

Tencent 腾讯 | CSIG
云与智慧产业事业群

业务全链路监控实践

业务应用全景监控&链路追踪



个人简介



张加浪

腾讯云 高级工程师

腾讯云 监控解决方案架构师

2017年加入腾讯，一直从事AIOps相关研究和开发工作。腾讯云智能监控负责人，负责腾讯云内外部智能监控系统平台建设，支撑了QQ、空间、微视和腾讯云的CVM、腾讯会议等上百个产品的智能监控。专注于结合公共技术(AI、大数据处理)与监控特性为业务建设智能化监控产品。

目录

1. 应用监控现状
2. 应用场景&实践
3. 客户案例

云原生应用的特点

开发模式

Waterfall



Agile



DevOps



系统架构

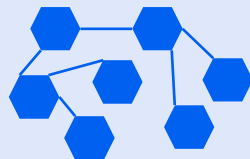
Monolithic



N-Tier

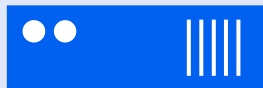


Microservices



部署模式

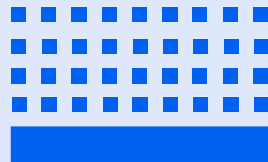
Physical Server



Virtual Servers



Containers



基础设施

Data Center



Hosted



Cloud



- 效率要求越来越高

随着DevOps模式的普及，规划、开发、测试、交付的效率越来越高

- 系统更加复杂

架构从开始的一体化到分层模式，再到现在的微服务架构模式

- 环境动态性增强

容器化的部署模式动态性增强，每个实例的生命周期变得更短

- 上下游依赖更多

云原生应用依赖云上的各类产品，上下游变得更多

服务监控覆盖

服务应用监控(业务保障)

业务监控

指标: 在线人数、订单情况等
维度: 产品类别、区域等

服务监控

指标: 服务请求量、失败率、耗时等
维度: 集群、服务模块、接口、地域等
[包括服务框架如Spring MVC等]

中间件监控

对象: Kafka、Redis、ElasticSearch、Mysql等
指标: 使用率、容量、失败[堆积]等

资源监控

对象: 服务器、TKE等
指标: CPU使用率、内存使用率、健康状态等

基础监控(云化保障)

应用监控现状

应用层面



用户很难准确看到业务应用全局



- 微服务模式开发盛行，架构逻辑被拆分更新变得更复杂
- 多语言/多框架使得同一套标准监控变得不可行
- 服务请求链路明确，但监控覆盖不全，存在断点
- 应用质量标准未能跟上其业务速度，部分还停留与纯日志形式

数据层面



数据孤岛，看不见拿不到，无法有效应用



- 数据“孤岛”，找数据难，拿数据更难
- 数据通道不畅，无法及时获取数据
- 数据质量参差不急，难以融合，难以使用
- 数据未能量化成对应业务场景，数据无法使用反而成为负担

监控层面



监控无法整体实现，服务质量没保障



- 监控能力缺失，难以实现主动场景监控，通常都是被动响应
- 监控孤立形成不了端到端，以致排障难，根因定位不了
- 告警不易触达，导致一些监控告警未生效
- 服务性能质量量化难，服务治理与优化难落地

应用监控痛点

服务模块隔离，难以全面覆盖 1

- ✓ **不愿覆盖**，为敏捷开发，业务模块开发技术、框架、模式独立，不愿共享。
- ✓ **不敢覆盖**，故障定责，模块缺陷或不足不愿暴露。
- ✓ **不能覆盖**，服务模块定位不同，有的聚焦日志，有的聚焦指标，有的聚焦探测

监控质量不高，难以主动保障 3

- ✓ **准确率、召回率低**，基于人为经验聚焦独立模块配置告警规则，业务更新迭代后监控告警准确率、召回率下降。
- ✓ **告警风暴**，服务更新迭代老的监控不敢下，不断激增，最终形成告警风暴。



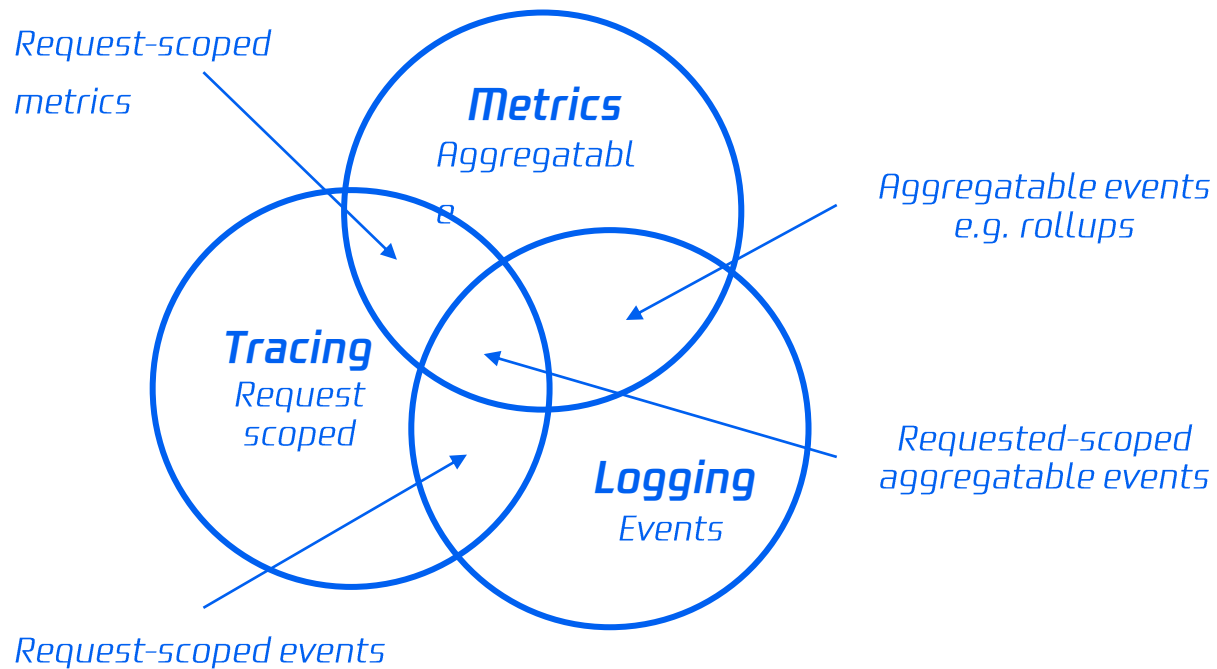
2 定位排查难，误导信息多

- ✓ **信息多而杂**，为尽可能主动收集日志或指标信息，导致过度记录，真实异常却又无从下手。
- ✓ **根因找不到**，监控未全覆盖，监控有了关联不上。
- ✓ **技术孤岛**，语言、框架、监控非标，各成一套很难统一。

