第二届CCF·夜莺开发者创新论坛

中国北京　2024.7.26

主办方：中国计算机学会 | 承办方：CCF开源发展委员会、夜莺项目开源社区

字节跳动观测数据埋点标准化实践

舒博

字节跳动可观测团队 高级工程师

中国北京　2024.7.26

主办方：中国计算机学会 | 承办方：CCF开源发展委员会、夜莺项目开源社区

大纲

- 背景
- 埋点标准化的挑战与拆解思路
- 实践与效果
- 总结

中国计算机学会 CCF CHINA COMPUTER FEDERATION
Nightingale

背景

随着字节业务规模越来越大，稳定性治理 就成为了一个越来越重要的话题。

观测数据标准化 及 配套的数据链路就是后续稳定性建设的数据基石。

统一的观测数据标准， 可以极大程度提升团队间排障效率，从人肉分析提升到更大程度自助/自动化排障的阶段。

埋点标准化的重要性

提高研发效率且降低研发协同成本

为AIOps 提供强有力的数据支撑

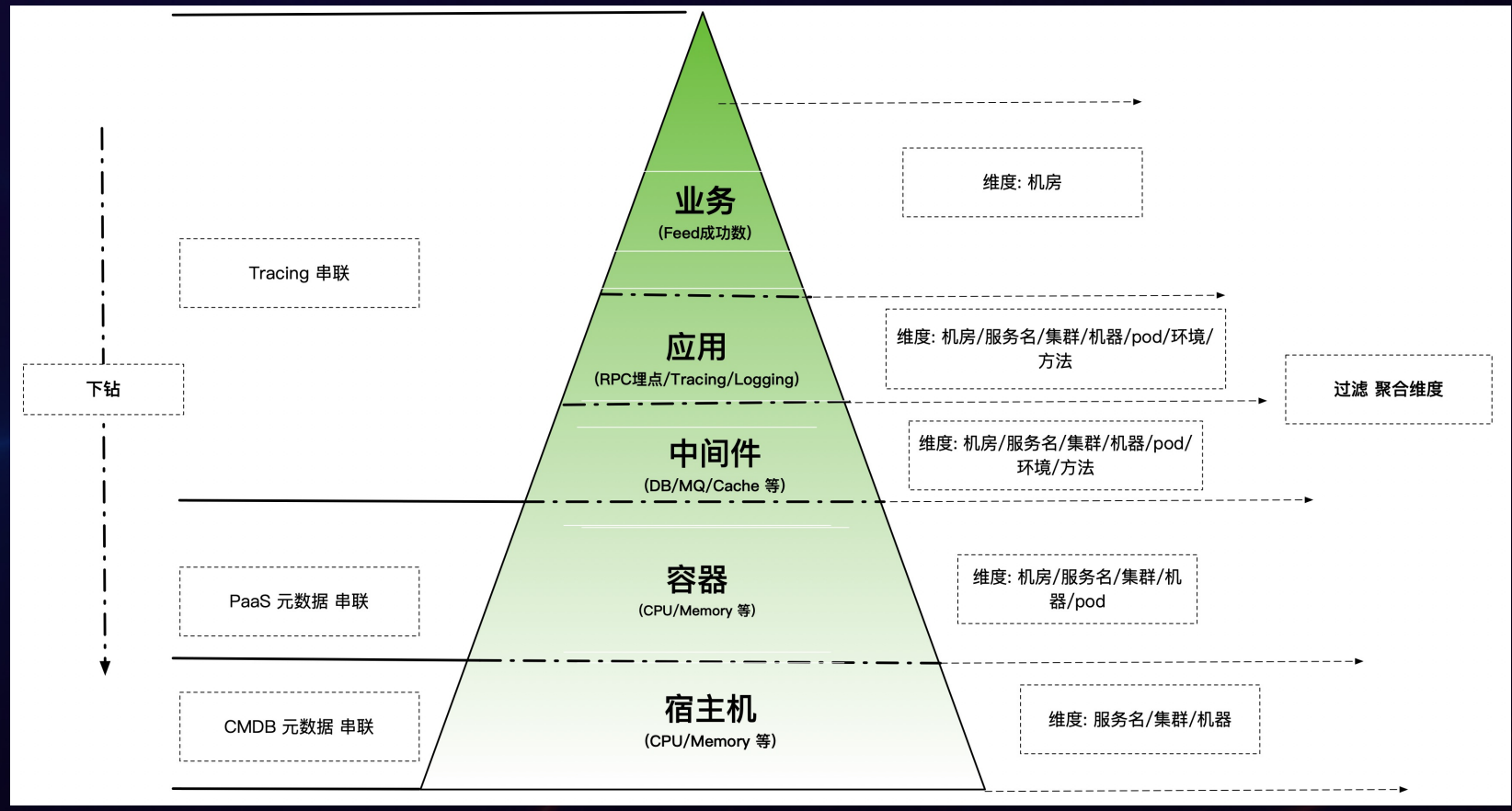
中国计算机学会 CHINA COMPUTER FEDERATION
Nightingale

