架构师如何弥合理想与现实的冲突

陈超

汽车之家技术总监

InfoQ

精彩继续！
更多一线大厂前沿技术案例

北京站

GMTC
全球大前端技术大会

时间：9月15–16日
地点：北京·国际会议中心

扫码查看大会详情>>

北京站

QCon
全球软件开发大会

时间：9月17–19日
地点：北京·富力万丽酒店

扫码查看大会详情>>

杭州站

ArchSummit
全球架构师峰会

时间：9月26–27日
地点：杭州·和达希尔顿逸林酒店

扫码查看大会详情>>

自我介绍

现任汽车之家技术总监，统筹主机厂广告与数科业务技术发展
腾讯云TVP，QCon优秀出品人，ArchSummit明星讲师，Apache Committer

- 2011–2015　百度凤巢Tech Leader
- 2015–2016　某创业公司架构师，基础设施部负责人
- 2016–2020　猫眼娱乐架构师，平台技术部负责人
- 2020–2022　百度MEG ToB垂类技术委员会主席，爱番番业务部总架构师

是一场什么分享

不是什么

- 不是从0–1的实践手册
- 不是架构师技能树的科普分享
- 不是网上小抄整合而来的知识杂烩
- 不是照本宣科就能解决问题的银弹

是什么

- 是有配套工具的方法论分享
- 是观察、感知与思考
- 是试图从新维度上回答问题的尝试
- 是一场自己也没有完美答案的漫谈

大纲

- 澄清·什么是架构师
- 纾困·架构师之痛与思考
- 洞见·架构理念和趋势

“你总是提及的那个词，他的含义与你想表达的意思并不一样”

— 《公主新娘》

「架构师大爆炸」时代里，我们可能会这么定义架构师

独立设计系统?

手撸源码?

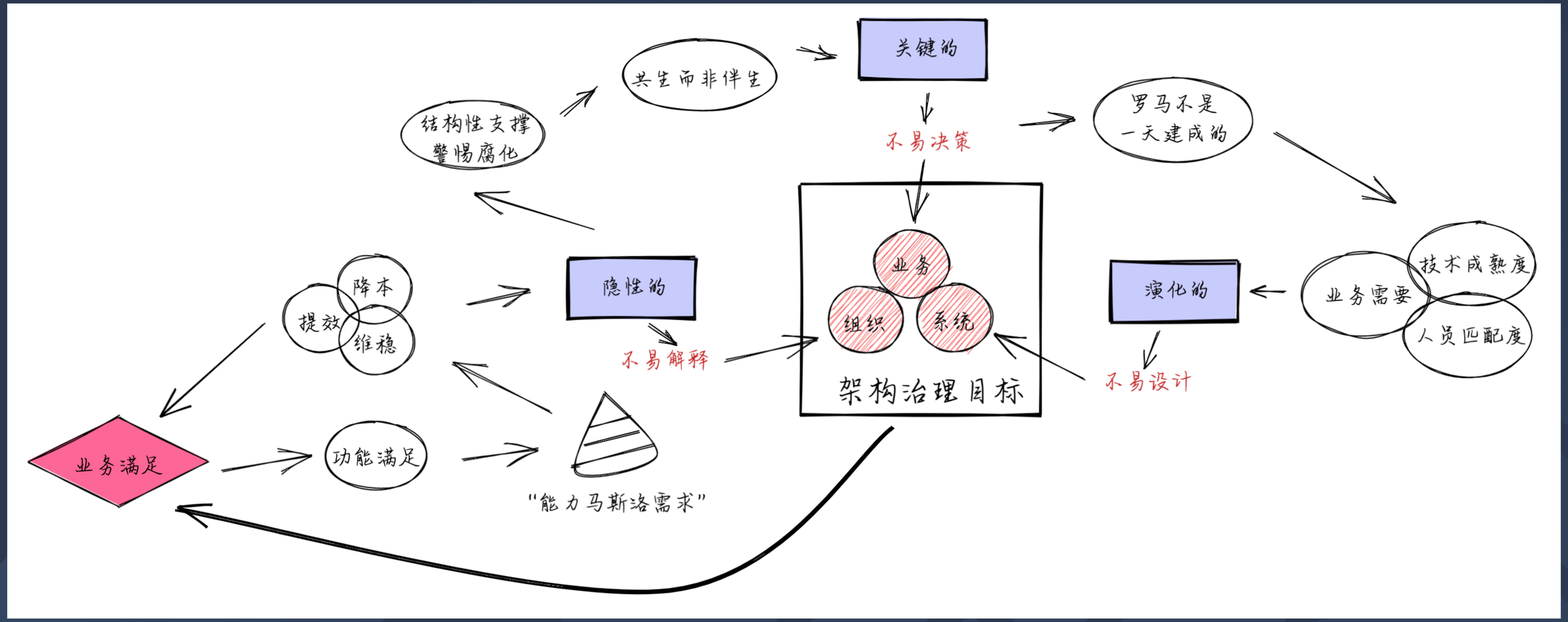
快速转化需求?

