# Metrics in ServiceComb Java Chassis 1.0.0-m1

让微服务运行状态清晰可见

郑扬勇

微服务架构师

开源能力中心

# 目录

- Metrics简介
- 基于BMI示例演示
- Metrics技术细节

# Metrics是什么

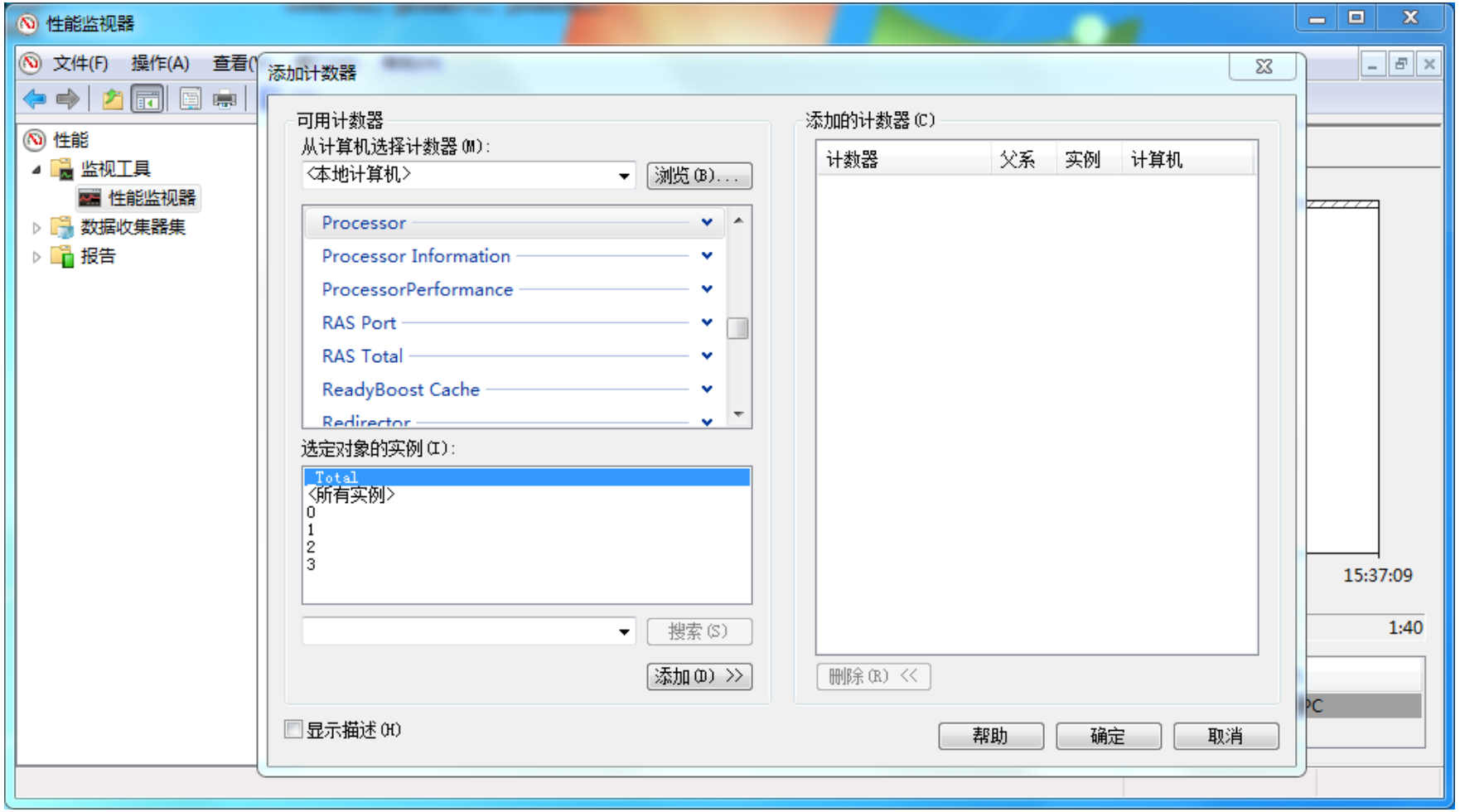
- 直译是“度量”，不同的领域定义有所区别，在微服务领域中的定义：

  ***“对微服务的某个指标给予一个可量化程度的测量”***

- Metrics应该具备的特性：
  - Comparative（可对比）：指标能够在不同的微服务或同一个微服务的多个实例之间比较；
  - Understandable（易理解）：指标所衡量的对象、计算方法和输出的结果值都是容易理解的；
  - Ratio（理想的比例）：理想结果可预见，可以立即用于比较。

