Microservices with Spring

monolith

- default development mode
- easy to run
- easy to deploy
- easy to understand

monolith?

- easily deploy changes?
- modularization?
- **scalability?**

monolith?

"routines are constructed on rational principles so that families fit together as building blocks. [you can] safely regard components as black boxes"
-- M. D. McIlroy (1968)

component

- around business functionality
- easily replaceable
- easily maintainable
- explicit API


SOA?


unix utilities?

microservices!

microservice on slide

- low overhead
- replaceable
- maintainable
- explicit API
- designed for failure
- maintained by a single team
- …

real-life Spring microservice

"small executable Spring-Boot based JAR with REST API"

challenges

versioning | repositories | deployment | continuous integration | integration testing | configuration management | monitoring | expensive cross-component communication | failed components | lots of RAM | API versioning | centralized logging | responsibility fights

DEMO

running

640 GB RAM ought to be enough for everybody

testing

integration tests

integration tests

integration tests

communication

cross-component communication

1) is expensive
2) hard to test
3) prone to failure

“dumb pipes, smart endpoints”

HTTP | lightweight messaging

#rabbitmq #kafka

“dumb pipes, smart endpoints”

use transparent payloads

#json

epic fial

your components **will** fail

monitoring

establish monitoring very early

centralized logging

log cross-component communication

#graylog #logstash

responsibility fights

which service will contain
the functionality?

configuration

large-scale deployment
requires centralized configuration

#consul #heureka

deployment

continuous integration is a must

#jenkins #travis

large scale-deployment

one-line deployment

#docker

#ansible

shared libraries

treat it like a binary dependency

use proper repos

#nexus

versioning

establish a versioning scheme

#graylog #logstash

experiences

- expect high mental step to "micro"
- hellgate of tools opens
- juggle lots of challenges at the same time

positives

- enforces best practices
- allows scaling
- rewrite over maintain wins in long-term
- makes you devops
- makes you polyglot

positives

- enforces best practices
- allows scaling
- rewrite over maintain wins in long-term
- makes you devops
- makes you polyglot
- monolith looks suddenly easy

@RoboNovotny