提高研发效率且降低研发协同成本

面向排障: 跨层间 上下文过滤便捷 并统一术语

历史数仓分析 整体pipeline 逻辑和 适配成本降低很多

降低用户的学习曲线 以及 心智理解 负担



为AIOps 提供强有力的数据支撑

清华大学裴丹老师的 < AIOps 落地的 15 条原则 > 的架构路线 也提到了 数据的重要性：

数据知识驱动、算法代码联动、人机协同

观测数据是基石之一

有了数据标准化和统一的访问体验，为后续稳定性终极目标 MTTR 1-5-10 提供了数据层面的保障， 包括同层数据的聚合/过滤 以及跨层数据的下钻和上卷 都会有统一的使用姿势。

埋点标准化的挑战与拆解思路

挑战： 历史上可观测性埋点质量偏低

思路： 分层&向后兼容推进埋点标准化

卡点： 识别和解决

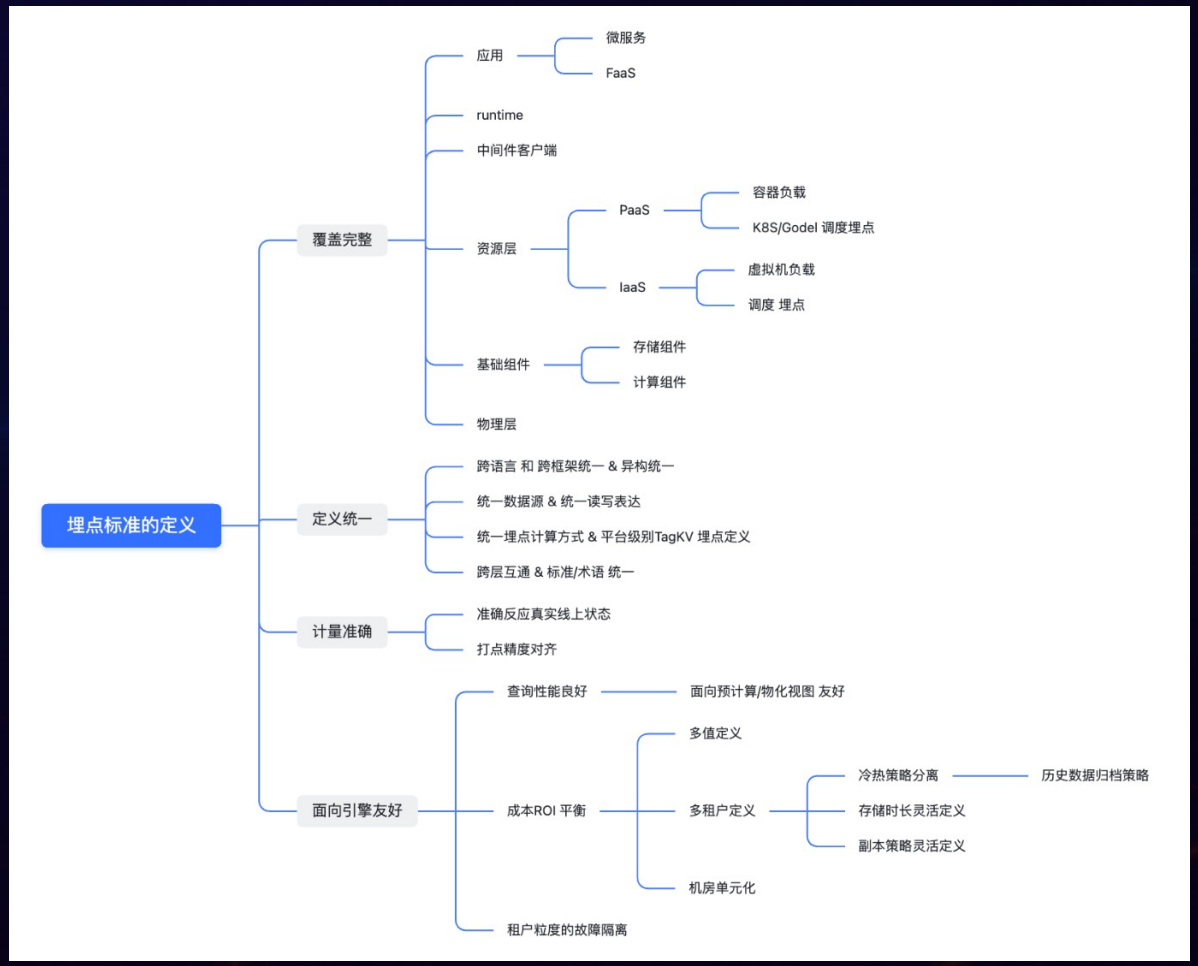
埋点标准化的定义：

覆盖完整

定义统一

计量准确

面向引擎友好



中国计算机学会 CHINA COMPUTER FEDERATION
Nightingale

2020 年之前， 字节整体观测数据埋点质量情况

	M. T. L 横向覆盖是否完整	定义统一	计量准确	面向引擎友好
TLB	N/A	高	中 之前打点丢失问题较严重	低 •之前指标打点 对于配置预计算不友好 •指标名膨胀也比较严重
微服务	中 20 年前 Tracing 方案还在 V1 版本	高	中 之前遇到高基数的指标会被封禁	低 •之前指标打点 对于配置预计算不友好 •指标名膨胀也比较严重 •加权计算也不好实现
语言 runtime	N/A	低 Golang & c++ 框架 不同的版本定义的指标格式都不太一样	高	低 •之前指标打点 对于配置预计算不友好
容器指标	低 •没有日志采集覆盖	高	中 之前遇到高基数的指标会被封禁	低 •之前指标打点 对于配置预计算不友好
基础架构 存储 & 数据库	低 存储、数据库、MQ 客户端也没有黄金指标打点 没有日志采集覆盖	低 不同存储、数据库、MQ 产品打点格式 都不一	中	低 •之前指标打点 对于配置预计算不友好

服务端观测数据质量大致分 3 类问题：

- 同层数据 / 跨层数据不统一
- 观测多模态数据类型 [指标、日志、链路]数据定义不统一
- 观测数据格式面向引擎不够友好， 比如所有的数据都在 default 租户一个大仓，再如很多观测指标定义对于预计算不友好。

思路： 分层和向后兼容推进埋点标准化

[问题一] 同层数据 / 跨层数据不统一

解决方案: 协作组件设计打点规范

- 微服务RPC: 引入多租户+多值 & 对齐TagKV 术语
- TLB: 引入多租户+ 多值 & 面向预计算友好
- 容器指标: 引入多租户+多值 & 对齐TagKV 术语:

思路： 分层和向后兼容推进埋点标准化

[问题二] 观测多模态数据类型 [指标、日志、链路]数据定义不统一

解决方案: 采集覆盖+埋点术语统一

- 日志采集覆盖: 微服务+容器
- 链路覆盖 : SDK 基线版本升级
- TLB: 引入多租户+ 多值 & 面向预计算友好
- 多模观测数据埋点术语统一: 统一 M.T.L SDK TagKV定义; 埋点vendor SDK 感知运行时环境 定义TagKV

