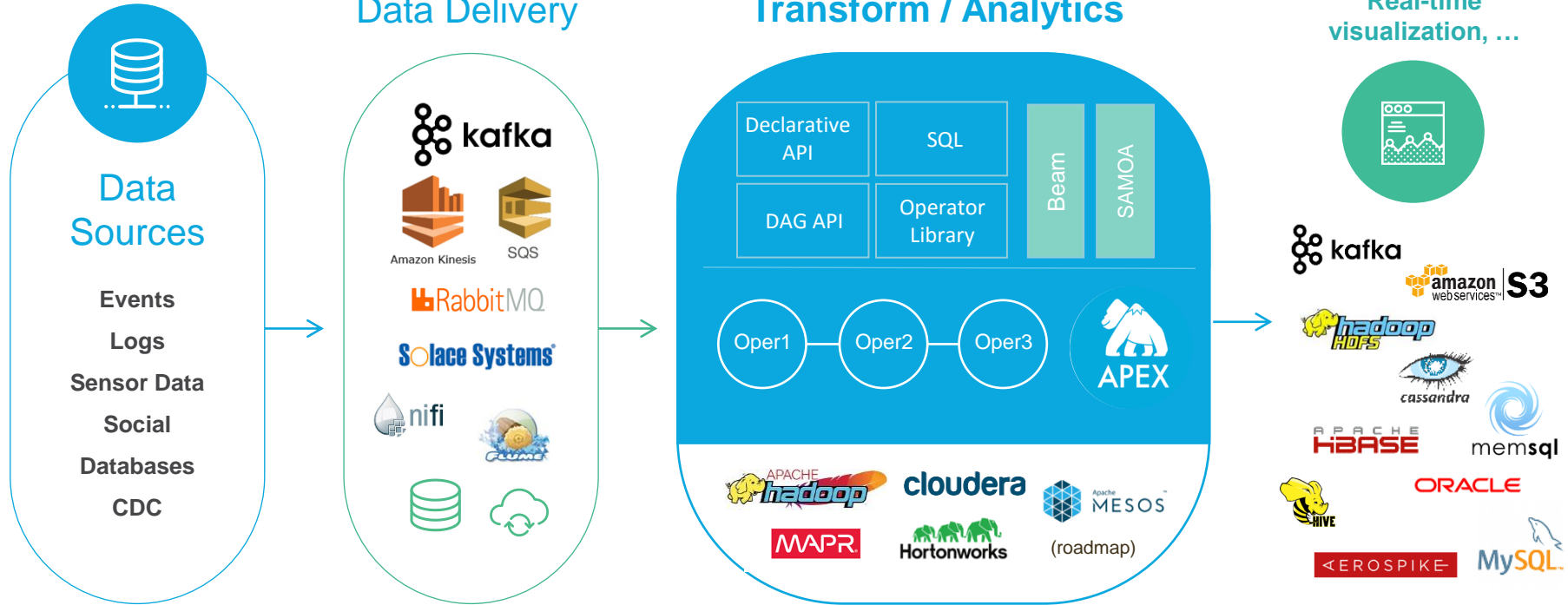




Apache Apex: Next Gen Big Data Analytics







Thomas Weise <thw@apache.org> @thweise
PMC Chair Apache Apex, Architect DataTorrent
Apache Big Data Europe, Sevilla, Nov 14th 2016

Stream Data Processing



Industries & Use Cases



 Financial Services	 Ad-Tech	 Telecom	 Manufacturing	 Energy	 IoT
Fraud and risk monitoring	Real-time customer facing dashboards on key performance indicators	Call detail record (CDR) & extended data record (XDR) analysis	Supply chain planning & optimization	Smart meter analytics	Data ingestion and processing
Credit risk assessment	Click fraud detection	Understanding customer behavior AND context	Preventive maintenance	Reduce outages & improve resource utilization	Predictive analytics
Improve turn around time of trade settlement processes	Billing optimization	Packaging and selling anonymous customer data	Product quality & defect tracking	Asset & workforce management	Data governance

HORIZONTAL

- Large scale ingest and distribution
- Real-time ELTA (Extract Load Transform Analyze)
- Dimensional computation & aggregation

- Enforcing data quality and data governance requirements
- Real-time data enrichment with reference data
- Real-time machine learning model scoring

Apache Apex

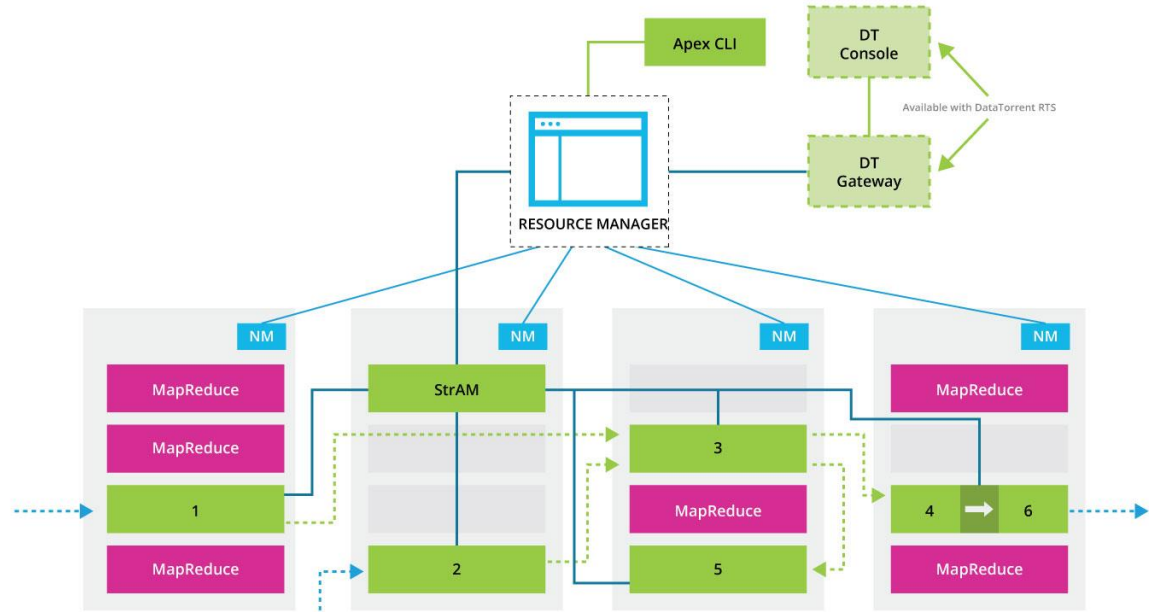


- In-memory, distributed stream processing
 - Application logic broken into components (operators) that execute distributed in a cluster
 - Unobtrusive Java API to express (custom) logic
 - Maintain **state** and metrics in member variables
 - Windowing, event-time processing
- Scalable, high throughput, low latency
 - Operators can be scaled up or down at runtime according to the load and SLA
 - Dynamic scaling (elasticity), compute locality
- Fault tolerance & correctness
 - Automatically recover from node outages without having to reprocess from beginning
 - State is preserved, checkpointing, incremental recovery
 - End-to-end exactly-once
- Operability
 - System and application metrics, record/visualize data
 - Dynamic changes and resource allocation, elasticity

