Criteria	Value	Description
Data structure	Unstructured	Data that are mainly available as text or images with only limited structure and available meta-data.
	Semi-structured	Data that are available in e.g. self-describing formats such as XML.
	Structured	Data that are available as relational data or graph data.
Data Reachability	Real-time	Timeliness is paramount to the use and the implementation should actively seek to minimize the delays that occur between the source and destination systems.
	Near-real-time	A minimal delay between data being generated and acquisition is acceptable, the implementation can strike a balance between batch and real-time communication. The platform can rely on techniques such as micro-batching or periodic polling which introduce delays but simplify the architecture and avoid some of the cost and complexity of true real-time processing.
	Batch (minutes / hours / days)	Data is available and updated sometime after it has been updated in the sourcing layer. The platform will batch data or operations together to gain efficiencies in transport and processing. This can be particularly important when dealing with large volumes.

Batch VS NRT

Zillions of data once

Gigabyte data per second

Batch VS NRT

Completed in longer time

7*24 Non-Stop availability

Batch VS NRT

Make future Plan & Decisions

Immediate Decision & Action

Analytics

- Is when you have plenty of time for analysis
- Is when you explore patterns and models in historic data
- Is when you plan and forecast the future instead of (re)acting immediately

Technology

- Query previously stored data
- Store the data in HDFS/Cosmos and Process with Hadoop/Scope
- Generally rely on disks/storage

NRT

Analytics

- Is when you don't have time
- Is when you analyze data as it comes
- Is when you already have a fixed model, and data flying in fits it 100%
- Is when you (re)act immediately

Technology

- Process events in streaming
- Store only for checkpoint reasons
- Mostly in memory and hit disk rarely

NRT Scenarios

Applications	Fraud/Bot detection Log Processing
Web	Site Analytics & Monitoring Recommendation & Personalization
Mobile Apps	Network metrics analysis Geolocation based ads

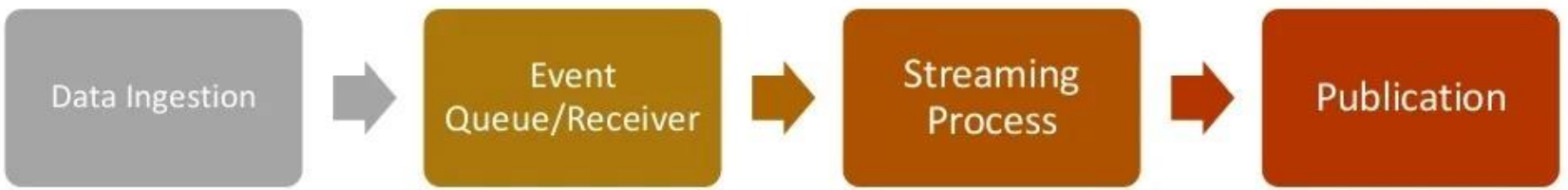
NRT Computing Model

- **Event Processing/Mini Batch Processing**
  - Cooking, enrichment, and transfer
- **Time Window based Counter/Aggregation**
  - Real Time Metrics/Monitoring
  - Active User, Usage statistics
- **Buffer data for processing**
  - Multi-Chunk Streaming
  - Sessionization
- **State based updating**
  - User info tracking/updating
  - New user detection
- **Interact with Historic/external data**
  - Fraud/Bot Detection, ML Prediction
  - Streaming SQL

