STORM 框架简介

Luying Li
Twitter
06/22/2014

内容

- 什么是 Storm?
- 使用案例
- 核心概念
- Word Count 例子
- Storm 与 Node.js

什么是 STORM?

- 分布式、高容错的实时计算系统
- 免费 开源
- 发布于2010年
- 由Backtype开发 (之后被Twitter收购）
- Github上最流行JVM项目之一

开源大数据解决方案

Solution	Developer	Type	Description
Storm	Twitter	Streaming	Twitter's new streaming big-data analytics solution
S4	Yahoo!	Streaming	Distributed stream computing platform from Yahoo!
Hadoop	Apache	Batch	First open source implementation of the MapReduce paradigm
Spark	UC Berkeley AMPLab	Batch	Recent analytics platform that supports in-memory data sets and resiliency
Disco	Nokia	Batch	Nokia's distributed MapReduce framework
HPCC	LexisNexis	Batch	HPC cluster for big data

From: http://www.ibm.com/developerworks/opensource/library/os-twitterstorm/index.html?ca=dat

STORM

- 确保数据被完全处理
- 简单易用的API
- 低延迟性
- 可扩展性
- 高容错性
- 多语言性
- 本地模式支持开发与测试
- 高性能: 每秒每个节点处理一百万个tuples
  - 处理器: 2x Intel E5645@2.4Ghz
  - 内存: 24 GB

