HUAWEI

应用架构演进技术：微服务和中间件

高俊华

LEADING NEW ICT

内容

- 微服务
- 中间件

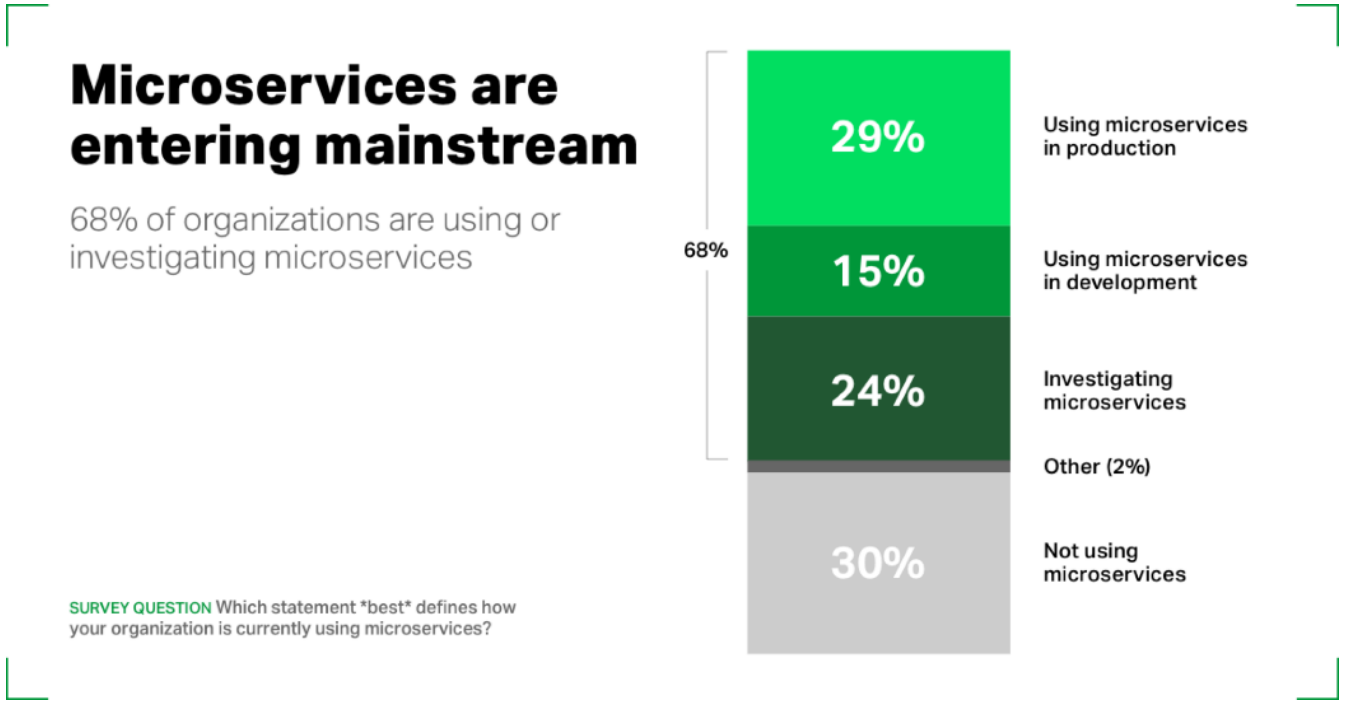
你遇到过这些问题么？

- 痛点1：技术架构/平台升级难
- 痛点2：测试、部署成本高
- 痛点3：可伸缩性差
- 痛点4：构建全功能团队难
- 痛点5：异常破坏性大

http://www.infoq.com/cn/articles/from-monolith-application-to-micro-service

你知道微服务么？

微服务化趋势



"Service-oriented architecture composed of loosely coupled elements that have bounded contexts"

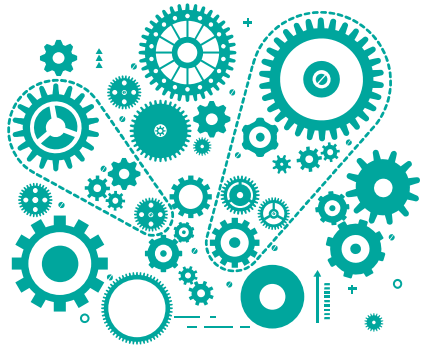
Adrian Cockcroft
(former Cloud Architect at Netflix)

https://www.nginx.com/resources/library/app-dev-survey/

Gartner认为：
1）微服务在未来2-5年内成为主流；
2）微服务架构的三个核心目标：
- development agility（敏捷开发）
- deployment flexibility（灵活部署）
- precise scalability（精准弹性）