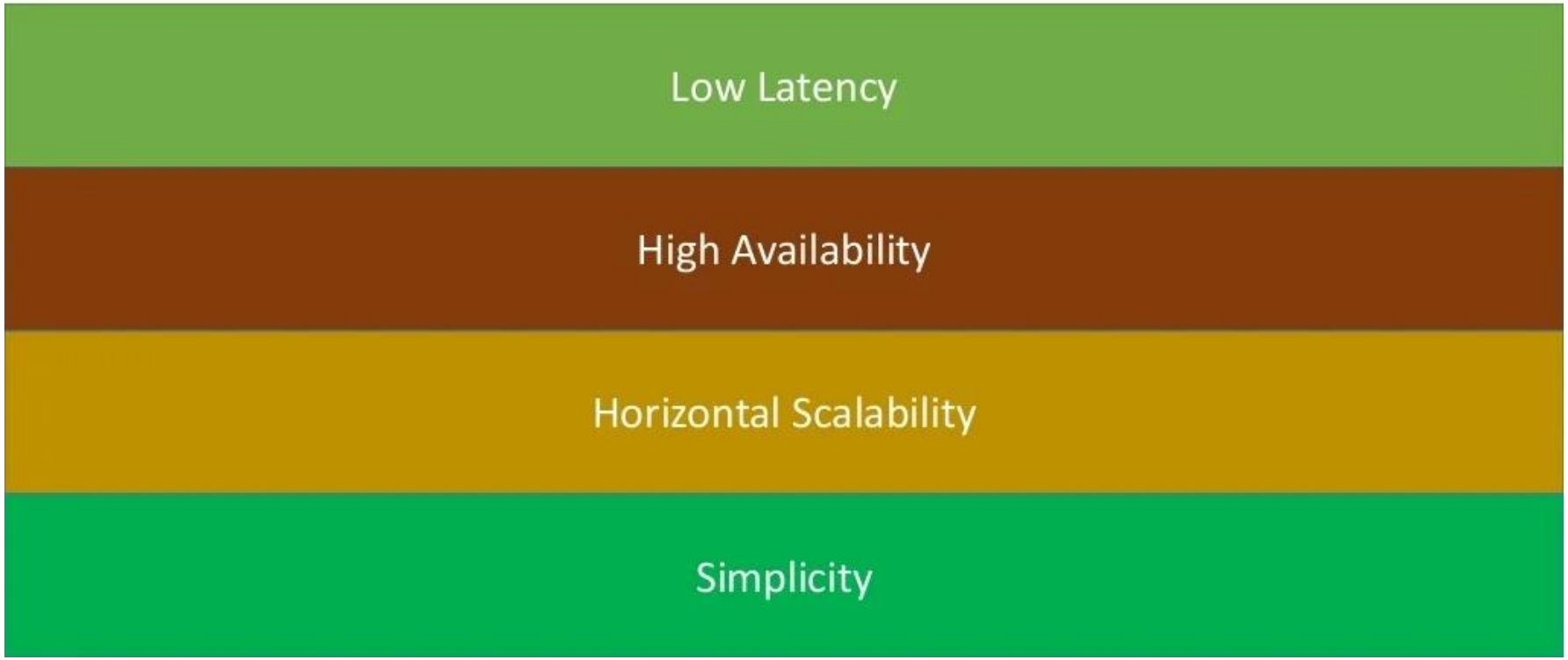
Architecture & Technology

Storm, Spark Streaming

NRT Architecture



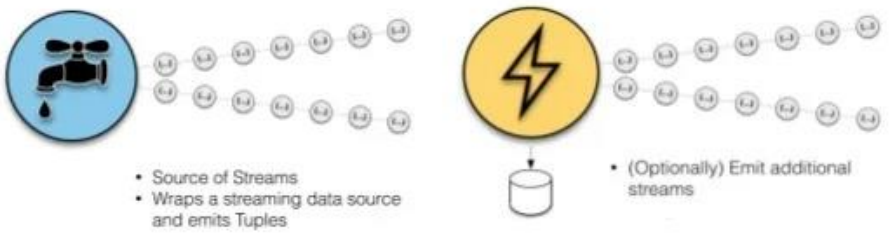
Architecture Goal

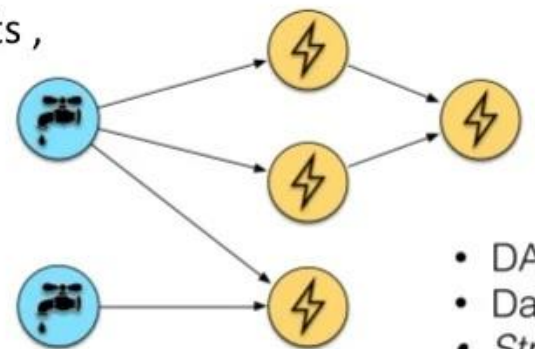


Apache Storm in a Slide

STORM

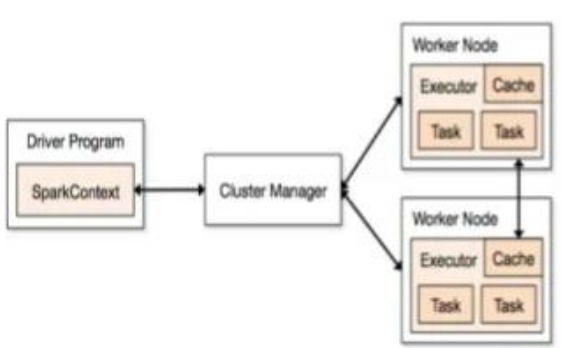
- Distributed Streaming Computation system.
  - Originated at BackType and then Twitter
  - Fast,
  - Scalable
  - Fault Tolerant
- STORM is a DAG (Topology) of Spouts , Bolts
  - Radom shuffling
  - Group shuffling