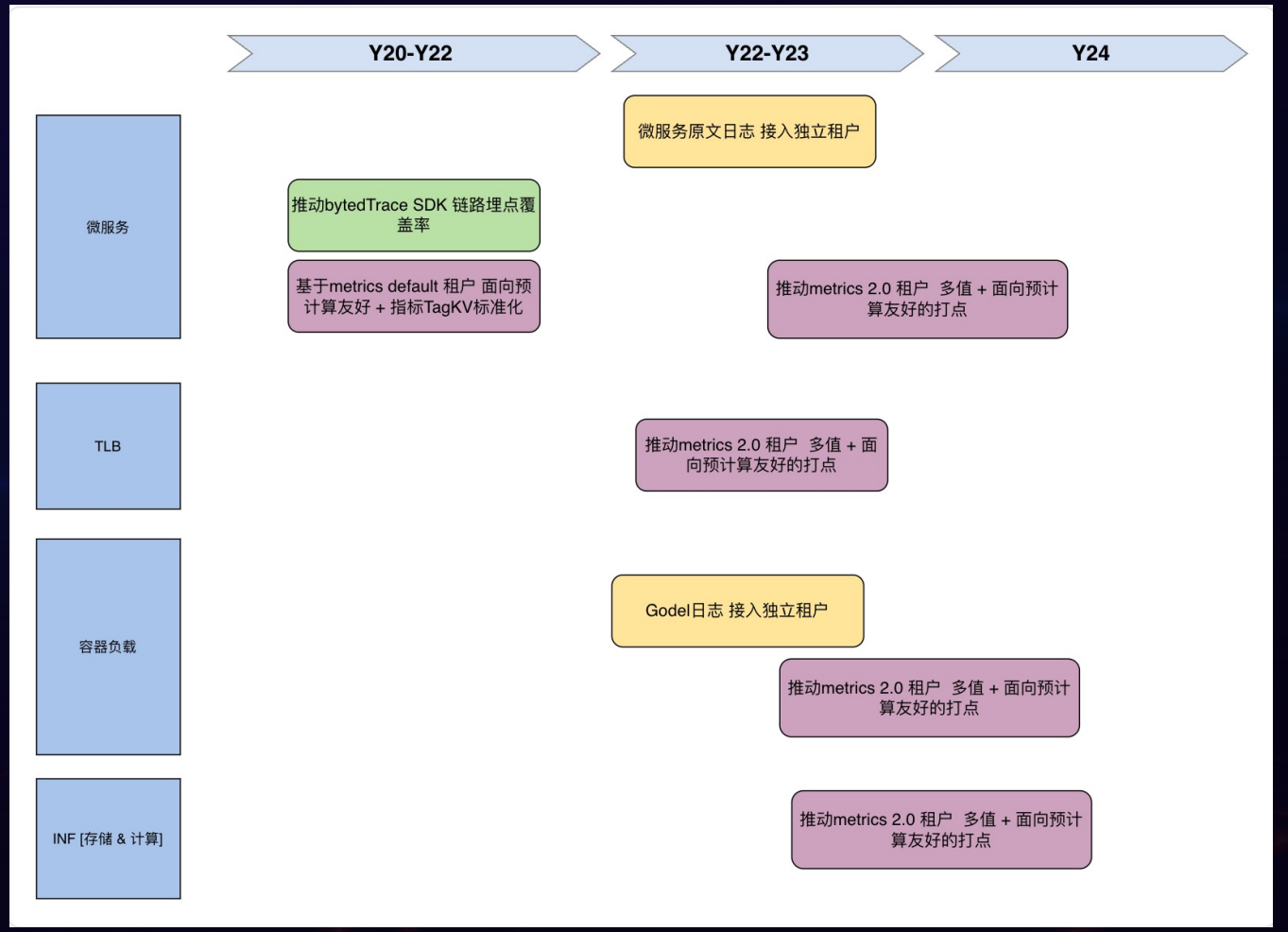
思路： 分层和向后兼容推进埋点标准化

[问题三] 观测数据格式面向引擎不够友好

解决方案:指标标准化

引入metrics 2.0 [多租户 & 多值]: 2.0 SDK 在性能开销&传输效率都优于1.0

团队历年来在多个观测对象上埋点做出的业务推进



中国计算机学会 CHINA COMPUTER FEDERATION
Nightingale

卡点：识别和解决