Native Hadoop Integration



- YARN is the resource manager
- HDFS for storing persistent state



Application Development Model



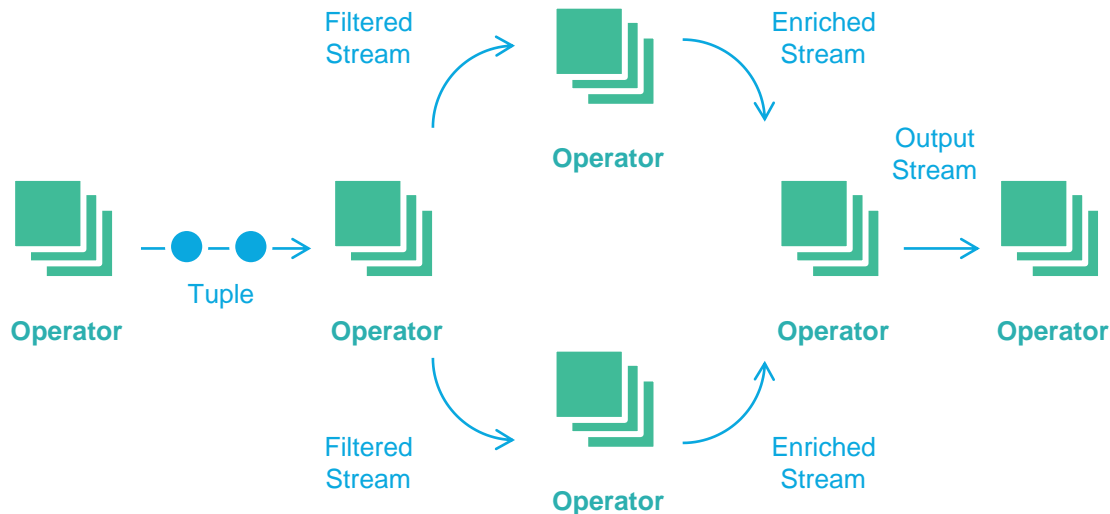
A **Stream** is a sequence of data tuples

A typical **Operator** takes one or more input streams, performs computations & emits one or more output streams

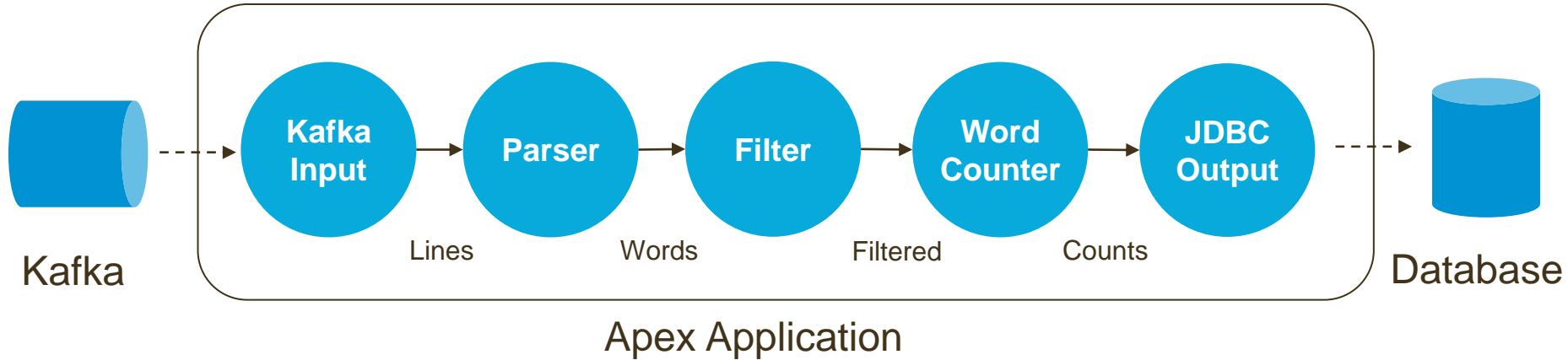
- Each Operator is YOUR custom business logic in java, or built-in operator from our open source library
- Operator has many instances that run in parallel and each instance is single-threaded

Directed Acyclic Graph (DAG) is made up of operators and streams

Directed Acyclic Graph (DAG)



Development Process



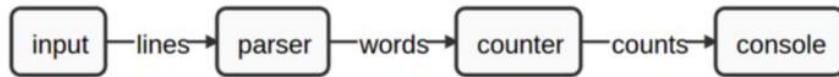
- Operators from library or develop for custom logic
- Connect operators to form application
- Configure operator properties
- Configure scaling and other platform attributes
- Test functionality, performance, iterate

Application Specification



DAG API (compositional)

```
@ApplicationAnnotation(name = "MyFirstApplication")
public class Application implements StreamingApplication
{
    @Override
    public void populateDAG(DAG dag, Configuration conf)
    {
        LineReader lineReader = dag.addOperator("input", new LineReader());
        Parser parser = dag.addOperator("parser", new Parser());
        UniqueCounter counter = dag.addOperator("counter", new UniqueCounter());
        ConsoleOutputOperator cons = dag.addOperator("console", new ConsoleOutputOperator());
        dag.addStream("lines", lineReader.output, parser.input);
        dag.addStream("words", parser.output, counter.data);
        dag.addStream("counts", counter.count, cons.input);
    }
}
```



Java Stream API (declarative)

```
ApexStream stream = StreamFactory.fromFolder(localFolder)
    .flatMap((String input) -> Arrays.asList(input.split(" ")))
    .countByKey().print();
```


Developing Operators



```
public class Parser extends BaseOperator {

    public transient final DefaultInputPort<String> input = new DefaultInputPort<String>() {
        @Override
        public void process(String s) {
            String[] words = s.split("[\\p{Punct}\\\\s\\\\\\\\\"\\\\'\\\\\\\\"]+");
            for (String word : words) {
                output.emit(word);
            }
        }
    };

    public transient final DefaultOutputPort<String> output = new DefaultOutputPort<>();
}

public class UniqueCounter<K> extends BaseOperator {
    private Map<K, MutableInt> counts = new HashMap<>();

    public transient final DefaultInputPort<K> input = (tuple) -> {
        MutableInt count = counts.get(tuple);
        if (count == null) {
            count = new MutableInt();
            counts.put(tuple, count);
        }
        count.increment();
    };

    @Override
    public void endWindow() {
        for (Map.Entry<K, MutableInt> entry : counts.entrySet()) {
            output.emit(new KeyValPair<K, Integer>(entry.getKey(), entry.getValue().toInteger()));
        }
    }

    public transient final DefaultOutputPort<KeyValPair<K, Integer>> output = new DefaultOutputPort<>();
}
```

Operator Library



Messaging

- Kafka
- JMS (ActiveMQ, ...)
- Kinesis, SQS
- Flume, NiFi

NoSQL

- Cassandra, HBase
- Aerospike, Accumulo
- Couchbase/ CouchDB
- Redis, MongoDB
- Geode

RDBMS

- JDBC
- MySQL
- Oracle
- MemSQL

File Systems

- HDFS/ Hive
- NFS
- S3

Parsers

- XML
- JSON
- CSV
- Avro
- Parquet

Transformations

- Filter, Expression, Enrich
- Windowing, Aggregation
- Join
- Dedup

Analytics

- Dimensional Aggregations
(with state management for
historical data + query)

