

# 微服务监控告警Prometheus

## 架构和实践

**讲师：杨波**

研发总监/资深架构师

第

1

部分

课程概述

# 课程大纲

- 01 课程概述
- 02 监控模式分类
- 03 BusDevOps和度量驱动开发(MDD)
- 04 Prometheus简介
- 05 Prometheus架构设计
- 06 Prometheus基本概念
- 07 Prometheus起步查询实验(Lab01)
- 08 Prometheus+Grafana展示实验(Lab02)
- 09 Prometheus+Alertmanager告警实验(Lab03)

10

Java应用埋点和监控实验(Lab04)

11

NodeExporter系统监控实验(Lab05)

12

Spring Cloud Actuator监控实验(Lab06)

13

Prometheus监控最佳实践

14

开源时序数据库比较

15

开源分布式监控平台ZMon简介

16

微服务监控体系总结

17

参考资源和后续课程预览

# 课程概述和亮点



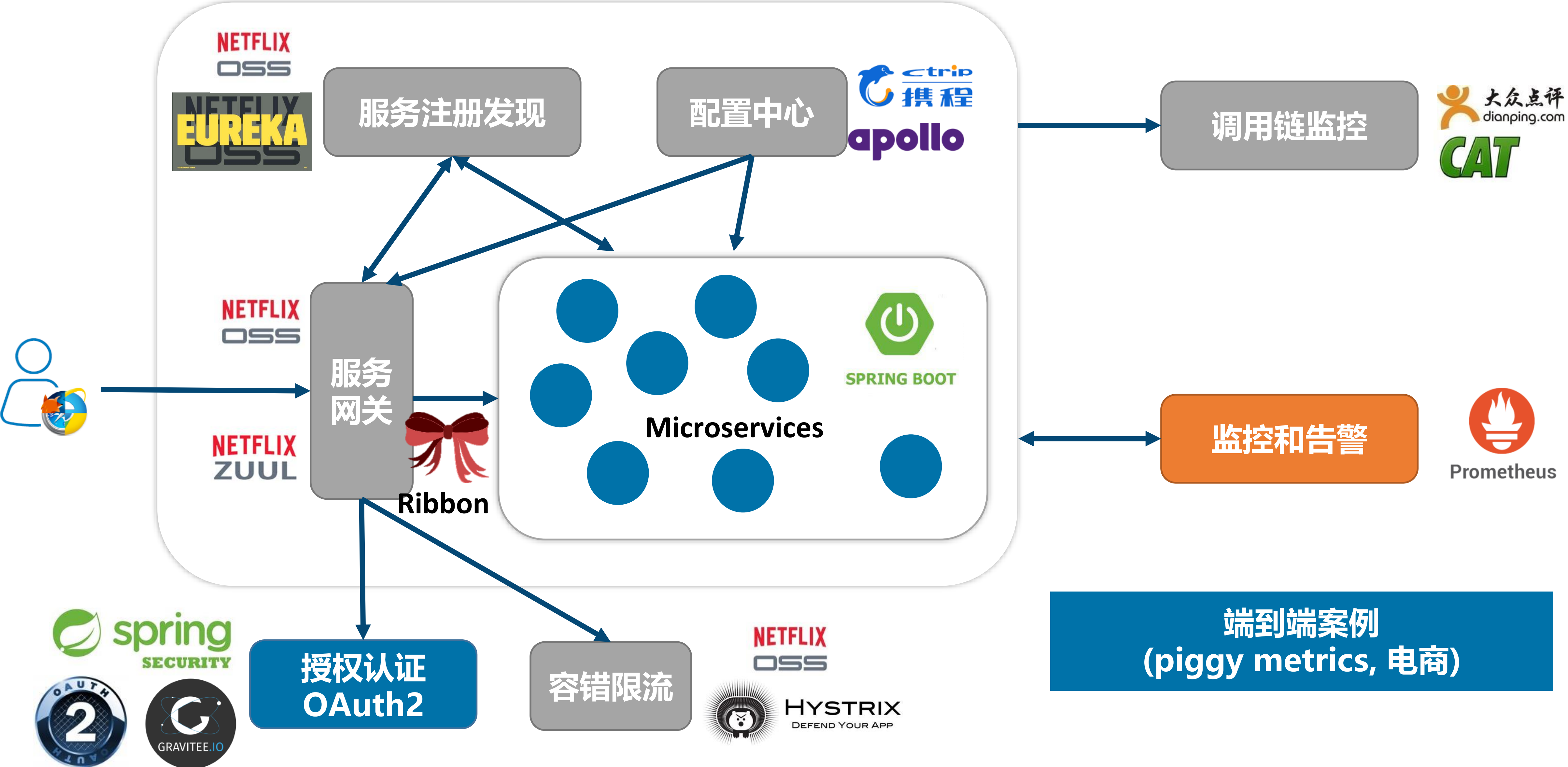
- 1 波波的微服务基础架构体系的**第7个**模块
- 2 **测量驱动开发(MDD)**理念介绍
- 3 **时间序列和Metrics**监控原理
- 4 CNCF监控组件**Prometheus**深度剖析
- 5 Prometheus+Grafana**案例和实验**
- 6 Prometheus监控**最佳实践**
- 7 **主流开源时序数据库**比较
- 8 分布式监控平台**ZMon**简介

# 杨波的微服务基础架构体系2018





# 架构和技术栈预览



第

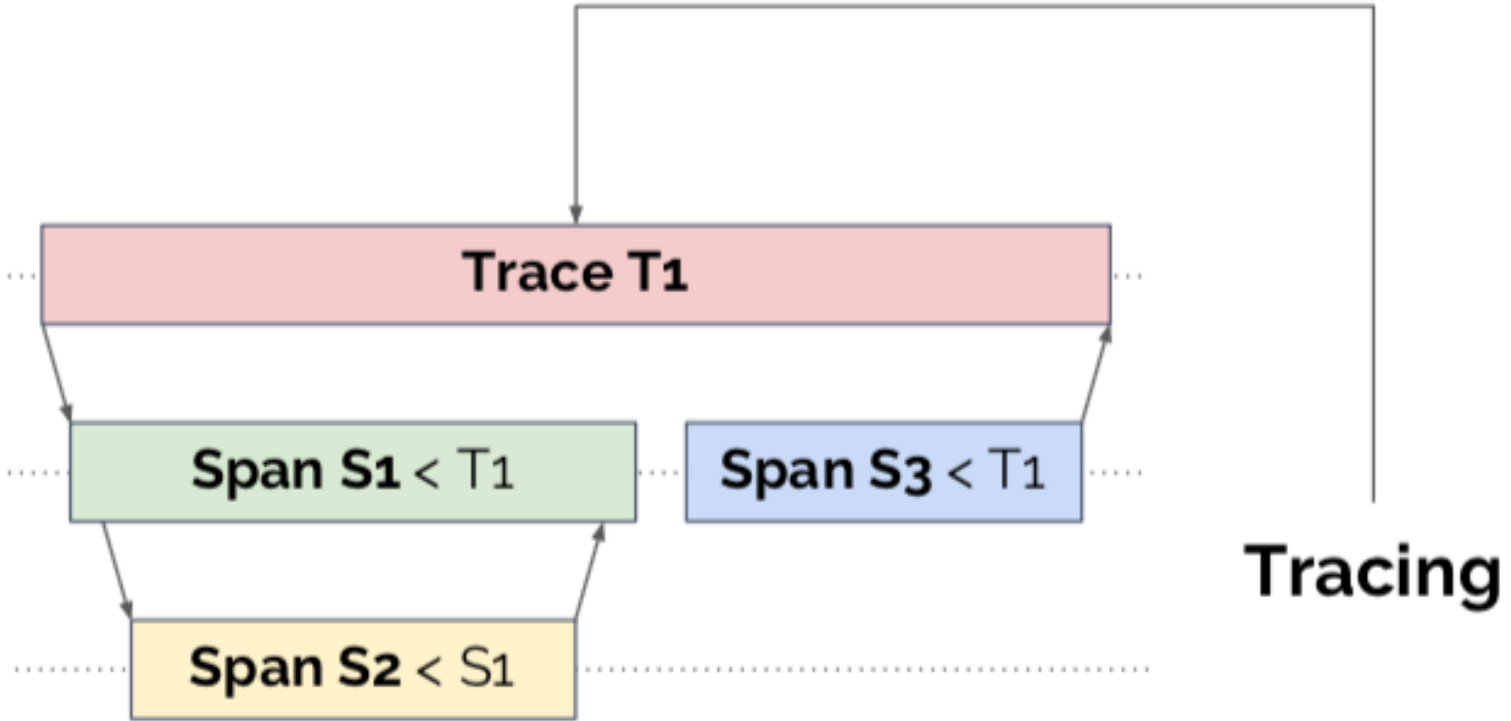
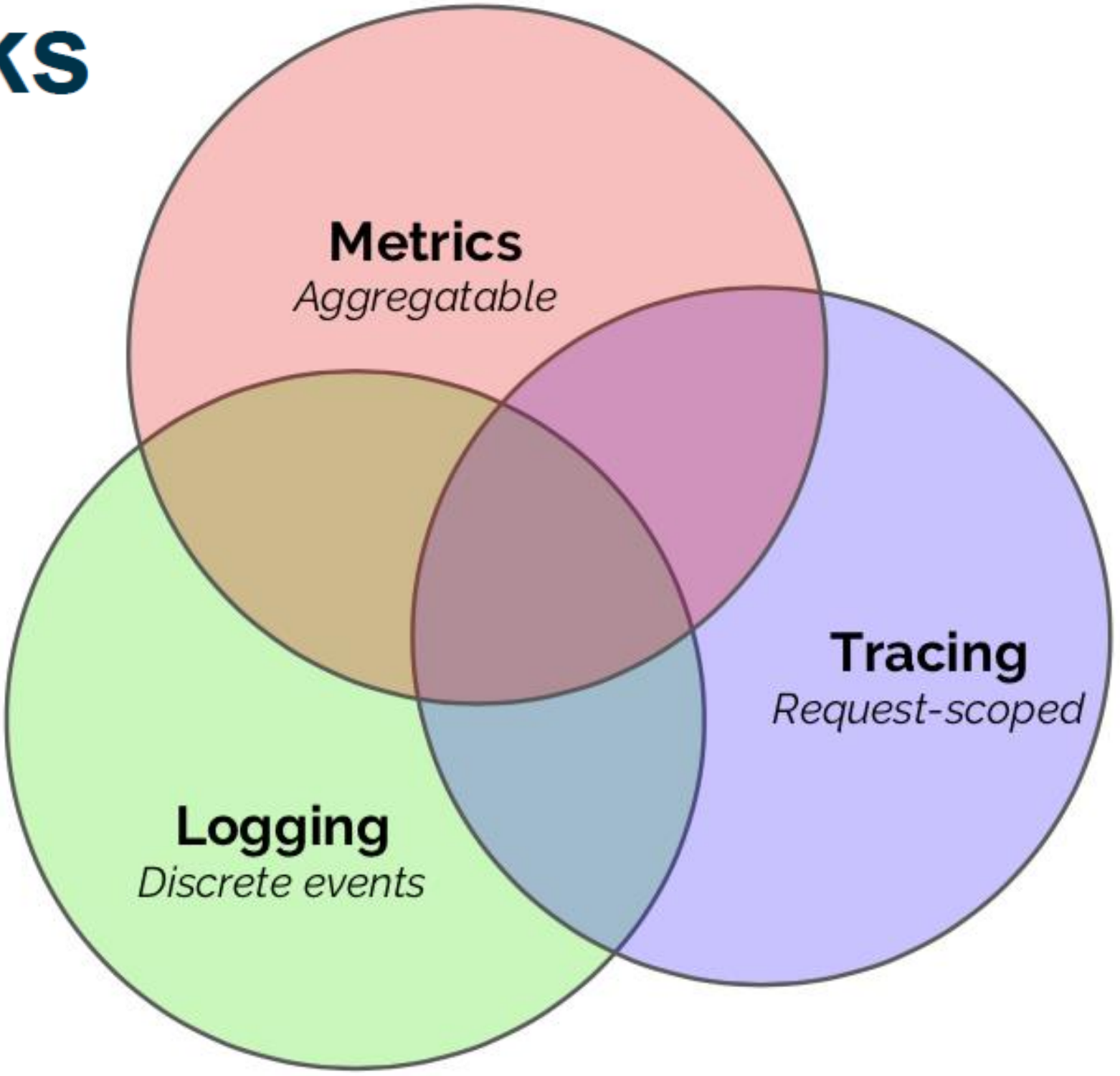
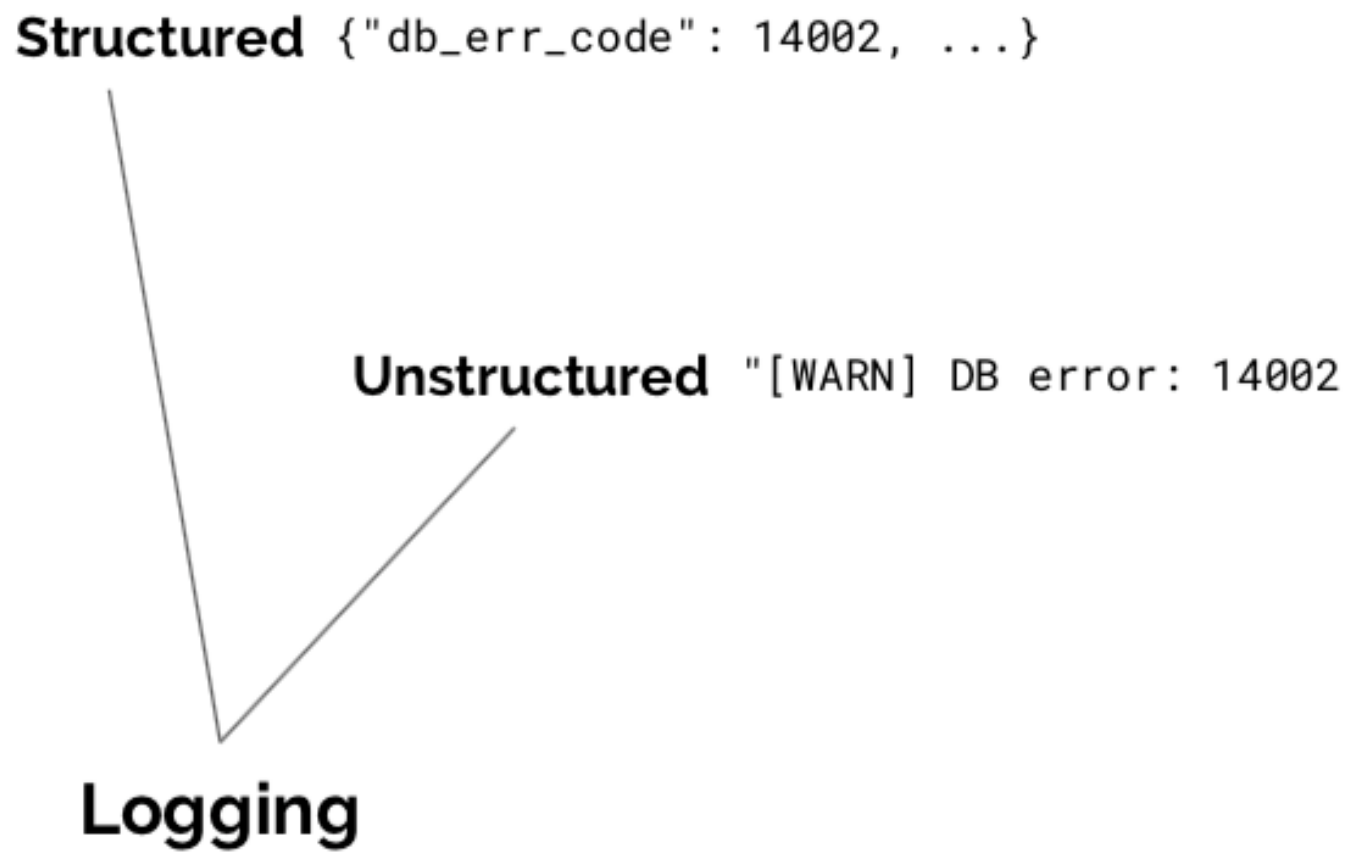
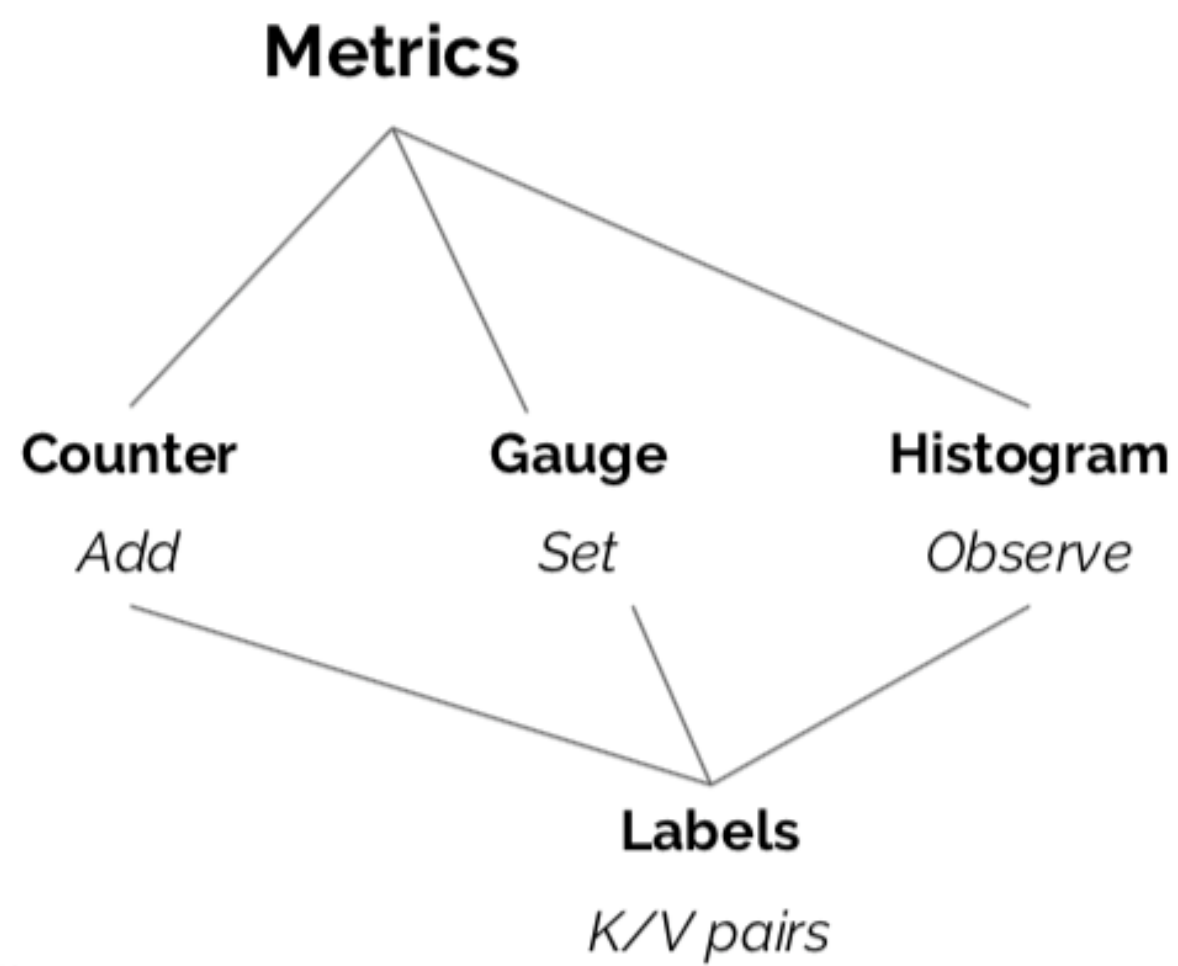
2

