# 2018年上海QCon会议总结

会议时间：2018年10月18-20日

会议地点：上海

这次全球软件开发大会的议题主要由国内互联网公司分享其内部技术创新和实践，也有少数国外技术专家分享议题（主要是开发语言相关的进展，包括Java API、GO语言最新特性等）。在开发大会的进行过程中，出现的热门技术词汇:DevOps/AIOps、微服务（K8s、ServiceMesh、Serverless）、人工智能、区块链等，这些技术是此次大会的核心内容，从这些议题中可以看出微服务是此次QCon会议的重点，DevOps/AIOps也是围绕微服务进行。当然这次开发大会中，也有部分互联网企业带来的关于大数据技术的使用议题，例如快手公司分享的如何使用Hadoop构建对象存储平台。

这次软件开发大会，本人主要关注的议题是大数据技术在统一监控平台及构建底层服务架构两个方面；之外也参加了新技术相关的议题，例如阿里的Dubbo Mesh、AWS的Serverless等；作为软件开发人员，也参加了最新开发技术的一些议题，包括Java API Design最佳实践、响应式编程Eclipse Vert.x。下面进行详细介绍。

# 运维管控平台（DevOps/AIOps）

根据对议题的分析，微服务架构逐步成为目前互联网公司开发业务的主流技术手段，微服务是将应用分解为微小而互相连接的服务，其优点在于应用程序被拆分成了可管理的模块或服务，每个服务都独立扩展，便于开发者进行快速开发与部署；服务之间通过规范的API进行交互。但是确定是微服务的分布式特点，每个应用可能会拆分成几十或者数百的单服务，这给产品的运维带来的很大的挑战，因此需要统一的运维管理平台来对这些服务进行管理、运维及优化运行性能。

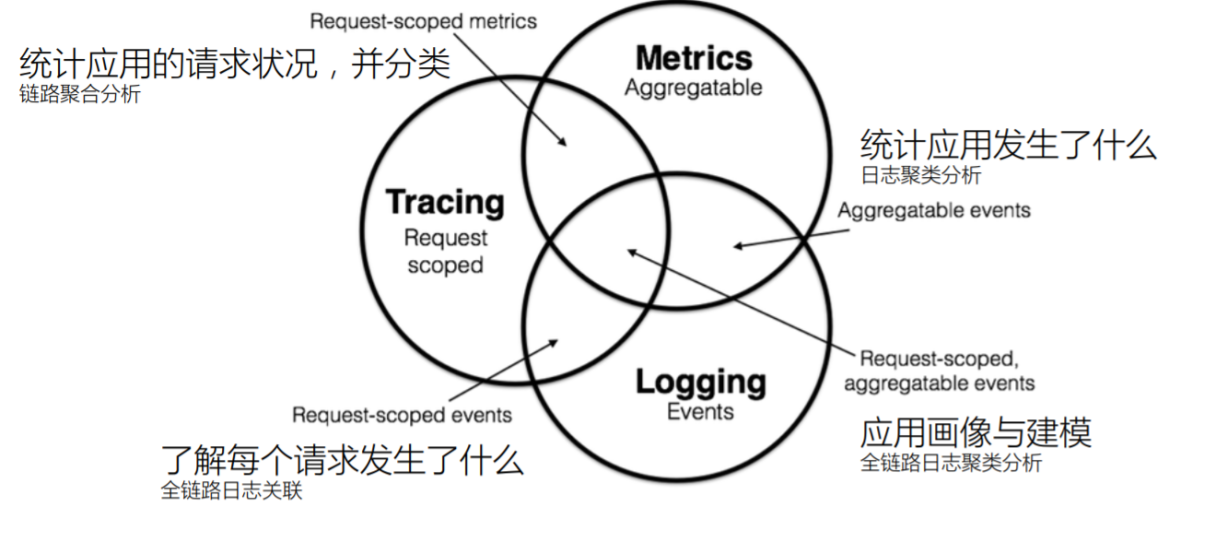
针对这方面的议题很多，阿里的下一代大数据实时监控系统、百度的平台动态降级及故障自愈、美团点评的监控平台CAT等，还有苏宁、随手记、喜马拉雅等公司的分享。下面是该方面内容的总结

## 监控指标

构建一个监控平台，首先要对监控哪些指标及作用有一个清晰的认识，在“随手记统一监控平台”议题中对这些指标有一个清晰的划分：

1. 日志，提供事件的细节，在统一监控平台中要收集各服务运行的日志，对其中有价值的信息进行汇总分析；可以通过日志发现服务的故障，辅助运维人员进行故障的准确定位及服务恢复
2. Metric/Time-series，实时监控服务的运行性能，这部分指标包括主机的硬件资源（磁盘、CPU、内存及网络等）；服务本身的运行指标，JVM使用、队列大小、RPC调用延迟等，通过Metrics可以知悉当前服务的运行状况，及早发现问题进行服务的调整，例如服务的负载均衡、服务的扩容等
3. Tracing（调用链），在微服务系统架构中，服务之间通过API调用来实现业务功能，服务站之间的调用逻辑复杂，很难通过单一服务的分析对整个业务的问题进行定位，因此需要调用链的分析实现服务的自愈、服务的断路等，防止单一服务造成整个业务的瘫痪

这三种监控指标涵盖了目前监控平台要处理的信息，各指标针对的目标也是不同的，但是将这几种指标进行关联分析，对运维及开发人员有更大的价值，如下图：



例如通过Tracing和Logging分析，可以达到全链路日志关联分析，了解到每个请求发生了什么；通过Logging和Metrics关联分析，日志聚类分析，可以统计应用发生了什么；将三者进行关联分析，可以对应用进行画像与建模。