4 非标、不全、不准，服务治理难

- ✓ **治理无依据**，非标、不全、不准的监控或观测性现状，导致服务治理没有依据无从下手。
- ✓ **治理无动力**，业务政治通常都有量化目标，空谈优化很难落地。
- ✓ **治理无工具**，业务故障需分析后及时处理，缺少有效主动监控与分析工具，只能被动响应与处理。

应用的可观测性



指标

衡量应用系统当前的状态，信息量少，可通过添加维度来添加额外信息，相同维度之间的指标可聚合〔累加、平均〕，通过指标告警可以快速发现异常

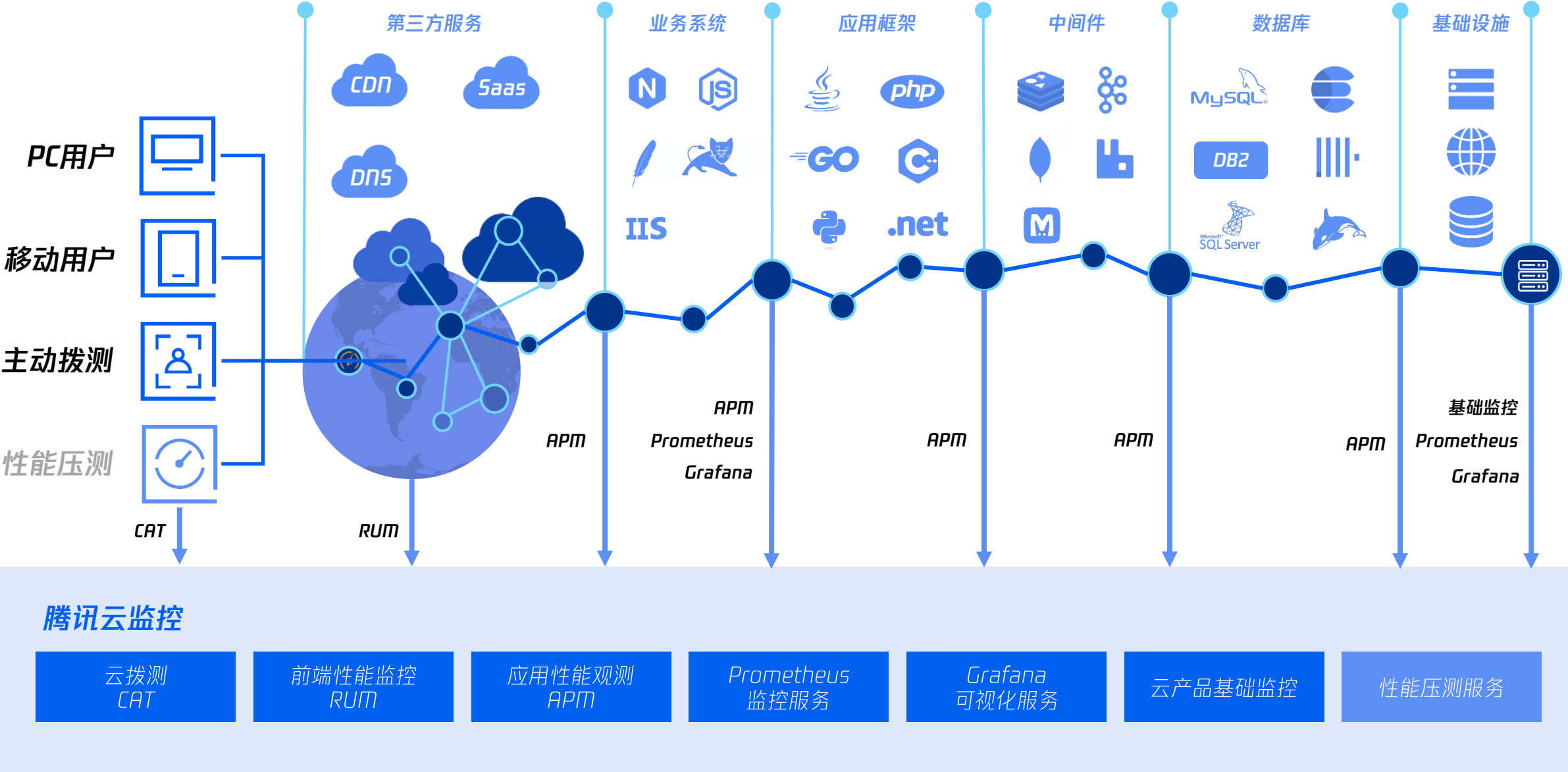
链路

一个请求从接收到处理完毕整个生命周期跟踪路径，相比指标多了请求相关的信息，通过排查链路可以快速定位问题

日志

应用运行产生的事件或程序执行过程中产生的日志数据，可以详细的解释系统运行的过程，通过应用的日志来排除故障

端到端的一体化监控解决方案概况



目录

1. 应用监控现状
2. 应用场景&实践
3. 客户案例

链路应用使用角色

研发

- 掌控应用全景：通过全链拓扑框架做应用改造或分析
- 快速发现与定位问题：及时主动发现并分析定位处理问题
- 代码深度优化：基于线上实际实时性能链路数据及详情做服务深度优化
- 跨团队合作：大业务尤其是微服务框架下清晰的链路有助于团队联合开发

运维

- 掌控应用全景：结合业务整体流量、性能情况，服务依赖关系做保障
- 微服务治理：量化业务链路“短板”，主动治理优化
- 服务日常运维：发布变更及时观测，异常问题主动发现
- 提升业务质量：针对性量化客户操作质量并在优化或业务反馈