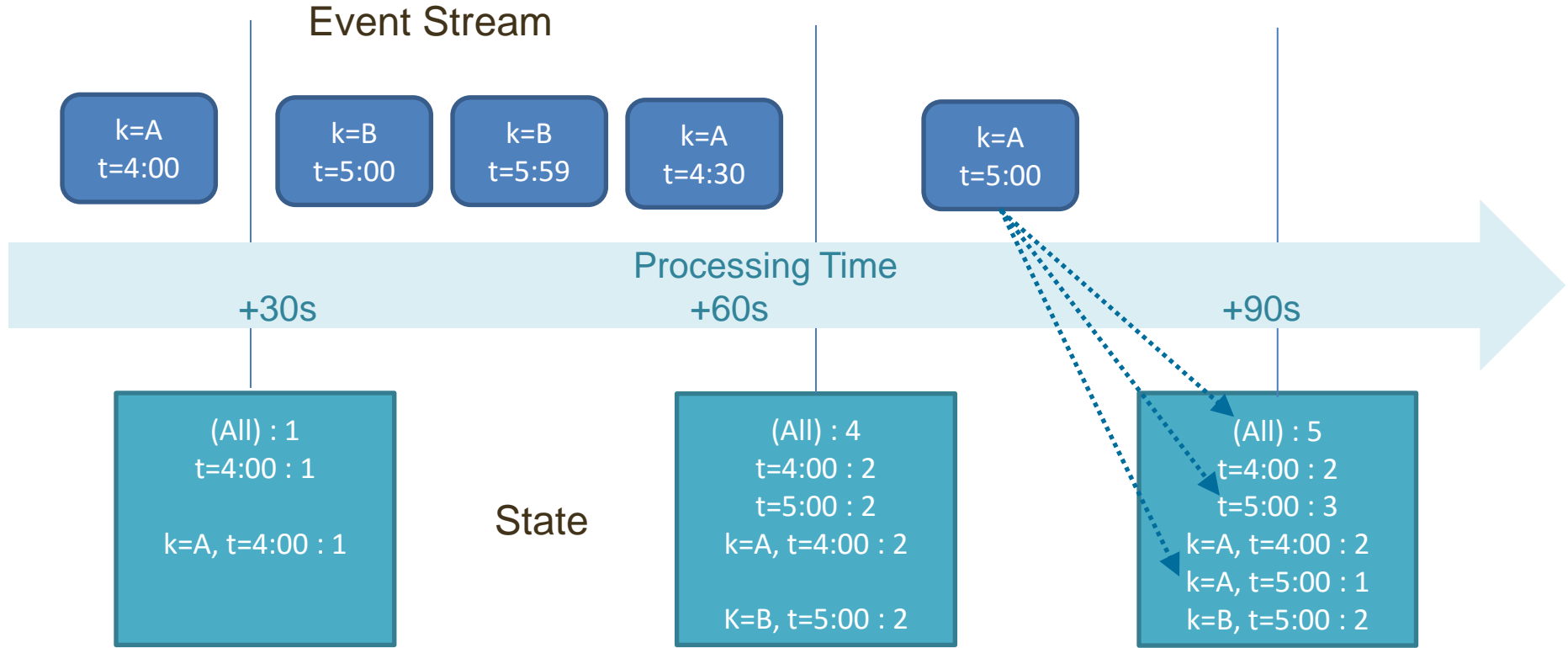
Protocols

- HTTP
- FTP
- WebSocket
- MQTT
- SMTP

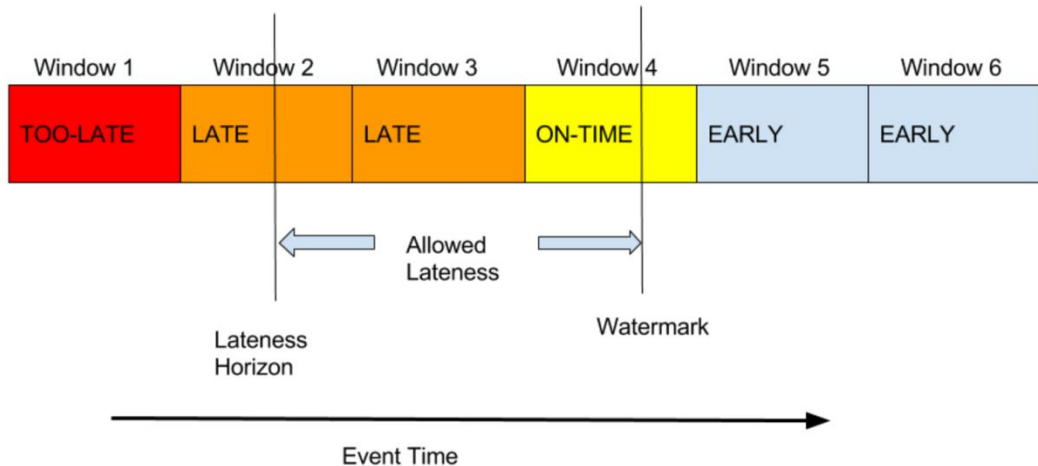
Other

- Elastic Search
- Script (JavaScript, Python, R)
- Solr
- Twitter

Stateful Processing with Event Time



Windowing - Apache Beam Model



Event-time

Session windows

Watermarks

Accumulation

Triggers

Keyed or Not Keyed

Allowed Lateness

Accumulation Mode

Merging streams

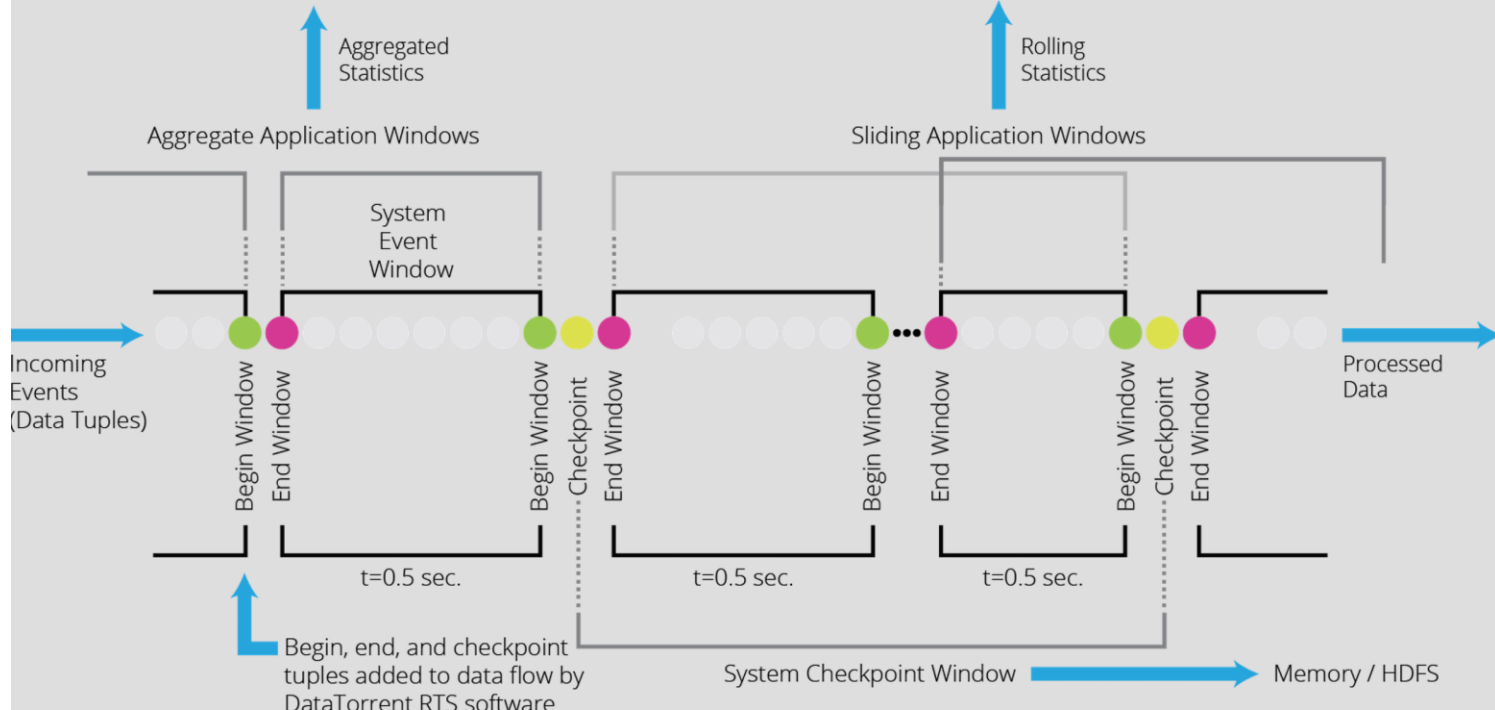
```
ApexStream<String> stream = StreamFactory
    .fromFolder(localFolder)
    .flatMap(new Split())
    .window(new WindowOption.GlobalWindow(), new
        TriggerOption().withEarlyFiringsAtEvery(Duration.millis(1000)).accumulatingFiredPanels())
    .countByKey(new ConvertToKeyVal()).print();
```