在听其他监控平台的分析时，其他监控平台也都覆盖了这三种指标，例如苏宁立体式监控体系，其系统架构对这三者进行了明确的定位：



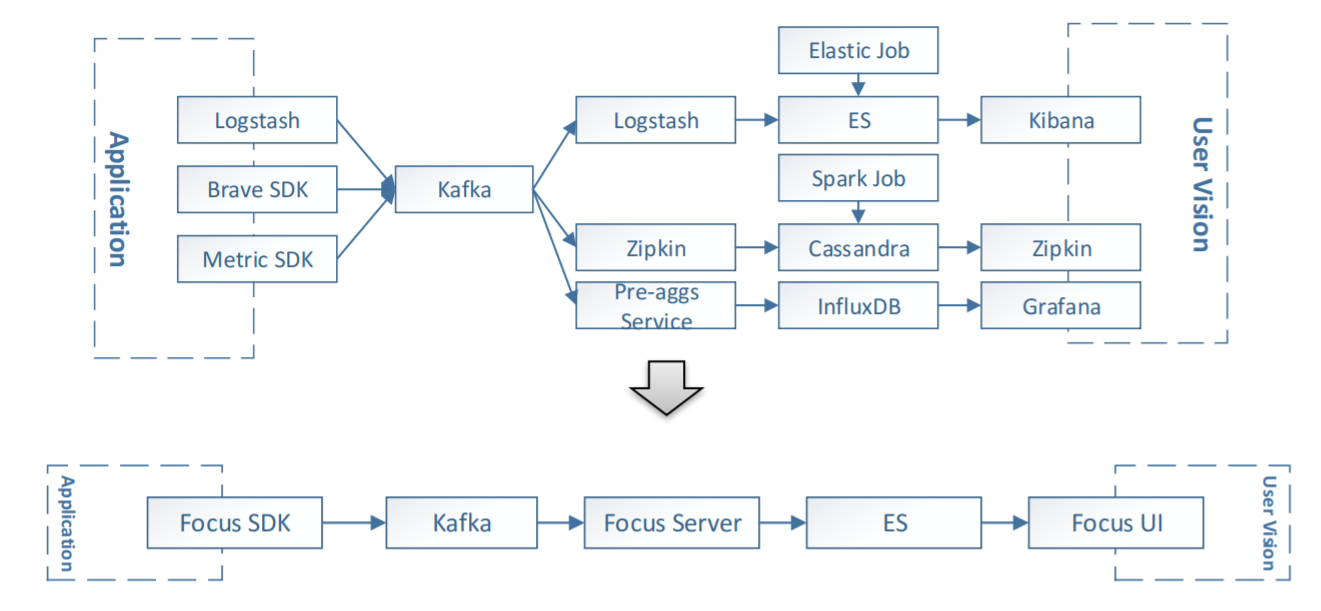
其监控体系包括：日志监控、端测监控、基础设施监控等，对应了以上的三种指标。美团监控体系（CAT监控系统），其监控的内容也包括了日志查询、资源监控、服务端监控等，其Trace监控也是CAT监控系统的核心。其他公司的监控平台分享也会涵盖着三者，不再一一介绍。

## 系统架构

在第一部分分析了监控系统要收集及分析哪些指标，这里分析如何收集这些指标。在听分享的过程中，很容易发现每个公司都着重强调监控系统的“实时性”。在该部分内容，本人参加了阿里的“下一代大数据实时监控系统”、苏宁的“大企业级别立体式监控的构建”及随手记的“统一监控平台Focus”的分享。

**1）随手记监控系统架构**

其中随手记的系统架构体系更能反映上面的三个指标的不同收集及处理过程，如下图所示：



随手记监控系统监控

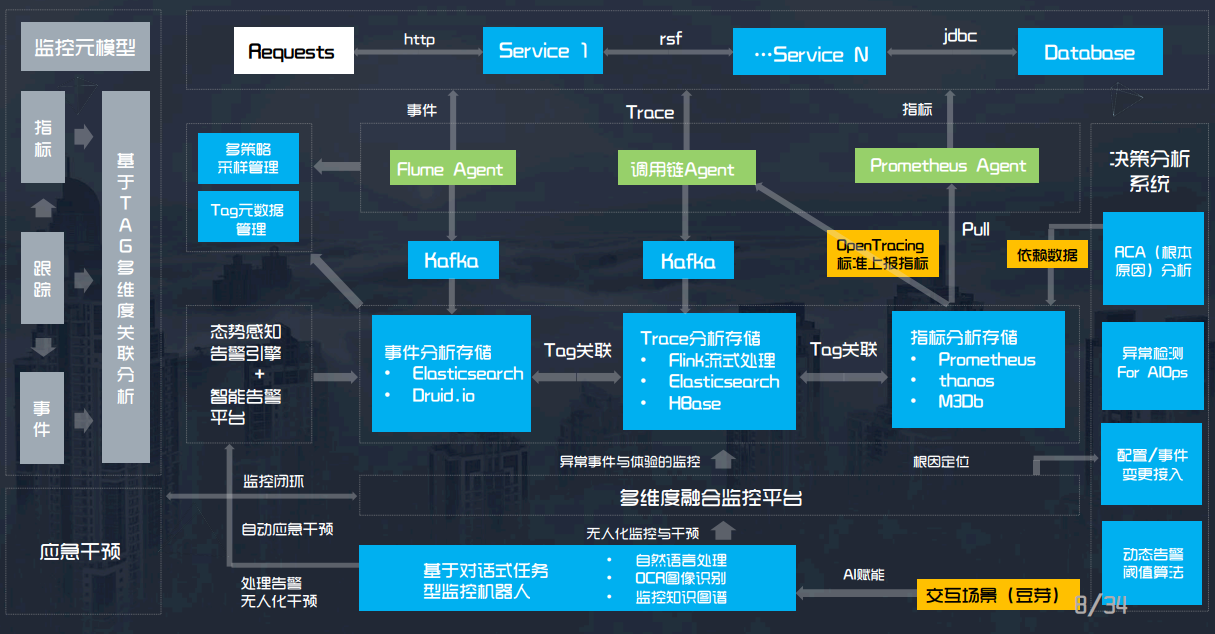
可以将监控平台的系统架构分成以下几层：

* 收集层，对监控指标的收集分别使用了Logstash、Brave(Zipkin)和Metric SDK
* 传输层，收集层收集到指标后，将其存入Kafka
* 分析层，随手记的分析分成两个层次，对收集到的指标通过相应技术进行预处理获取初始结果，然后再通过Spark/ES对结果进行挖掘及查询
* 分析存储，分析层对结果进行处理后将结果存储到数据库中，并通过前台监控平台展示给用户。其中Tracing指标存储到Cassandra、Metric数据存储到Influx DB、日志数据存储到ES中。
* 数据展示层，便于开发者及运维人员查看，提供UI界面，如Kibana/Grafana等