使用案例

流数据处理

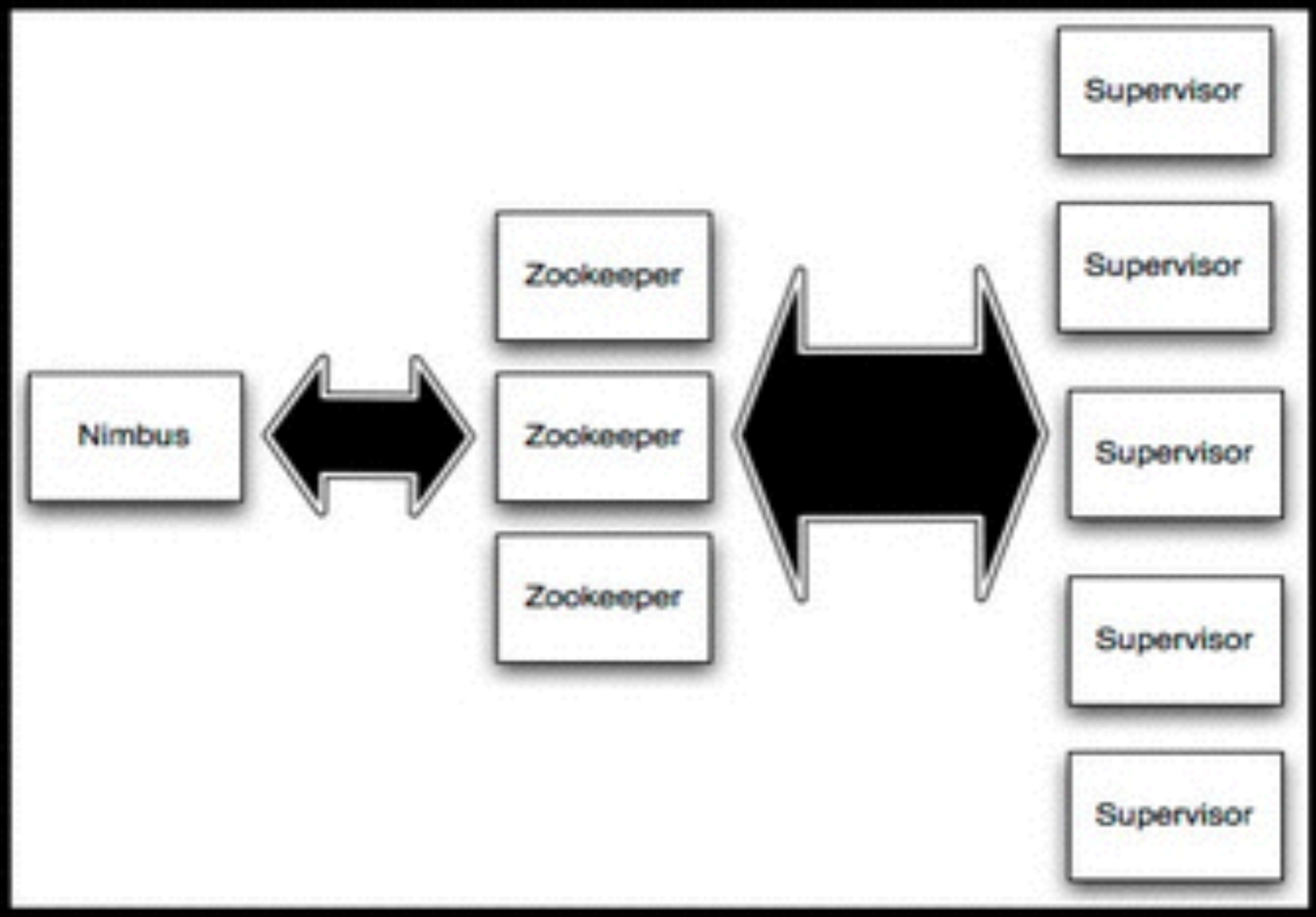
分布式RPC

持续计算

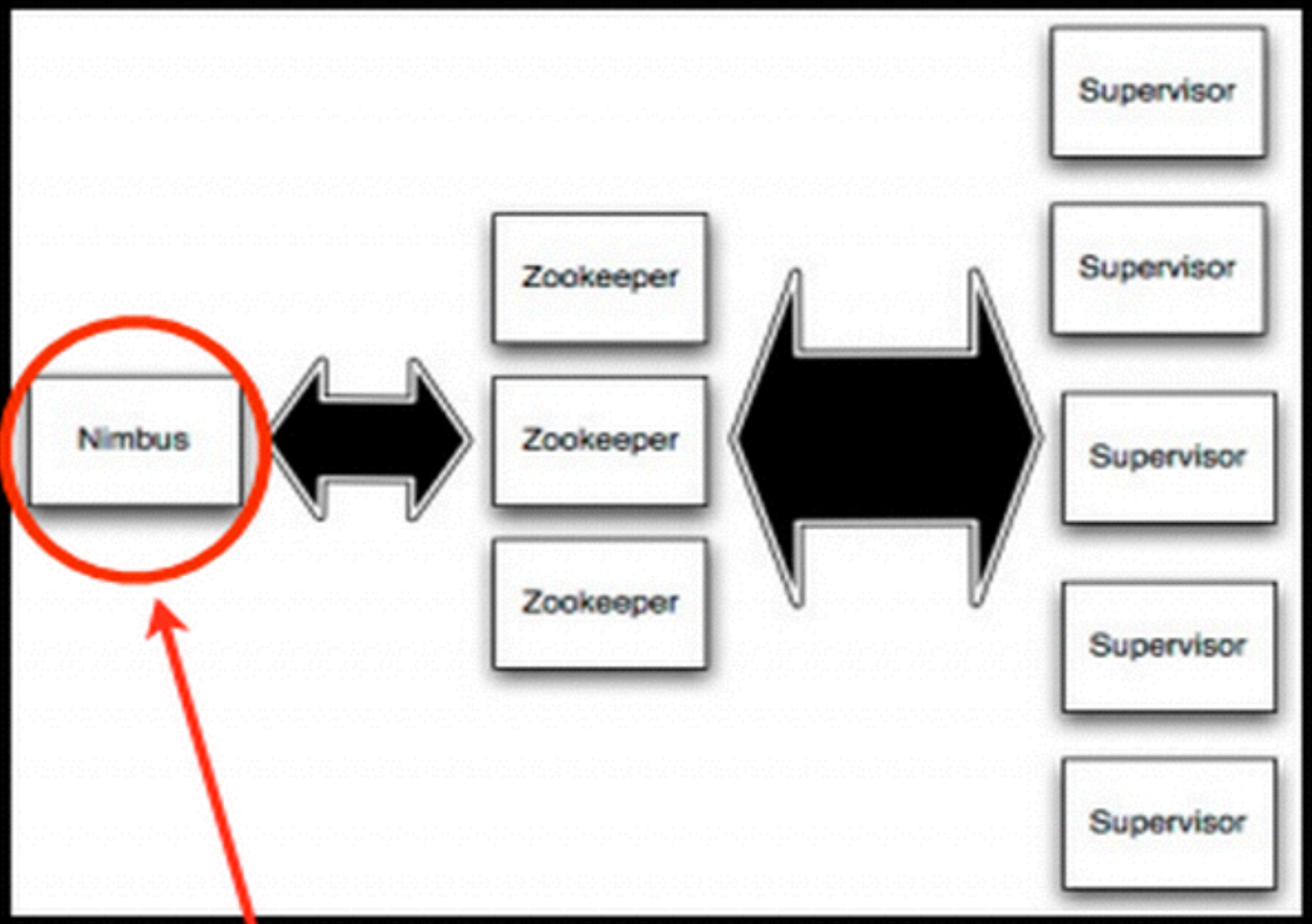
使用 Storm 的公司和项目

The Weather Channel
WebMD
Alibaba.com
FullContact
IDEXX LABORATORIES
Spotify
GROUPON
Flipboard
Cerner
Aeris COMMUNICATIONS
Baidu 百度
parc A Xerox Company
淘宝网 Taobao.com
rubicon PROJECT
rocketfuel
HolidayCheck.com
and many others