# 最常见的Metrics

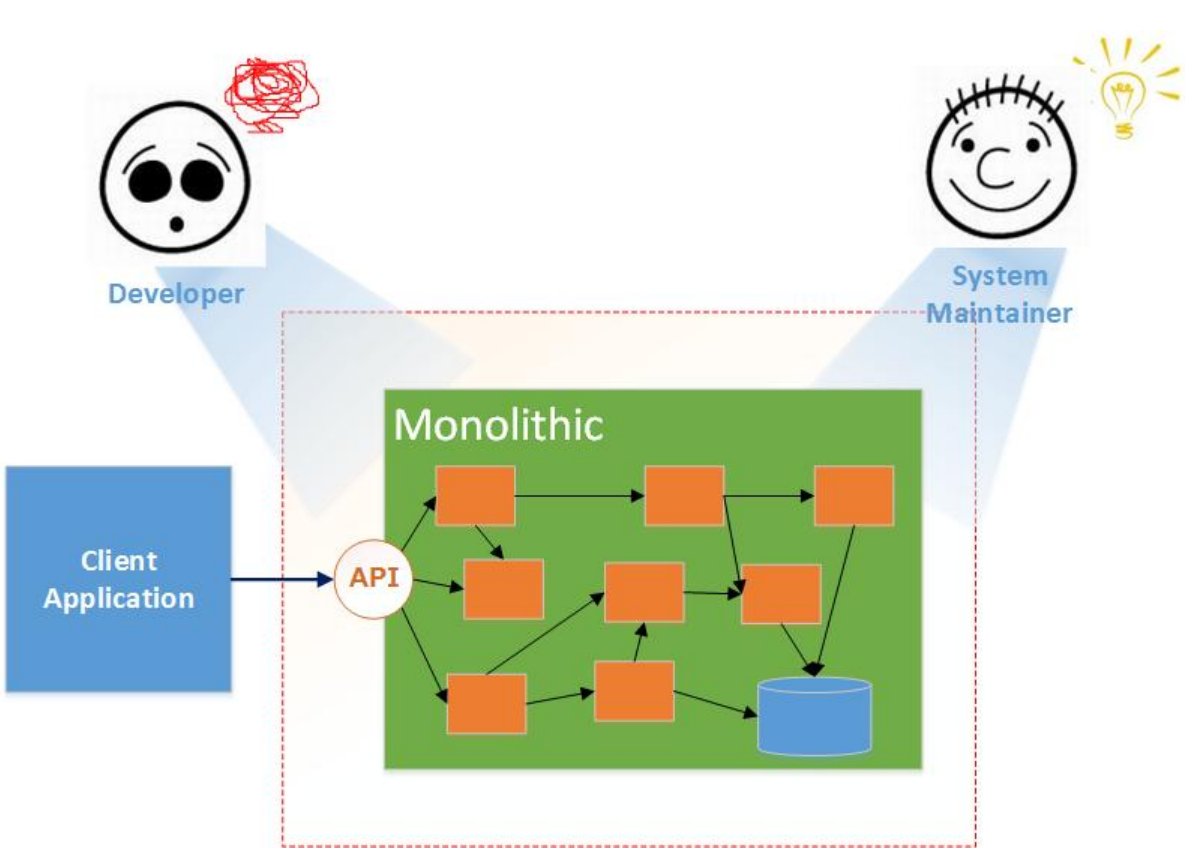
- 衡量Metrics实现优劣的标准：
  - 关键指标覆盖全
  - 计量准确
  - 高性能低资源占用
  - 无侵入或低侵入