领域规划?

级别大于Pn?

业务驱动?

架构治理旨在满足不断发展的业务需要并确保系统之上的工程师获得最大幸福感



架构师是这道艺术题的答卷人，是难以被定义的，我们倾向找特征

看职责

- 系统不断满足业务，人员持续获得幸福感

看权限

- 角色模糊、一体化治理、头重脚轻介入

看范围

- 相对大领域的架构Owner/领域专家

看难度

- 流量、并发、业务复杂度的成功经验

大纲

- 澄清·什么是架构师
- **纾困·架构师之痛与思考**
- 洞见·架构理念和趋势

你是否也碰到这样的问题

- 这明明和最佳实践相悖，为什么我要做一个烟囱/洋葱架构
- 这API一点都不Rest，为什么要禁用Path Variable?
- 为什么他们团队不配合我这个更好的架构落地，一问就是没时间?
- 为什么我的这个系统拦截了这么多爬虫流量，但是价值却一直被质疑?
- 我觉得我很委屈，做了很多事老板不认价值，一出问题老板就认为是我的锅?
- 如何说服老板让公司的服务治理框架转向Servicemesh架构?
- 我要选择亲和k8s的原生架构，还是选择自建?
- 为什么Eureka要选择一条和Zookeeper完全不一样的实践路径?

问题分类归纳

知识陷阱

- 学习路径焦虑感
- 广而不精

实践悖论

- 最佳实践反模式
- 选型灾难

孤独症候

- 协同困境
- 理念孤岛

虚实失衡

- 无法形成方法论
- 过虚导致不落地

知识陷阱

知识陷阱 – 困境 – 海量知识下的焦虑

学习路径焦虑感

- 这么多技术要学，我该先学哪个
- 每次有新技术发布，我尽量要多深入了解细节

- 启动成本递增，选择带来的不选择
- 低水平精力转化带来的碎片化知识结构

广而不精

- 我基础架构的各个领域均有一定程度的掌握

- 全面发展等于全面平庸?