STORM 集群

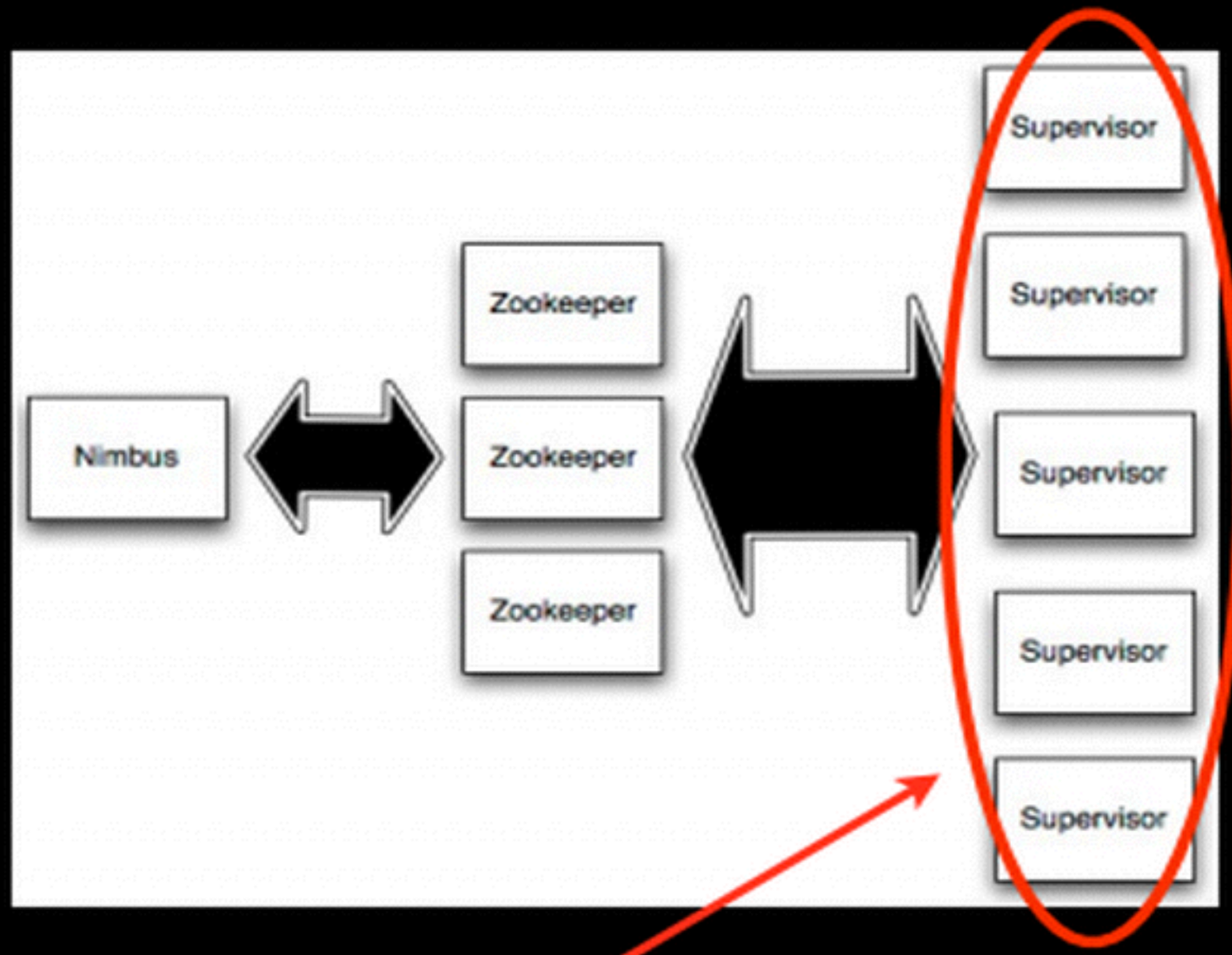

Supervisor
Zookeeper
Supervisor
Nimbus
Zookeeper
Supervisor
Zookeeper
Supervisor
Supervisor