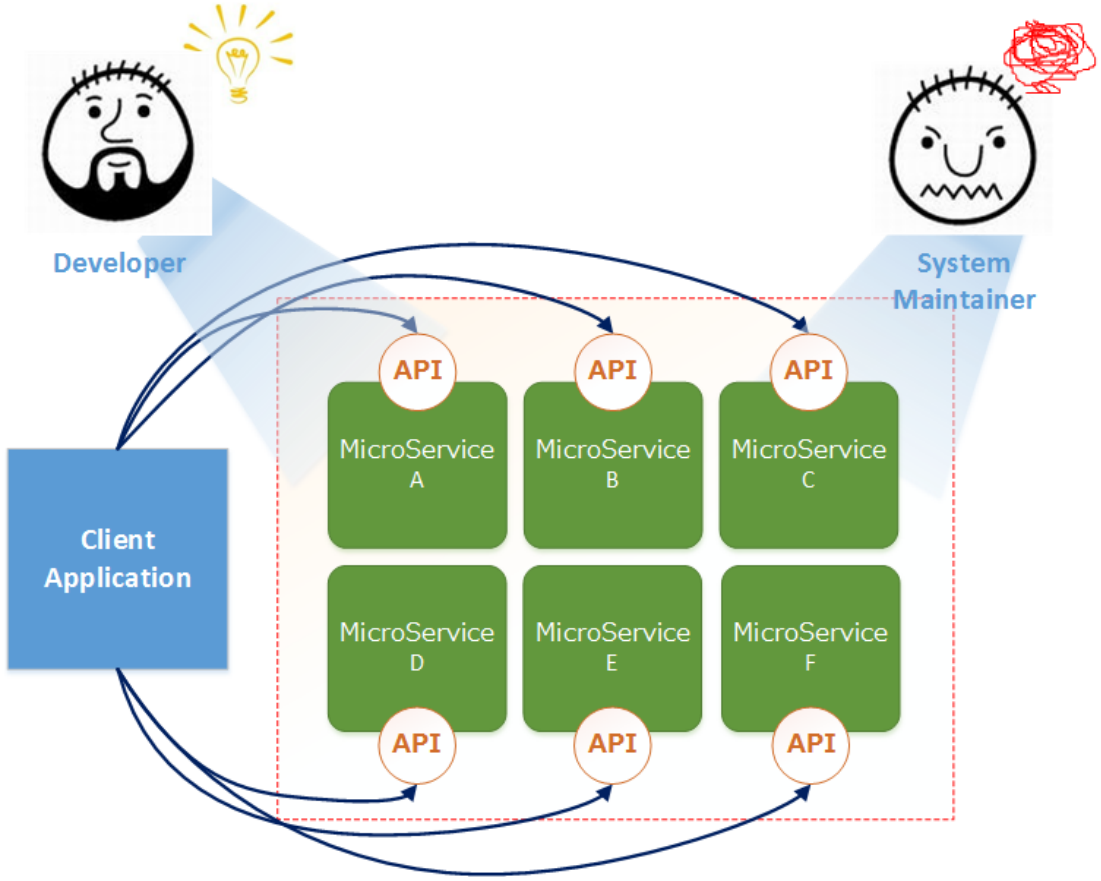


# Metrics的分类

- Metrics有很多种分类方式，在技术实现上我们偏向以取值方式区分为两种：
  - 1. 直接取值
    - 任何时候都能够立刻获取到最新值，例如资源使用率，包括CPU使用率，线程数，Heap使用数据等等，还有调用累加次数，当前队列长度等等。
  - 2. 统计取值
    - 经过一个特定的时间周期才能够统计出值，这个时间间隔我们可以称为窗口周期（Window Time）或统计周期，例如：
      - a) 多值取其一的，比如Max、Min、Median（中位值）；
      - b) 与时间相关的，比如TPS（transaction per second）；
      - c) 与个数相关的，比如累加平均值、方差等等；
    - **获取此类Metrics的值，返回的是上一个周期的统计结果，具有一定的延后性。**

# 为什么需要Metrics



# 开源领域的Metrics实现

Netflix Servo ： https://github.com/Netflix/servo

Dropwizard Metrics： https://github.com/dropwizard/metrics

Spring Boot Actuator： https://github.com/spring-projects/spring-boot/tree/master/spring-boot-project/spring-boot-actuator

# 开源领域的Metrics比较

| 比较项 | Netflix Servo | Dropwizard Metrics | Spring Boot Actuator |
| --- | --- | --- | --- |
| **计数器（Monitor）的实现** | 性能相对高效 | 功能相对丰富 | 较弱，但是支持使用Dropwizard Metrics集成实现强化 |
| **数据发布** | Push模式，自带三种Observer | Pull或Push，自带几种Reporter | Pull或Push，自带两种Repository |
| **框架相关** | 无框架绑定 | 无框架绑定（不绑定Dropwizard） | 绑定Spring Boot |
| **侵入性** | 有，需要自己打点 | 有，需要自己打点 | 按需，Http自动记录基本信息，也可以自己打点 |
| **易集成** | 是 | 是 | 是（限Spring Boot中启用） |

