



# Microservices pitfalls

Addressing the most frequent pitfalls when  
transitioning to Microservices

# Lothar Schulz



**Head of Engineering**

[lotharschulz.info](http://lotharschulz.info)

[lnkd.in/lotharschulz](https://www.linkedin.com/company/lotharschulz)

# Magnus Kulke



**Software Engineer**

[github.com/mkulke](https://github.com/mkulke)

[lnkd.in/magnuskulke](https://www.linkedin.com/company/magnuskulke)

- **Micro vs. Macro**
- **Domains**
- **Contracts**
- **API Flavors**
- **Distributed Systems**
- **Observability**





**Micro or Macro**

---

**Empty vessels make  
the most noise**

# Micro Services

loosely coupled service oriented architecture with bounded contexts

microservice architectural style is an approach to developing **a single application as a suite of small services**, each running in its own process and communicating with lightweight mechanisms, often an HTTP resource API. These services are built around business capabilities and independently deployable by fully automated deployment machinery. There is a bare minimum of centralized management of these services, which may be written in different programming languages and use different data storage technologies.

[Adrian Cockcroft: State of the Art in Microservices](#)  
[James Lewis and Martin Fowler: Microservices Guide](#)

# Micro Services

Modularisation - Modular programming

focus **independent, interchangeable modules**

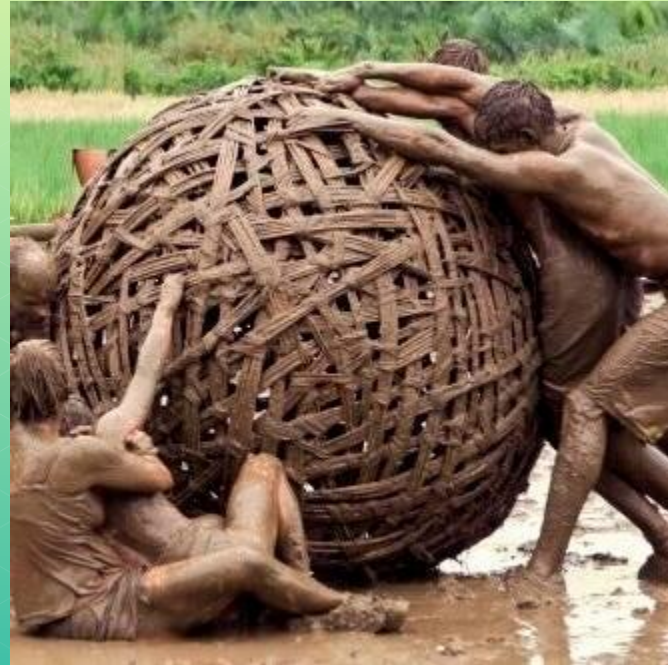
## Decomposition

- Interfaces
- Contracts
- Packages
- Data

# Big Ball of Mud

## Forces

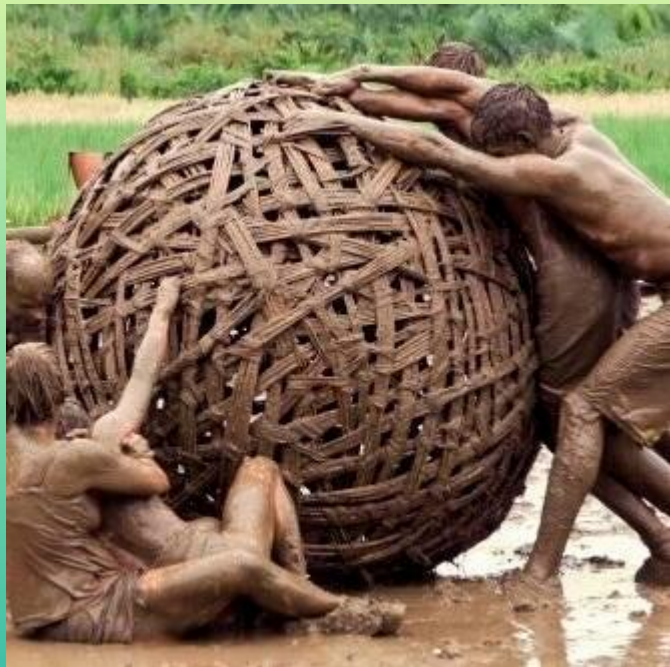
- Time
- Cost
- Experience
- Skill
- Complexity
- Change
- Scale



<http://www.laputan.org/mud/>



# Big Ball of Mud



## Throw away code

Debug logs in Production Code

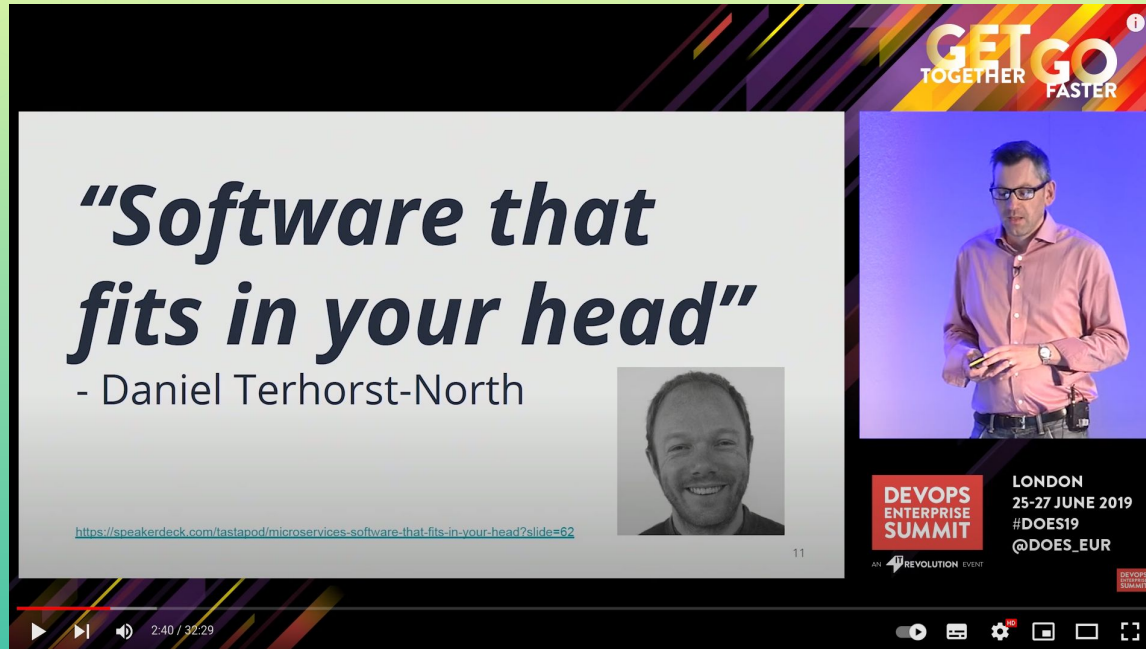
## HotFix Git Branch Selection/Cherry Picking

Fixes NOT deployed

<http://www.laputan.org/mud/>



# How small is *micro* ?



The screenshot shows a video player interface. The main content is a presentation slide with the title **"Software that fits in your head"** in a large, bold, black serif font. Below the title is the name **- Daniel Terhorst-North** in a smaller, black sans-serif font. To the right of the text is a small, square, black and white portrait of a man with a beard and glasses, smiling. Below the portrait is a URL: <https://speakerdeck.com/tastapod/microservices-software-that-fits-in-your-head?slide=62>. The slide number **11** is in the bottom right corner. The video player has a progress bar at the bottom showing **2:40 / 32:29**. On the right side of the video player, there is a vertical banner for the **GET GO TOGETHER FASTER** event. Below this banner is a photo of a man in a pink shirt and glasses, holding a small object. Further down is the **DEVOPS ENTERPRISE SUMMIT** logo, which includes the text **AN ACCELERATION EVENT**. To the right of the logo, it says **LONDON 25-27 JUNE 2019 #DOES19 @DOES\_EUR**. At the bottom right of the banner is a small **DEVOPS SUMMIT** logo.