Fault Tolerance



- Operator state is checkpointed to persistent store
 - Automatically performed by engine, no additional coding needed
 - Asynchronous and distributed
 - In case of failure operators are restarted from checkpoint state
- Automatic detection and recovery of failed containers
 - Heartbeat mechanism
 - YARN process status notification
- Buffering to enable replay of data from recovered point
 - Fast, incremental recovery, spike handling
- Application master state checkpointed
 - Snapshot of physical (and logical) plan
 - Execution layer change log

Checkpointing State

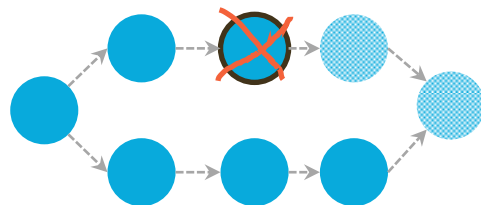
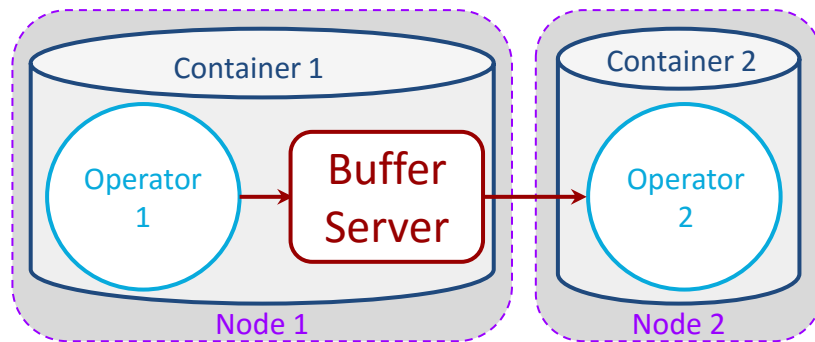


- Distributed, asynchronous
- Periodic callbacks
- No artificial latency
- Pluggable storage

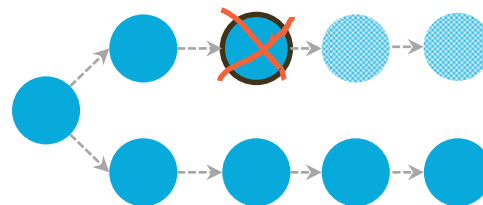
Buffer Server & Recovery



- In-memory PubSub
- Stores results until committed
- Backpressure / spillover to disk
- Ordering, idempotency

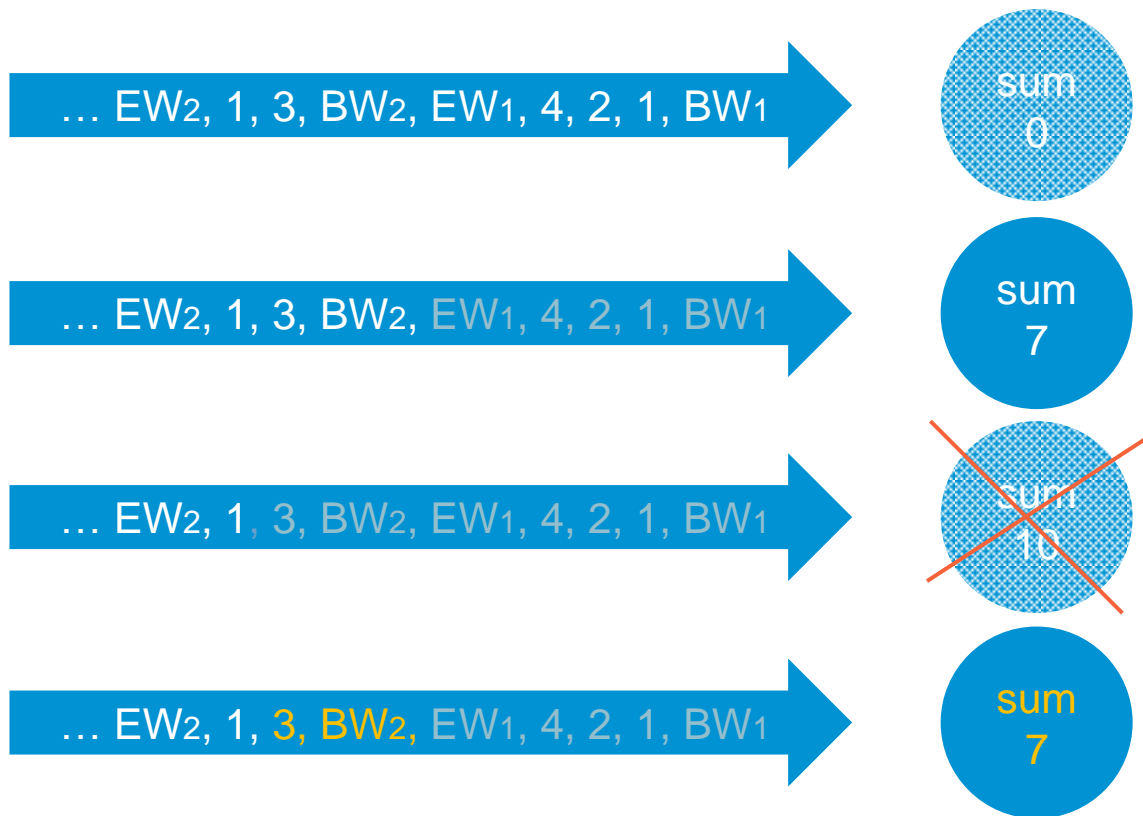


Downstream Operators reset



Independent pipelines
(can be used for speculative execution)

Recovery Scenario



Processing Guarantees



At-least-once

- On recovery data will be replayed from a previous checkpoint
 - No messages lost
 - Default, suitable for most applications
- Can be used to ensure data is written once to store
 - Transactions with meta information, Rewinding output, Feedback from external entity, Idempotent operations

At-most-once

- On recovery the latest data is made available to operator
 - Useful in use cases where some data loss is acceptable and latest data is sufficient

Exactly-once

- At-least-once + idempotency + transactional mechanisms (operator logic) to achieve end-to-end exactly once behavior