- 通过上面的比较可以看出，三种Metrics对于微服务的支持都很弱，例如微服务包含若干个Operation，不写代码无法全面获取每一个Operation的各类指标（调用数、TPS、Latency等等）。

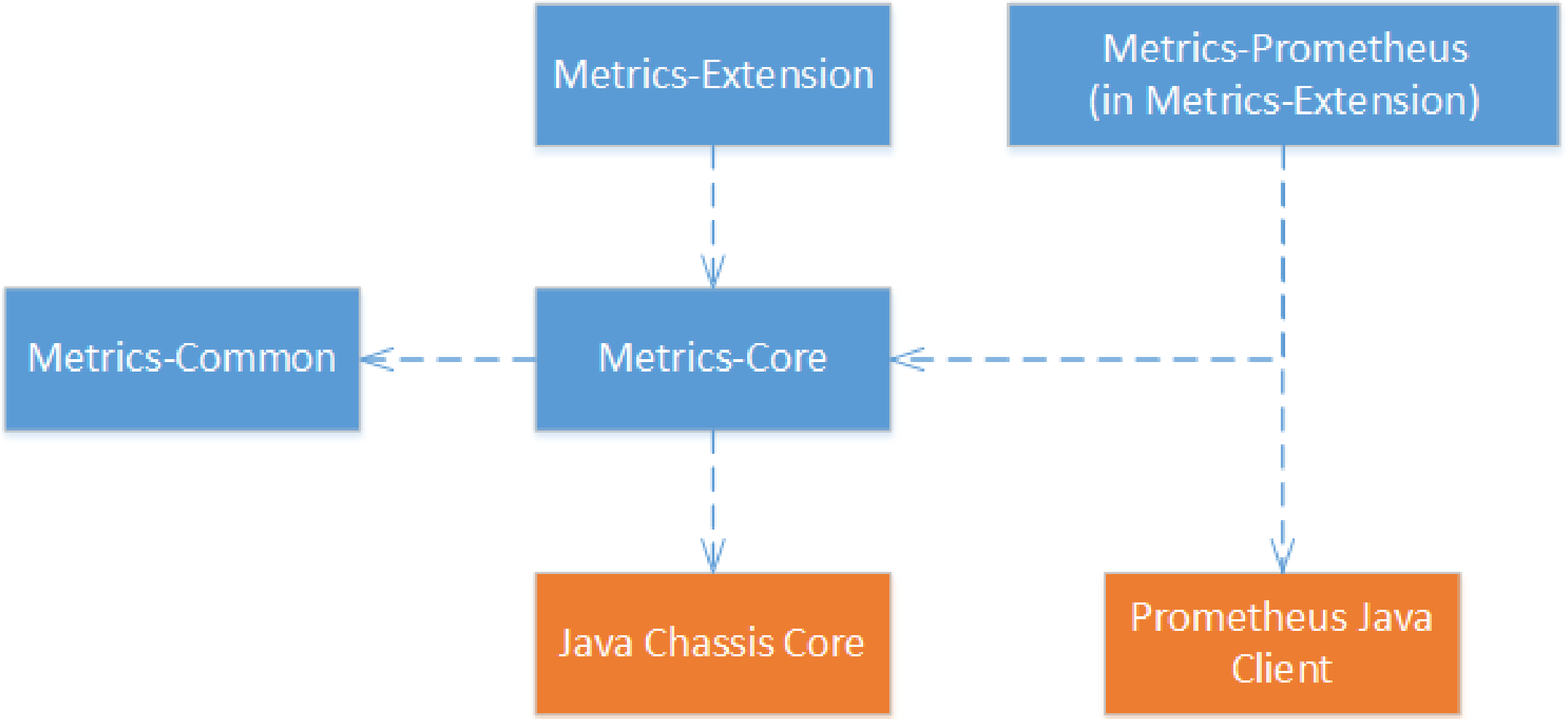
# ServiceComb Java Chassis中的Metrics

- ServiceComb Java Chassis是一个包含了服务注册，服务发现，服务配置以及管理功能的微服务框架，因此我们决定提供内置的更强大的Metrics功能：
  - **开箱即用，不写一行代码输出关键Metrics，全面覆盖调用数、TPS、Latency等；**
  - 基于Netflix Servo，使用固定统计周期（稍后会详细介绍）；
  - 多维度统计，帮助用户抽丝剥茧快速定位问题，支持的维度包括：
    - 微服务实例（Instance）级和操作（Operation）级；
    - 操作结果成功（Success）和失败（Failed）（开发中）；
    - Transport区分Rest和Highway（评估中）。

