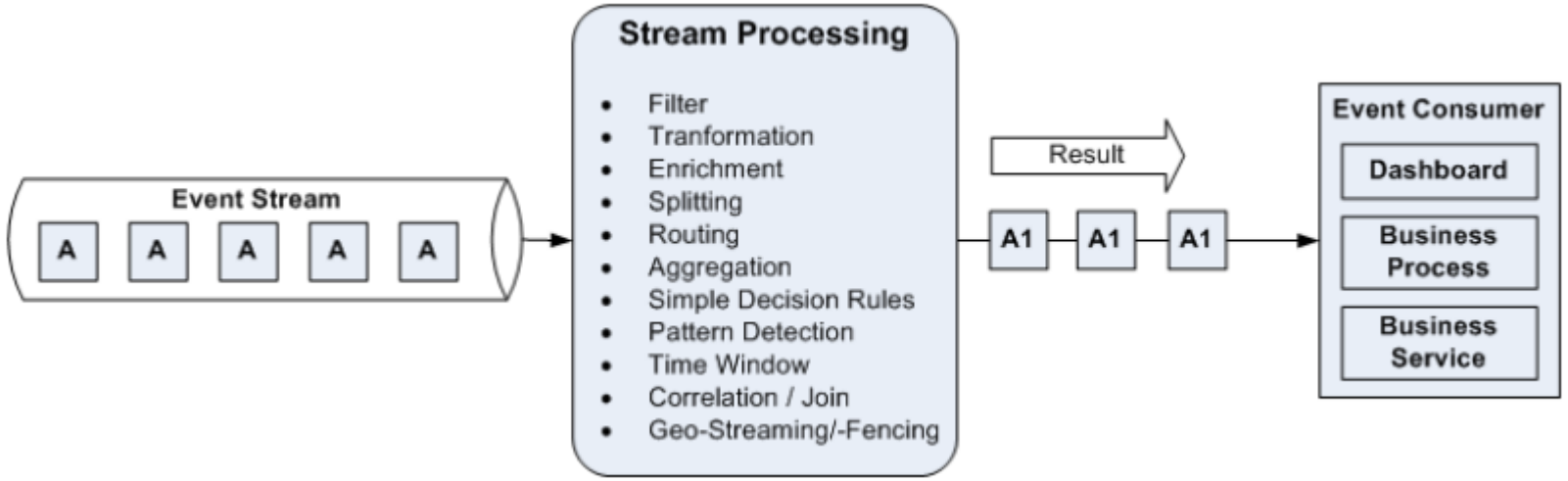
Storm and Trident

ixAT Solutions – ixatsolutions@gmail.com

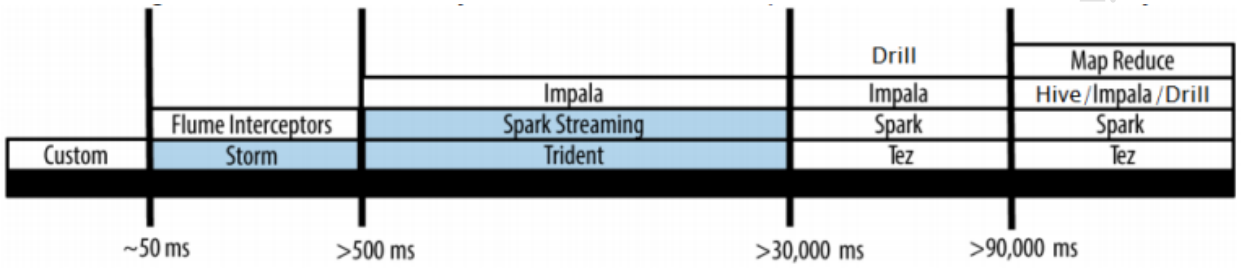
Stream Processing

- Infrastructure for continuous data processing.
- Computational model can be as general as MapReduce but with the ability to produce low-latency results
- Data collected continuously is naturally processed continuously aka. Event Processing / Complex Event Processing (CEP)



Processsing Models

- Batch Processing
  - Familiar concept of processing data en masse
  - Generally incurs a high-latency
- (Event-) Stream Processing
  - A one-at-a-time processing model
  - A datum is processed as it arrives
  - Sub-second latency
  - Difficult to process state data efficiently
- Micro-Batching
  - A special case of batch processing with very small batch sizes (tiny)
  - A nice mix between batching and streaming
  - At cost of latency
  - Gives stateful computation, making windowing an easy task