知识陷阱 – 解决思路 – T型知识体系构筑

克制

- 预设主攻领域
- 借力深剖架构与浅挖源码

打透

- 适当为难自己
- 关键设计洁癖

行动

- 先动起来，警惕“永远在规划”
- 艾宾浩斯遗忘曲线
- 善用工具
- 到开源中去

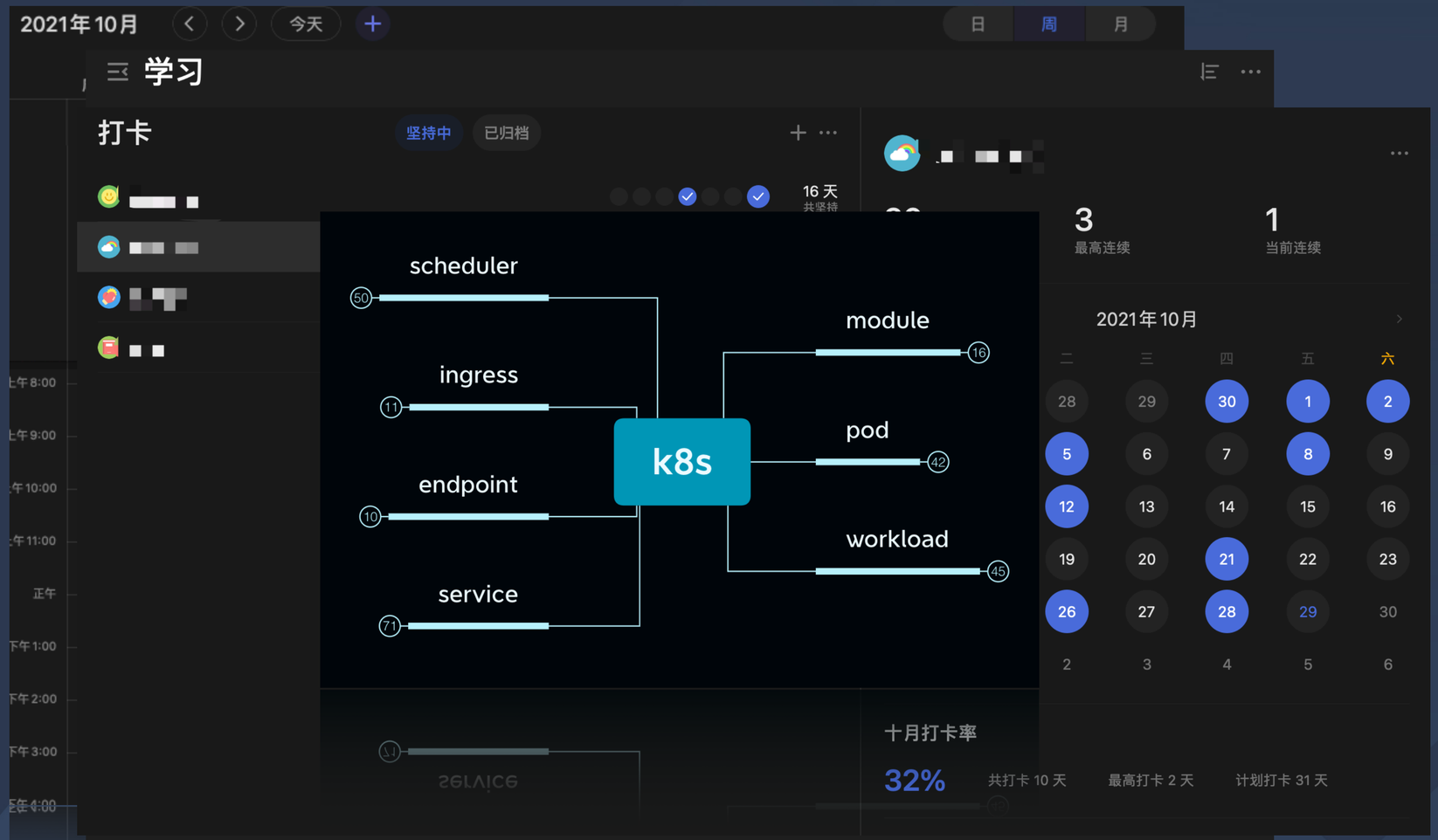
全面

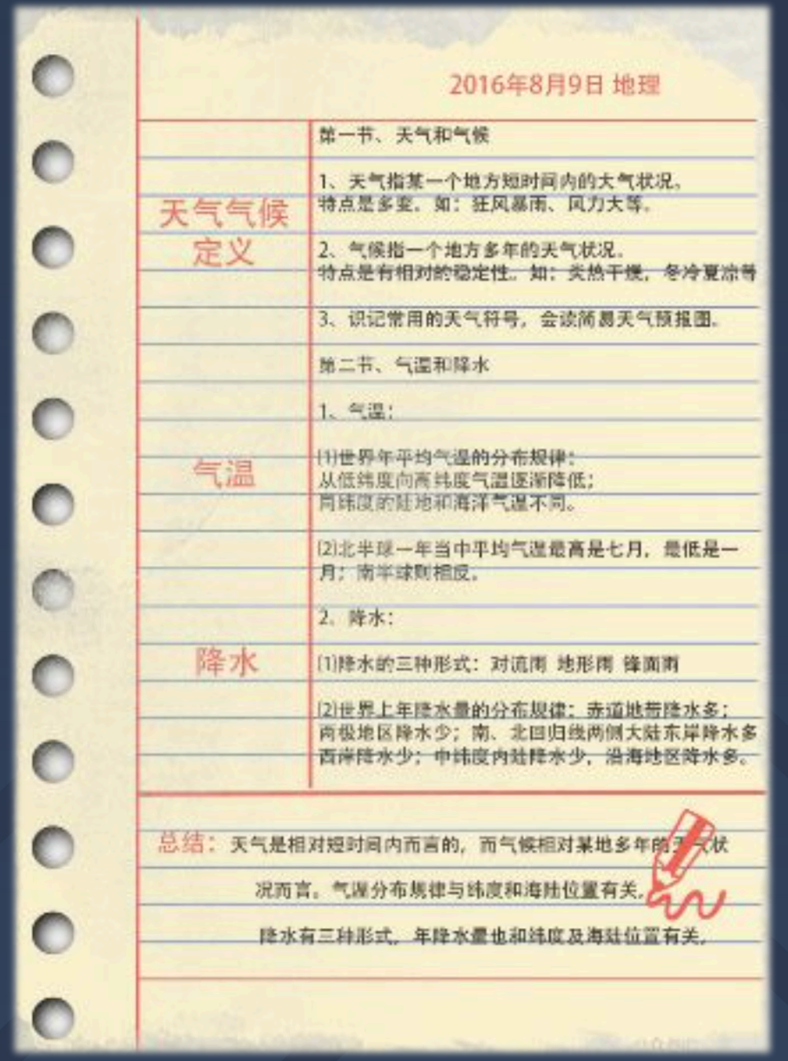
- 全栈心态
- 跨领域学习

奖励

- 阶段性产出带来获得感
- 打卡与分享

知识陷阱 – 工具 – GTD、日程、打卡、康奈尔笔记、脑图





实践悖论

实践悖论 – 困境 – 复合式问题带来的思维混沌

最佳实践反模式

- 为什么我要做一个烟囱/洋葱架构?
- 为什么要禁用Path Variable?
- “两个披萨”原则的微服务拆分为什么不灵了?

- 业务快速试错、职责分离、组织分离的常见选择
- 对反爬、监控、鉴权等设施带来的额外计算消耗
- 运维能力无法跟上后的系统失控

选型灾难

- Eureka比Zk更为简单易用，也更流行
- pb序列化性能和压缩率极高，流量大了一定要用
- Kafka比起其他MQ来说永远是选型的政治正确

- Eureka变更机制脆弱、项目闭源、麻烦的“自我保护”
- pb可读性的急剧下降带来的各种系统的改造成本
- 高流量大量partition情况下性能的急剧退化