部分

## 监控模式分类



# 四种主要监控方式



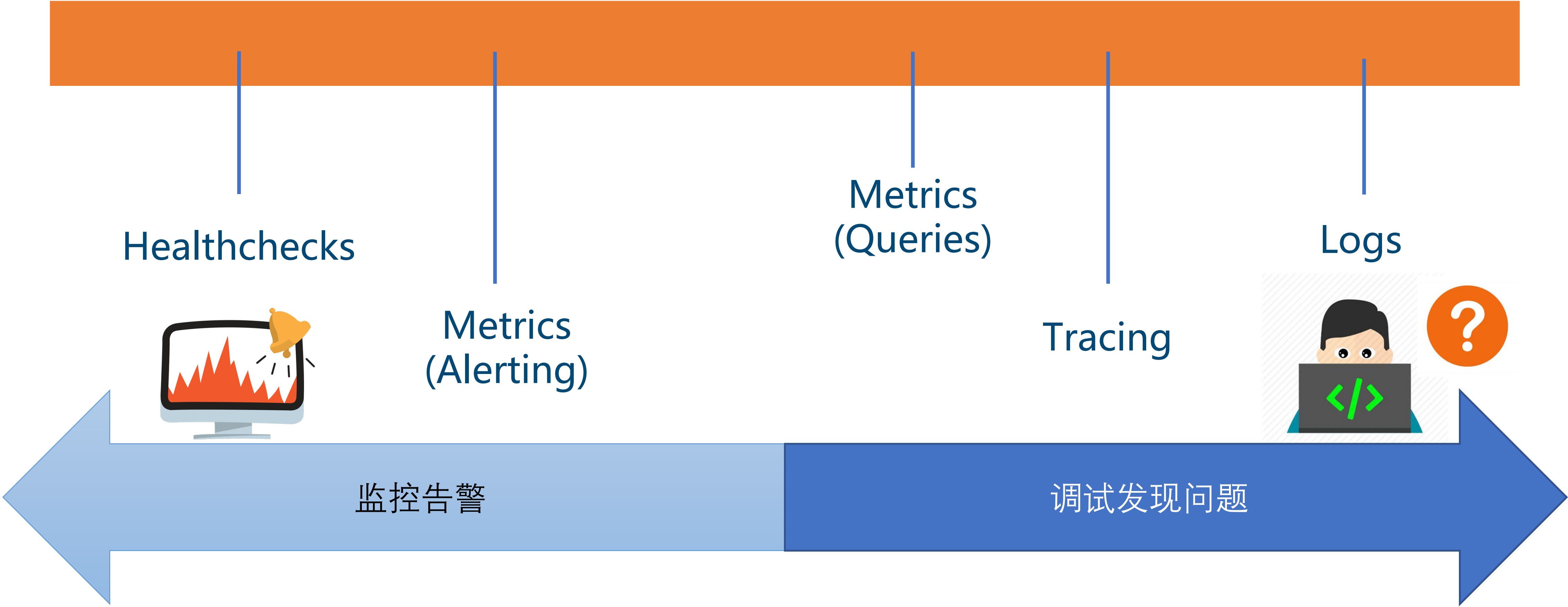
# 比较

	Metrics	Logging	Tracing
CapEx	Medium	Low	High
OpEx	Low	High	Medium
Reaction	High	Medium	Low
Investigation	Low	Medium	High



<https://peter.bourgon.org/go-for-industrial-programming/>

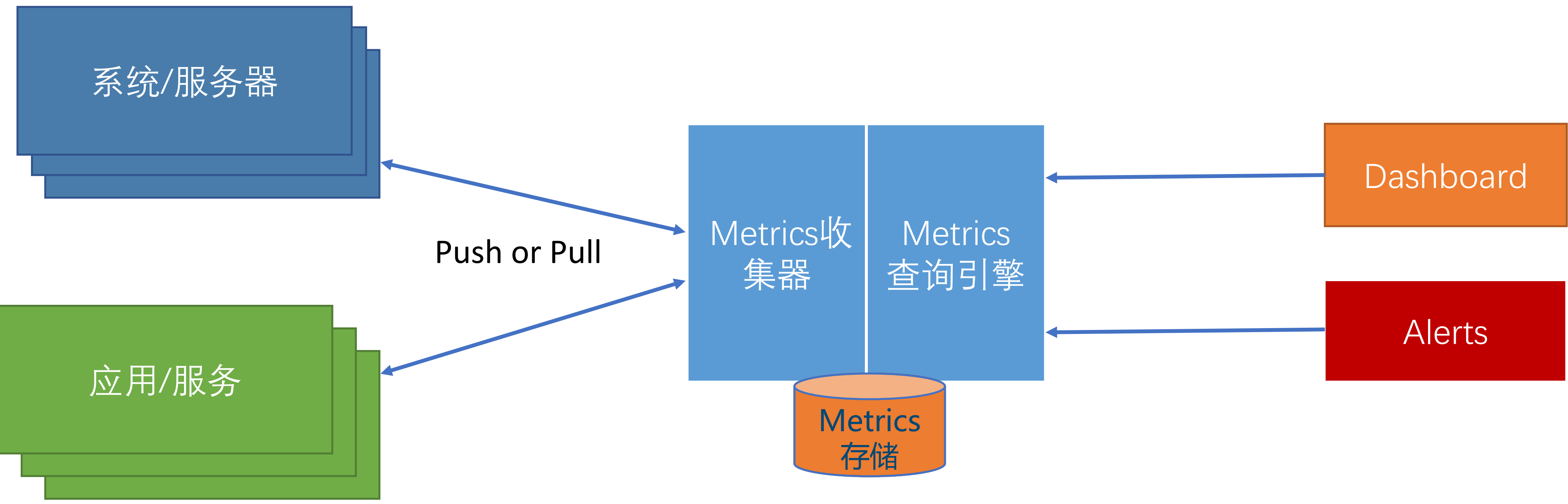
# 适用场景



# Metrics监控分层

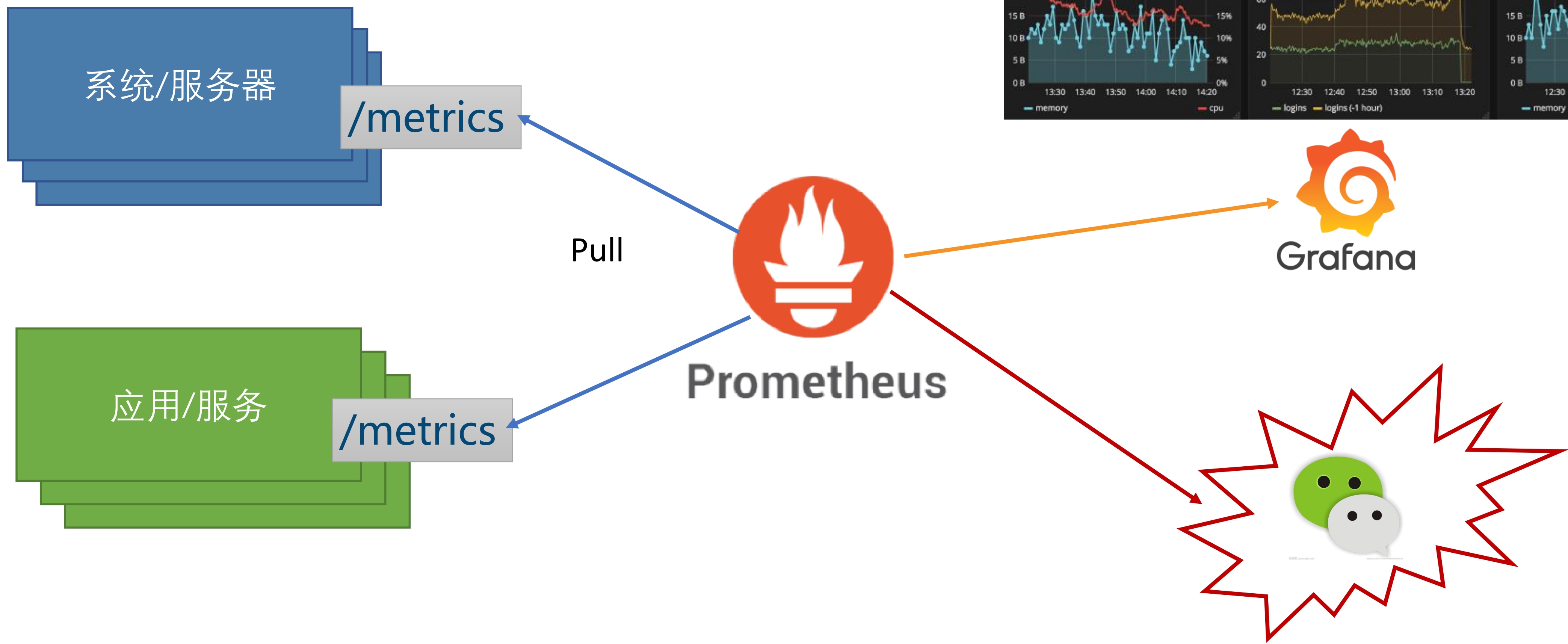


# Metrics监控架构模式





# Prometheus监控Metrics



第

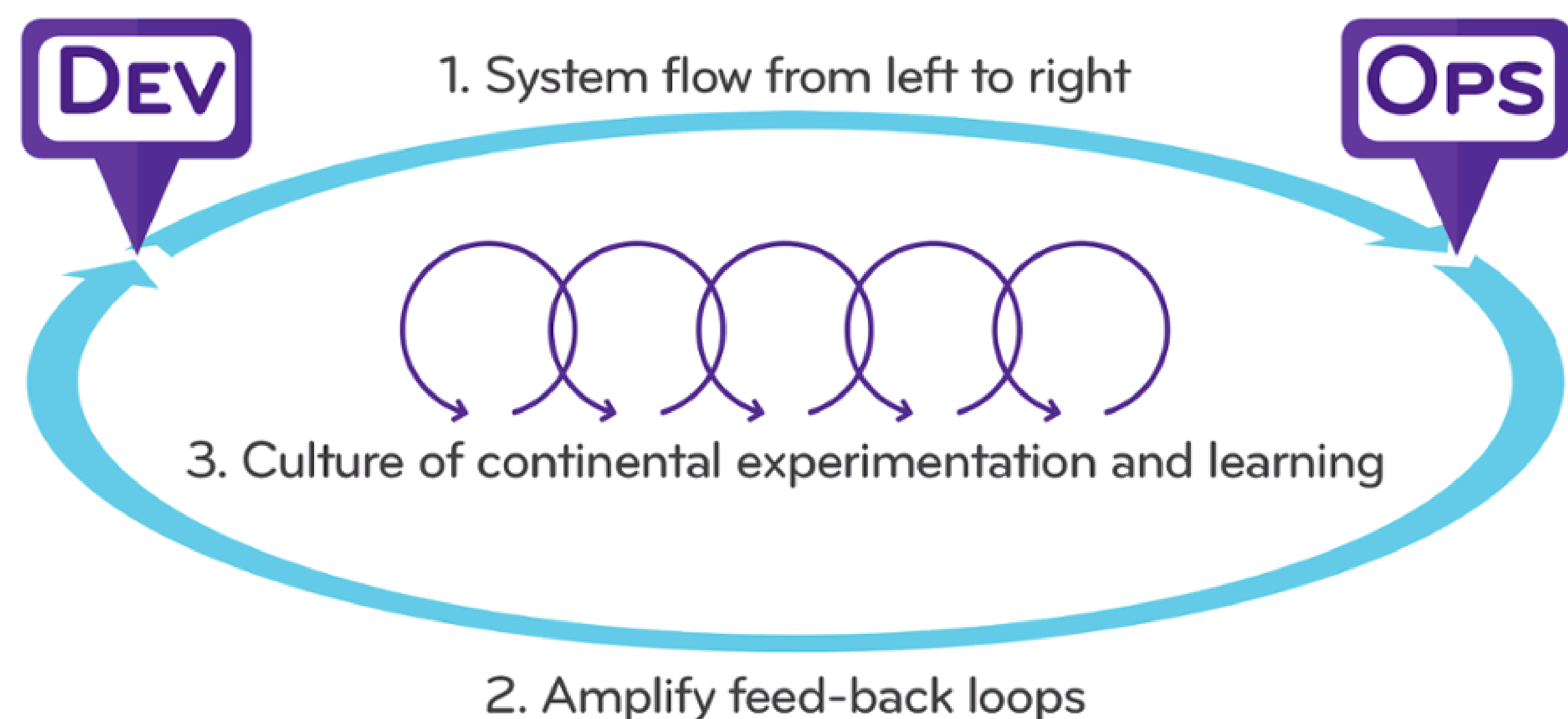
3

部分

**BusDevOps和测量驱动开发MDD**



# DevOps核心原理: Three Ways



## 凤凰项目 一个IT运维的传奇故事

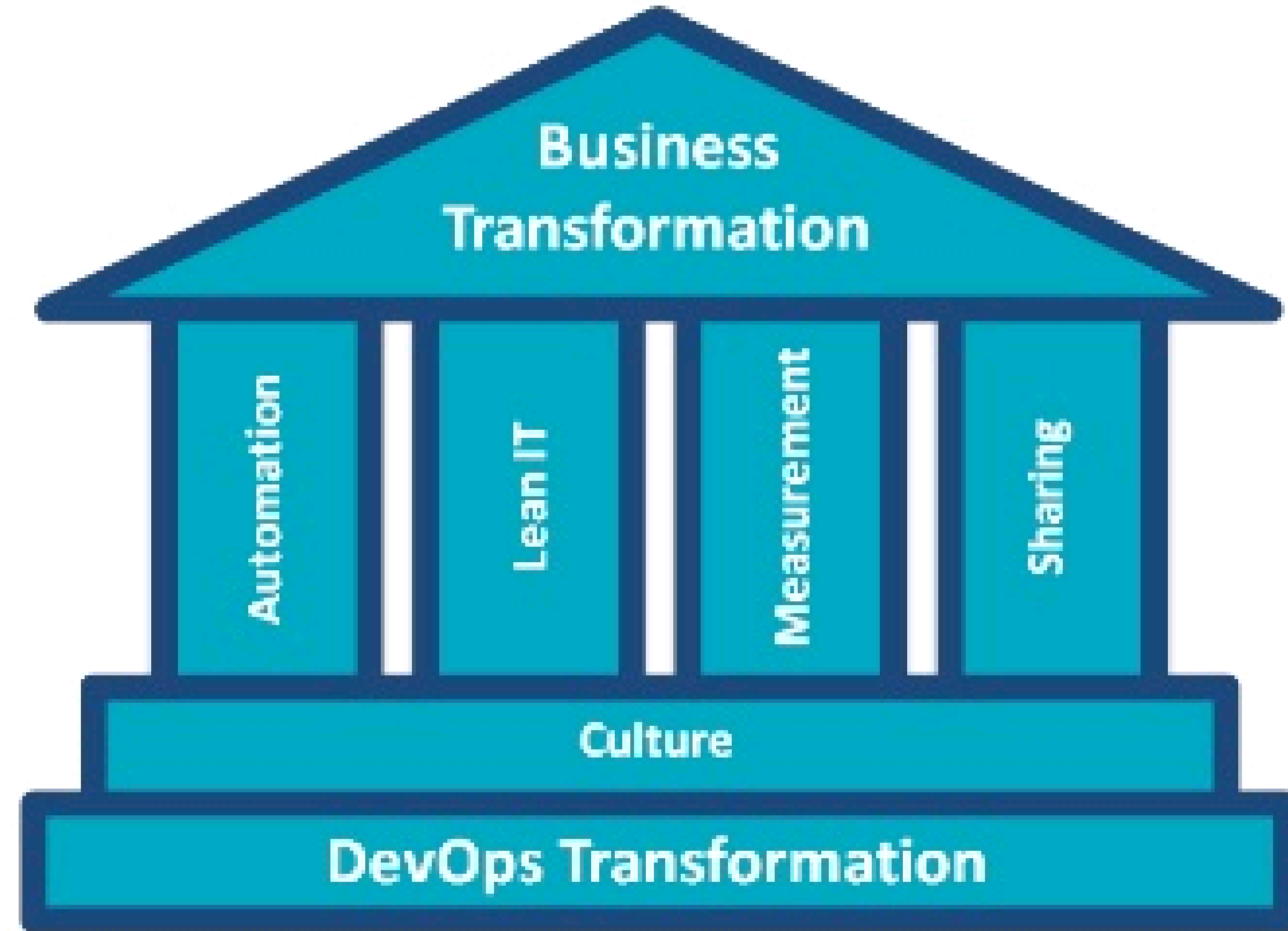
The Phoenix Project  
A Novel About IT, DevOps, and Helping Your Business Win

[美] Gene Kim Kevin Behr George Spafford / 著 成小雷 / 译 张坤 / 审校

中国工信出版集团 人民邮电出版社  
POSTS & TELECOM PRESS

# DevOps~CALMS模型

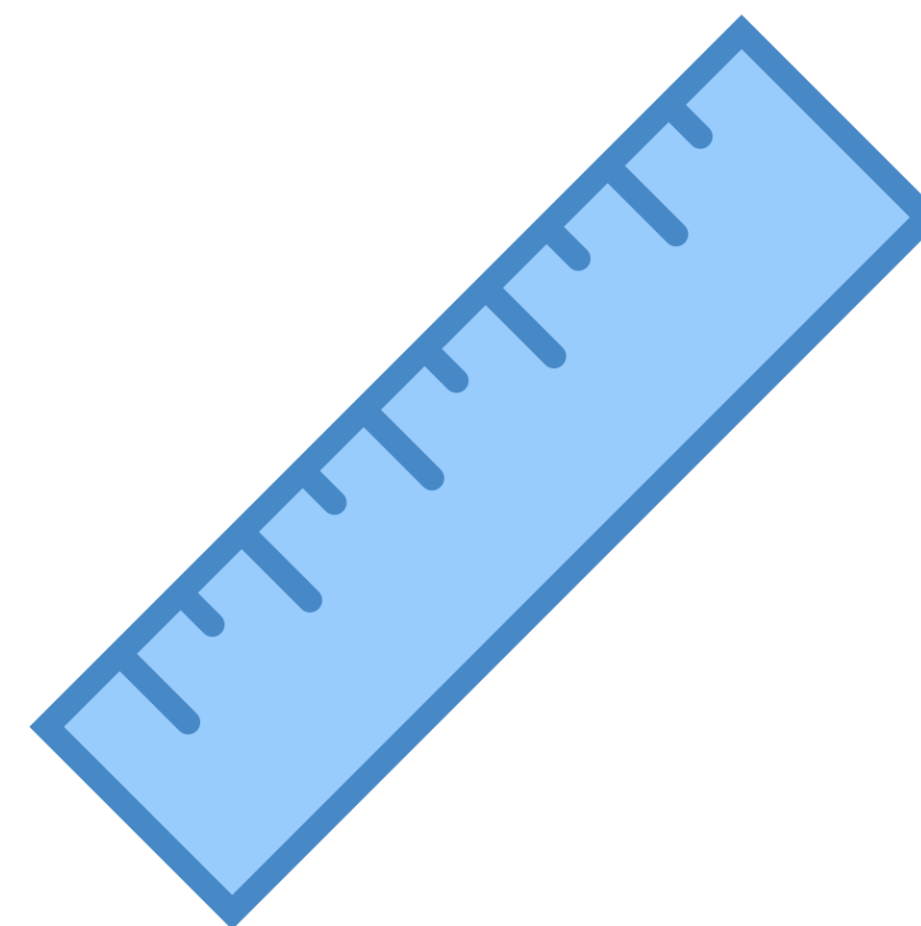
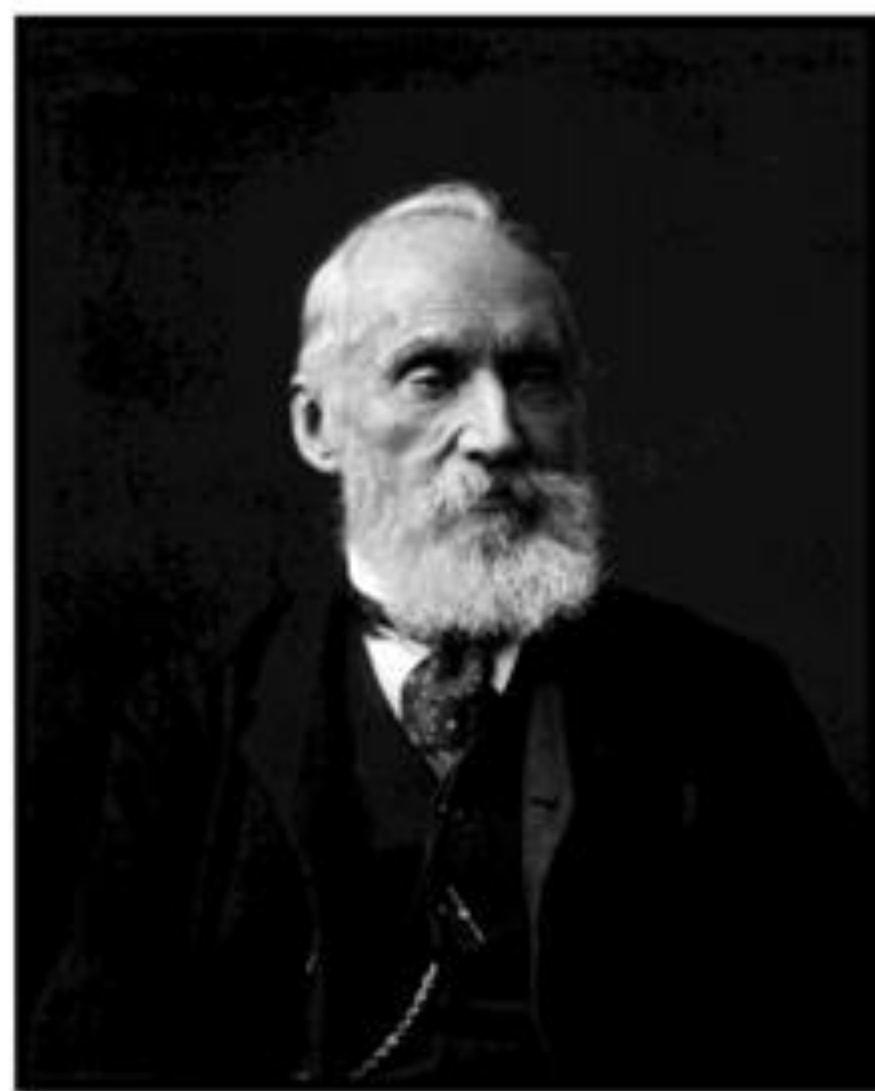
- Culture
- Automation
- **Lean**
- Measurement
- Sharing



