

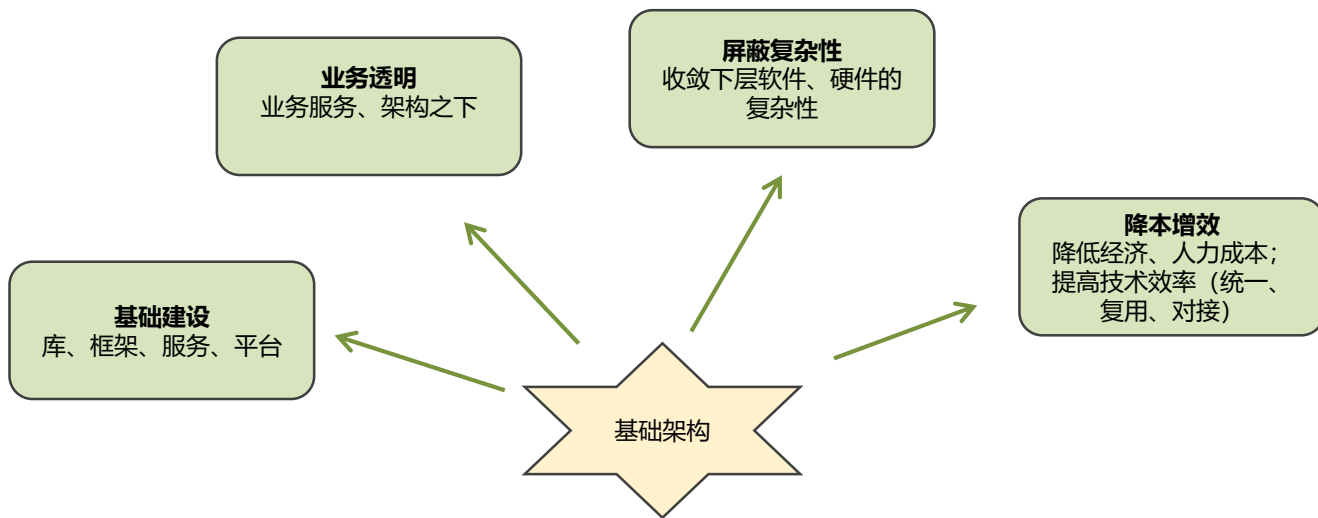


基础架构与中间件图谱

许雪里

www.xuxueli.com

基础架构是什么？





单体应用/Monolith Application

开发效率低
代码维护难
稳定性差
新人懵逼

故障
项目延期
跟不上业务发展
无法扩展

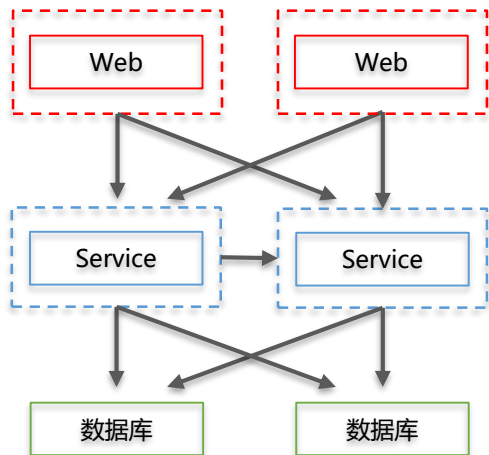
坑

优点:

- 1、为人所熟知
- 2、IDE友好
- 3、易于共享
- 4、易于测试
- 5、易于部署
- 6、运行稳定

缺点:

- 1、代码重复率高
- 2、需求变更困难
- 3、妨碍持续交付
- 4、功能孤岛
- 5、共生共灭
- 6、局限的弹性与扩展能力
- 7、阻碍技术革新



面向服务架构/Service-Oriented Architecture

依赖关系复杂
时延
团队进度协同困难
运维效率底
事务一致性
新人懵逼
稳定性差
问题定位难

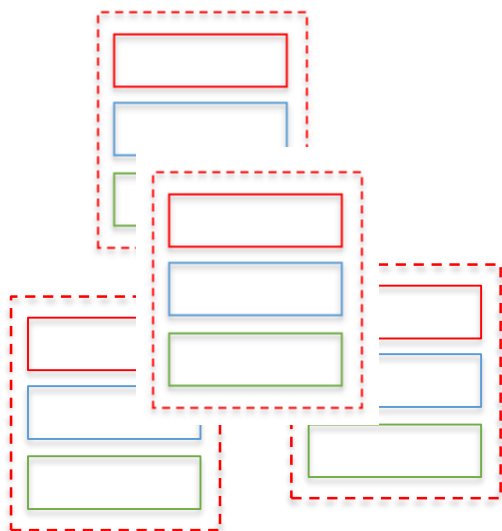
坑

优点:

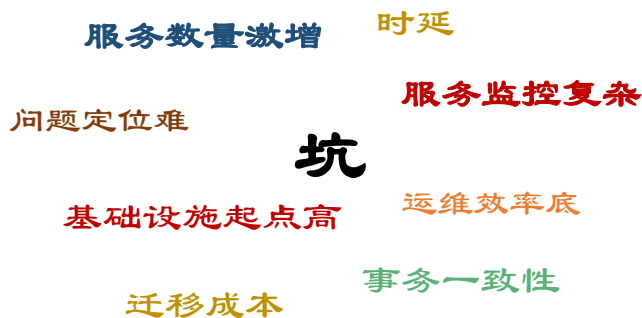
- 1、水平扩展、服务可伸缩
- 2、逻辑/资源复用
- 3、业务专注
- 4、独立测试部署
- 5、敏捷交付

缺点:

- 1、依赖关系复杂
- 2、可靠性差
- 3、开发/测试/运维困复杂
- 4、事务一致性
- 5、问题定位困难
- 6、基础设施&团队要求高



微服务/Micro Services



优点:

- 1、有效拆分
- 2、独立部署 (服务自治)
- 3、避免依赖
- 4、敏捷运维
- 5、敏捷交付

缺点:

- 1、服务数量激增
- 2、调用链路变长
- 3、服务监控复杂
- 4、迁移成本
- 5、人员技能培训成本
- 6、基础设施&团队要求高

1、业务初始时期



解决方案：开源

目标：快速迭代、按时上线功能

优点：

- 1、性价比
- 2、稳定性
- 3、学习和维护成本

2、业务快速发展时期



解决方案：二次开发（基于开源）

目标：解决业务特性要求
（金融-强一致性、互联网-高性能）

挑战：

- 1、开源软件无法完善满足业务；
- 2、业务迭代迅速，供技术改进资源有限

优点：

- 1、成本低，改动小，影响低
- 2、兼容性、契合业务行业
- 3、封装复杂性

3、业务形成规模，且趋于稳定



解决方案：自研

目标：量身打造，契合行业特性

开源产品的局限性：

- 1、依赖太重
- 2、功能不足，只满足部分需求
- 3、注重通用性，行业特性需求无法满足；

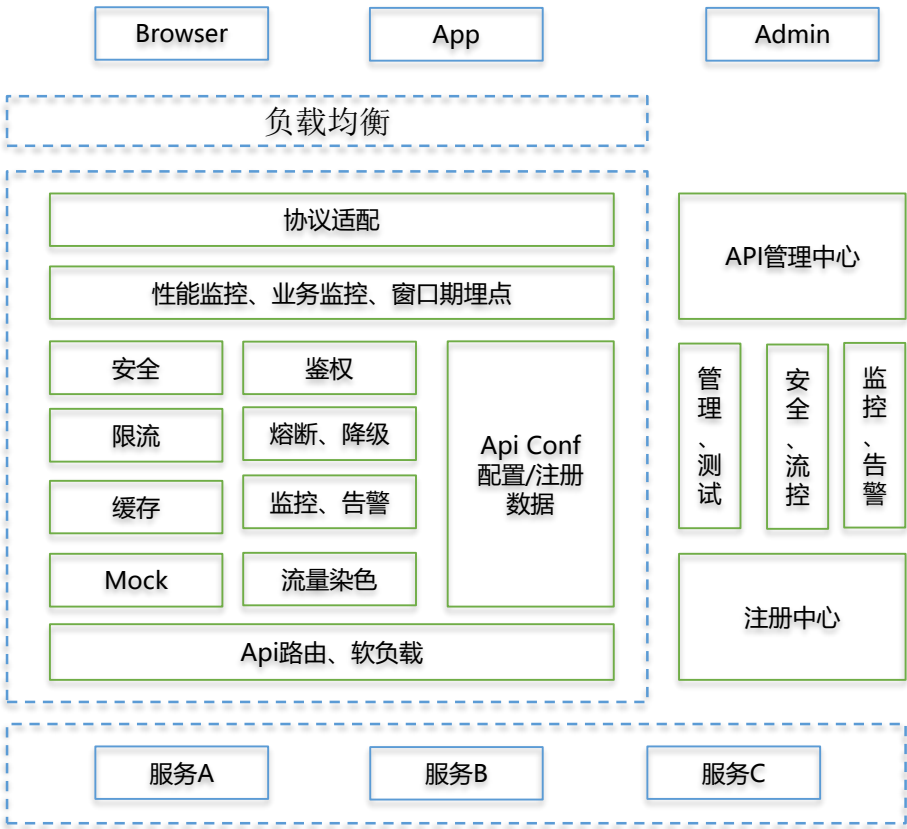
条件：

- 1、目标明确，业务模式成型
- 2、人力充足

优点：

- 1、降低成本
- 2、技术栈统一：复用、对接
- 3、收敛复杂度

- 1、高性能 (Performance)：** TPS、响应时间、性能计数器；
如：异步、空间换时间、代码和算法优化；
- 2、高可用 (Availability)：** 系统正常运行的概率、时间占有率（减少停工时间），99.99、99.999；
如：负载均衡、主备、故障转移、质量检测与提升；
- 3、可扩展性 (Scalability)：** 通过增加逻辑单元实现系统处理能力线性增长，包括水平与垂直扩展；
如：集群、负载均衡、分布式协同；
- 4、平台化：** 从锤炼产品的角度去权衡功能点，提高用户体验。
常规的“场景-框架”的开发模式，升级为“场景-平台-用户”的开发模式，效率高、体验好。
- 5、领域化：** 基础架构领域化划分，各模块高内聚、低耦合，职责单一。促进架构统一。
- 6、自动化运维：** 促进开发、QA和运维部门之间的沟通与协作，快速及时的交付产品和服务。
如：常规调度任务的起停、日志追踪与扩缩等需运维介入，而接入XXL-JOB后，开发可轻松单独完成。
- 7、成本：** 经济、学习、运维成本；