# 基于BMI示例演示（15min）

- 示例项目BMI介绍
- 如何使用ServiceComb Java Chassis Metrics
- 如何与普罗米修斯集成

# 依赖关系


Metrics-Extension
Metrics-Prometheus
(in Metrics-Extension)
Metrics-Common
Metrics-Core
Java Chassis Core
Prometheus Java
Client

# Metrics列表

| Metrics | 子项 | 说明 |
| --- | --- | --- |
| 微服务资源使用 | CPU、ThreadCount、Heap、NonHeap | 基本的资源使用状态 |
| Consumer端 | Latency、CallCount、TPS | 包含操作级别和微服务实例级别 |
| Producer端 | waitInQueue、lifeTimeInQueue、executionTime、Latency、CallCount、TPS | 包含操作级别和微服务实例级别 |

- 对于时延类的Metrics，都包含max、min、average三个指标

# 数据发布格式的选择

- 不同的实现发布格式都有所不同

| Spring Boot Actuator | Prometheus HTTP Server |
| --- | --- |
| { "counter.status.200.root": 20,<br>"counter.status.200.metrics": 3,<br>"counter.status.200.star-star": 5,<br>"counter.status.401.root": 4,<br>"gauge.response.star-star":<br>6, …"datasource.primary.active": 5,<br>"datasource.primary.usage": 0.25 } | # HELP calculator.metricsEndpoint.metrics<br>Producer Side<br>servicecomb_calculator_metricsEndpoint_<br>metrics_producer_lifeTimeInQueue_averag<br>e 0.0<br>…<br>servicecomb_calculator_metricsEndpoint_<br>metrics_producer_executionTime_total 0.0 |

- 经过权衡，我们最后决定发挥ServiceComb契约的优势，直接返回RegistryMetric实体对象

# 使用固定统计周期

## 劣势：

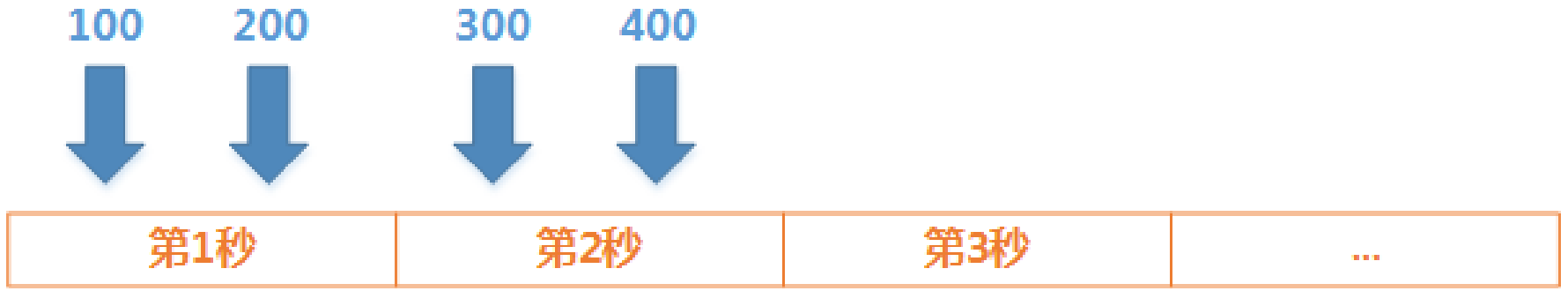
- 由于是预设，很有可能与查询（抓取）间隔不同。例如你设置了统计间隔为15秒，但是使用Prometheus作为监控系统设置抓去间隔为5秒，那么Prometheus必定会在三次采样中获取一次重复数据；
- 设定存在取舍，不同的场景对周期的需求不同；但可以通过设置多周期弥补。（例如统计报告中的日报、周报、月报、季报、年报）