# DevOps理念： 要提升必先测量

**“If you can’t measure it, you can’t improve it.”**

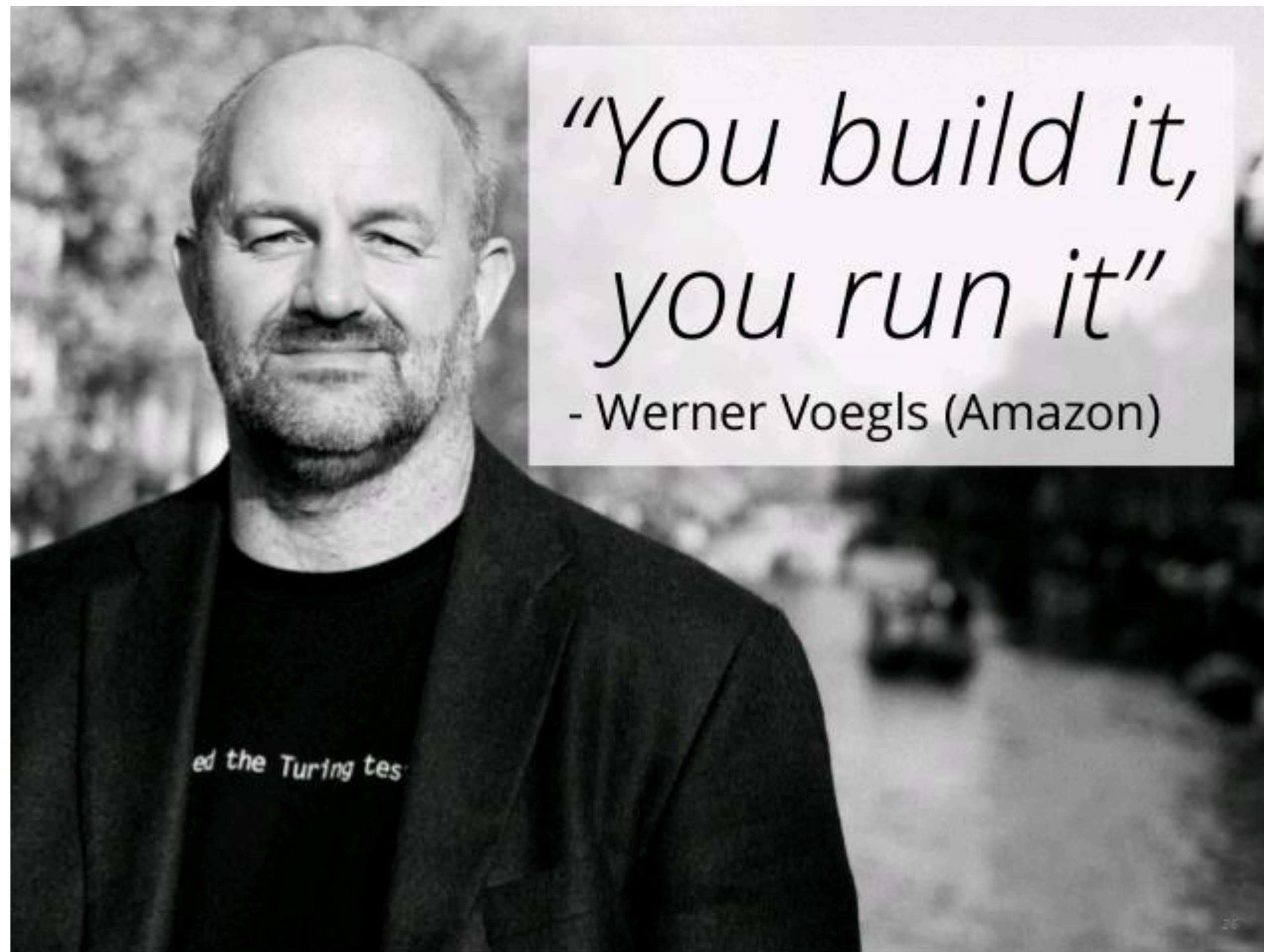
**Lord Kelvin**



*给开发人员一把测量反馈“尺”*



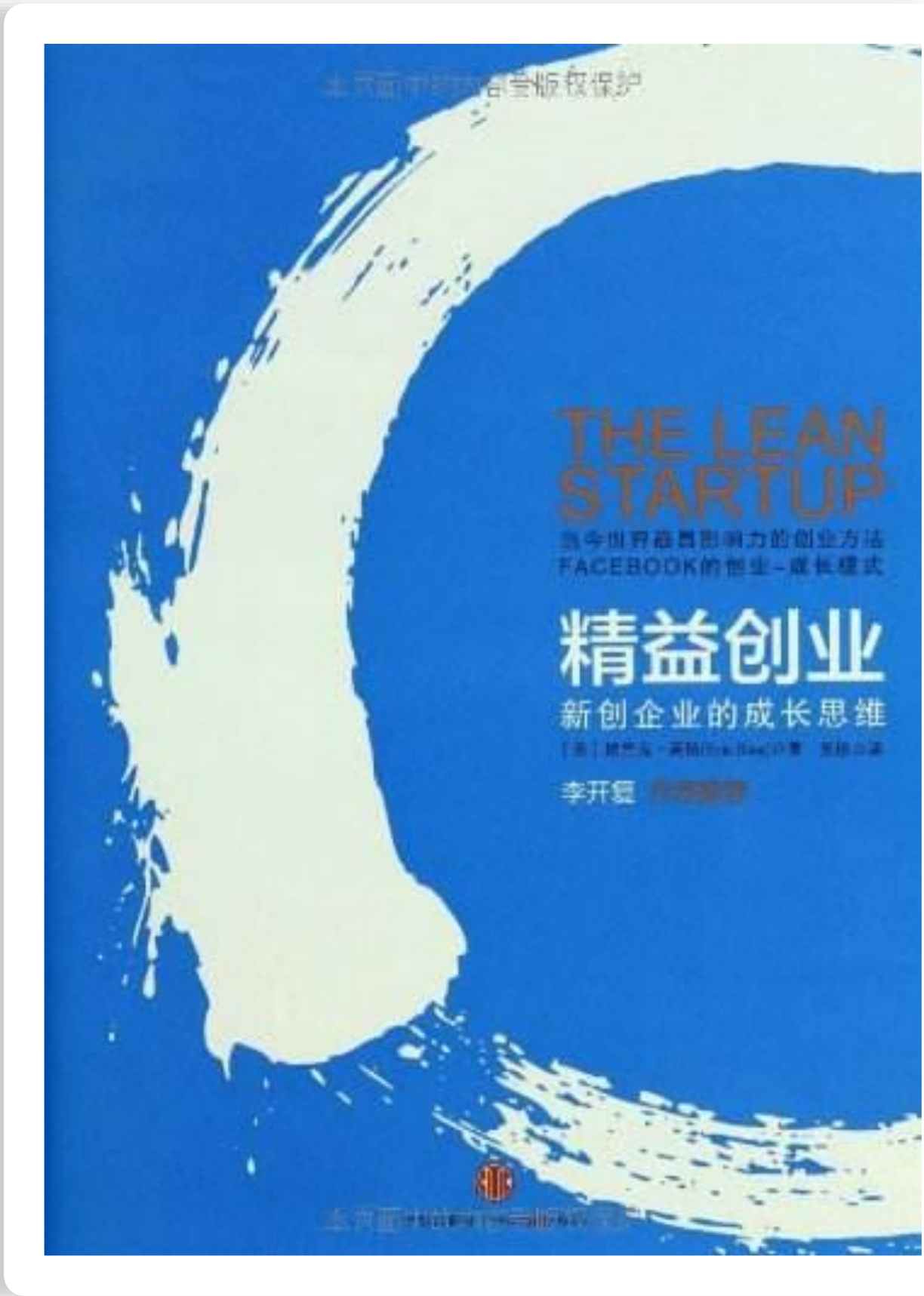
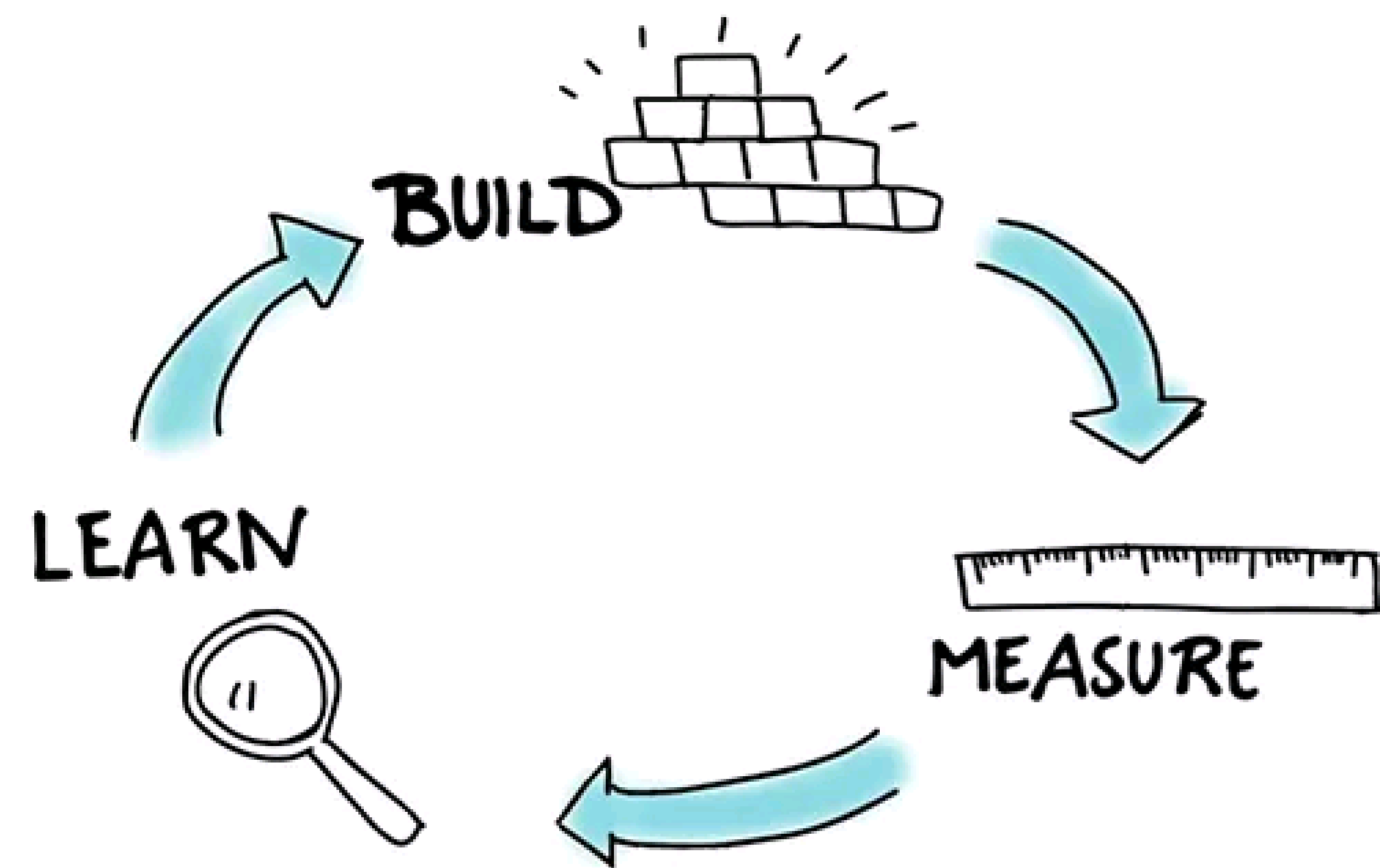
# DevOps实践：研发自助监控



***You build it, you  
run it,  
you monitor it.***

**– 架构师杨波**

# 精益创业反馈环



# 现状

Business

月度和年度报表  
财务报表  
Google Analytics

Dev

?

QA

测试覆盖报表  
CI持续集成报表  
性能测试报表

Ops

CPU, Mem, IO,  
Disk  
  
Zabbix/Ganglia  
/Nagios



# 问题

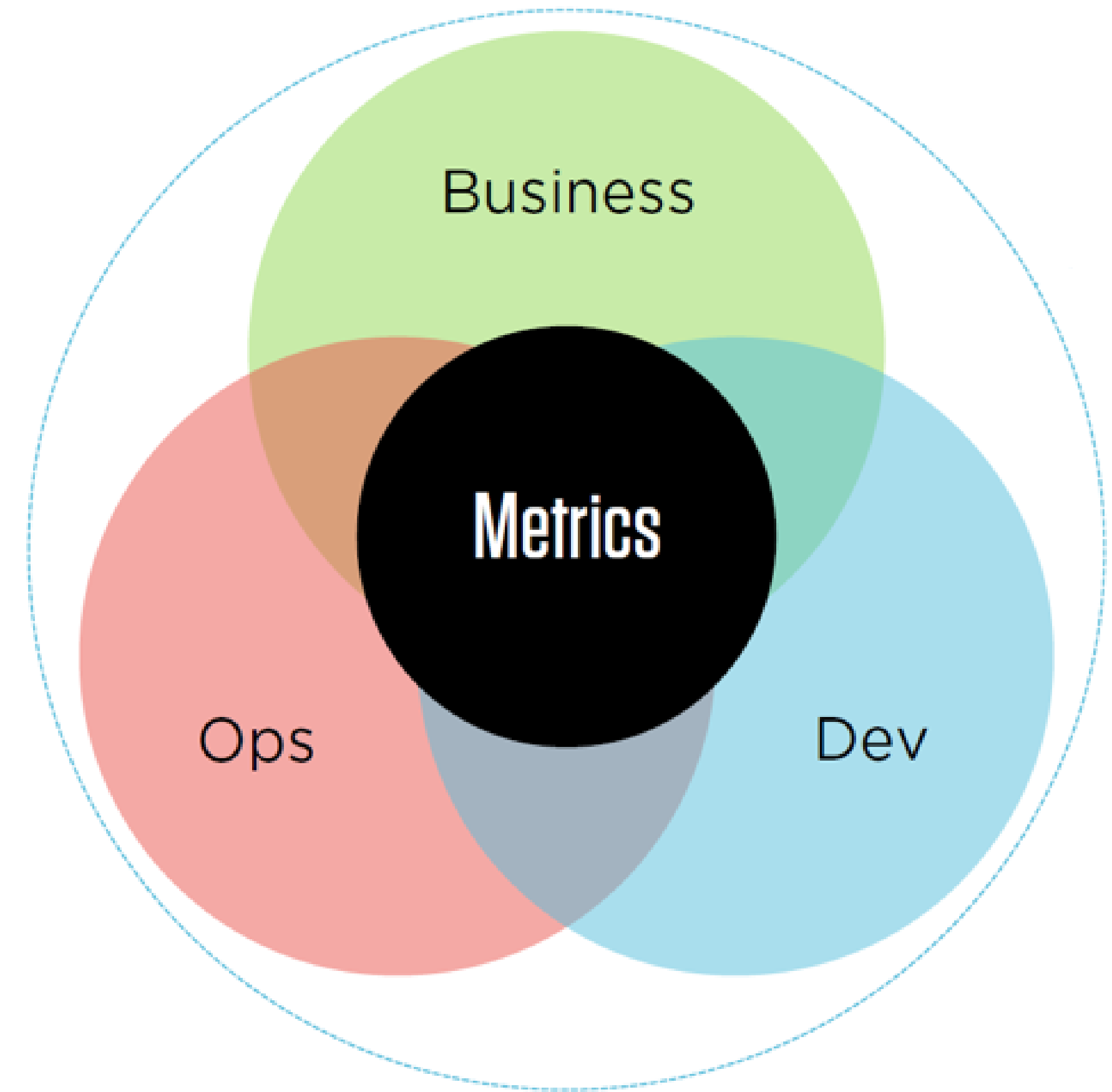
- 运维人员只专注系统监控(日志, 负载度量), 没有应用监控能力和上下文
- 开发人员只管实现功能, 没有DevOps和度量意识
- 应用监控空白, 对应用状态无感知, 主要靠蒙
- 业务对关键应用指标无感知, 很多功能开发了也无人用





# Metrics Driven Development(MDD)

- “... is a practice where metrics are used to drive the entire application development”
- 是一种实践，主张整个应用开发由度量指标驱动
  - InfoQ/2012
  - <https://www.infoq.com/articles/metrics-driven-development>

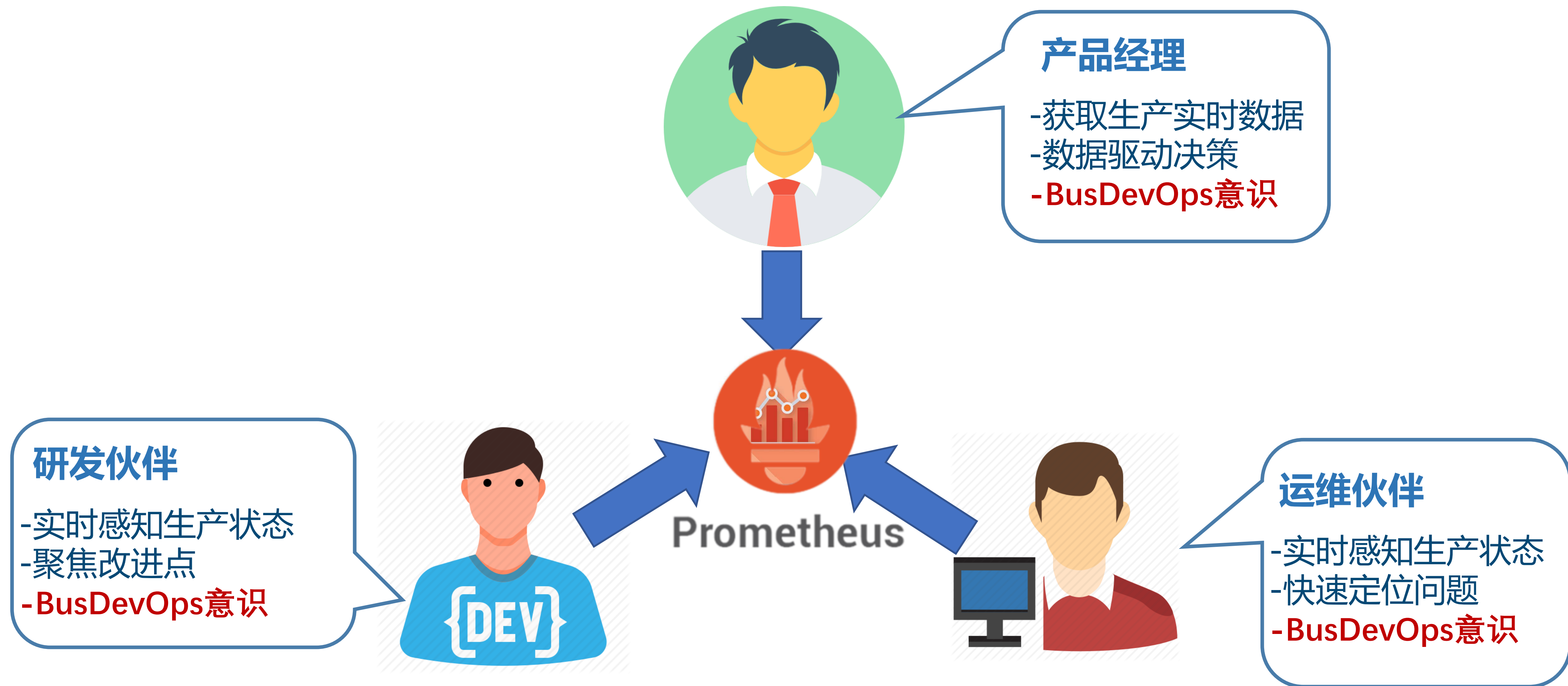


# MDD核心原理

- **开发功能前先定义度量指标(Bus/Dev/Ops)**
  - Define metrics before development
- **开发人员自助埋点**
  - Instrumentation-as-Code
- **真实性的唯一来源**
  - Single Source of Truth
- **关键指标的共同视角**
  - Shared view of key metrics
- **使用度量进行决策**
  - Use metrics when making decisions
- **开发持续维护指标**
  - Maintain and follow metrics



# MDD好处



# 第 4 部分

## Prometheus简介

# 什么是Prometheus

- 开源**监控工具**
- **时间序列数据库**TSDDB, golang实现
- **Soundcloud**研发, 源于谷歌**borgmon**
- **多维度**(标签), **拉模式**(Pull-based)
- **白盒&黑盒**监控都支持, **DevOps**友好
- **Metrics & Alert**, 不是 logging/tracing
- **社区生态丰富**(多语言, 各种exporters)
- **单机性能**
  - 消费百万级时间序列
  - 上千个targets

<https://prometheus.io/>



 Open Source

Prometheus is 100% open source and community-driven. All components are available under the [Apache 2 License](#) on [GitHub](#).