上面的系统架构图中，可以清晰的看到每种指标的处理链路。

1. 苏宁监控的系统架构

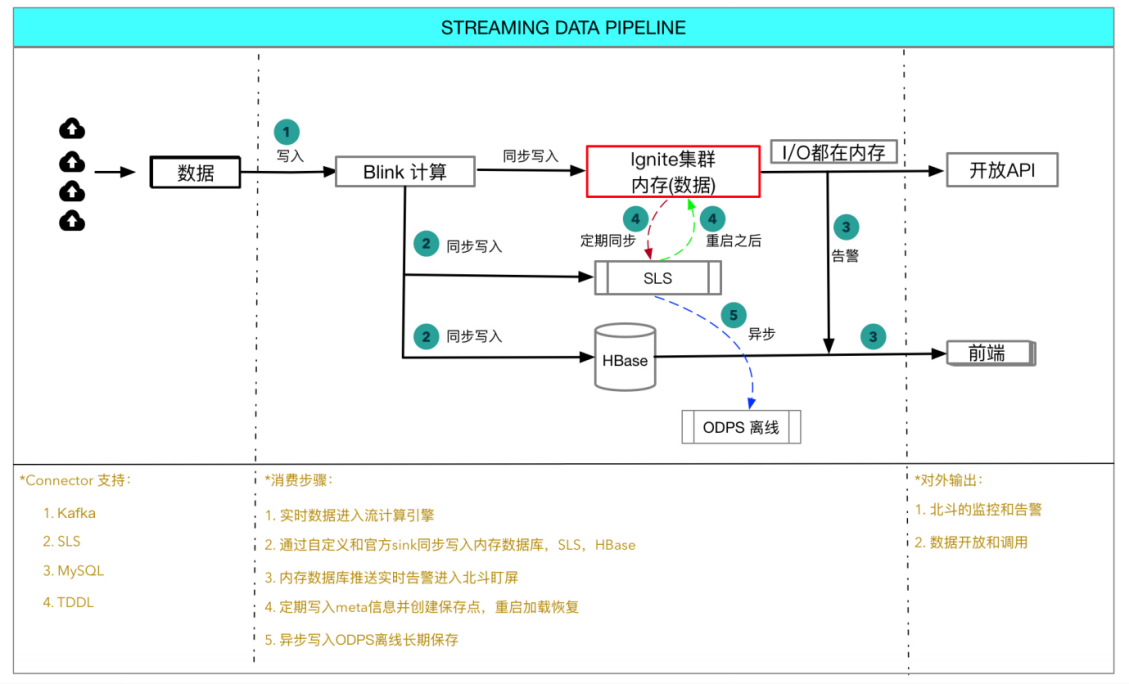
苏宁监控体系的系统架构也比较清晰，如下图：



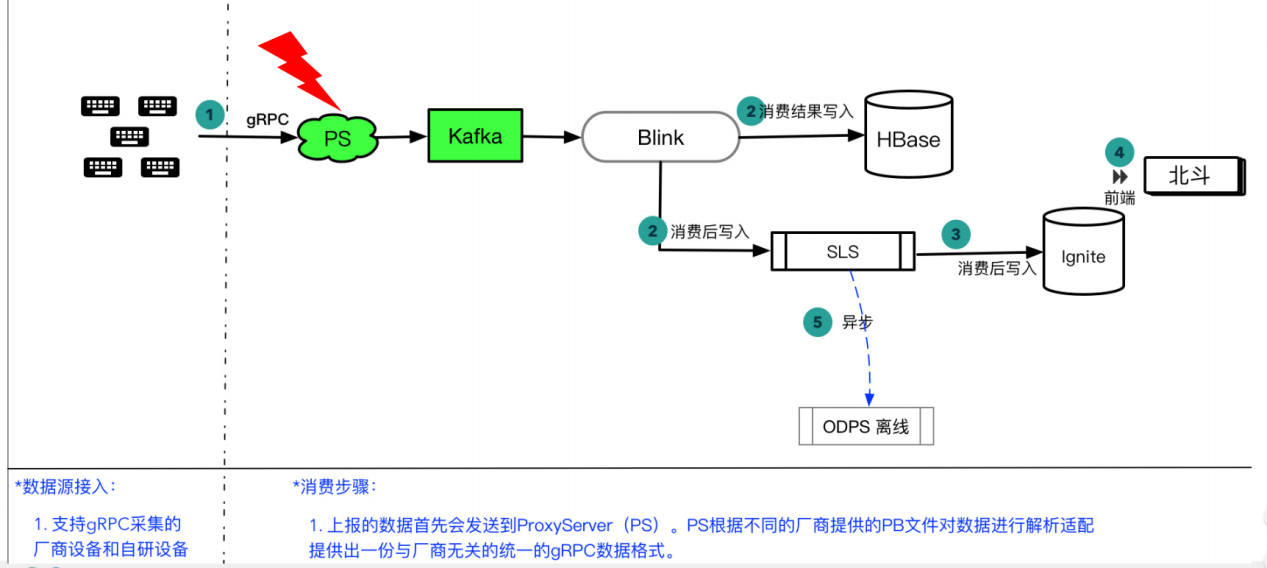
* 采集层，分别使用Flume Agent、Tracking Agent及Prometheus Agent采集日志、调用链及metric指标
* 传输层，与“随手记”相同，采集层将数据传输到Kafka中
* 分析层，对不同的指标不同的分析技术，日志：Eleasticsearch；Tracing，Flink及ES；Metric，使用Promethues
* 分析结果存储，数据分布存储到Druid.io、HBase及M3Ob中
* 数据展示层

1. 阿里，下一代大数据实时监控系统

该监控系统只是针对网络性能的监控，其架构中使用大数据技术，也是本人着重听的议题，其系统架构如下：



其采集的数据也是存入Kafka中，如下面的框架所示：



其计算层使用了Blink(Apache Flink的优化实现)，比较新颖的技术是使用了Ignite内存数据网格来存储数据，处理结果存入内存中，以供前端进行查询。

1. 总结，根据以上几个典型架构的分析，当然还有其他监控系统包括美团的CAT(已开源)，喜马拉雅监控架构等，其架构与上述的几种大同小异，基本上都采用主流的技术

* 采集层，Flume Agent、LogStash、Brace（Zikpkin）等，各公司也会定制开发采集Agent
* 传输层，基本上都是使用Kafka作为消息传输系统
* 分析层，为了保证实时性，常用Spark作为数据处理引擎，其他架构如Flink/Blink，这些系统的特点是支持SQL，基于内存计算，计算处理速度较快。Kafka Streaming计算目前也逐渐被使用。
* 数据存储层，Trace指标基本上使用NoSQL数据库（HBase/Cassandra）等，Metrics指标使用时序数据库（Opentsdb/Influx），日志使用ES存储
* 数据展示层，前端没有什么统一标准，其实现相对简单

需要注意的是对于Trace的采集基本是通过API的方式嵌入到代码中，目前很好是公司使用字节码侵入的方式实现。

## 监控信息的分析和目标

以上分析了监控系统的数据采集与系统架构，但是使用监控系统为了一方面是为了提供运维效率及便于开发人员查看问题，实现系统的实时监控预警；另一方面是为了保证系统的高效稳定的运行，根据监控指标进行系统的恢复。针对第一点，系统的实时监控预警，基本上没有什么异议，其实现也相对简单，即通过定义预警规则进行系统的异常预警，在这里不再介绍。

监控系统目前已经能够做到实时监控系统异常，但是如何根据这些系统异常进行系统的调整，目前大部分的公司都是运维人员进行主动的调整，显然不能保证实时解决问题。因此需要通过监控系统进行故障自愈，在会议中百度“信息流产品动态降级以及故障自愈”及阿里搜索平台在DevOps&AIOps进行了阐述，这里总结如下监控系统的发展趋势：

