Spark Streaming

OutLine

- Spark Streaming简介
- Spark Streaming架构
- Spark Streaming实践

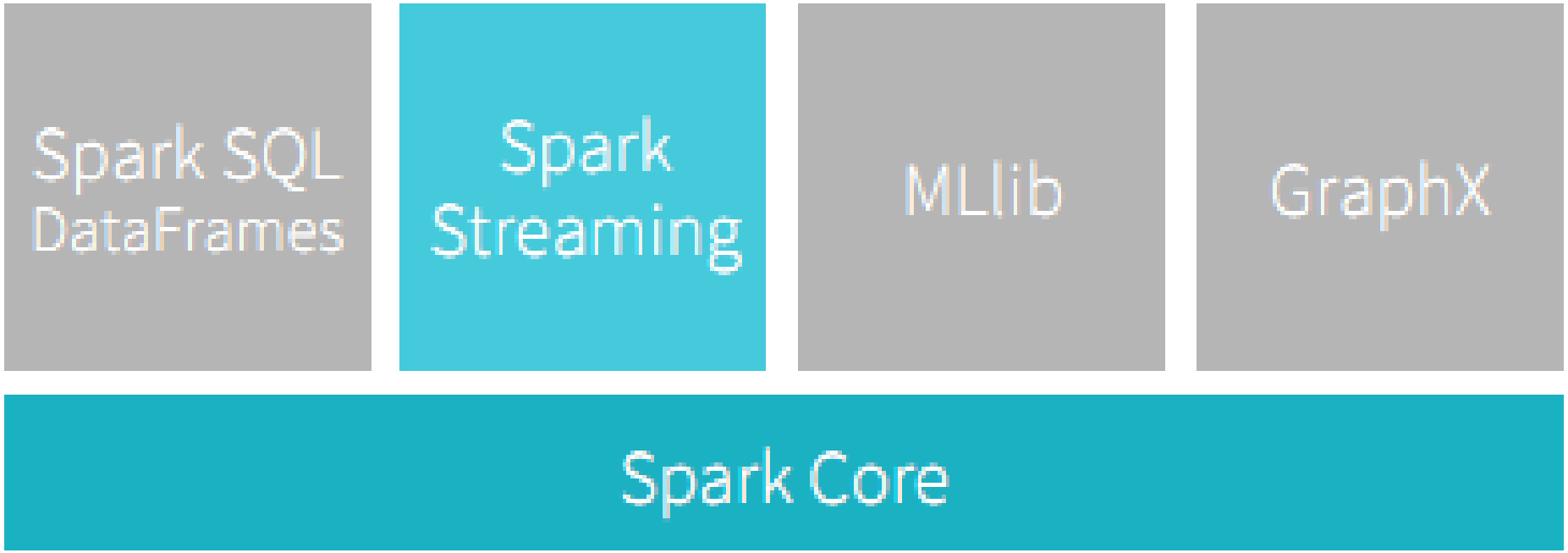
Spark Streaming简介

- Spark Streaming是Spark核心API的一个扩展，可以实现高吞吐量的、具备容错机制的实时流数据的处理
- 支持多种数据源获取数据：