End-to-End Exactly Once



- Important when writing to external systems
- Data should not be duplicated or lost in the external system in case of application failures
- Common external systems
 - Databases
 - Files
 - Message queues
- Exactly-once = at-least-once + idempotency + consistent state
- Data duplication must be avoided when data is replayed from checkpoint
 - Operators implement the logic dependent on the external system
 - Platform provides checkpointing and repeatable windowing

Scalability

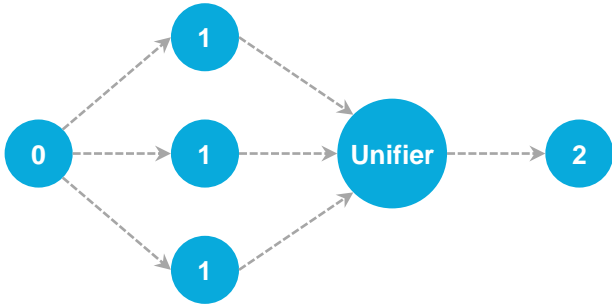


Unifier

Logical Diagram



Physical Diagram with operator 1 with 3 partitions

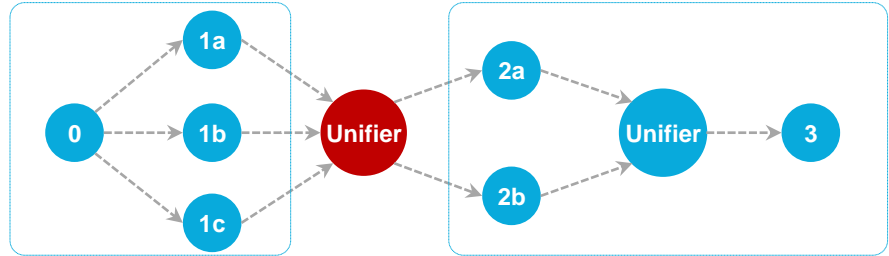


NxM Partitions

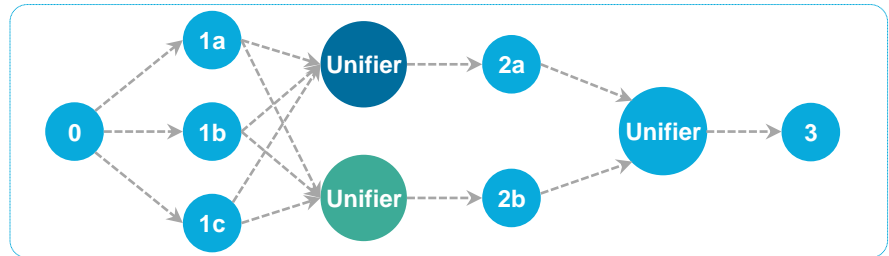
Logical DAG



Physical DAG with (1a, 1b, 1c) and (2a, 2b): Bottleneck on intermediate Unifier



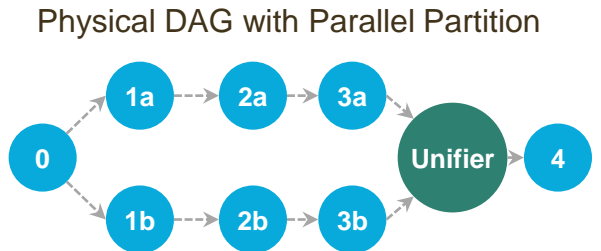
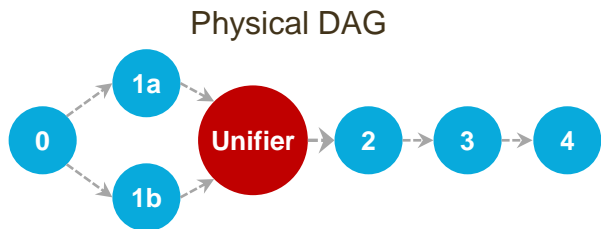
Physical DAG with (1a, 1b, 1c) and (2a, 2b): No bottleneck



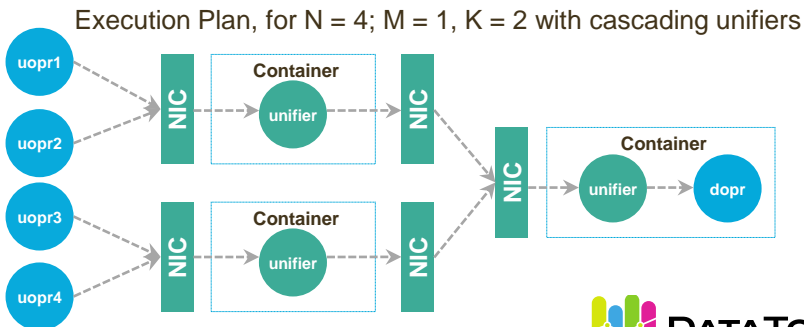
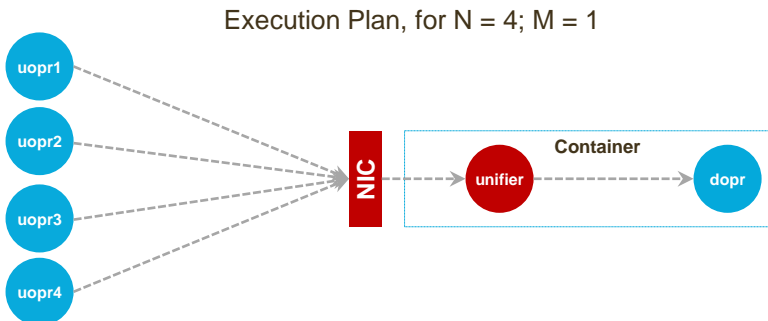
Advanced Partitioning



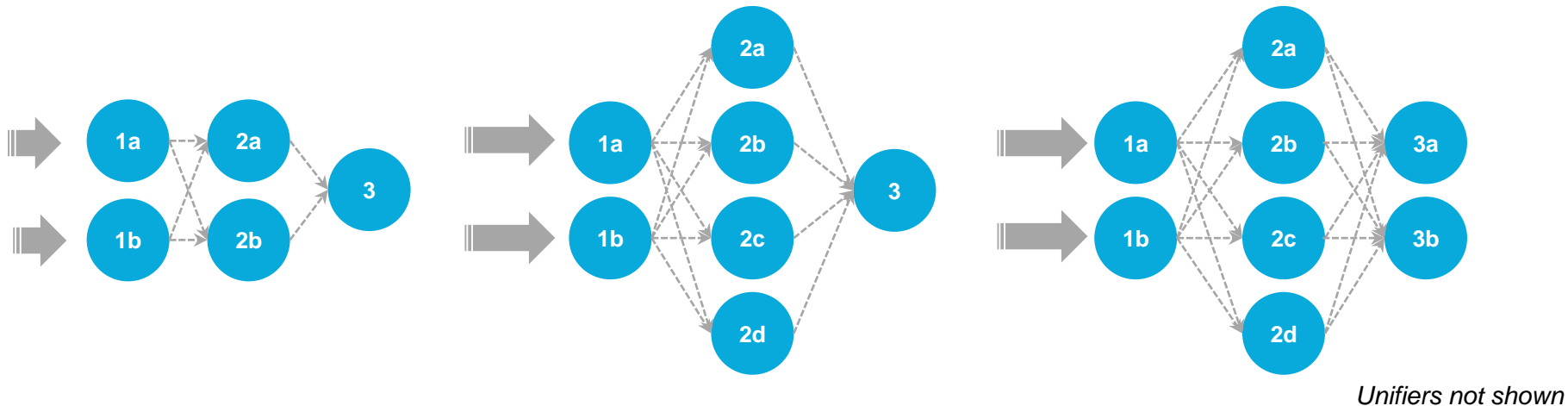
Parallel Partition



Cascading Unifiers



Dynamic Partitioning



- Partitioning change while application is running
 - Change number of partitions at runtime based on stats
 - Determine initial number of partitions dynamically
 - Kafka operators scale according to number of kafka partitions
 - Supports re-distribution of state when number of partitions change
 - API for custom scaler or partitioner