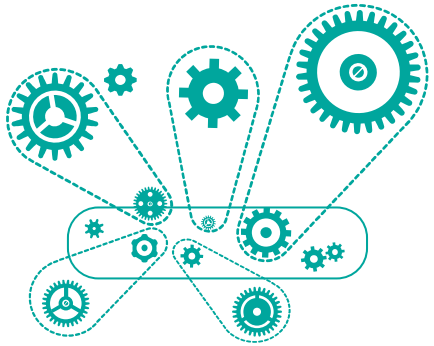
应用架构的变迁……

第一代：单体架构



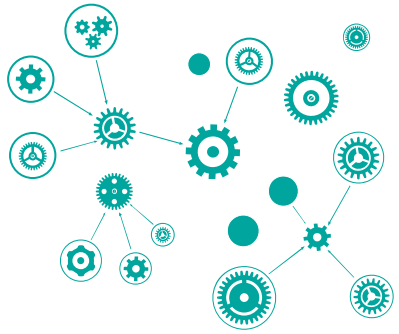
- 紧耦合
- 系统复杂、错综交互，动一发而牵全身
- 重复制造各种轮子：OS、DB、Middleware
- 完全封闭的架构

第二代：SOA架构



- 松耦合
- 通常通过ESB进行系统集成
- 有状态
- 大团队：100~200人
- TTM: 1年、半年、月
- 集中式、计划内停机扩容

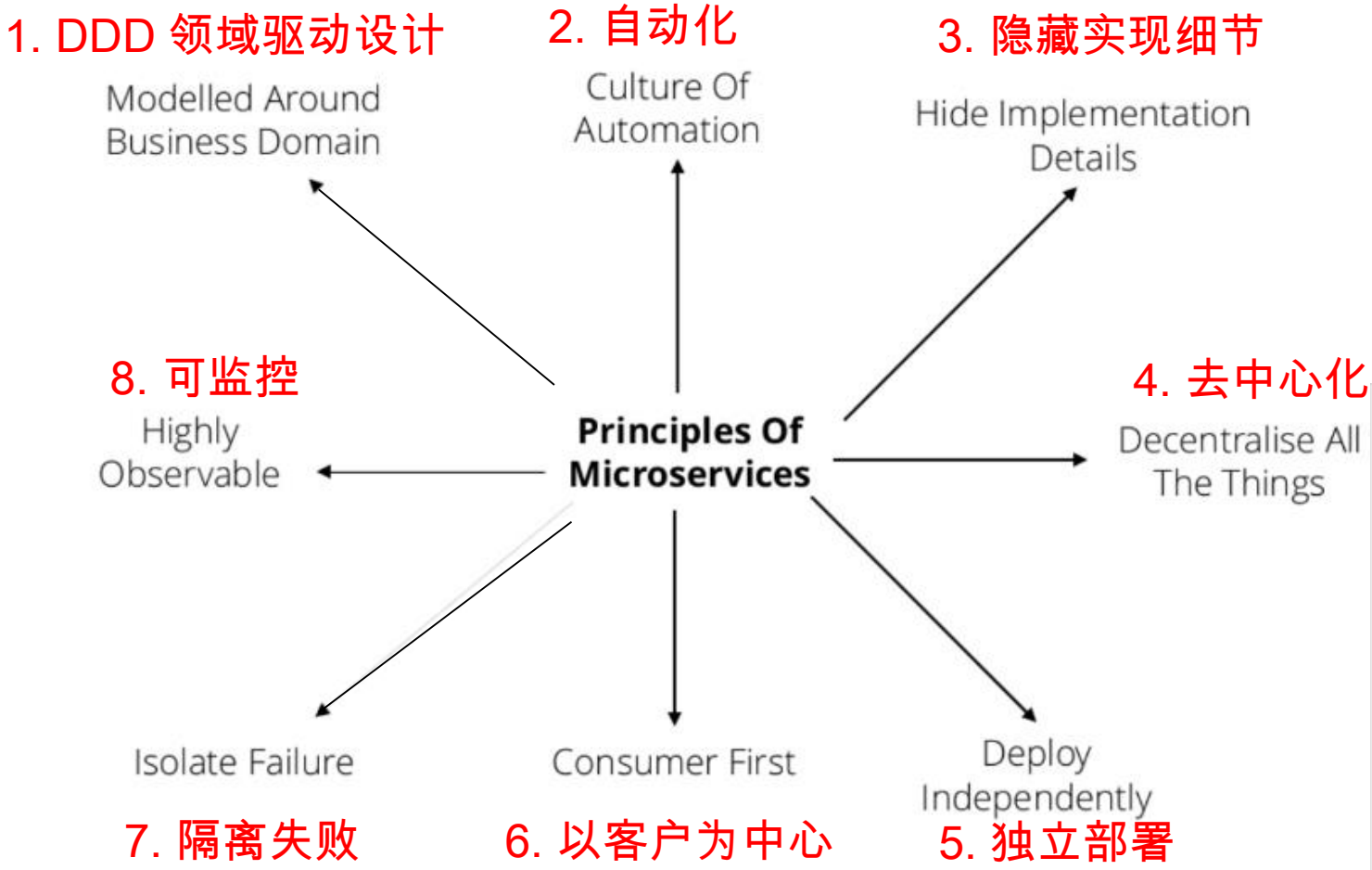
第三代：微服务架构



- 解耦
- 小团队：2 Pizza Team
- TTM: 按天、周进行升级发布
- DevOps: CI, CD, 全自动化
- 可扩展性：自动弹性伸缩
- 高可用：升级、扩容不中断业务