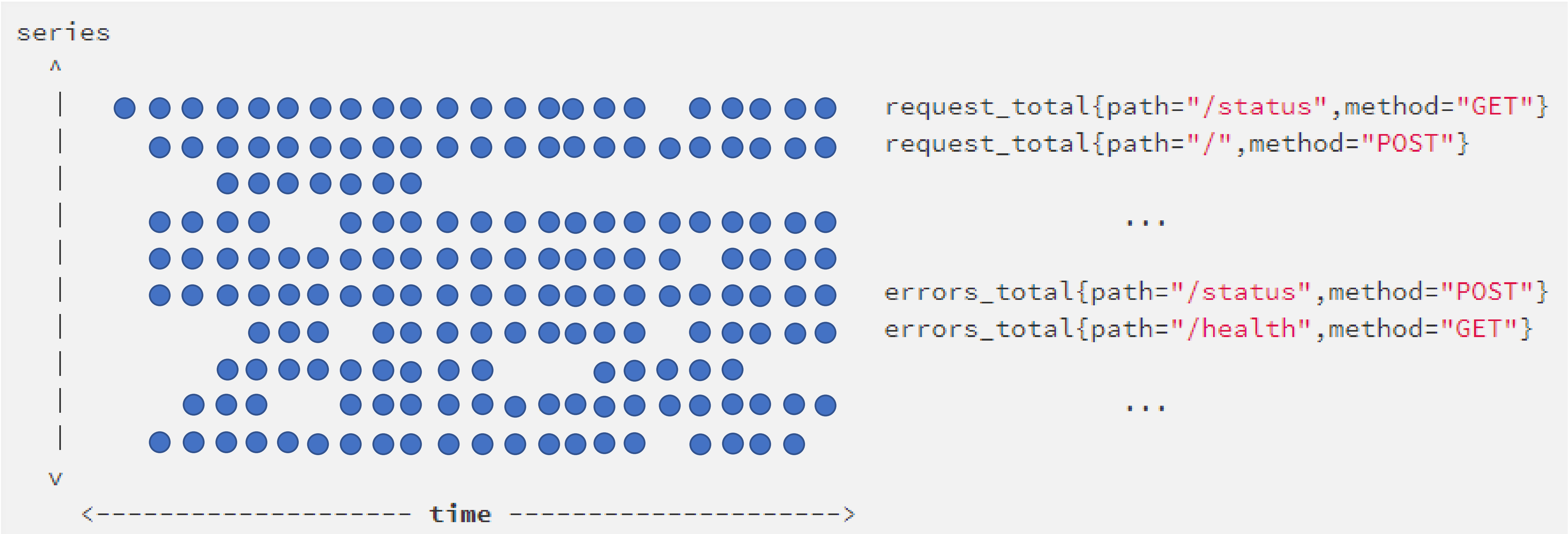
 Star **19,616**

We are a [Cloud Native Computing Foundation](#) member project.

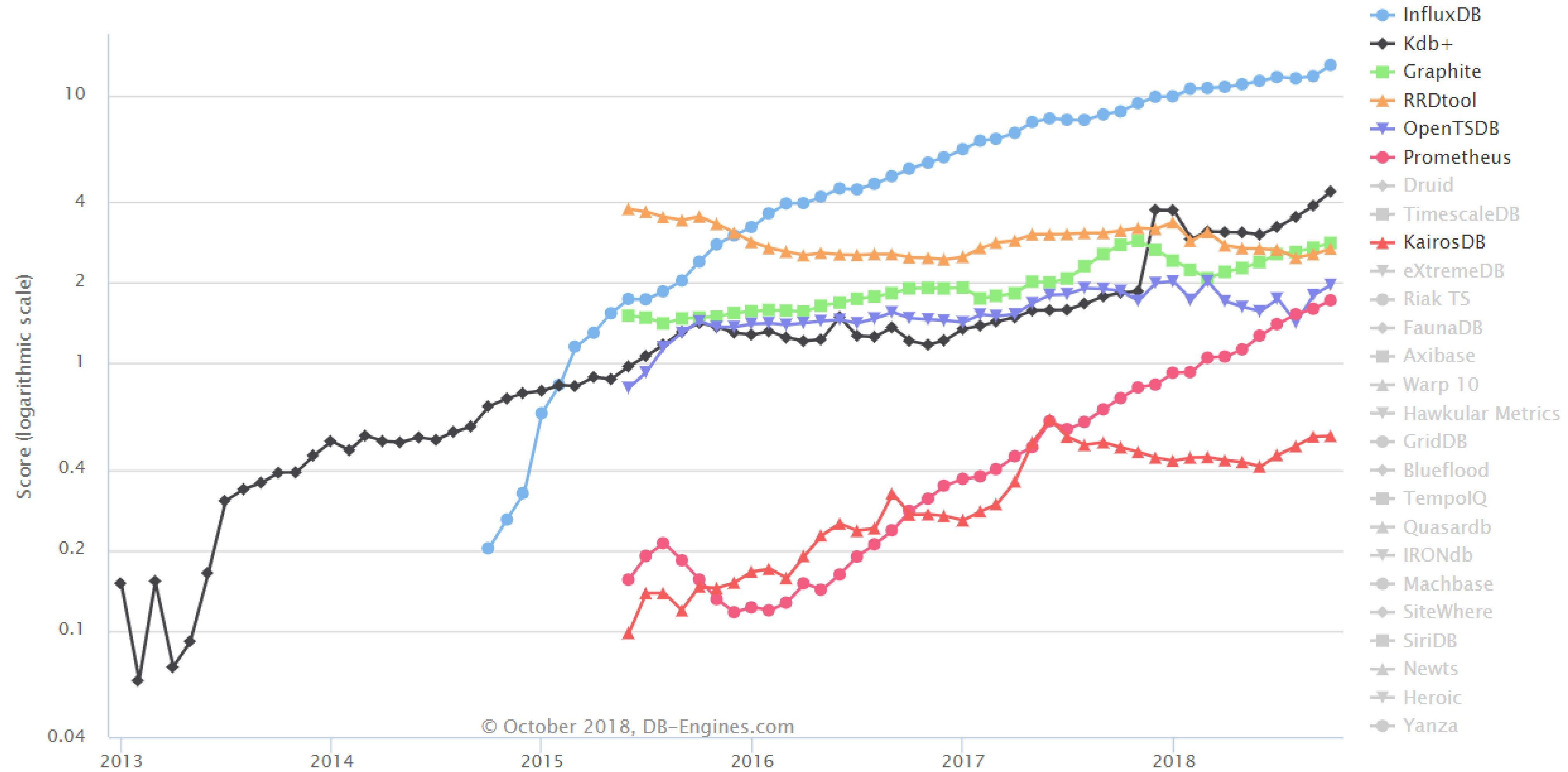


# 时间序列(Time Series)

(t0, v0), (t1, v1), (t2, v2), .....



## DB-Engines Ranking of Time Series DBMS



[https://db-engines.com/en/ranking\\_trend/time+series+dbms](https://db-engines.com/en/ranking_trend/time+series+dbms)



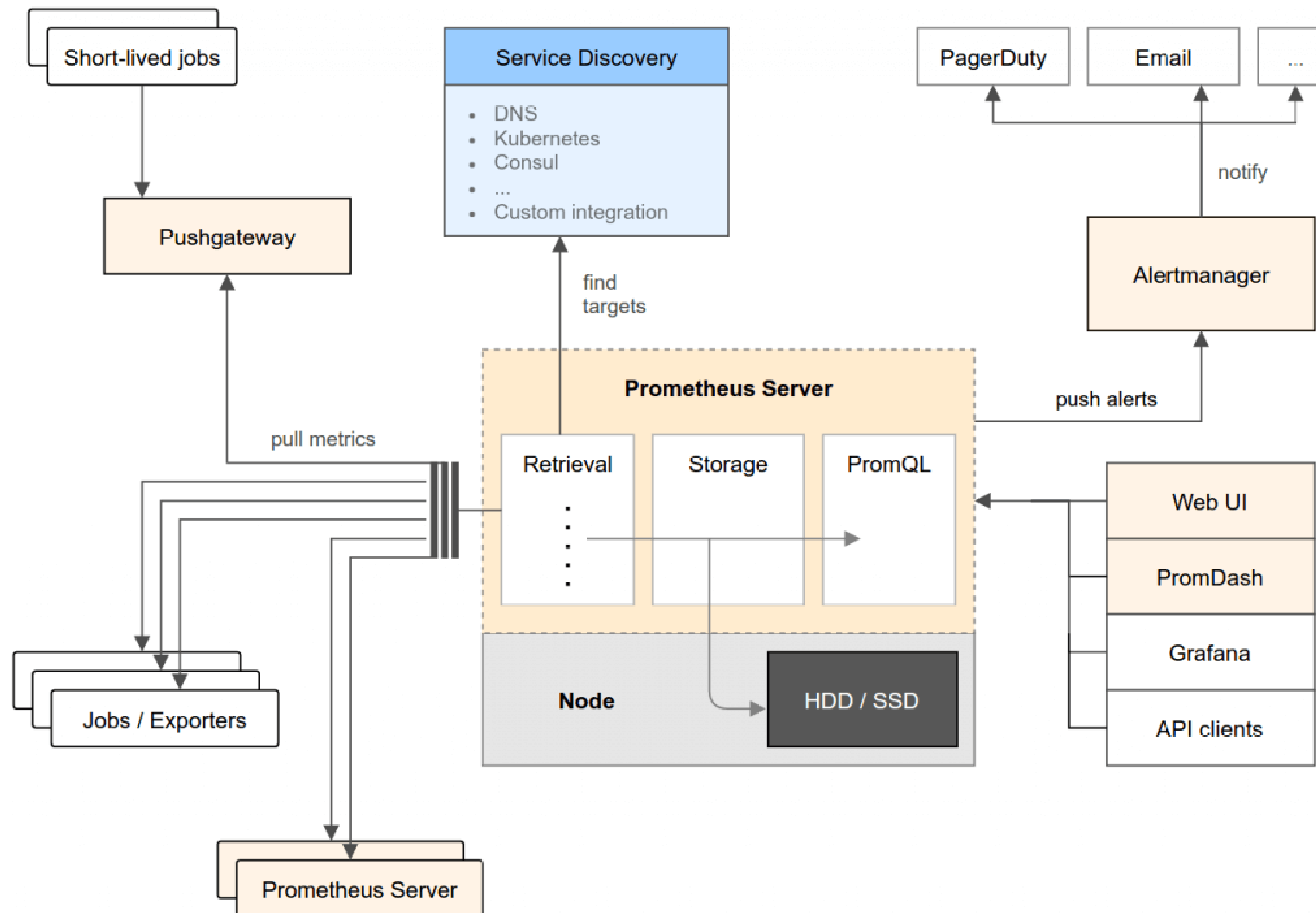
第

5

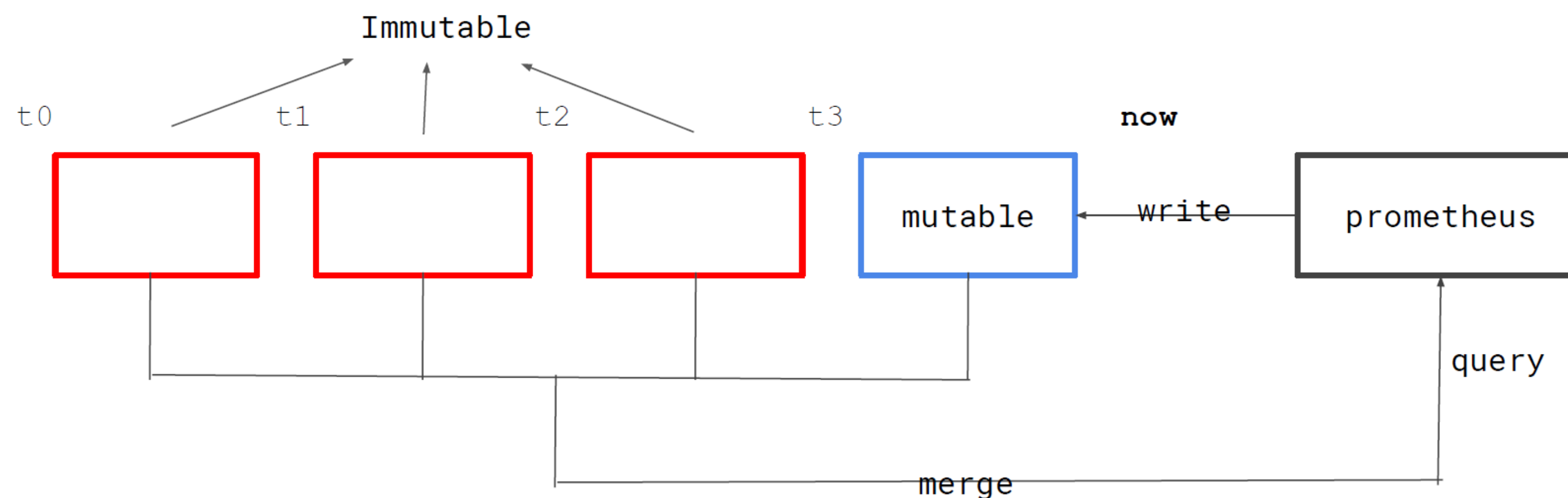
部分

# Prometheus架构设计

# Prometheus架构设计



# Prometheus 2.0存储设计



<https://coreos.com/blog/prometheus-2.0-storage-layer-optimization>

```
prom-prom tree data
data
├── 01BW78CHMFQ1NZ7YD7M2PSYDVW
│   ├── chunks
│   │   └── 000001
│   ├── index
│   ├── meta.json
│   └── tombstones
├── 01BW78CJVV9YKEFCWKJPAQ0M4X
│   ├── chunks
│   │   └── 000001
│   ├── index
│   ├── meta.json
│   └── tombstones
├── 01BW7F88WFZD9BQZKRCRHRM90Y
│   ├── chunks
│   │   └── 000001
│   ├── index
│   ├── meta.json
│   └── tombstones
├── lock
├── wal
│   ├── 000001
│   └── 000003
7 directories, 15 files
```