领导/业务

- 掌控应用全景：通过全景知道业务运行健康情况
- 量化业务质量：基于量化数据可清晰知道业务质量与客户影响情况

应用场景

问题发现与定位

服务异常链路上报，也可主动策略监控告警，量化异常趋势。基于详细调用链与异常明细可分钟级定位到问题。



DevOps集成与强化

在研发、测试和运维团队中统一工具和考核指标，与DevOps工具链的集成提升效率。



应用全景观测

业务变更或活动，可主动基于应用全景做护航保障。也可实现业务质量的量化与优化。



投诉与定责

先于用户投诉发现问题，快速给出问题结论，明确问题责任范围，变被动为主动。

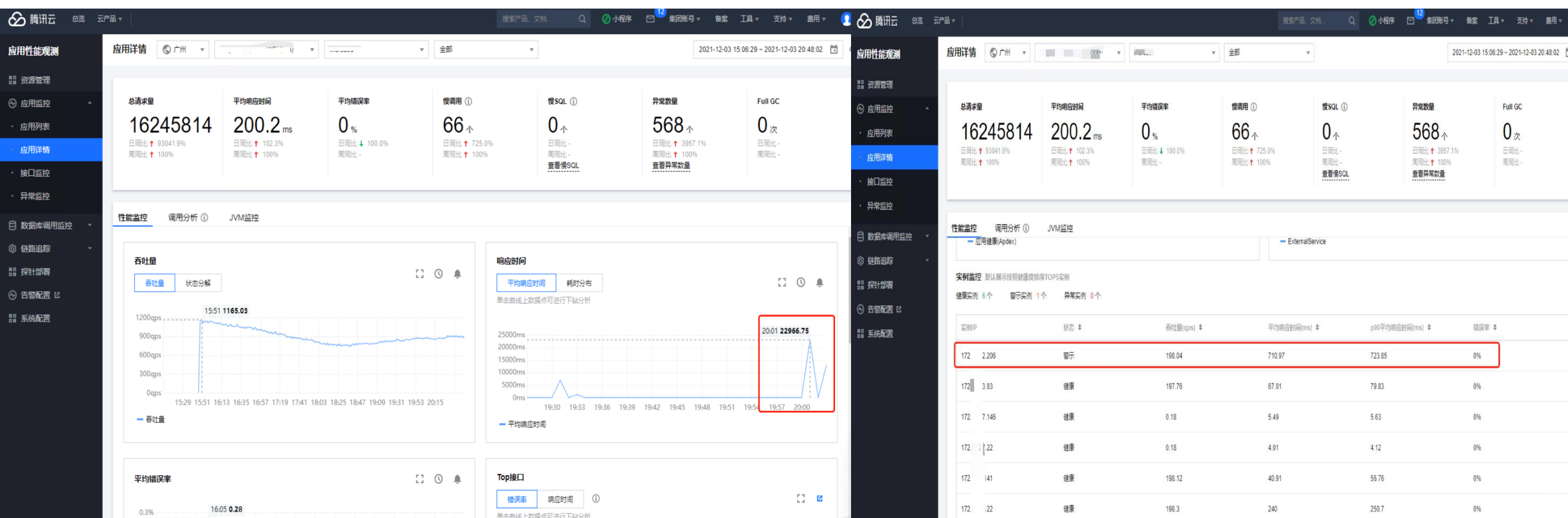


服务治理

基于全局应用链路可有效分析出业务服务“短板”，然后针对性做优化以及优化后效果对比等。



实践场景1-变更异常










业务发布灰度变更时，应用大盘上出现了响应时长突增的情况。看到这个问题快速查看了服务器实例情况发现有实例耗时异常汇聚，定位到是业务灰度变更引起，做了服务回滚和问题修复。

