B站变更防控的设计与实践

2024年4月

https://sre-elite.com

个人介绍

SREelite

刘 昊 / 平台工程负责人

- 从业十余年，专注于运维效能、质量运营等领域。
- 2017年加入哔哩哔哩，先后负责了B站运营研发、中间件研发和SRE体系等方向，构建了B站的统一作业&流程&鉴权服务，主导了数据库&缓存相关中间件的自研落地。
- 目前主要负责SRE体系化建设和人员转型培训，设计落地应急响应、变更防控、蓝军演练、运维数据资产和资产成本等系统，持续优化业务稳定性、提升人员效率和降低资产成本。

目录

CONTENT

SREelite

01

变更领域思考

SREelite

什么是熟悉又陌生的事情？

SREelite

熟悉

- 走上软件这一行当，就踏上了变更这条不归路
- 构建、发布、创建集群、扩容、缩容、升级、打补丁...
- 变更流程：计划、执行、验收
- 变更三板斧：可灰度、可观测、可回滚
- 人人都是变更小能手，经手的变更没一千也有八百

变更

陌生

- 变更，一个软件领域无处不在的东西，从来没有像对待软件一样对待过它
- 当我们聊变更时，我们在聊什么？提交的变更计划书？撰写的SOP？执行的变更过程？
- 我待变更如初恋，变更虐我千百遍。在变更手上翻过的车，没十次也有八次了。
- 每个平台都在认真构建变更，从手动到自动，但是变更的问题并没有被杜绝
- 变更难道是薛定谔的猫？

为何在当下去被聚焦？

SREelite

鞭子抽疼了

外　驱　力

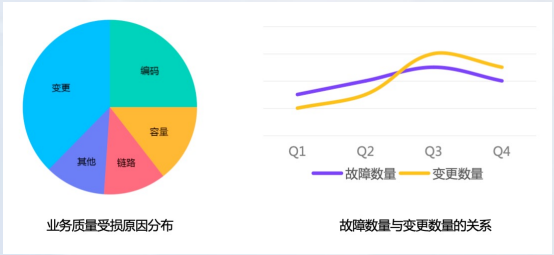
- 故障数成上升趋势
  - 降本后遗症显现
  - 运营力度更深入
- 故障原因越发奇妙
  - 缘分到了
  - 愚蠢所致
  - 莫名奇妙
- 变更在故障原因占比是绝对大户
  - 60-75%

内　驱　力

- 总被动应急，精神持续紧张，堵不如疏
- 稳定性能力版图的缺失
  - 整体战略中心在往前移
  - 补足稳定性领域的能力缺失
- 需要提振信心，增强成就感
  - 从被动防御要转向主动进攻
  - 工作中心需要从故障后转移到故障前
- 原有的稳定性推进项目ROI在逐渐变低
  - 过往的稳定性建设逐步进入下半场
  - 持续、快速的擦干净屁股，难度大，成本高

不想起夜了

图源：https://divante.com/blog/10-companies-that-implemented-the-microservice-architecture-and-paved-the-way-for-others/



变更防控	混沌工程	风险治理	应急响应	可观测	智能预警
	前置保障			后置保障	

围绕变更整活的思路

SREelite

一个中心 通过管理变更来降低故障发生率和影响面，是提升稳定性的一条可持续、高ROI的路线

行业经验实践

ITIL

ITIL V2 强调变更管理的流程和控制，着重于确保变更的有效性和最小化对业务的影响。它强调了变更管理流程的重要性，并提出了 Change Advisory Board（CAB）等概念来审查和批准变更请求。

ITIL V3 强调了与持续交付和敏捷方法的结合，以更好地应对变化和客户需求。它引入了新的概念，如变更评审（Change Evaluation）和变更评估（Change Evaluation），以更全面地管理变更的影响和效果。

ITIL 4 强调了灵活性、适应性和持续改进，引入了更多与现代技术和方法相关的概念，如敏捷、DevOps 和持续交付。变更管理不再是独立的流程，而是作为整个服务价值系统的一部分。