- Spark Streaming接收Kafka、Flume、HDFS等各种来源的实时输入数据，进行处理后，处理结构保存在HDFS、DataBase等各种地方

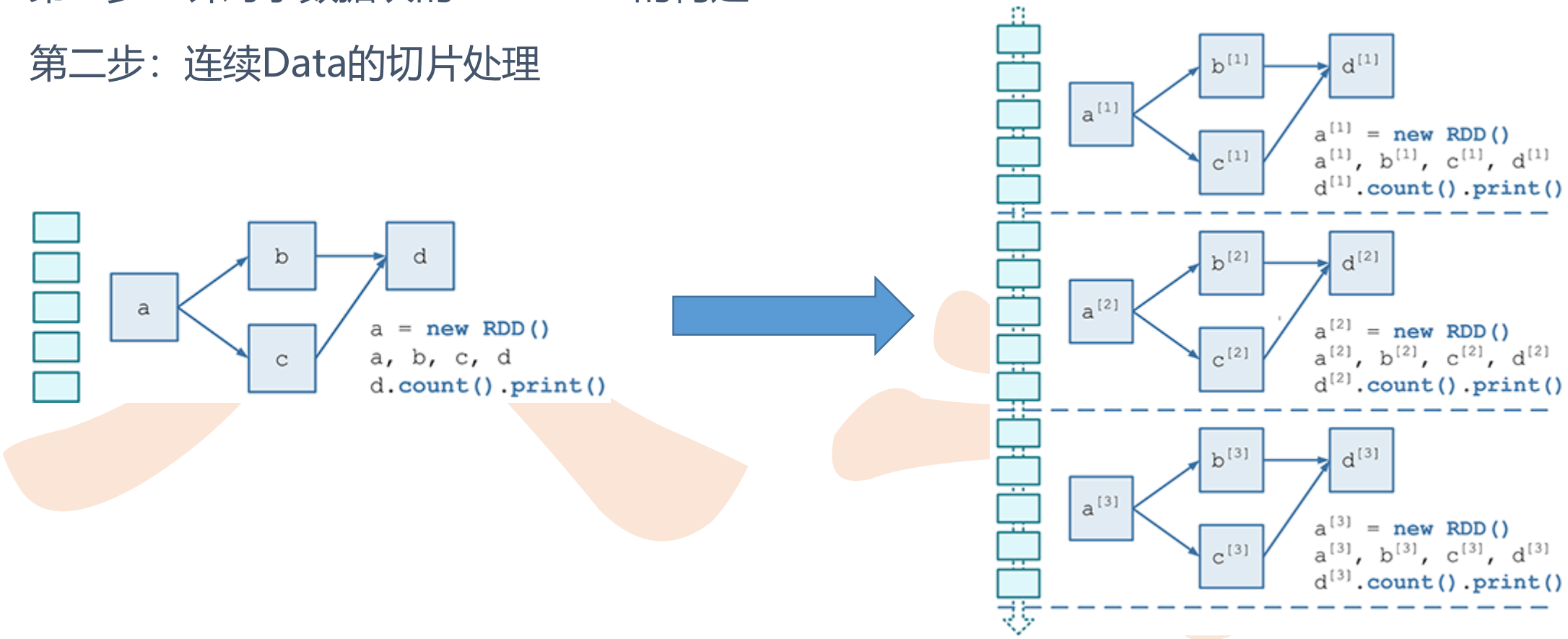
Spark Core与Spark Streaming

- 两者关系如图:



Spark Core与Spark Streaming

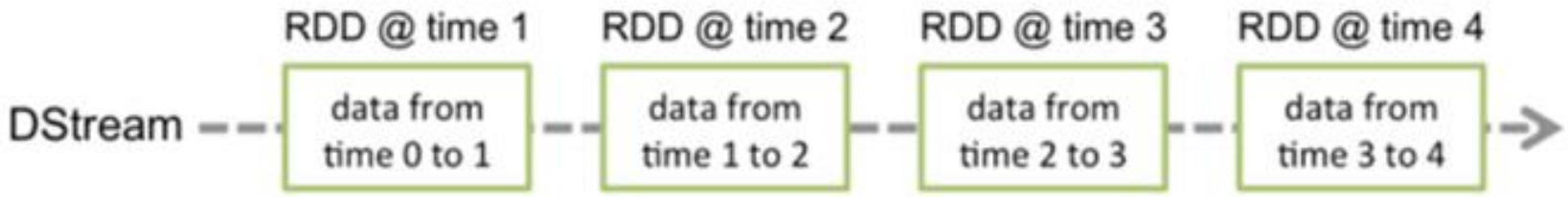
- 第一步：针对小数据块的RDD DAG的构建
- 第二步：连续Data的切片处理



