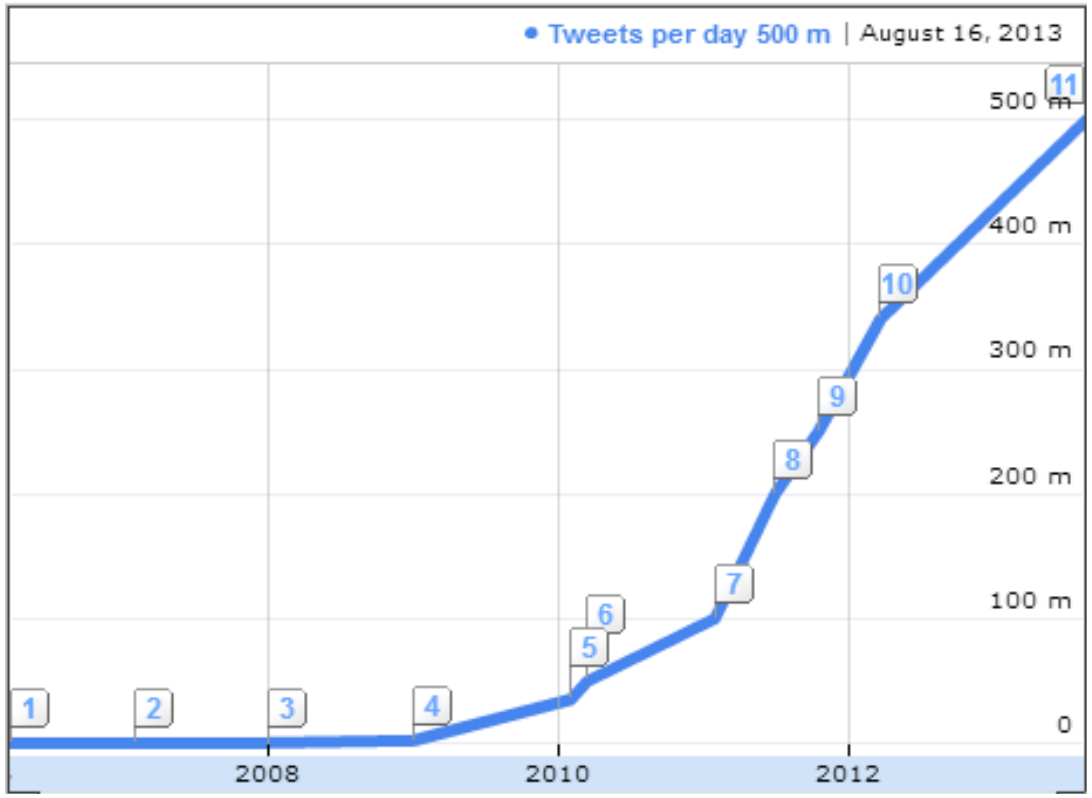
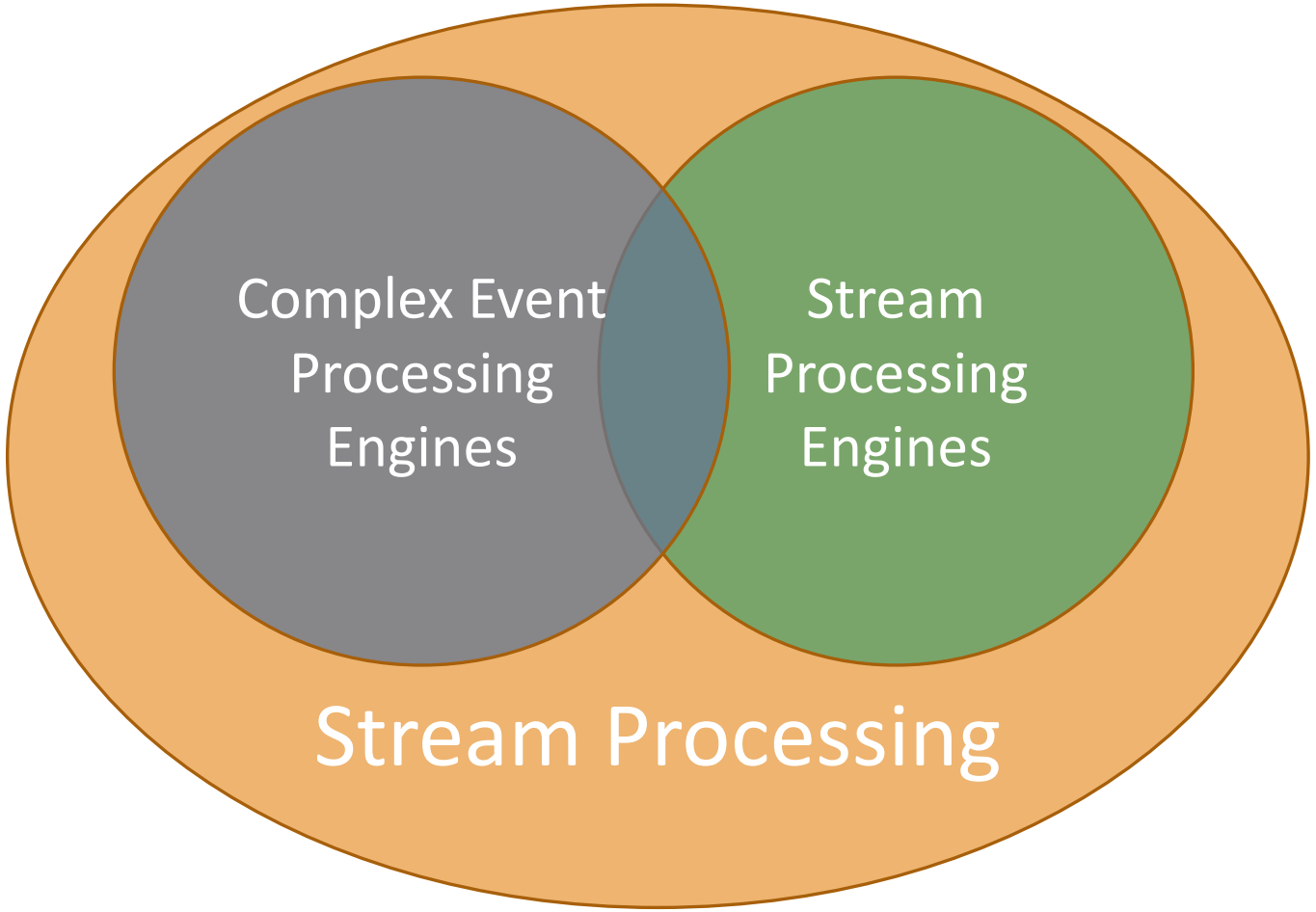
Stream Processing

