Hortonworks

Real Time Processing with Hadoop

Hortonworks. We do Hadoop.

Topics – Hortonworks and Apache Storm

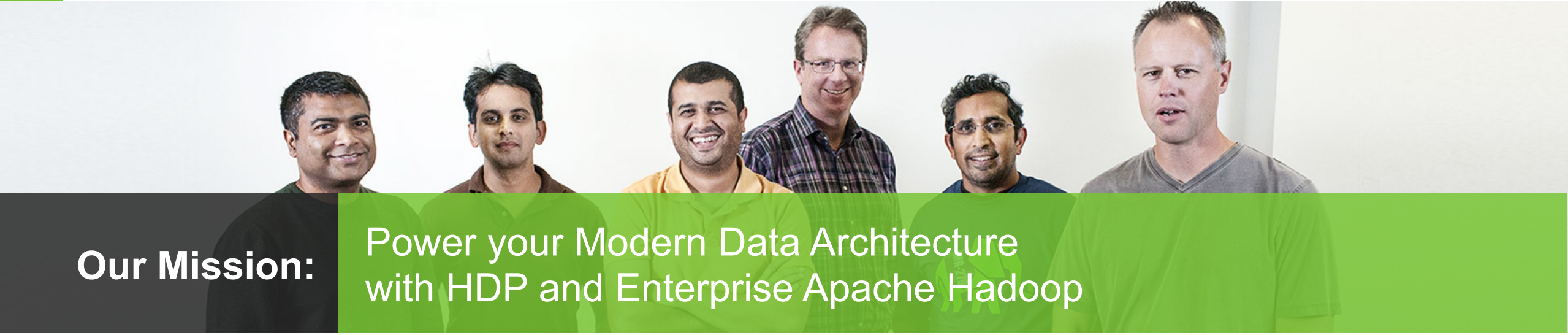
- Hortonworks and Enterprise Hadoop
- Stream Processing with Apache Storm – Introduction
- Apache Storm – Key Constructs
- Apache Storm – Data Flows and Architecture
- Apache Storm Usage Cases and Real World Walkthrough
- Apache Storm Demo and How To

Hortonworks

Hortonworks

Hortonworks and Enterprise Hadoop

Hortonworks. We do Hadoop.



Our Mission:

Power your Modern Data Architecture with HDP and Enterprise Apache Hadoop

Who we are

June 2011: Original 24 architects, developers, operators of Hadoop from Yahoo!
June 2014: An enterprise software company with 420+ Employees

Our model

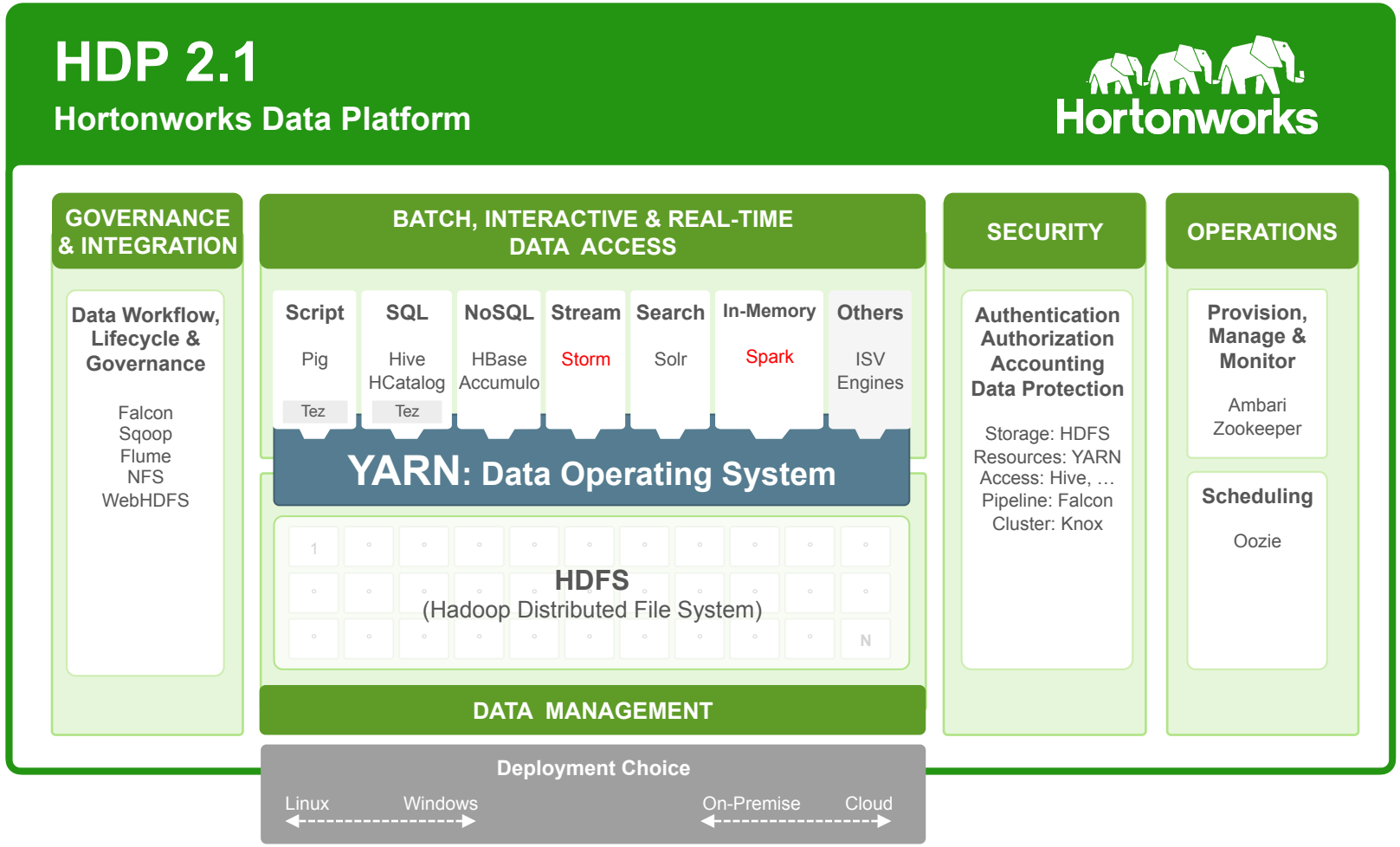
Innovate and deliver Apache Hadoop as a complete enterprise data platform completely in the open, backed by a world class support organization

Key Partners

Microsoft TERADATA SAP hp sas CSC accenture …and hundreds more

Hortonworks

HDP delivers a comprehensive data management platform



YARN is the architectural center of HDP

- Enables batch, interactive and real-time workloads
- Single SQL engine for both batch and interactive
- Enable existing ISV apps to plug directly into Hadoop via YARN

Provides comprehensive enterprise capabilities

- Governance
- Security
- Operations

The widest range of deployment options

- Linux & Windows
- On premise & cloud

Hortonworks

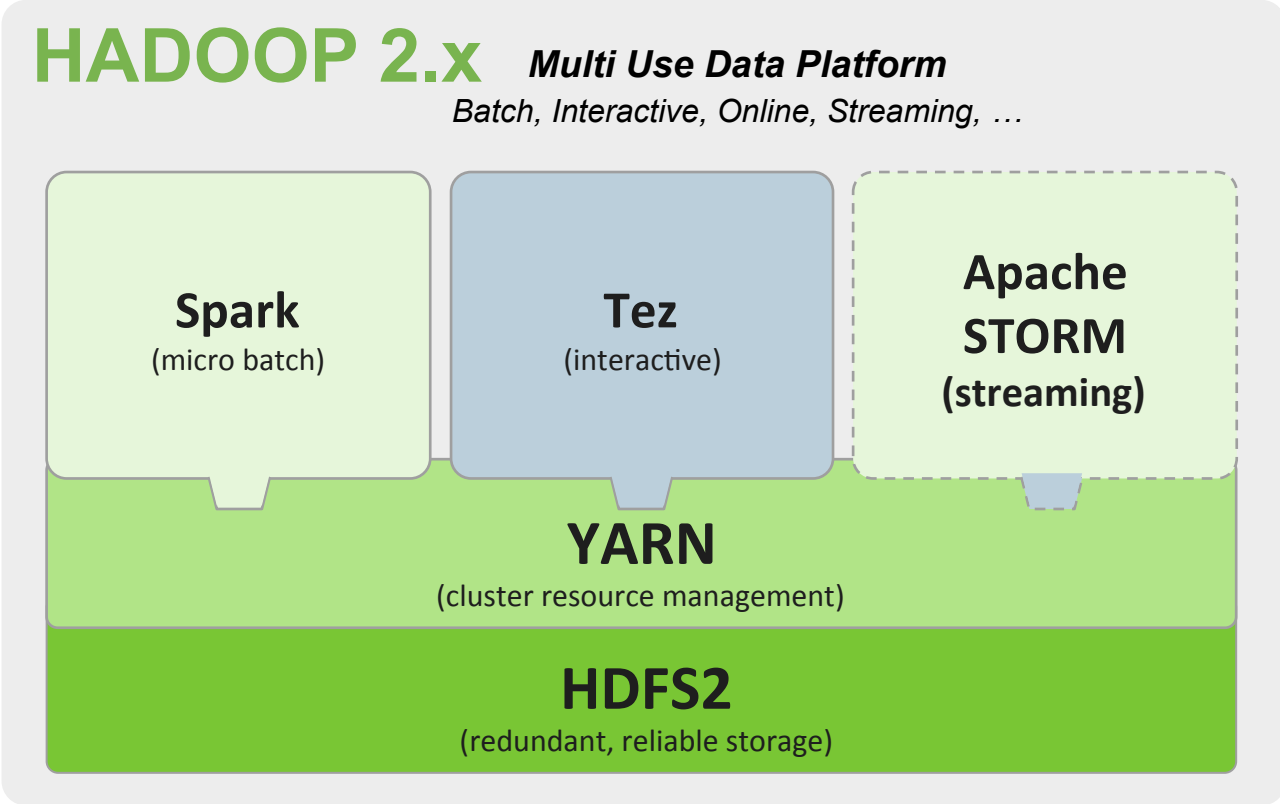
Choice Of Tool For Stream Processing

Storm

- Event by Event processing
- Suited for sub second real time processing
  - Fraud detection