Spark Streaming简介

- Spark Streaming将接收到的实时流数据，按照一定时间间隔，对数据进行拆分，交给Spark Engine引擎处理，最终得到一批批的结果

- 每一批数据，在Spark内核对应一个RDD实例
- Dstream可以看做一组RDDs，即RDD的一个序列



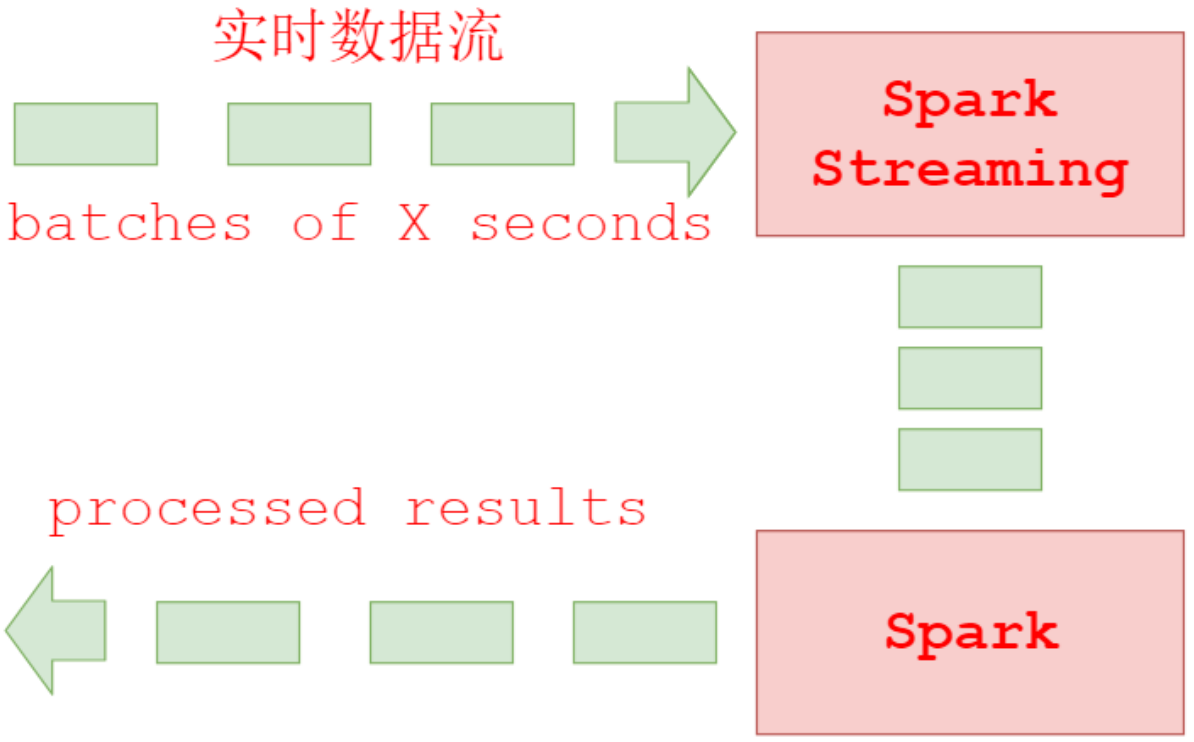
DStream

- Dstream：Spark Streaming提供了表示连续数据流的、高度抽象的被称为离散流的DStream
- 任何对DStream的操作都会转变为对底层RDD的操作

- Spark Streaming程序中一般会有若干个对DStream的操作。DStreamGraph就是由这些操作的依赖关系构成

DStream

- 将连续的数据持久化、离散化，然后进行批量处理
- 为什么?
  - 数据持久化：接收到的数据暂存
  - 离散化：按时间分片，形成处理单元
  - 分片处理：分批处理