ITIL的经验反思：单纯靠管理手段并不能有效管理起变更，更多管理的是人

+ 通用经验实践

减少数量

- 秉承不做就不错的原则
- 通过降低变更的数量来减少由变更产生的风险和导致的故障数
- 效果不明显，也不现实）

提高质量

- 要么不做，做就做最好
- 减少变一步看一步
- 减少一把梭allin
- 减少闭着眼变更

通用的经验反思：软件领域，控制数量未必有效；提高变更质量，在全集团内又会变成人的管理问题，难以得到有效的整体性提升

= 基于红线和文化的软件工程建构

数字化

- 软件问题，用软件工程的思维去解决才是正途
- 变更作为一个独立的领域，需要专项解决
- 统一收录变更元信息，建设变更的元数据中心
- 统一托管变更行为，实现统一的中心化管控
- 转化规章制度到系统策略，通过中控实现策略在全变更域的控制
- 基于收录的变更行为数据，围绕CMDB+Trace+时间建立南北、东西、前后的全向分析能力
- 引入统计+AI能力实现智能化

文化宣导：文化宣传周、新人培训、平台引导

立规立矩：盯着才有用：建设变更法（基线/惩罚）

存在的风险点：数字化手段和法规文化要循序渐进建设，相辅相成，避免单一建设脱节

再看价值

SREelite

战略价值 通过提升变更的可控性，降低由变更导致的故障数量，尽而提升业务稳定性

价 值 定 性

- 标准化能力落地
  - 定义/描述标准
  - 流程/过程标准
- 技术框架成型，建立体系管控能力，具备健全的运营手段
  - 输出一套标准/完整的解决方案
  - 点线面:从解决单一点的问题,到可以解决面的问题，通过统一化管理，实现变更源平台、各职能角色、各IT对象在一个平面的管控运营
  - 形成了落地和推进的有利手段，让过程化管理有明确载体，并且能渗透进各个实际的变更流程中去
- 补足可观测能力
  - 传统均围绕：架构观测（CMDB）、链路观测（trace）、服务观测（指标/日志/profile）
  - 容易缺失的一环：变更可观测
- 技术普惠性
  - 过往只有核心大平台在变更建设方面的有足够的时间和成本投入
  - 通过将技术框架、技术能力抽象独立出来，让更多的中小型平台获利

价 值 定 量

- 故障数量减少
  - 引入变更管控之后，拦截掉的故障数
- 故障影响缩短
  - 统一变更数据中心的建立，为故障的成因分析带来数据基石
  - 通过变更自身数据、变更部署对象静态拓扑、变更对象调用动态拓扑的分析，快速锁定变更问题点，缩短缩短影响时长
- 故障防御能力提升
  - 核心场景的防御覆盖度（百分比）
  - 全场景的防御水位（分位数）

02

变更体系设计

SREelite

https://sre-elite.com

从挑战而生的方案

SREelite

核心思考模型

用户模型分析场景：用户价值 = 新体验 - 旧体验 - 替换成本

交易模型分析场景：相对价格 = (直接成本 + 交易成本) ÷ 效用组合

技术

- 变更平台多：CD平台、各Paas类平台、配置平台等
- 职能角色多：研发、测试、SRE、SYS等
- 变更对象多：服务、容器、集群、实例、服务器等
- 改造成本过高！

解法

- 统一变更模型
- 统一变更过程
- 变更分级，适应性防控
- 建设变更领域生态，提升能力丰富度
- 高优场景切入，快速拿结果
- 持续推广运营，量变引起质变

建设一个独立领域的平台

文化

- 变更风险意识薄弱
- 企业红线不够红
- 在平台建设中，一味追求效率，没有沉淀相关变更方法论

解法

- 游说高层，认识到变更领域建设的长效收益
- 完善企业管理制度和条例
- 通过各类活动（安全生产周/安全生产月），增强宣讲力度

组织建设

体系化解决方案

SREelite

变更体系化

技术支撑

产品能力

概览大盘 | 托管接入 | 检索订阅 | 防控配置 | 变更分析 | 数据可视

核心能力

变更托管	变更感知	变更防控	价值可视	开放能力
平台场景	检索订阅	防控检查项	纬度汇总	SDK/MSG
OpenAPI	变更关联	同步/异步阻断	下钻分析	变更预检
消息队列	影响分析	多变更联动	数据通晒	变更风险