Monoliths vs Microservices is Missing the Point–Start with Team Cognitive Load - Team Topologies  
<https://speakerdeck.com/tastapod/microservices-software-that-fits-in-your-head?slide=62>

Addressing the most frequent pitfalls when transitioning to Microservices - 2023 04 13 - Lothar Schulz



# **Domains**

---

**None of your concern!**  
**Slicing microservices**  
**properly**

# Database as Microservice



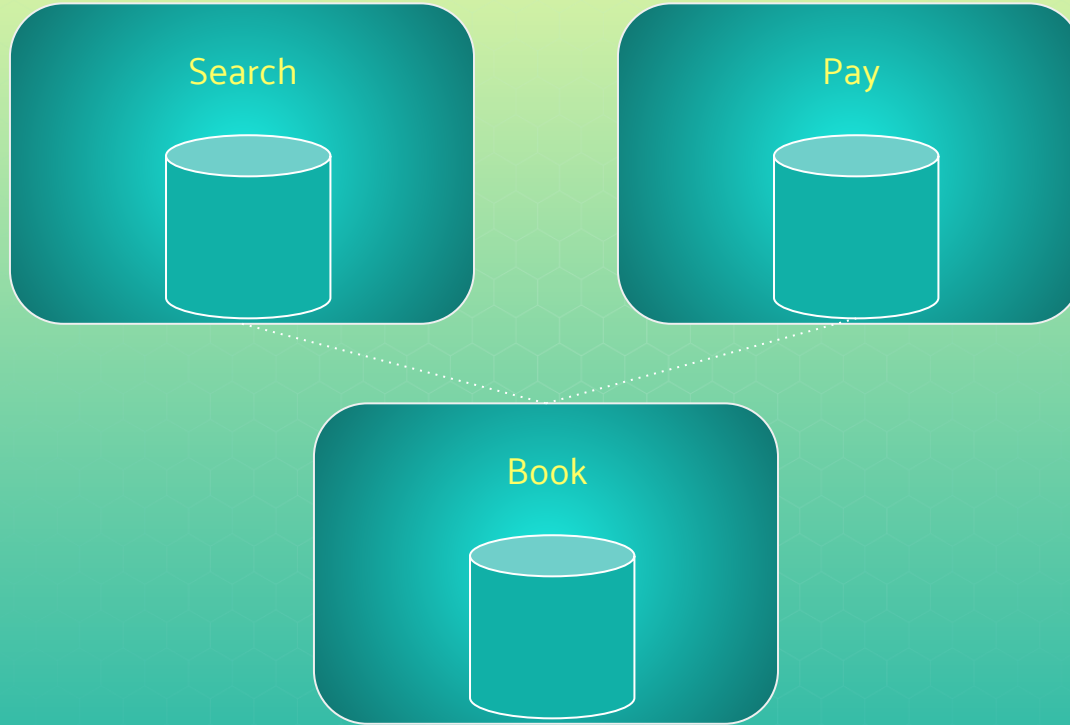