Spark Streaming

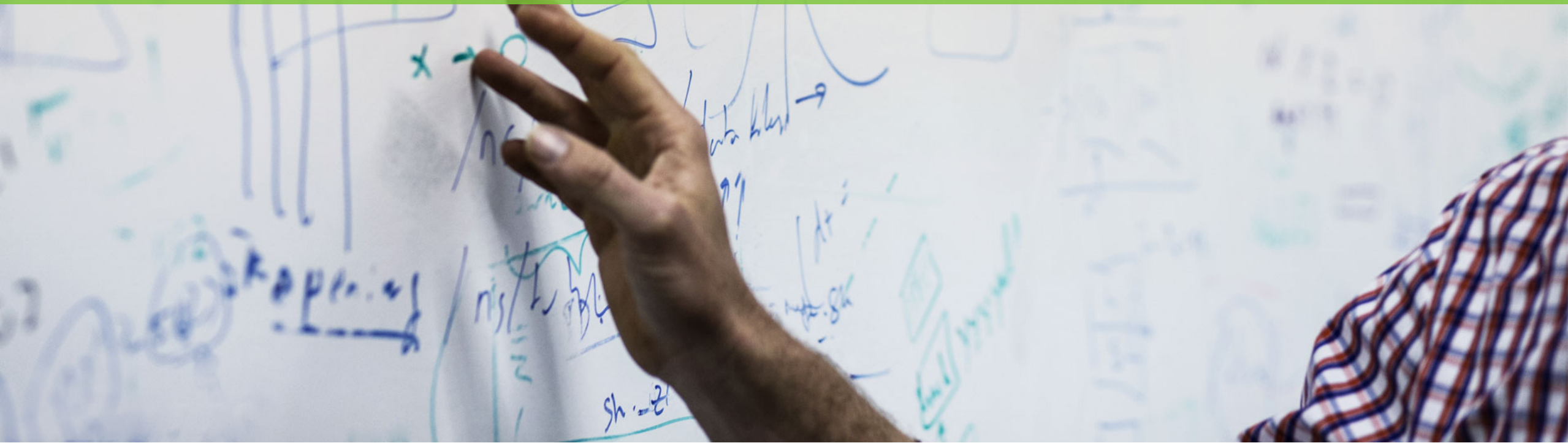
- Micro-Batch processing
- Suited for near real time processing
  - Trending Facebook likes



Hortonworks

Hortonworks

Stream Processing with Apache Storm



Why stream processing IN Hadoop?

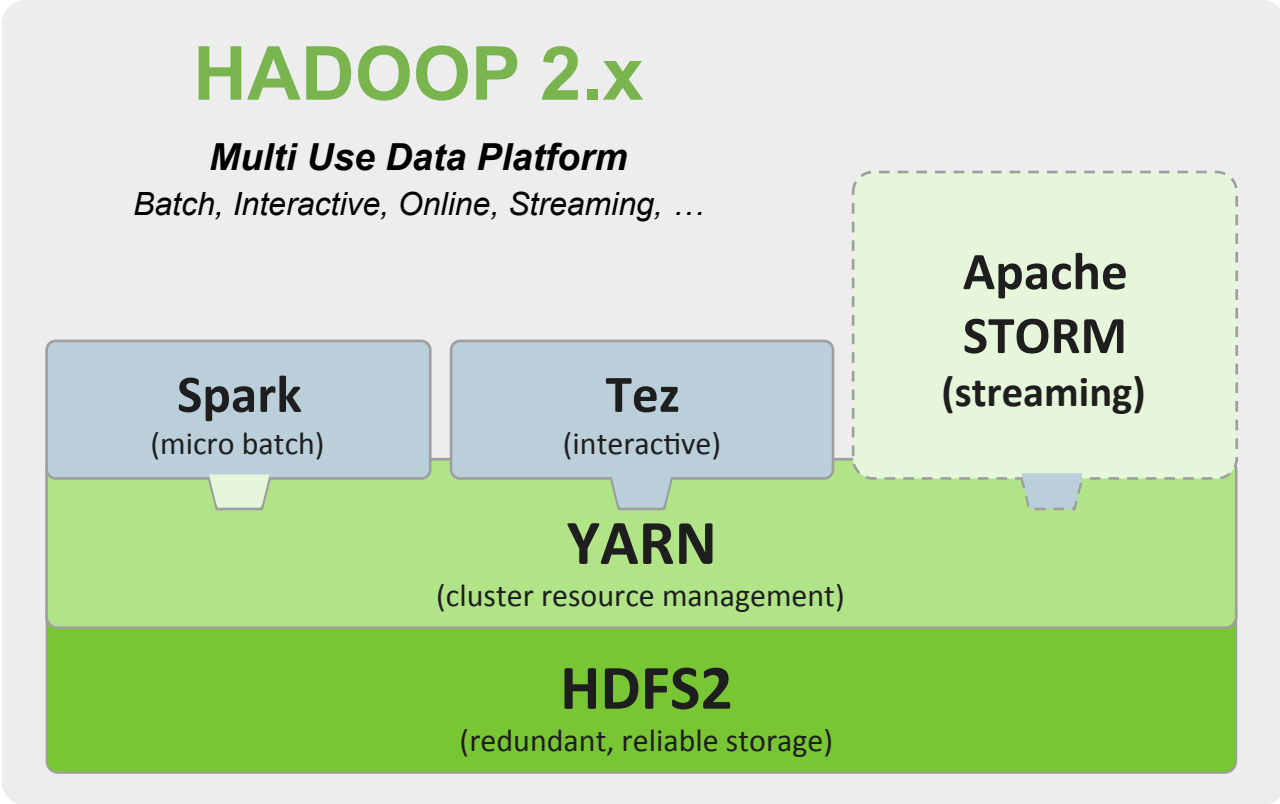
Stream processing has emerged as a key use case

What is the need?

- Exponential rise in real-time data
- Ability to process real-time data opens new business opportunities

Why Now?

- Economics of Open source software & commodity hardware
- YARN allows multiple computing paradigms to co-exist in the data lake



Hortonworks

Storm Use Cases – Business View

Sentiment **Clickstream** **Machine/Sensor** **Logs** **Geo-location**

Monitor real-time data to..

	Prevent	Optimize
Finance	- Securities Fraud - Compliance violations	- Order routing - Pricing
Telco	- Security breaches - Network Outages	- Bandwidth allocation - Customer service
Retail	- Shrinkage - Stock outs	- Offers - Pricing
Manufacturing	- Device failures	- Supply chain
Transportation	- Driver & fleet issues	- Routes - Pricing
Web	- Application failures - Operational issues	- Site content

Hortonworks

Stream processing very different from batch

Factors		Streaming	Batch
Data	**Freshness**	Real-time (usually < 15 min)	Historical – usually more than 15 min old
	Location	Primarily memory (moved to disk after processing)	Primarily in disk moved to memory for processing
Processing	**Speed**	Sub second to few seconds	Few minutes to hours
	Frequency	Always running	Sporadic to periodic
Clients	**Who?**	Automated systems only	Human & automated systems
	Type	Primarily operational systems	Primarily analytical applications

Hortonworks

Key Requirements of a Streaming Solution

Data Ingest	• Extremely high ingest rates – millions of events/second
Processing	• Ability to easily plug different processing frameworks • Guaranteed processing – at least once processing semantics
Persistence	• Ability to persist data to multiple relational and non- relational data stores
Operations	• Security, HA, fault tolerance & management support

Hortonworks

Apache Storm Leading for Stream Processing

Open source, real-time event stream processing platform that provides fixed, continuous, & low latency processing for very high frequency streaming data

Highly scalable	• Horizontally scalable like Hadoop • Eg: 10 node cluster can process 1M tuples per second
Fault-tolerant	• Automatically reassigns tasks on failed nodes
Guarantees processing	• Supports at least once & exactly once processing semantics
Language agnostic	• Processing logic can be defined in any language
Apache project	• Brand, governance & a large active community

Hortonworks

Storm Use Cases – IT operations view

Continuous processing	• Continuously ingest high rate messages, process them and update data stores
High speed data aggregation	• Aggregate multiple data streams that emit data at extremely high rates into one central data store
Data filtering	• Filter out unwanted data on the fly before it is persisted to a data store
Distributed RPC response time reduction	• Extremely resource(CPU, mem or I/O) intensive processing that would take long time to process on a single machine can be parallelized with Storm to reduce response times to seconds

Hortonworks

Hortonworks

Apache Storm Key Constructs

Hortonworks. We do Hadoop.