1）故障的自动感知预警，运维人员进行分析并执行预案，这也是大多数公司达到的程度

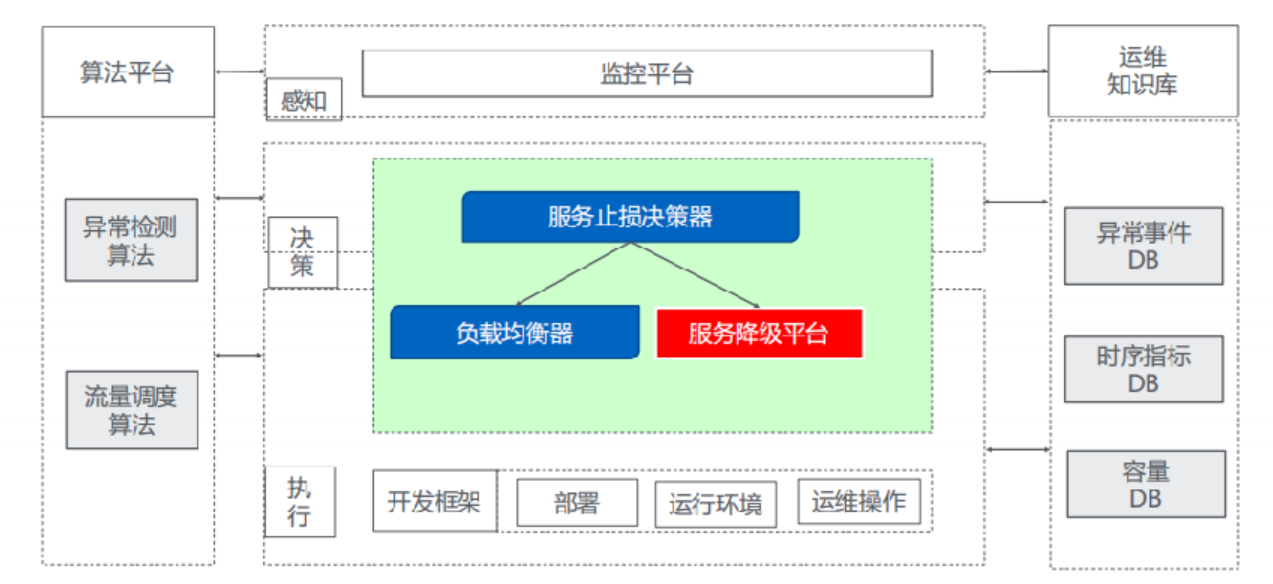
2）自动预案，指标异常，自动感知并执行预案或者流量切换

3）弹性自愈技术，根据监控结果进行智能定位，在绝大多数故障发生前分析出故障关联的源事件并形成止损决策集

4）完全智能，结合弹性自愈，定位根因，基于AI实现全系统故障智能自愈和止损

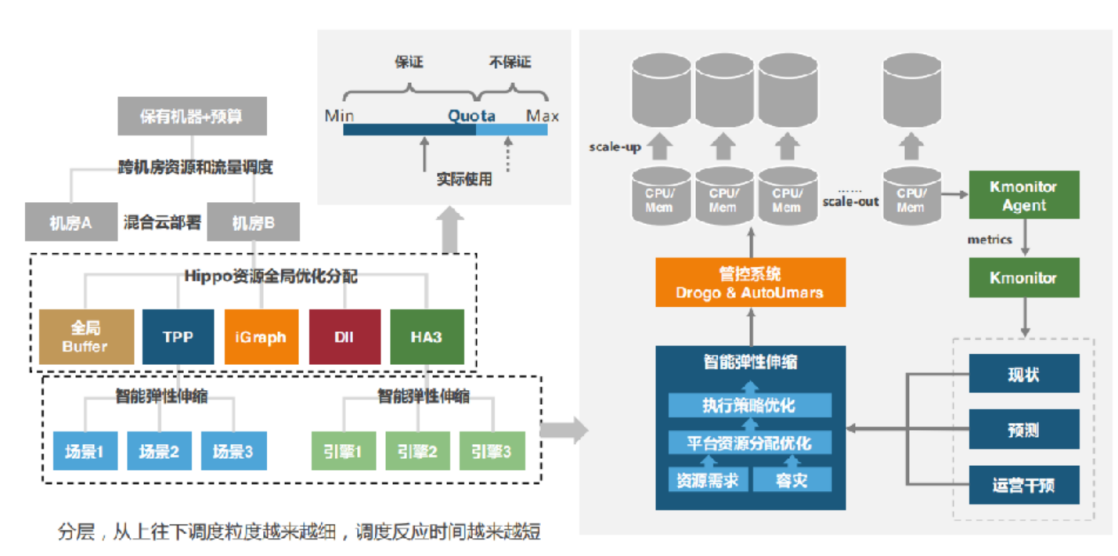
监控系统的最终目标是完全智能，但是其实现自动处理故障本身是很复杂的事情，也是目前监控系统的发展方向。

百度公司在分享了其自动运维的案例，服务的自动止损方案，如下图：

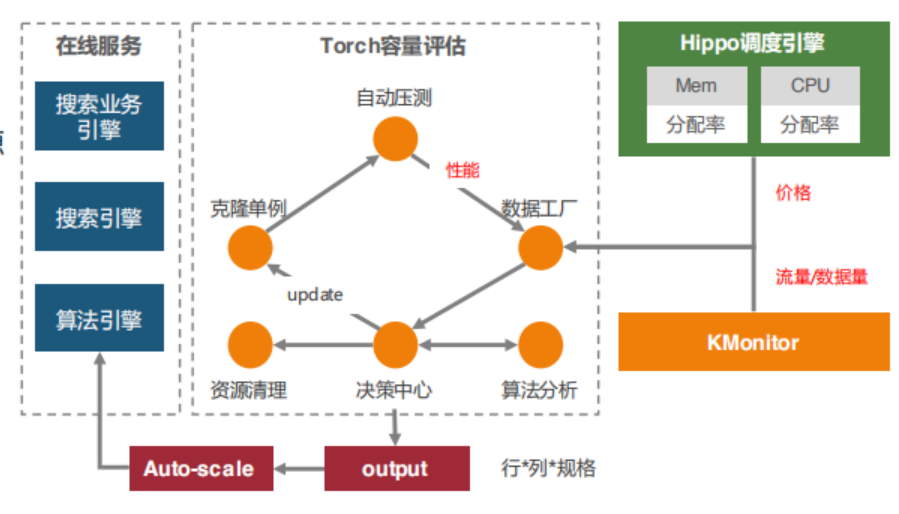


动态降级是缩减出问题系统的能力，并将请求切换到其他系统中。

阿里的AIOps系统中实现弹性可扩容方案，如下图所示：



其根据监控系统中获取的业务的变化，自动申请合理的容器资源，如图：



可以看到，目前监控系统智能化是主流分析，其结合AI能够提高系统的可用性及性能，也是目前监控系统的最终目标。

# 微服务、大数据系统架构及编程

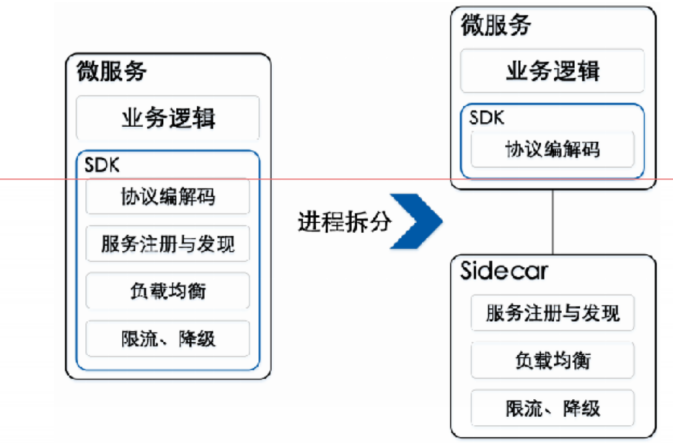
本人对大数据技术在运维管控平台的应用进行了系统的的学习，其他课题也大概的学习了下，但是没有深入的研究。