变更实例模型

变更对象 | 变更环境 | 变更内容 | 操作人 | 变更时间 | 变更范围 | 变更影响

变更元模型

变更平台 | 变更场景 | 变更防御项 | 工单流 | 批次流 | 前后岗哨

数据支撑

服务树 | CMDB | 可观测

技术落地

研发过程

- 代码变更
- 配置变更

基础架构

- 流量调度系统
- 资源管控系统
- 作业调度系统
- 常态维护系统

业务配置

- 运营活动
- 开关变更

跨领域赋能

应急响应

- 变更定位
- 变更溯源

攻防演练

- 变更故障模拟
- 异常变更模拟

变更大模型

- 变更领域问答
- 场景缺陷分析
- 场景防控建议

组织文化

组织架构支撑

- 安全生产委员会
- 稳定性负责人

制度建设

- 安全生产要求
- 安全变更规范
- 违规处罚

文化宣贯

- 培训考试
- 专题分享
- 活动宣传

03

落地实践经验

SREelite

https://sre-elite.com

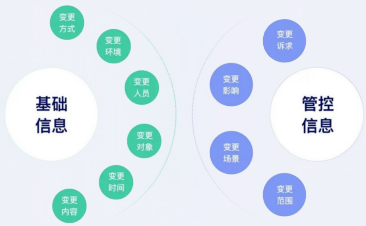
变更定义

SREelite

通过建设统一的变更描述模型和过程模型，拉齐人与系统的认知，并通过合理分级，确保有效发力。

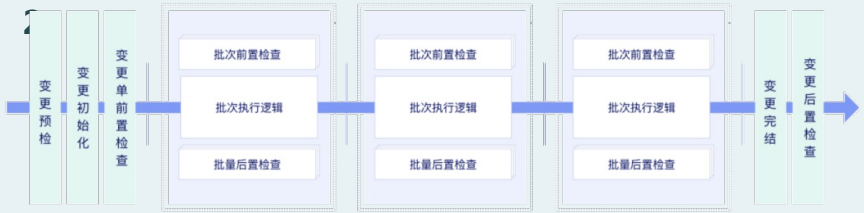
统一的变更描述模型

- 基本信息
  - 变更来源：用以标识具体的变更平台和平台唯一标识
  - 变更时间：变更具体执行的时间
  - 变更人员：变更实际的操作人
  - 变更对象：包括变更对象类型(应用/数据库/服务器等)和变更对象唯一标识
  - 变更环境：变更所生效的环境，比如UAT/PRE/PROD等
  - 变更内容描述：对变更操作的详细描述，包括release notes 和详情描述，对于传达变更目的和实施细节非常重要。
- 管控信息
  - 变更目标：即期望通过变更来达成的目的，包括日常迭代、BUG修复、故障处理等;
  - 变更场景：指具体的变更动作，包括迭代发布、配置发布、服务器上下线等;
  - 变更范围：即变更在线上环境的生效范围，包括机器分批生效模式以及流量比例分批生效等;
  - 变更影响：指本次变更影响的服务和基础设施



统一变更过程模型

- 一个变更的生效往往是基于一定范围和流程的，围绕这个过程可以建立变更流模型。
- 1、无法拆分的变更流模型：
- 一次性生效模型，适用于某些无法拆分或拆分成本较高的变更场景，例如 DB 配置类型变更。在这种模型下，变更控制防御能力主要集中在配置变更前后。
- 2、逐步生效的变更流模型：
- 按机器分批生效：将变更分为若干批次，在每个批次中逐步将变更应用到不同的机器上。这种方式允许在不同的批次上实施更多的风险管控，以便更精细地监控和评估变更对系统的影响。
- 按流量比例分批生效：允许按照预定的流量比例逐步引导用户访问到经过变更的系统。这种方式同样可以在不同阶段上实施风险控制，适用于需要更细粒度控制用户流量的情况。
-