# Database as Microservice



# Monolith first



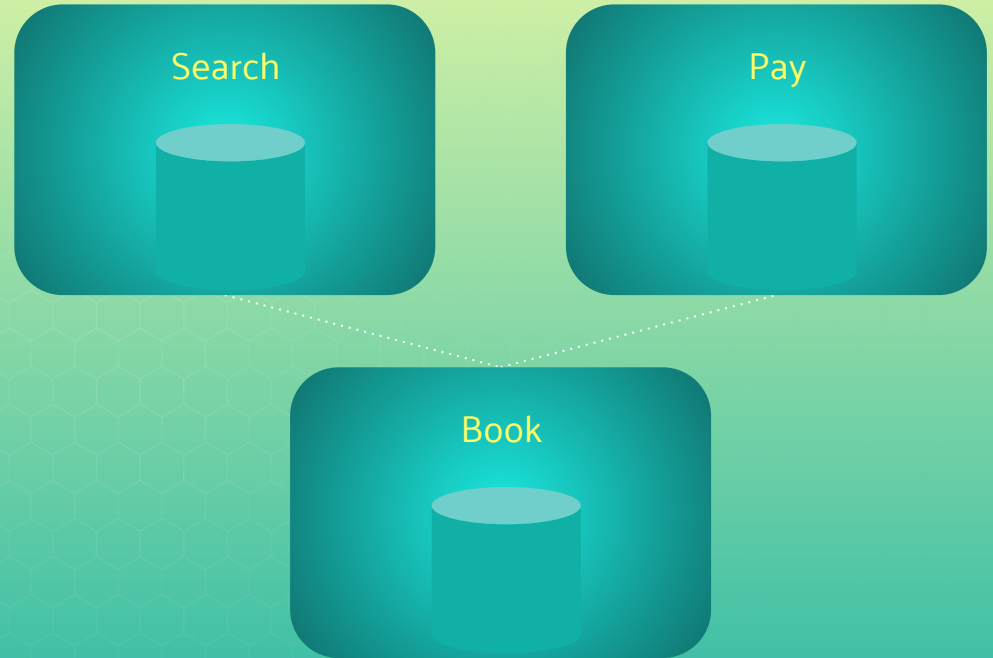
# Domains





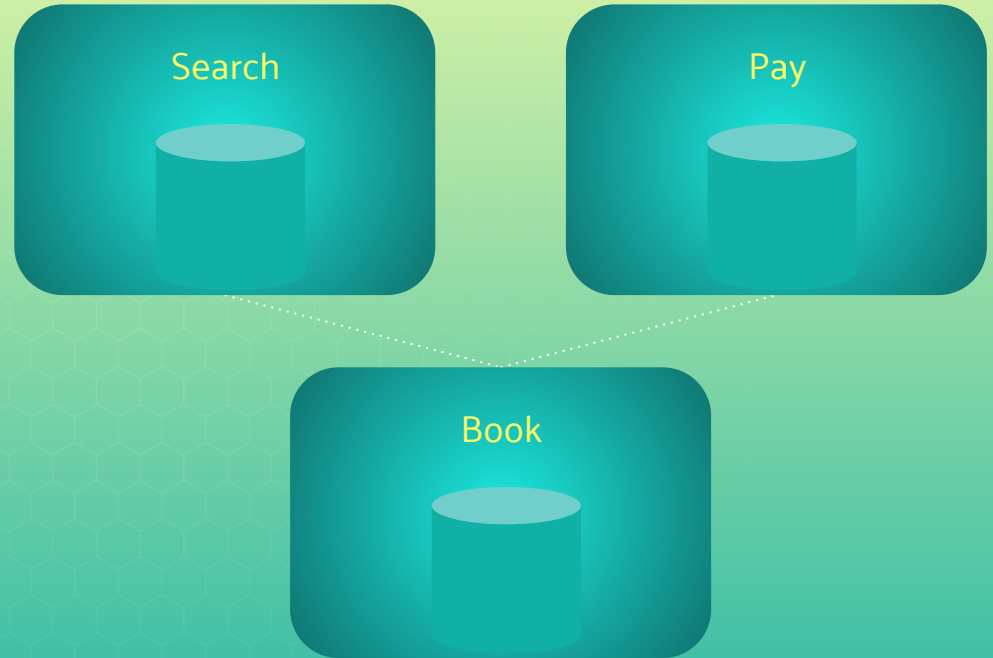
# Domains

## Scaling



# Domains

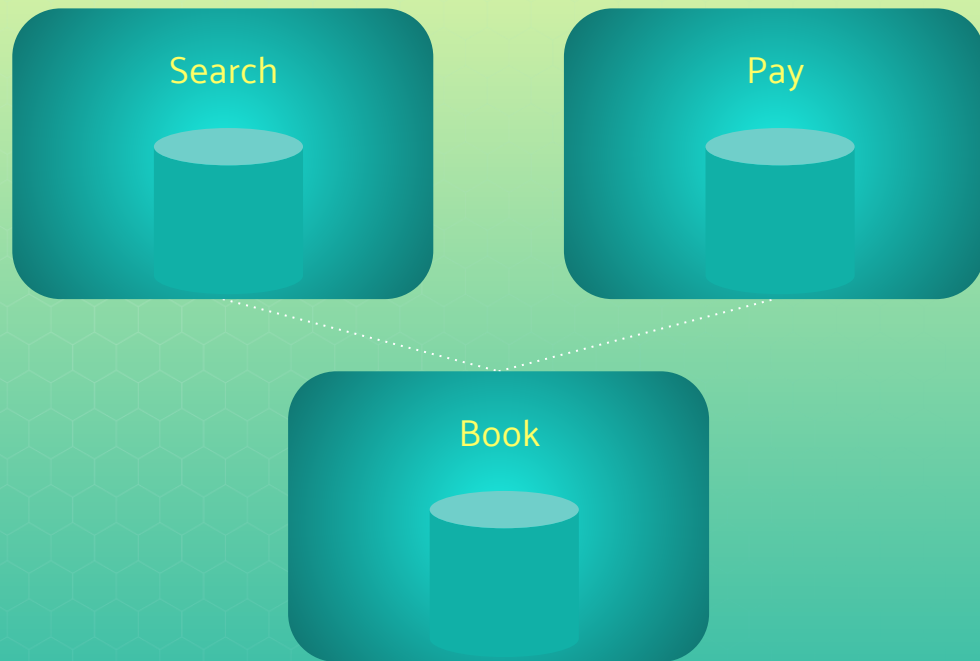
Scaling  
- Vertical



# Domains

## Scaling

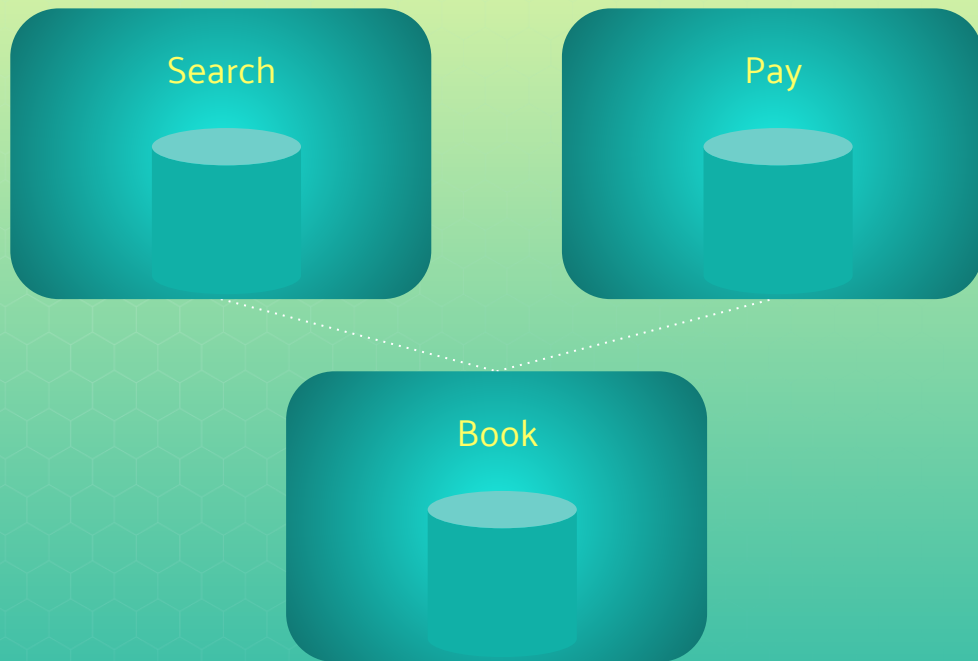
- Vertical
- Horizontal



# Domains

## Scaling

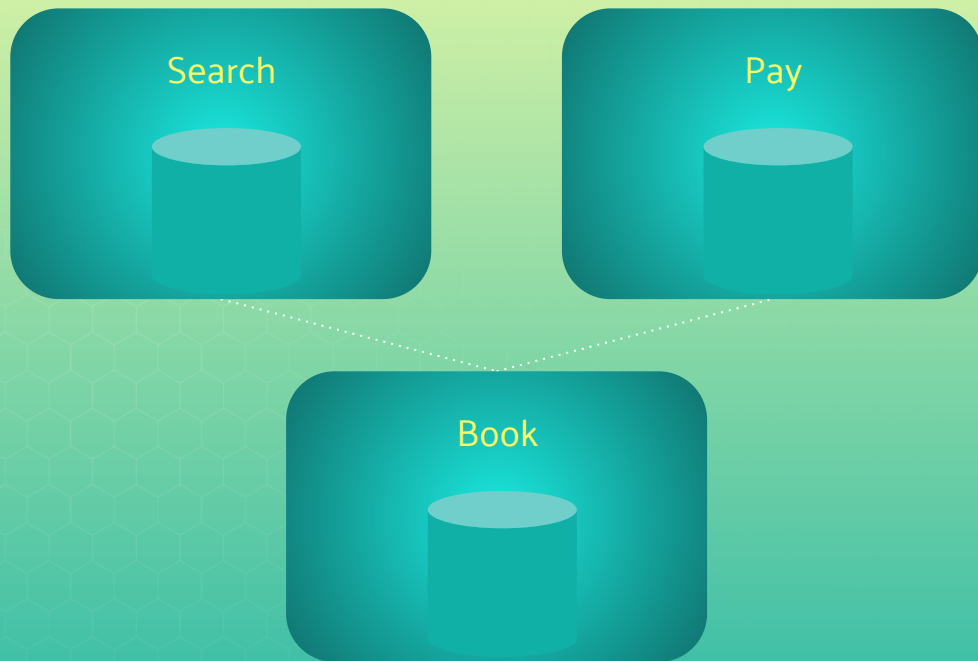
- Vertical
- Horizontal
- Partitioning



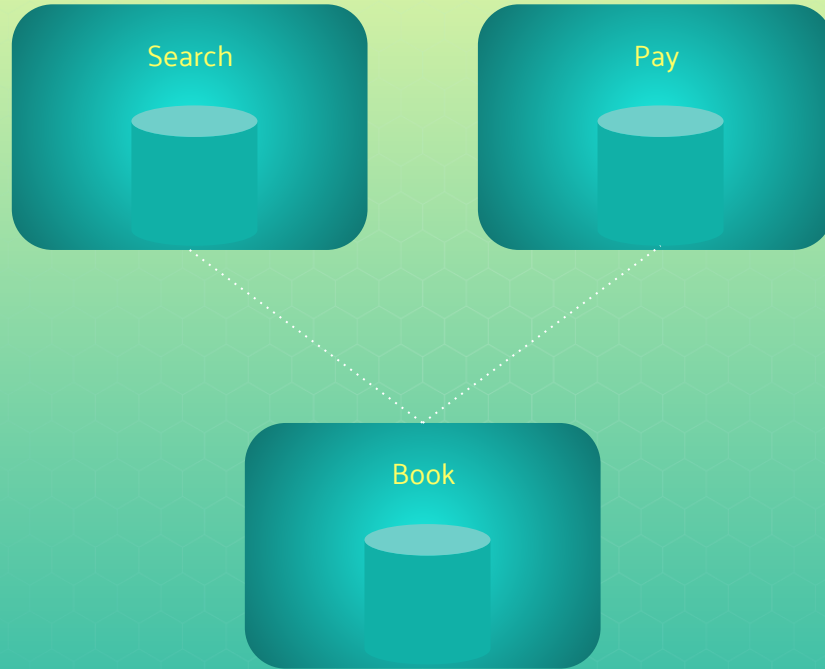