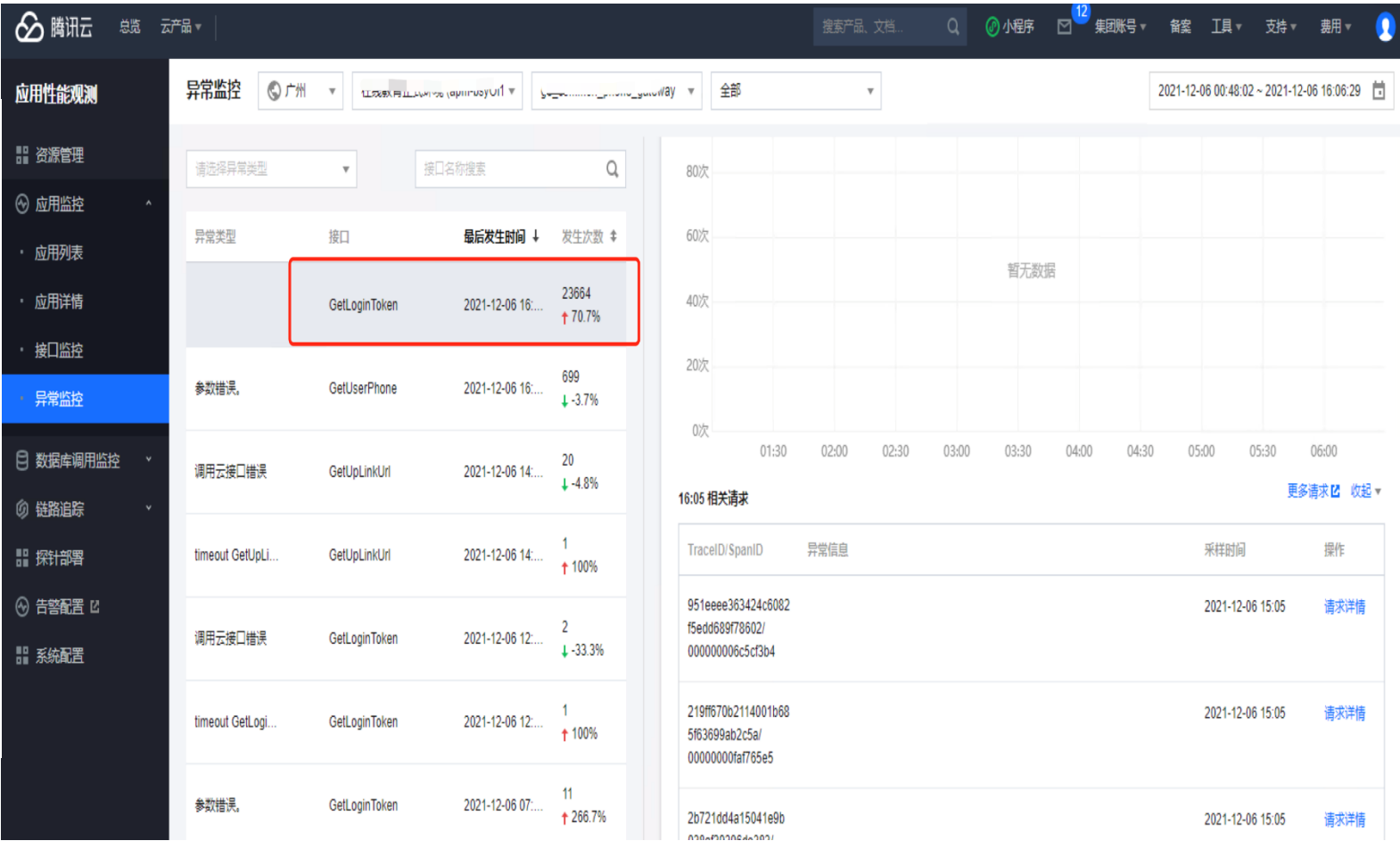
实践场景2 - 异常监控告警

「应用性能观测」持续触发告警通知

尊敬的腾讯云用户，您好！
您账号（账号ID：）下的「应用性能观测」告警持续触发，请关注！

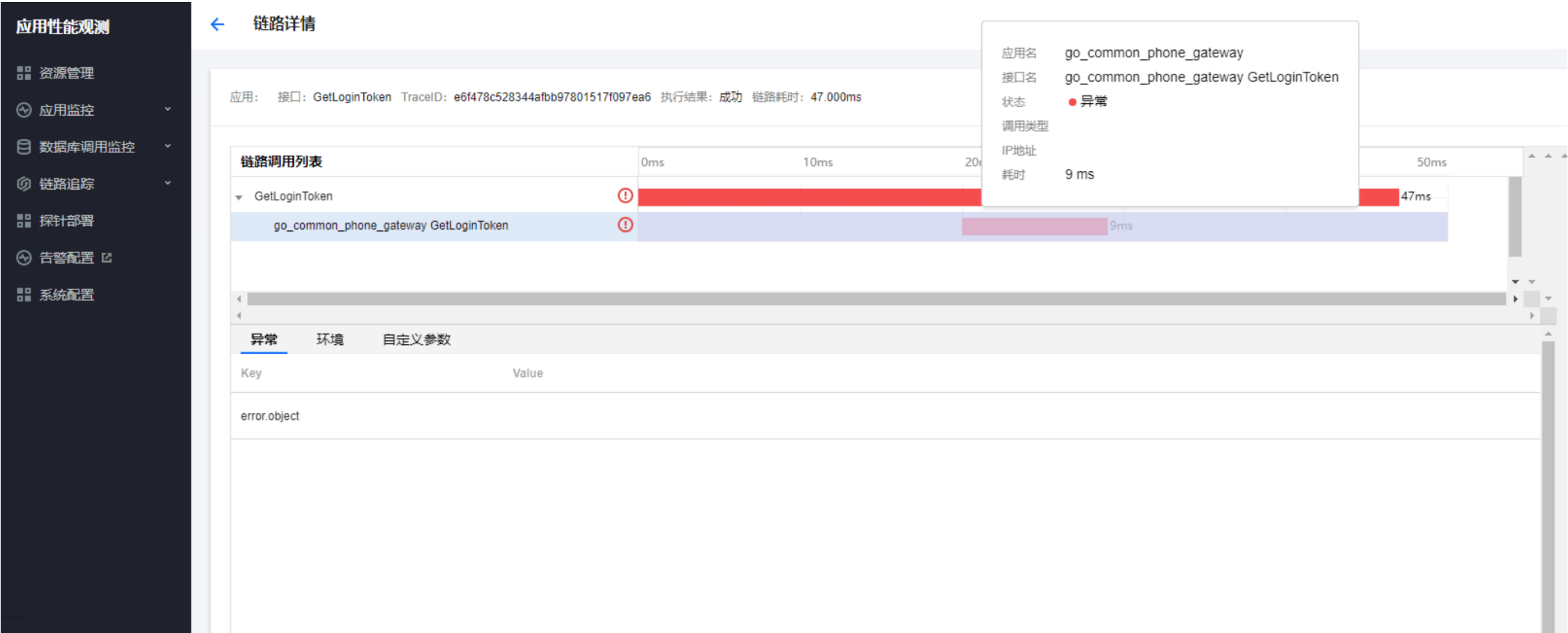
告警内容：性能指标 | 错误率
告警对象：业务系统=tap自监控，应用=tap调用角色=服务端
当前数据：错误率)
告警策略：请求服务接口质量
触发时间：2021-- (UTC+08:00)
持续时间：分钟