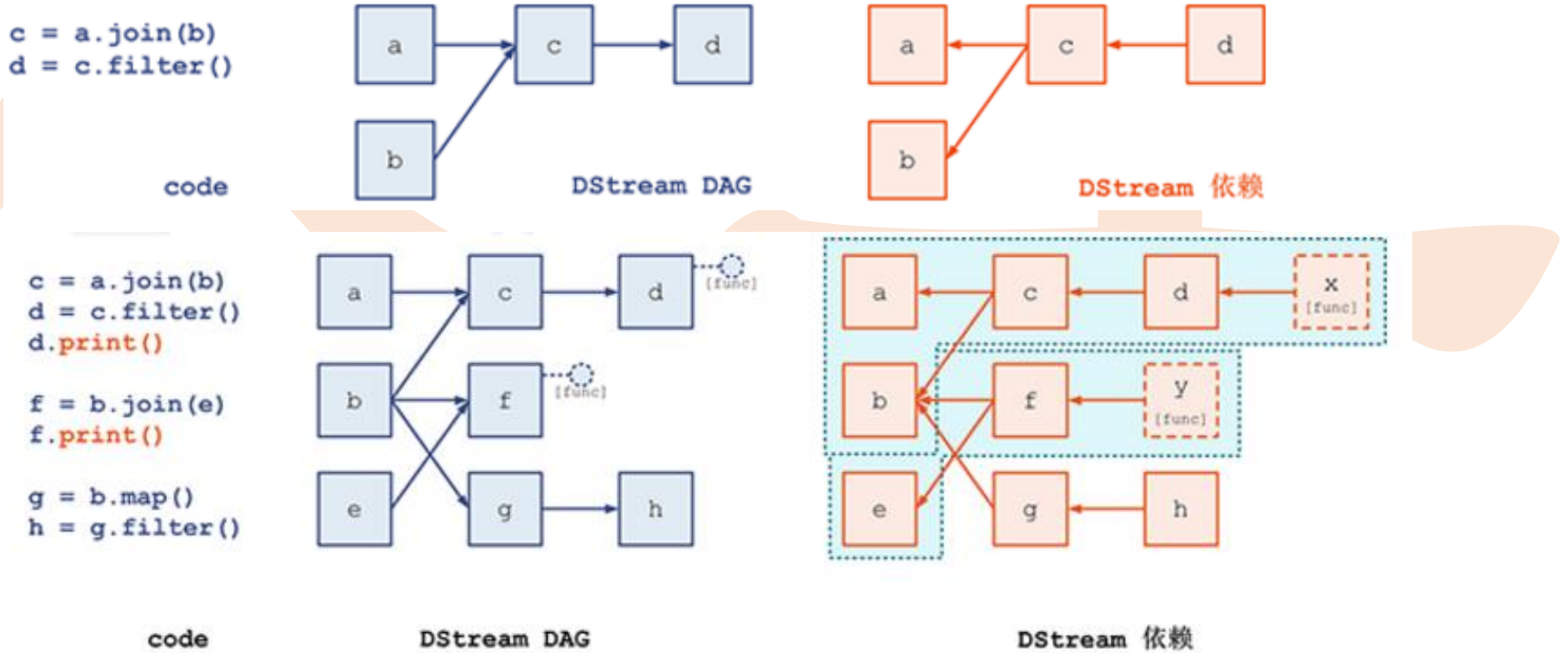


DStream

- 作用Dstream上的Operation分成两类：
  - Transformation：转换算子
    - Spark支持RDD进行各种转换，因为DStream是由RDD组成的Spark Streaming提供了一个可以在DStream上使用的转换集合，这些集合和RDD上可用的转换类似
    - 转换应用到DStream的每个RDD
    - Spark Streaming提供了reduce和count这样的算子，但不会直接触发DStream计算
    - Map、flatMap、join、reduceByKey
  - Output：执行算子、或输出算子
    - Print
    - saveAsObjectFile、saveAsTextFile、saveAsHadoopFiles：将一批数据输出到Hadoop文件系统中，用批量数据的开始时间戳来命名
    - forEachRDD：允许用户对DStream的每一批量数据对应的RDD本身做任意操作

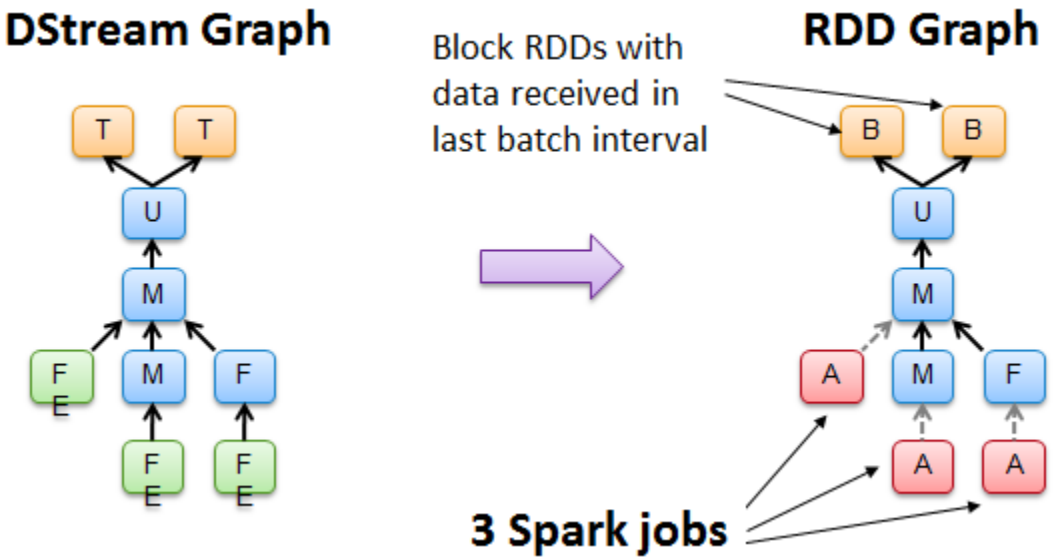
DStream Graph

- 一系列transformation操作的抽象
- 例如：
  - c = a.join(b), d = c.filter() 时， 它们的 DAG 逻辑关系是a/b → c，c → d，但在 Spark Streaming 在进行物理记录时却是反向的 a/b ← c, c ← d，目的为了追溯