# Domains

## Scaling

- Vertical
- Horizontal
- Partitioning
  - Sharding

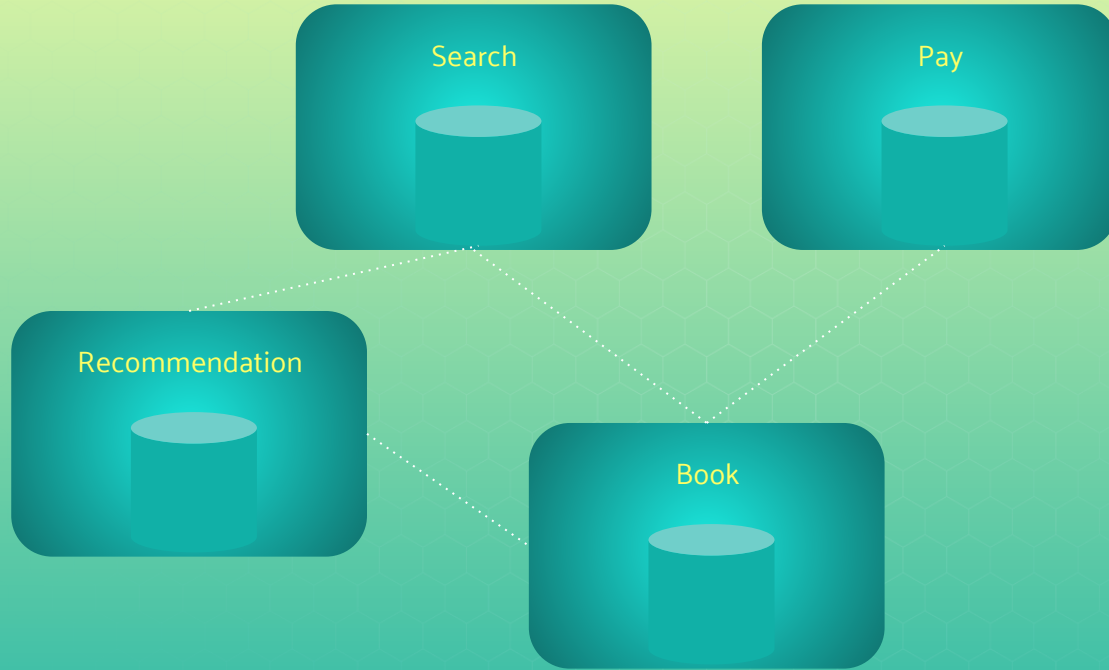


# Domains – Bounded Contexts

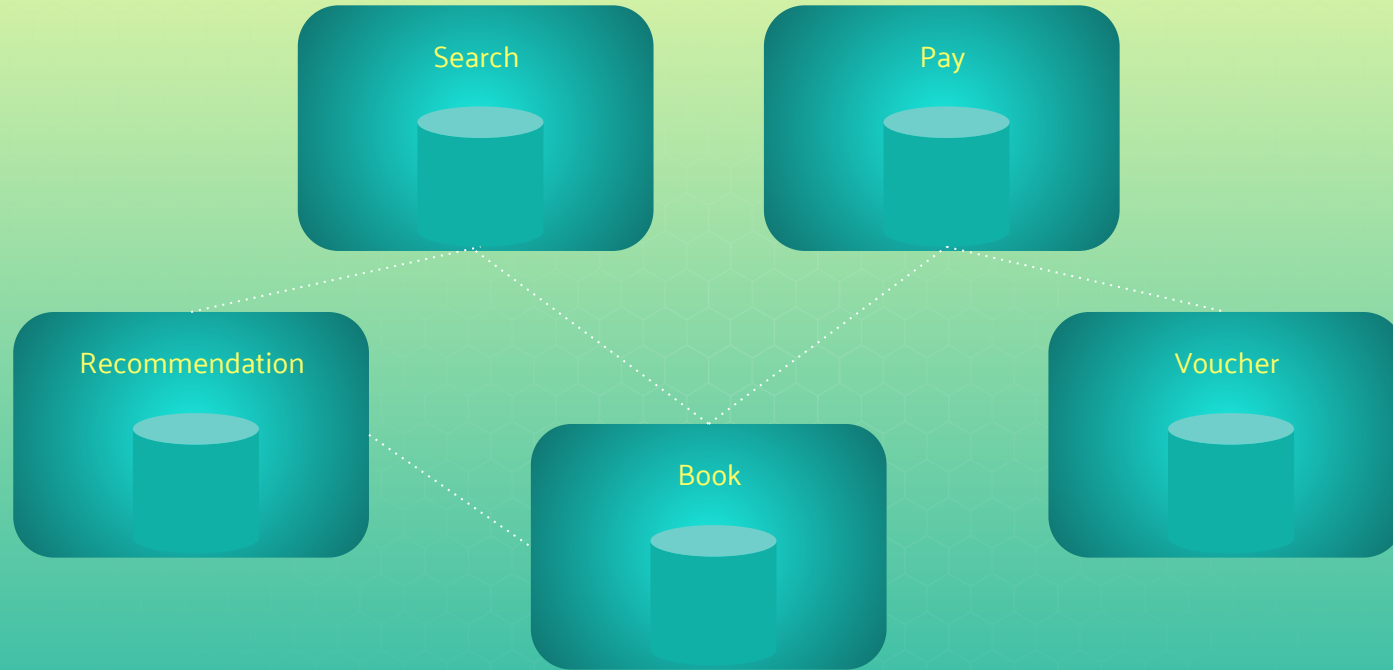




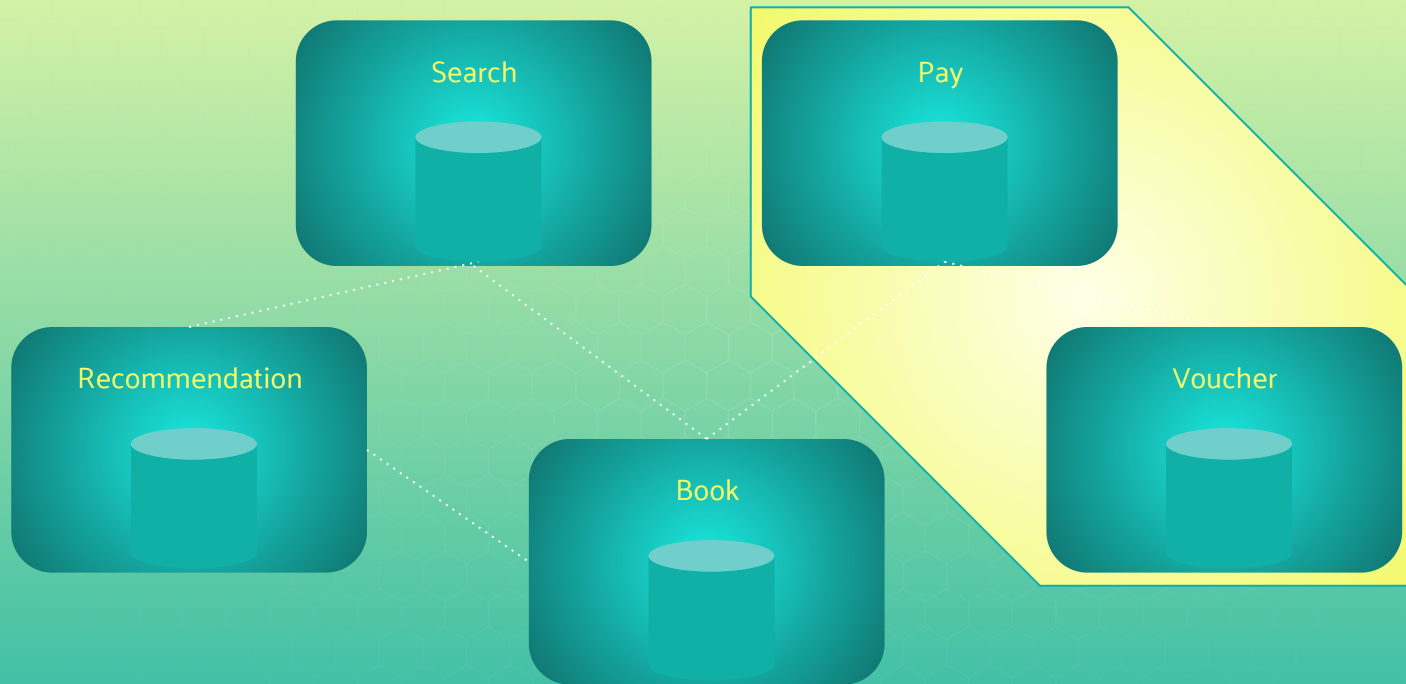
# Domains – Bounded Contexts



# Domains – Bounded Contexts



# Domains – Bounded Contexts





**Contracts**

---

**Lawyer up!**

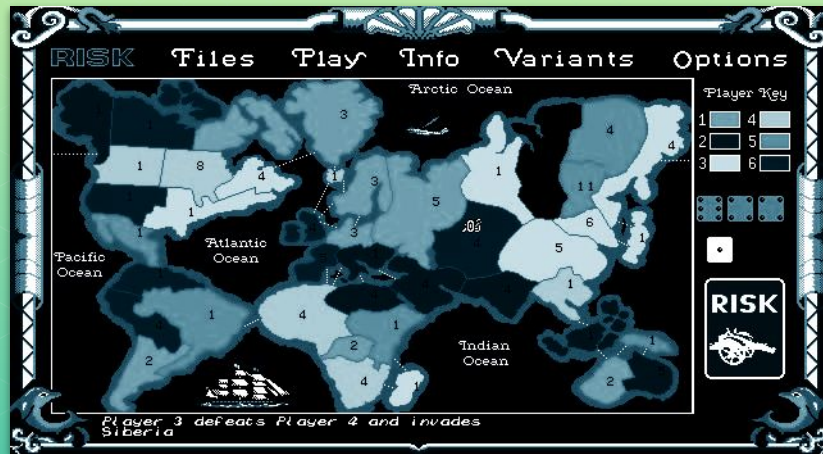
**Ambiguities and Unmet  
Expectations**