此致
腾讯云团队



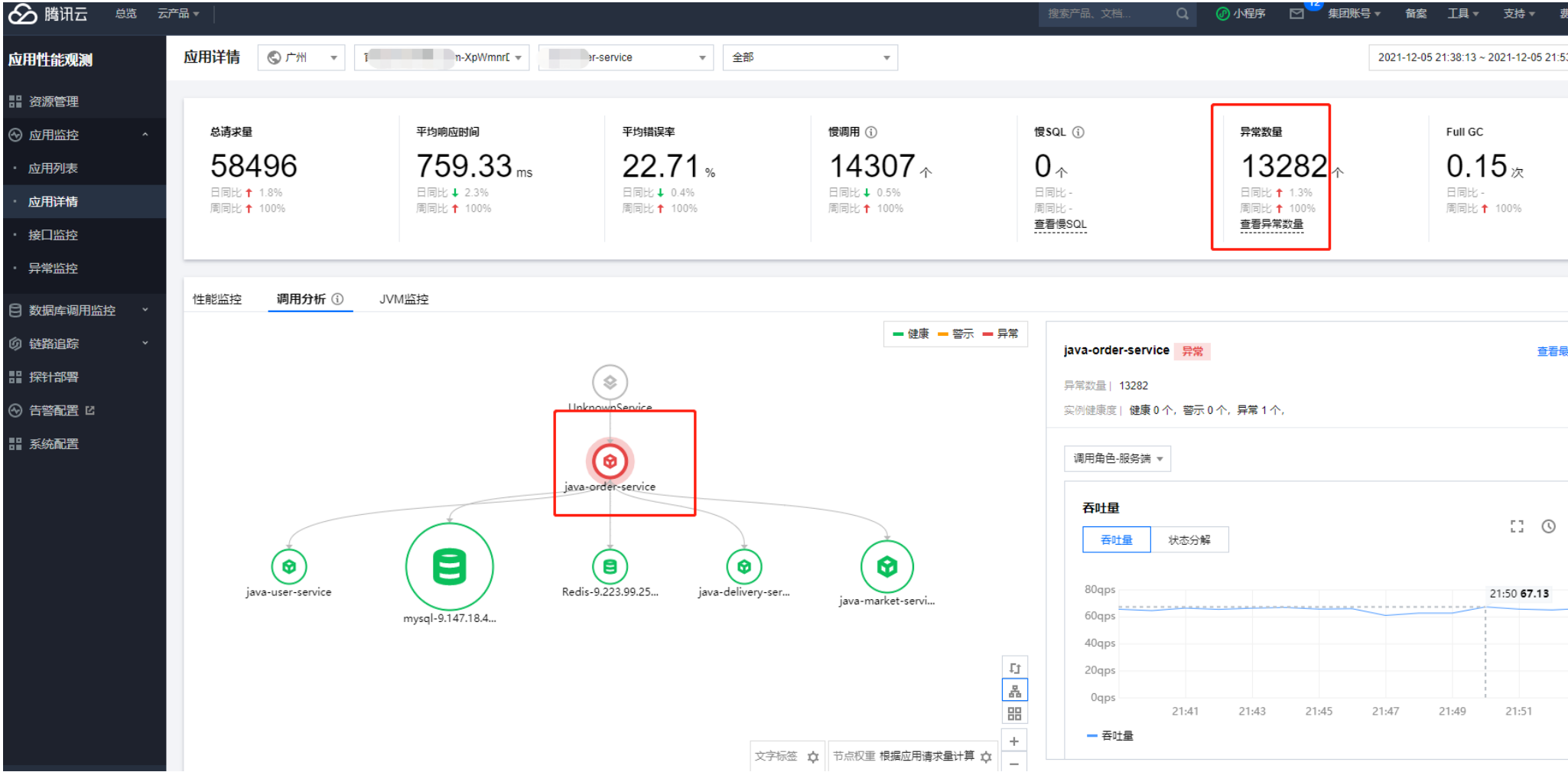
业务指标异常时可由主动设置的策略触达告警，然后具体在定位的异常接口查看具体情况

实践场景2 - 异常监控告警



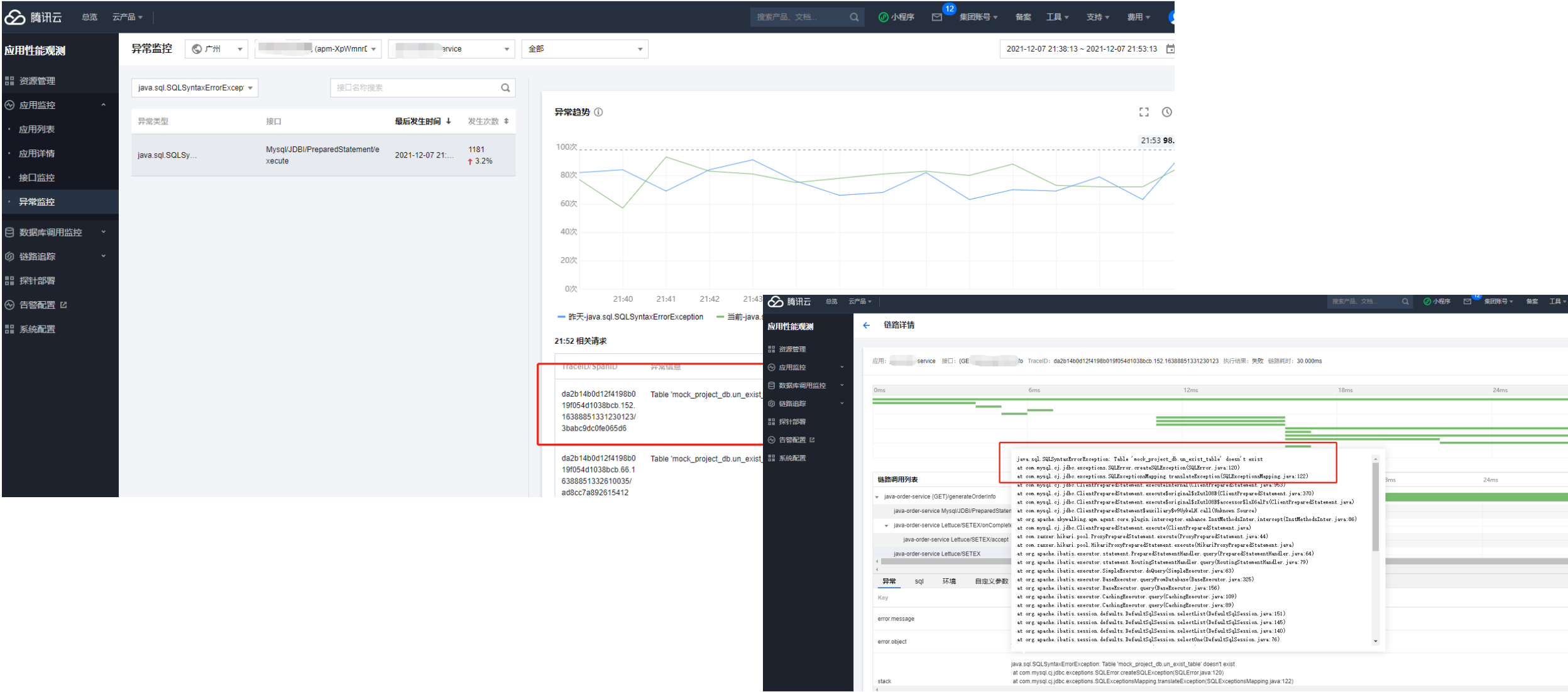
分析定位是获取登录`token`时，带入参数异常

实践场景3 - 服务SQL异常



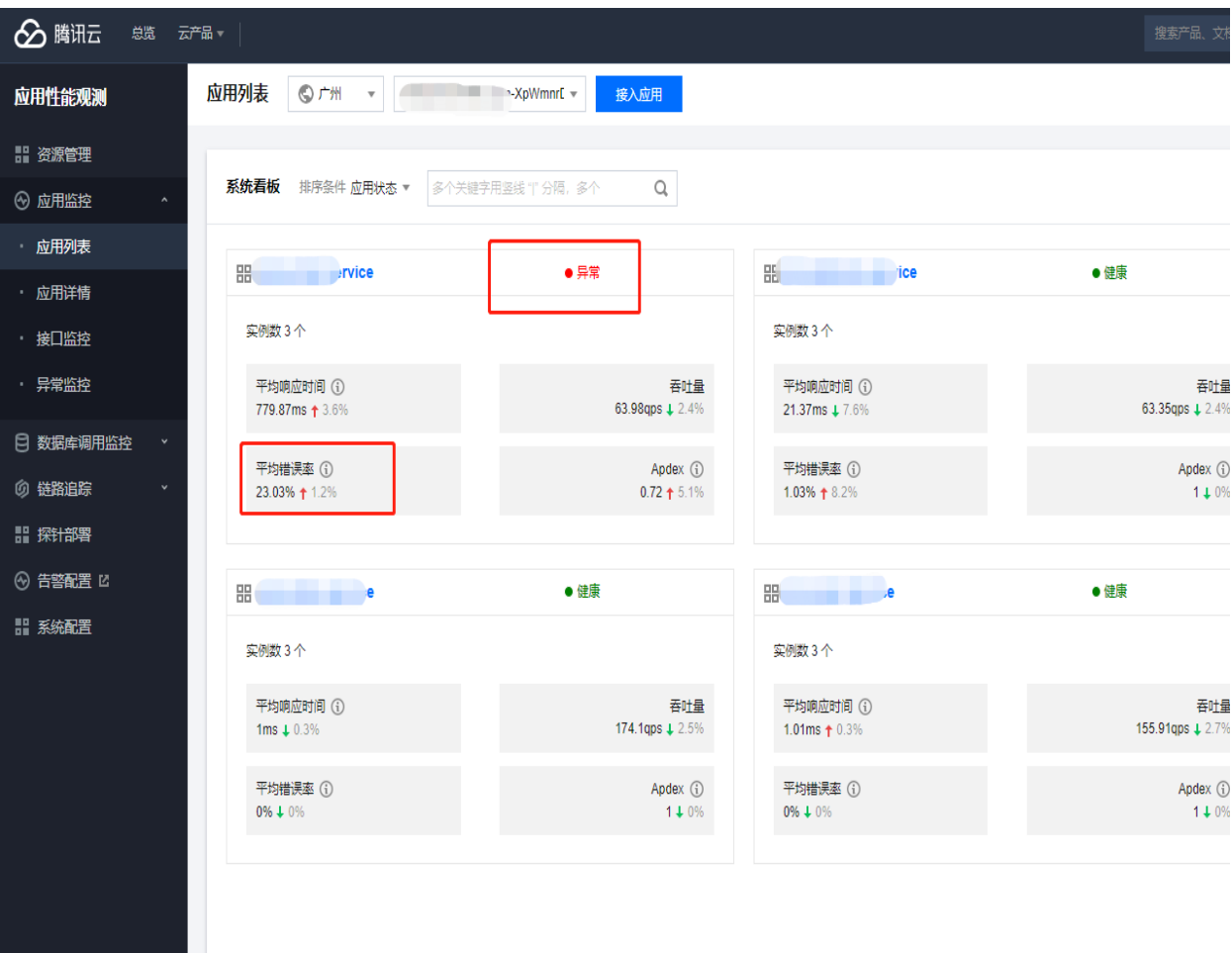
业务告警保障，在应用拓扑中显示异常应用，并且异常数量有明显上升，通过具体应用异常下转分析

实践场景3 - 服务SQL异常



具体查看异常接口情况，分析主要是接口异常，验证确定是后端服务上线数据库表变更，但服务配置里没有更新导致表找不到

实践场景4 -DevOps集成与强化



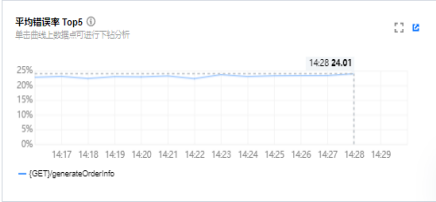