DAVID OSTROVSKY | PROOFPOINT

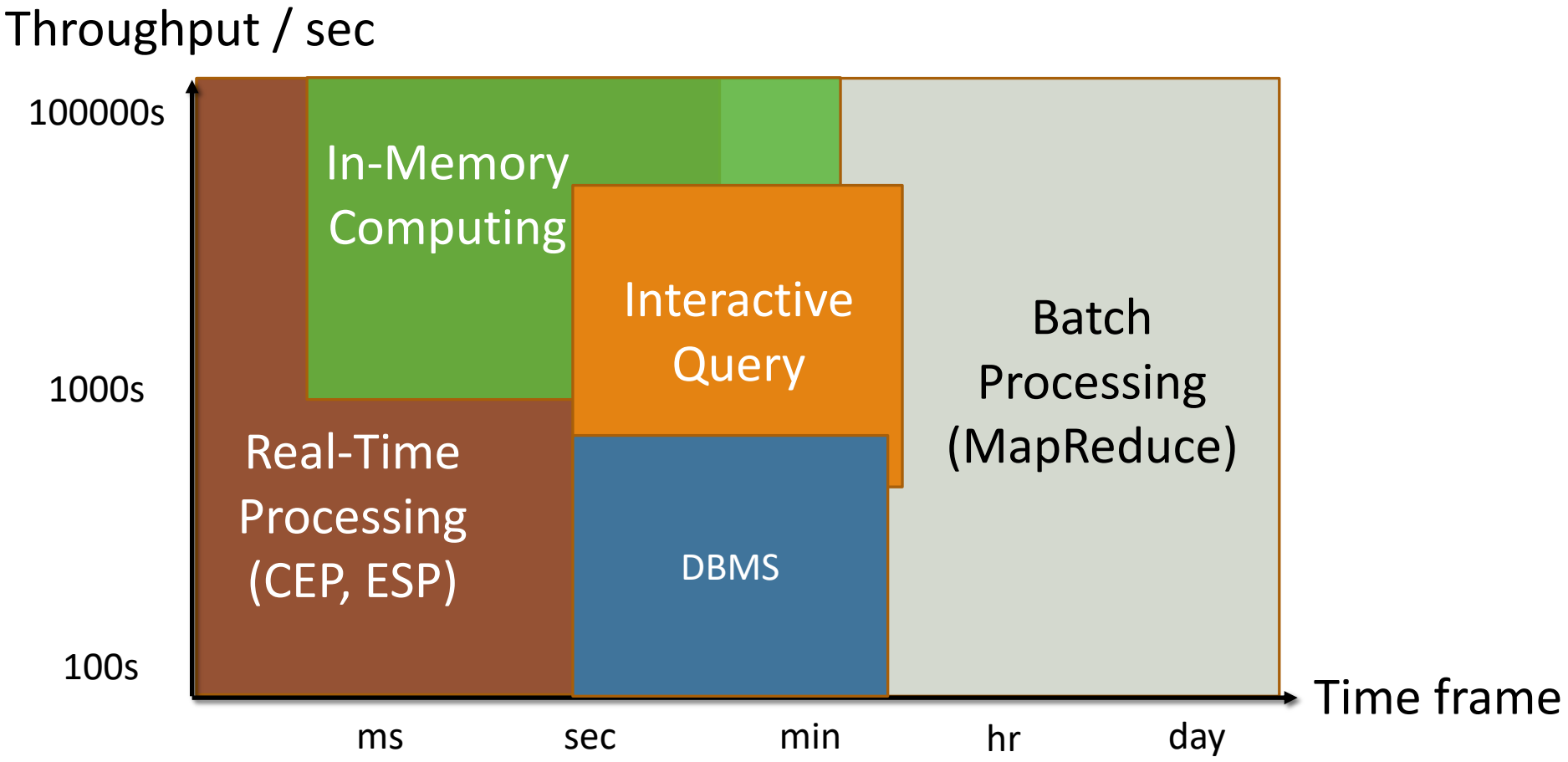
Why Streaming?