# Microservices are (also/primarily?) a social tool

- There is a relation between architecture and team setup
- **“Any organization that designs a system (defined broadly) will produce a design whose structure is a copy of the organization’s communication structure.”**

Conway’s Law

- Enables teams to make autonomous decisions



# Service Boundaries are Defined by Contracts

- Codify expectations towards an API from the consumer's perspective
  - Behaviour: does not change unexpectedly
  - Availability: when can we retire an API?
- How to express such a contract?
  - Machine readable: Swagger/OpenAPI, JSON Schema, GraphQL
  - API Versions
- Abstain from breaking changes
  - Additional properties?
  - Extending enums?
- Make everything optional: Protobuf3



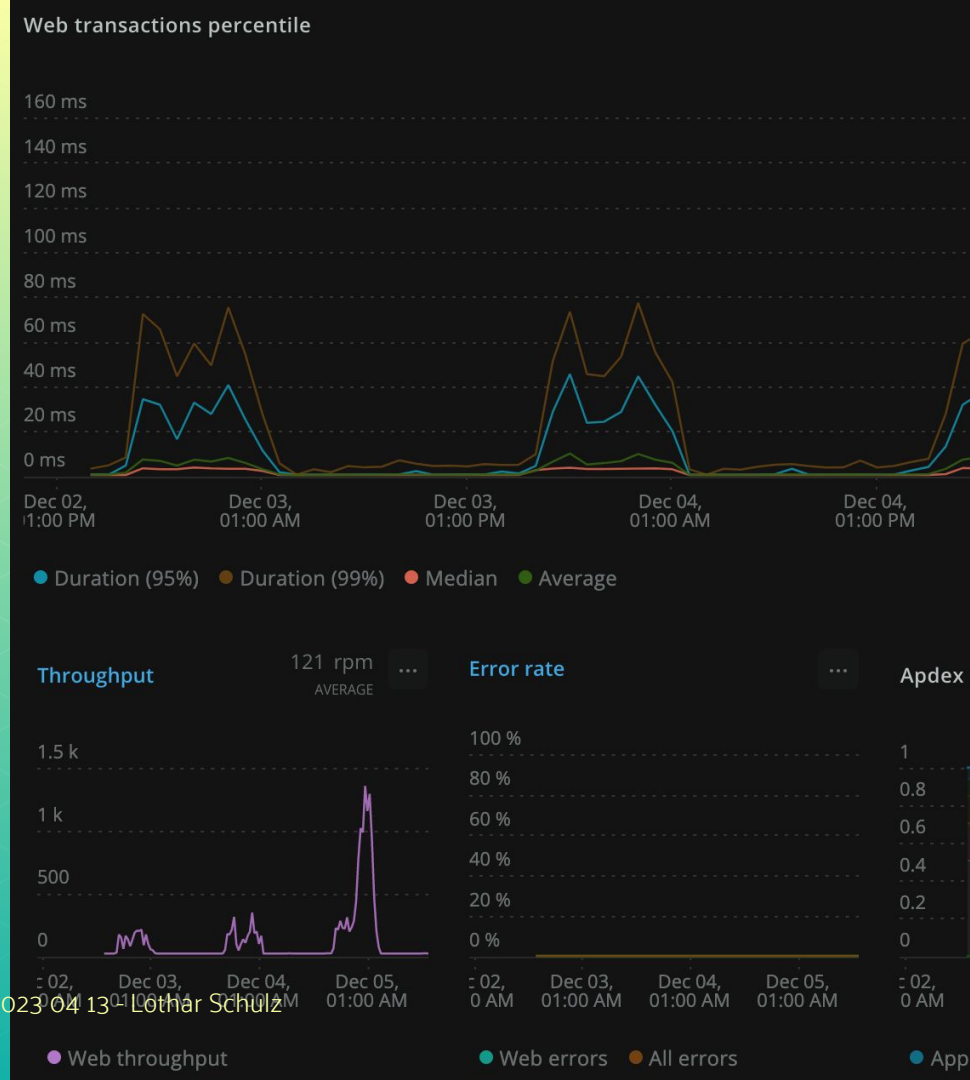
# Problem: A Schema might not be expressive enough

- Documents can be formally correct
- But semantics have changed
  - References in a document
  - Content: New ID for entity
- Pragmatic solution: Contract tests

```
{
  "fares": [
    {
      "id": "1",
      "name": "single ticket",
      "value": 1.7,
      "currency": "USD"
    }
  ],
  "trip": {
    "from": "Canal St",
    "to": "Union Square Subway",
    "fareId": "1"
  }
}
```

# Performance Characteristics

- Service level objectives
- Rate limits
- Request budgets



# The Other Side: Protection from Harmful Workloads

- Unforeseen (ab)use patterns
- How to attribute incoming traffic?
  - Correlation Ids
  - Callers need to tag their requests
- Manage access
  - Service Accounts
  - Declarative: Service Mesh
  - Rate Limiting