实践悖论 – 解决思路 – 没有绝对意义上的最佳实践

不预设地广泛了解

- 知识陷阱规避并培养架构理念
- 摒弃地盘意识，模糊边界
- 抬头看路，月亮永远大于六便士

三维结构化对比

- 系统、组织、业务
- 功能与非功能性
- 数据密集型与计算密集型

由慢到快决策

- 慢思考，通盘考虑
- 资源和时间永远是不够的
- 避免群体决策和摇摆型决策

实践悖论 – 工具 – 技术选型指引表

考量维度	考量点	方案A	方案B
		方案与成本比较	
需求	需求本身是否合理、重要，是否要做减法		
	需求是否为快速试错的实验性需求		
	对当前和向后的业务需求的迎合度是否ok		
效能	引入带来的一次性成本（第一次接入/改造的成本）		
	引入带来的持续性成本（可扩展性、新需求的接入是否会导致成本线性/指数增长）		
	现有人员掌控住该架构的成本（运维、二次开发成本、学习曲线、社区活跃度）		
性能	应用层核心性能（核心接口读、写）		
	数据层核心性能（读、写、写入到可读）		
稳定性	应用层架构容灾能力		
	数据层不丢不重		
组织	维护升级的沟通协作成本		
	选型在部门范围内是否收敛统一		
	是否会带来组织和系统风险（如是否容易出现大SQL的思维引导、如引起逻辑过度下沉、使用一个小众技术栈、不可控的技术放开如应用脚本化将导致系统逐步走向不可控）		
其他	其他原因，如招投标时采用更先进/主流的技术能让系统具备更优异的表现从而更有吸引力等		
总体建议			