Streaming Data



Types of Data Processing

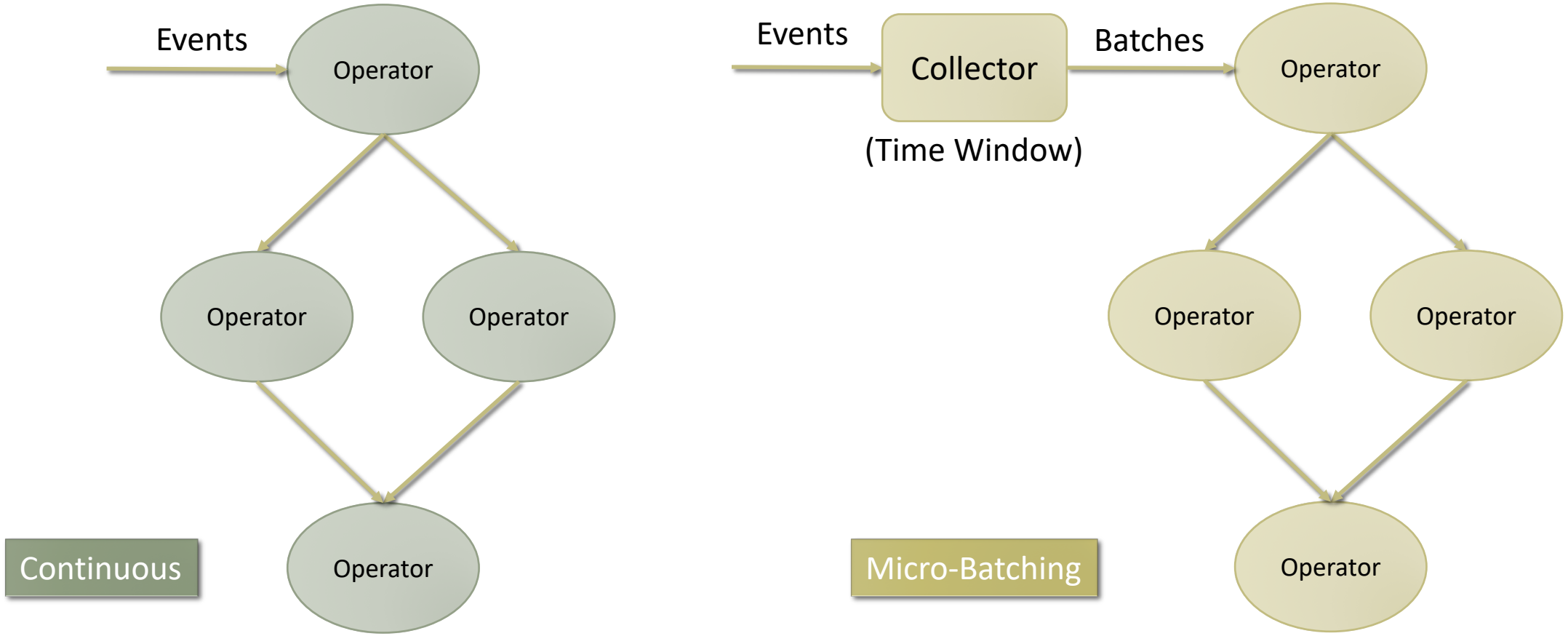


All Apache, all the Time

No Love for Microsoft?