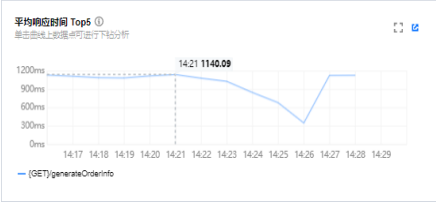
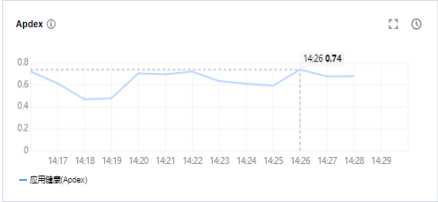
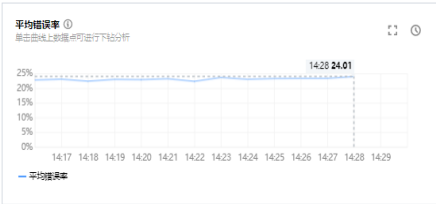
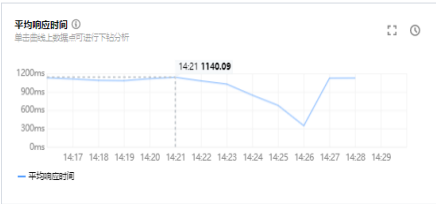
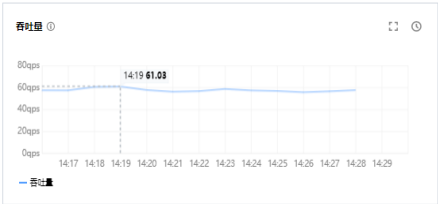
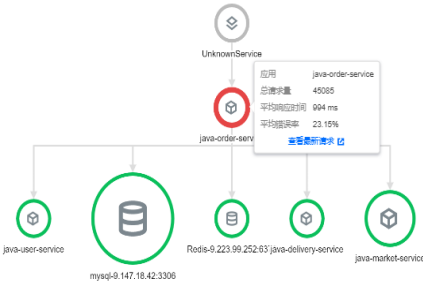
DevOps
发布、变更、流程管理、流量迁移、监控报警等

可观测性&监控覆盖
拨测、前端监控、应用性能监控、基础资源监控

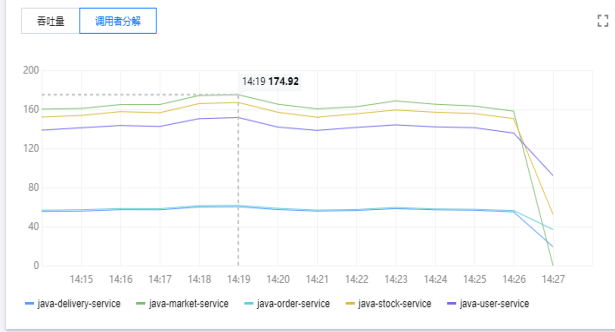
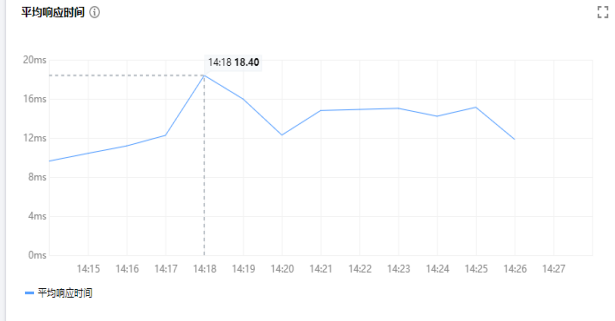


