7.2 Spark Streaming

7.2.1 Spark Streaming设计

7.2.2 Spark Streaming与Storm的对比

7.2.3 从“Hadoop+Storm”架构转向Spark架构

7.2.1 Spark Streaming设计



- Spark Streaming可整合多种输入数据源，如Kafka、Flume、HDFS，甚至是普通的TCP套接字。经处理后的数据可存储至文件系统、数据库，或显示在仪表盘里



图7-8 Spark Streaming支持的输入、输出数据源

7.2.1 Spark Streaming设计



- Spark Streaming的基本原理是将实时输入数据流以时间片（秒级）为单位进行拆分，然后经Spark引擎以类似批处理的方式处理每个时间片数据

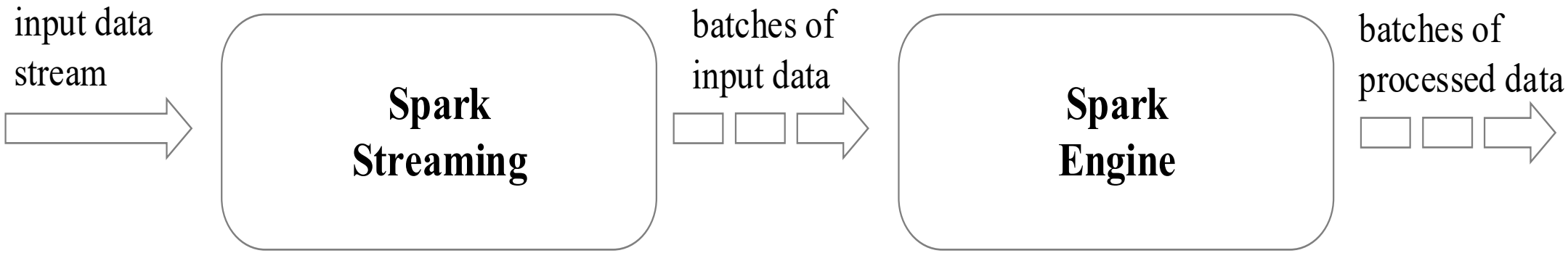

图7-9 Spark Streaming执行流程

7.2.1 Spark Streaming设计



- Spark Streaming最主要的抽象是DStream（Discretized Stream，离散化数据流），表示连续不断的数据流。在内部实现上，Spark Streaming的输入数据按照时间片（如1秒）分成一段一段，每一段数据转换为Spark中的RDD，这些分段就是Dstream，并且对DStream的操作都最终转变为对相应的RDD的操作