第

6

部分

# Prometheus基本概念

# Metrics种类

- **Counter(计数器)**

- 始终增加
- http请求数, 下单数



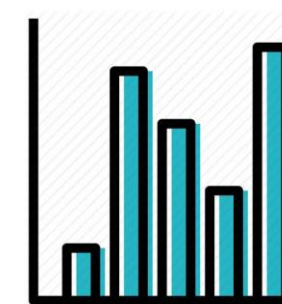
- **Gauge(测量仪)**

- 当前值的一次快照(snapshot)测量, 可增可减
- 磁盘使用率, 当前同时在线用户数



- **Histogram(直方图)**

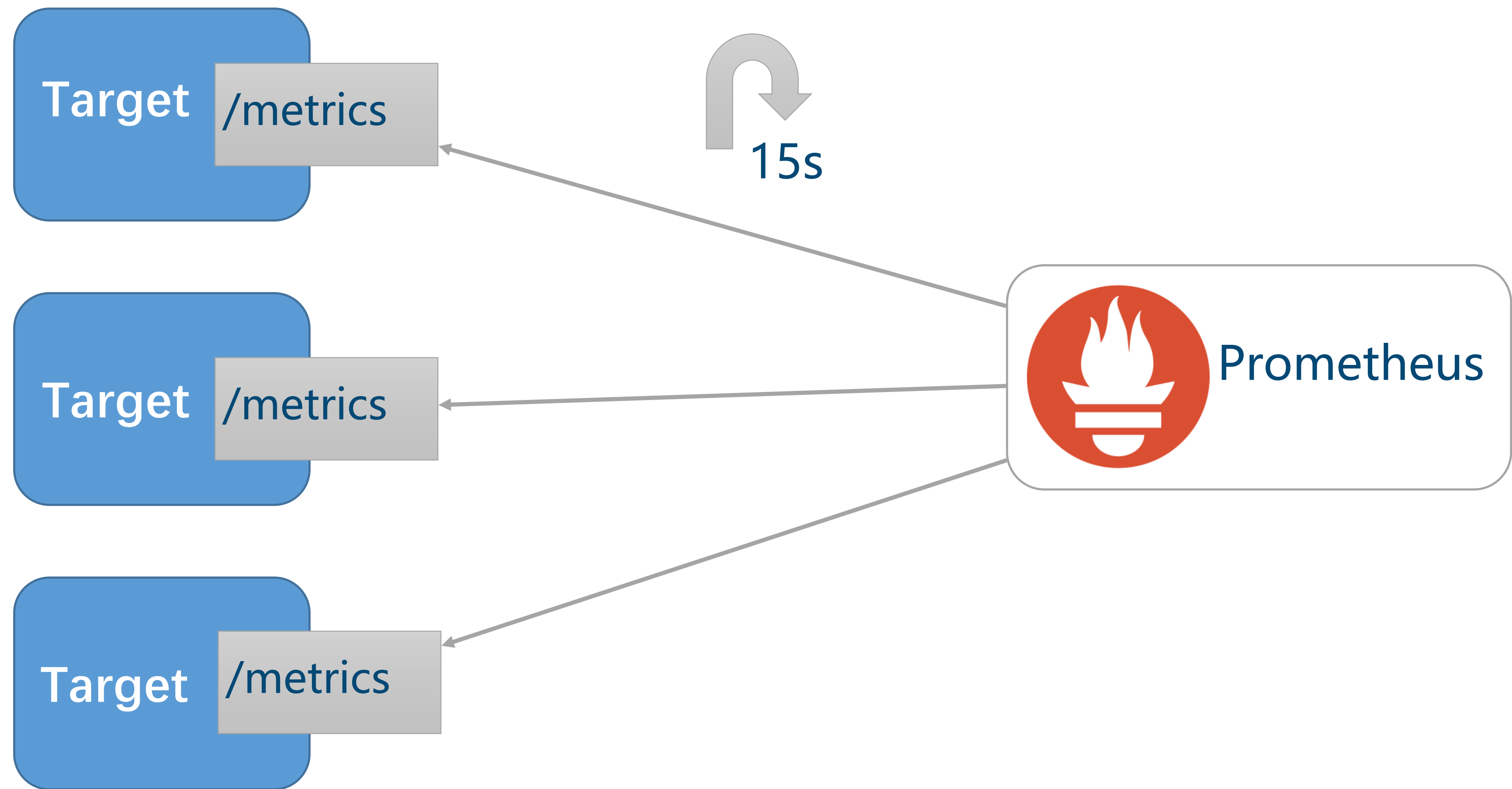
- 通过分桶(bucket)方式统计样本分布



- **Summary(汇总)**

- 根据样本统计出百分位
- 客户端计算

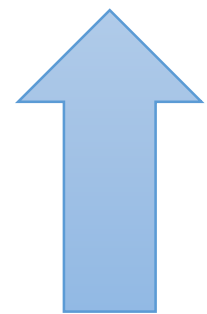






**# HELP http\_requests\_total Total number of HTTP requests made.**  
**# TYPE http\_requests\_total counter**

**http\_requests\_total{code="200", path="/status"} 8**



Metric Name



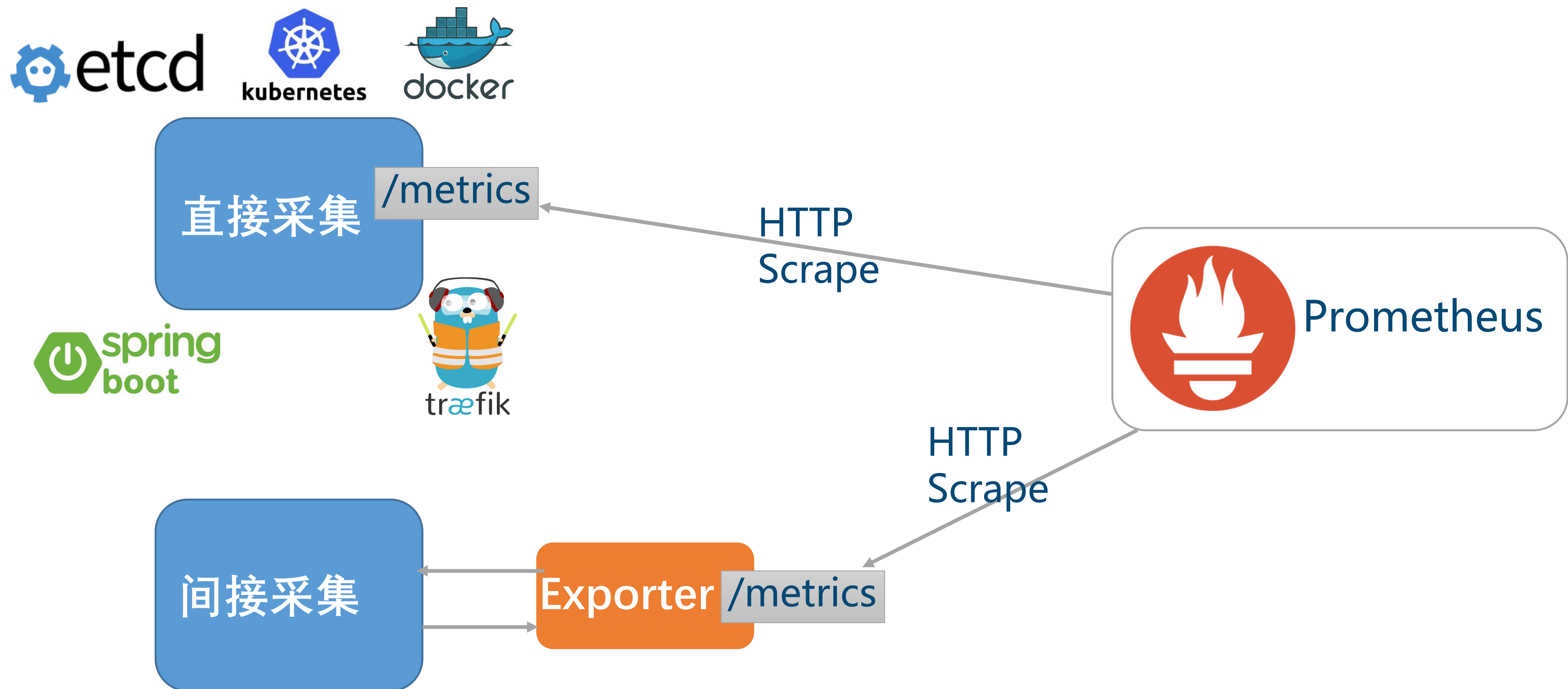
Label



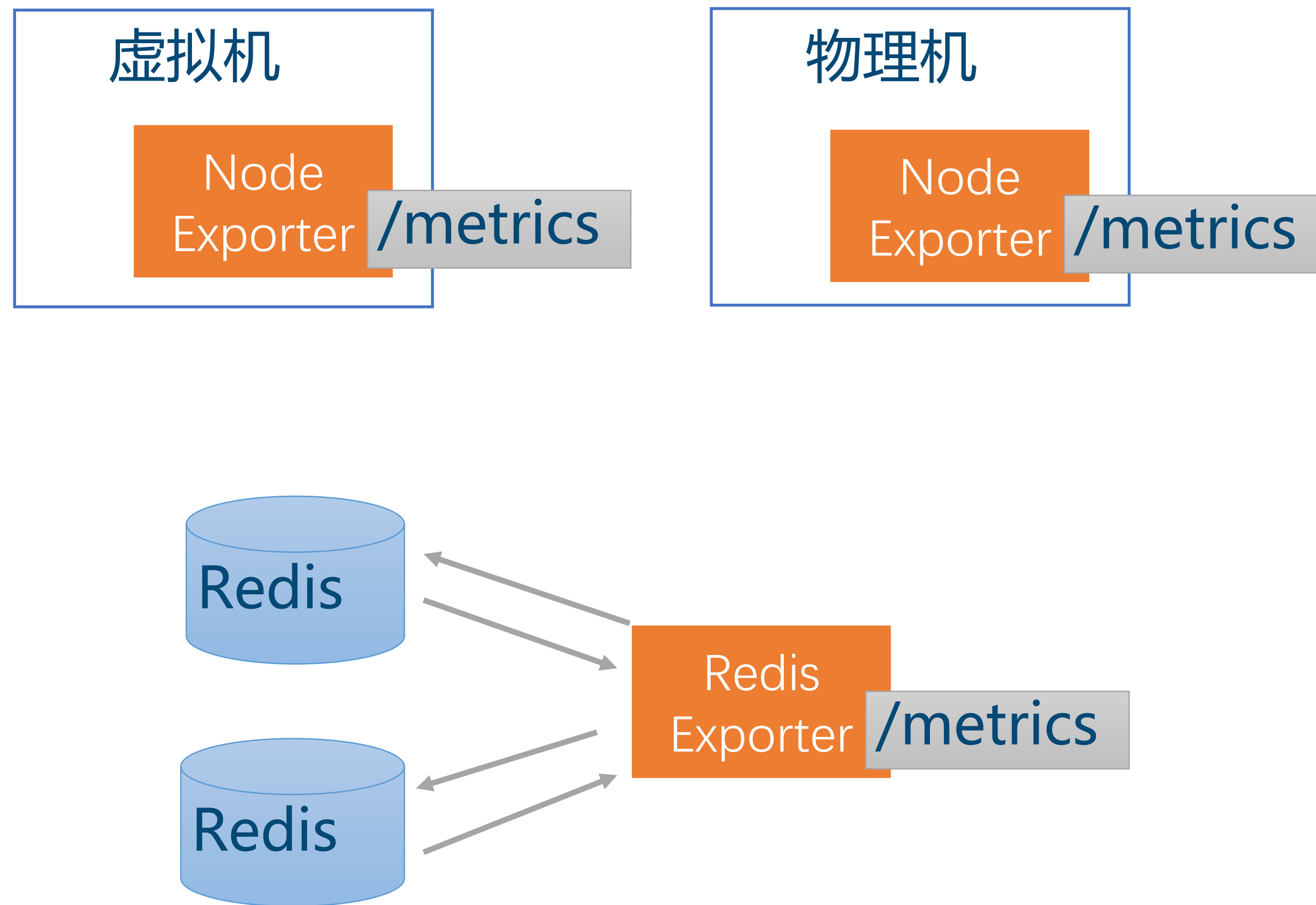
Value



# Scrape metric

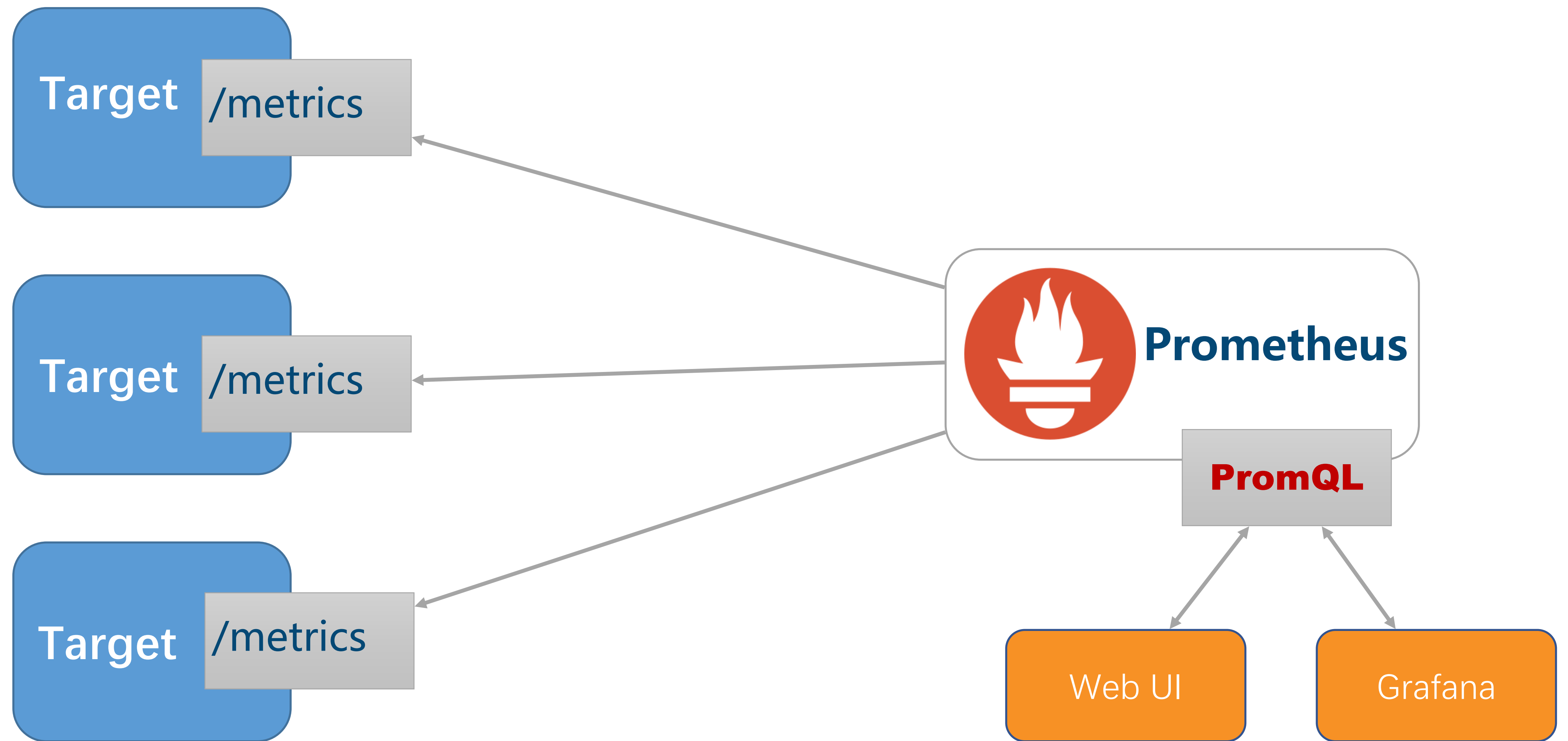


# Exporters



- **OS – Node Exporter**
  - Linux, Windows
- **Database**
  - Mysql, Postgres, CouchDB...
- **Messaging**
  - **Kafka**, RabbitMQ, NATS...
- **Logging**
  - ElasticSearch, Fluentd, Telegraf...
- **Key-Value**
  - Redis, Memcached...
- **WebServer**
  - Apache, Nginx...
- **Proxy**
  - Haproxy, Varnish...
- **DNS**
  - BIND, PowerDNS, Unbound
- **BlackBox**

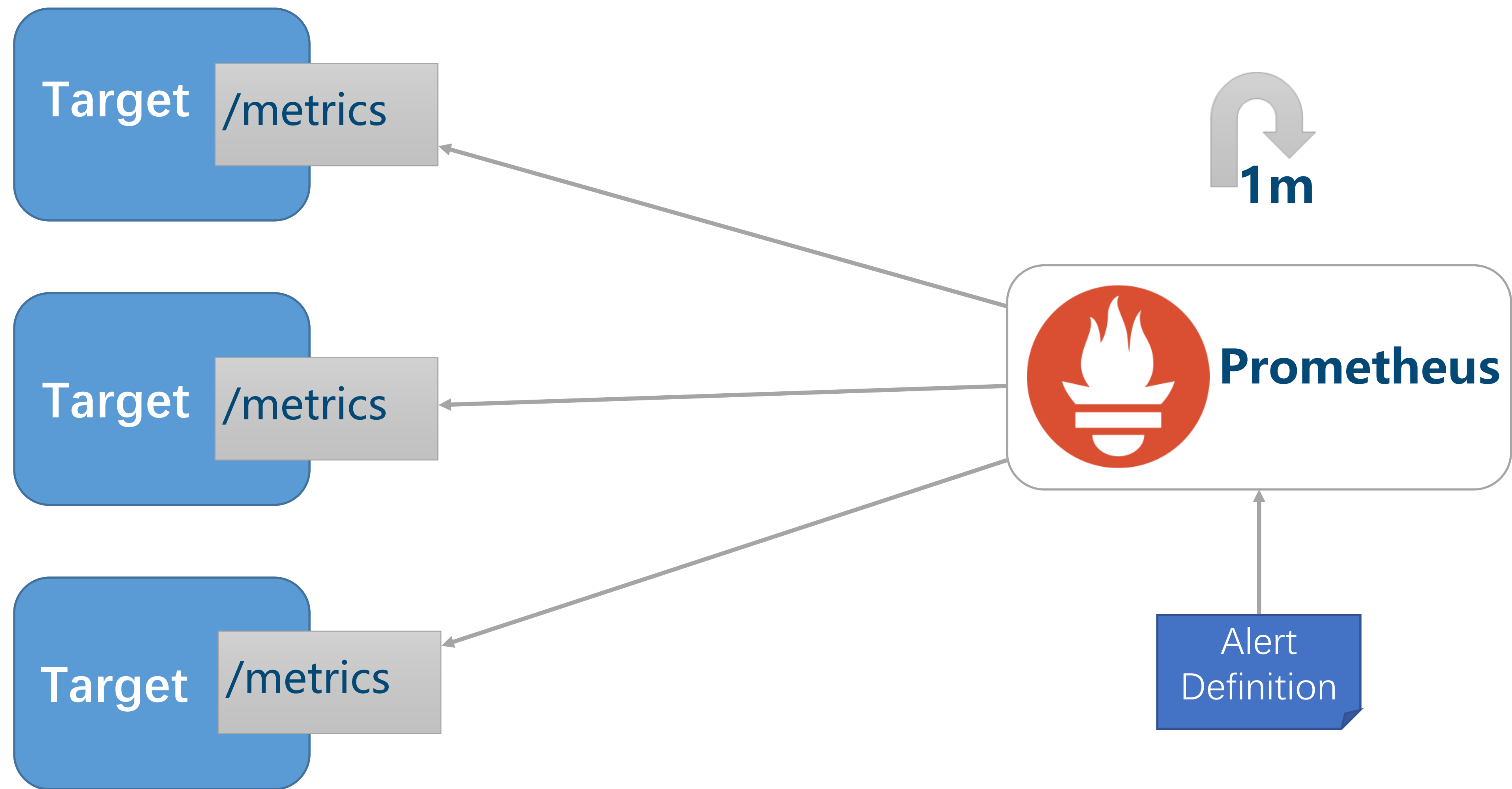
<https://prometheus.io/docs/instrumenting/exporters/>



# HTTP错误百分比

**sum(rate(http\_requests\_total{status="500"}[5m])) by (path) /  
sum(rate(http\_requests\_total[5m])) by (path)**

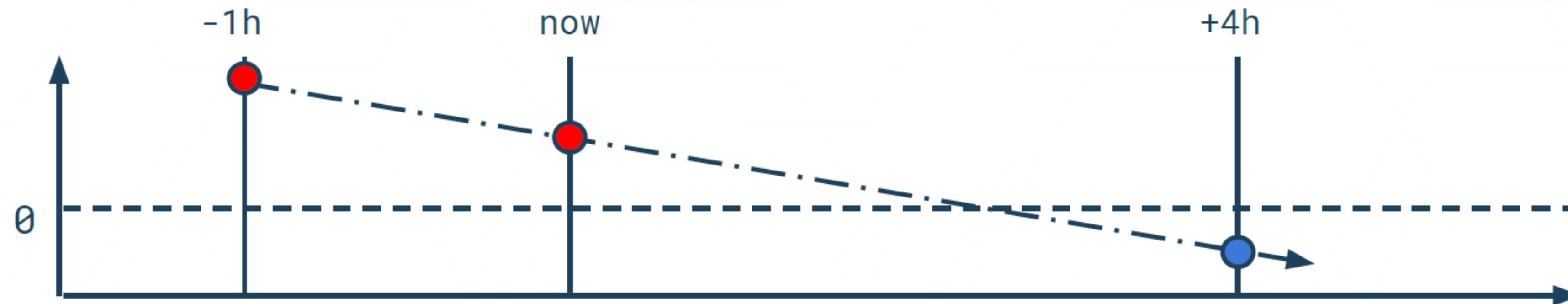
<b>{path="/status"}</b>	<b>0.0039</b>
<b>{path="/"}</b>	<b>0.0011</b>
<b>{path="/api/v1/topics/:topic"}</b>	<b>0.087</b>
<b>{path="/api/v1/topics}</b>	<b>0.0342</b>