How dynamic partitioning works



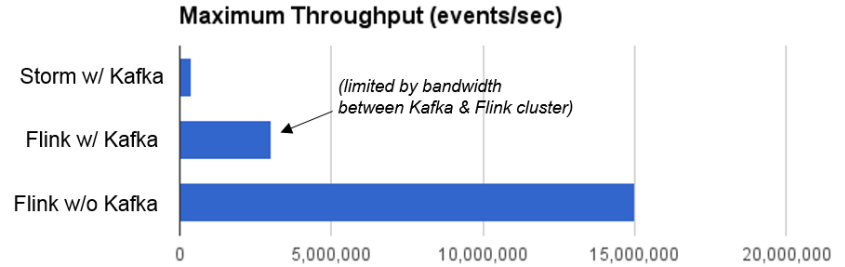
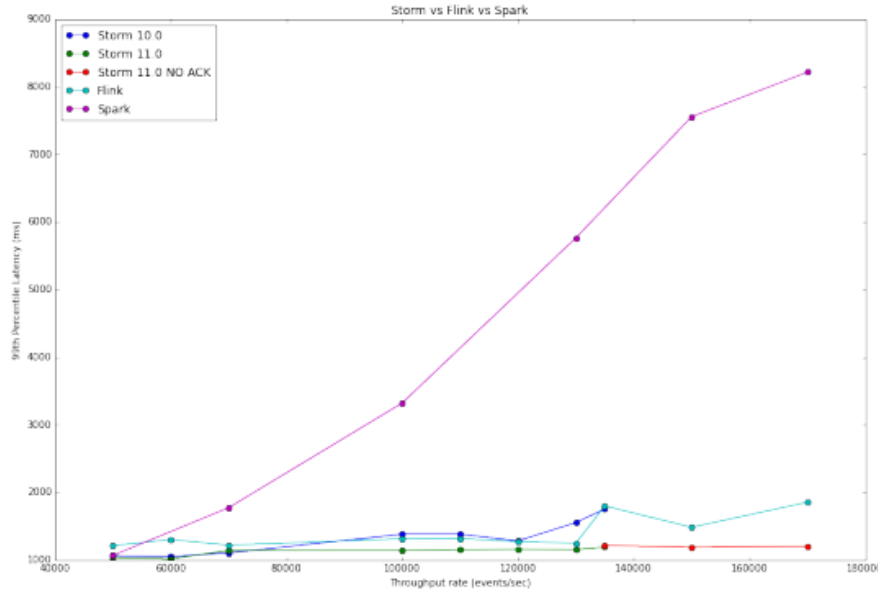
- Partitioning decision (yes/no) by trigger (StatsListener)
 - Pluggable component, can use any system or custom metric
 - Externally driven partitioning example: KafkaInputOperator
- Stateful!
 - Uses checkpointed state
 - Ability to transfer state from old to new partitions (partitioner, customizable)
 - Steps:
 - Call partitioner
 - Modify physical plan, rewrite checkpoints as needed
 - Undeploy old partitions from execution layer
 - Release/request container resources
 - Deploy new partitions (from rewritten checkpoint)
 - No loss of data (buffered)
 - Incremental operation, partitions that don't change continue processing
- API: [Partitioner interface](#)

Compute Locality



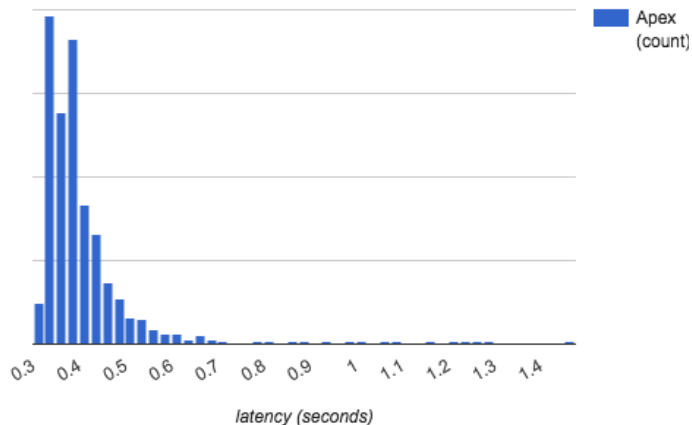
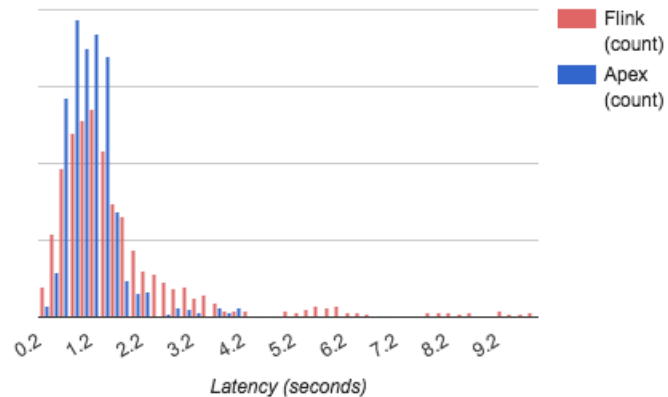
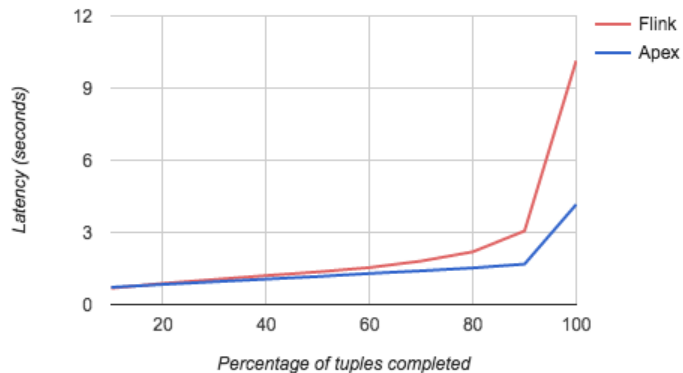
- By default operators are deployed in containers (processes) on different nodes across the Hadoop cluster
- Locality options for streams
 - RACK_LOCAL: Data does not traverse network switches
 - NODE_LOCAL: Data transfer via loopback interface, frees up network bandwidth
 - CONTAINER_LOCAL: Data transfer via in memory queues between operators, does not require serialization
 - THREAD_LOCAL: Data passed through call stack, operators share thread
- Host Locality
 - Operators can be deployed on specific hosts
- (Anti-)Affinity
 - Ability to express relative deployment without specifying a host

Performance: Throughput vs. Latency?