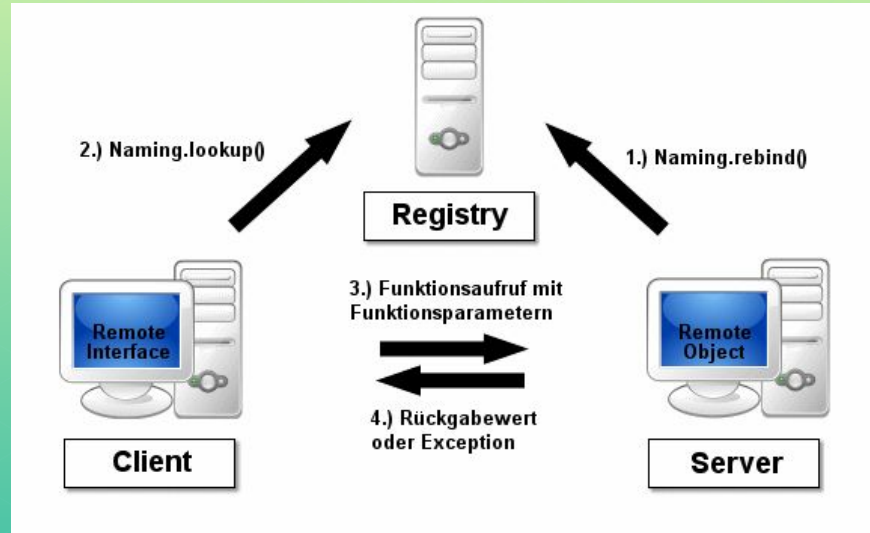


# API flavors

---

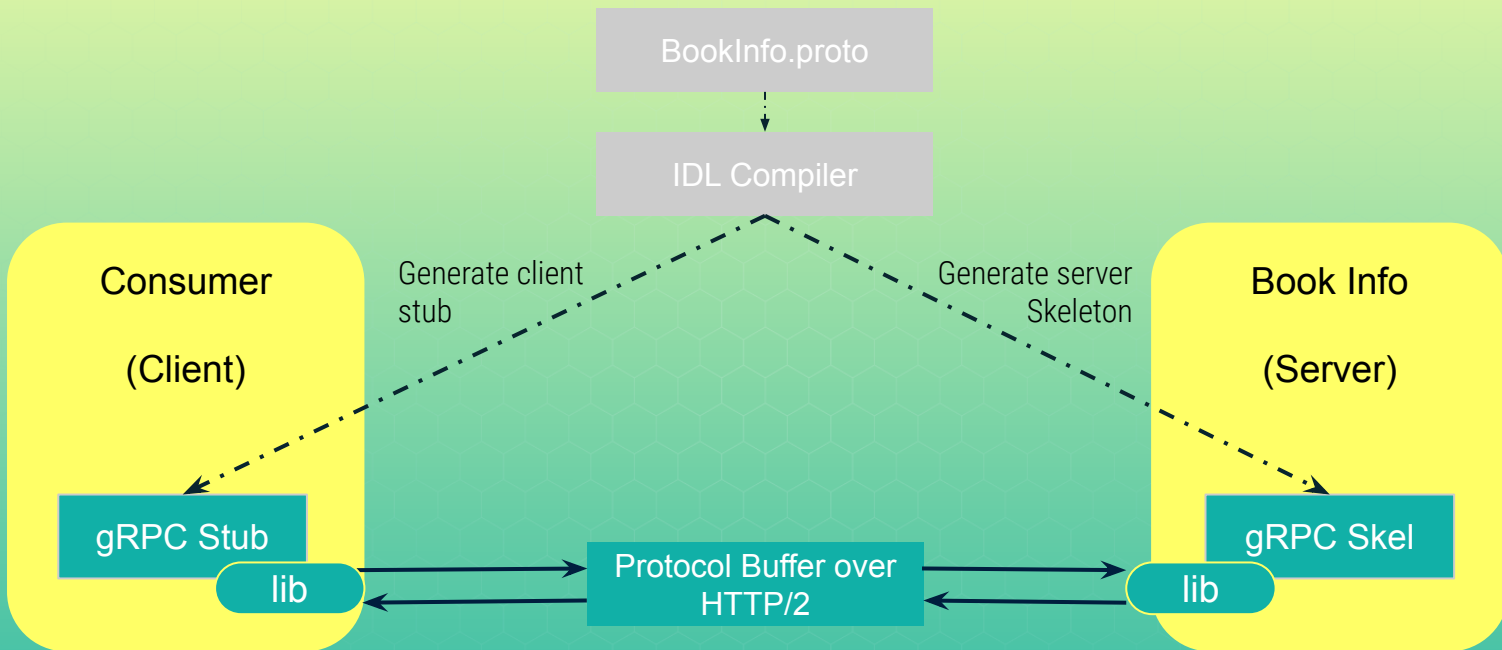
**Which style would you  
prefer?**

# Remote Procedure Calls



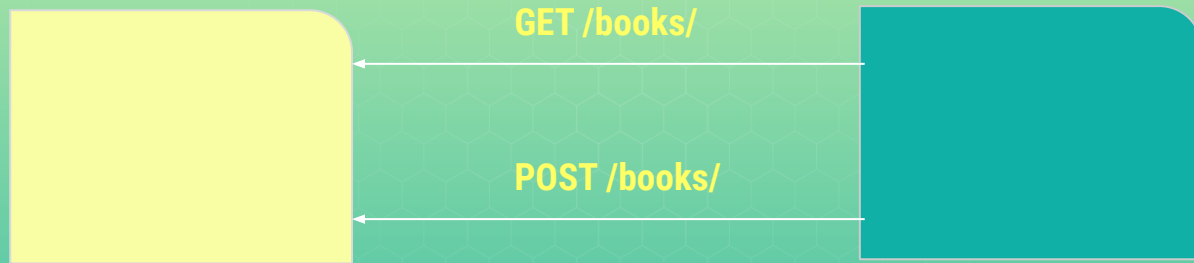
[https://de.wikipedia.org/wiki/Remote\\_Method\\_Invocation](https://de.wikipedia.org/wiki/Remote_Method_Invocation)

# Remote Procedure Calls – GRPC





# HTTP APIs

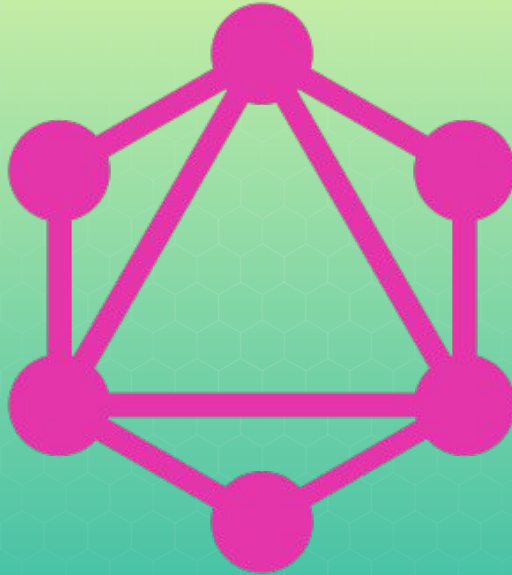


# REST Maturity Models

- Level 0 - remote interactions
- Level 1 - Resources
  - /books/123
  - /books/123/author
- Level 2 - HTTP Verbs
  - GET, POST, PUT, HEAD, PATCH, DELETE
- Level 4 - Hypermedia Controls
  - <link rel="slots/book" uri="/slots/123"

<https://martinfowler.com/articles/richardsonMaturityModel.html>

# GraphQL



# GraphQL

```
orderBy: {field: $createdat, direction: $order}
```

This order direction broke the pagination. Although there are more repositories on GitHub for the specific organization I tested that with, the repositories are not listed in the result payload.

## The Fix

Remove the order directive to workaround the pagination issue:

```
orderBy: {field: $createdat, direction: $order}
```

Now the query looks like this:

<https://www.lotharschulz.info/2020/03/18/how-to-break-github-graphql-pagination/>  
<https://www.lotharschulz.info/2021/02/13/github-graphql-pagination-orderby-issue-fixed/>



# **Distributed Systems**

---

**Your Consensus is a  
House of Cards**

# Scenario I: DockerHub

- Rate limits
  - Urgent rollback, 3am
  - Node cannot pull *redis:latest* 🐱
- DNS Load Balancing
- DNS transport is UDP
- UDP Packages are limited in size
- Per Spec DNS allows  $\leq 512$  bytes

```
; <<>> DiG 9.10.6 <<>> @1.1.1.1 registry.hub.docker.com
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 59050
;; flags: qr rd ra; QUERY: 1, ANSWER: 5, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
;; QUESTION SECTION:
;registry.hub.docker.com.      IN      A

;; ANSWER SECTION:
registry.hub.docker.com. 276 IN      CNAME   elb-hub.us-east-1.aws.dcr.io.
elb-hub.us-east-1.aws.dcr.io. 876 IN    CNAME   us-east-1-elbhub-1t5fblb53f6sl-411513349.us-east-1.elb.amazonaws.com.
us-east-1-elbhub-1t5fblb53f6sl-411513349.us-east-1.elb.amazonaws.com. 36
us-east-1-elbhub-1t5fblb53f6sl-411513349.us-east-1.elb.amazonaws.com. 36
us-east-1-elbhub-1t5fblb53f6sl-411513349.us-east-1.elb.amazonaws.com. 36

;; Query time: 21 msec
;; SERVER: 1.1.1.1#53(1.1.1.1)
;; WHEN: Mon Dec 07 10:40:29 CET 2020
;; MSG SIZE rcvd: 222
```

# Scenario I: DockerHub, cont.

- DNS responses > 512 bytes fall back to TCP
  - Your sysadmin might not know this
  - Security Group blocks tcp/53
- Not all resolvers are alike / agree on the spec
  - Glibc “salvages” truncated DNS messages
  - Golang DNS resolver (Docker) does not
  - Quick fix: `CGO_ENABLED=1`

## Scenario 2: DNS, again (it's always DNS)

- Our J2EE service is stuck in an exception loop
  - Logs a lot of large stack traces (lots of lines)
- Engineers integrate cool .io SaaS for tailing logs in Logstash
  - Every line a request to cool .io data sink
  - Every line a hostname is resolved
- Cloud Providers disapproves, starts rate-limiting DNS for the service's node
- K8S api-server/node comm. is affected.
  - Node is marked as broken
  - Scheduler moved ever-crashing service to fresh, healthy node
- Repeat



## Scenario 3: Seemingly unlimited resources



## Scenario 3: Seemingly unlimited resources, cont.

### ISPs & Facebook Outage, Oct 2021

- Facebook is removed from DNS servers worldwide
- Gazillions of IoT devices, SmartTVs, etc are deployed and want to talk to facebook's servers
- Resolving fb fails, sloppy programs retry host resolution in a hot loop (no exponential backoff, no jitter)
- ISPs DNS get overwhelmed by requests
- Internet is 💥 not just fb





# **Observability**

---

## **How to X-Ray a hairball**



A close-up, blurred image of a computer screen displaying lines of JavaScript code. The code is colorful and appears to be part of a web application, possibly related to a carousel or slider, as indicated by the visible text like 'carousel', 'slide', and 'interval'. The image is heavily blurred, showing horizontal streaks of text across the frame. The background is dark, and the overall aesthetic is that of a digital or tech-themed background.