服务DevOps发布变更部署，发版后通过应用大屏发布观察可及时确认变更质量，快速发现异常回滚

实践场景5 - 服务治理



数据库概览 官方Demo环境 mock_project_db 全部 15分钟



调用请求	调用应用	耗时(ms)
select * from un_exist_table limit 1	java-user-service	19.01
select * from un_exist_table limit 1	java-market-service	18.35
select * om un_exist_table limit 1	java-user-service	16.88
select * om un_exist_table limit 1	java-order-service	15.99
select * from mock_project_userinfo where id = ?	java-delivery-service	14.7

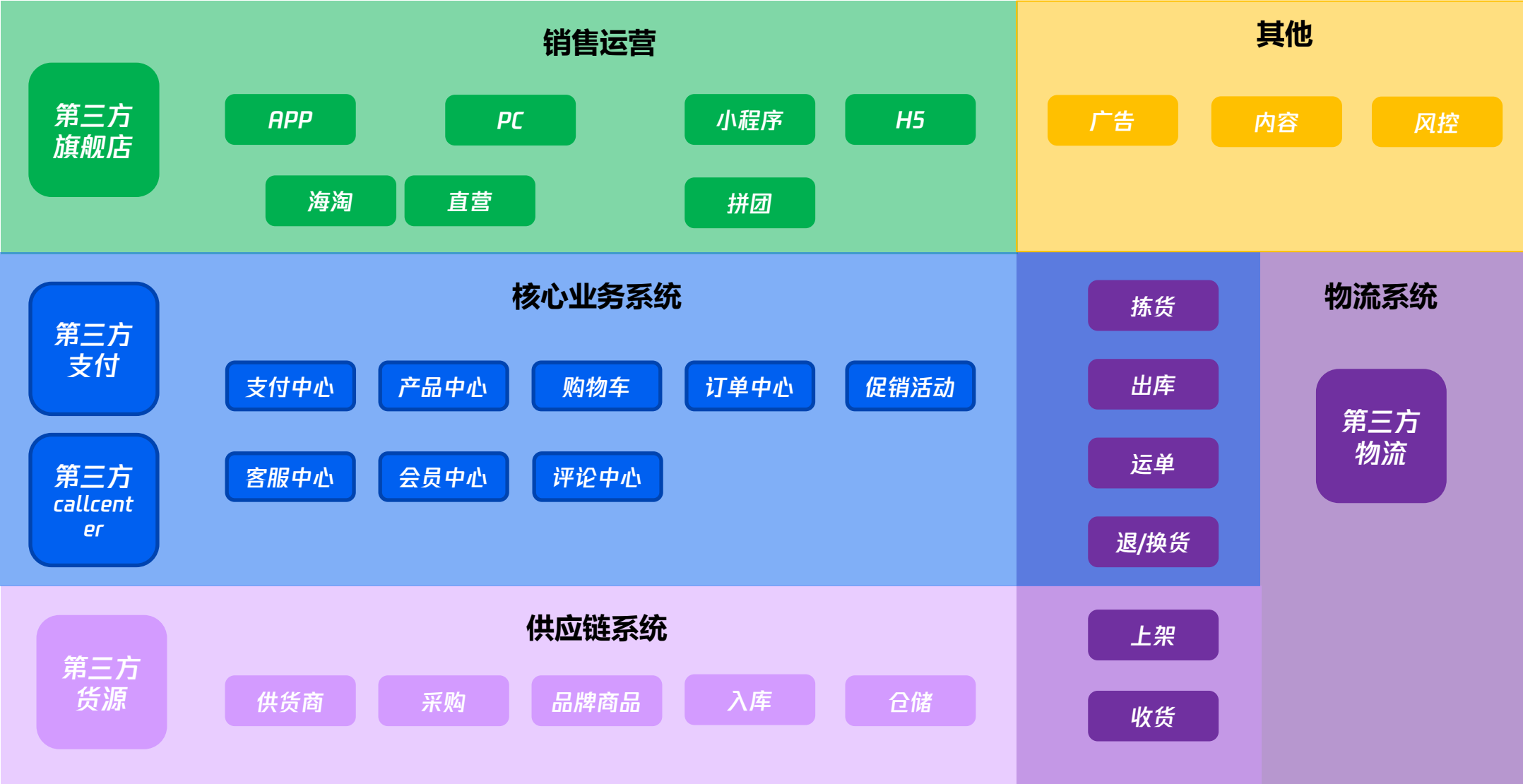
调用应用	调用次数(次/秒)	查询次数(次/秒)	写入次数(次/秒)
java-market-service	128587		
java-stock-service	122354		
java-user-service	111116		
java-order-service	45457		
java-delivery-service	44520		

微服务框架主动关键链路指标量化，可及时发现服务框架下的“短板”，并实时看到治理优化效果。业务逻辑SQL主动分析，慢查询TOPN，异常SQL的TOPN调用者快速定位。