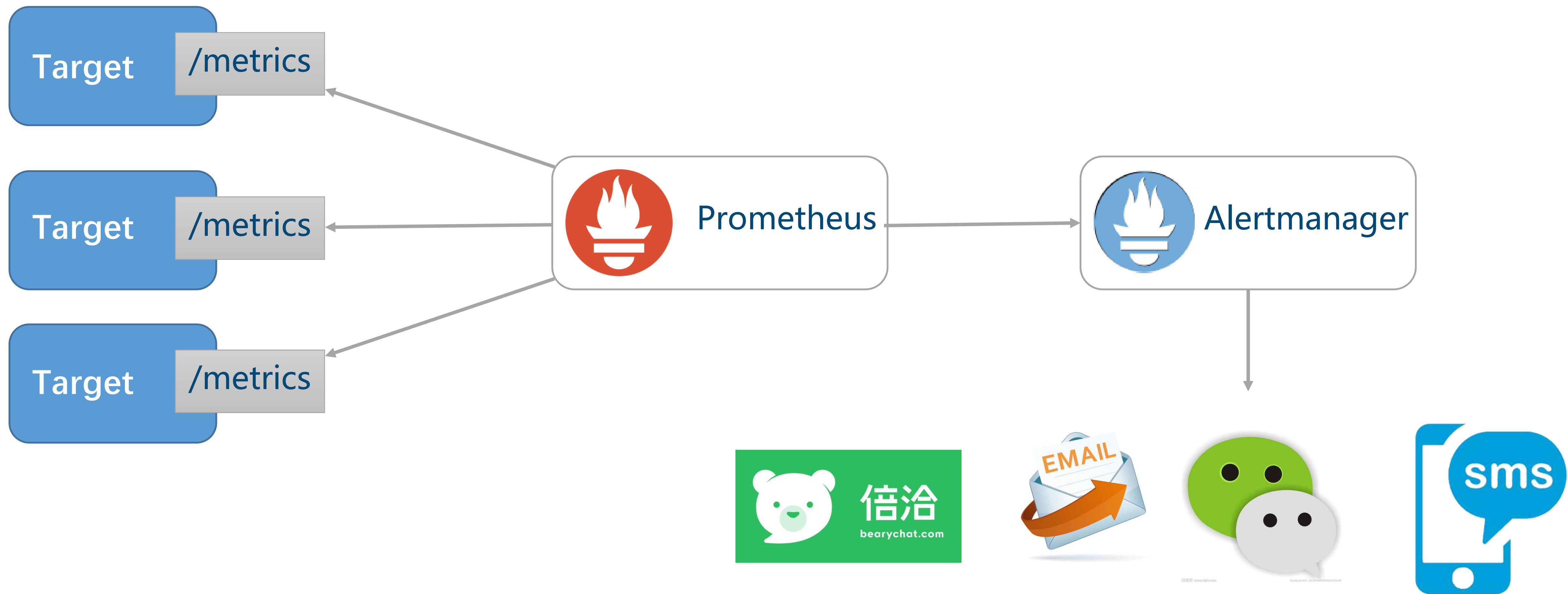


# 4小时内磁盘是否会满?

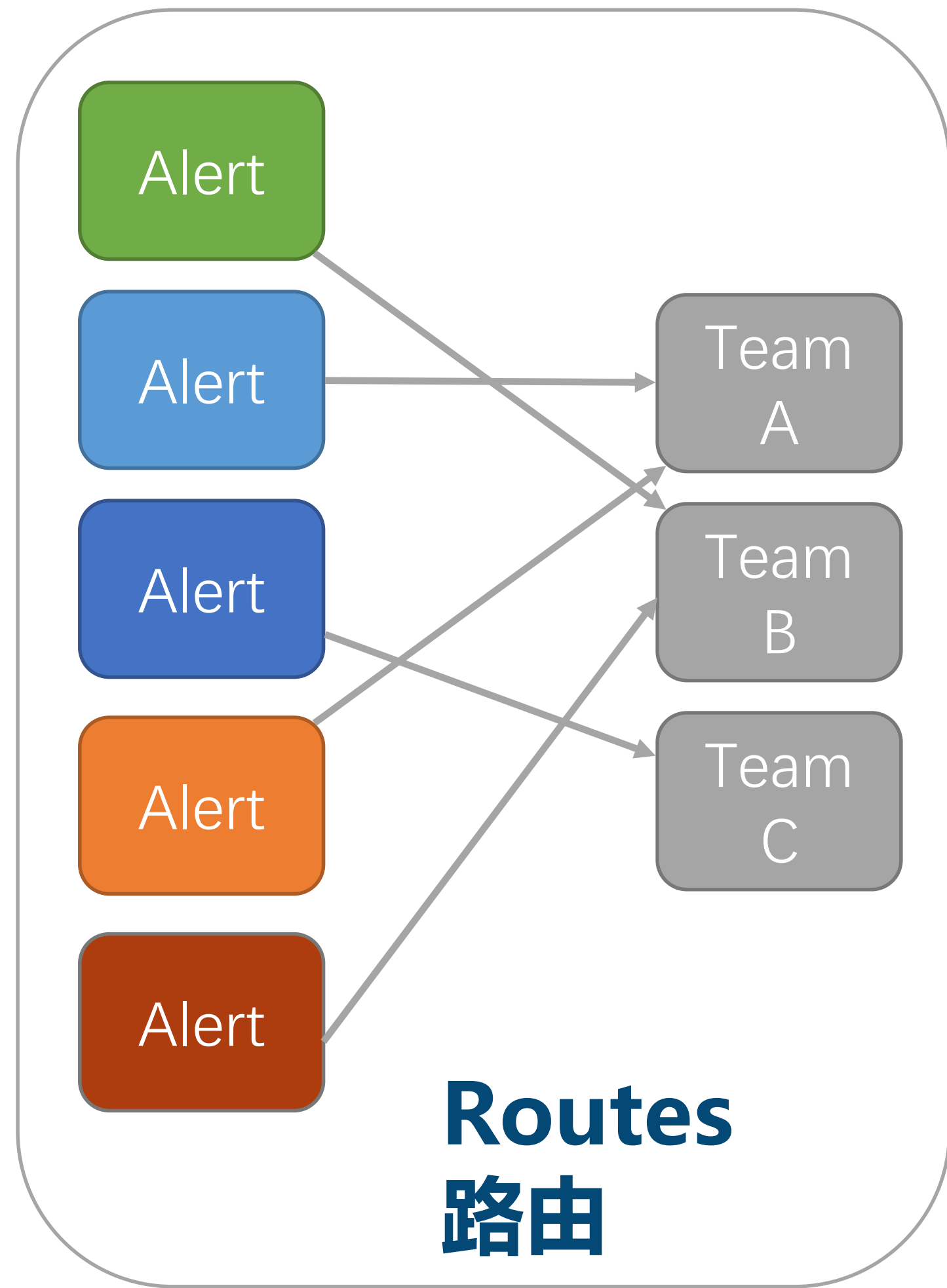
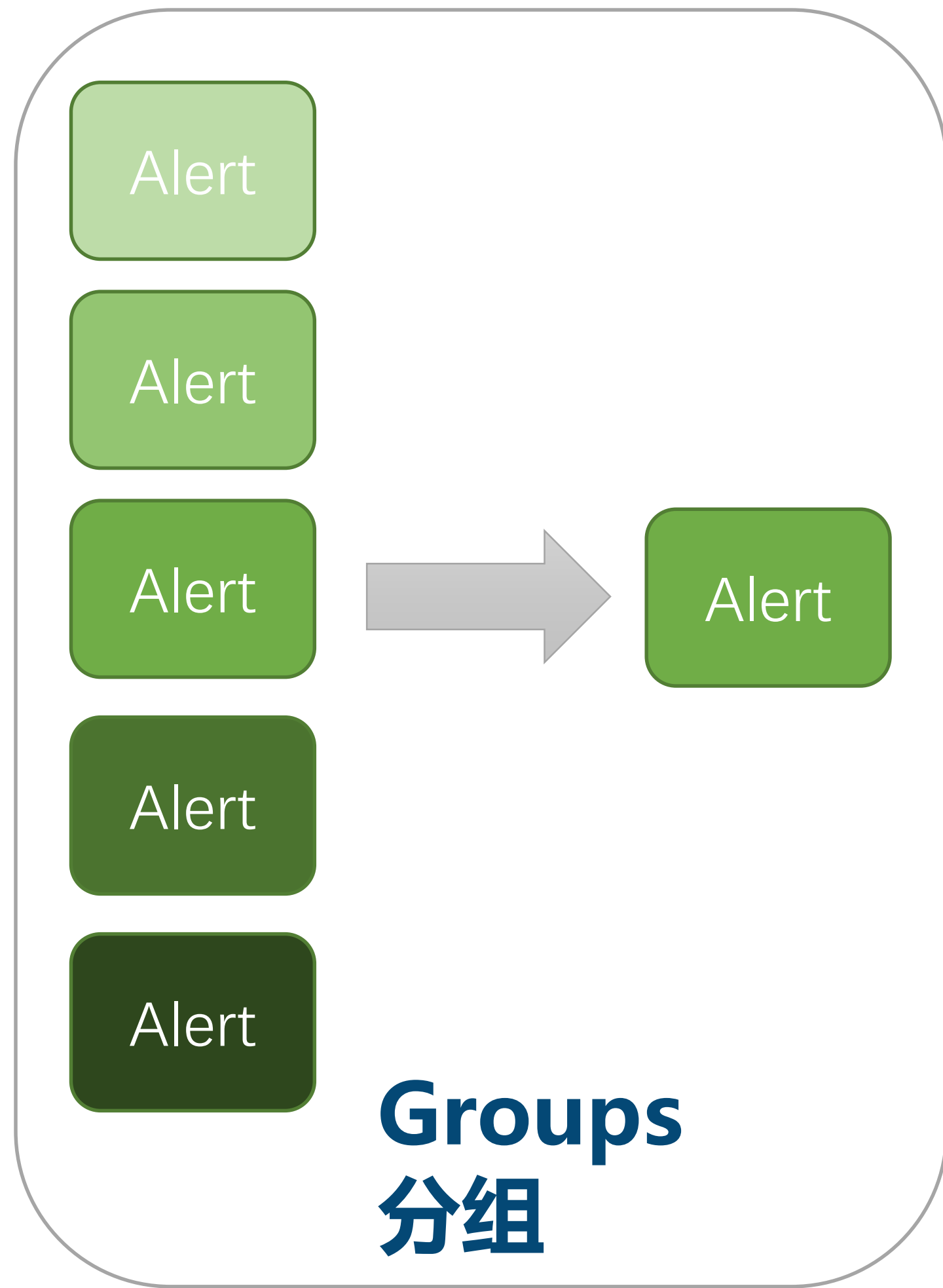
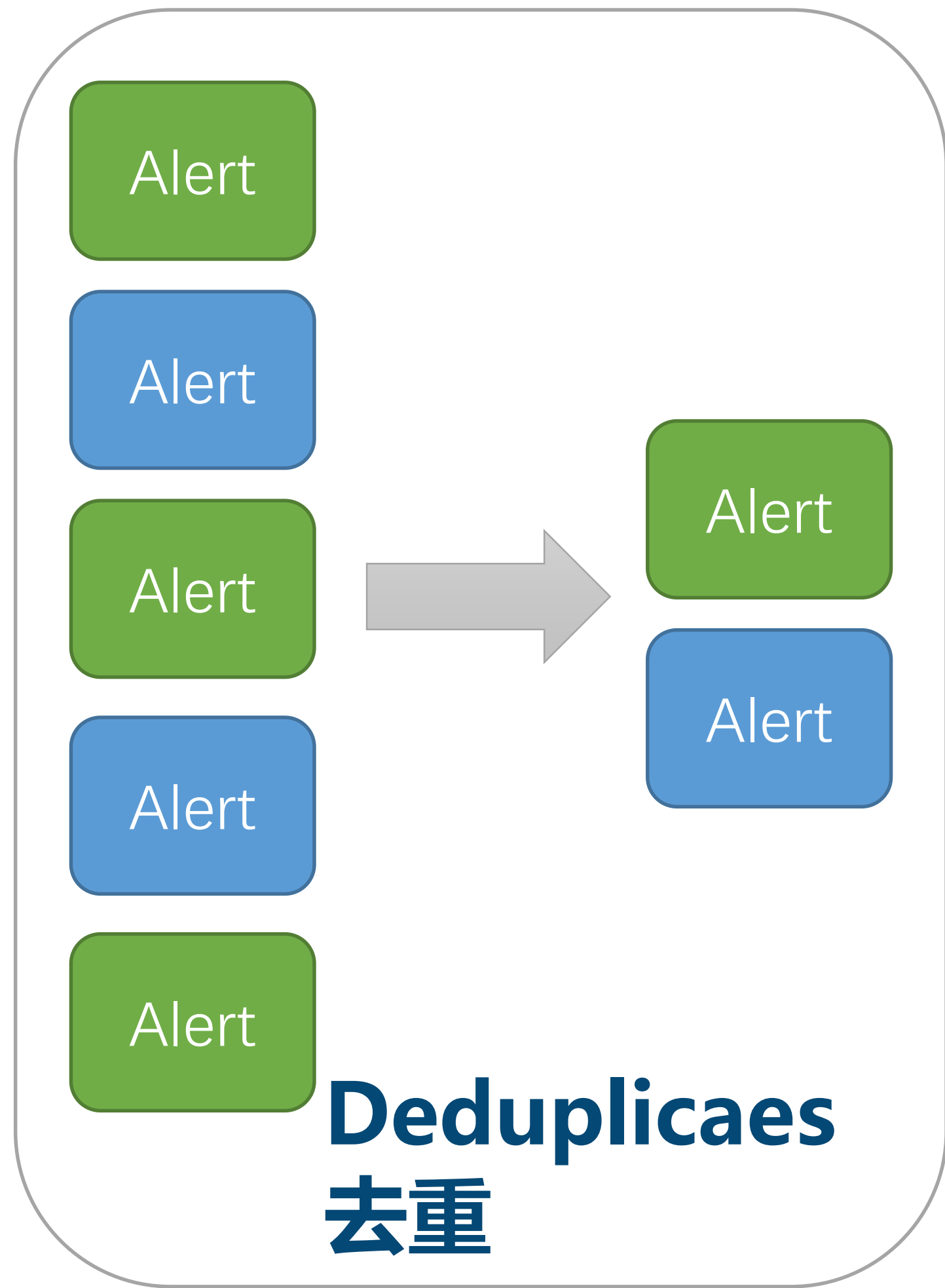
**ALERT DiskWillFullIn4Hours**

**IF  $\text{predict\_linear}(\text{node\_filesystem\_free}[1h], 4*3600) < 0$**







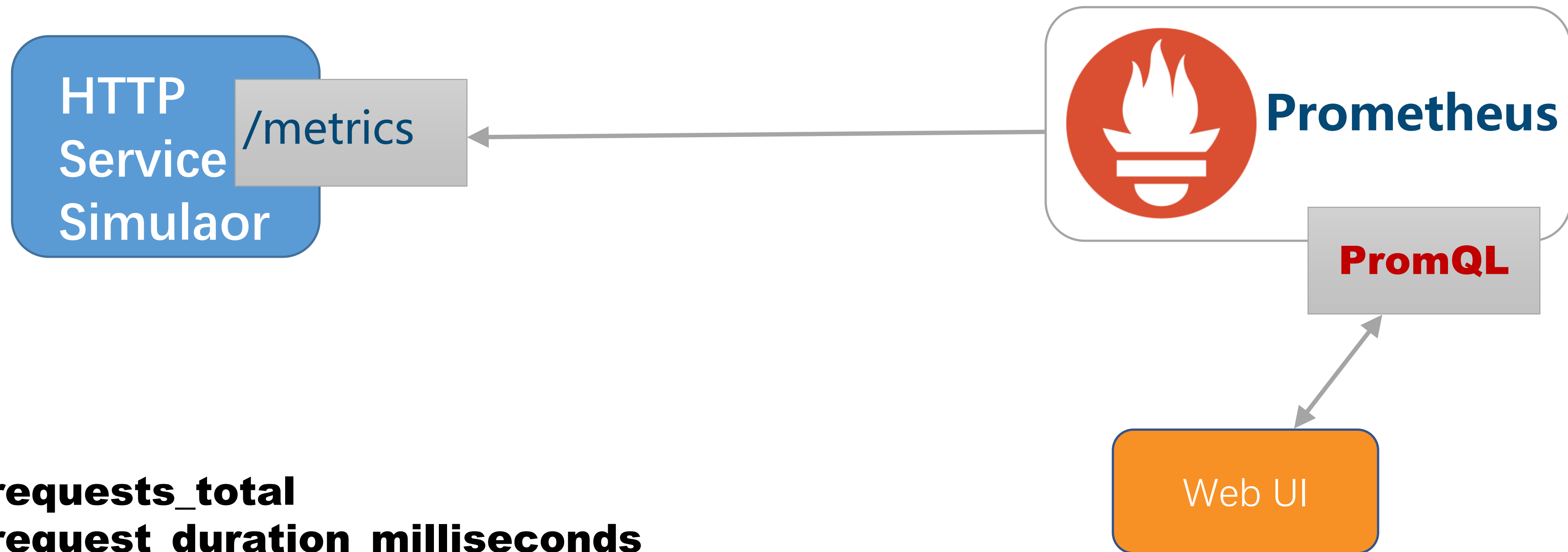


第

7

部分

# Prometheus起步查询实验(Lab01)



**http\_requests\_total**  
**http\_request\_duration\_milliseconds**

# 思考练习

- **Request Rate**

- 求http-simulator的总体请求率(1分钟滚动窗口)
- 求每分钟多少是错误请求
- 求对/users端点的错误请求率

- **Latency distribution**

- 求延迟中位数
- 求错误请求的90百分位延迟，按端点分组
- 对/users端点的请求延迟，是否满足3个9都在400ms以内的SLO?



第

8

部分

## Prometheus+Grafana展示实验 (Lab02)



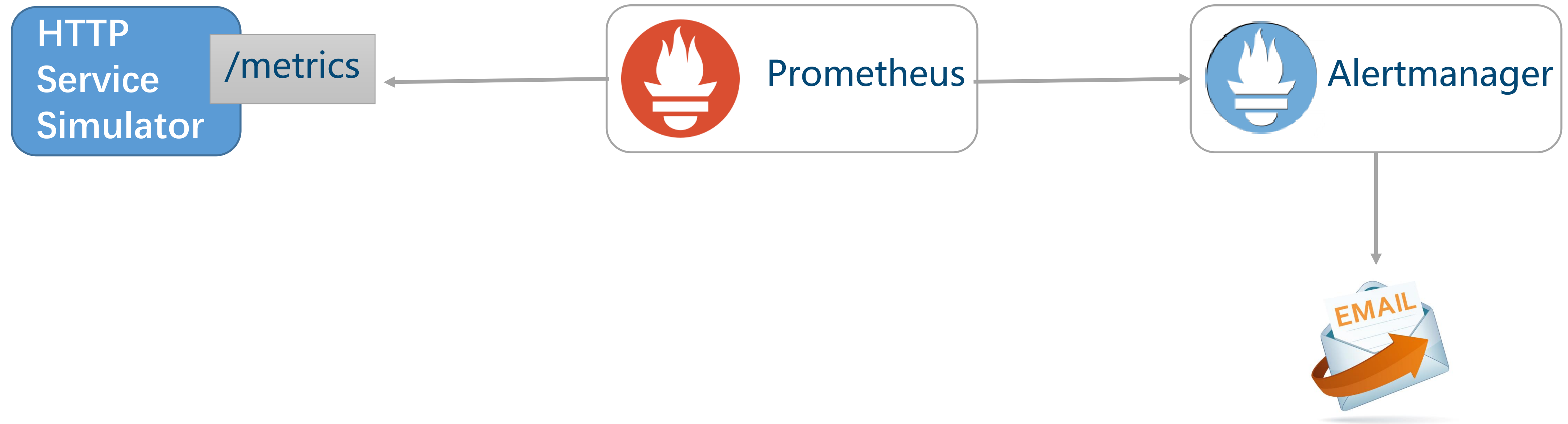
第

9

部分

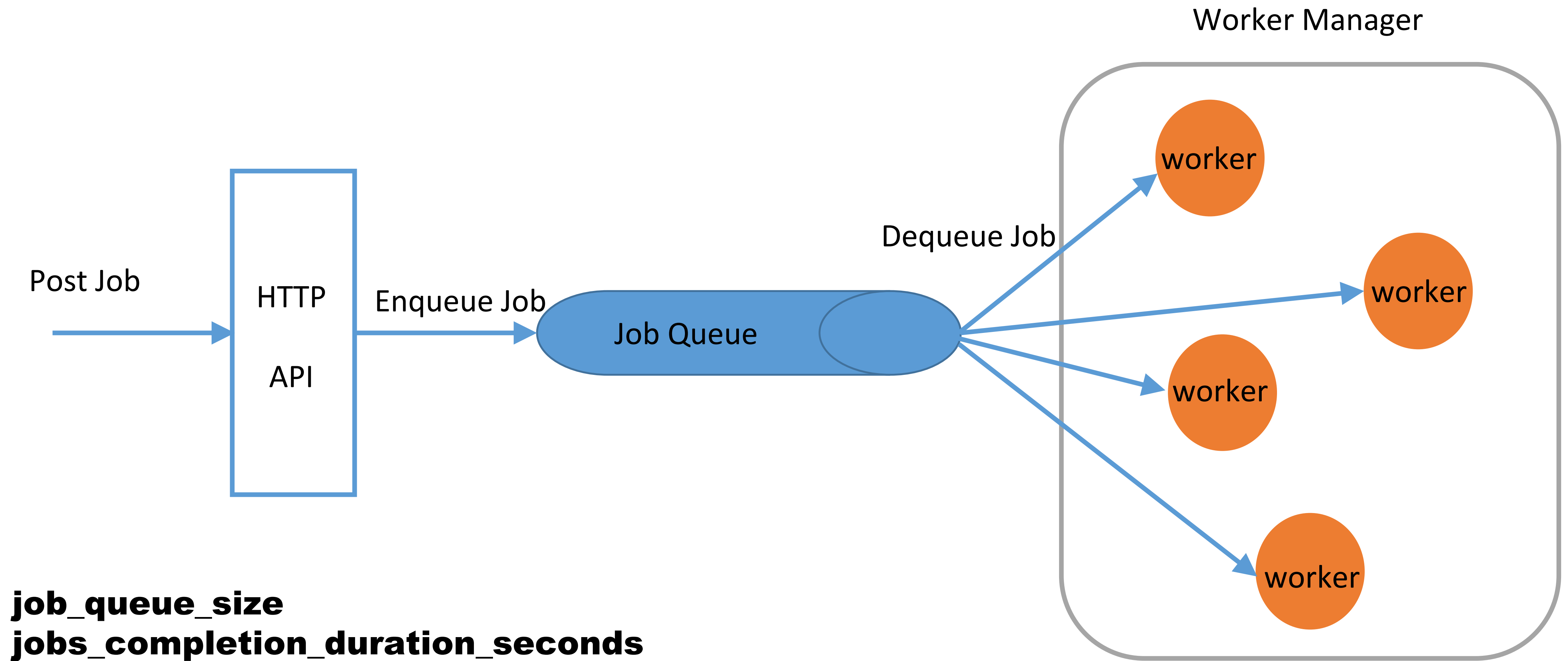
Prometheus+Alertmanager告警实验 (Lab03)





# 第10部分

## Java应用埋点和监控实验 (Lab04)



# 思考练习

- **其它埋点**
  - 完成的job总数
  - 当前的活跃job数
  - 当前的活跃worker数
  - 当前的空闲worker数



# 第11部分

## NodeExporter系统监控实验 (Lab05)

# 第12部分

## Spring Boot Actuator监控实验 (Lab06)

# 第13部分

## Prometheus监控最佳实践



# 四个黄金指标(Google)

## ■ **延迟：** 服务请求所需耗时

- 例如HTTP请求平均延迟

## ■ **流量/吞吐：** 衡量服务容量需求

- 例如每秒处理HTTP请求数

## ■ **错误：** 衡量错误发生的情况

- 例如HTTP 500错误数

## ■ **饱和度：** 衡量资源使用情况

- 例如CPU/内存/磁盘使用量

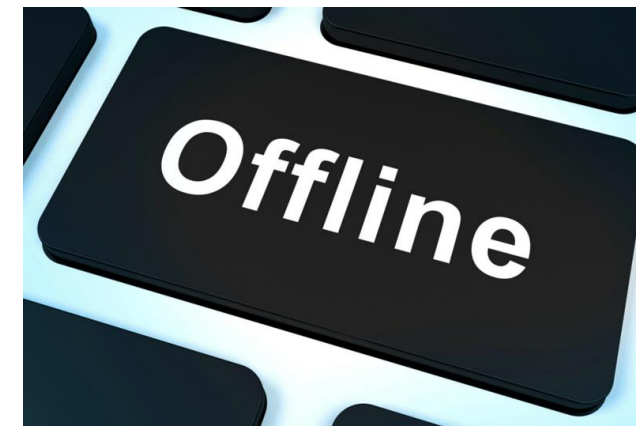


# 三类系统的监控



## Online Serving System

- **RED方法**
  - Requests
  - Errors
  - Duration
- **Cache**
  - 命中率



## Offline Serving System

- **USE方法**
  - Utilization
  - **Saturation**
  - Errors
- **线程池类似**



## Batch Jobs

- **Pushgateway**
- **指标**
  - Job运行时长
  - 每阶段时长
  - 失败/成功数

# Metrics命名

- 一般建议: `library_name_unit_suffix`
- `snake_case`
- **Suffix后缀**
  - `_total` -> `counter`
  - `_counter`, `_sum` -> `summary`
  - `_bucket` -> `histogram`
- **Unit**
  - 无前缀基本单位
  - `seconds`, `bytes`, `ratios`
- **Name&Library**
  - 直接表意
- **Name vs Label**
  - `appName`, `instanceId`/`Ip`