应用向CloudNative演进，微服务是CloudNative的事实标准

微服务设计原则




12-FACTORS

I. 基准代码
一份基准代码，多份部署

II. 依赖
显式声明依赖关系

III. 配置
在环境中存储配置

IV. 后端服务
把后端服务当作附加资源

V. 构建，发布，运行
严格分离构建和运行

VI. 进程
以一个或多个无状态进程运行应用

VII. 端口绑定
通过端口绑定提供服务

VIII. 并发
通过进程模型进行扩展

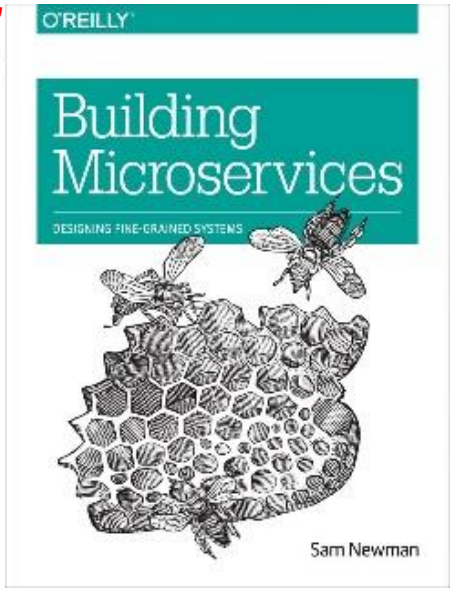
IX. 易处理