虚实失衡

虚实失衡 – 困境

无法形成方法论

- 我认为方法论没有用，是忽悠老板的
- 这个事情我列一个要做的todo然后推进
- 为什么老是找不到做事的思路

- 永远在执行
- 问题驱动开发
- 能力难以迁移

过虚导致不落地

- 我有很多方法论，但怎么落地呢

- 低质量方法论反而成为负担
- 路径信心丢失

虚实失衡 – 解决思路 – 务虚和务实之间需要找到平衡点

Bottom–up

- 自底向上去发现和归纳
- 避免单点思考
- 冰山分析与鱼骨图分析

Top–down

- 顶层分析、方法论驱动
- 结构化解构
- SMART度量

虚实失衡 – 工具

- 单点训练
- 垂直训练
- 水平训练
- 泛化训练

虚实失衡 – 工具 – 案例 – 单点训练 – 单一Feature突破

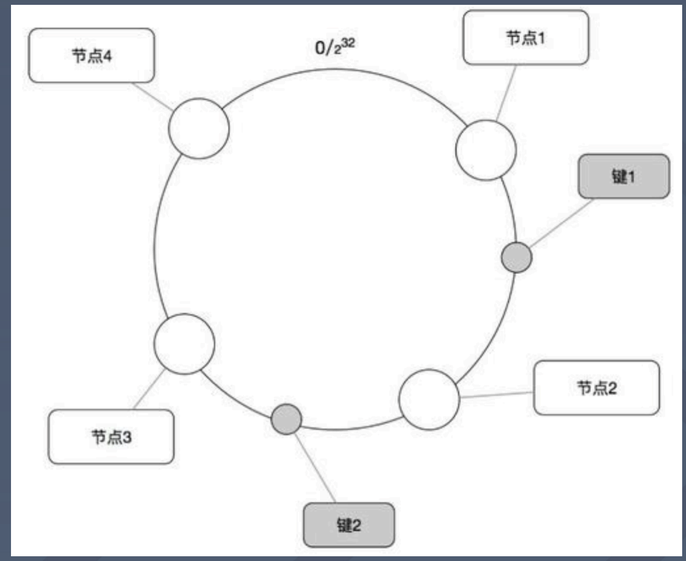
场景　学习Docker

洞察　“Docker通过虚拟化来达到标准化”

延展
- 基础环境：超线程、Docker、KVM
- 硬Proxy：DNS、四/七层负载、Mesh
- 软Proxy：SDK本身也是一种虚拟化
- 代码设计：设计模式中充斥着虚拟化

归纳　“虚拟化是解决问题的一大利器”

应用：一致性哈希防雪崩



虚实失衡 – 工具 – 案例 – 垂直训练 – 垂直的场景深挖

场景　评估系统容量

洞察　“我们对于什么是容量并没有统一的认知”

延展
- 定义：什么是容量
- 方法：该怎么测容量
- 工具：用什么测容量
- 效率：怎么提高测容量的效率
- 收益：怎么拿测容量的收益

归纳　应该从认知 - 落地 - 优化三个层面来评估容量

应用：如何推动容量评估

虚实失衡 – 工具 – 案例 – 水平训练 – 全链路推演

场景 大促前压测

洞察 “压测是事前演练，演练系统容灾和容量水平”

延展

故障前：压测、故障模拟、容量评估

故障中：诊断、限流、熔断、降级、自愈

故障后：复盘、记录、专项优化

归纳 **“稳定性治理是面向故障生命周期的闭环治理”**

应用：业务如何维稳？

虚实失衡 – 工具 – 案例 – 泛化训练 – 无边界延展

场景 学习Dubbo

洞察 “Dubbo本质上是SmartClient治理手段”

延展

中心治理：nginx、haproxy、atlas

融合治理：dubbo、motan、hsf、sofa

贴合治理：smartstack、local agent

归纳 “能力就近下沉是微服务的最大公约数”

方法论一：高并发维稳核心在于“分离”

方法论二：高并发维稳核心在于“一控四化”

方法论三：服务治理需要“一可二高三化”

方法论四：解决治理黑洞需要“3S”发展

……

应用：如何采集APM数据?

孤独症候

“如果你只担心技术问题，那么恐怕你看到的问题远不及一半”

——《微服务设计》

孤独症候 – 困境

协同困境

- 为什么别人不听我的?
- 为什么横向推动架构升级如此困难?
- 做了很多工作但不被老板认可价值，怎么办?

- 上、下、peer管理脱序
- landing困难，上任三把火到畏首畏尾

理念孤岛

- 明明我这个是最优解，为什么反对声那么多?
- 架构就应该有洁癖才能保证不被腐化
- 老板为什么对Servicemesh架构并不感冒?