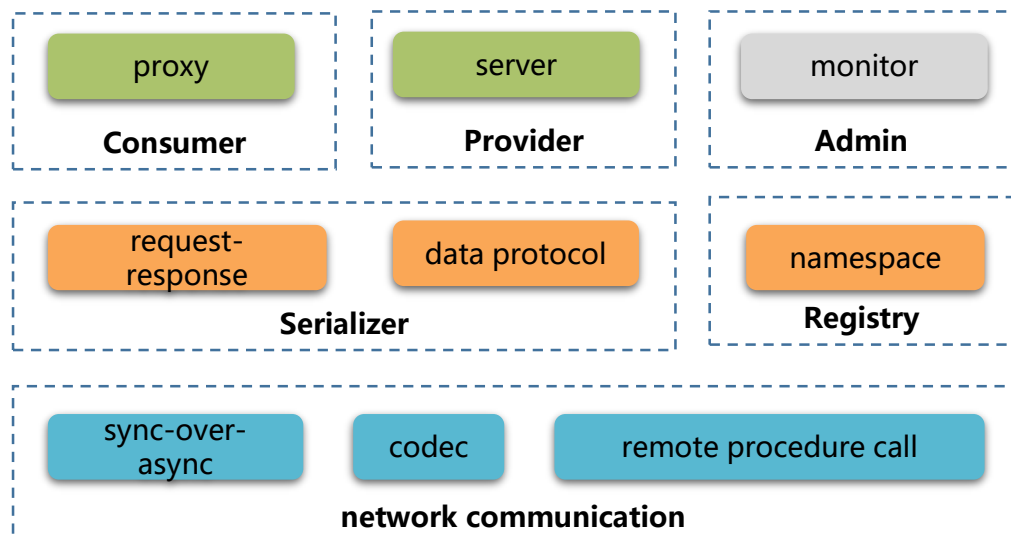


概述:
提供统一的对外的API入口。
涵盖API发布、管理、运维、监控的全生命周期

承接上下游请求与服务，处理所有的非业务功能。

特性:
负载均衡、安全、鉴权、限流、熔断/降级、缓存、
监控报警等

增强特性:
1、参数定制
2、聚合
3、缓存
3、染色代理



概述:

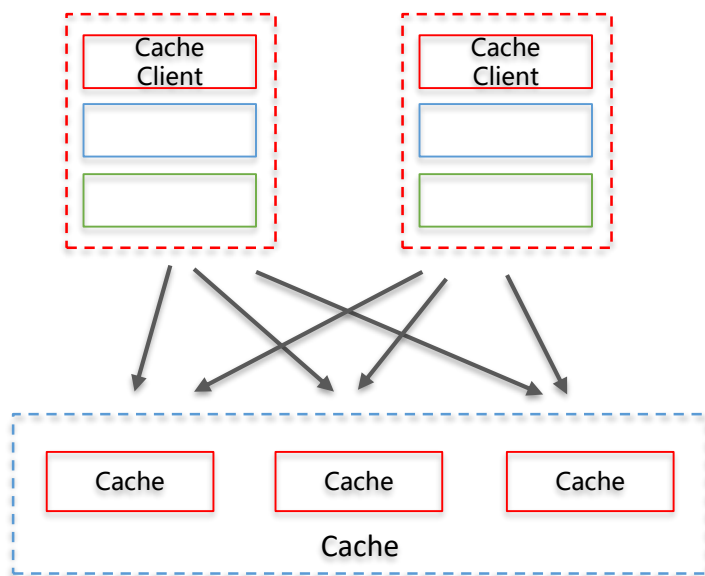
高效构建分布式应用，
提供远程调用能力时，
不损失本地调用的语义简洁性
。

特性:

- 1、透明调用
- 2、服务注册与发现
- 3、软负载
(1+1问题)

增强特性:

- 1、限流
- 2、熔断
- 3、路由策略、权重
- 4、服务监控



概述：

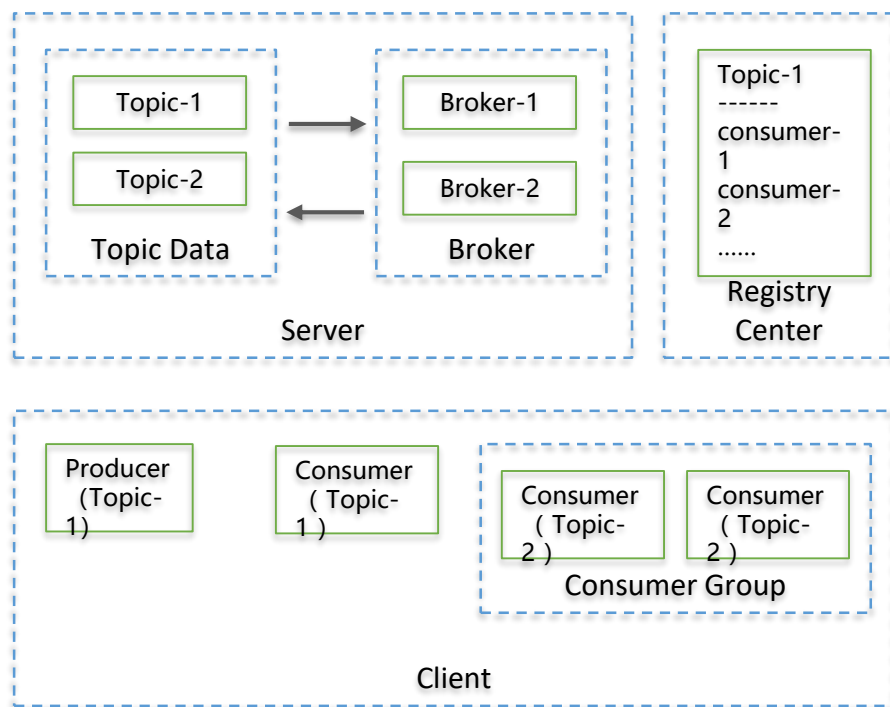
性能优化银弹，“缓存为王”