## WELCOME TO THE ISILON CLUSTER SUMMARY DASHBOARD

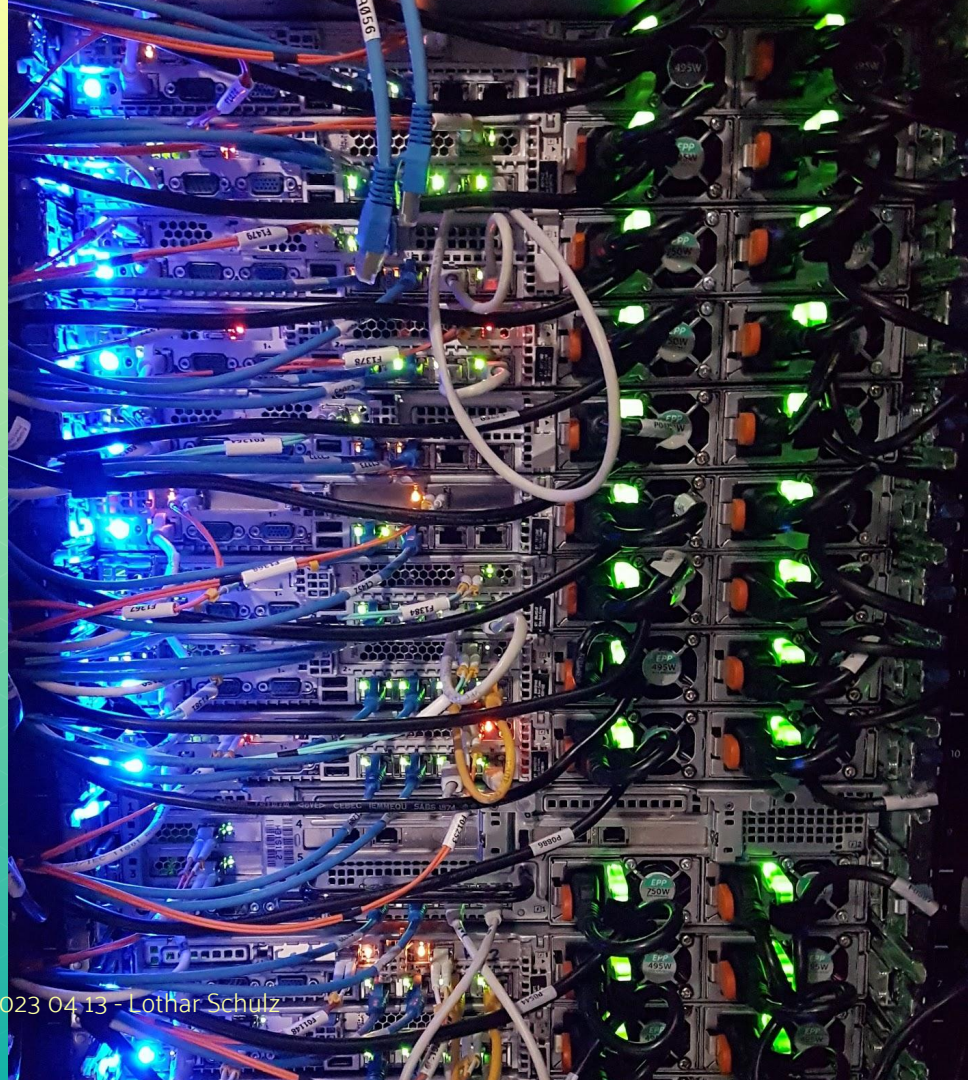
Total Nodes <a href="#">↗</a> <b>3</b>	Nodes Down <a href="#">↗</a> <b>0</b>	Alert Status <a href="#">↗</a> <b>Attention</b>	Cluster CPU <a href="#">↗</a>  <b>14.80%</b>	Cluster Capacity <a href="#">↗</a>  <b>97%</b>	NFSv3 Throughput <a href="#">↗</a> <b>34 Bps</b>	NFSv3 Op/s <a href="#">↗</a> <b>1 ops</b>	NFSv3 Latency <a href="#">↗</a> <b>0.07 ms</b>	SMB2 Throughput <a href="#">↗</a> <b>N/A</b>	SMB2 Op/s <a href="#">↗</a> <b>N/A</b>	SMB2 Latency <a href="#">↗</a> <b>N/A</b>
Total Nodes <a href="#">↗</a> <b>9</b>	Nodes Down <a href="#">↗</a> <b>0</b>	Alert Status <a href="#">↗</a> <b>Healthy</b>	Cluster CPU <a href="#">↗</a>  <b>14.40%</b>	Cluster Capacity <a href="#">↗</a>  <b>30%</b>	NFSv3 Throughput <a href="#">↗</a> <b>891 Bps</b>	NFSv3 Op/s <a href="#">↗</a> <b>48 ops</b>	NFSv3 Latency <a href="#">↗</a> <b>0.7 ms</b>	SMB2 Throughput <a href="#">↗</a> <b>N/A</b>	SMB2 Op/s <a href="#">↗</a> <b>N/A</b>	SMB2 Latency <a href="#">↗</a> <b>N/A</b>
Total Nodes <a href="#">↗</a> <b>9</b>	Nodes Down <a href="#">↗</a> <b>0</b>	Alert Status <a href="#">↗</a> <b>Attention</b>	Cluster CPU <a href="#">↗</a>  <b>8.10%</b>	Cluster Capacity <a href="#">↗</a>  <b>30%</b>	NFSv3 Throughput <a href="#">↗</a> <b>41.7 MBps</b>	NFSv3 Op/s <a href="#">↗</a> <b>25.7K ops</b>	NFSv3 Latency <a href="#">↗</a> <b>3 ms</b>	SMB2 Throughput <a href="#">↗</a> <b>187 kBps</b>	SMB2 Op/s <a href="#">↗</a> <b>83 ops</b>	SMB2 Latency <a href="#">↗</a> <b>1.6 ms</b>
Total Nodes <a href="#">↗</a> <b>3</b>	Nodes Down <a href="#">↗</a> <b>0</b>	Alert Status <a href="#">↗</a> <b>Healthy</b>	Cluster CPU <a href="#">↗</a>  <b>81.6%</b>	Cluster Capacity <a href="#">↗</a>  <b>7%</b>	NFSv3 Throughput <a href="#">↗</a> <b>119.2 MBps</b>	NFSv3 Op/s <a href="#">↗</a> <b>110.5K ops</b>	NFSv3 Latency <a href="#">↗</a> <b>21 ms</b>	SMB2 Throughput <a href="#">↗</a> <b>N/A</b>	SMB2 Op/s <a href="#">↗</a> <b>N/A</b>	SMB2 Latency <a href="#">↗</a> <b>N/A</b>
Total Nodes <a href="#">↗</a> <b>16</b>	Nodes Down <a href="#">↗</a> <b>0</b>	Alert Status <a href="#">↗</a> <b>Healthy</b>	Cluster CPU <a href="#">↗</a>  <b>24.3%</b>	Cluster Capacity <a href="#">↗</a>  <b>69%</b>	NFSv3 Throughput <a href="#">↗</a> <b>2.076 MBps</b>	NFSv3 Op/s <a href="#">↗</a> <b>3.00K ops</b>	NFSv3 Latency <a href="#">↗</a> <b>2 ms</b>	SMB2 Throughput <a href="#">↗</a> <b>57.1 kBps</b>	SMB2 Op/s <a href="#">↗</a> <b>713 ops</b>	SMB2 Latency <a href="#">↗</a> <b>0.7 ms</b>
Total Nodes <a href="#">↗</a> <b>15</b>	Nodes Down <a href="#">↗</a> <b>0</b>	Alert Status <a href="#">↗</a> <b>Attention</b>	Cluster CPU <a href="#">↗</a>  <b>18.00%</b>	Cluster Capacity <a href="#">↗</a>  <b>69%</b>	NFSv3 Throughput <a href="#">↗</a> <b>125.2 MBps</b>	NFSv3 Op/s <a href="#">↗</a> <b>33.8K ops</b>	NFSv3 Latency <a href="#">↗</a> <b>4 ms</b>	SMB2 Throughput <a href="#">↗</a> <b>N/A</b>	SMB2 Op/s <a href="#">↗</a> <b>N/A</b>	SMB2 Latency <a href="#">↗</a> <b>N/A</b>
Total Nodes <a href="#">↗</a>	Nodes Down <a href="#">↗</a>	Alert Status <a href="#">↗</a>	Cluster CPU <a href="#">↗</a> 	Cluster Capacity <a href="#">↗</a> 	NFSv3 Throughput <a href="#">↗</a> <b>146.7</b>	NFSv3 Op/s <a href="#">↗</a> <b>18.0K</b>	NFSv3 Latency <a href="#">↗</a>	SMB2 Throughput <a href="#">↗</a>	SMB2 Op/s <a href="#">↗</a>	SMB2 Latency <a href="#">↗</a>