如何高效推动业务升级？

如何进一步提升核心组件的埋点质量？

如何保障观测数据迁移对于在线核心观测大盘和报警影响最小化？

如何降低对接的人力成本？

如何高效推动业务升级

BytedTrace SDK 集成 RPC 框架

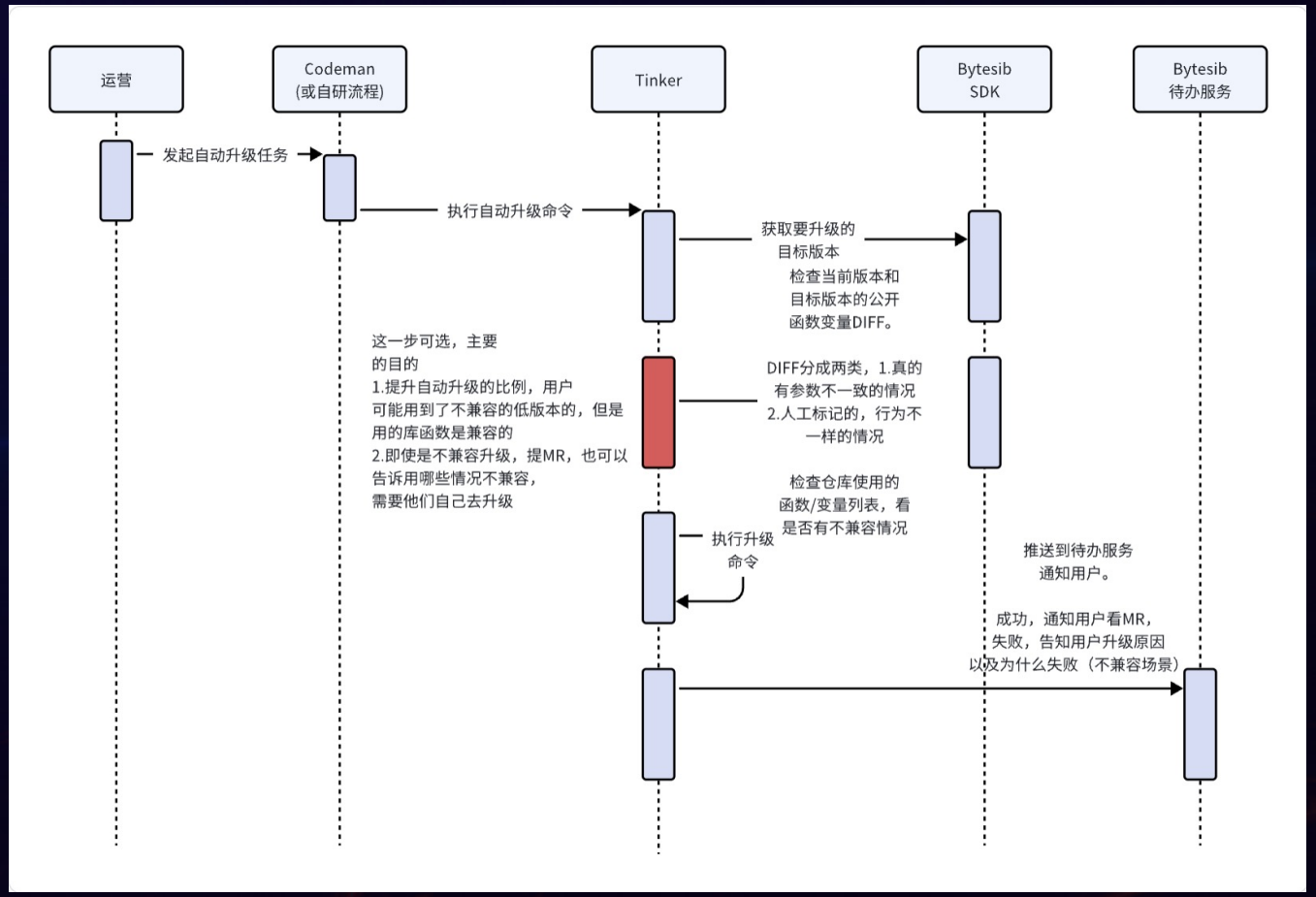
BytedTrace 团队为公司常用框架和组件集成了 BytedTrace SDK。
目前对于大部分 RPC 框架和存储端 Client 都已完成。