## 微服务

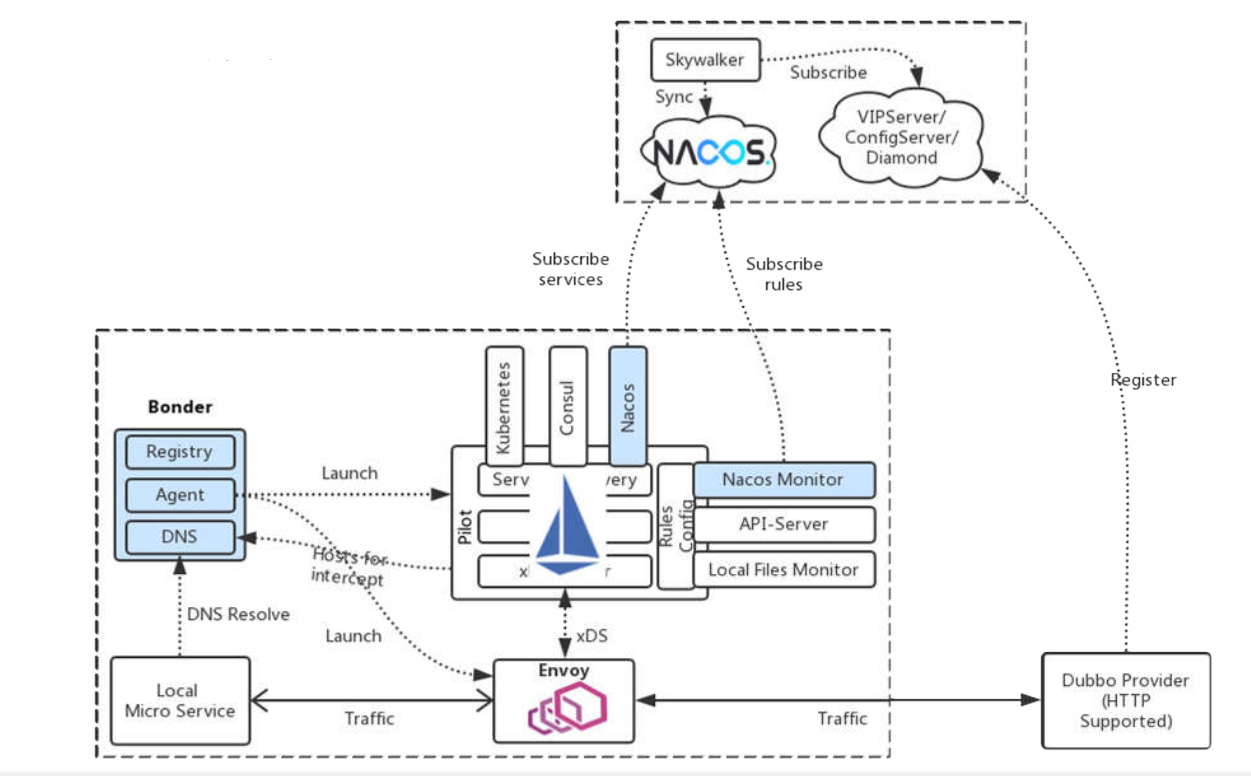
微服务是这次会议的核心主题，其中大部分的议题是关于微服务的，相关内容包括K8s在各公司的实践、Service Mesh及Serverless等前沿技术，包括运维管控平台很多也是围绕着微服务做监管。该议题中，详细的听了阿里的Dubbo Mesh和亚马逊的Serverless相关的主题研究

1. **阿里Dubbo Mesh**

Service Mesh（服务网格）为微服务提供服务间的基础设施层，用于在云原生应用复杂服务拓扑中实现可靠的请求传递，其本质是将微服务中业务逻辑以外涉及到服务管理的模块抽取出来形成系统共用模块，如下图所示：



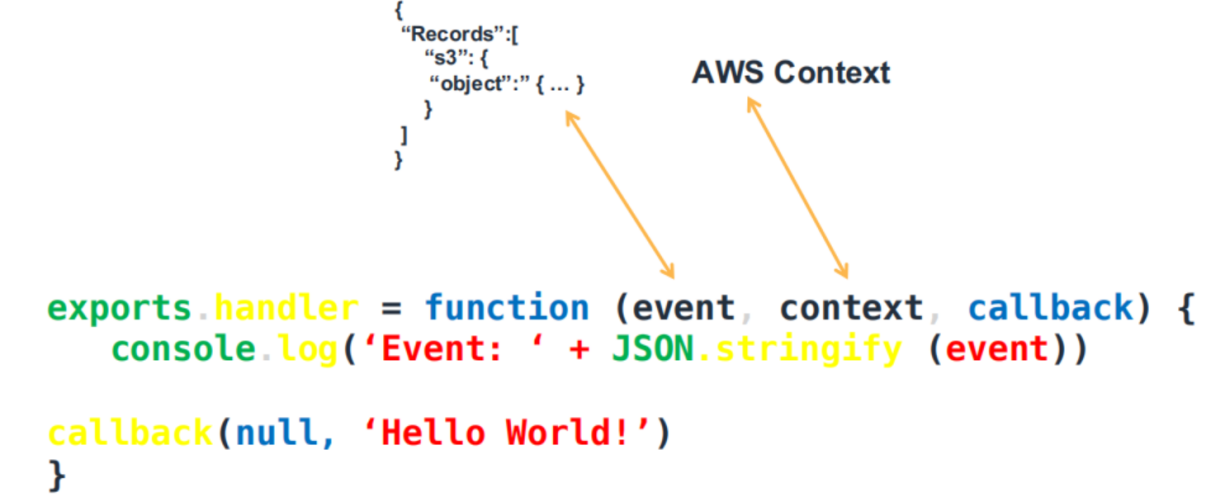
Kubernets逐渐成为微服务调度的标准，针对K8S的Service Mesh是Istio，其中核心组件是Envoy、Pilot和Mixer三部分，Envoy作为Sidecar部署到各服务的Pod中，代理所有进出流量。Dubbo是阿里开源的分布式服务框架，提供高性能和透明的RPC远程服务调用，因此阿里对Dubbo与Envoy进行结合，使Envoy支持Dubbo协议，如下图所示：



