约定：

1.字体

中文：微软雅黑

应为：Arial Black

## 第一章 相遇 kafka

每个企业都是由数据支撑着。我们获取、分析、操作和创造更多的消息作为输出。无论是日志消息，指标、用户行为、发文，还是其它的应用程序，它们都创造数据。数据的每一个字节都代表特定的含义，一些重要的事将告诉我们接下来需要做的事情。为了了解这些数据的含义，我们需要知道数据是从哪里创建出来的，会在哪里被分析处理。我们在类似于亚马逊这样的网站上浏览，点击自己感兴趣的条目，这些信息将在不久后作为推荐信息展示给我们。

我们越快的做到这些，我们的组织就越灵活，响应能力越强。我们在改变数据上花费努力越少，我们越能够集中精力关注于核心业务。这就是在以数据驱动的企业中传播途径作为关键组件的原因。我们如何将数据变得和数据本身一样重要。

任何时候，科学家都不会赞同这种说法，这是因为我们没有足够多的数据。然而，我们可以赞同，关于我们获取什么样的数据，拿到数据解决了问题就可以了。获取我们对的，或者你是对的，再者我们都错了。我们继续向前！

-- Neil deGrasse Tyson

## 发布和订阅消息

在讨论Apache Kafka 规范之前,理解 publish/subscribe messaging 的概念是重要的，它为什么重要。Publish/Subscribe messaging 是一种publisher具备发送一段数据的特性，而不是明确地直接发送给receiver。然而，publisher 以某种方式将消息分类，receiver 订阅确切的类别来接受消息。Pub/Sub 系统常常带有一个broker , 一个中心节点，消息被发送到中心节点上。

## 如何入门

Pub/Sub 的许多应用场景都是以一个简单的队列或者进程间通信通道开始的。例如，你创建一个应用程序，这个应用需要在各个地方发送监控消息。并且有一个app直接连接到该应用，检测数据通过连接传输，在app的控制面板上展示检测数据。如图1-1所示:

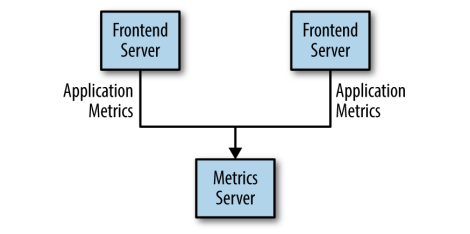


图1.1

这是一个开始检测的简单问题的解决方案，不久之后，你要花费很长时间分析测量值，并且在控制面板上也不能很好的工作。你需要修改应用程序将测量数据写到两个系统，需要开启一个新的服务接受测量数据，存储数据和分析数据。但是现在有三个或更多的应用程序在生成测量数据，它们以相同的方式连接着两个服务，你的同事认为开启轮询服务是一个好主意，因此你在每个应用上加入了一个服务为了请求而提供测量数据。过了一段时间，更多的应用通过这些服务获取各自的测量数据并利用测量数据达到各自的目的。结构如图1-2所示，在图中甚至很难去追踪连接。

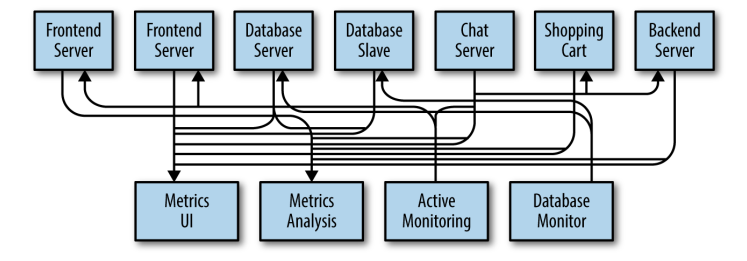


图1-2 多维发布者直接连接

到此为止技术债的累加很明显了，因此你决定付出并还清债务。你在外围建立了一个单独的应用来接受所有应用产生的metrics 数据，并为任何需要metrics数据的系统提供查询服务。这样可以减少架构的复杂性，如图1-3所示。恭喜你，你已经建立了一个 publish-subscribe messaging 系统。

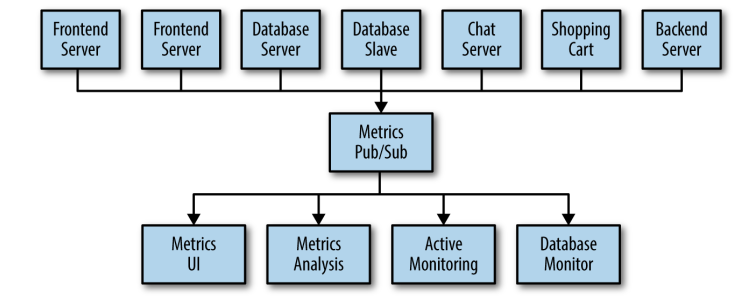


图1-3 一维 publish/subscribe 系统

## 单队列系统

以此同时，你已经利用metrics 发起了一场战争，你的同事在处理一些任务时也做了相似的工作，一个是日志消息，另外一个在网站前端追踪用户行为并提供谁在设备上学习等信息给开发者，为管理人员创建一些报表。你们都遵循相似的方法来构建外围系统去解耦从订阅者到发布者的信息。图1-4展示了一个包含三个Pub/Sub 结构的系统。

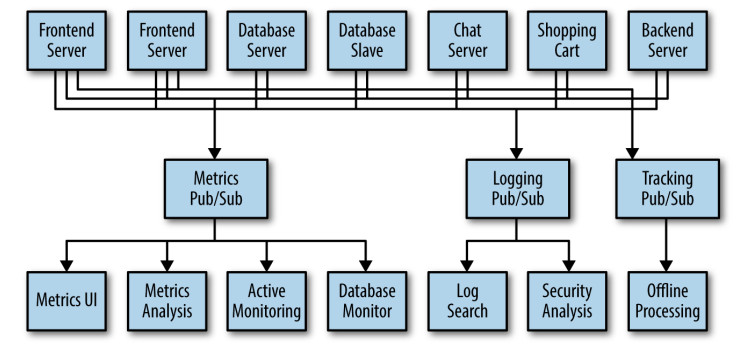


图1-4 多Publish/Subscribe 系统