STORM 集群



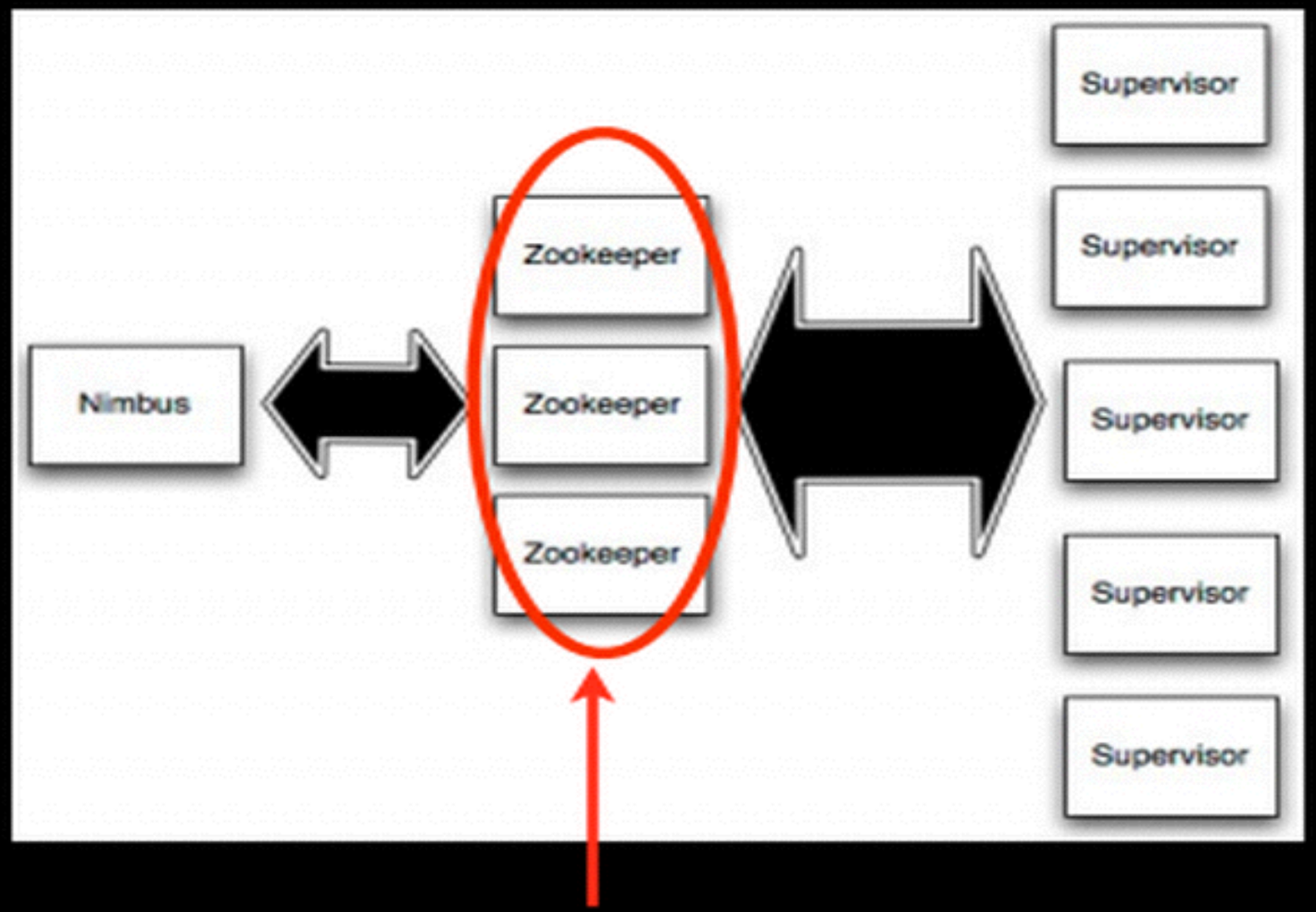
控制节点 (类似于 Hadoop 中的 JobTracker)

STORM 集群



Supervisor负责接收Nimbus分配的任务，管理Worker进程

STORM 集群



之间的协调工作由Zookeeper完成

STORM 与 HADOOP 对比

	Hadoop	Storm
System Characters	JobTracker	Nimbus
	TaskTracker	Supervisor
	Child	Worker
Application Name	Job	Topology
Components	Mapper/Reducer	Spout/Bolt

核心概念

- Tuple
- Stream
- Spout
- Bolt
- Topology
- Cluster
- Zookeeper
- Nimbus
- Supervisor
- Worker

计算元组 (TUPLE)