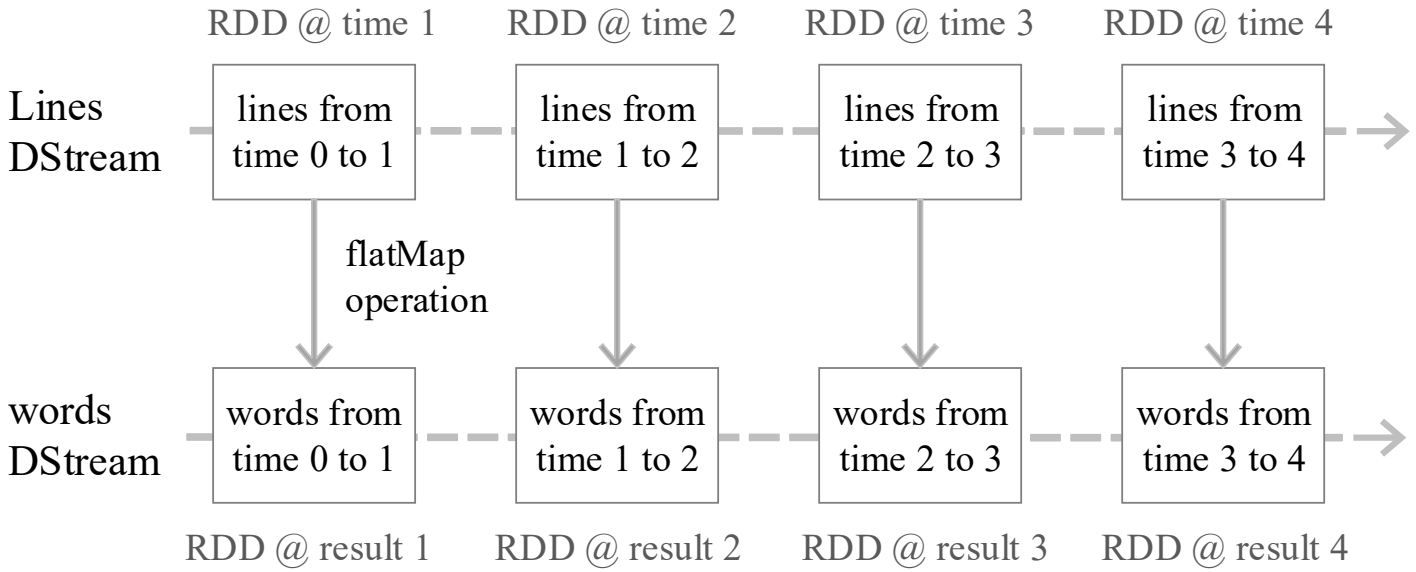

图7-10 DStream操作示意图

7.2.2 Spark Streaming与Storm的对比



- Spark Streaming和Storm最大的区别在于，Spark Streaming无法实现毫秒级的流计算，而Storm可以实现毫秒级响应
- Spark Streaming构建在Spark上，一方面是因为Spark的低延迟执行引擎（100ms+）可以用于实时计算，另一方面，相比于Storm，RDD数据集更容易做高效的容错处理
- Spark Streaming采用的小批量处理的方式使得它可以同时兼容批量和实时数据处理的逻辑和算法，因此，方便了一些需要历史数据和实时数据联合分析的特定应用场合