集群方式：

- 1、客户端集群：一致性hash/虚拟节点
- 2、服务端集群：redis3.0 cluster

三大挑战

- 缓存穿透：缓存命中率
- 缓存击穿：热点缓存过期
- 缓存雪崩：缓存批量过期



概述:

- 1、异步
- 2、解耦
- 3、消除峰值

消息模式:

- 1、广播消息
- 2、并发队列 (分片)
- 3、串行队列

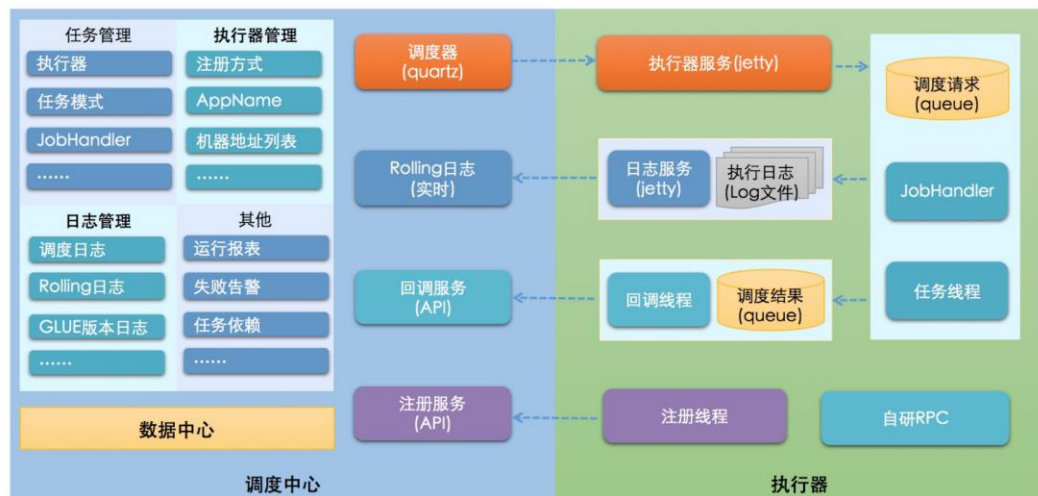
特性:

- 1、持久化
- 2、事务消费
- 3、消息可追踪
- 4、延迟消息
- 5、消息重试

基础架构领域之——任务调度



基础架构与中间件图谱



XXL-JOB架构图 v1.9

概述:

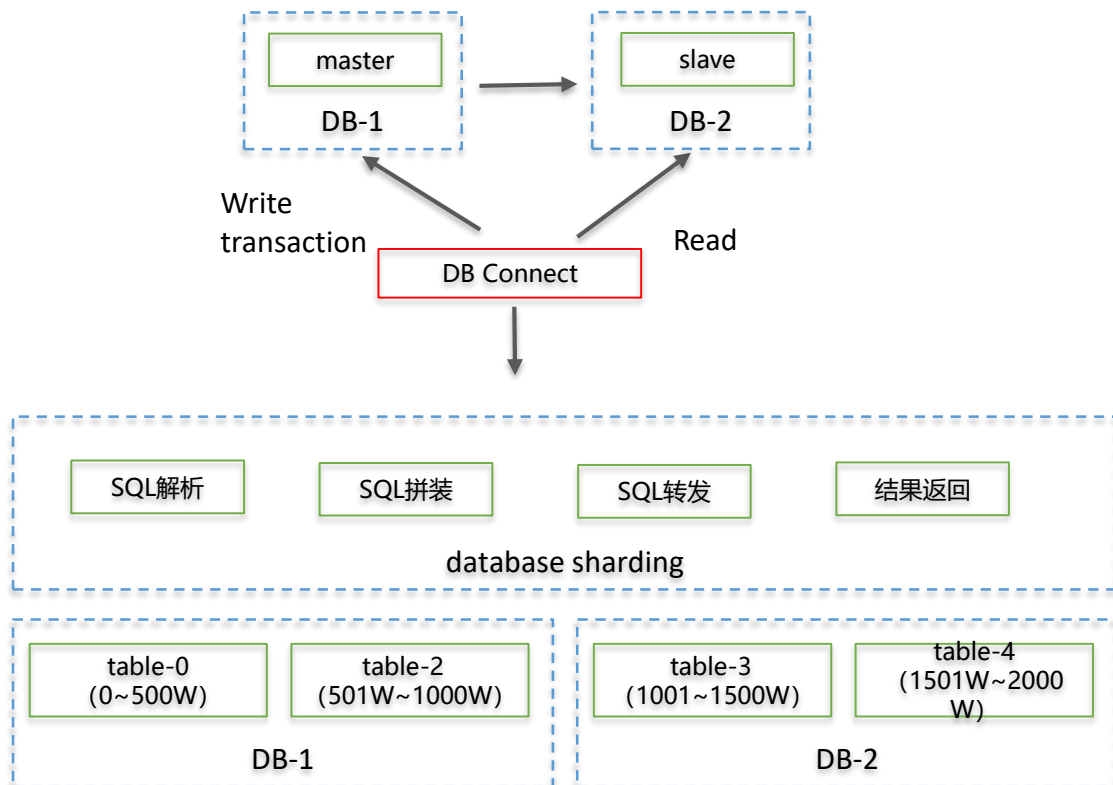
一些周期性、指定时间点等方式自动触发的异步业务逻辑，他们拥有一个显著的共享，即“与时间相关”。

场景

索引同步、pv统计、订单超时处理

核心特性

- 1、定时触发;
- 2、HA/集群
- 3、弹性扩缩
- 4、故障处理
- 5、阻塞处理
- 6、解耦
- 7、高性能
- 8、平台化
- 9、自运维



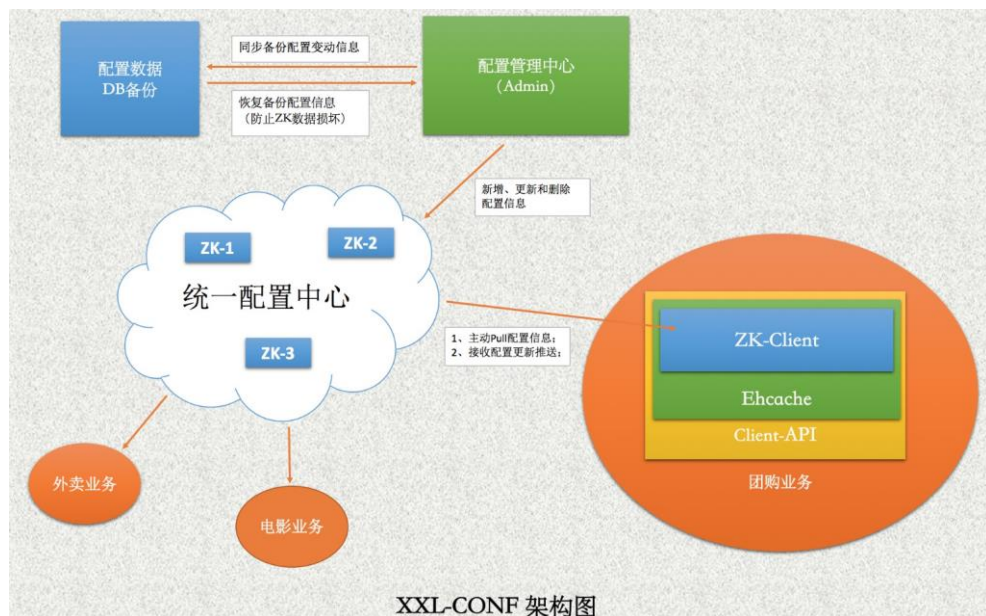
概述:

海量数据的存储和访问。

核心特性

- 1、读写分离;
- 2、分库分表
- 3、动态配置/切换
- 4、监控

基础架构领域之——配置中心



概述：

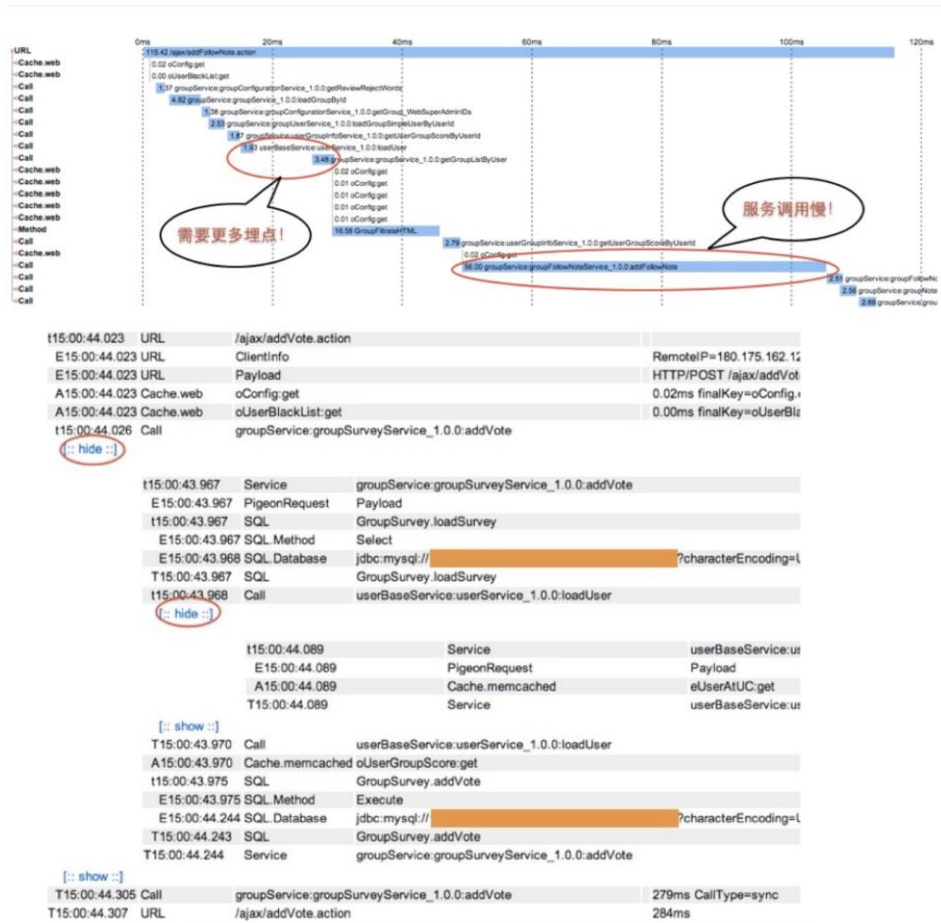
提供统一的配置管理服务。
在线操作，毫秒级变更生效，
统一上线包；

场景

JDBC 连接、日志 Level、业务功能开关

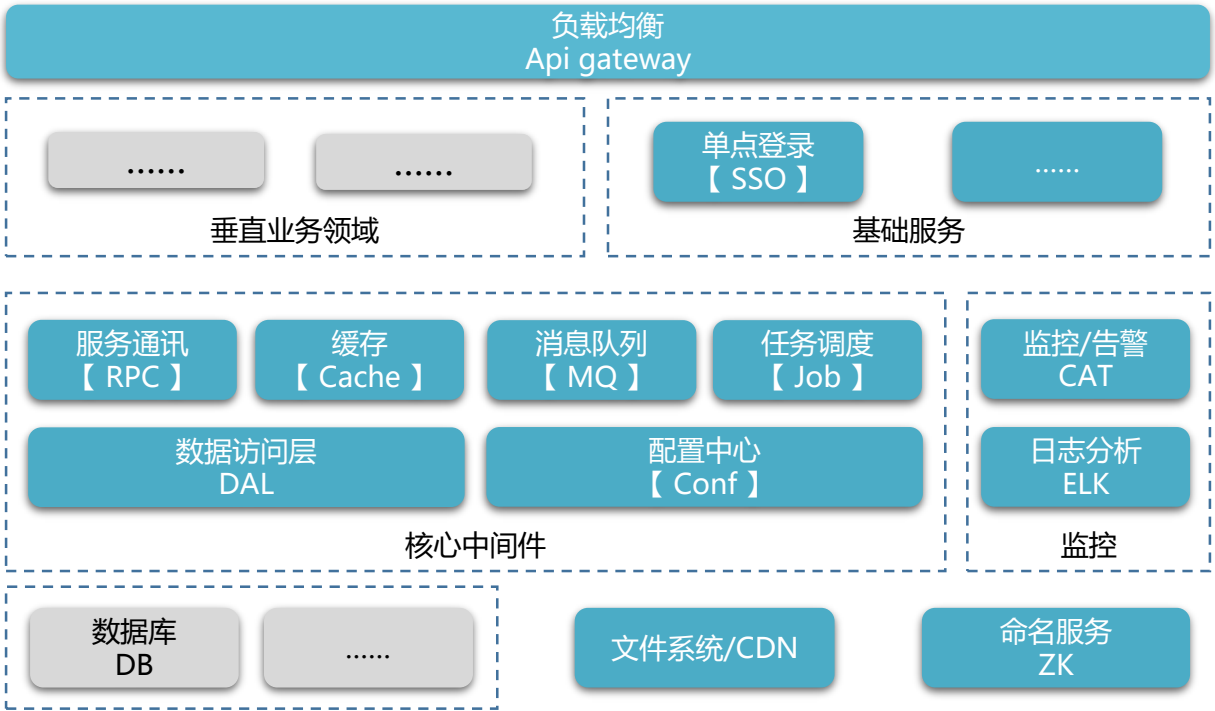
核心特性

- 1、在线管理
- 2、动态推送
- 3、毫秒级生效
- 3、配置备份
- 4、API 友好：api + xml + annotation
- 5、配置变更监听
- 6、配置权限控制
- 7、历史版本回滚
- 8、.....



概述：
系统的性能/业务指标、健康状况的监控与告警；

- 核心特性
- 1、Transaction：一段逻辑执行时间和次数，95/99线
 - 2、Event：一段逻辑的执行次数，打点、成功率
 - 3、Heartbeat：机器和JVM状态信息，MEM、GC等
 - 4、Metric：业务指标，次数/平均值/总和等类型
 - 5、Trace：基本的trace信息，如log4j的info信息
 -
 - 6、Problem：系统异常或LongService
 - 7、Dependency：统之间实时调用数据信息，包括远程服务、数据库、缓存等
 - 8、State：自身状态监控，包括处理消息数、丢失消息，以及系统大盘等；
 - 9、告警：Problem、性能下降时，邮件或短信自动告警



XXL 生态圈

www.xuxueli.com