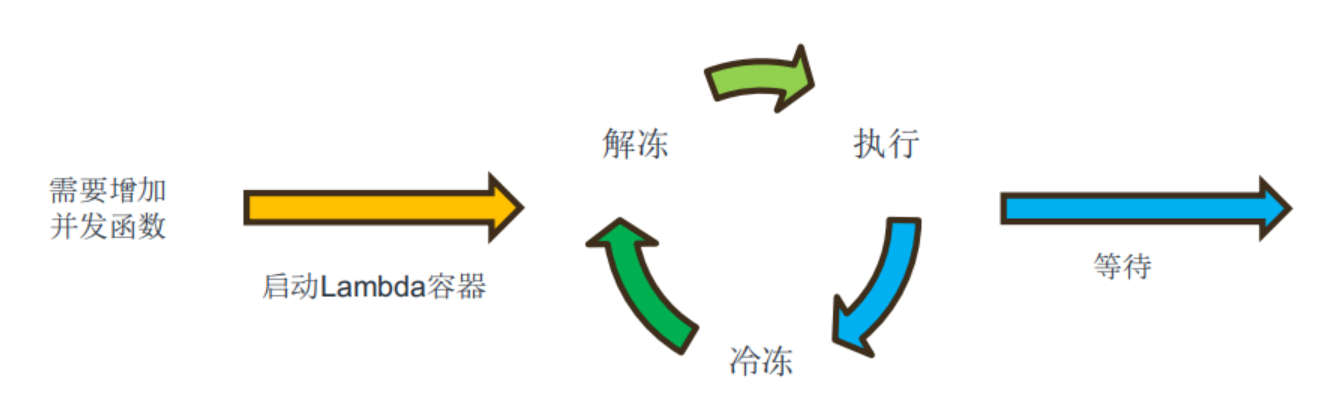
1. **AWS Serverless**

Serverless平台允许用户运行应用程序，包括计算、存储和网络，无需用户启动和管理单个机器，其中AWS的Serverless架构是目前在云平台已经比较成熟，因此选择性的听了AWS的“基于无服务器的架构落地与实践”。

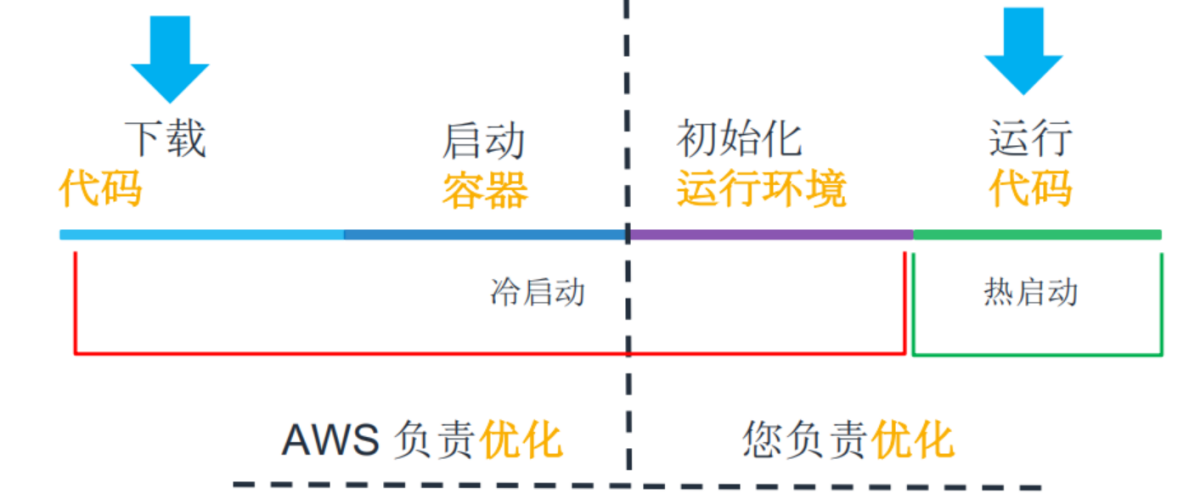
AWS的Serverless基于AWS Lambda开发，其代码示例如下：



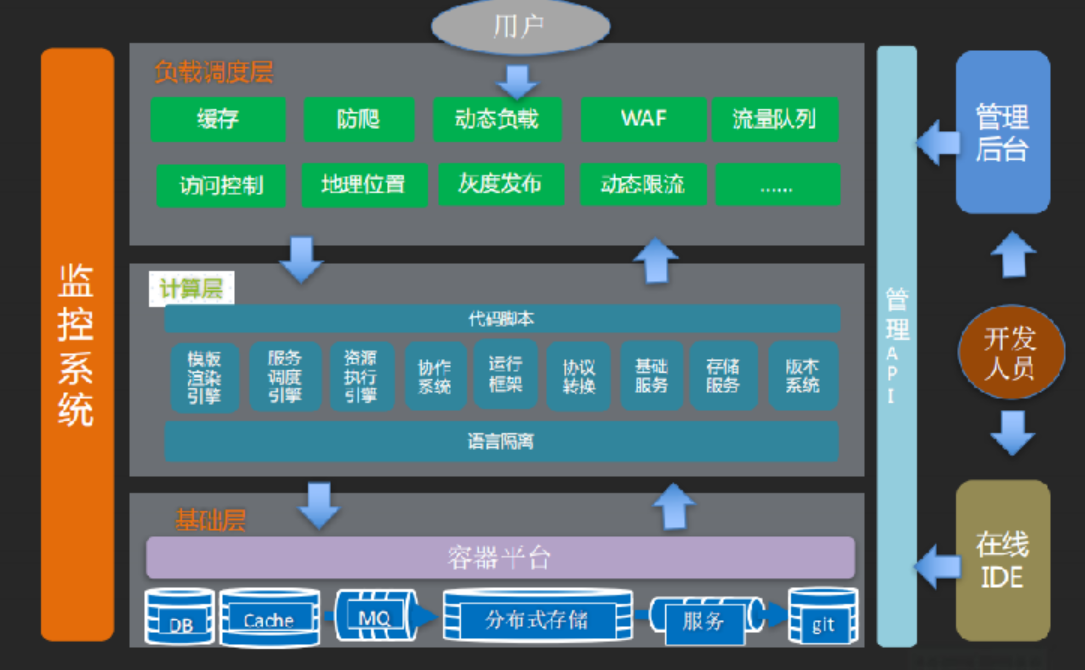
开发者通过定义Lambda函数，并提交到AWS Serverless平台即可运行定义的服务，在平台中其Lambda的生命周期如下图：



其初始化过程如下图：



Serverless是微服务领域的另外一个前沿技术，目前各个平台都推出了相应的组件，例如在K8s上运行的Knative组件，Spring Cloud Function等，在听各个议题的过程中，发现目前各公司对Serverless大多数在探究阶段，其使用实例不太多，其中同程艺龙分享了”Serverless架构应用与实践”，其平台架构如下所示：