变更等级分层

- 合理的管控分层，更有利于落地
- 在执行上，有的放矢，有张有弛
- 避免拿起锤子，全是钉子

等级	含义	数量	声明周期定义	使用场景
G0	仅做事件同步，无管控能力	一个	仅有一个事件同步节点	适用于无任何风险的变更，但数据需要提供给相关人员做检索、审计等场景
G1	仅做单节点的变更前后切面管控	一个	生命周期中仅有一个变更节点	适用于低风险变更，无需复杂风险管控能力，仅做事前准入性和事后完整性检查的场景
G2	有完整的变更工单，且工单下关联了至少1个批次的子节点	多个	生命周期分为四个阶段(工单开始>各批次节点开始>各批次节点结束>工单结束)	适用于多数逐步生效的变更，并需要配套进行风险管控的场景
G3	在完整的变更工单感知基础上，增加了变更提单阶段感知	多个	生命周期在G2等级的基础上，前置增加了变更提单阶段	适用于非技术人员或专人代理执行变更，需要系统辅助进行事前风险分析的场景
G4	在变更提单感知的基础上，增加变更无人值守的决策能力	多个	生命周期在G3等级的基础上，变更提单后增加无人值守决策阶段	适用于需要系统进行无人值守代理执行变更的场景

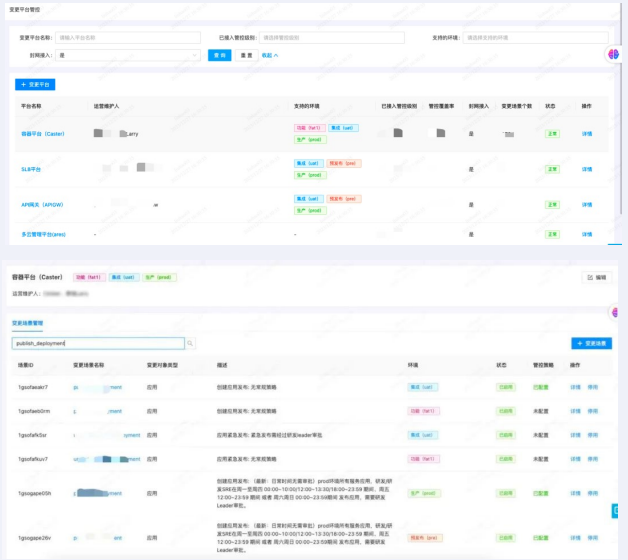
变更托管

SREelite

通过托管变更元信息、行为信息和过程信息实现对于变更的全方位管控，具备干预能力，丰富运营数据。

变更元信息托管

- 录入产生变更业务平台和场景信息
- 平台信息：名称、负责人、覆盖环境、管控覆盖度、覆盖管控级别、描述等
- 场景信息：名称、环境、变更对象、变更内容、管控策略、描述等



变更信息托管

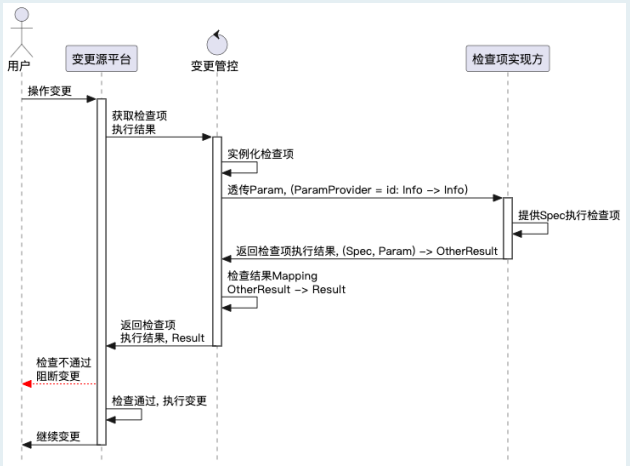
- 变更平台通过OpenAPI、SDK接入管控平台
- 像描述接口定义、类定义一样去描述变成场景，实现变更行为的详细上报。
  - 变更场景的实例化信息
  - 变化前后置信息描述
  - 变化内容对比
  - ...

```
type ChangeScene struct {
    // 归属变更平台
    Platform
    // 关联变更资源类型
    SourceType
    // 基础信息(变更内容)Schema
    ContentSchema
    // 管控信息(批次内容)Schema
    StepSchema
    // 检查项
    []Navigation
    ...
}
```

```
type ChangeControl interface {
    // 变更初始化
    InitChange()
    // 变更结束
    FinishChange()
    // 变更批次开始
    StartChangeStep()
    // 变更批次结束
    EndChangeStep()
}
```