# Tailor towards audience

Example:

- 24x7
- the engineering teams
- Management
- End customers



# Service Level Objectives

**Intuition, experience**, and an **understanding** of what engineers know about the services they serve is used to define

- service level indicators (SLIs),
- objectives (SLOs),
- and agreements (SLAs).

# Guidance – The Four Golden SRE Signals

- **request latency** - request response time and/or timeout rate
- **traffic / system throughput** - demand placed on the system - http requests, static & dynamic
- **error rate** - proportion of service errors
- **saturation** - measures the system fraction, emphasizing the resources that are most constrained (e.g., in a memory-constrained system, show memory; in an I/O-constrained system, show I/O).
- **availability** - what's the uptime of a service



# Guidance – DORA

**Google's DevOps Research and Assessment (DORA) team**

**DevOps Reports**

[Key Metrics to measure DevOps Performance](#)

# Guidance – DORA

key metrics that indicate the performance of a software development team:

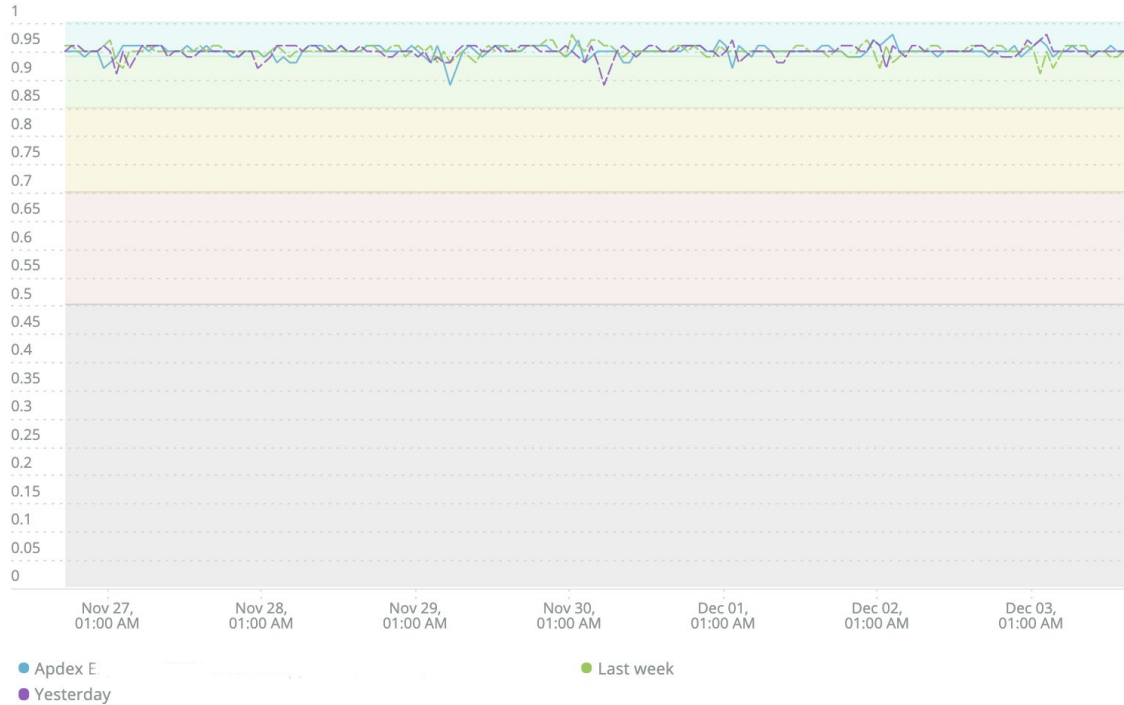
- **Deployment Frequency** - How often an organization successfully releases to production
- **Lead Time for Changes** - The amount of time it takes a commit to get into production
- **Change Failure Rate** - The percentage of deployments causing a failure in production
- **Time to Restore Service** - How long it takes an organization to recover from a failure in production
- **Reliability** - How long it takes an organization to recover from a failure in production

[Key Metrics to measure DevOps Performance](#)  
[DORA 2021 state of DevOps](#)

# Results

## Apdex /

Since 1 week ago compared with 1 week ago



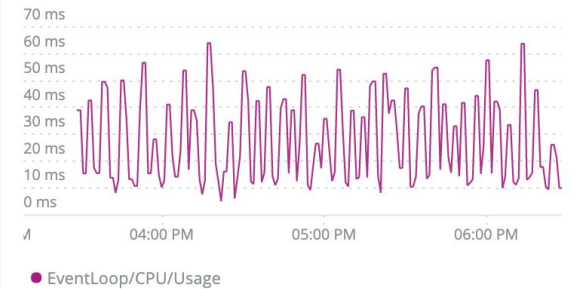
## API Throughput

Since 12 hours ago



## Max CPU Ticks

Since 3 hours ago



# Results

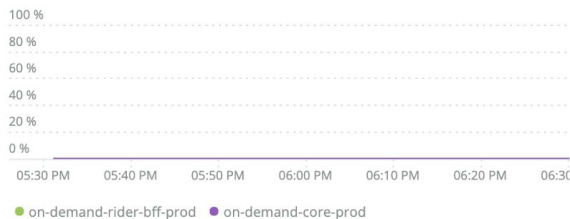
Error percentage (averaged over 1 minute)  
calls with http status code > 499

Since 1 minute ago

0 %  
5xx Errors

Error percentage over time (averaged over 1 minute)  
calls with http status code > 499

Since 60 minutes ago



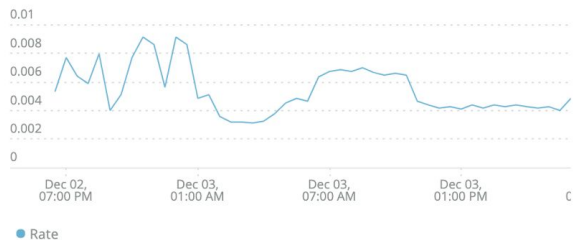
avg duration of 98%ile since one month ago

Since 1 month ago

APP NAME ↕	AVG DURATION INSECONDS (98%) ↕
-prod	3.313
-prod	0.258

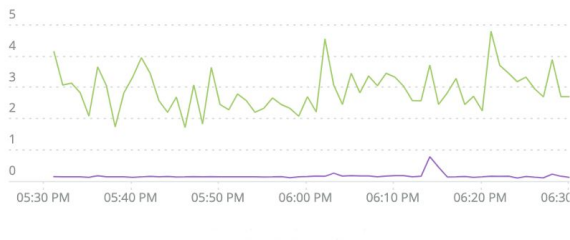
SLO: 98% of throughput < 1 s od core

Since 1 day ago



avg duration of 98%ile over one minute timeseries

Since 60 minutes ago



SLO: 98% of throughput < 1 s

Since 1 day ago



# Your Questions Please

- Micro or Macro
- Domains
- Contracts
- API Flavors
- Distributed Systems
- Observability