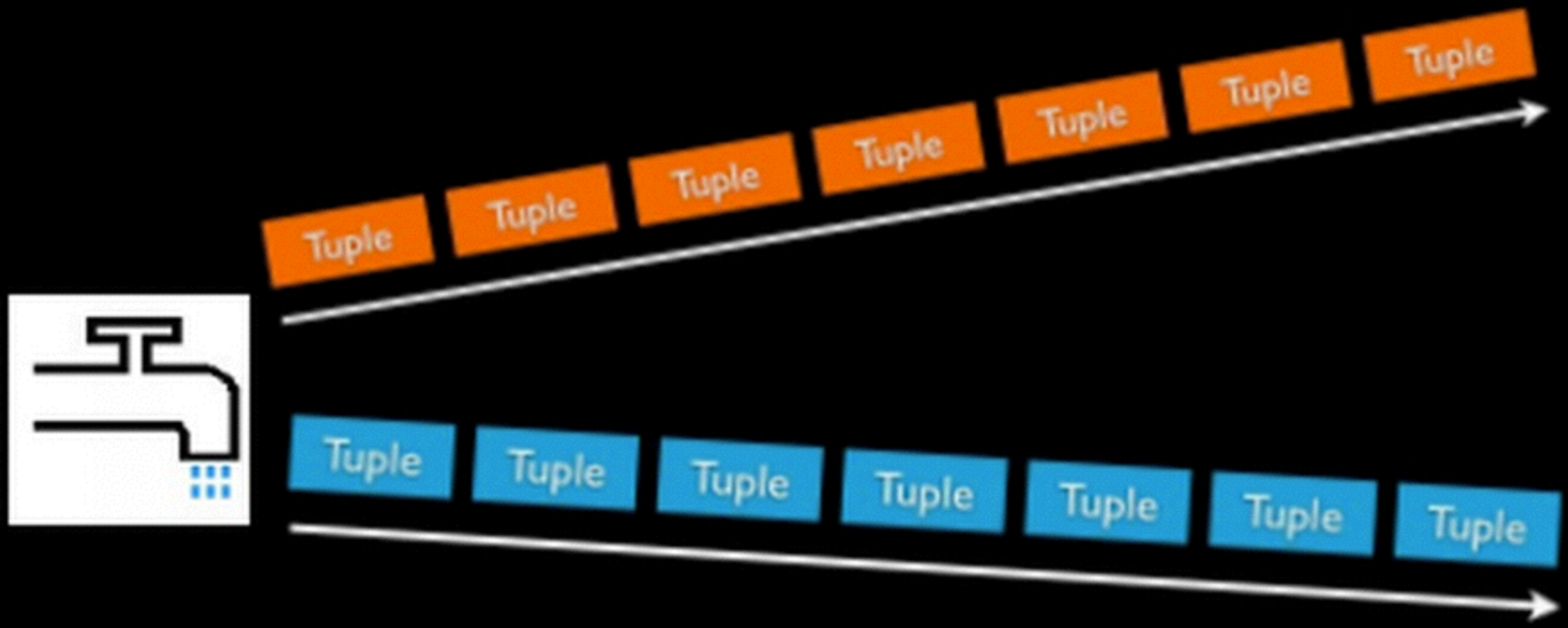
```
[ 198735697, "foobar", { "ip" : "10.0.0.1" } ]
```

storm支持所有的基本类型、字符串、字节数组或自定义类型作为tuple的值类

流 (STREAM)

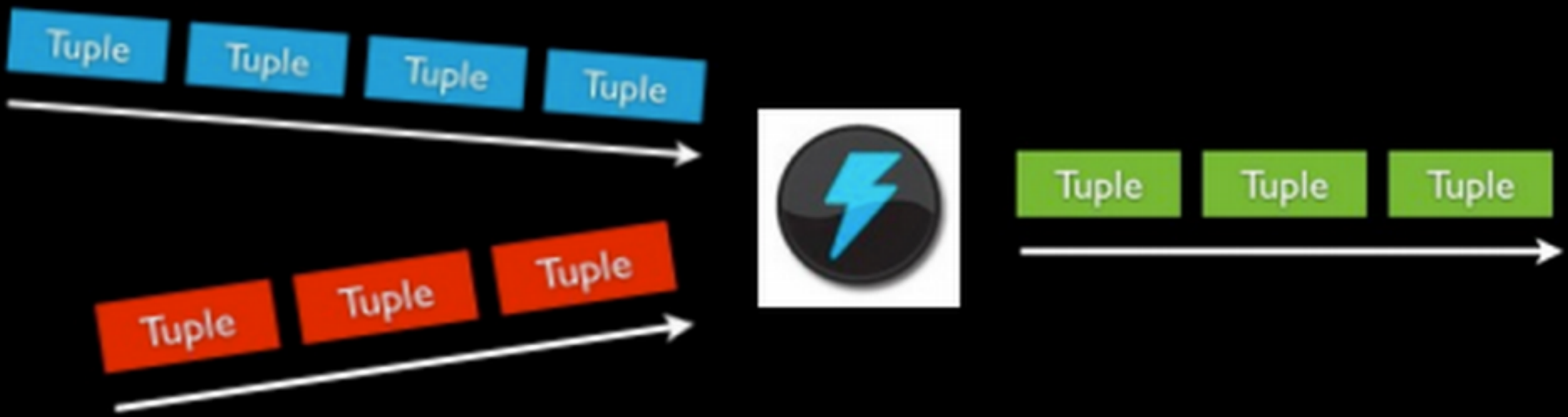
无边界连续的Tuples序列

消息源 (SPOUT)



流 (Stream) 的源头

消息处理者 (BOLT)