变更过程托管

- 实现变更过程的精细化拆分，拆分前后阶段，拆分批次，拆分批次前后阶段等
- 基于变更单、批次单的前后阶段增加切面的检查项能力嵌入
- 通过检查项实现过程的检测和干预



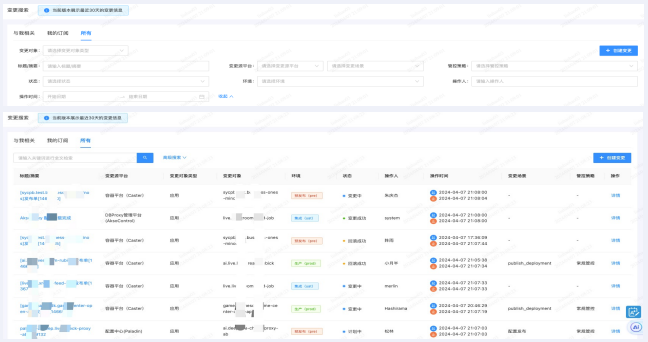
变更感知

SREelite

围绕全域变更信息进行整合，建立分词和关联，通过多种检索和触达手段来实现变更信息在人与系统间的灵活流通。

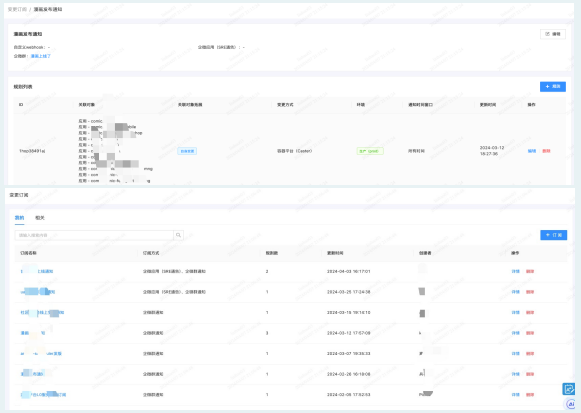
变 更 检 索

- 检索场景
  - 围绕单一对象的时间跨度搜索
  - 围绕单一平台的时间跨度搜索
  - 围绕单变更人的变更行为搜索
- 检索纬度
  - 变更标题、变更状态
  - 变更源平台、变更对象、环境
  - 操作人、影响范围
- 开放能力
  - 全量/纬度变更消息投递



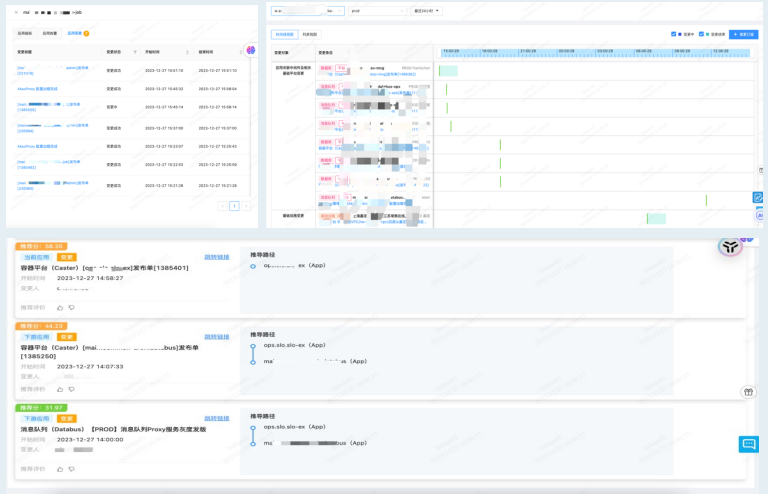
变 更 订 阅

- 订阅能力描述
- 订阅纬度
  - 服务、业务、组织
  - 自身变更、上下游变更
  - 变更平台、变更场景
  - 环境、时间
- 触达机制：个人、群、webhook等
- 开放能力
  - 定向消息投递
  - 接口回调