<div>XXL-JOB 分布式任务调度平台</div> <div>Star4,140</div> <div>Homepage</div>	<div>XXL-API API管理平台</div> <div>Star452</div> <div>Homepage</div>	<div>XXL-CONF 分布式配置管理平台</div> <div>Star267</div> <div>Homepage</div>
<div>XXL-MQ 分布式消息队列</div> <div>Star148</div> <div>Homepage</div>	<div>XXL-RPC 分布式服务通讯框架</div> <div>Star143</div> <div>Homepage</div>	<div>XXL-CACHE 分布式缓存管理平台</div> <div>Star88</div> <div>Homepage</div>
<div>XXL-SSO 分布式单点登录框架</div> <div>Star203</div> <div>Homepage</div>	<div>XXL-GLUE 分布式逻辑管理平台</div> <div>Star79</div> <div>Homepage</div>	<div>XXL-CRAWLER 分布式爬虫框架</div> <div>Star156</div> <div>Homepage</div>
<div>XXL-EXCEL Java对象和Excel转换工具</div> <div>Star128</div> <div>Homepage</div>	<div>XXL-HEX Web Api框架</div> <div>Star50</div> <div>Github</div>	<div>XXL-SEARCH 搜索应用参考示例</div> <div>Star39</div> <div>Github</div>

Thanks & Q/A

Q/A