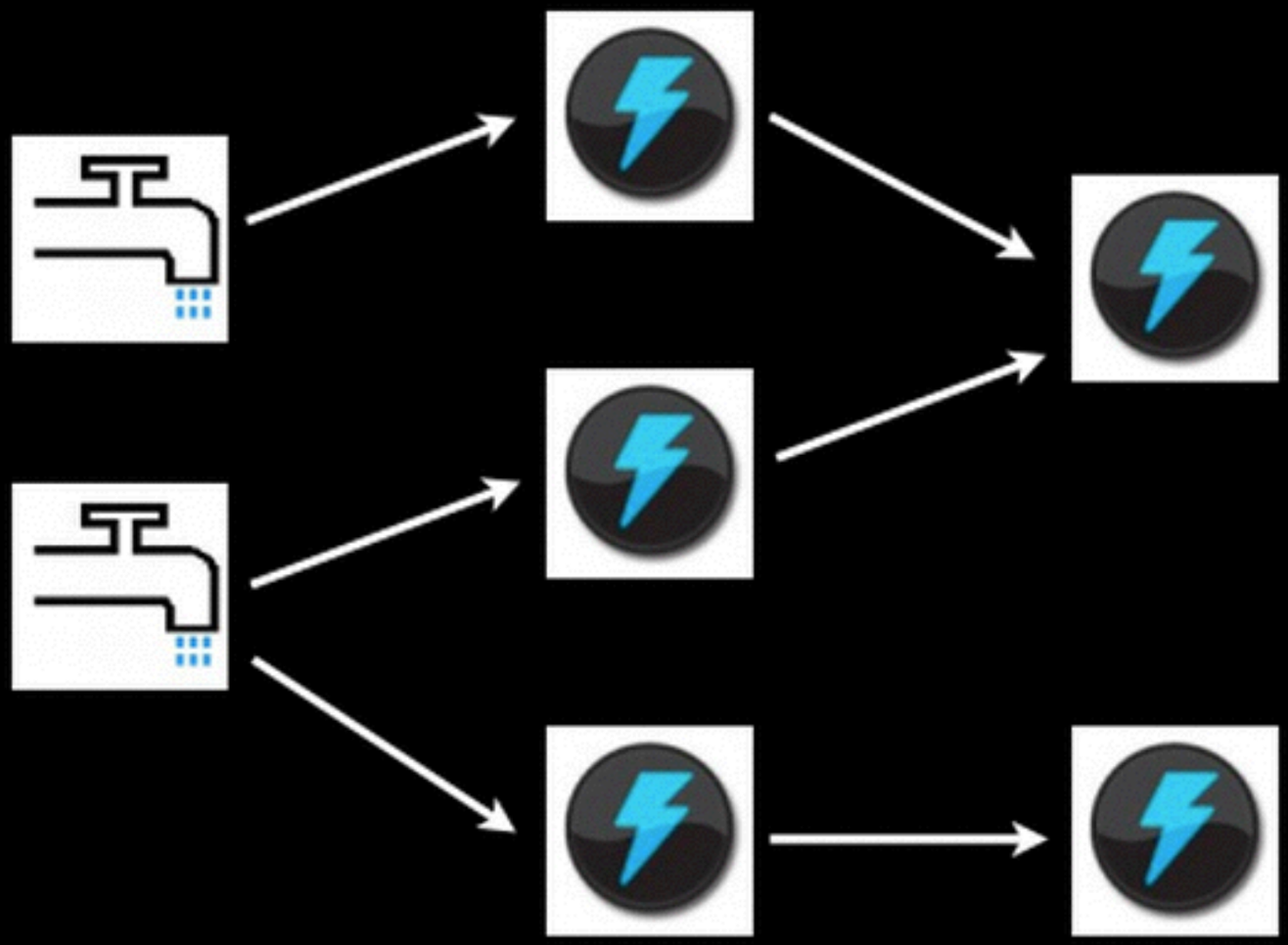


处理输入 Streams 并产生新的输出 Streams

消息处理者 (BOLT)

