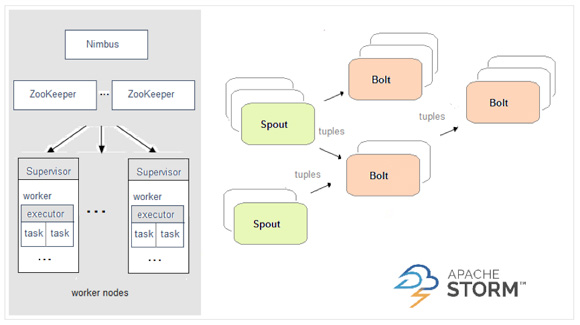
许多分布式计算系统都可以实时或接近实时地处理大数据流。本文将对三种Apache框架分别进行简单介绍，然后尝试快速、高度概述其异同。

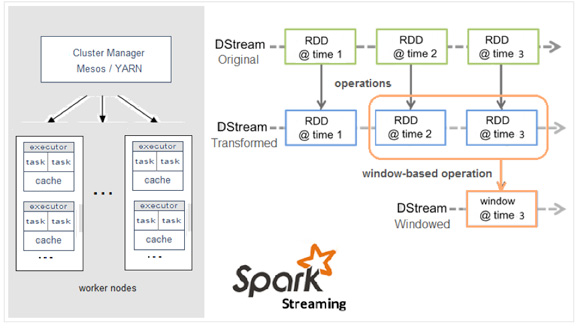
**Apache Storm**

在[Storm](https://storm.apache.org/)中，先要设计一个用于实时计算的图状结构，我们称之为拓扑（topology）。这个拓扑将会被提交给集群，由集群中的主控节点（master node）分发代码，将任务分配给工作节点（worker node）执行。一个拓扑中包括spout和bolt两种角色，其中spout发送消息，负责将数据流以tuple元组的形式发送出去；而bolt则负责转换这些数据流，在bolt中可以完成计算、过滤等操作，bolt自身也可以随机将数据发送给其他bolt。由spout发射出的tuple是不可变数组，对应着固定的键值对。



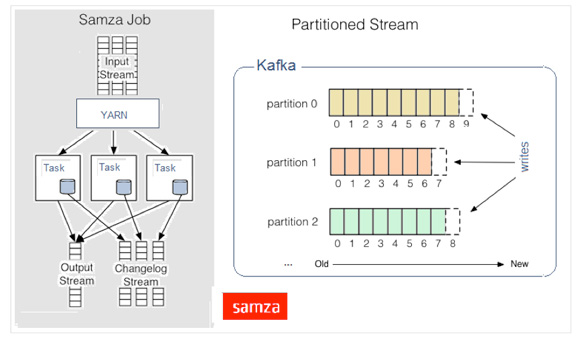
**Apache Spark**

[Spark Streaming](https://spark.apache.org/streaming/)是核心Spark API的一个扩展，它并不会像Storm那样一次一个地处理数据流，而是在处理前按时间间隔预先将其切分为一段一段的批处理作业。Spark针对持续性数据流的抽象称为DStream（DiscretizedStream），一个DStream是一个微批处理（micro-batching）的RDD（弹性分布式数据集）；而RDD则是一种分布式数据集，能够以两种方式并行运作，分别是任意函数和滑动窗口数据的转换。



**Apache Samza**

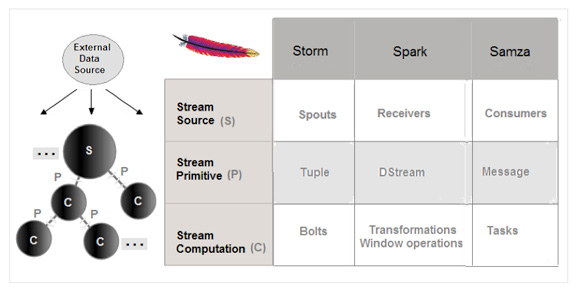
[Samza](http://samza.apache.org/)处理数据流时，会分别按次处理每条收到的消息。Samza的流单位既不是元组，也不是Dstream，而是一条条消息。在Samza中，数据流被切分开来，每个部分都由一组只读消息的有序数列构成，而这些消息每条都有一个特定的ID（offset）。该系统还支持批处理，即逐次处理同一个数据流分区的多条消息。Samza的执行与数据流模块都是可插拔式的，尽管Samza的特色是依赖Hadoop的Yarn（另一种资源调度器）和Apache Kafka。