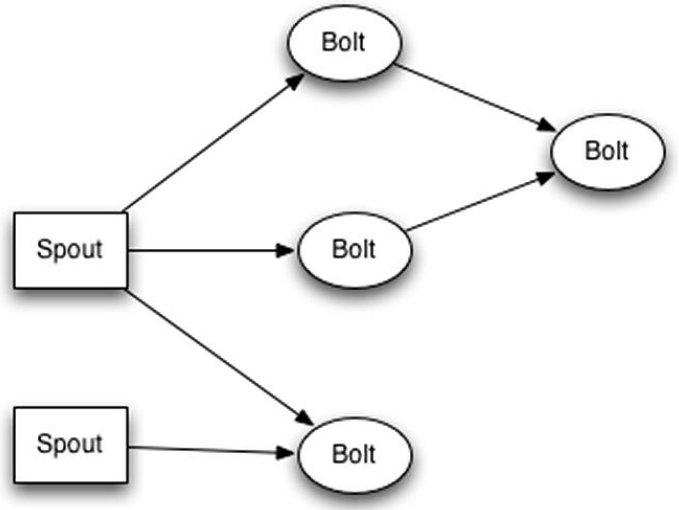


Choices

- Latency
  - Is performance of streaming application paramount
- Development Cost
  - Is it desired to have similar code bases for batch and stream processing =>lambda architecture
- Message Delivery Guarantees
  - Is there high importance on processing every single record, or is some normal amount of data loss acceptable
- Process Fault Tolerance
  - Is high-availability of primary concern

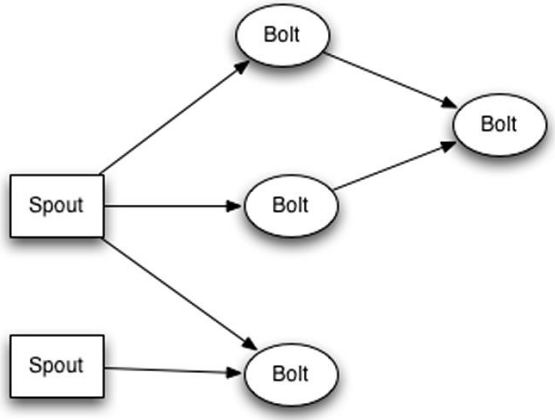
Storm Overview

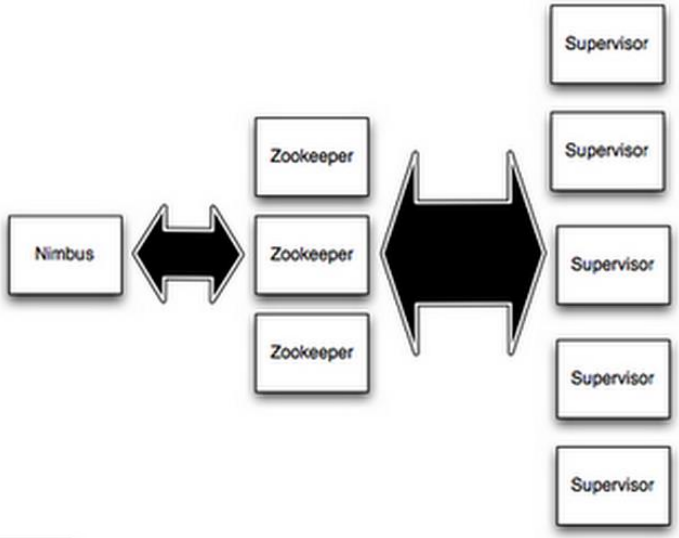
- DAG Processing of never ending streams of data
  - Open Sourced: https://github.com/nathanmarz/storm/wiki
  - Used at Twitter plus > 24 other companies
  - Reliable - At Least Once semantics
  - Think MapReduce for data streams
  - Java / Clojure based
  - Bolts in Java and 'Shell Bolts'
  - Not a queue, but usually reads from a queue.
- Related:
  - S4, CEP, InfoStreams, HStreaming,…
- Compromises
  - Static topologies & cluster sizing – Avoid messy dynamic rebalancing
  - Nimbus – SPOF
- Strong Community Support, commercial support from InfoChimps