按服务优先级来看，公司当前 P0 服务已有 94%接入 Bytedtrace SDK

level	V2 服务 接入率	V1+V2 服务接入率	V2 服务数	V1+V2 服务数	服务 总数
P0	0.9546	0.9597	1682	1691	1762
P1	0.8988	0.9150	2008	2044	2234
P2	0.8233	0.8577	5964	6213	7244
P3	0.8190	0.8558	267	279	326

如何高效推动业务升级

联合 bytesib 团队计划实现 sdk 自动升级功能



中国计算机学会 CHINA COMPUTER FEDERATION
Nightingale

如何进一步提升核心组件的埋点质量？

容器 / Runtime 打点格式 设计思路

层次	核心组件/着手点	埋点标准化设计思路
业务层	Metrics 2.0 SDK	内置统一公共的TagKV，提供横向跨语言、跨服务的TagKV统一
应用层	Runtime 指标、RPC 指标	横向上，提供统一的、跨语言的指标名定义 纵向上，对齐Metrics 2.0 SDK公共TagKV规范
容器层	与调度合作，对容器指标采集agent(sysprobe)进行标准化改造	对齐Metrics 2.0 SDK公共TagKV规范

如何保障观测数据迁移对于在线核心观测大盘和报警影响最小化？

推动SDK的升级，指标名、TagKV的语义统一，必然会引起存量观测大盘、报警规则的不一致。 看板在千级别数量级、涉及的报警规则的数量在十万级别

中国计算机学会 CHINA COMPUTER FEDERATION
Nightingale