Key Constructs in Apache Storm

Basic Concepts of a Topology

Tuples and Streams, Sprouts, Bolts

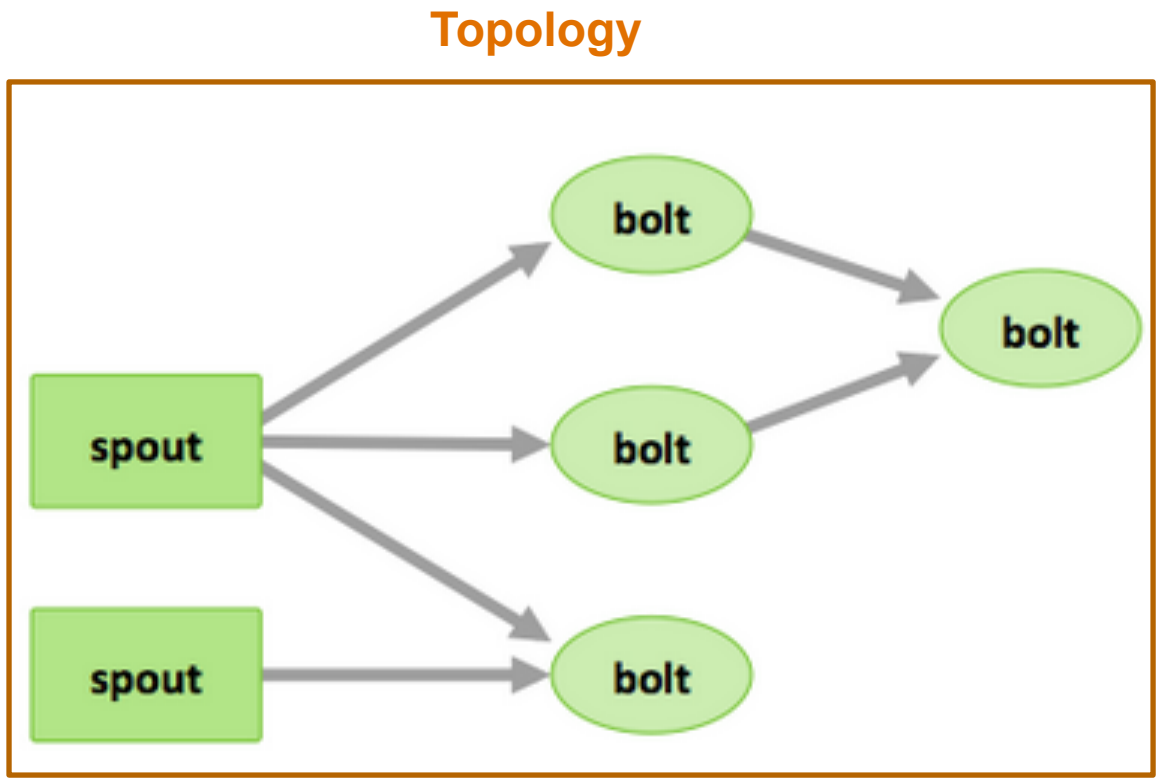
Field Grouping

Architecture and Topology Submission

Parallelism

Processing Guarantee

Basic Concepts



Spouts: Generate streams.

Tuple: Most fundamental data structure and is a named list of values that can be of any datatype

Streams: Groups of tuples

Bolts: Contain data processing, persistence and alerting logic. Can also emit tuples for downstream bolts

Tuple Tree: First spout tuple and all the tuples that were emitted by the bolts that processed it

Topology: Group of spouts and bolts wired together into a workflow

Hortonworks

Tuples and Streams

What is a Tuple?

- Fundamental data structure in Storm. Is a named list of values that can be of any data type.

```
new Values(driverId, truckId, eventTime, eventType, longitude, latitude, eventKey, correlationId);
```

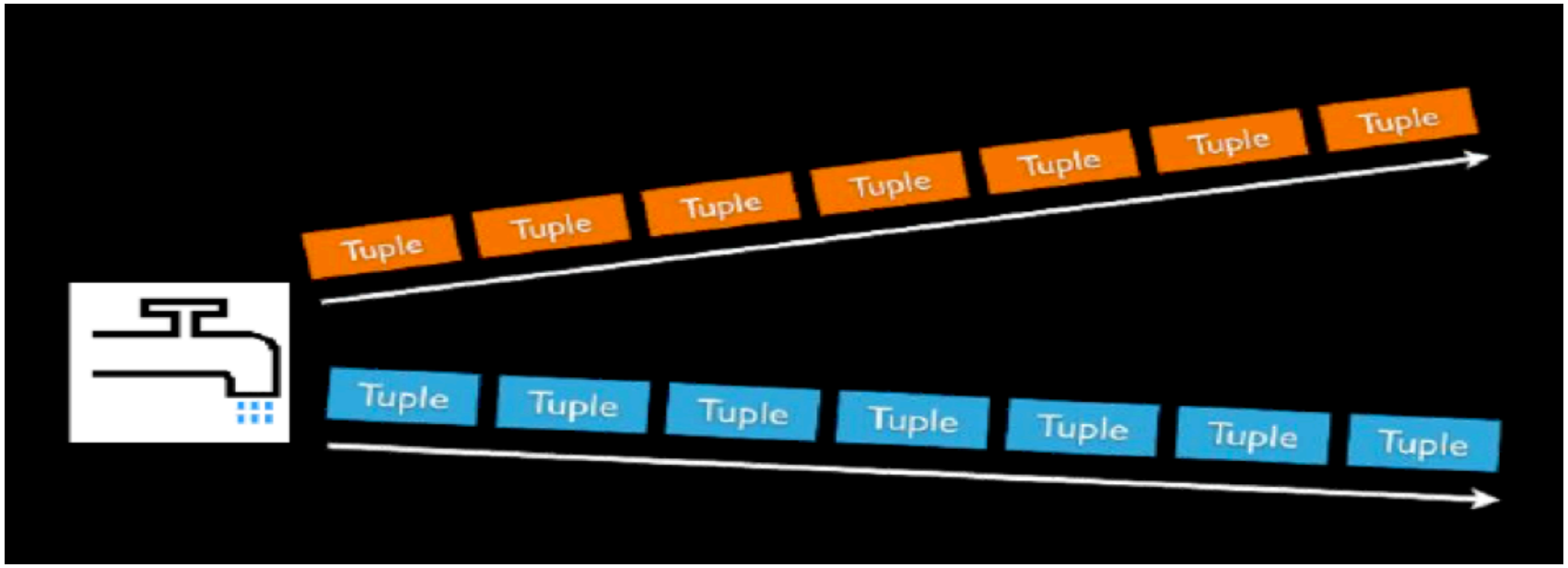
- **What is a Stream?**
  - An unbounded sequences of tuples.
  - Core abstraction in Storm and are what you “process” in Storm

Hortonworks

Spouts

What is a Spout?

- Generates or a source of Streams
  - E.g.: JMS, Twitter, Log, Kafka Spout
- Can spin up multiple instances of a Spout and dynamically adjust as needed



Hortonworks

Bolts

What is a Bolt?

- Processes any number of input streams and produces output streams
- Common processing in bolts are functions, aggregations, joins, read/write to data stores, alerting logic
- Can spin up multiple instances of a Bolt and dynamically adjust as needed

Example of Bolts:

1. HBaseBolt: persist stream in Hbase
2. HDFSBolt: persist stream into HFDS as Avro Files using Flume
3. MonitoringBolt: Read from Hbase and create alerts via email and messaging queues if given thresholds are exceeded.

Fields Grouping

Question: If multiple instances of single bolt can be created, when a tuple is emitted, how do you control what instance the tuple goes to?

Answer: Field Grouping

What is a Field grouping?

- Provides various ways to control tuple routing to bolts.
- Many field grouping exist include shuffle, fields, global..

Example of Field grouping in Use Case

- Events for a given driver/truck must be sent to the same monitoring bolt instance.