Orleans

Processing Model



Programming Model

Continuous	Micro-Batch	Micro-Batch	Continuous	Continuous*

* Has a batch abstraction on top of streaming

API and Expressiveness

Compositional

```
public class PrinterBolt extends BaseBasicBolt
{

  public void execute(Tuple tuple, ...) {
    System.out.println(tuple);
  }
}
```

```
topology.setBolt("print", new PrinterBolt())
        .shuffleGrouping("twitter");
```

Declarative

```
val ssc = new StreamingContext(conf, Seconds(1))
ssc.socketTextStream("localhost", 9999)
    .flatMap(_.split(" "))
    .map(word => (word, 1))
    .reduceByKey(_ + _)
    .print()
```

API and Expressiveness

Compositional	Compositional	Declarative	Both	Declarative
JVM, Python, Ruby, JS, Perl	JVM	JVM, Python	JVM	JVM, Python*

* Only for the DataSet API (batch)

Storm + Trident

Topology:

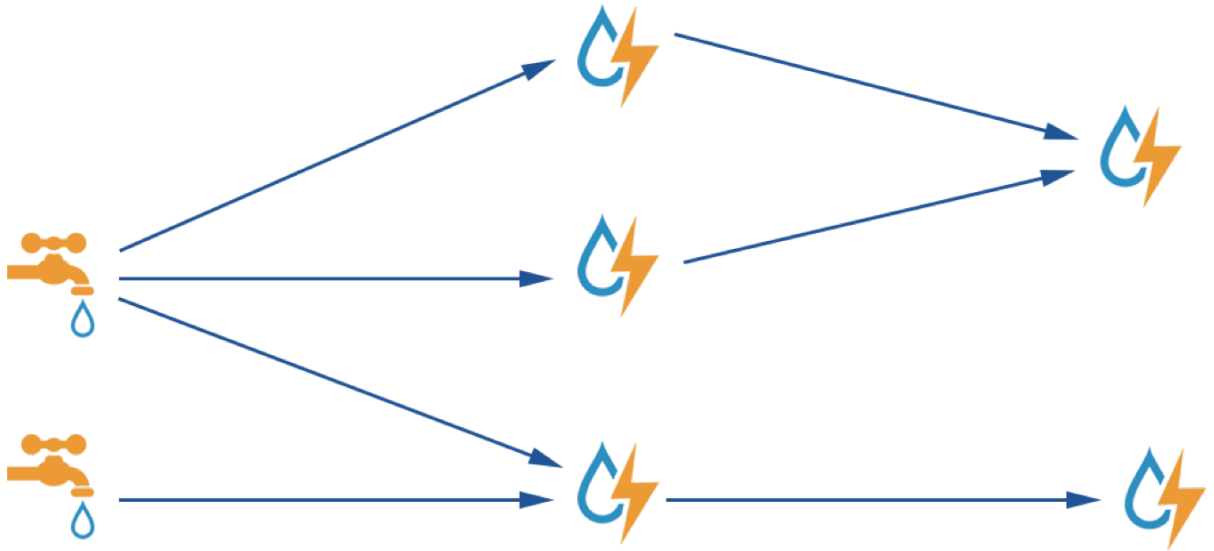
- Spouts
- Bolts

Stream Groupings:

- Shuffle
- Fields
- All
- …

Nimbus (Master)

- Workers



Spark Streaming

Resilient Distributed Datasets (RDD)