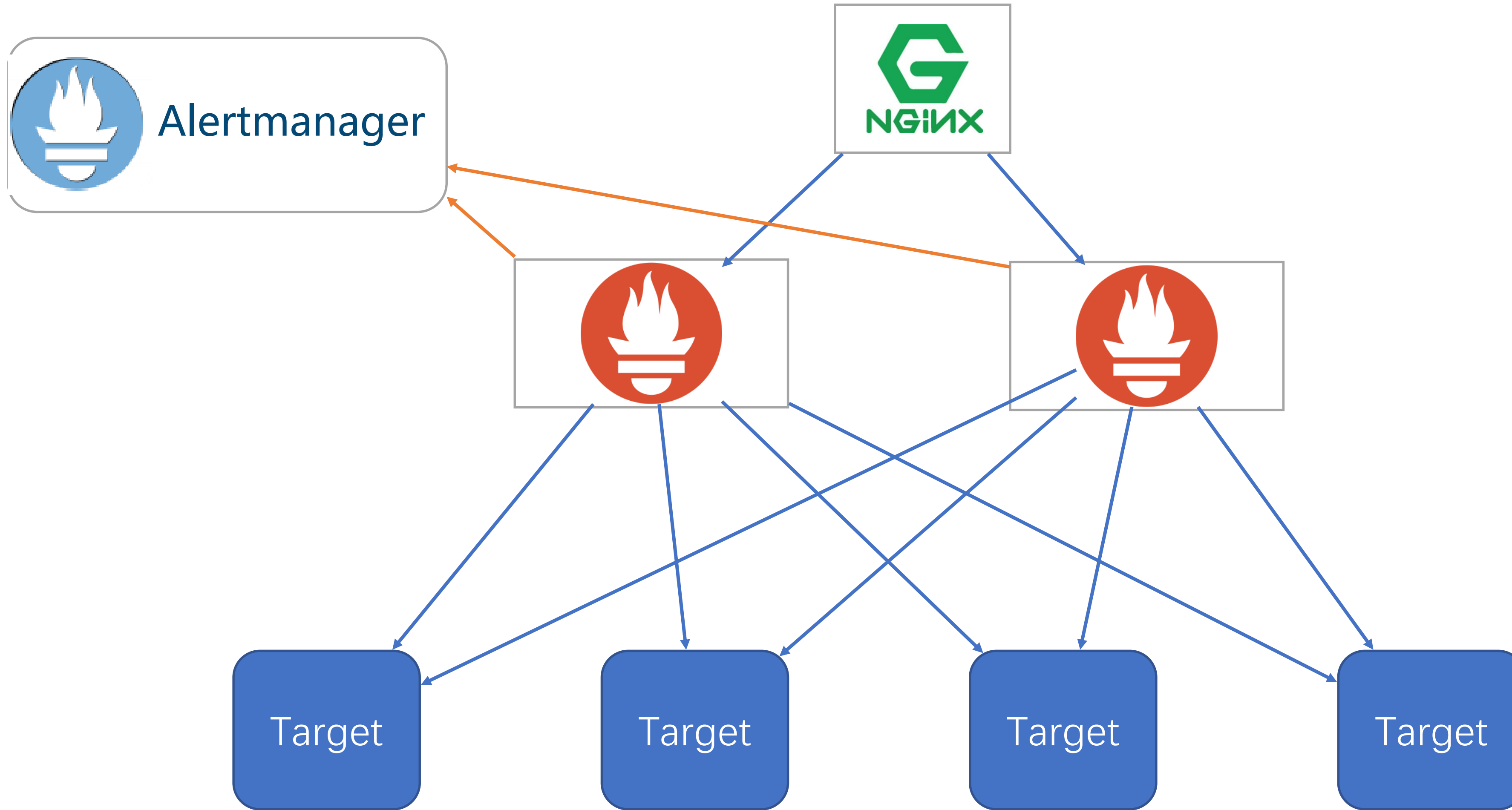


# Cardinality(基数)

- Label的可能取值
- 新增一个Label值=新增一个时间序列
- 经验值：单实例Cardinality  $\leq 10$ 个
- 不适合做Label
  - Email地址
  - 用户名
  - IP地址
  - HTTP Path
- 关注10个最大的metrics
- 高Cardinality场景用Log系统

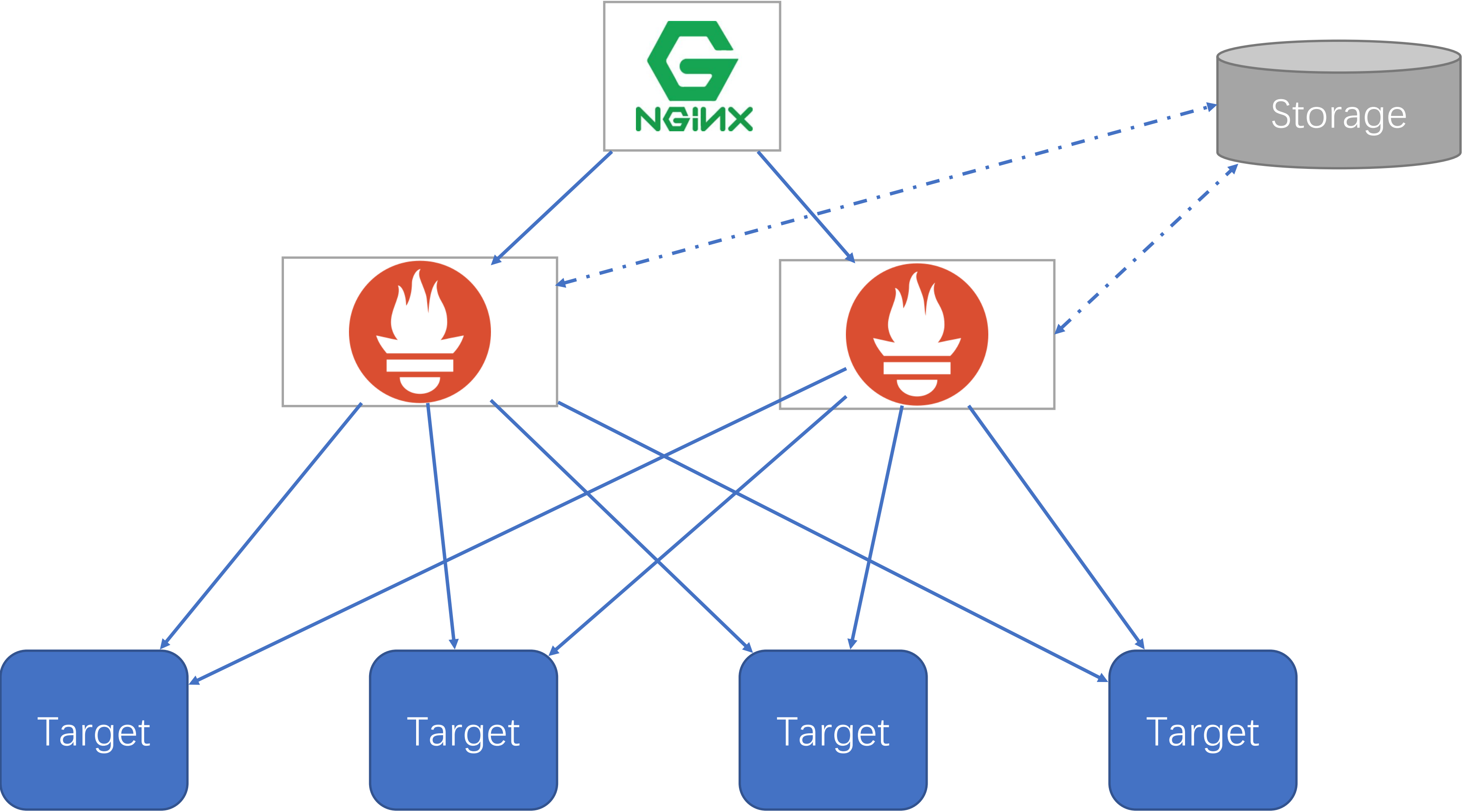


# 简单HA



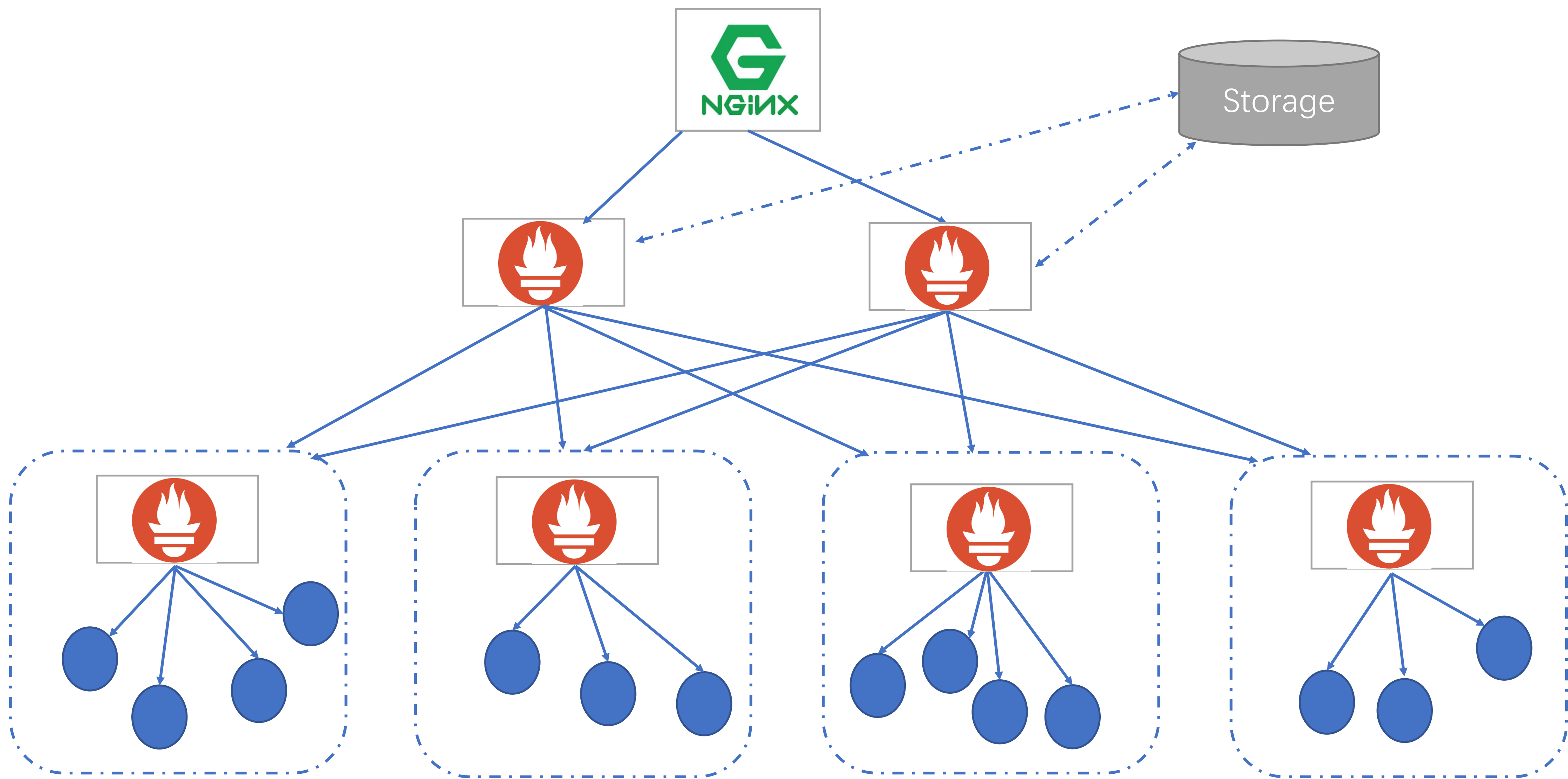


# 基本HA+远程存储





# 基本HA+远程存储+联邦集群



# 第14部分

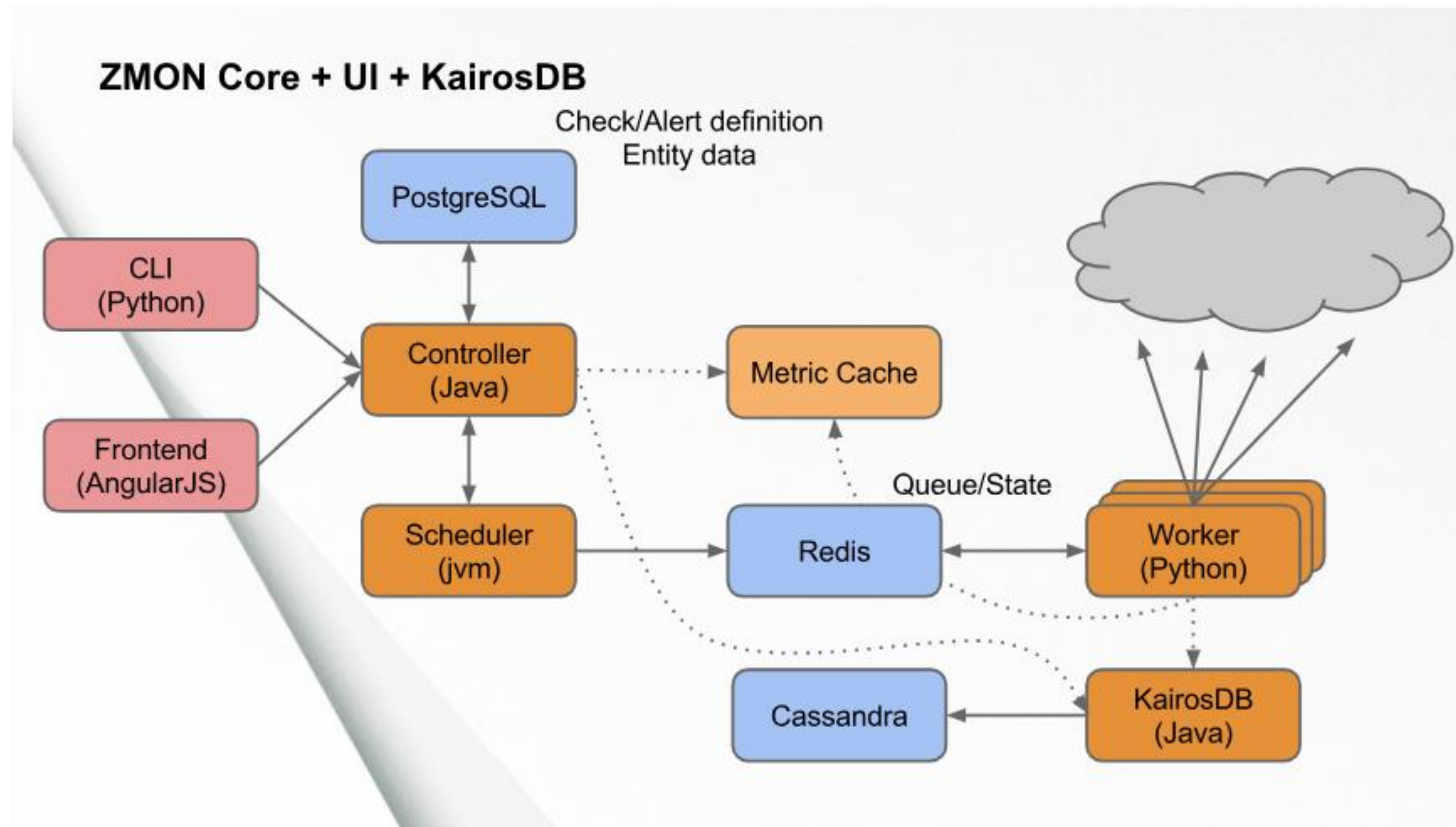
## 主流开源时序数据库比较

	OpenTSDB	KariosDB	Promethes	InfluxDB
开源时间	2010	2013	2012	2013
类型	时序数据库	时序数据库	时序数据库	分析数据库
分布式	支持	支持	Federation	商业
存储	Hbase	Cassandra	定制	定制
采集模式	Push	Push	Pull	Push
Grafana集成	支持	支持	支持	支持
计算函数	良好	一般	丰富	丰富
告警模块	无	无（ZMon）	有	有
查询语言	HTTP API	HTTP API	PromQA	InfluxQL
Web UI	支持	支持	支持	支持
维度支持	Tag	Tag	Label	Tag+Field
预聚合	Roll-up	Roll-up	Recording Rule	Continous Query
实现语言	Java	Java	Golang	Golang
github活跃度	~3.2k	~1.3k	~19.6k	~14.6k
商业支持	无	无	无	有
适用	中大规模	中大规模	中小规模	中小规模

# 第15部分

## 开源分布式监控平台ZMon简介

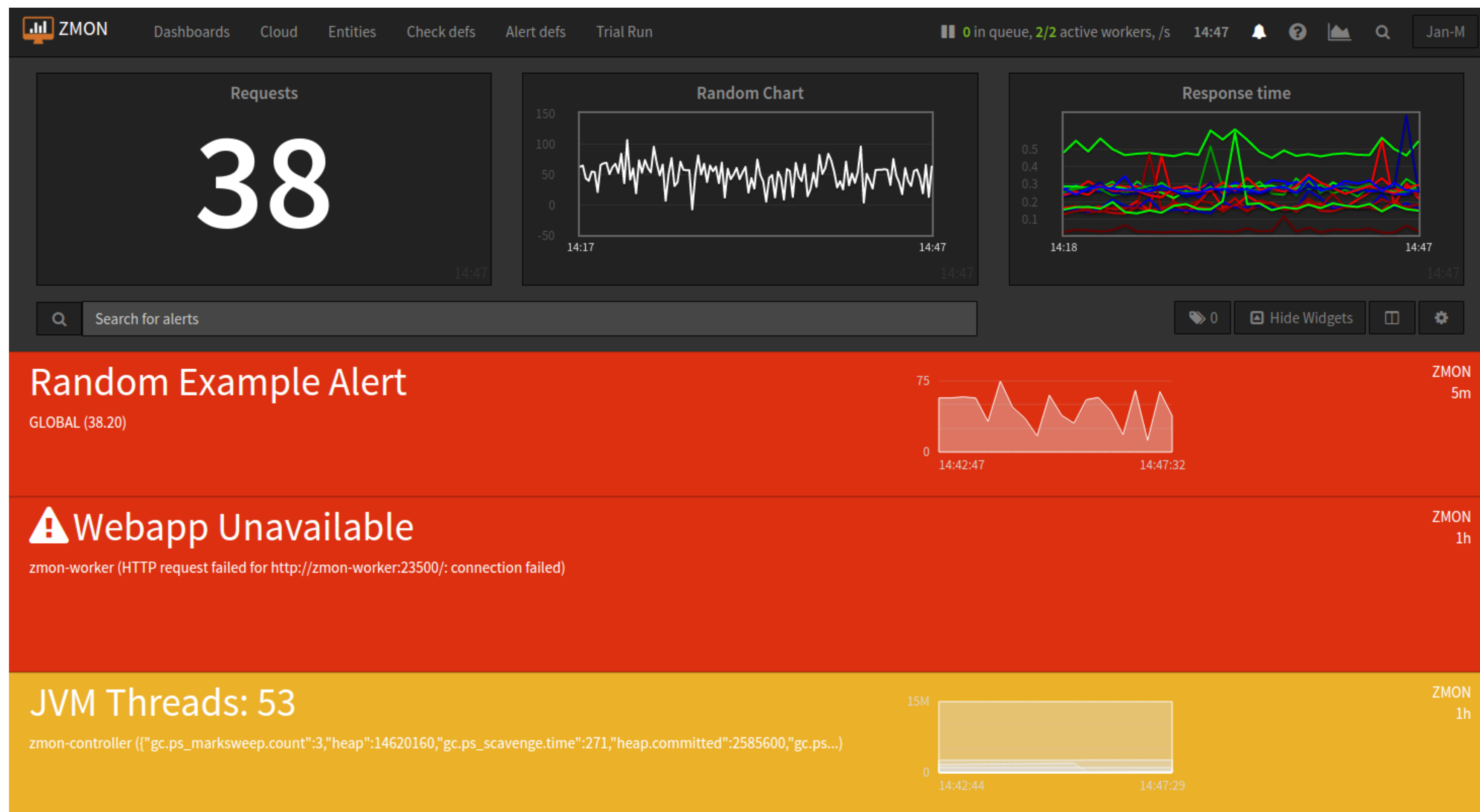
# ZMon架构



- Zalando开源
- 分布式监控告警系统
- 拉模式(Check)
- Python定义 Check/Alert
- DevOps团队自治监控
- 基于KairosDB

<https://opensource.zalando.com/zmon/>

# ZMon告警大盘





# Python表达式

Alert: Webapp Unavailable

ID: 5, Status: , Team: ZMON

Edit

Clone

Inherit

Delete

History

Trial Run

Evaluate

Cleanup

Comments (0)

Description

One of the ZMON Demo web applications is not available

Condition

value not in [200, 400, 404]

Responsible Team

ZMON

Details

Check Name, ID, History

Webapp HTTP Status, ID: 7, History

Check Team

ZMON

Check Command

# see https://zmon.readthedocs.org/en/latest/user/check-commands.html#http  
http('/', timeout=5).code()

Check Interval

1m

Check Entities

type = demowebapp

Parameters

(no parameters)

Entities Filter

(no entities)

Excluded Entities Filter

(no excluded entities)

Notifications

(no notifications defined)

Time Period

(no time periods defined)

Tags

- Alerts/Checks
- Downtimes (0)
- History
- Children (0)

Alerts (1)

In downtime (0)

OK (4)

Filter entities

<div></div> <div></div>	Name	Last run	Active since	Value	Captures
<div></div> <div></div>	zmon-scheduler	18.11.16 14:49 (2s ago)		404	{}
<div></div> <div></div>	zmon-kairosdb	18.11.16 14:49 (2s ago)		200	{}
<div></div> <div></div>	zmon-eventlog-service	18.11.16 14:49 (2s ago)		400	{}
<div></div> <div></div>	zmon-controller	18.11.16 14:49 (2s ago)		200	{}
<div></div> <div></div>	zmon-worker	18.11.16 14:49 (2s ago)	18.11.16 12:51 (1h ago)	HTTP request failed for http://zmon-worker:23500/: connection fa...	{}



# Alert状态和历史

type:demowebapp

ID	Type	Alerts		
zmon-controller	demowebapp	OK	5d	OK
zmon-eventlog-service	demowebapp	OK		
zmon-kairosdb	demowebapp	OK		
zmon-scheduler	demowebapp	10h	OK	OK
zmon-worker	demowebapp	5d		

# Alert调试

## Trial Run

Trial Run allows the end user to test check commands and alert conditions before using them on the system.

