Spring Cloud Deep(er) Dive

Joris Kuipers, Hands-on architect
Trifork Amsterdam
@jkuipers

What is Spring Cloud?

Set of projects helping to build *distributed systems*

Cloud, Microservices, Self-contained Systems, etc.

Agenda
(for people that didn't see Josh's talk)

From a service's Point of View:

1. Who am I? (this time)
2. Where are the others?
3. How do I talk to them?

Agenda
(for people that *did* see Josh's talk)

What the ***
just happened?

Who Am I (this time)?

Centralized Configuration

Centralized Configuration: **What?**

► Config for instances of *same* service
► Config shared with *other* services

Centralized Configuration: **Why?**

▶ Different servers

▶ Different environments

▶ Push updates

Centralized Configuration: How?

Spring Cloud Config Server

HASHICORP™

CONSUL

Config Demo

Where Are The Others?

Service Registries

Service Registries: why?

"No Service Is An Island"

▶ Advertise own presence

▶ Locate other services / middleware

▶ Might want to choose *instance* itself

Service Registries: why?

Ops:

▶ What services are there?

▶ How to check their health?

Service Registries: How?

Service Registry Demo

How Do I Talk To Them?

Robust inter-service communication

Inter-service communication

Sync
vs
Async

Synchronous communication

- ▶ Queries, commands w/ confirmations
- ▶ Concerns:
  - ▶ blocking / non-blocking
  - ▶ load balancing
  - ▶ 'robust' calls

Synchronous Inter-service Calls

Hystrix: Circuit Breakers

Hystrix: Circuit Breakers

als antwoord op @rbrtwnklr

Ben Christensen @benjc... 🌐 10-06-15

@rbrtwnklr @crichardson @_Javaslang note that circuit breakers are highly overrated. We could drop that from Hystrix and hardly miss it.

⤺ ⇄ ★ 2 👤+

Hystrix: Bulk Heading

als antwoord op @rbrtwnklr
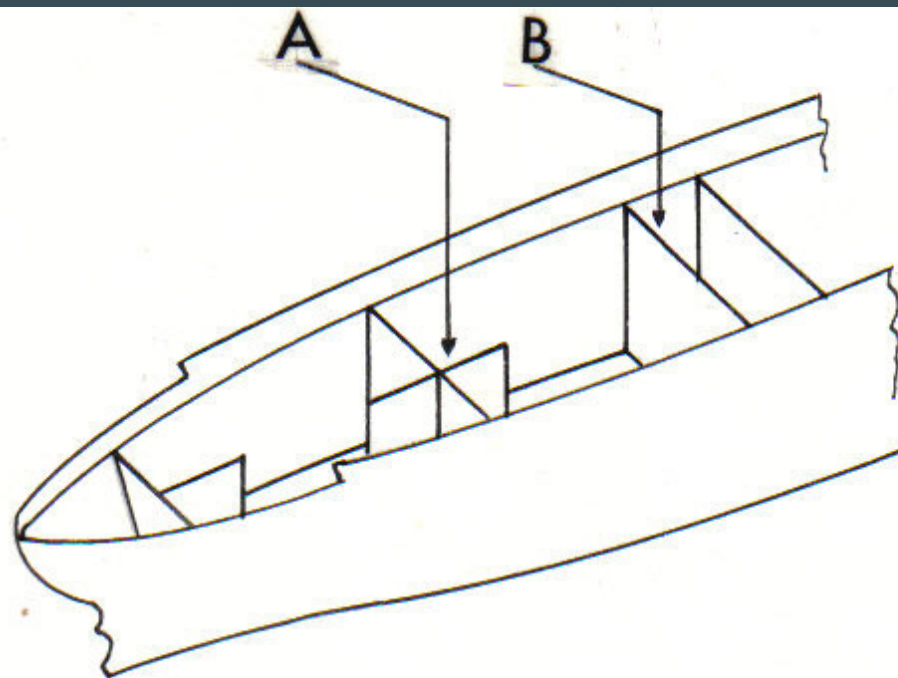
Ben Christensen @benjc... 🌐 10-06-15
@rbrtwnklr @crichardson @_Javaslang the most important part of Hystrix is the concurrency limits, the "bulkheading". That protects systems.

★ 2

Hystrix: Bulk Heading

A   B

BULKHEAD

169A

which is a dividing partition B = Watertight cross-wise type

Async communication: *messaging*

▶ Events, triggers, request/response, ...

▶ Various brokers / protocols / APIs

▶ Common patterns

Messaging patterns

▶ Point-2-point vs. **Pub-sub**

▶ Competing consumers

▶ Partitioning

▶ Durable subscriptions

Async Communication Demo

Message sending & receiving through Spring Cloud Stream

Other questions

▶ What makes things so slow?

▶ How to propagate e.g. config changes?

▶ How to propagate authentication info?

▶ What about "significant others"?

Other questions

▶ ## What makes things so slow?

  ▶ Distributed tracing with **Spring Cloud Sleuth**

▶ ## How to propagate e.g. config changes?

  ▶ **Spring Cloud Bus**

▶ ## How to propagate authentication info?

  ▶ **Spring Cloud Security** with OAuth2

▶ ## What about "significant others"?

  ▶ **Spring Cloud Cluster** for coordination