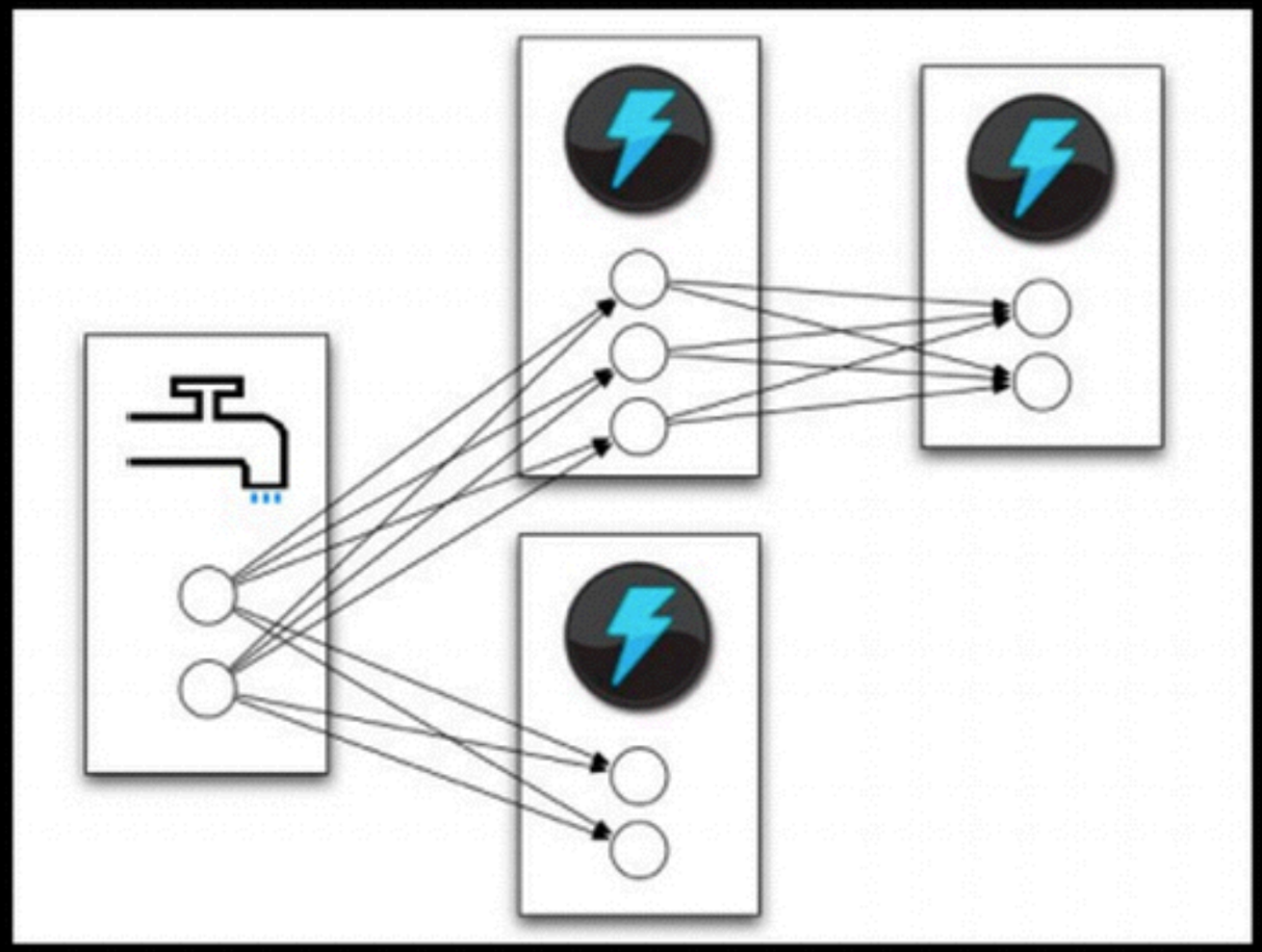
- 函数操作
- 过滤 (Filter)
- 聚合 (Aggregate)
- 合并 (Join)
- 访问数据库

计算拓扑 (TOPOLOGY)