<https://yahooeng.tumblr.com/post/135321837876/benchmarking-streaming-computation-engines-at>
<http://data-artisans.com/extending-the-yahoo-streaming-benchmark/>

High-Throughput and Low-Latency



Apex, Flink w/ 4 Kafka brokers
2.7 million events/second, Kafka latency limit

Apex w/o Kafka and Redis:
43 million events/second with more than 90
percent of events processed with the latency
less than 0.5 seconds

<https://www.datatorrent.com/blog/throughput-latency-and-yahoo/>

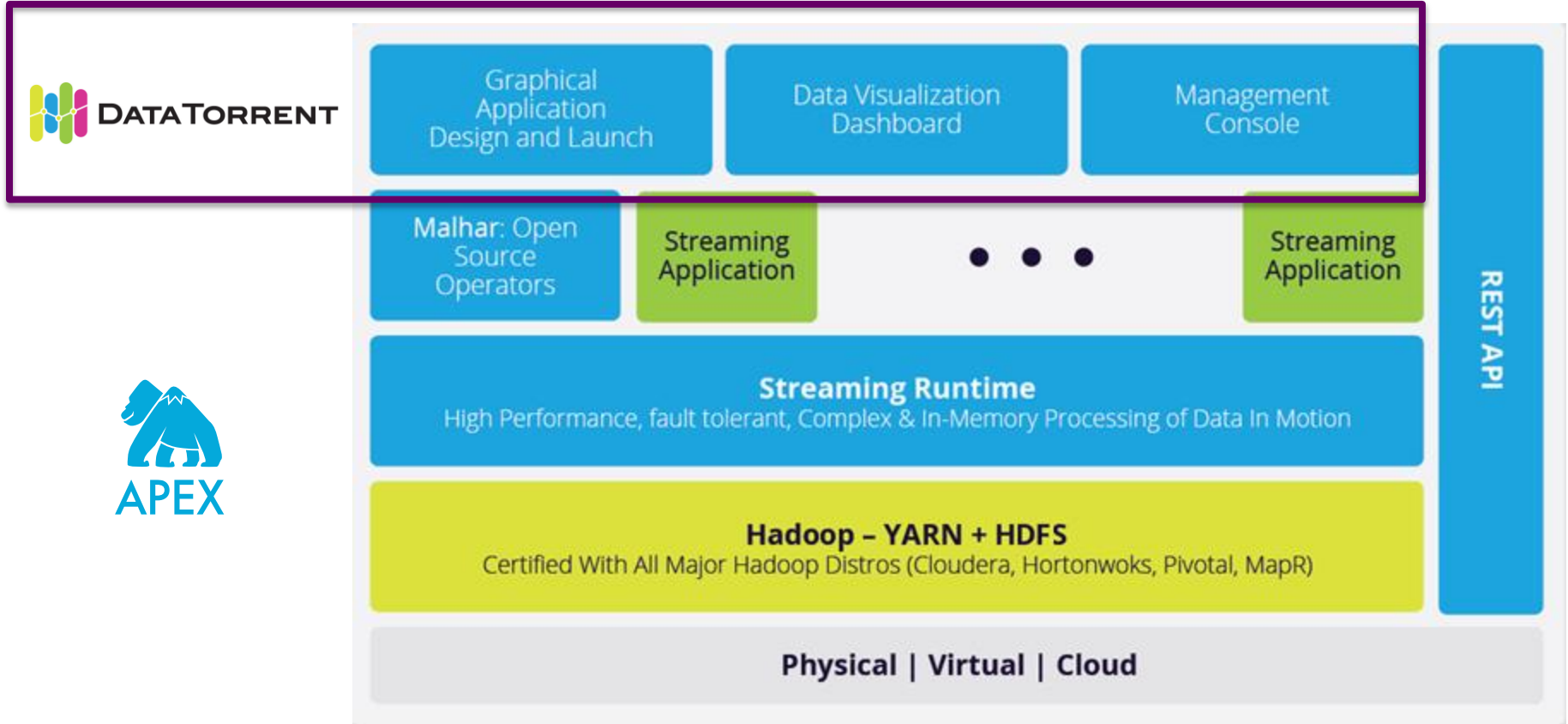
Recent Additions & Roadmap



- Declarative Java API
- Windowing Semantics following Beam model
- Scalable state management
- SQL support using Apache Calcite
- Apache Beam Runner, SAMOA integration

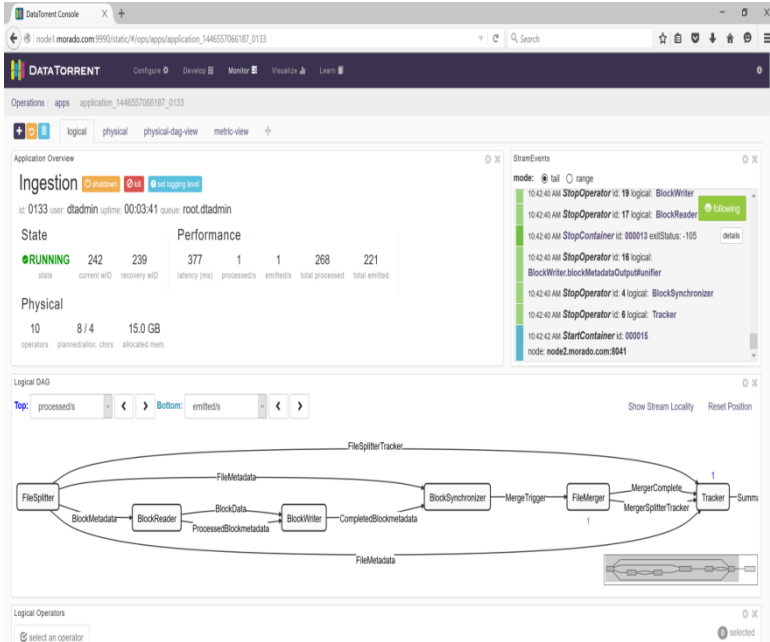
- Enhanced support for Batch Processing
- Support for Mesos
- Encrypted Streams
- Python support for operator logic and API
- Replacing operator code at runtime
- Dynamic attribute changes
- Named checkpoints