变 更 分 析

- 检索场景
  - 故障定界
  - 根因分析
- 检索纬度
  - 时间
  - 静态拓扑：部署架构
  - 动态拓扑：Trace调用



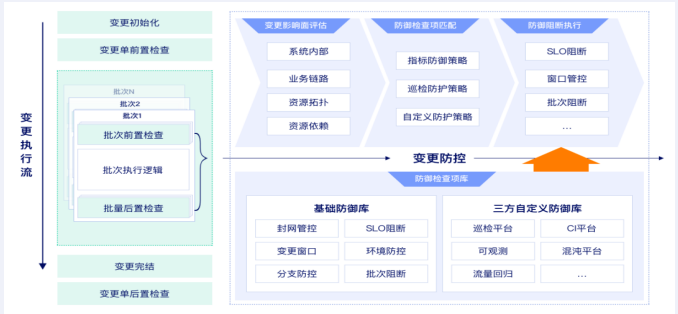
变更防控

SREelite

基于通用的防控技术方案，围绕通用防御能力+自定义防御能力结合的方式，最大化对变更场景进行防御布控。

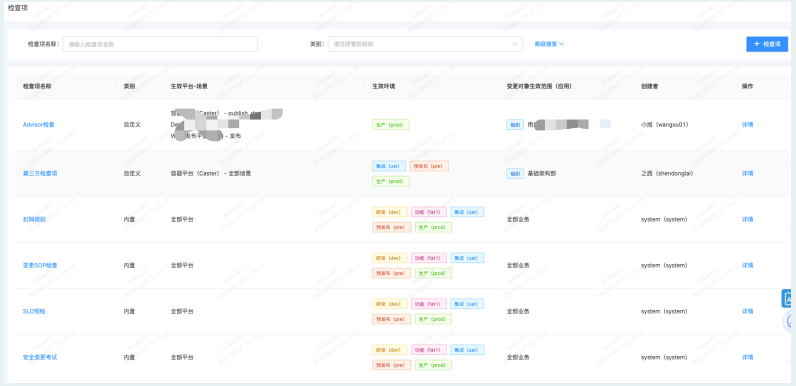
防控技术方案

- 依托于变更流模型的前后阶段
- 链式执行：准入->执行->准出
- 弃用策略
  - 观察者模式：触发检查，反馈检查结果，不做干预
  - 执法者模式：触发检查，反馈检查结果，实时干预
- 防控策略
  - 阻断：阻断变更过程
  - 预警（提示）：提示异常，不做阻断
  - 升级审批：阻断变更过程，并产生审批流
- 熔断策略
  - 忽略：跳过该检查项
  - 阻断：阻断变更过程

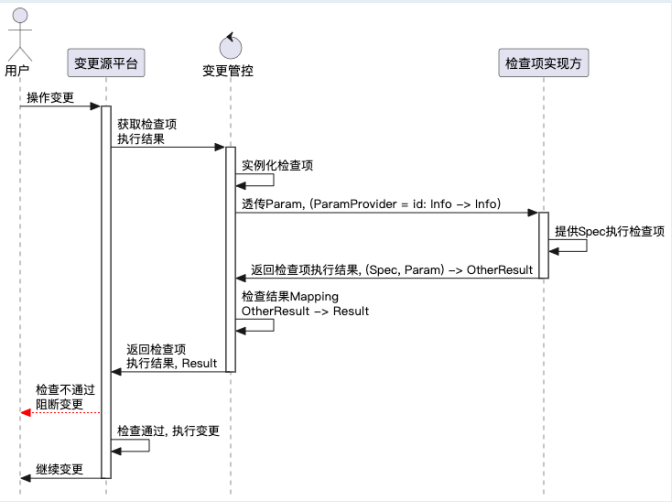


防控能力概述

- 通用检查项
  - 封网拦截
  - SLO检查
  - 批次检查
  - 环境监察
- 自定义检查项
  - 安全扫描
  - 业务代码发布SOP
  - ...



技术交互模式



变更价值挖掘与可视

SREelite

通过多维度视角对变更数据进行挖掘，发掘风险点，明确优先级，可视项目价值。

- 数据分类
  - 覆盖性数据
    - 平台、场景、防控等级
  - 防御性数据
    - 触发防御、防御拦截、拦截效率
  - 运行性数据
    - 变更量
  - 特殊关注
    - 紧急通道、活动封网
- 钻取纬度
  - 平台、场景、防御检查项
  - 组织、业务、应用、人
  - 时间