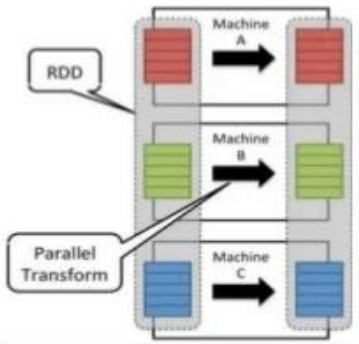


- DAG of Spouts and Bolts
- Data Flow Representation
- *Streaming Computation*

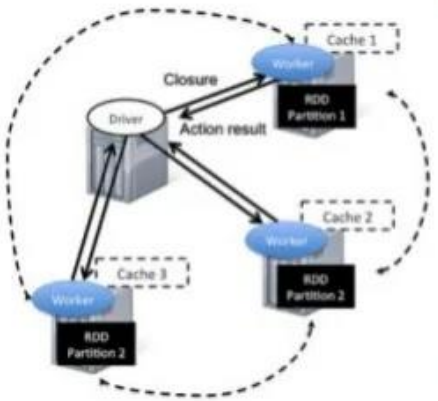
Apache Spark in a Slide



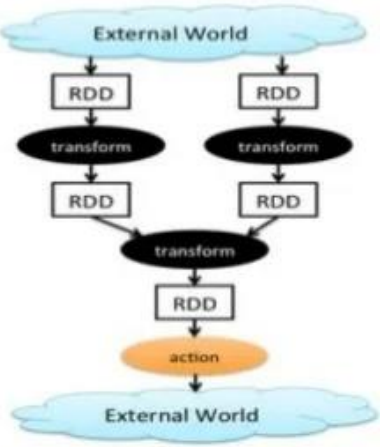
Driver: Launches various parallel operations on a cluster.
Worker: Execute the operations on each Node



RDD: Logically centralized entity but physically partitioned across multiple machines.



RDD can optionally be cached in memory and hence provide fast access. (Disk is also supported)

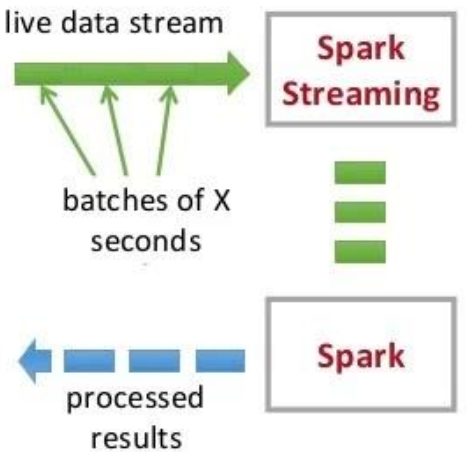


Application logic are expressed in terms of a sequence of Transformation and Action.

Spark Streaming in a Slide

Run a streaming computation as a series of very small, deterministic batch jobs

- Chop up the live stream into batches of X seconds (Batch Size as low as 0.5 Second, Latency is about 1 second)
- Spark treats each batch of data as RDDs and processes them using RDD operations
- Finally, the processed results of the RDD operations are returned in batches



NRT Solution Comparison

	Storm	Spark Streaming
Processing Model	Record-at-a-time	Mini batches
Latency	Sub-Second	Few Seconds
Fault tolerance – every record processed	At least once (May be duplicates)	Exactly once
Resource Manager	Mesos	Yarn, Mesos
Persist Storage	-	-
API Language	Java (And others)	Scala, Java, Python
Computation/Transform	Bolts	Map, GroupBy, Join, Filter, Reduce, …
Stateful Operation	No	UpdateStateByKey
Window Operation	No	Window Operation
Output	Bolts	HDFS, File, Console, ForeachRDD(Socket)