快速启动和优雅终止可最大化健壮性

X. 开发环境与线上环境等价
尽可能的保持开发，预发布，线上环境相同

XI. 日志
把日志当作事件流

XII. 管理进程
后台管理任务当作一次性进程运行

https://12factor.net/zh_cn/

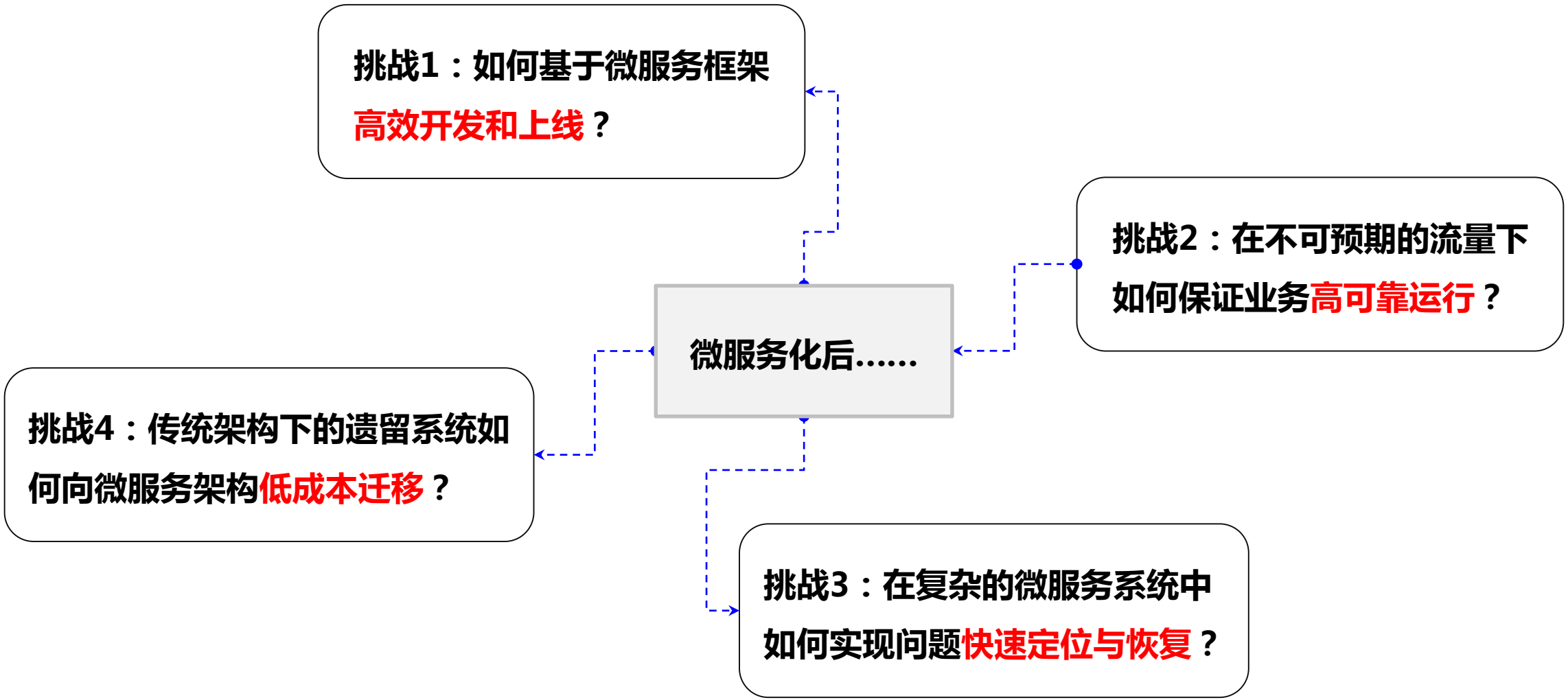


@velocityconf

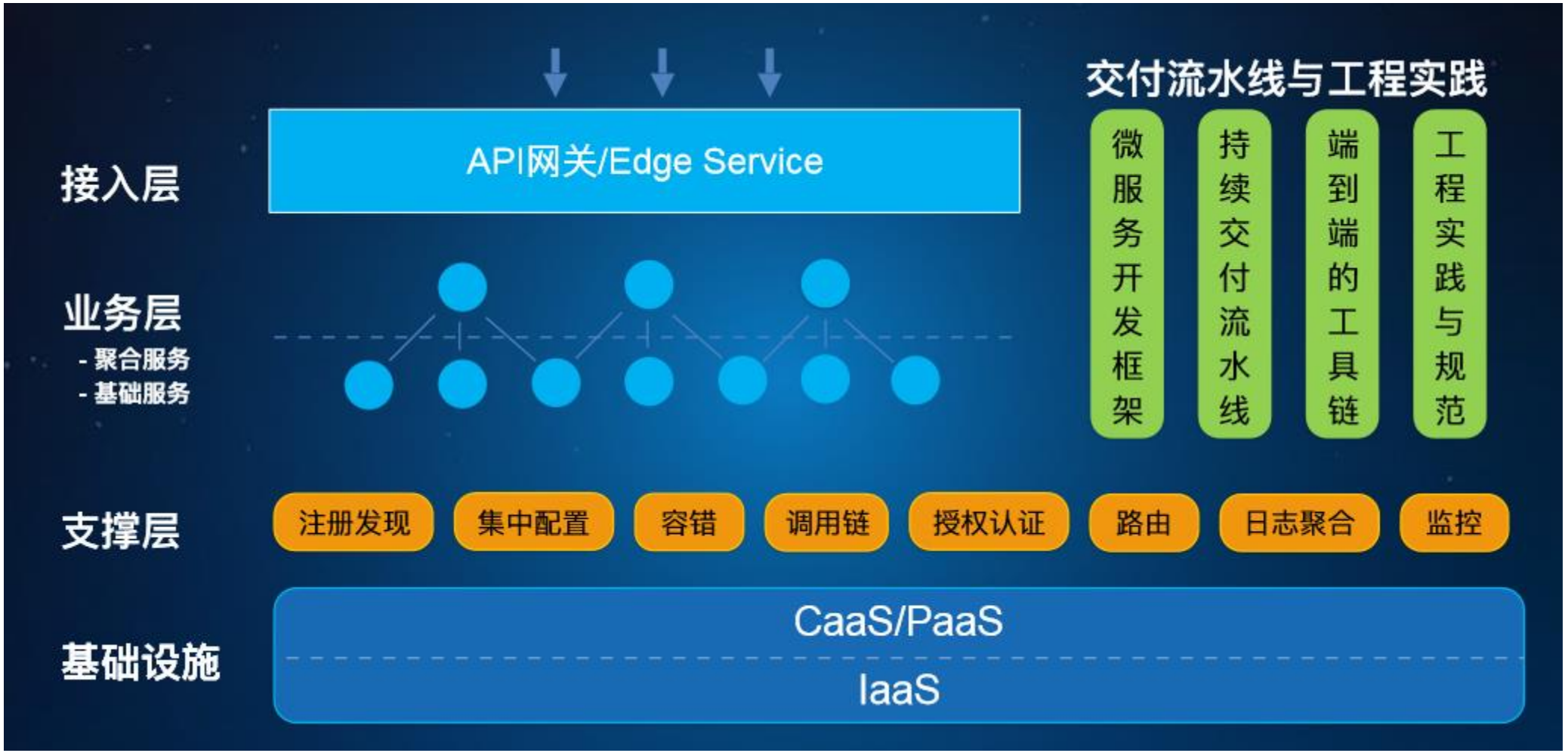
@samnewman

然而，有了微服务就完美了么？

微服务带来新的挑战……