如何保障观测数据迁移对于在线核心观测大盘和报警影响最小化？

语义化指标替换 --- Measurement

对原始 metrics 打点的语义化封装

识别在不同条件下，能使用对应版本的指标查询以及对应的 tagkv

如何保障观测数据迁移对于在线核心观测大盘和报警影响最小化？

1.通过元信息获取观测对象的信息

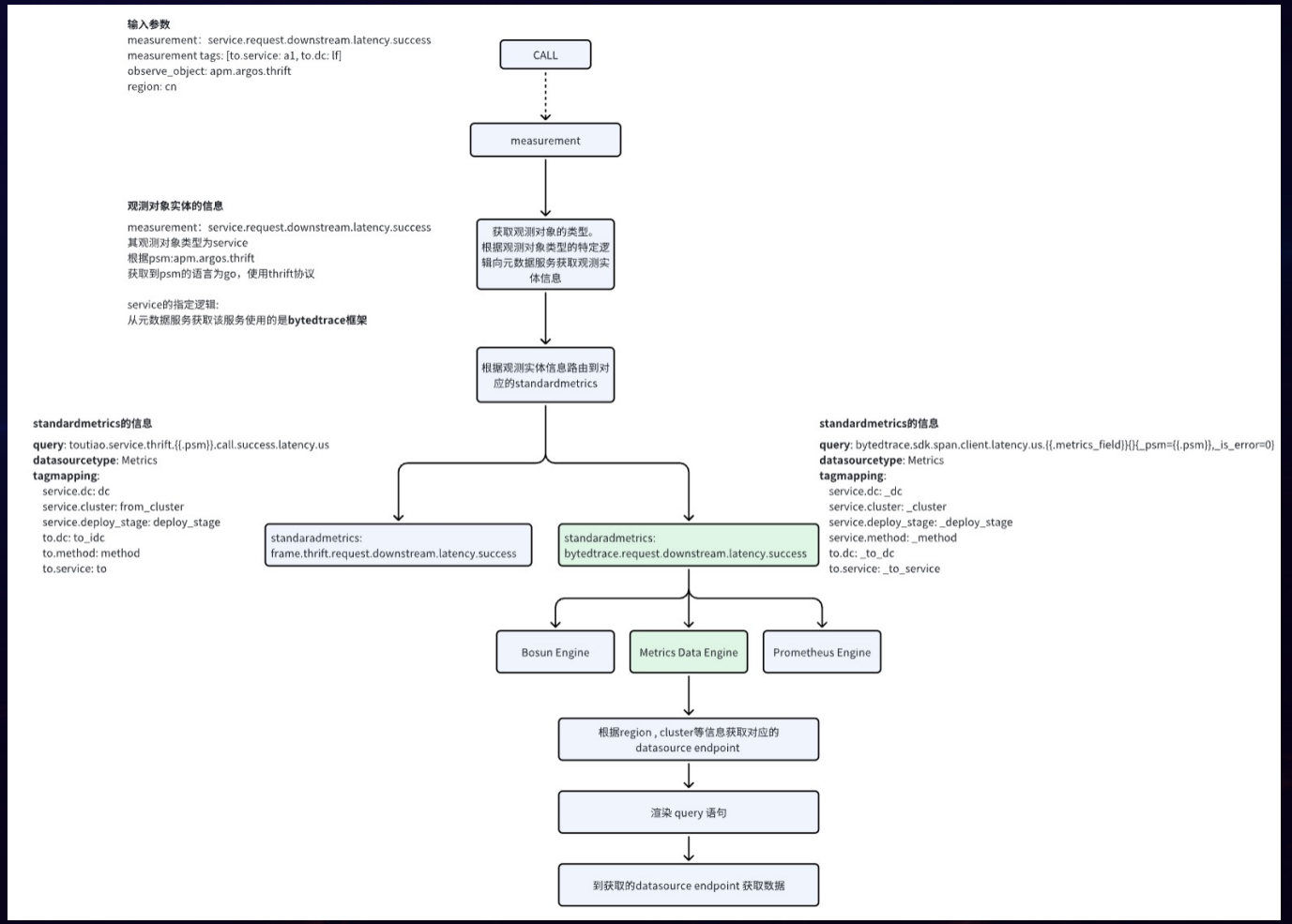
2.根据信息和处理逻辑，选择应使用的standardmetrics。

3.根据输入region等信息和standardmetrics的datasource type，获取数据引擎和数据源endpoint。

4.渲染standardmetrics的query语句，参数为:

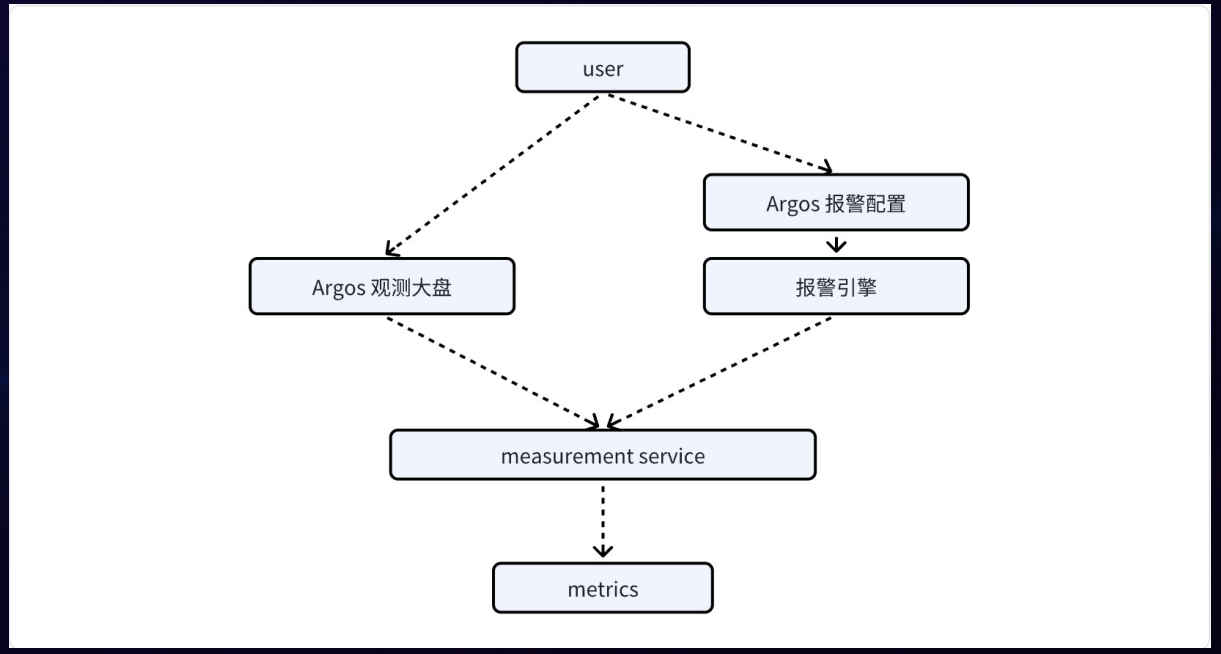
1. 指标中的具体观测对象实体
2. 经过预先设定的mapping映射，将各个指标之间有差异的tag key 对齐。

5.根据渲染完成的query 去数据引擎层中请求对应的数据源endpoint 获得数据。



如何保障观测数据迁移对于在线核心观测大盘和报警影响最小化？

抽象出 measurement 服务
作为观测大盘和报警的一个数据源。
在尽量不需要用户介入的情况下完成数据打点的迁移和替换，这里包括观测大盘和报警能力



rpc、http框架，tce容器、faas、bernard平台、以及tlb、redis、mysql、bytedoc、bmq、rmq等基础组件，及go runtime等都做了统一的标准化语义封装。这些语义化封装在服务端Argos观测平台上都有体现。