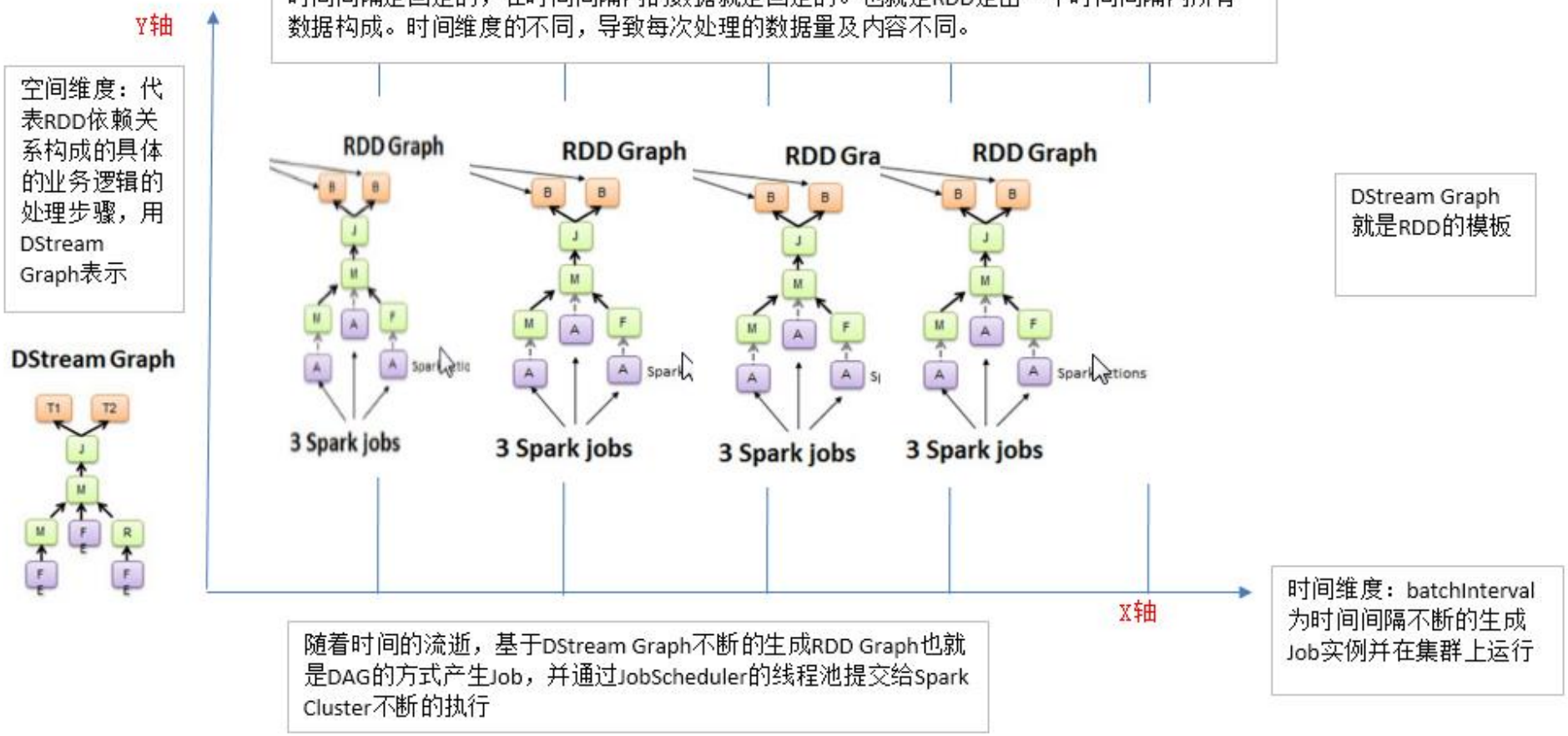
DStream Graph

- Dstream之间的转换所形成的的依赖关系全部保存在DStreamGraph中， DStreamGraph对于后期生成RDD Graph至关重要
- DStreamGraph有点像简洁版的DAG scheduler，负责根据某个时间间隔生成一序列JobSet，以及按照依赖关系序列化



DStream Graph

每个时间间隔会积累一定的数据，这些数据可以看成由event组成(假设以kafka或者Flume为例)，时间间隔是固定的，在时间间隔内的数据就是固定的。也就是RDD是由一个时间间隔内所有数据构成。时间维度的不同，导致每次处理的数据量及内容不同。



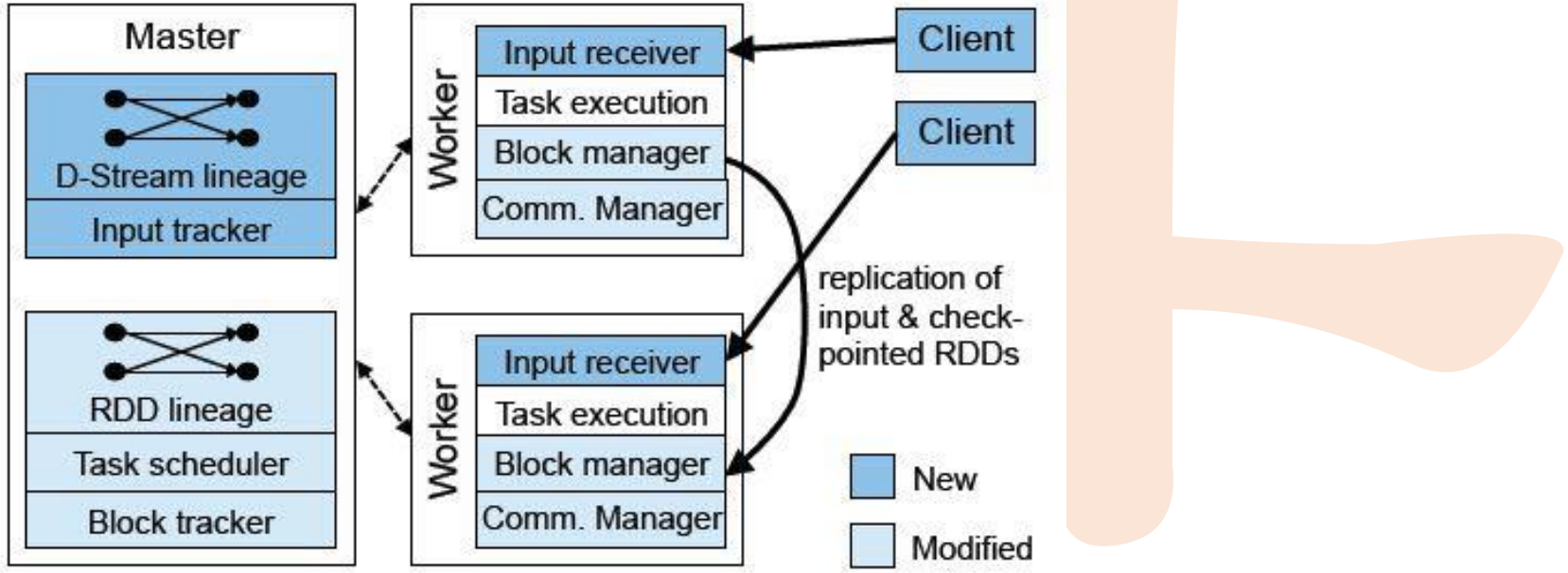
随着时间的流逝，基于DStream Graph不断的生成RDD Graph也就是DAG的方式产生Job，并通过JobScheduler的线程池提交给Spark Cluster不断的执行