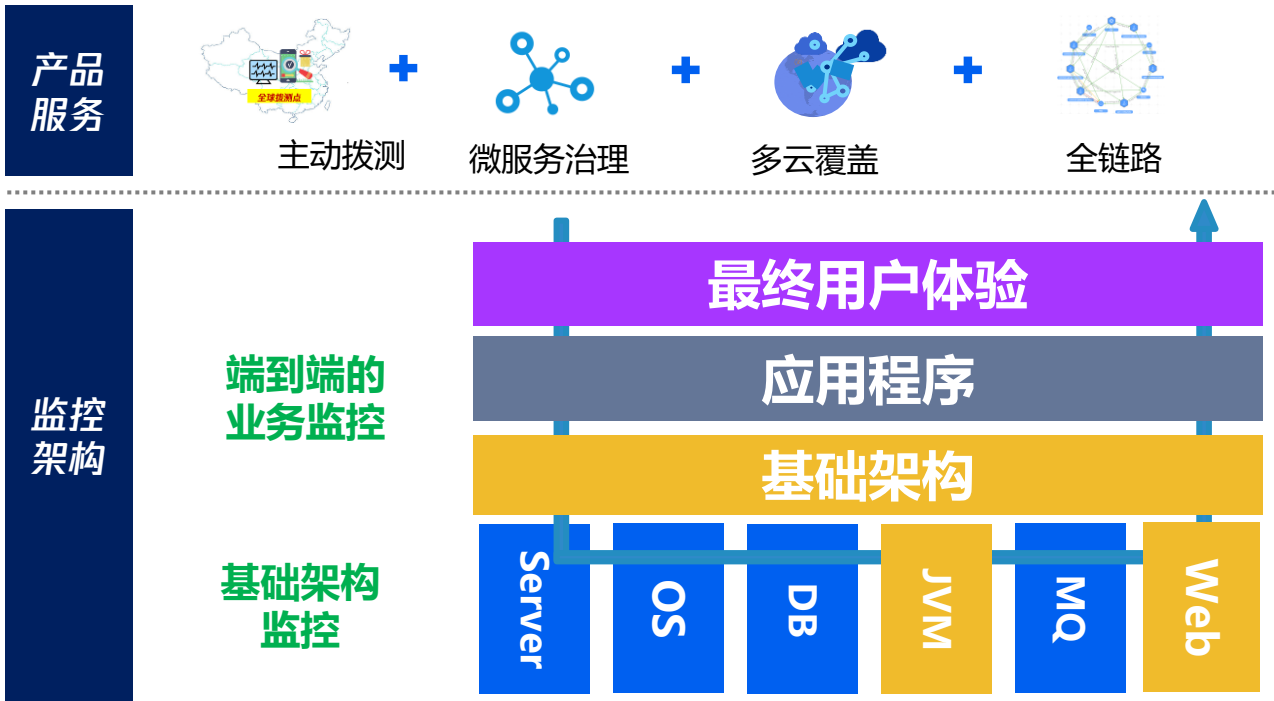
目录

1. 应用监控现状
2. 应用场景&实践
3. 客户案例

某电商客户案例

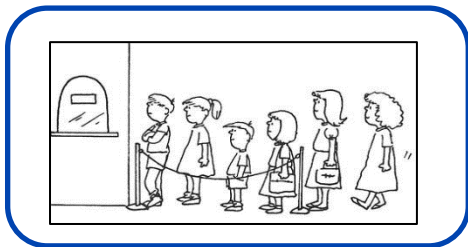


某电商客户案例

背景与需求		解决方案		实施效果	
背景	<ul style="list-style-type: none">网络情况监控，有效知道终端客户量化体验，并能量化运营商效果；客户在上云过程中希望能够有效保障好IDC与云的业务质量，当发现异常时可有效迁移；服务上云是业务优化的一个机遇，希望能够在过程中针对全链路做分析和针对性优化；	产品服务		端到端的业务监控 基础架构监控	<div>全链路监控覆盖</div> <div>实现了业务从终端体验拨测到后端具体逻辑模块的覆盖</div>
	需求				方案优势
<div>故障定位加强</div> <div>单点分析变成全链路分析，覆盖了的场景可实现分钟级定位原因</div>					

某银行客户案例

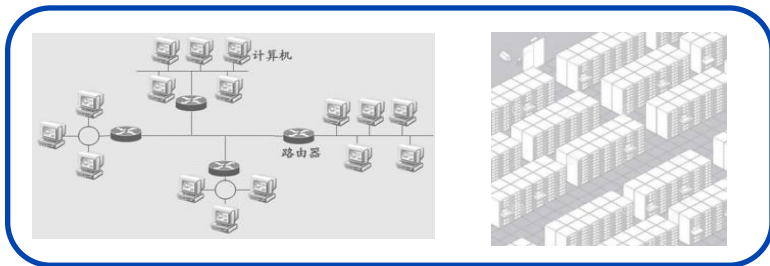
终端交互



用户终端体验

服务质量量化

后端逻辑



问题主动发现

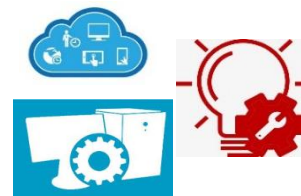
快速分析处理

数字化服务

内部服务
解决问题



生成产品
产生价值



某银行客户案例

背景与需求

背景

- 客户终端APP为主要用户入口，因涉及模块多且交叉开发希望能做优化；
- 内部全链路监控，主动监控问题，并能全链路分析处理问题；**
- 原服务中已有自建使用skywalking方式链路监控监控服务需非侵入式并且性能稳定。

需求

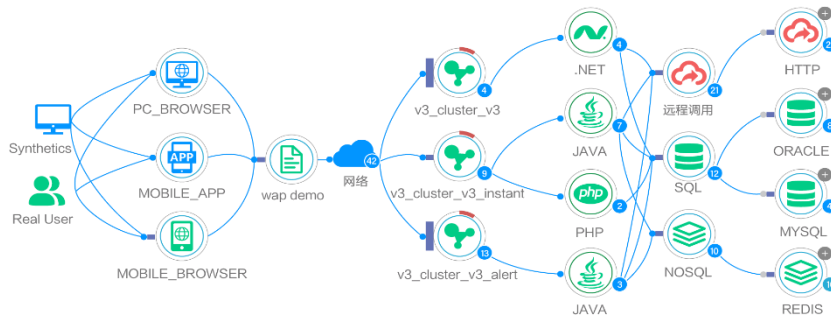
- 微服务性能分析：**链路及量化的性能指标以及Apdex评分；
- 前端性能优化：**针对APP上不同的业务模块所涉及的元素做深度性能分析；
- 非侵入式&开源支持：**在服务上引入非侵入式探针，兼容使用skywalking方式。

解决方案

产品服务



监控架构



方案优势

- 新老方案支持：**新上线服务使用TAW的探针采集，老服务中原有的skywalking可直接兼容；
- 快速标准性能分析：**前端、后端服务性能场景化直接量化，并且展示是支持Top排序；
- 端到端问题分析：**服务请求前端到后端覆盖，有问题可快速实现从前到后详细跟踪分析；

实施效果

全链路监控覆盖

实现端到端的全链路覆盖和性能量化

新老服务整合

新服务的链路
与老的（skywalking）
链路整合

服务性能优化

通过实际接入采集的性能数据找到具体的性能短板并针对性优化

Thanks

腾讯云监控公众号



腾讯云监控微信好友