中国计算机学会 CHINA COMPUTER FEDERATION
Nightingale

如何保障观测数据迁移对于在线核心观测大盘和报警影响最小化？

帮助业务平滑迁移到新租户，且确保新老指标查询方式都可查询, 是推动业务租户迁移中遇到比较大的课题。 Metrics团队发起对用户无感知的被动租户迁移方案 ----

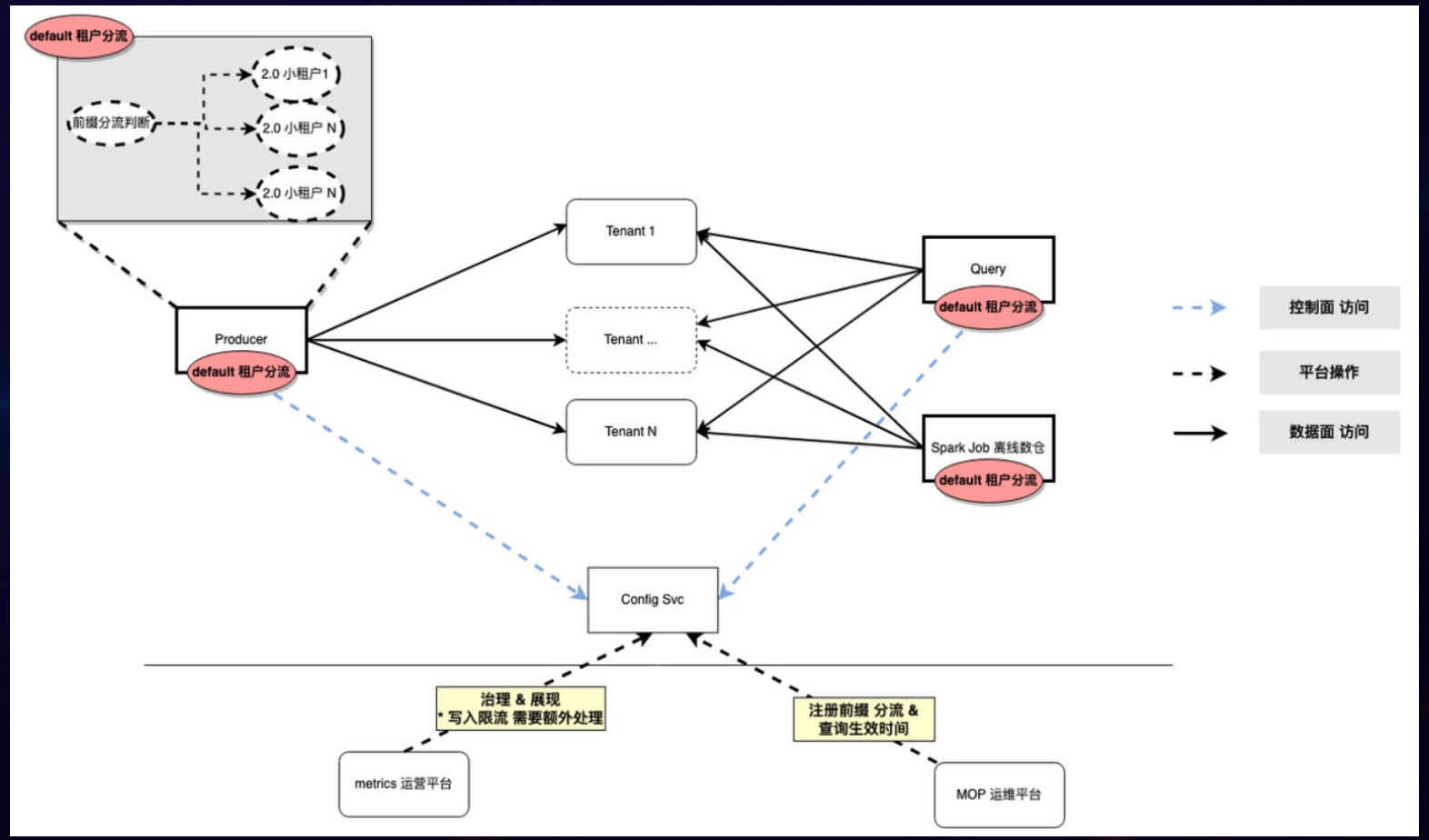
Metrics 前缀分流

中国计算机学会 CHINA COMPUTER FEDERATION
Nightingale

如何保障观测数据迁移对于在线核心观测大盘和报警影响最小化？

根据特定的指标前缀，一级/二级前缀经过配置将指标分别路由到不同的新租户

在新租户上支持查询翻译，支持在用户不修改查询租户的情况下，使用Default租户查询依然可以正常查询到数据



中国计算机学会 CHINA COMPUTER FEDERATION
Nightingale

如何降低对接的人力成本？

metrics 租户运营平台化

•独立租户自助申请。租户创建表单提供了非常多的定制化参数给到用户，用户可以结合自己的使用场景，比如业务打点精度、部署的机房、热存的保存时长、是否需要使用冷存等，来定制化独立租户。