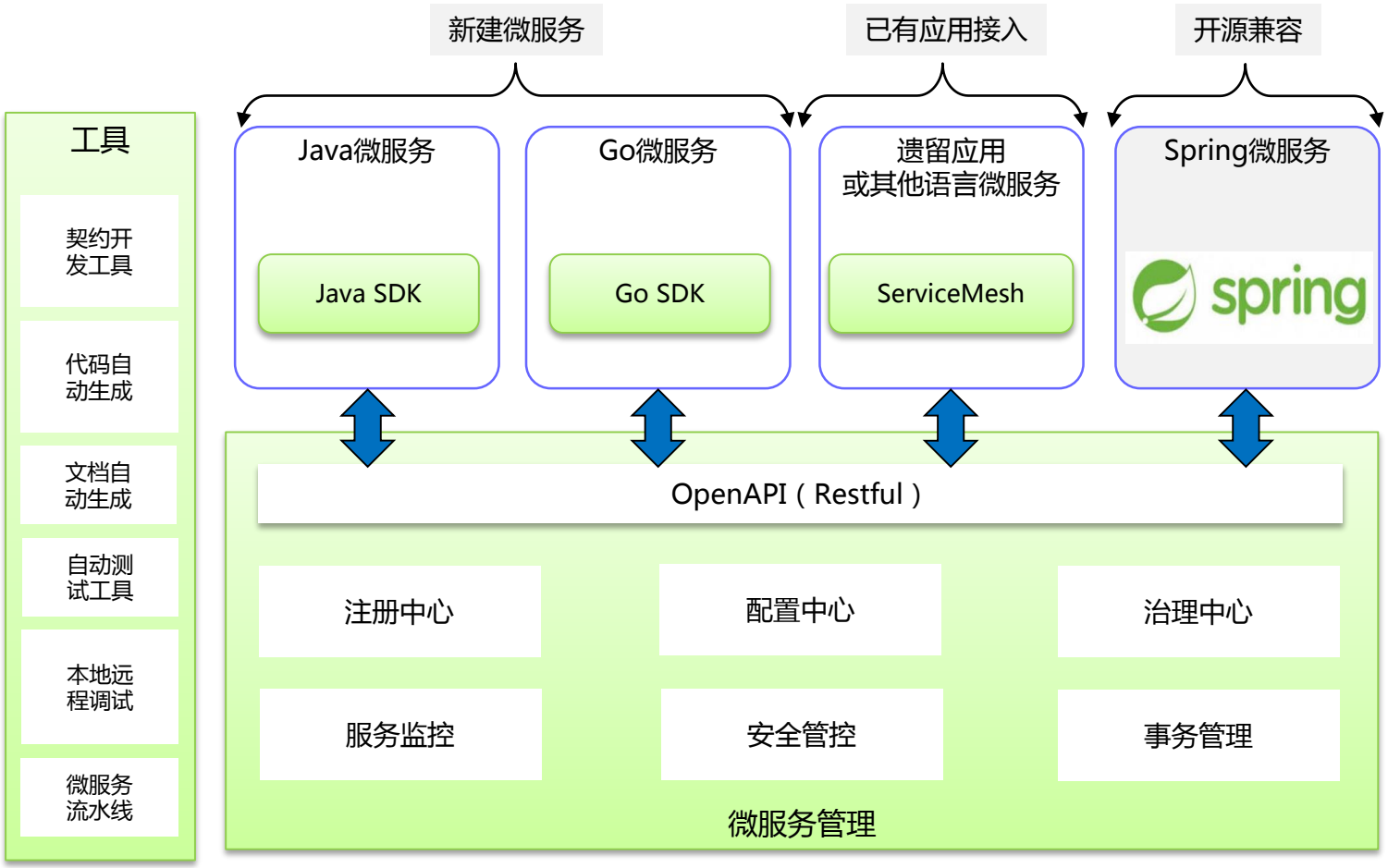

挑战1：如何基于微服务框架
高效开发和上线？
挑战2：在不可预期的流量下
如何保证业务高可靠运行？
微服务化后……
挑战4：传统架构下的遗留系统如
何向微服务架构低成本迁移？
挑战3：在复杂的微服务系统中
如何实现问题快速定位与恢复？

微服务生态系统


交付流水线与工程实践
微服务开发框架
持续交付流水线
端到端的工具链
工程实践与规范
接入层
API网关/Edge Service
业务层
- 聚合服务
- 基础服务
支撑层
注册发现
集中配置
容错
调用链
授权认证
路由
日志聚合
监控
基础设施
CaaS/PaaS
IaaS

华为微服务引擎CSE

为企业应用微服务化提供一站式解决方案



- **多语言微服务SDK**
  - ✓ JAVA SDK
  - ✓ GO SDK
- **简单易用的管理控制台**
  - ✓ ServiceMesh
  - ✓ 微服务注册中心
  - ✓ 微服务配置中心
  - ✓ 微服务治理中心
  - ✓ 微服务监控仪表盘
  - ✓ 微服务安全管控
  - ✓ 微服务事务管理框架
- **一系列配套的开发工具**
  - ✓ 流水线
  - ✓ 本地/远程调试
  - ✓ 代码/文档自动生成
  - ✓ ……