## 优势：

- 具备高性能的同时又能保持极低的开销。

# 使用多周期适应不同的场景需求

假定有三个Monitor，为最大值max、最小值min和合计sum；定义了两个统计周期分别为1秒和2秒；
触发计数如下图所示：

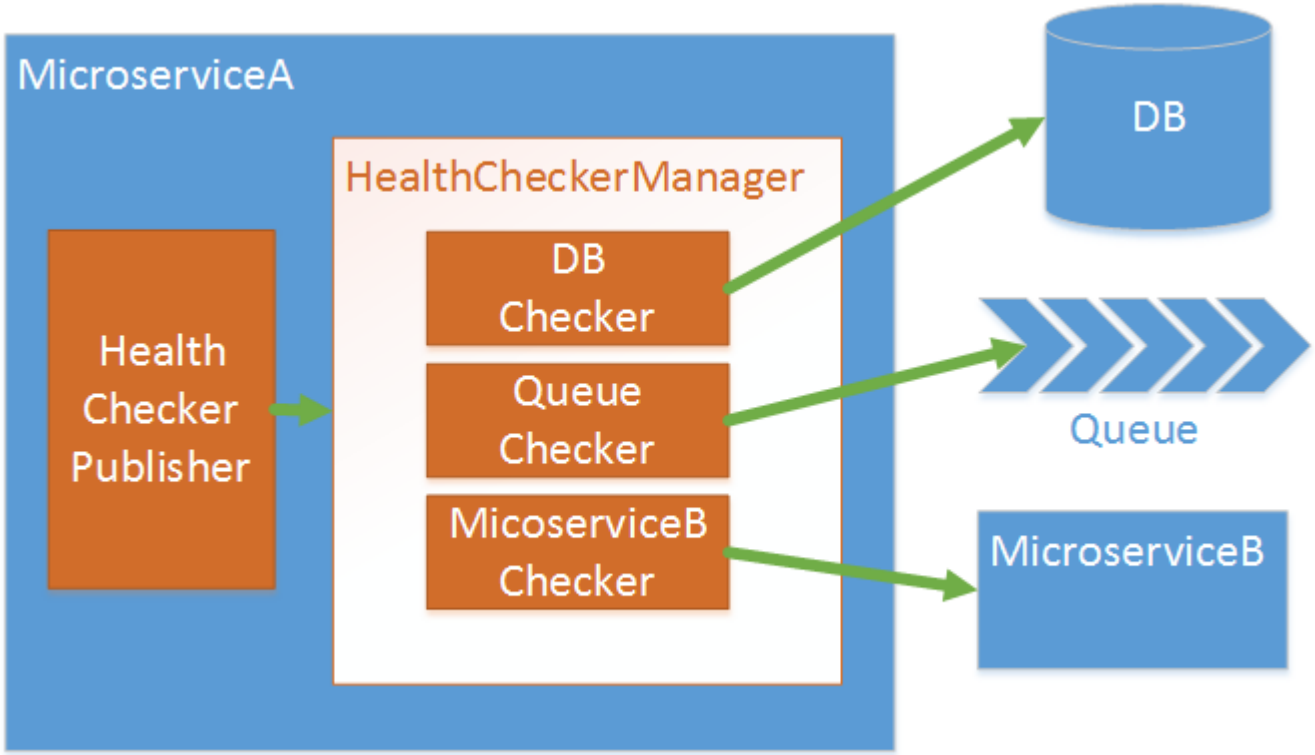



获取1秒周期的值返回为min=100 max=200 sum=300
获取2秒周期的值返回为min=0 max=0 sum=0

获取1秒周期的值返回为min=300 max=400 sum=700
获取2秒周期的值返回为min=100 max=400 sum=1000

# 支持Health Check

- 通过Health Check让微服务支持检查依赖组件的状态并返回，可以用于制定策略，也可以用于Dashboard展现
  - 微服务很可能依赖数据库、其它微服务或中间件，这些组件状态正常是微服务能够正常提供服务的前提；
  - 相比Metrics返回一个状态值，Health Check的返回更丰富，可以附带额外信息，例如详细的错误Trace。



# Q & A

我们已经进入Apache孵化，欢迎大家参与贡献，项目地址：
https://github.com/apache/incubator-servicecomb-java-chassis

我们的官方微服务论坛，欢迎大家来看看：
http://forum.huaweicloud.com/forum-622-1.html
现在正在搞嘉年华的活动哦，结束后快去抢礼包！
http://forum.huaweicloud.com/thread-5570-1-1.html

官方微信号

公众号

# 谢谢

官方微信号

公众号