OutLine

- Spark Streaming简介
- Spark Streaming架构
- Spark Streaming实践

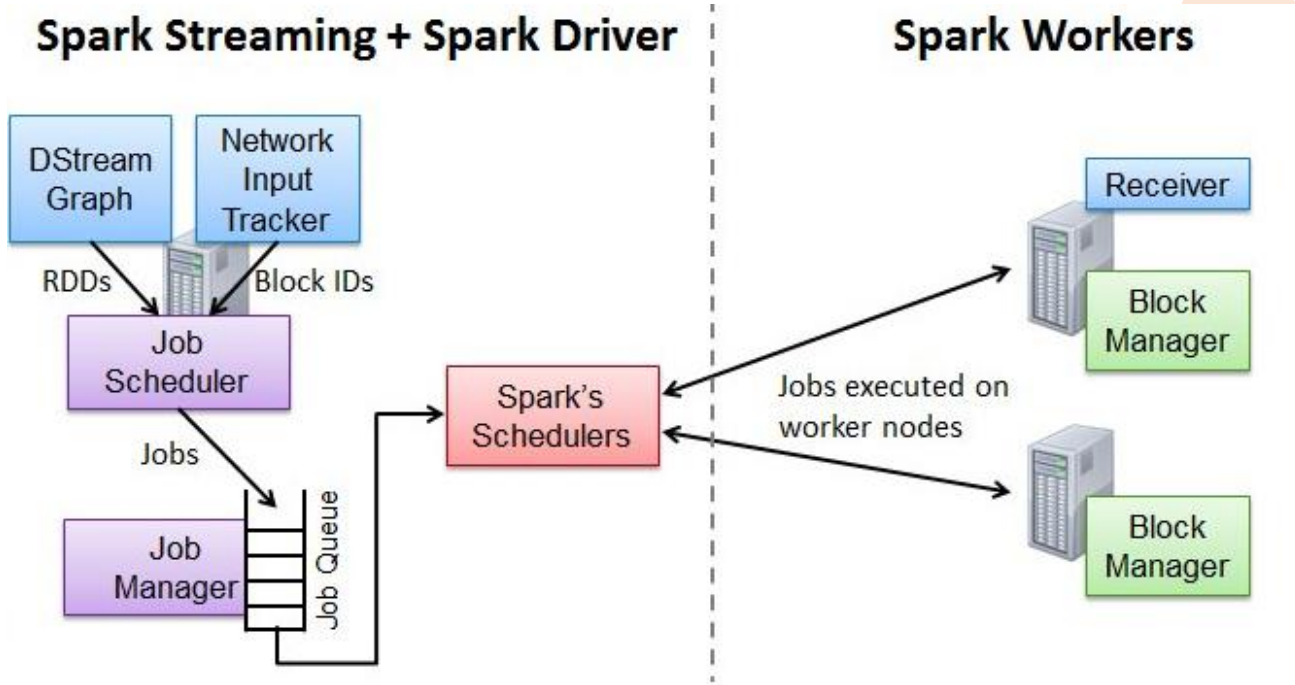
Spark Streaming架构

- 整个架构由3个模块组成：
  - Master：记录Dstream之间的依赖关系或者血缘关系，并负责任务调度以生成新的RDD
  - Worker：从网络接收数据，存储并执行RDD计算
  - Client：负责向Spark Streaming中灌入数据