Hortonworks

Different Fields Grouping

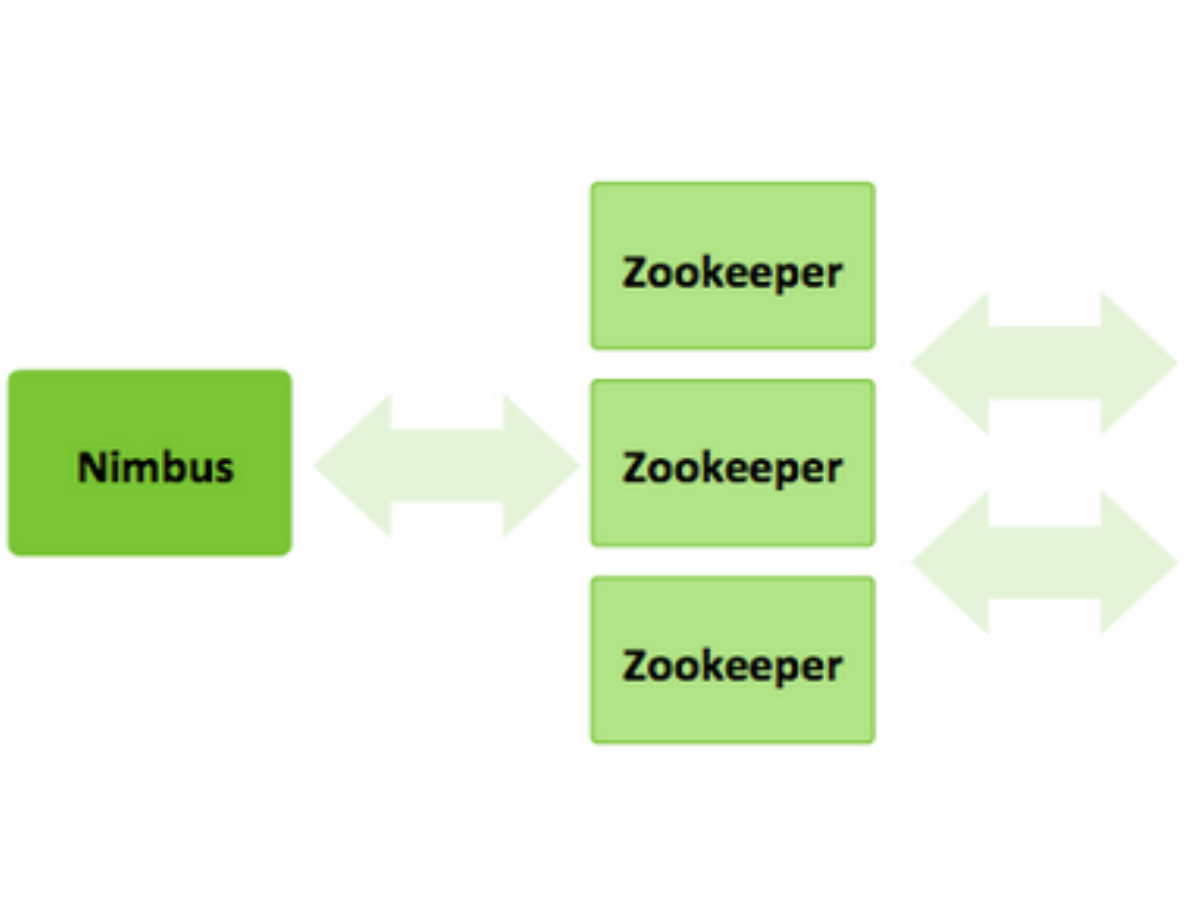
Grouping type	What it does	When to use
Shuffle Grouping	Sends tuple to a bolt in random round robin sequence	- Doing atomic operations eg.math operations
Fields Grouping	Sends tuples to a bolt based on or more field's in the tuple	- Segmentation of the incoming stream - Counting tuples of a certain type
All grouping	Sends a single copy of each bolt to all instances of a receiving bolt	- Send some signal to all bolts like clear cache or refresh state etc. - Send ticker tuple to signal bolts to save state etc.
Custom grouping	Implement your own field grouping so tuples are routed based on custom logic	- Used to get max flexibility to change processing sequence, logic etc. based on different factors like data types, load, seasonality etc.
Direct grouping	Source decides which bolt will receive tuple	- Depends
Global grouping	Global Grouping sends tuples generated by all instances of the source to a single target instance (specifically, the task with lowest ID)	- Global counts..

Hortonworks

Data Processing Guarantees in Storm

Processing guarantee	How is it achieved?	Applicable use cases
At least once	Replay tuples on failure	- Unordered idempotent operations - Sub second latency processing
Exactly once	Trident	- Need ordered processing - Global counts - Context aware processing - Causality based - Latency not important

Architecture



Nimbus(Management server)

- Similar to job tracker
- Distributes code around cluster
- Assigns tasks
- Handles failures

Supervisor(slave nodes):

- Similar to task tracker
- Run bolts and spouts as ‘tasks’

Zookeeper:

- Cluster co-ordination
- Nimbus HA
- Stores cluster metrics
- Consumption related metadata for Trident topologies

Hortonworks

Parallelism in a Storm Topology

Supervisor

- The supervisor listens for work assigned to its machine and starts and stops worker processes as necessary based on what Nimbus has assigned to it.

Worker (JVM Process)

- Is a JVM process that executes a subset of a topology
- Belongs to a specific topology and may run one or more executors (threads) for one or more components (spouts or bolts) of this topology.

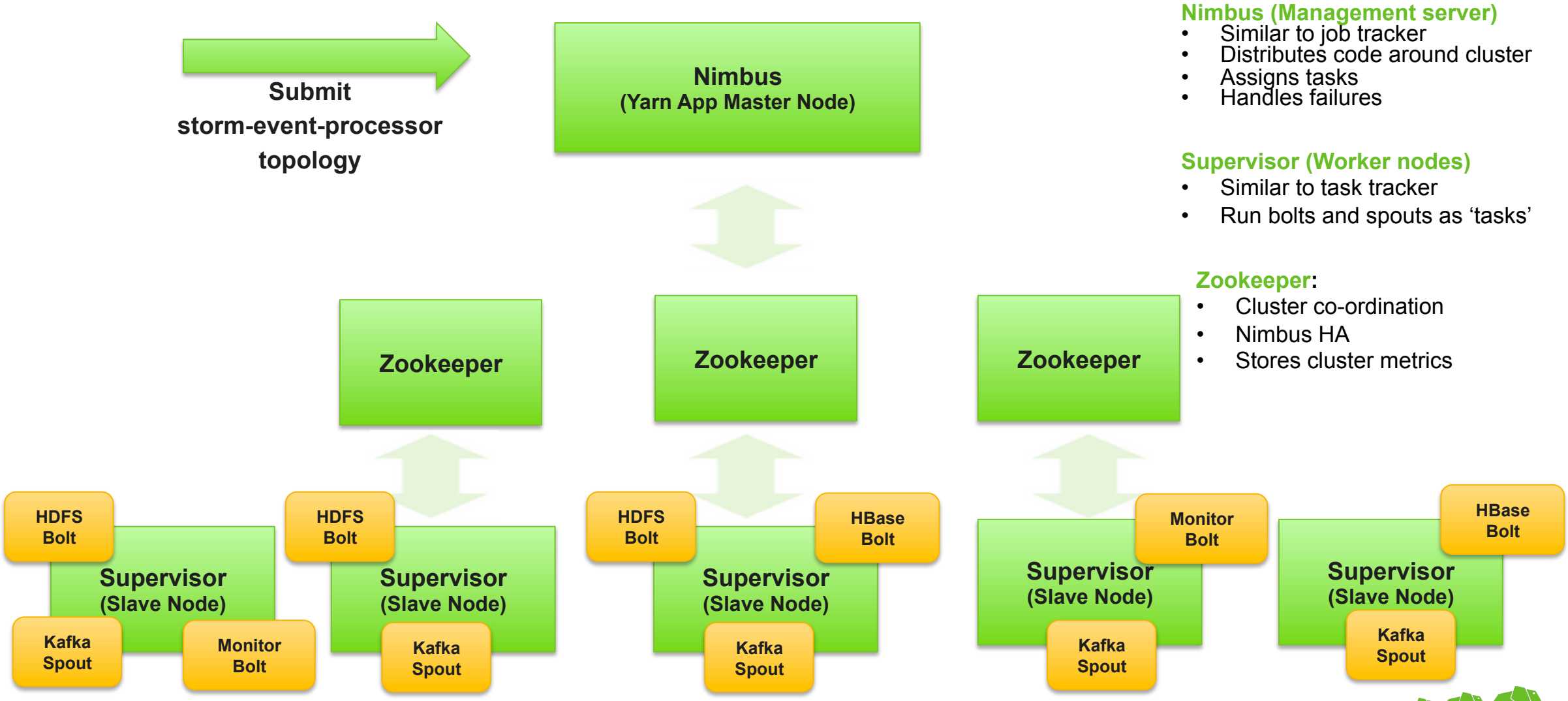
Executors (Thread in JVM Process)

- Is a thread that is spawned by a worker process and runs within the worker's JVM
- Runs one or more tasks for the same component (spout or bolt)