开源微服务框架ServiceComb

ServiceComb

Github Star:300+

开发者：20+

参与厂商：软通动力、猪八戒、绿城盟拓、中软国际等

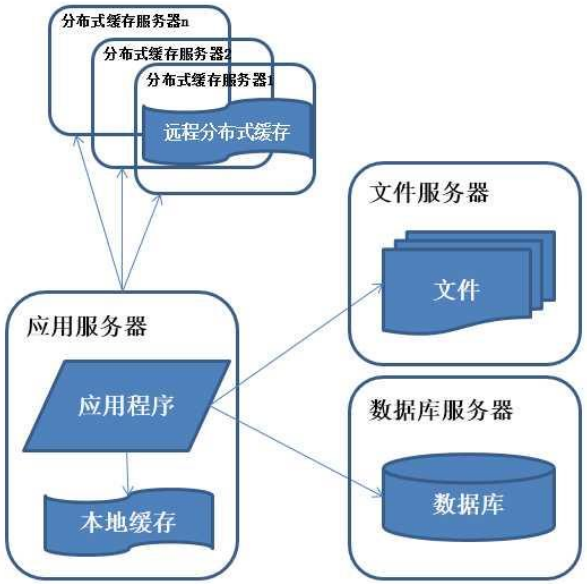
- Github：https://github.com/ServiceComb
- 官网：https://www.servicecomb.io

内容

- 微服务
- 中间件

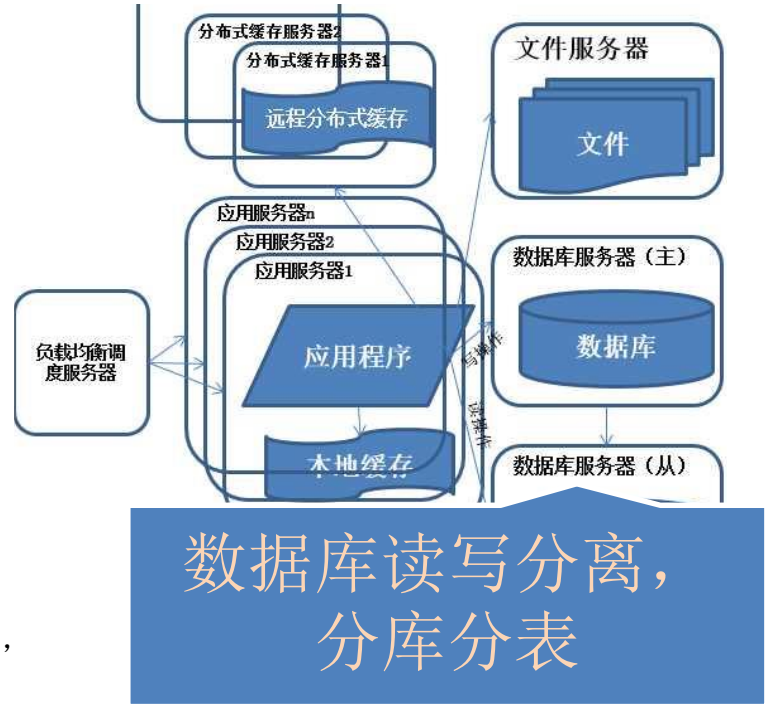
系统架构演化的另外一些问题……

使用缓存改善性能



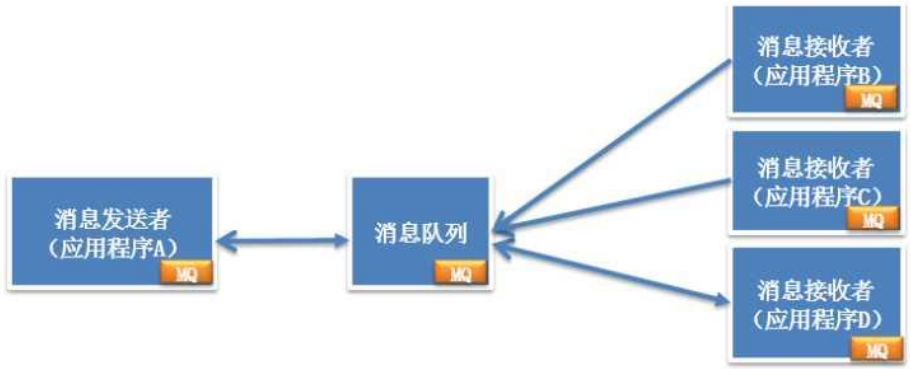
数据库中访问较集中的一小部分数据存储在缓存服务器中，减少数据库的访问次数，降低数据库的访问压力。