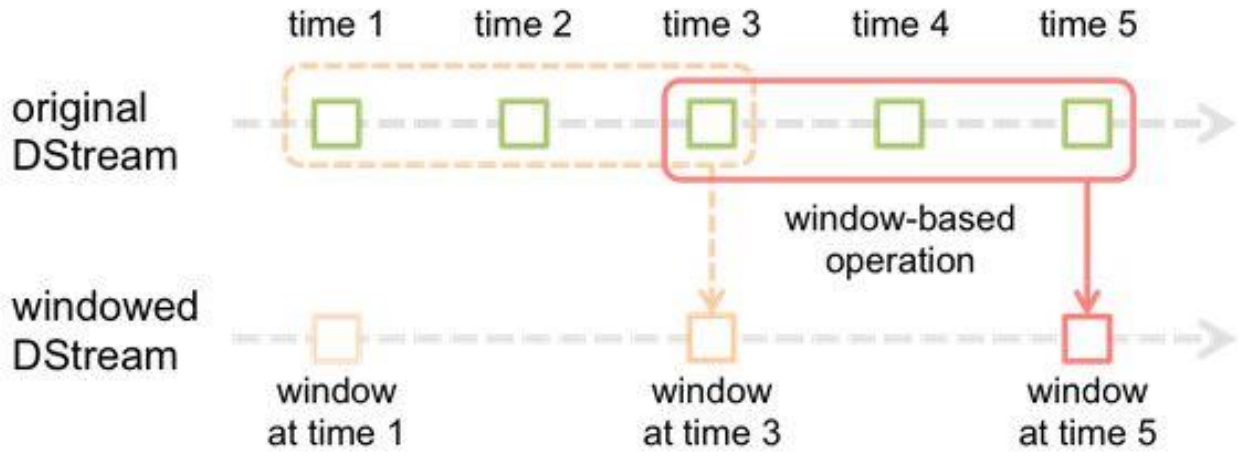
Spark Streaming作业提交

- Network Input Tracker：跟踪每一个网络received数据，并且将其映射到相应的input Dstream上
- Job Scheduler：周期性的访问DStream Graph并生成Spark Job，将其交给Job Manager执行
- Job Manager：获取任务队列，并执行Spark任务



Streaming窗口操作

- Spark提供了一组窗口操作，通过滑动窗口技术对大规模数据的增量更新进行统计分析
- Window Operation：定时进行一定时间段内的数据处理



- 任何基于窗口操作需要指定两个参数：
  - 窗口总长度（window length）
  - 滑动时间间隔（slide interval）

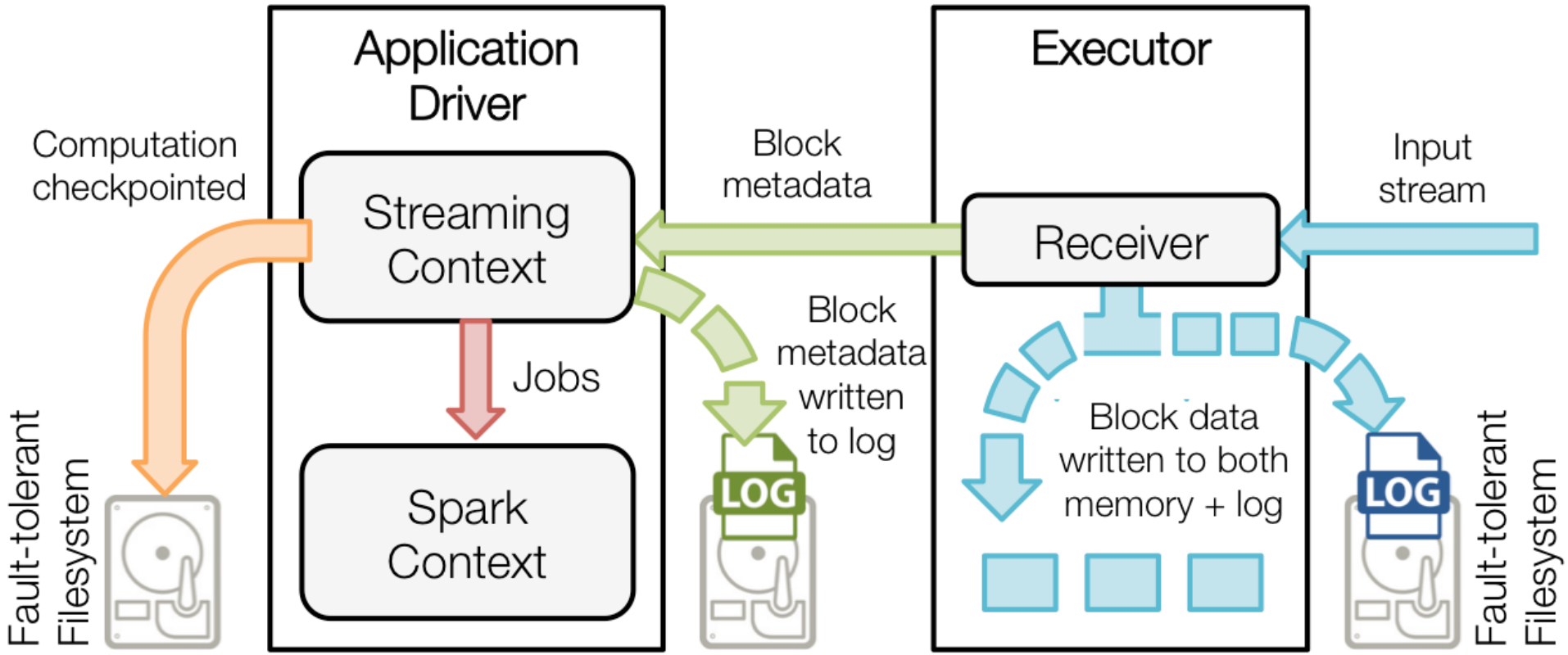
```java
// Reduce last 30 seconds of data, every 10 seconds
val windowedWordCounts = pairs.reduceByKeyAndWindow(_ + _, Seconds(30), Seconds(10))
```