Tasks

- Performs the actual data processing and is run within its parent executor's thread of execution

Storm Components and Topology Submission



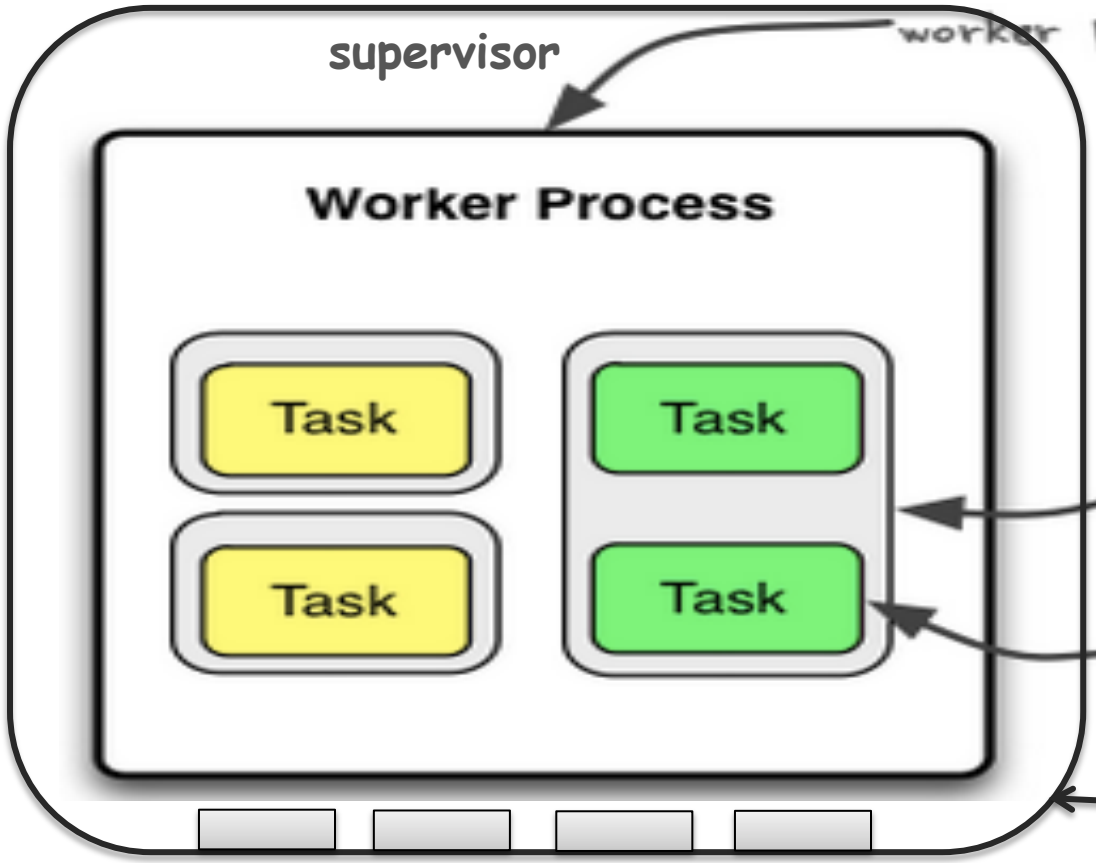
Nimbus (Management server)
- Similar to job tracker
- Distributes code around cluster
- Assigns tasks
- Handles failures

Supervisor (Worker nodes)
- Similar to task tracker
- Run bolts and spouts as 'tasks'

Zookeeper:
- Cluster co-ordination
- Nimbus HA
- Stores cluster metrics

Relationship Between Supervisors, Workers, Executors & Tasks

A machine in a Storm cluster may run one or more worker processes for one or more topologies. Each worker process runs executors for a specific topology.



One or more executors may run within a single worker process, with each executor being a thread spawned by the worker process. Each executor runs one or more tasks of the same component (spout or bolt).

A task performs the actual data processing.

Each supervisor machine in storm has specific Predefined ports to which a worker process is assigned

Source: http://www.michael-noll.com/blog/2012/10/16/understanding-the-parallelism-of-a-storm-topology/

Hortonworks

Parallelism Example

Example Configuration in Use Case:

- Supervisors = 4
- Workers = 8
- spout.count = 5
- bolt.count = 16

How many worker threads per supervisor?

- 8/4 = 2

How many total executors(threads)?

- 16 + 5 = 21

How many executors(threads) per worker?