- 不是在说服，就是在去说服的路上
- 不理解而带来的孤岛效应

孤独症候 – 解决思路

融入

对齐

推动

呈现

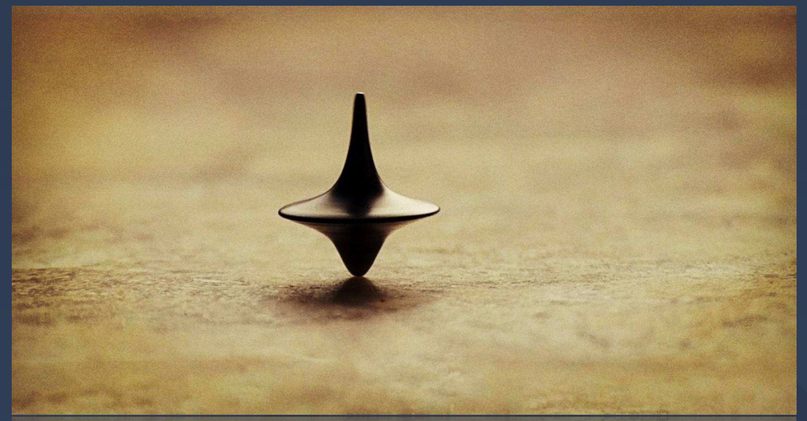
孤独症候 – 融入和对齐

理解鲶鱼效应

- 足够的授权
- 必然会带来不适感
- 快速能力展现
- 吐槽不能解决问题

对齐解决80%问题

- 组与织一样重要
- 不闭环的OKR形同虚设
- 避免陷入过度目标管理
- 价值传递，愿景构建



“让陀螺永不倒下”

孤独症候 – 解决思路 – 推动与呈现

避免群体决策

避免无效自尊

倾听与平衡

警惕破窗效应

勇气与担当

结构化呈现

自顶向下

由果及因

多不如精

细节藏着魔鬼

大纲

- 澄清·什么是架构师
- 纾困·架构师之痛与思考
- **洞见·架构理念和趋势**

架构核心理念

分治和分层是
架构的杀手级手段

擅用第一性原理来
对抗路径依赖

认清马斯洛需求
存活是根本

网络是
分布式架构
第一挑战

万物皆在熵增
架构的本质即在于反熵增

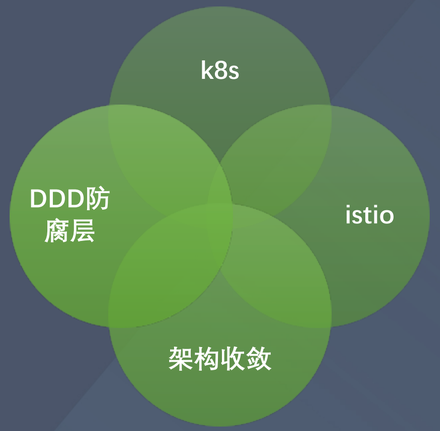
没有绝对意义上的
最佳实践

波特五力解构

黄金架构趋势

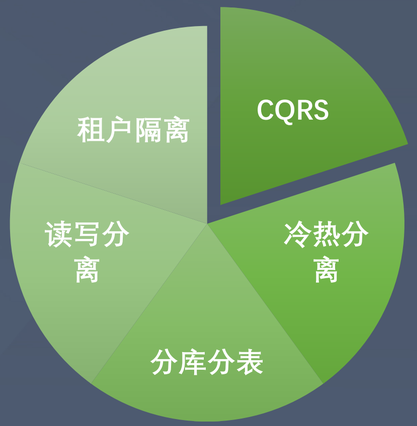
标准化

标准化是解放人效的根本



分离化

没有什么是分而治之解决不了的



流量本地化

网络永远是不可靠的，资源永远是不够的

服务云化

生产资料马太效应

黑盒化

边际成本治理需求

Take Away

四看

- 职责：系统、业务、人员
- 权限：模糊
- 范围：大领域
- 难度：成功经验

四痛

- 知识陷阱：构筑T型知识体系
- 实践悖论：广泛、结构性对比、慢到快决策
- 虚实失衡：洞察、延展、归纳反复锤炼
- 孤独症候：融入对齐、推动呈现

五化

- 标准化
- 分离化
- 流量本地化
- 服务云化
- 黑盒化

架构的本质在于反熵增

- 隐性而关键
- 需要持续投入
- 不易被理解

《数字人才发展体系：粮仓模型白皮书》

—— 推动数字人才全面发展

扫描左侧二维码
免费下载白皮书

极客时间 | 企业版

THANKS

Global

Architect Summit