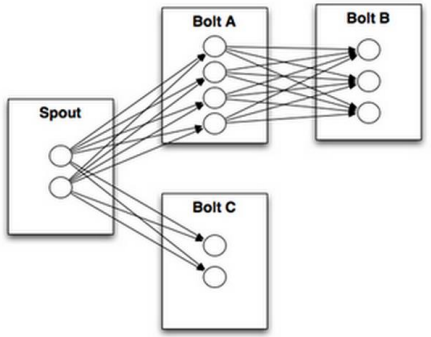


Storm Concepts

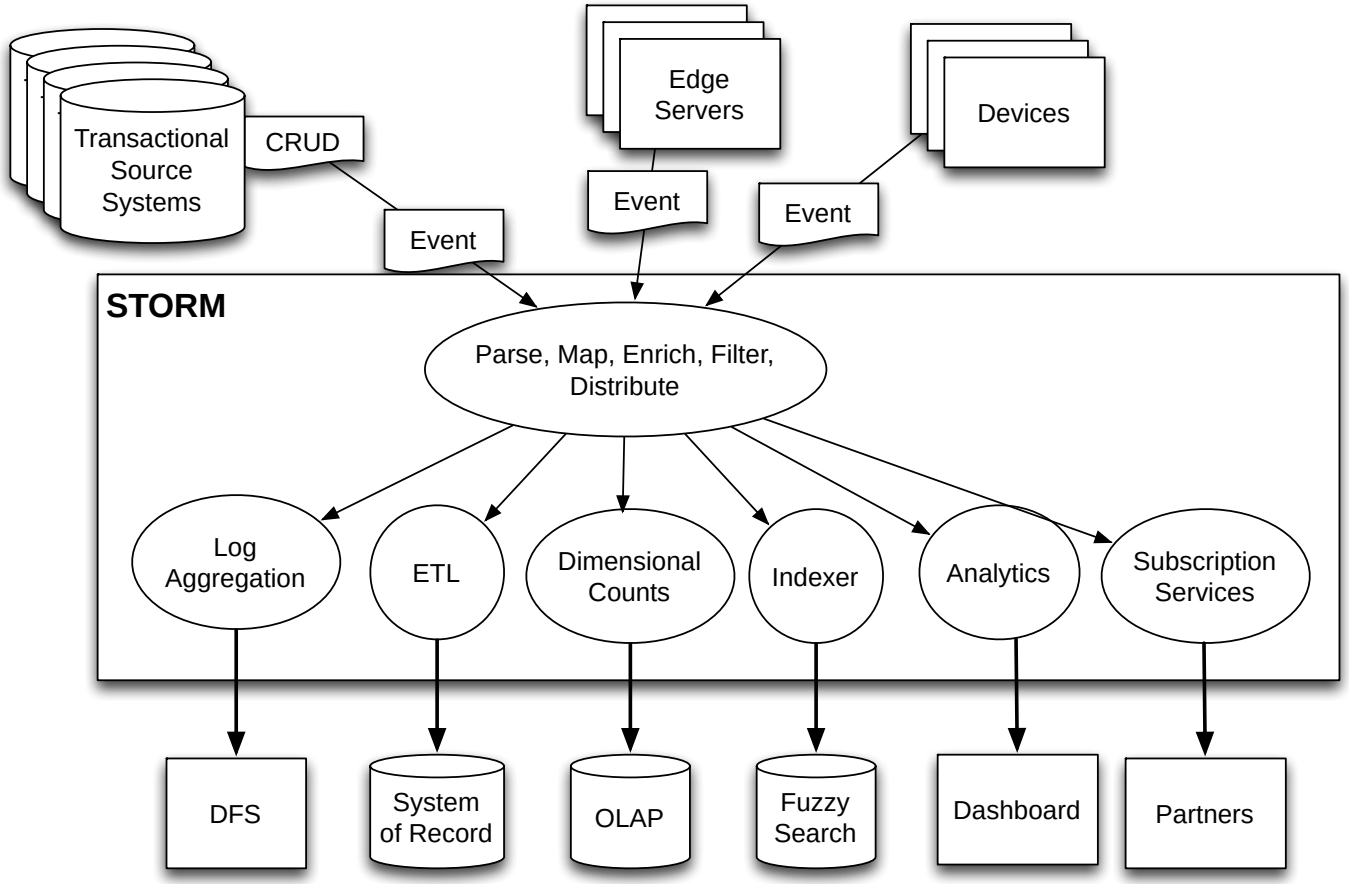
- Cluster
  - Supervisor
  - Worker
- Topology
- Streams
- Spout
- Bolt
- Tuple
- Stream







Storm & Big Data Patterns



What is Trident exactly?

- Trident is
  - Abstraction on top of Storm Infrastructure
  - Topology
    - Divided into streams
    - Optimized
  - DRPC compatible
  - Added to Storm 0.8.0 – Aug 2012
- Simplifies building topologies
- Core data model is the stream
  - Processed as a series of batches (micro-batches)
  - Stream is partitioned among nodes in cluster
- 5 kinds of operations in Trident
  - Operations that apply locally to each partition and cause no network transfer
  - Repartitioning operations that don't change the contents
  - Aggregation operations that do network transfer
  - Operations on grouped streams
  - Merges and Joins

Comparision

	Core Storm	Storm Trident	Spark Streaming
Community	> 100 contributors	> 100 contributors	> 280 contributors
Adoption	***	*	*
Language Options	Java, Clojure, Scala, Python, Ruby, ...	Java, Clojure, Scala	Java, Scala Python (coming)
Processing Models	Event-Streaming	Micro-Batching	Micro-Batching Batch (Spark Core)
Processing DSL	No	Yes	Yes
Stateful Ops	No	Yes	Yes
Distributed RPC	Yes	Yes	No
Delivery Guarantees	At most once / At least once	Exactly Once	Exactly Once
Latency	sub-second	seconds	seconds
Platform	Storm Cluster, YARN	Storm Cluster, YARN	YARN, Mesos Standalone, DataStax EE