deploy your first microservice

microservice pitfalls

Code reuse

do not abstract everything

sometimes copy paste gives you code decoupling

no - copy paste is not a solution to all problems ;)

do not write nanoservices - who will support it?

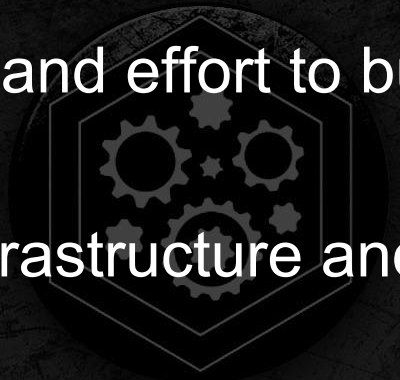
Too many technology stacks

pick a right tool for the job but don’t exaggerate

why would you ever want to code in Brainfuck or Whitespace?

someone will support it afterwards - want to do it? ;)

Management issues



have to invest time and effort to build foundations

have to invest in infrastructure and devops

feature delivery pace will decrease for some time

Questions?

Links

Microservice Hackathon

Accurest - Consumer Driven Contracts implementation