•租户变更申请。在使用过程中，用户可以根据实际使用情况来调整租户运行中的一些参数，比如数据存储策略、写入管控策略。

中國計算機學會 CHINA COMPUTER FEDERATION
Nightingale

实践与效果 --- 标准化质量体现

	M. T. L 横向覆盖是否完整	定义统一	计量准确	面向引擎友好	成本收益
TLB	N/A	高	高[20 年前 低] •通过 2.0 SDK 三个特性， 基本消除丢点的问题	高 [20 年前 低] 实际效果： •面向预计算友好	1.Metrics 2. 0 打点商品成本相对 1.0 **下降 94%**。 2.Metrics 2. 0 很好地解决了打点封禁问题，特别是在一些配置量巨大的核心集群，解决了其超过 90%打点无法查询的情况。 3.Metrics2. 0 TLB 机器成本初步统计主容器和 adaptor 打平，同时相对 1.0 节约了 ms2 的 15000 核资源。
微服务	高 [20 年前 中] •80%以上 PSM 覆盖到 bytedTrace SDK 集成	高	中+ •高基数的指标封禁问题 由于迁移到了新租户 可以做封禁阈值定制化 •[计划中] 升级 bytedTrace 内的 metrics 2.0 SDK 降低丢点的风险	高 [20 年前 低] 实际效果： •面向预计算友好	1.以计算关键组件 Consumer 为例，**新租户只需要老租户 20%的资源**，就可以完成相同数据的写入计算（下面说明会介绍推导方法）；其他写入计算类组件也类似 2.以存储关键组件 tsdc 为例，**新租户只需要老租户 55%的资源**，就可以完成想通过数据的写入、存储
语言 runtime	N/A	高 [20 年前 低] •统一了不同语言和框架的 runtime 打点格式	高	低	
容器指标	中 [20 年前 低] •Godel 接入日志租户	高	高 [20 年前 中] •引入多值 降低指标名数量 •高基数的指标封禁问题 由于迁移到了新租户 可以做封禁阈值定制化 •通过 2.0 SDK 三个特性， 基本消除丢点的问题	高 [20 年前 低] •之前指标打点 对于配置预计算不友好	进行中
基础架构 存储 & 数据库	低 •[进行中] 目前有 10+组件 在接入 metrics 2.0 SDK + 租户独立	低 •不同存储、数据库、MQ 产品打点格式 都不一	高 [20 年前 中] •引入多值 降低指标名数量 •高基数的指标封禁问题 由于迁移到了新租户 可以做封禁阈值定制化 •通过 2.0 SDK 三个特性， 基本消除丢点的问题	中 [20 年前 低] •打点格式调整的支持预计算配置	以 mysql 迁移为例 •Mysql 租户 成本**节省 45。7%** •Mysql 租户 带宽**节省了 80%**

实践与效果 ---效果总结

赋能 AIOps 跨层根因定位

通过指标标准化 & 多模观测数据 [指标， 日志，链路]标签术语的标准化，实现面向微服务的上卷 & 下钻关联分析。

也使得跨层问题根因分析有了可能性

实践与效果 ---效果总结

Metrics 存储收益

稳定性： 由于定义了租户， 就可通过逻辑租户映射到物理资源来降低故障半径， 减少不同租户间流量相互干扰。

成本： 通过每个租户的副本数 & 存储时长 TTL 以及打点最小精度 和 多值定义来最大程度降低写入流量 以及存储容量的成本。

流量来看，老集群：

1. 写入网关流量下降：**6.5%**
2. 写入网关视角： byetrace指标在新租户流量只有老租户的 **17.6%**
3. BMQ入流下降：**12.3%~14%**
4. tsdc DataPoint流量下降：**9.7%**

关键资源使用看，老集群：

1. Producer平均cpu使用率下降：**8.2%**
2. consumer平均cpu使用率下降：**9.5%**
3. Streaming平均cpu使用率下降：**26.5%**
4. tsdc内存使用下降：**13.2%**

总结与规划

经过近 3 年的 bytedTrace 推广 & metrics 2.0 租户迁移，字节后端观测数据质量 无论在覆盖完整度、定义统一、计量准确、面向引擎友好四个方面都有相当程度的改善，也为后续全景全栈排障 进而 AIOps 提供了坚实的基础。

未来会在如下两方面继续推进：

- 协助服务端观测平台的监控和报警方向 进一步闭环上述标准化数据访问、下线老数据的写入流量 以及推广到全球的多区域；
- 推动火山引擎云产品接入上述的数据标准化， 提供内外一致的数据质量体验。

中國計算機學會 CHINA COMPUTER FEDERATION
Nightingale

感谢聆听

Thank you for listening

中國計算機學會 CHINA COMPUTER FEDERATION
Nightingale