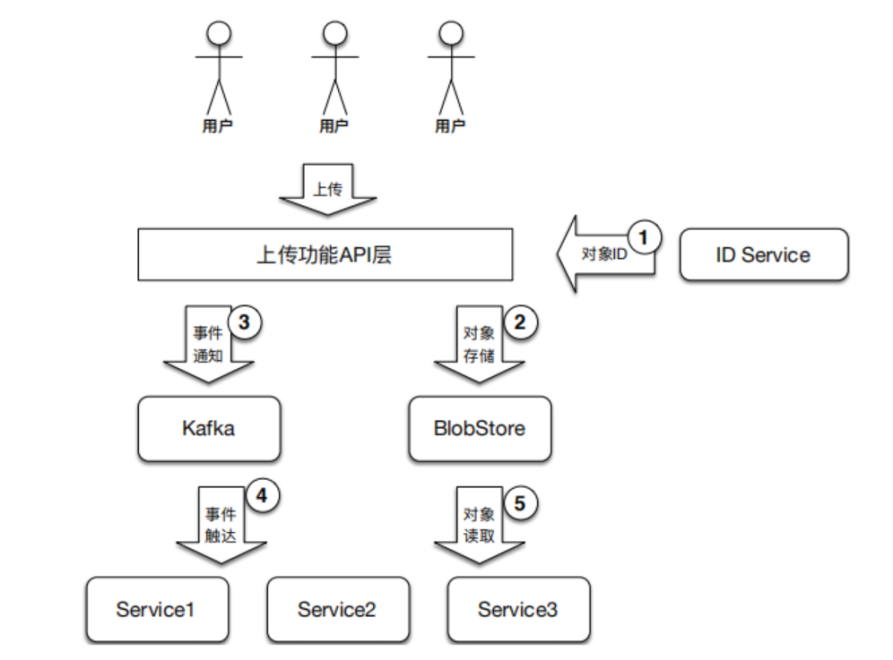


据介绍，同程的Web应用及一些逻辑简单的业务服务运行在该平台上。

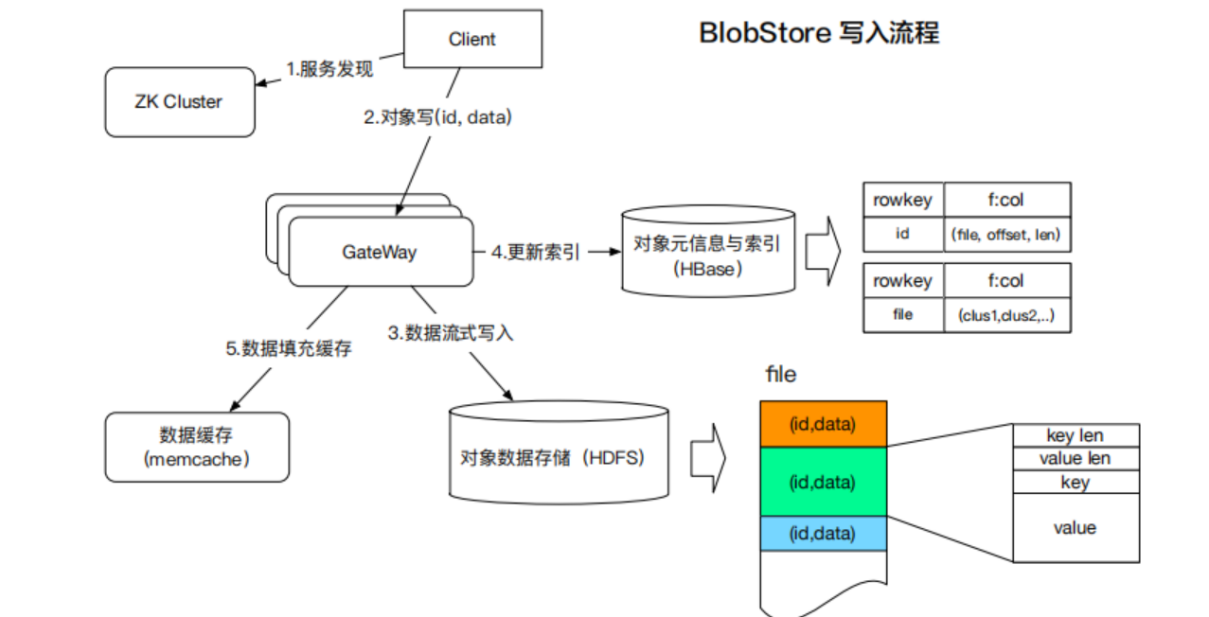
## 大数据系统架构

QCon会议中没有出现Hadoop、Spark等新技术等相关议题，与大数据技术相关的大部分是如何使用大数据技术为微服务架构服务，如服务监控、AIOPS等，还有一些议题是如何使用大数据技术搭建系统平台，着重听了快手公司的“如何快速打造高稳定性千亿级别对象存储平台”。

快手公司的业务系统用于存储和查询视频文件，因此需要海量数据存储需求，其使用大数据技术进行数据的存储和查询，其中涉及到技术包括：Kafka、HBase及Hadoop等，使用其构建一个对象存储平台，其业务流程如下：