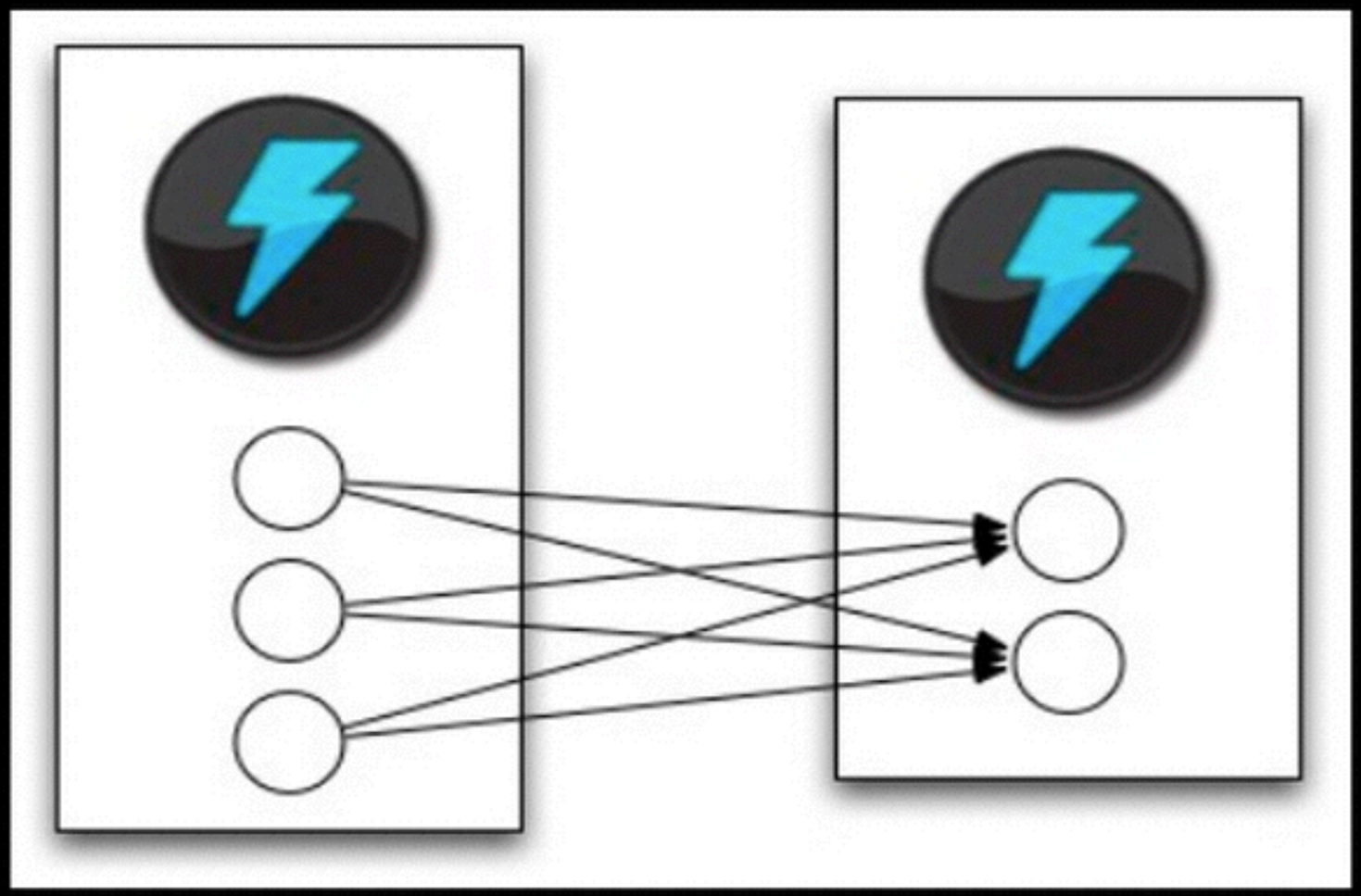
Spouts 和 Bolts 所组成的网络

任务 (TASK)



Spouts 和 Bolts 以很多 Task 的形式在 Topology 里面并行执行

流分组策略 (STREAM GROUPING)



Topology 如何在两个组件之间发送 Tuples

流分组策略 (STREAM GROUPING)

- Shuffle Grouping
- Fields Grouping
- All Grouping
- Global Grouping
- None Grouping
- Direct Grouping

WORD COUNT 例子

```
TopologyBuilder builder = new TopologyBuilder();

builder.setSpout("spout", new RandomSentenceSpout(), 5);

builder.setBolt("split", new SplitSentence(), 8)
.shuffleGrouping("spout");

builder.setBolt("count", new WordCount(), 12)
.fieldsGrouping("split", new Fields("word"));
```

RANDOM SENTENCE SPOUT

```
public class RandomSentenceSpout extends BaseRichSpout {
  SpoutOutputCollector _collector;
  Random _rand;

  @Override
  public void nextTuple() {
    Utils.sleep(100);
    String[] sentences = new String[]{
                  "the cow jumped over the moon",
                  ...
                  "i am at two with nature" };
    String sentence = sentences[_rand.nextInt(sentences.length)];
    _collector.emit(new Values(sentence));
  }
}
```

SPLIT SENTENCE BOLT

```
public static class SplitSentence extends BaseBasicBolt {

  @Override
  public void execute(Tuple tuple, BasicOutputCollector collector) {
    String sentence = tuple.getString(0);
    for(String word : sentence.split(" ")) {
      collector.emit(new Values(word));
    }
  }

  @Override
  public void declareOutputFields(OutputFieldsDeclarer declarer) {
    declarer.declare(new Fields("word"));
  }
}
```

WORD COUNT BOLT

```
public static class WordCount extends BaseBasicBolt {
  Map<String, Integer> counts = new HashMap<String, Integer>();

  @Override
  public void execute(Tuple tuple, BasicOutputCollector collector) {
    String word = tuple.getString(0);
    Integer count = counts.get(word);
    if (count == null)
      count = 0;
    count++;
    counts.put(word, count);
    collector.emit(new Values(word, count));
  }

  @Override
  public void declareOutputFields(OutputFieldsDeclarer declarer) {
    declarer.declare(new Fields("word", "count"));
  }
}
```

开发时的本地集群

```
Config conf = new Config();
conf.setDebug(true);

conf.setMaxTaskParallelism(3);
LocalCluster cluster = new LocalCluster();
cluster.submitTopology("word-count", conf,
    builder.createTopology());
Thread.sleep(10000);
cluster.shutdown();
```

多语言

- Storm 支持任何编程语言
- 生成 topologies
  - 易实现 - topologies 和 Nimbus 均是基于 Thrift
  - 实现 spouts 与 bolts 也称作 多语言组件 (multilang components) 或者 shelling
  - Spouts 和 Bolts 会被当做子进程来执行
  - 使用 JSON 消息通过标准输入输出来和这些子进程通讯
  - 该通讯协议是个只需要100行左右代码的库
  - 现已开发对应的 Ruby, Python 以及 Fancy 版本

STORM 与 NODE.JS

- Storm 支持多语言实现的 Node.js 版本
  - https://github.com/epokmedia/storm-node-multilang
    - 简易 类Storm 分布式应用实现的Node.js版本
  - https://github.com/ajlopez/SimpleStorm

STORM 相关资源

- http://storm.incubator.apache.org/
- https://github.com/nathanmarz/storm
- https://github.com/nathanmarz/storm-starter
- https://github.com/nathanmarz/storm-deploy

TWITTER 招聘

申请链接:
http://twitter.com/jobs

联系我: lli@twitter.com
关注我们:
@JoinTheFlock
@TwitterU