- 21/8 = 2 to 3

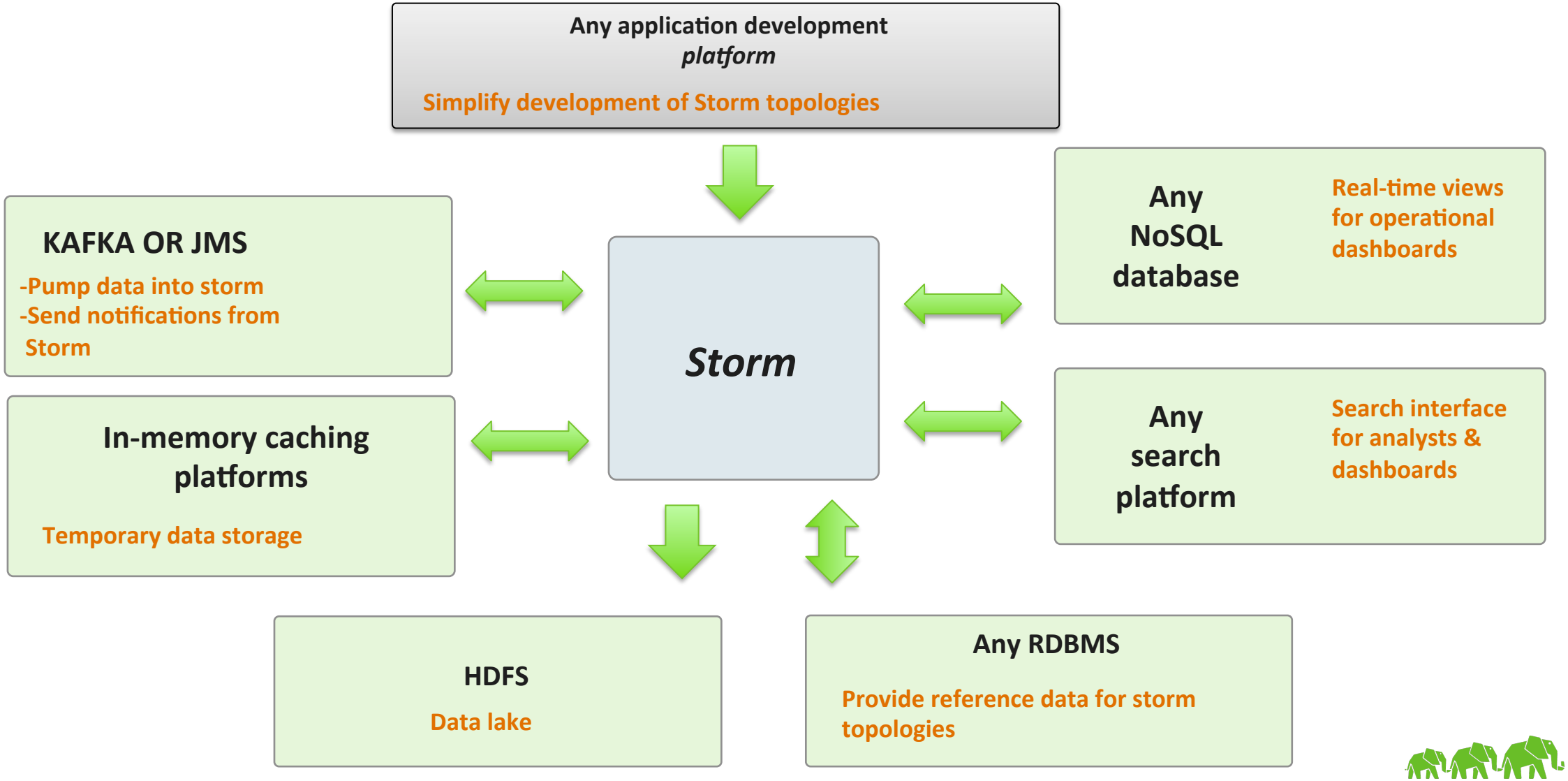
Hortonworks

Hortonworks

Apache Storm
Data Flows and Architecture

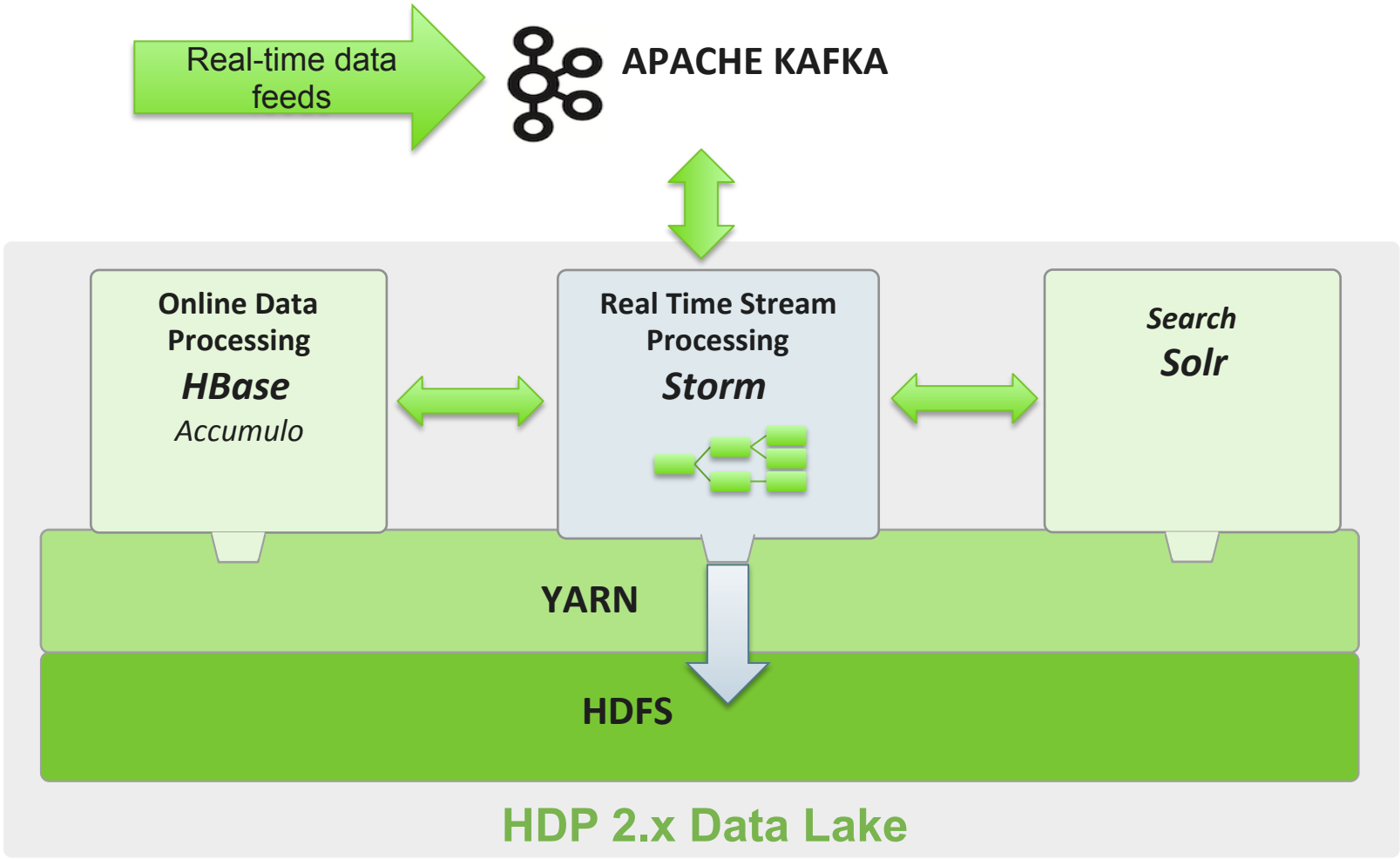
Hortonworks. We do Hadoop.