多台服务器通过负载均衡同时向外部提供服务，解决单台服务器处理能力和存储空间上限的问题。



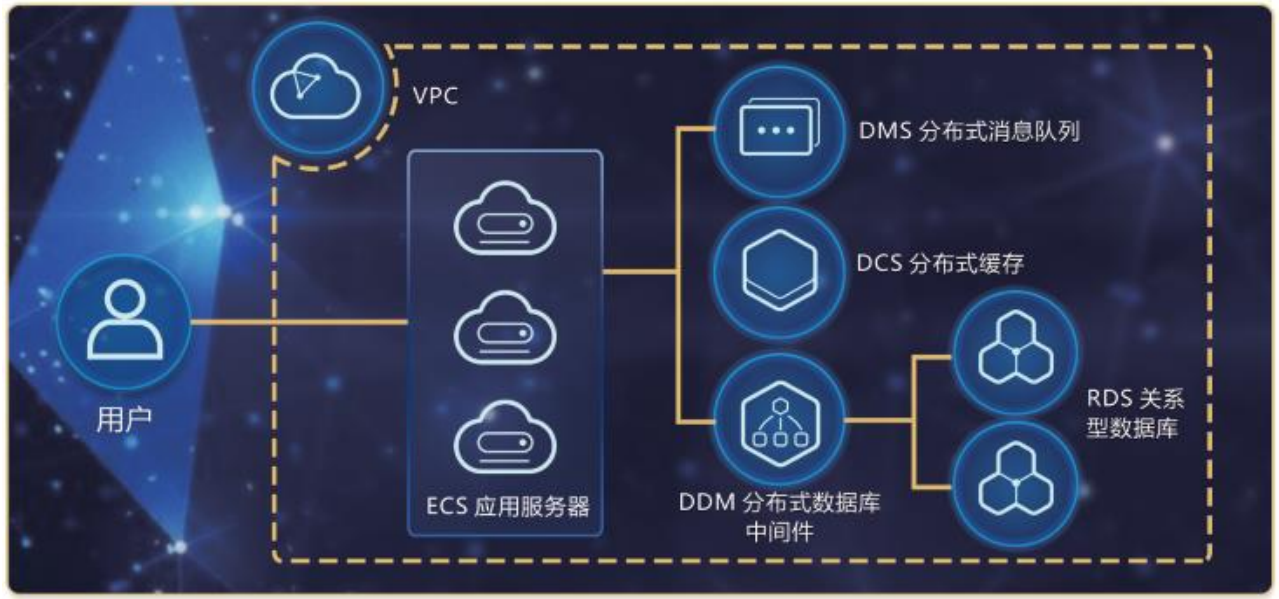
系统解耦，消息队列架构

消息对列通过消息对象分解系统耦合性，不同子系统处理同一个消息



https://www.zhihu.com/question/22764869/answer/31277656

华为云中间件



分布式消息服务（DMS）

完全托管的高性能消息队列服务，提供 Http API、TCP SDK、Kafka SDK 三种数据访问接口，为分布式应用提供灵活可靠的异步通信机制。

产品优势

- 提供普通队列、FIFO 保序队列、Kafka 队列，兼容原生 Kafka SDK，业务 0 改动迁移
- 消息过滤，消息回溯、消息重投，广播消息等丰富的消息能力
- 亿级消息堆积，可弹性扩展队列数，支持千万级并发，支持企业级高性能应用
- 同步落盘及多副本冗余，集群化部署确保数据和服务高可靠

分布式缓存服务（DCS）

兼容 Redis、Memcached、Ignite 等主流缓存引擎，基于双机热备的高可用架构，提供单机、主从、集群等丰富类型的缓存类型，满足用户高读写性能及快速数据访问的业务诉求。

产品优势

- 完美兼容原生开源协议，业务 0 改动迁移
- 一键弹性扩容，业务无感知
- 数据持久化和备份恢复确保数据高可靠
- 30+ 项监控指标，支持自定义监控阈值和告警策略

分布式数据库中间件（DDM）

专注于解决数据库分布式扩展问题，突破传统数据库的容量和性能瓶颈，通过读写分离、数据分片等实现海量数据高并发访问。

产品优势

- 业务不中断，自动完成水平拆分、平滑扩容
- 可承受 PB 级数据量、百万级并发量，十倍于单机数据库连接数
- 集群高可用，秒级故障自动恢复
- 兼容 MySQL 协议，业务代码 0 改动，透明读写分离

THANK YOU