**共同之处**

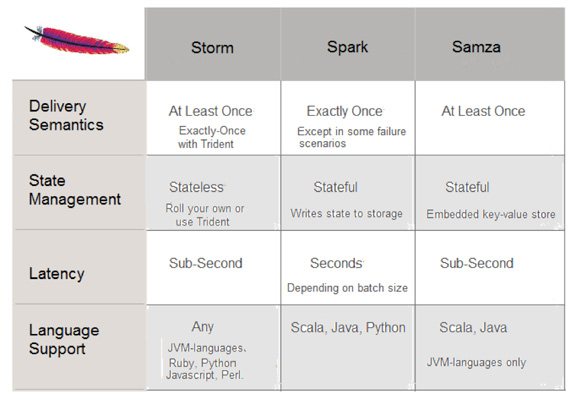
以上三种实时计算系统都是开源的分布式系统，具有低延迟、可扩展和容错性诸多优点，它们的共同特色在于：允许你在运行数据流代码时，将任务分配到一系列具有容错能力的计算机上并行运行。此外，它们都提供了简单的API来简化底层实现的复杂程度。

三种框架的术语名词不同，但是其代表的概念十分相似：



**对比图**

下面表格总结了一些不同之处：



数据传递形式分为三大类：

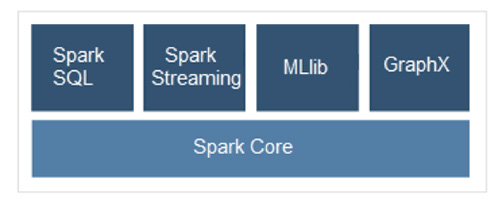
1. 最多一次（At-most-once）：消息可能会丢失，这通常是最不理想的结果。
2. 最少一次（At-least-once）：消息可能会再次发送（没有丢失的情况，但是会产生冗余）。在许多用例中已经足够。
3. 恰好一次（Exactly-once）：每条消息都被发送过一次且仅仅一次（没有丢失，没有冗余）。这是最佳情况，尽管很难保证在所有用例中都实现。

另一个方面是状态管理：对状态的存储有不同的策略，Spark Streaming将数据写入分布式文件系统中（例如HDFS）；Samza使用嵌入式键值存储；而在Storm中，或者将状态管理滚动至应用层面，或者使用更高层面的抽象Trident。

**用例**

这三种框架在处理连续性的大量实时数据时的表现均出色而高效，那么使用哪一种呢？选择时并没有什么硬性规定，最多就是几个指导方针。

如果你想要的是一个允许增量计算的高速事件处理系统，Storm会是最佳选择。它可以应对你在客户端等待结果的同时，进一步进行分布式计算的需求，使用开箱即用的分布式RPC（DRPC）就可以了。最后但同样重要的原因：Storm使用Apache Thrift，你可以用任何编程语言来编写拓扑结构。如果你需要状态持续，同时/或者达到恰好一次的传递效果，应当看看更高层面的Trdent API，它同时也提供了微批处理的方式。



**使用Storm的公司有：Twitter，雅虎，Spotify还有The Weather Channel等。**

说到微批处理，如果你必须有状态的计算，恰好一次的递送，并且不介意高延迟的话，那么可以考虑Spark Streaming，特别如果你还计划图形操作、机器学习或者访问SQL的话，Apache Spark的stack允许你将一些library与数据流相结合（Spark SQL，Mllib，GraphX），它们会提供便捷的一体化编程模型。尤其是数据流算法（例如：K均值流媒体）允许Spark实时决策的促进。

**使用Spark的公司有：亚马逊，雅虎，NASA JPL，eBay还有百度等。**

如果你有大量的状态需要处理，比如每个分区都有许多十亿位元组，那么可以选择Samza。由于Samza将存储与处理放在同一台机器上，在保持处理高效的同时，还不会额外载入内存。这种框架提供了灵活的可插拔API：它的默认execution、消息发送还有存储引擎操作都可以根据你的选择随时进行替换。此外，如果你有大量的数据流处理阶段，且分别来自不同代码库的不同团队，那么Samza的细颗粒工作特性会尤其适用，因为它们可以在影响最小化的前提下完成增加或移除的工作。

**使用Samza的公司有：LinkedIn，Intuit，Metamarkets，Quantiply，Fortscale等。**

**结论**

本文中我们只对这三种Apache框架进行了简单的了解，并未覆盖到这些框架中大量的功能与更多细微的差异。同时，文中这三种框架对比也是受到限制的，因为这些框架都在一直不断的发展，这一点是我们应当牢记的。

原文链接：[Streaming Big Data: Storm, Spark and Samza](http://www.javacodegeeks.com/2015/02/streaming-big-data-storm-spark-samza.html) （编译/孙薇 责编/周建丁 ）