整体覆盖情况

变更数据使用情况

核心场景
故障阻断率

核心场景+核心检查项目
故障阻断率

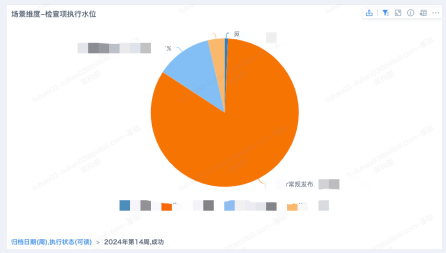
变更导致故障数量占比

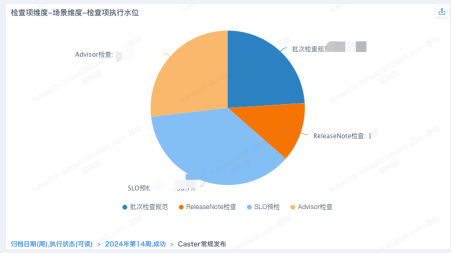
工作台展示

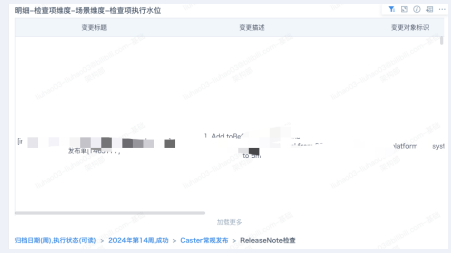
运营面板

检查项面板

钻取分析







演进历程

SREelite

历经三个大版本迭代：统一元信息&托管&感知->固化防控->落地防控框架->∞

找场景 → 变更元信息 变更过程信息 → 基础 观测能力 → 基础 管控能力 → 抽象 管控能力 → 数据面板 持续运营 → 智能 检测

建立标准价值 数据汇集价值

技术框架-v0.1

- 将标准化能力落地到技术实现上，通过技术协议、技术流程来承接标准化要求
- 统一托管内部的变更平台，变更场景，将变更场景清晰描述出来
- 统一收录已托管场景的变更实例，明确变更对象、变更内容、时间、范围等

建立防御标准 沉淀防御能力

技术框架-v1.0

- 将托管场景的管控级别，从上报提升到具备准入&准出防御能力
- 抽象出通用的防御能力，在核心的变更场景（高频/高危）上接入

数据驱动建设 智能检测阻断

技术框架-v2.0

- 围绕变更平台、变更对象和变更内容建设多维度全方位的数据中心
- 产出变更覆盖、变更防控和变更分析的多项关键指标，基于数据驱动变更工作的持续开展，并及时识别优先级

场景实践

SREelite

分享一个核心场景

容器发布场景（高频、高优、高质）

面临的问题

- 代码不经过集成、预发环境直接上生产
- 漏发、错发代码或配置
- 回滚时漏了配置，db migration等依赖

- 发布时才发现配置忘记变更
- 发布时才发现发错镜像
- 发布时才发现引入了存在bug的依赖

- 发完了才发现SLO跌了
- 并发度不合理容量压力过大
- 新版本问题还没暴露就发完了
- 发布过程中SLO指标变化不可追溯

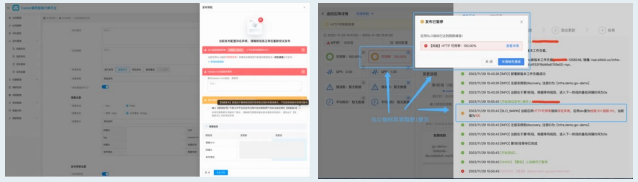
解 决 方 案

- 校验环境发布顺序
- 管控构建变更日志及发布版本日志
- 规范可构建/发布的分支
- 管控回滚方案

- 提供diff能力，在发布创建时即感知此次发布会引入的变更内容，包括但不限于：1. 应用配置2. 发布侧配置3. 镜像（代码、依赖库）4. 中间件（数据库、缓存服务）
- 增强发布预检能力，在发布创建时感知可能的风险，包括但不限于：1. SLO指标2. 校验变更日志及发布版本3. 容量

- 发布过程的实时上报，涉及信息：发布各阶段中 接流实例/总实例 比例、应用容量、发布单状态等信息
- 发布过程引入防控能力，基于以下指标提示和拦截：1. 业务 SLO 指标（可用率、错误数、延时等）；2. 业务容量（QPS、CPU 使用率、内存使用率）
- 发布过程的观测加强：1、突出**变更内容的差异提示**，2、突出新旧版本的指标间差异；**3、发布工作台增加实时的异常信息流提示**