Enterprise Tools

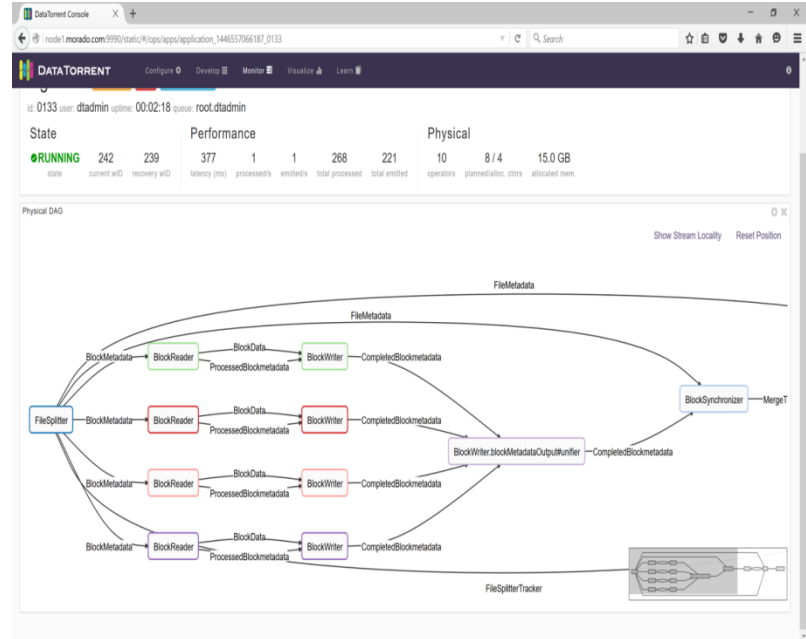


Monitoring Console

Logical View



Physical View

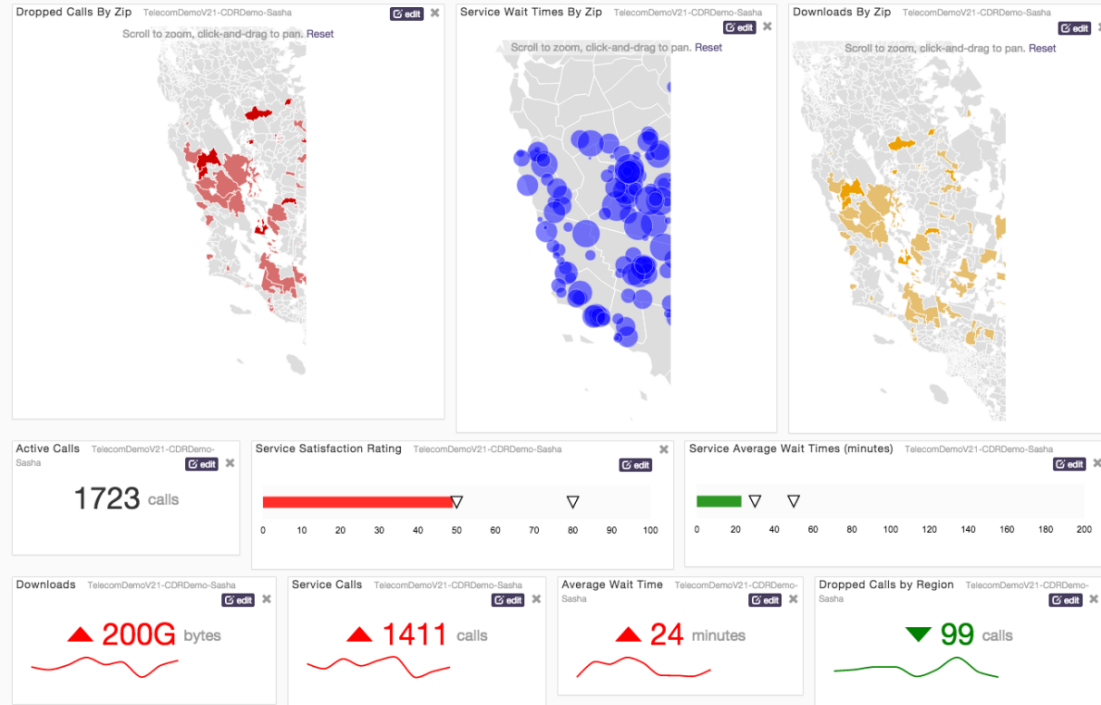


Real-Time Dashboards

DataTorrent Telecom Service Overview

360 degree view of operations

+ add widget auto generate save changes (5) settings



Maximize Revenue w/ real-time insights



PubMatic

PubMatic is the leading marketing automation software company for publishers. Through real-time analytics, yield management, and workflow automation, PubMatic enables publishers to make smarter inventory decisions and improve revenue performance

Business Need

- Ingest and analyze high volume clicks & views in real-time to help customers improve revenue
 - 200K events/second data flow
- Report critical metrics for campaign monetization from auction and client logs
 - 22 TB/day data generated
- Handle ever increasing traffic with efficient resource utilization
- Always-on ad network



Apex based Solution

- DataTorrent Enterprise platform, powered by Apache Apex
- In-memory stream processing
- Comprehensive library of pre-built operators including connectors
- Built-in fault tolerance
- Dynamically scalable
- Real-time query from in-memory state
- Management UI & Data Visualization console



Client Outcome

- Helps PubMatic deliver ad performance insights to publishers and advertisers in **real-time instead of 5+ hours**
- Helps Publishers visualize campaign performance and adjust ad inventory in real-time to maximize their revenue
- Enables PubMatic reduce OPEX with efficient compute resource utilization
- Built-in fault tolerance ensures customers can **always** access ad network

Industrial IoT applications



GE is dedicated to providing advanced IoT analytics solutions to thousands of customers who are using their devices and sensors across different verticals. GE has built a sophisticated analytics platform, Predix, to help its customers develop and execute Industrial IoT applications and gain real-time insights as well as actions.

Business Need

- Ingest and analyze high-volume, high speed data from thousands of devices, sensors per customer in real-time without data loss
- Predictive analytics to reduce costly maintenance and improve customer service
- Unified monitoring of all connected sensors and devices to minimize disruptions
- Fast application development cycle
- High scalability to meet changing business and application workloads



Apex based Solution

- Ingestion application using DataTorrent Enterprise platform
- Powered by Apache Apex
- In-memory stream processing
- Built-in fault tolerance
- Dynamic scalability
- Comprehensive library of pre-built operators
- Management UI console



Client Outcome

- Helps GE improve performance and lower cost by enabling real-time Big Data analytics
- Helps GE detect possible failures and minimize unplanned downtimes with centralized management & monitoring of devices
- Enables faster innovation with short application development cycle
- No data loss and 24x7 availability of applications
- Helps GE adjust to scalability needs with auto-scaling

Smart energy applications



Silver Spring Networks helps global utilities and cities connect, optimize, and manage smart energy and smart city infrastructure. Silver Spring Networks receives data from over 22 million connected devices, conducts 2 million remote operations per year

Business Need

- Ingest high-volume, high speed data from millions of devices & sensors in real-time without data loss
- Make data accessible to applications without delay to improve customer service
- Capture & analyze historical data to understand & improve grid operations
- Reduce the cost, time, and pain of integrating with 3rd party apps
- Centralized management of software & operations



Apex based Solution

- DataTorrent Enterprise platform, powered by Apache Apex
- In-memory stream processing
- Pre-built operators/connectors
- Built-in fault tolerance
- Dynamically scalable
- Management UI console



Client Outcome

- Helps Silver Spring Networks ingest & analyze data in real-time for effective load management & customer service
- Helps Silver Spring Networks detect possible failures and reduce outages with centralized management & monitoring of devices
- Enables fast application development for faster time to market
- Helps Silver Spring Networks scale with easy to partition operators
- Automatic recovery from failures

Who is using Apex?

- Powered by Apex
 - <http://apex.apache.org/powered-by-apex.html>
 - Also using Apex? Let us know to be added: users@apex.apache.org or @ApacheApex
- Pubmatic
 - <https://www.youtube.com/watch?v=JSXpgfQFcU8>
- GE
 - <https://www.youtube.com/watch?v=hmaSkXhHNu0>
 - <http://www.slideshare.net/ApacheApex/ge-iot-predix-time-series-data-ingestion-service-using-apache-apex-hadoop>
- SilverSpring Networks
 - <https://www.youtube.com/watch?v=8VORISKeSjl>
 - <http://www.slideshare.net/ApacheApex/iot-big-data-ingestion-and-processing-in-hadoop-by-silver-spring-networks>

Q&A

Resources



- <http://apex.apache.org/>
- Learn more - <http://apex.apache.org/docs.html>
- Subscribe - <http://apex.apache.org/community.html>
- Download - <http://apex.apache.org/downloads.html>
- Follow @ApacheApex - <https://twitter.com/apacheapex>
- Meetups - <https://www.meetup.com/topics/apache-apex/>
- Examples - <https://github.com/DataTorrent/examples>
- Slideshare - <http://www.slideshare.net/ApacheApex/presentations>
- https://www.youtube.com/results?search_query=apache+apex
- Free Enterprise License for Startups - <https://www.datatorrent.com/product/startup-accelerator/>