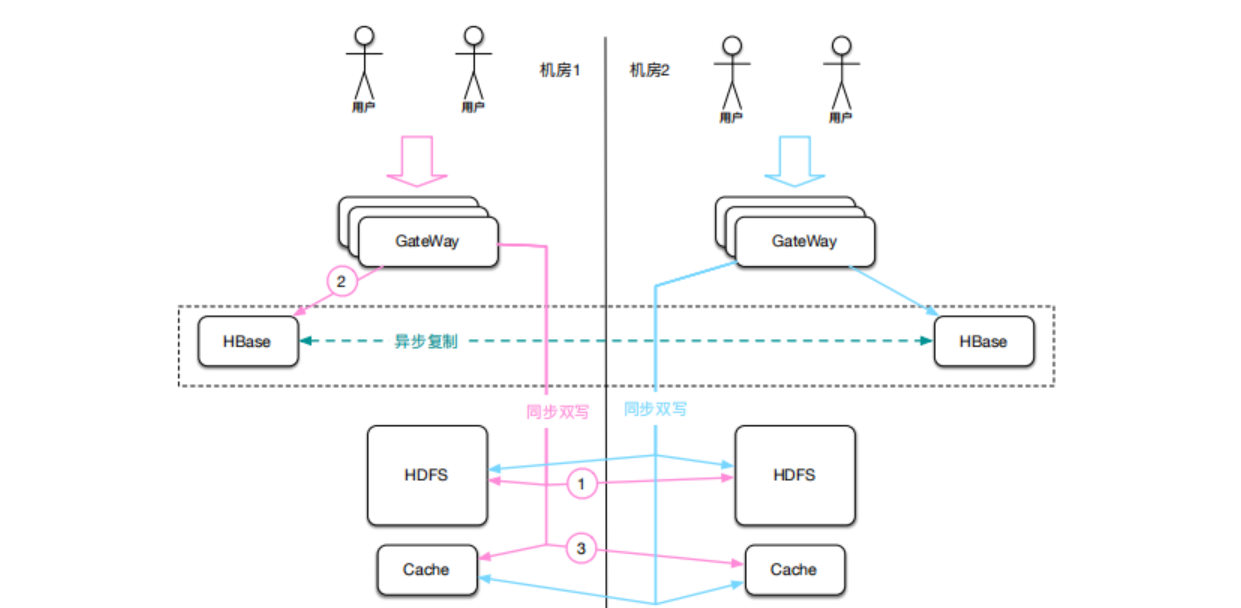


对象存储平台Blob Store使用HBase存储对象元数据与索引，使用HDFS来存储视频的对象数据，其系统架构如下：



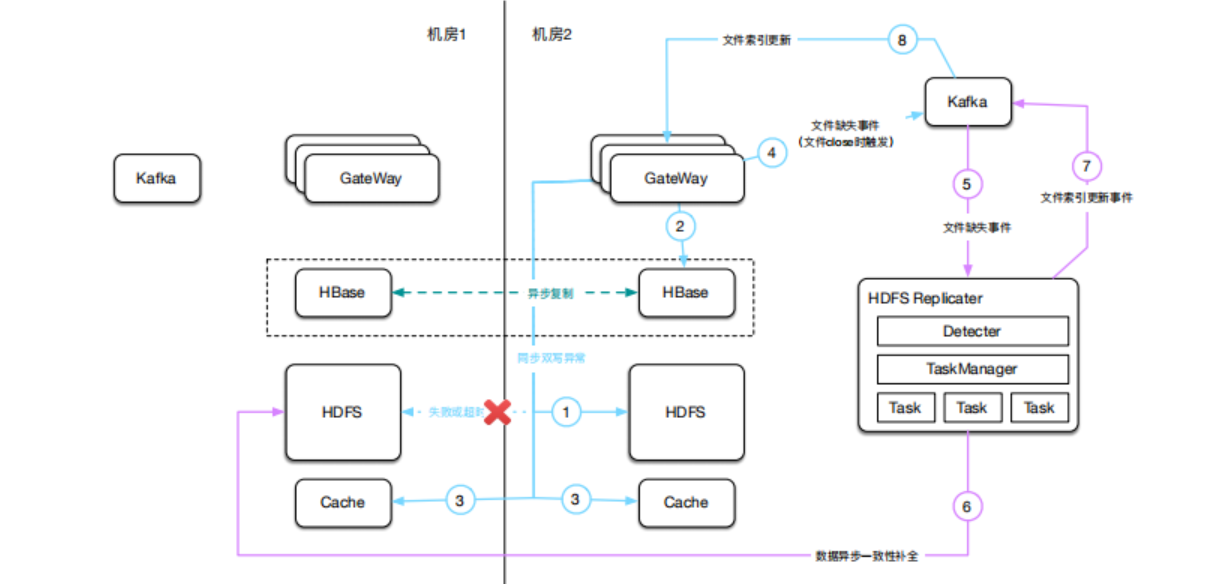
客户端写数据时，先将数据写入HDFS，然后构建对象数据的索引，将其存入HBase中，架构看起来简单实用。

为了保证系统的高可用性，快手对BlobStore采用了双机房的策略，如下图所示:



同步写入数据，当业务量需要扩展机房时，也是直接增加一个Hadoop集群。HBase的同步机制采用其开源的异步复制技术。

针对机房数据不一致的情况，快手开发了HDFS Replicater组件就那些数据的同步，如下图所示：



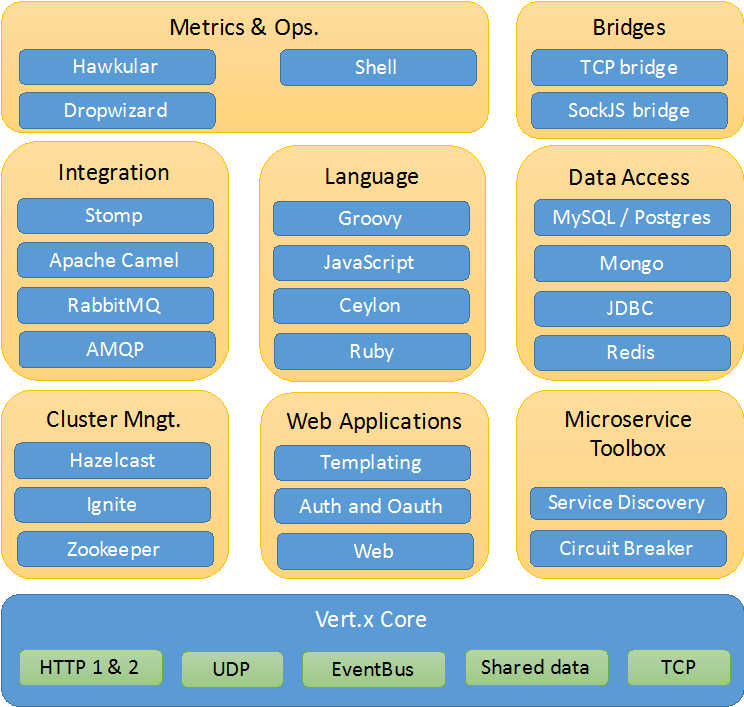
还有其他大数据相关议题，例如“百度新一代流式计算系统DStream3”等不再介绍

## 编程技术

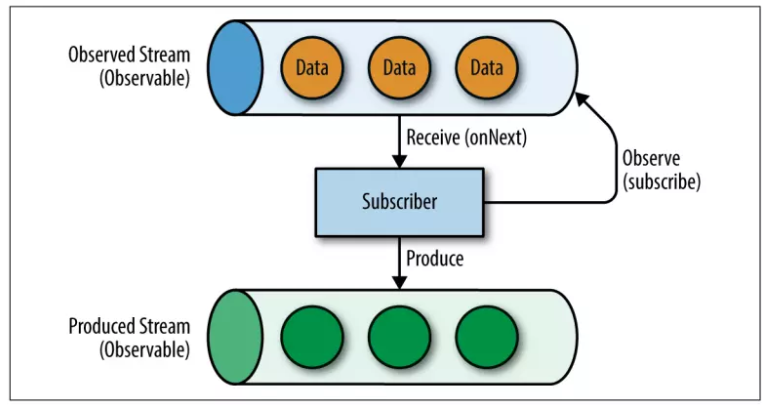
既然是软件开发大会，自然要有编程语言相关的议题，包括”构建Java API的艺术”，“The past、present、and future Of GO”，“Practical GO”等，可以看出在微服务领域GO语言已经成为开发了主流语言，这也与Kubernetes成为微服务的主流架构有关系。当然也有其他架构，如Spring Cloud，Vert.x等，其主要的开发语言是Java。由于对Go语言不怎么了解，因此该部分议题没怎么去听，java方面的议题听了“Reactive Applications With Eclipse Vert.x”，主要是对Reactive这个概念比较感情趣。

Reactive(响应式)编程是一种新的编程风格，也是目前开发领域比较热的领域，Spring WebFlux、RxJava等。Reactive应用是基于事件驱动的程序，其随时准备接受及处理事件，对事件、失败异常及用户立即响应，从而达到及时性的目的。

Eclipse Vert.x是一个在JVM开发reactive应用的框架，可用于开发异步、可伸缩、高并发的Web应用，Vert.x也是构建微服务的一种框架，其具有一个完整的生态：



Reactive是Vert.x的一个优势特点，其通过支持RxJava编程来实现及时响应的特点，RxJava是一个通过使用可观察序列来编写异步和基于事件的程序库，其核心是观察者模式的使用，如下图：



Reactive编程是一种开发模型，它以数据流和数据传播为驱动。在reactive编程中，刺激信号是数据的转移，叫做streams。

# 总结

总体来说，通过这次会议学习到了很多，看到了大数据技术在微服务领域可以发挥的作用。由于时间有限，对目前比较热的议题没有去学习，包括人工智能、区块链、前端新趋势、深度学习等。会议中也有部分议题质量不怎么高，在听之前需要先了解下，避免浪费时间。