7.2.3 从“Hadoop+Storm”架构转向Spark架构



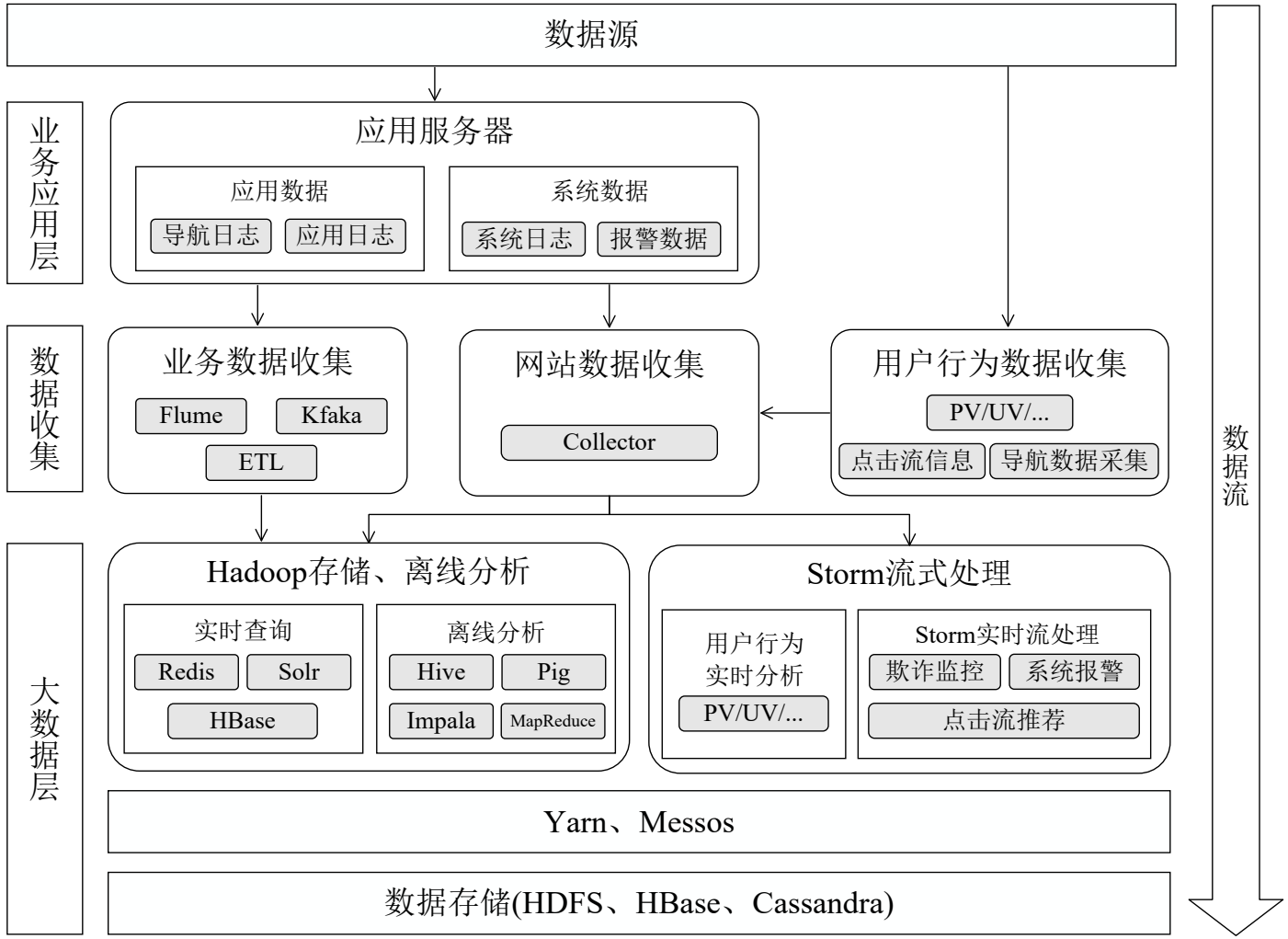

图7-11　采用Hadoop+Storm部署方式的一个案例

7.2.3 从“Hadoop+Storm”架构转向Spark架构



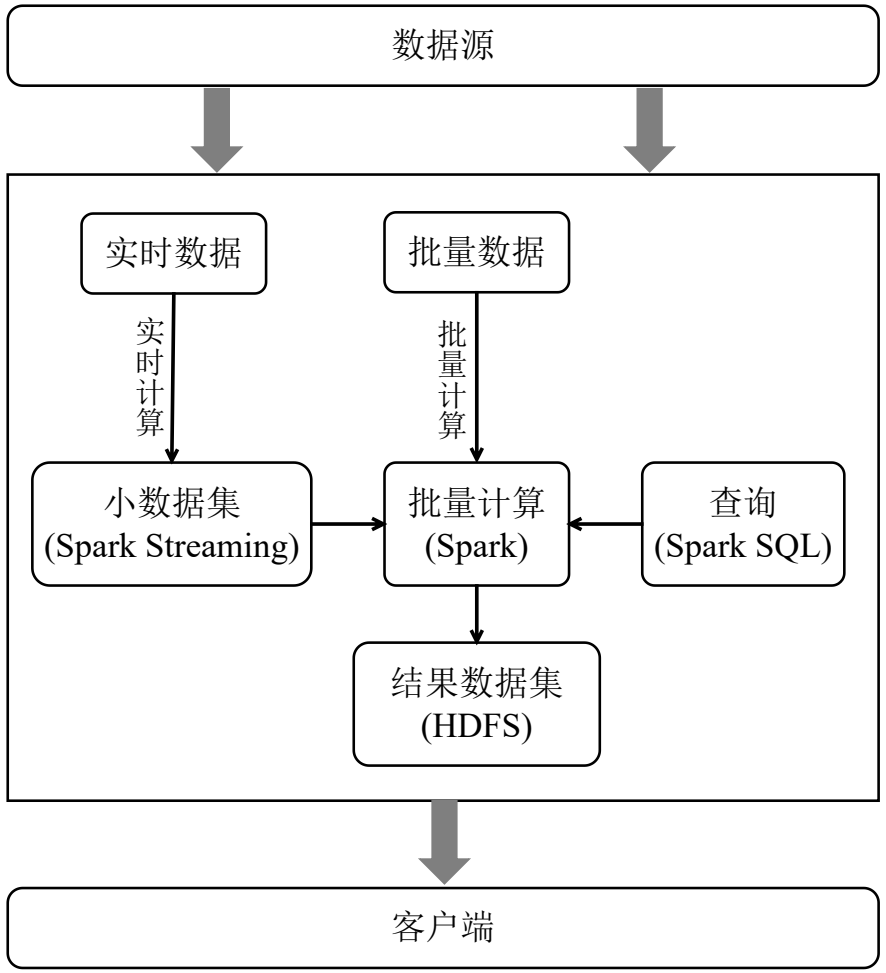

图7-12　用Spark架构满足批处理和流处理需求

采用Spark架构具有如下优点：

- 实现一键式安装和配置、线程级别的任务监控和告警；
- 降低硬件集群构建、软件维护、任务监控和应用开发的难度；
- 便于做成统一的硬件、计算平台资源池。