Streaming容错性分析

- 实时的流式处理系统必须是7*24运行的，同时可以从各种各样的系统错误中恢复，在设计之初，Spark Streaing就支持driver和worker节点的错误恢复。
- Worker容错：spark和rdd的保证worker节点的容错性。spark streaming构建在spark之上，所以它的worker节点也是同样的容错机制
- Driver容错：依赖WAL持久化日志
  - 启动WAL需要做如下的配置
  - 1：给streamingContext设置checkpoint的目录，该目录必须是HADOOP支持的文件系统，用来保存WAL和做Streaming的checkpoint
  - 2：spark.streaming.receiver.writeAheadLog.enable 设置为true

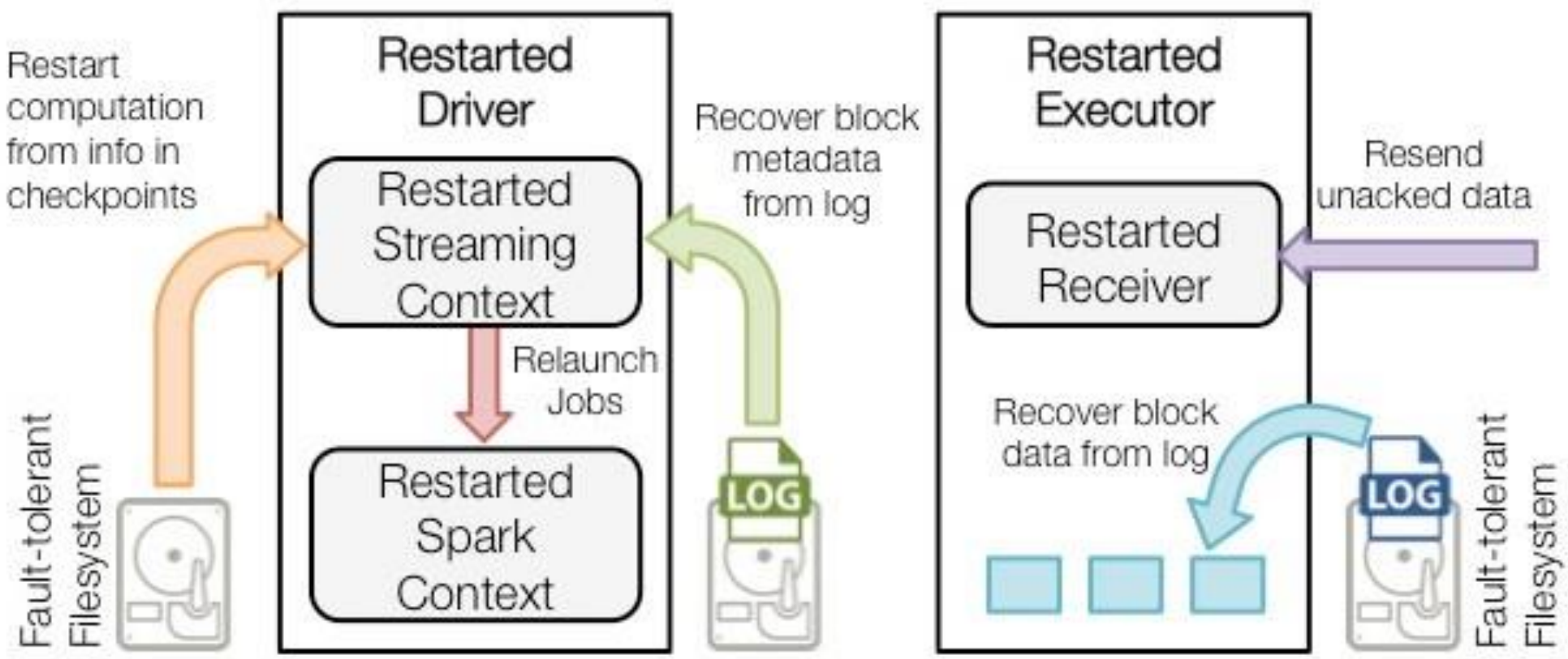
Streaming中WAL工作原理

- 流程梳理:



Streaming中WAL工作原理

- 当一个Driver失败重启后，恢复流程：



OutLine

- Spark Streaming简介
- Spark Streaming架构
- Spark Streaming实践

Q & A

@八斗学院