DStreams – sequences of RDDs

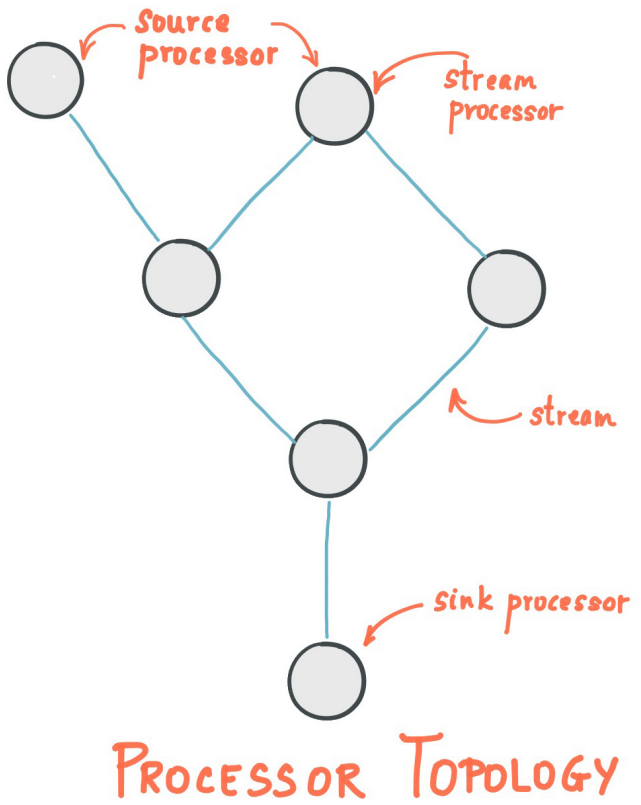
Kafka Streams

Uses Kafka for streaming

- Topics (streams)
  - Partitioned across Brokers
- Producers
- Consumers

Applications use the Streams API

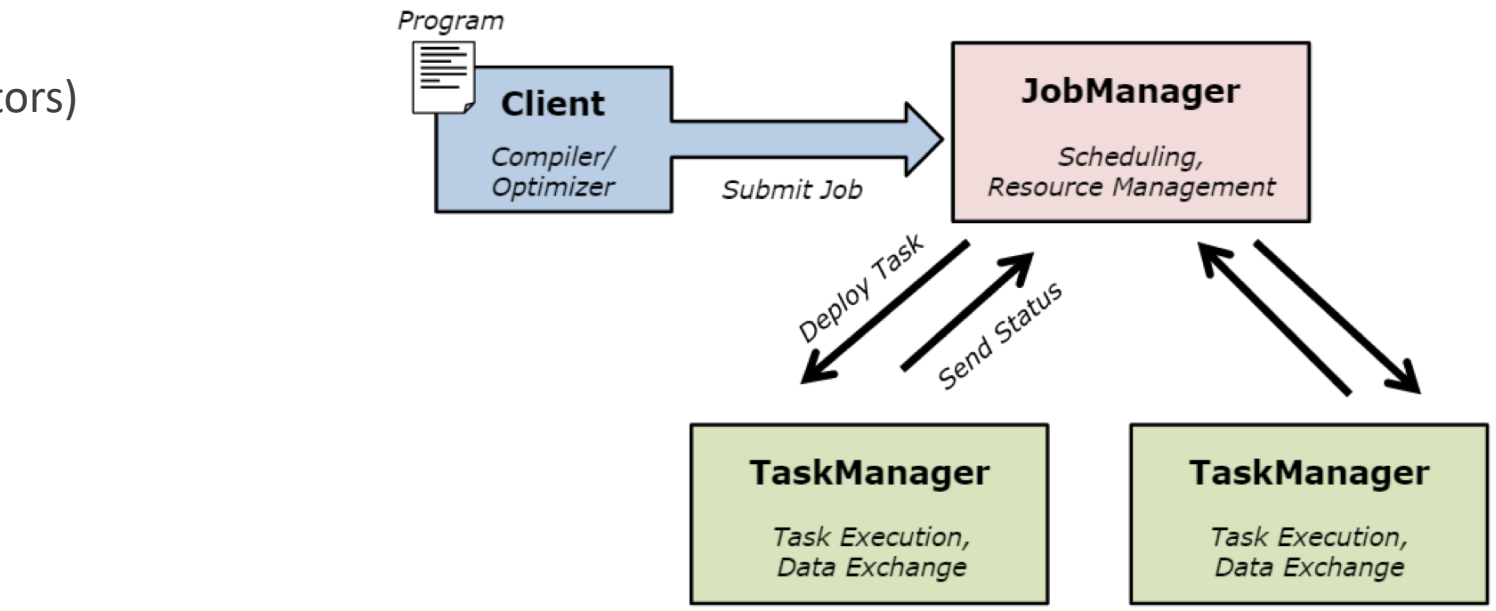
- KStream
- KTable



Flink

Dataflows

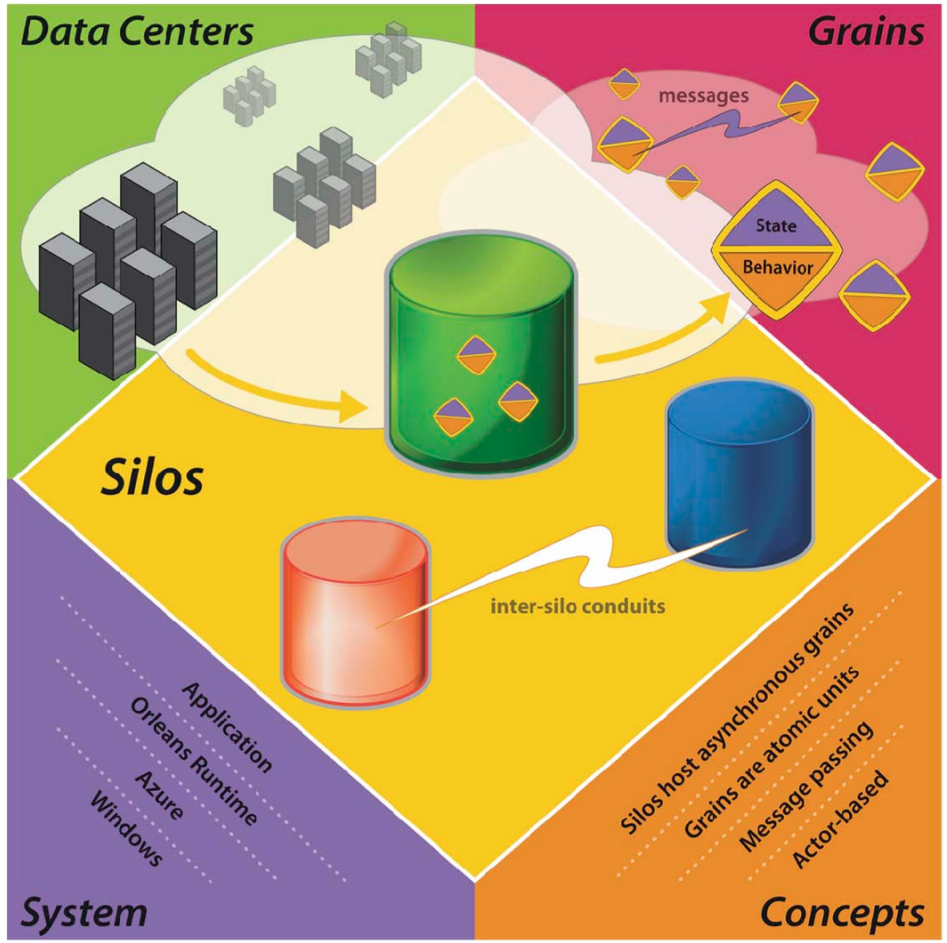
- Streams
  - Source(s)
  - Sink(s)
- Transformations (operators)



Orleans

Virtual Actor System in .NET

- Grains (operators)
- Silos (containers)
- Streams



Message Delivery Guarantees

Source	Sockets Twitter Streaming API Any non-repeatable	Files Simple Queues Any forward-only	Kafka, RabbitMQ Collections Stateful
Sink		Data Stores Sockets Files	HDFS rolling sink

Highest Possible Guarantee

At least once*	Exactly once*	Exactly once**	Exactly once*	Exactly once*

* Doesn't apply to side-effects

** Only at the batch level

Reliability and Fault Tolerance

ACK per tuple	RDD checkpoints	Partition offset checkpoints	Barrier checkpoints

State Management

Manual	**Dedicated state providers (memory, external)**	**RDD with per-key state**	**Local K/V store + changelog in Kafka**	**Stored with snapshots, configurable backends**

Performance

Latency	Low	Medium	Medium-High*	Low	Low**
Throughput	Medium	Medium	High	High	High

* Depends on batching

** For streaming, not micro-batching

Extended Ecosystem

SAMOA (ML)	**Trident-ML**	**Spark SQL, MLlib GraphX**	**N/A**	**CEP Gelly* FlinkML* Table API (SQL)***

* DataSet API (batch)

Production and Maturity

	Mature, many users, 260 contributors	**Relatively mature, many users 1092 contributors***	**New, built on mature components, enterprise backing, 306 contributors**	**New, high momentum, fewer users, 321 contributors**

* Spark, not just spark streaming

** Contributor numbers as of 12/6/2017


R. COFFEE
MR. RADAR
Everybody got that?