效 果 展 示



组织文化

SREelite

基于组织架构支撑的责任划分，变更制度建设，日常文化宣导，提升全员安全变更意识和行为规范。

组 织 支 撑

- 建立安全生产委员会
  - 全局稳定性决策、规划
  - 安全生产规则的”立法、司法、“
  - 整体应急协同、安全文化培养
  - 全局管控系统的规划与管理

制 度 建 设

- 各种规章制度建立
  - 安全变更门禁制度
  - 系统变更管理规范
  - 系统变更设计准则
  - ...

文 化 宣 导

安全生产月

学院培训

准入考试

技术分享

04

反思总结

SREelite

经验教训

SREelite

充分设计，找准切入，兼顾效率，保持效用，日拱一卒

充分设计

在各个领域中抽出来一个独立的领域做建设，存量的替换成本是极高的

平台愿意折腾的次数可能只有一次

变更领域内的核心要素一定要思考清晰，反复沙盘，再推进业务接入

- 标准定义
- 模型定义
- 交互协议

找准切入

前期阶段

- 寻找高频易错变更场景进行落地
- 寻找有合作意愿的团队
- 自己手中有场景最好，先打样

中期阶段

- 通过高频场景进行泛化，从组织层面进行辐射覆盖
- 寻找低频高危变更场景进行推进覆盖

后期阶段

- 为接入场景提供数据价值，基于数据驱动更多场景覆盖和接入&提升管控级别

兼顾效率

故障情况

- 绿色通道
- 白名单

延迟批次检测

- 牺牲批次的风险，减少用户体感
- 基于历史变更，动态减少变更过程的冷静期

自动盯盘，推进变更

- 系统自动观测指标
- 减少用户在变更过程中的盯盘和点击操作

保持效用

广度控制

- 执行力度，摊子别铺太大
- 到底要管控多细

深度控制

- 到底要对一个变更场景做多细致的防控

是不是瞎折腾了

- 变更不是解决新领域的问题，更多是旧领域的从新建构和构建
- 需要一定的时间，新方案引入的价值才能原有隐性变更建设的成效

日拱一卒

决心问题

- 需要持续性的投入建设，形成系统建设的架构约束

量变引起质变

- 变更的提升在中后期很枯燥无味，不见成效

形成机制、习惯

- 从强迫接收到主动托管

未来展望

SREelite

愿景：实现B站变更防控体系的充分性、高效率和有效性。持续保稳提效，在复杂多变的业务情形下借助数据&智能建立可持续的变更防控机制。

- 覆盖充分：确保变更平台、变更场景足量足额覆盖
- 回归充分：针对历史由变更触发的故障，建立起有效的变更手段，并联动混沌工程进行有效验证

- 感知有效：已发问题，能够回溯到变更信息
- 防控有效：确保已布控变更场景能够有效的阻断问题
- 恢复有效：联动预案，实现防控后的有效恢复

- 接入效率：进一步减少平台接入改造成本
- 变更效率：持续降低变更托管后用户的变更感知
- 防御效率：通过引入智能检测手段，提高变更过程中检测、防御的触发效率，减少变更引发风险的影响

Q&A

SREelite

https://sre-elite.com

附录

供 参 考

SREelite

https://sre-elite.com

“SRE精英联盟”概述

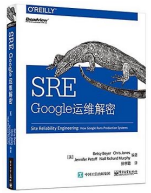
SREelite

2003年

Google

Google 成立 SRE 团队

2016年



孙宇聪翻译出版首部 SRE 著作

2020年

赵成牵头成立 SRE研讨社群

2023年11月

SREelite

SRE 精英联盟正式成立

2024年2月

《SRE 实践白皮书》首次正式发布

2024年2月29日

联盟首次直播

《SRE实践白皮书》

SREelite

- 经历数年，20 多位一线专家协作编写。
- 扫码下载 v1.0.1 。版本持续更新迭代中。
- 在官网 https://sre-elite.com/notice/下载最新版。

联系我们

SREelite

公众号

视频号

B 站

YouTube