Details

Stop

Alerts

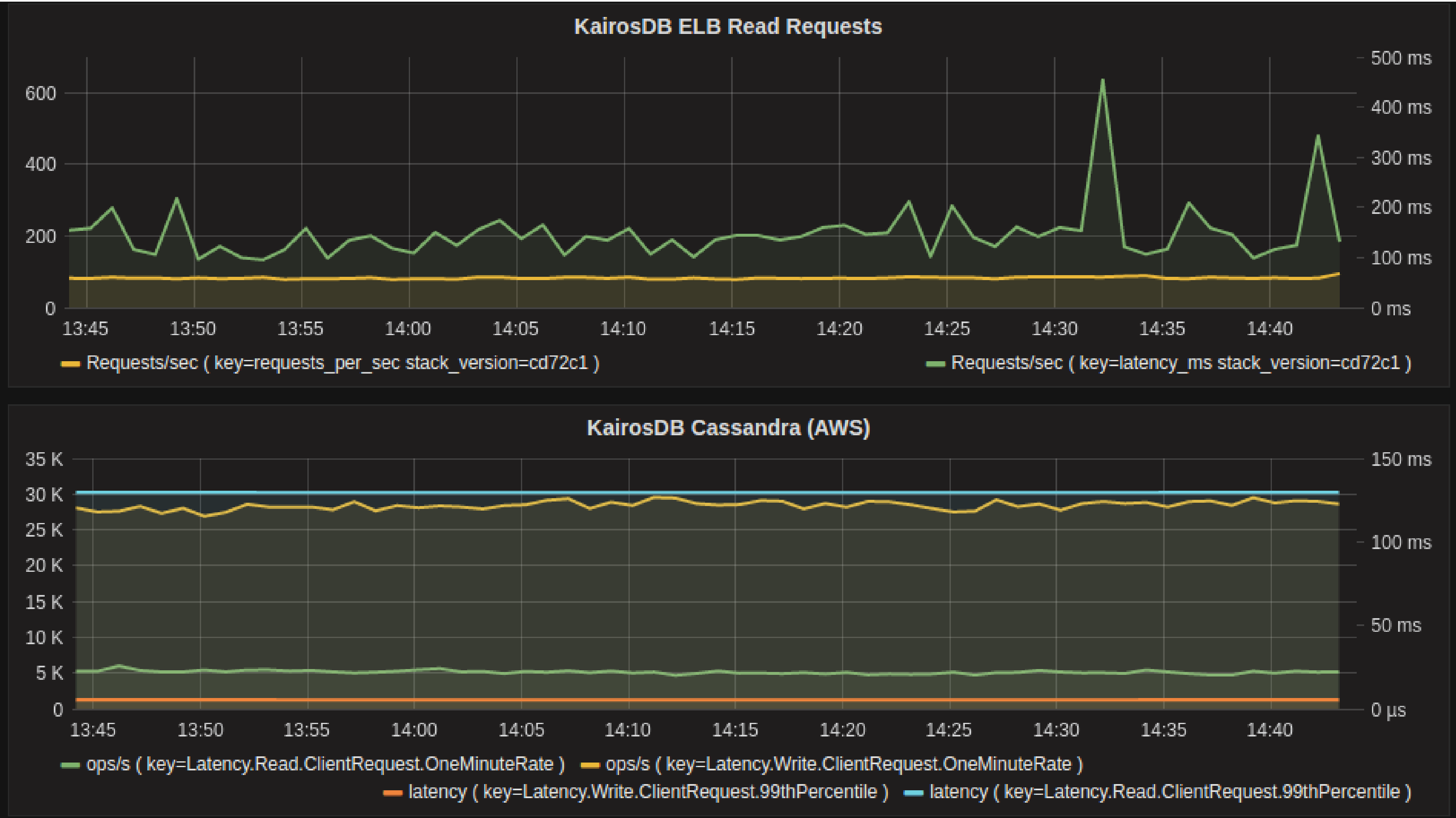
Alerts (12)

OK (51)

Filter entities

Name	Timestamp	Last run	Value	Captures
asa[team-service]	27.07.15 15:57 (1m ago)	233ms	{"eu-central-1":["Errno 111] Connection refused","eu-west-1":["Errno -2] Name or se...	{"problems": ["eu-central-1","eu-west-1"]}
bi[team-service]	27.07.15 15:57 (1m ago)	186ms	{"eu-central-1":["Errno -2] Name or service not known","eu-west-1":"OK"}	{"problems": ["eu-central-1"]}
cd[team-service]	27.07.15 15:57 (1m ago)	21ms	{"eu-central-1":"OK","eu-west-1":["Errno -2] Name or service not known"}	{"problems": ["eu-west-1"]}
core[team-service]	27.07.15 15:57 (1m ago)	749ms	{"eu-central-1":["Errno -2] Name or service not known","eu-west-1":["Errno -2] Nam...	{"problems": ["eu-central-1","eu-west-1"]}

# Grafana集成



# 参考

- **源码**

- <https://github.com/zalando-zmon>

- **Check commands reference**

- <https://docs.zmon.io/en/latest/user/check-commands.html>

- **Alert Functions reference**

- [https://docs.zmon.io/en/latest/user/alert-ref/alert reference functions.html](https://docs.zmon.io/en/latest/user/alert-ref/alert%20reference%20functions.html)

- **Notifications reference**

- <https://docs.zmon.io/en/latest/user/notifications.html>

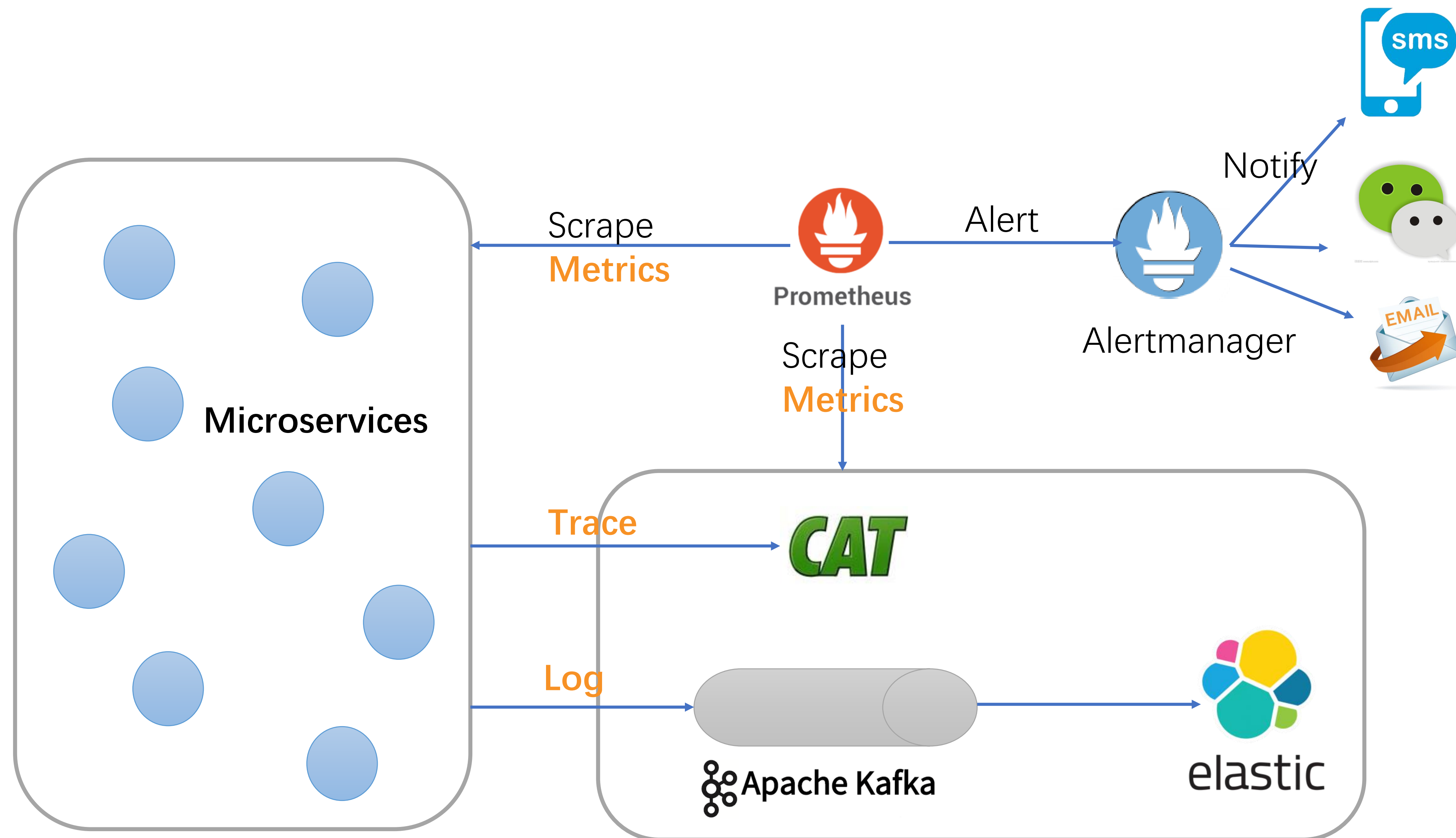
# 第16部分

## 微服务监控体系总结

# 四层轻量监控体系



# 参考架构



# 第17部分

## 参考资源和后续课程预览



# 参考文档

- **Metrics Driven Development**

- <https://www.infoq.com/articles/metrics-driven-development>

- **Go for Industrial Programming**

- <https://peter.bourgon.org/go-for-industrial-programming>

- **Prometheus官网**

- <https://prometheus.io>

- **Prometheus入门与实践**

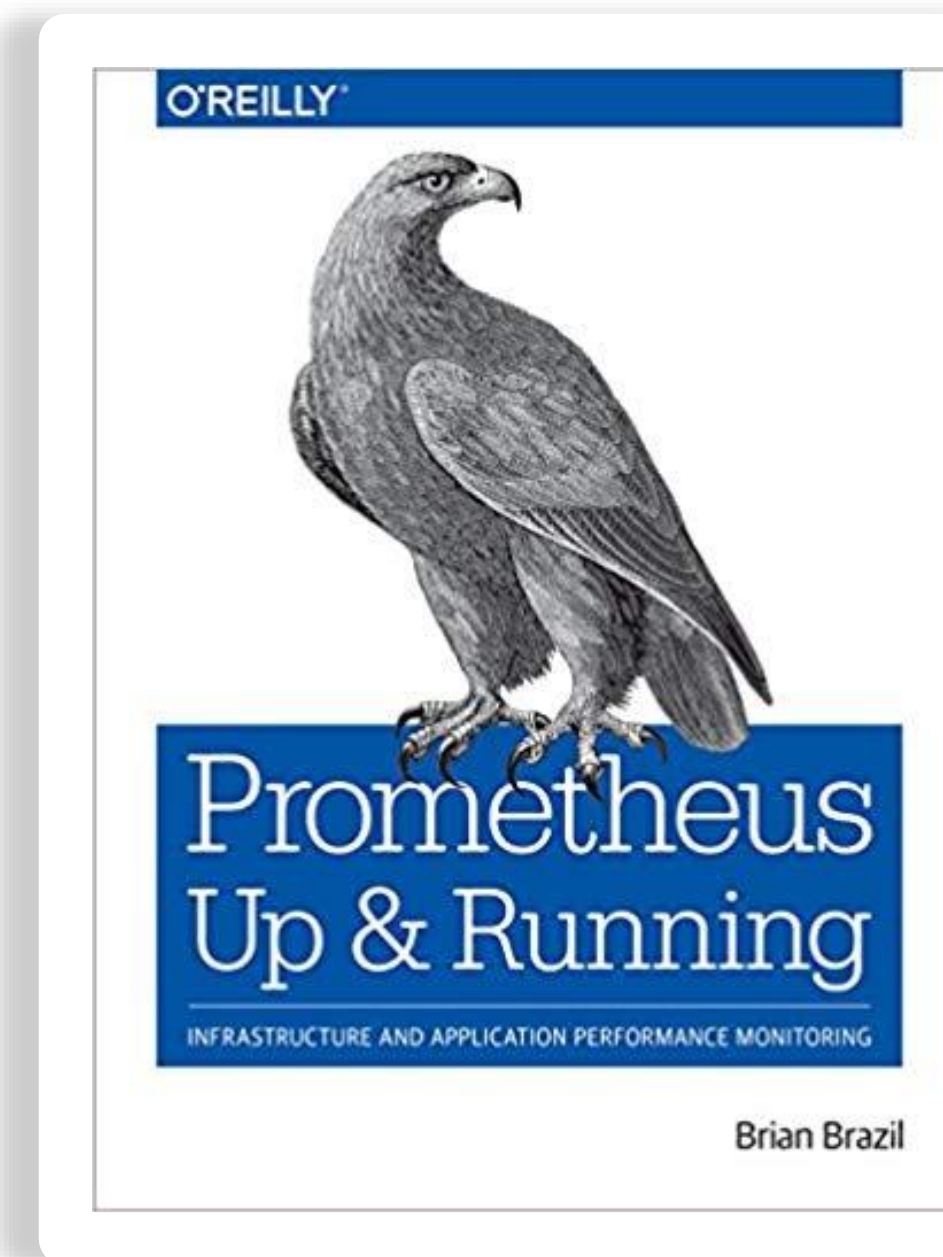
- <https://www.ibm.com/developerworks/cn/cloud/library/cl-lo-prometheus-getting-started-and-practice/index.html>

- **全面学习Prometheus (郑云龙)**

- <http://dockone.io/article/5716>

# 参考书籍

- **Prometheus Book(ebook) (郑云龙)**
  - <https://github.com/yunlzheng/prometheus-book>
- **Prometheus Up & Running(英文版)**



# 扩展方案

- 高可用和长期存储方案



- <https://github.com/improbable-eng/thanos>

- 长期存储方案

- <https://github.com/m3db/m3>



- 配置管理

- <https://github.com/line/promgen>

LINE

# 其它开源时序数据库产品

- **OpenTSDB**

- <http://opentsdb.net/>



- **KairosDB**

- <https://kairosdb.github.io/>



- **InfluxDB**

- <https://www.influxdata.com/>

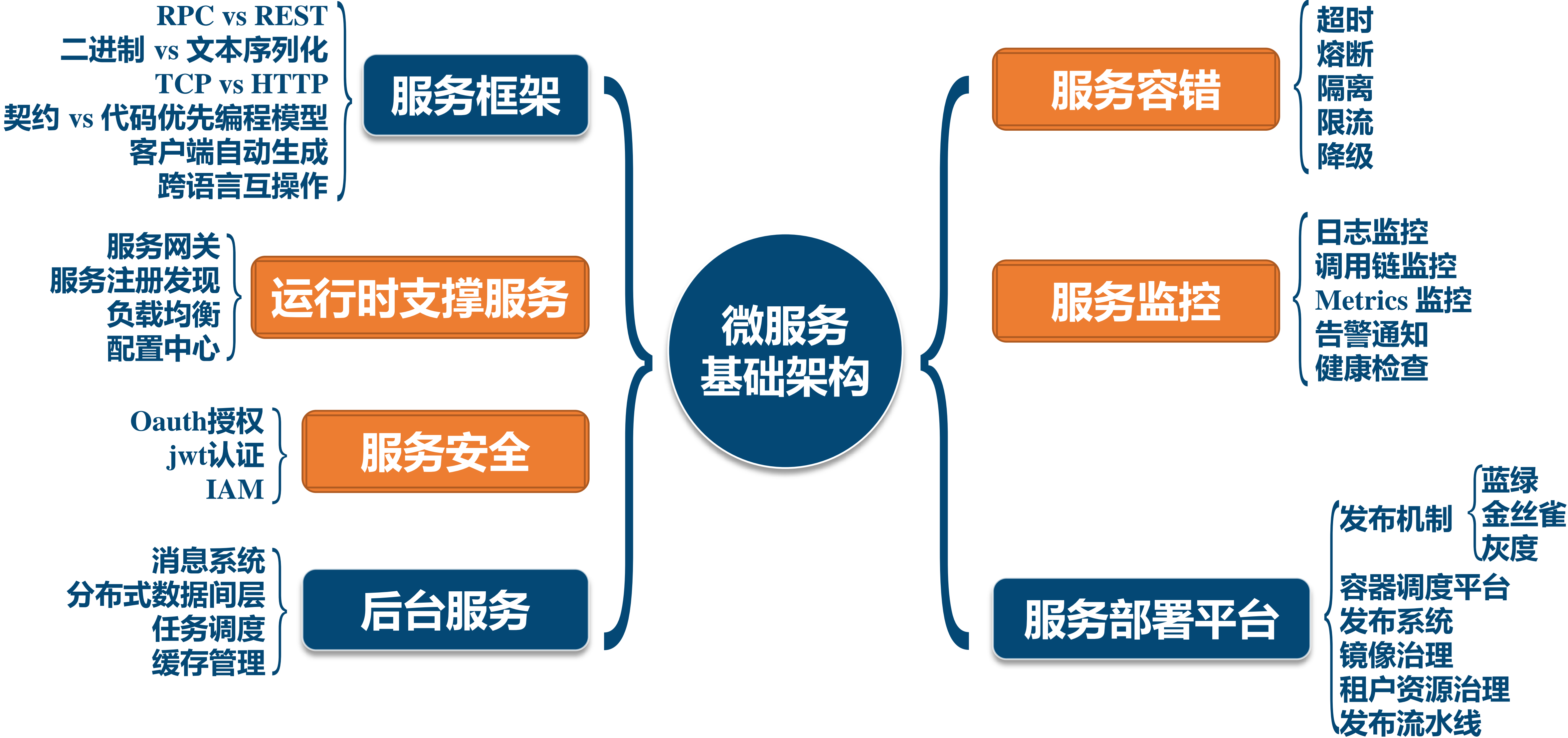


- **Graphite**

- <https://graphiteapp.org/>

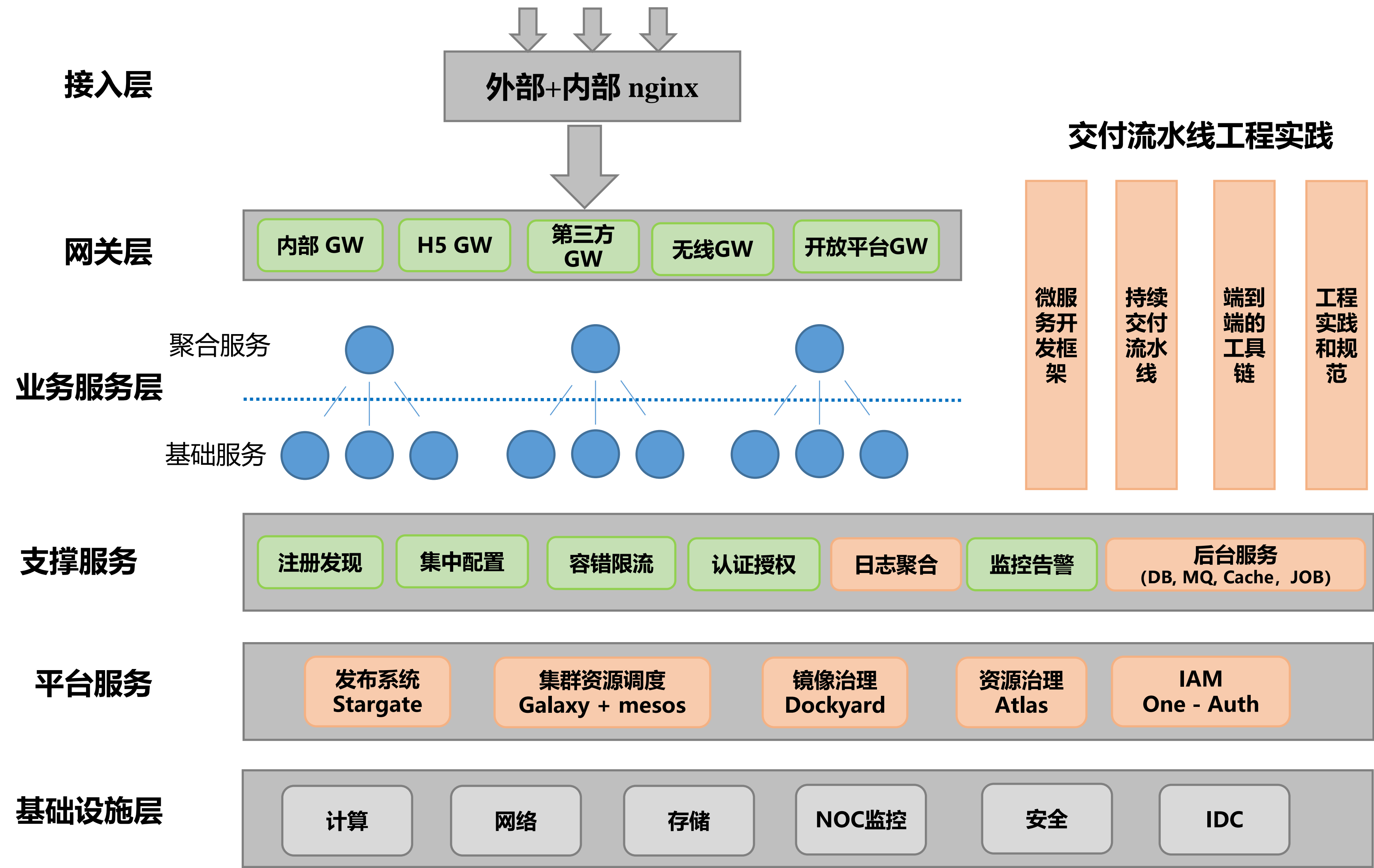


# 后续课程预览~2018课程模块



# 后续课程预览~技术体系

微服务架构总体技术体系





# 架构和技术栈预览