Flexible Architecture with Apache Storm



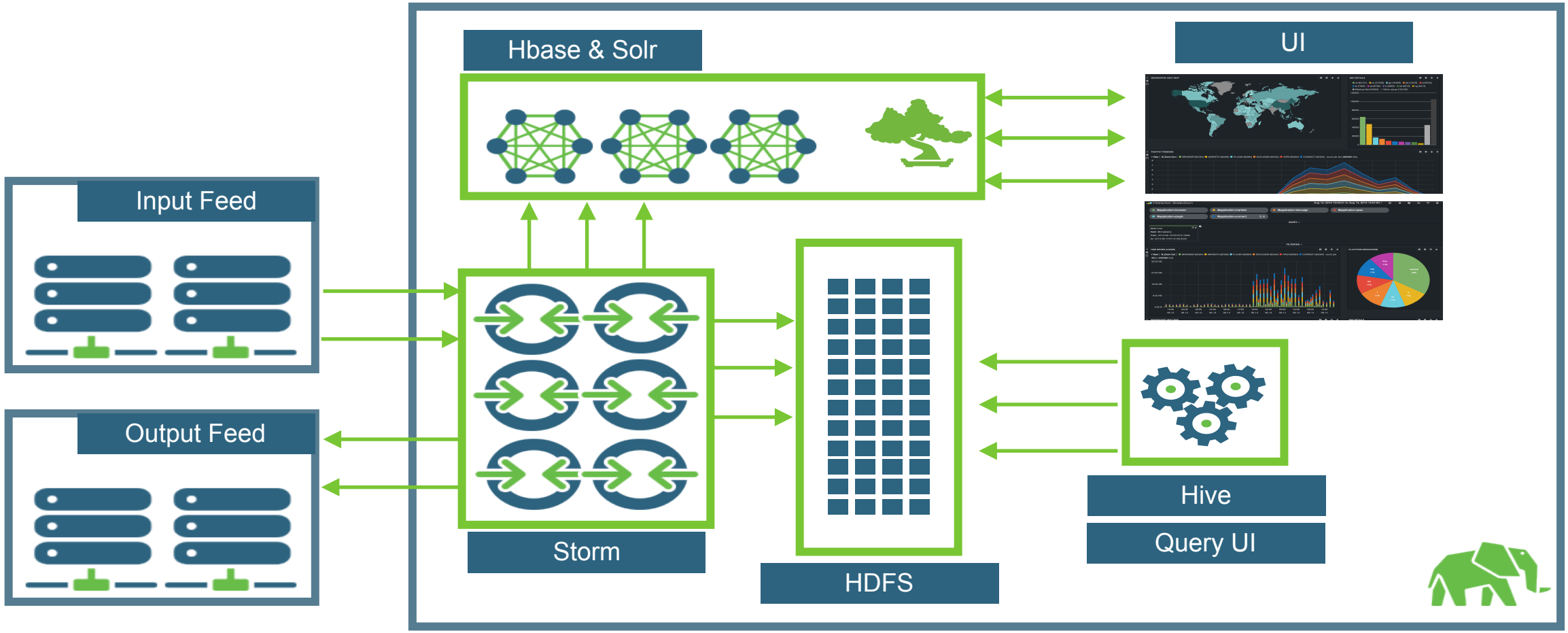
Hortonworks

An Apache Storm Solution Architecture Example



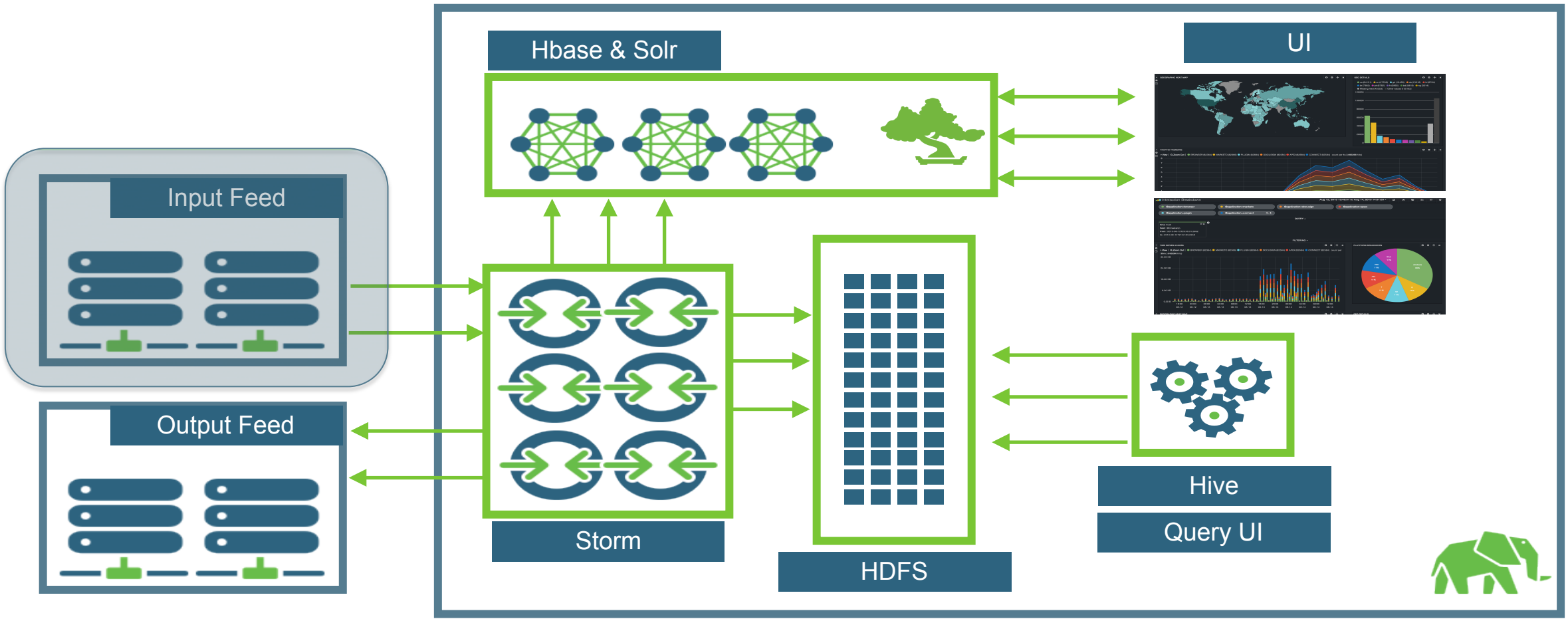
Hortonworks

HPD Data Flow and Architecture



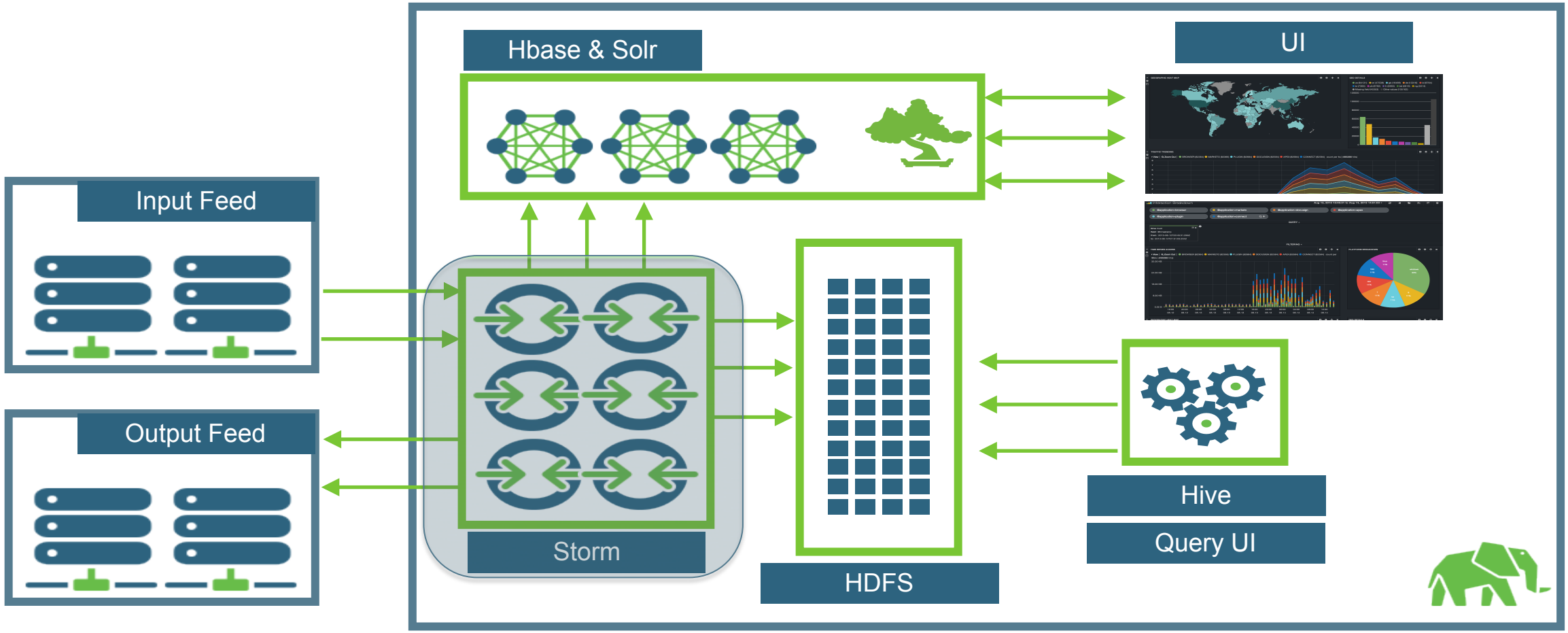
Hortonworks

HDP Data Flow and Architecture



Hortonworks

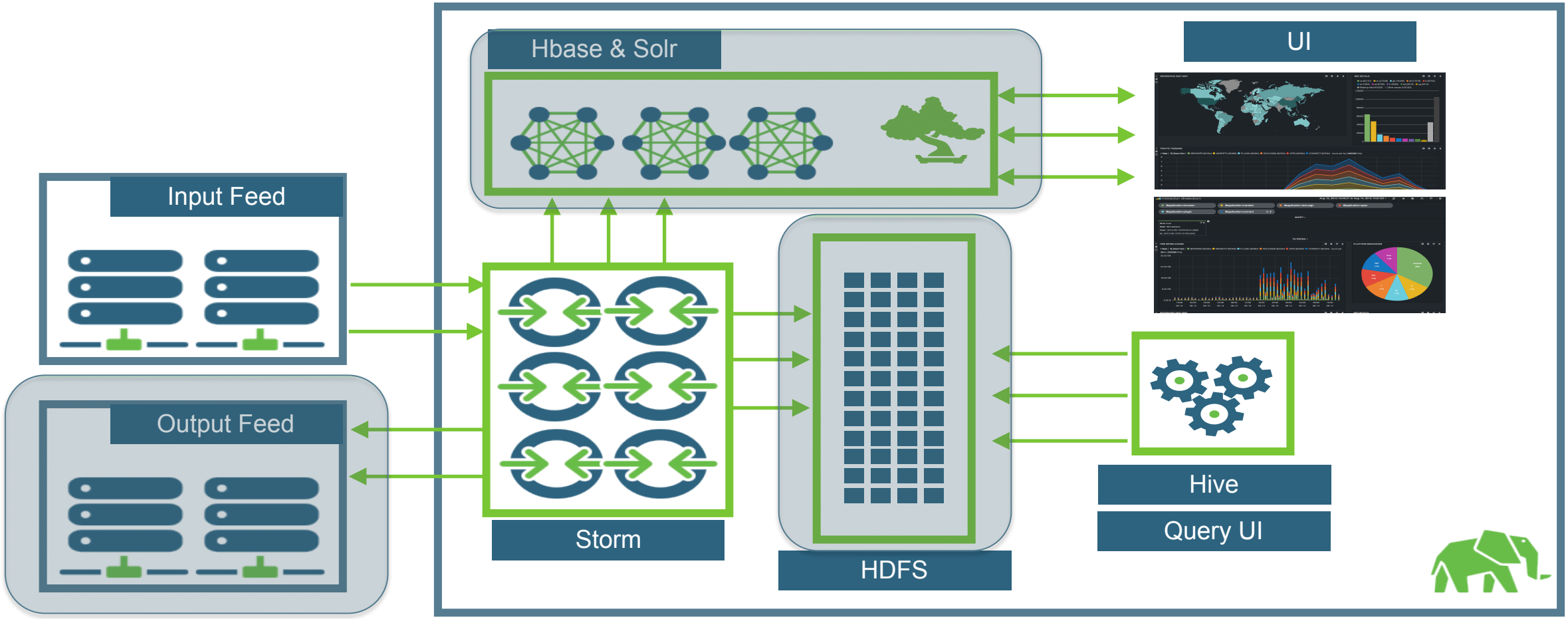
HDP Data Flow and Architecture



Ingest and Process Stream (Normalization/Micro ETL) in Real Time Using Storm

Hortonworks

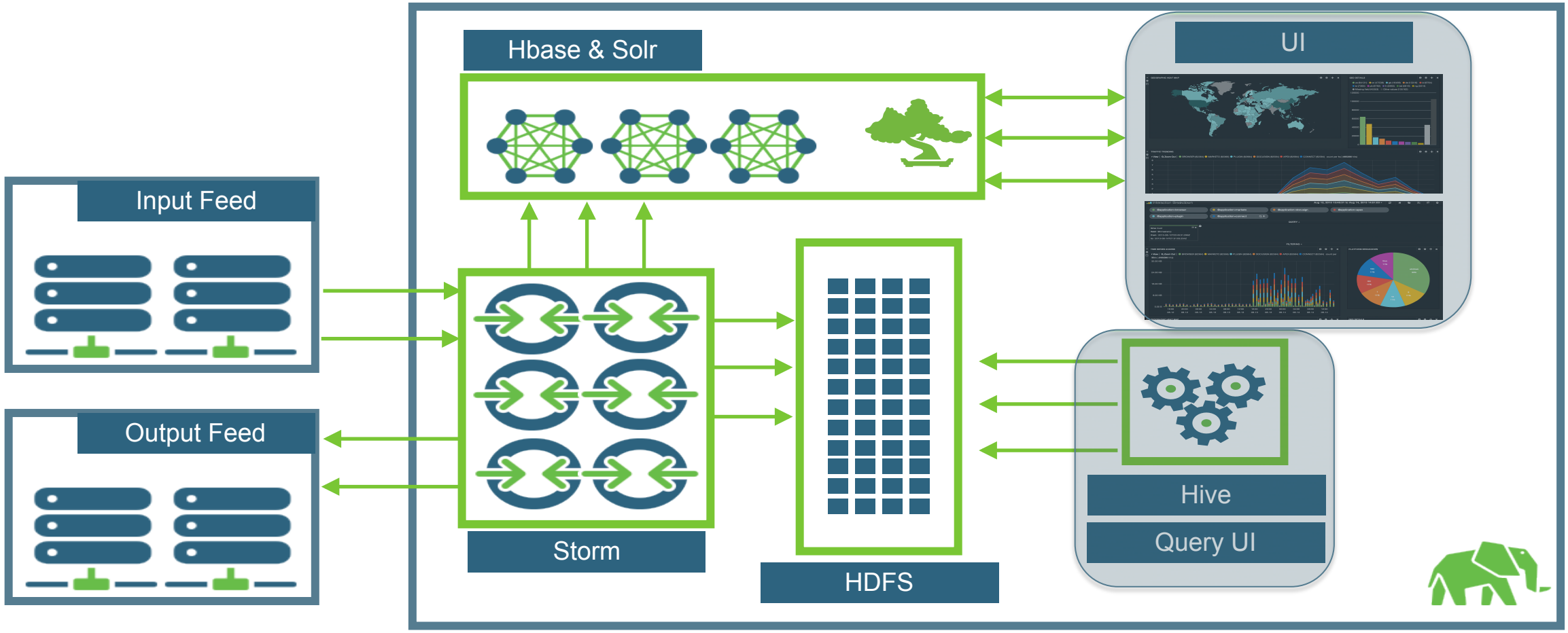
HDP Data Flow and Architecture



Publish to Output Feed (Available via RESTful API)
Push to Hbase & Solr and Sink to HDFS/HCatalog (Meta Model) for Ad-Hoc SQL

Hortonworks

HDP Data Flow and Architecture



Data Available for Search and Query (RESTful API Calls an Option as Well)

Hortonworks

Apache Storm – Code Walkthrough

Hortonworks. We do Hadoop.

Automated Stock Insight

Monitor Tweet Volume For Early Detection of Stock Movement

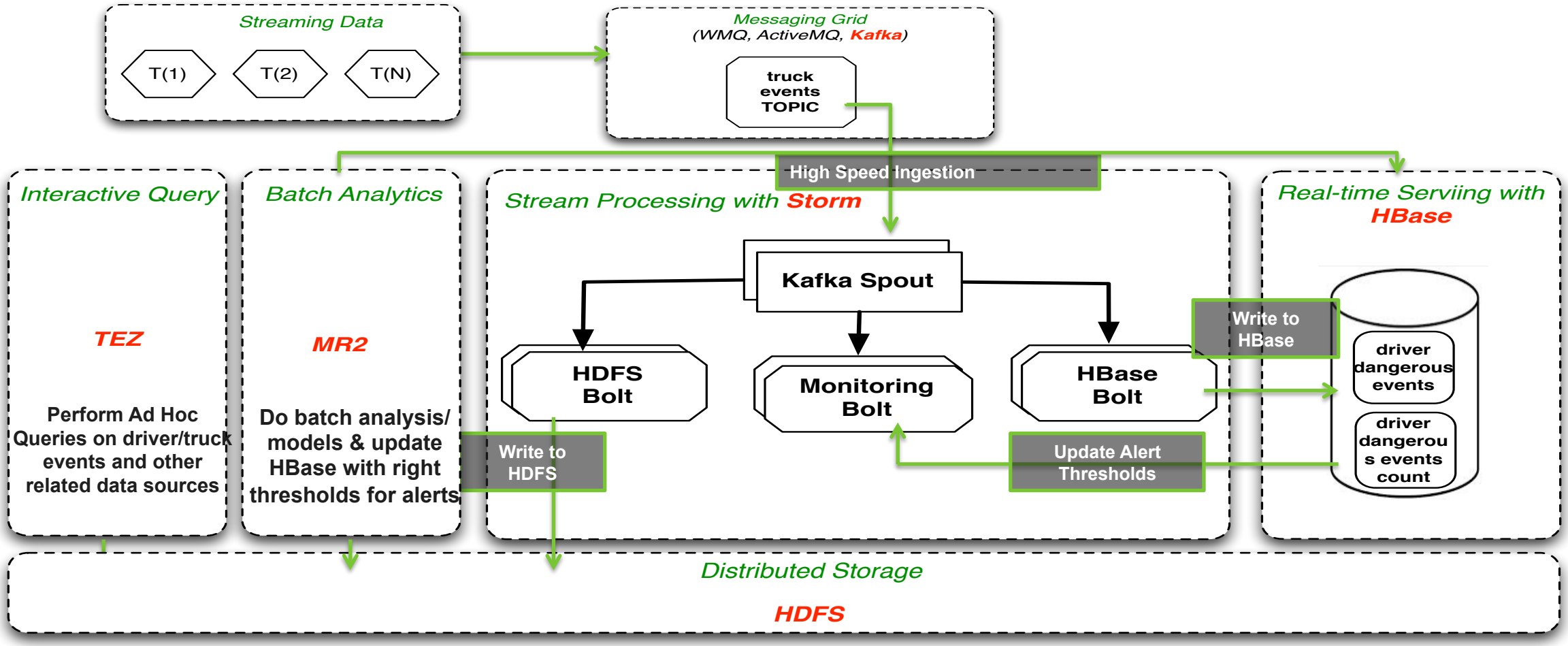
- Monitor Twitter stream for S&P 500 companies to identify & act on unexpected increase in tweet volume

Solution components

- **Ingest**:
  - Listen for Twitter streams related to S&P 500 companies
- **Processing**
  - Monitor tweets for unexpected volume
  - Volume thresholds managed in HBASE
- **Persistence**
  - HBase & HDFS
- **Refine**
  - Update threshold values based on historical analysis of tweet volumes

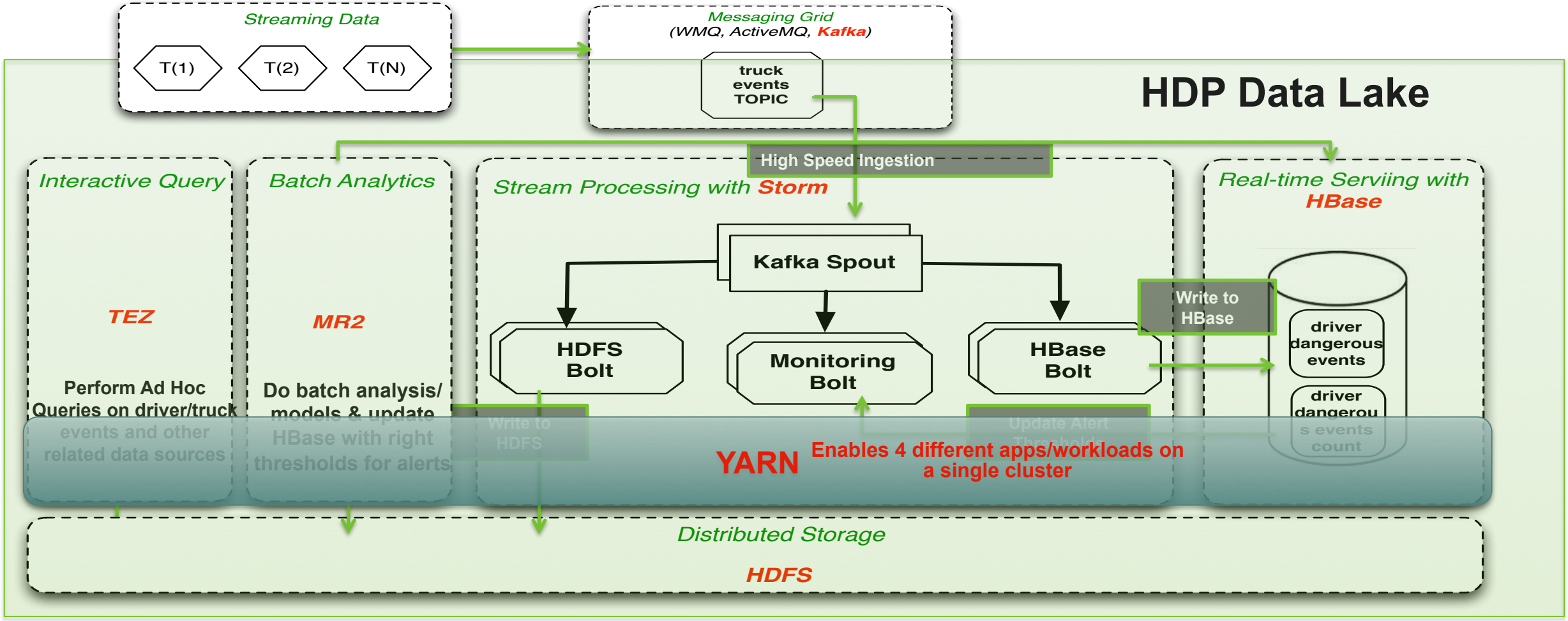
Hortonworks

High Level Architecture



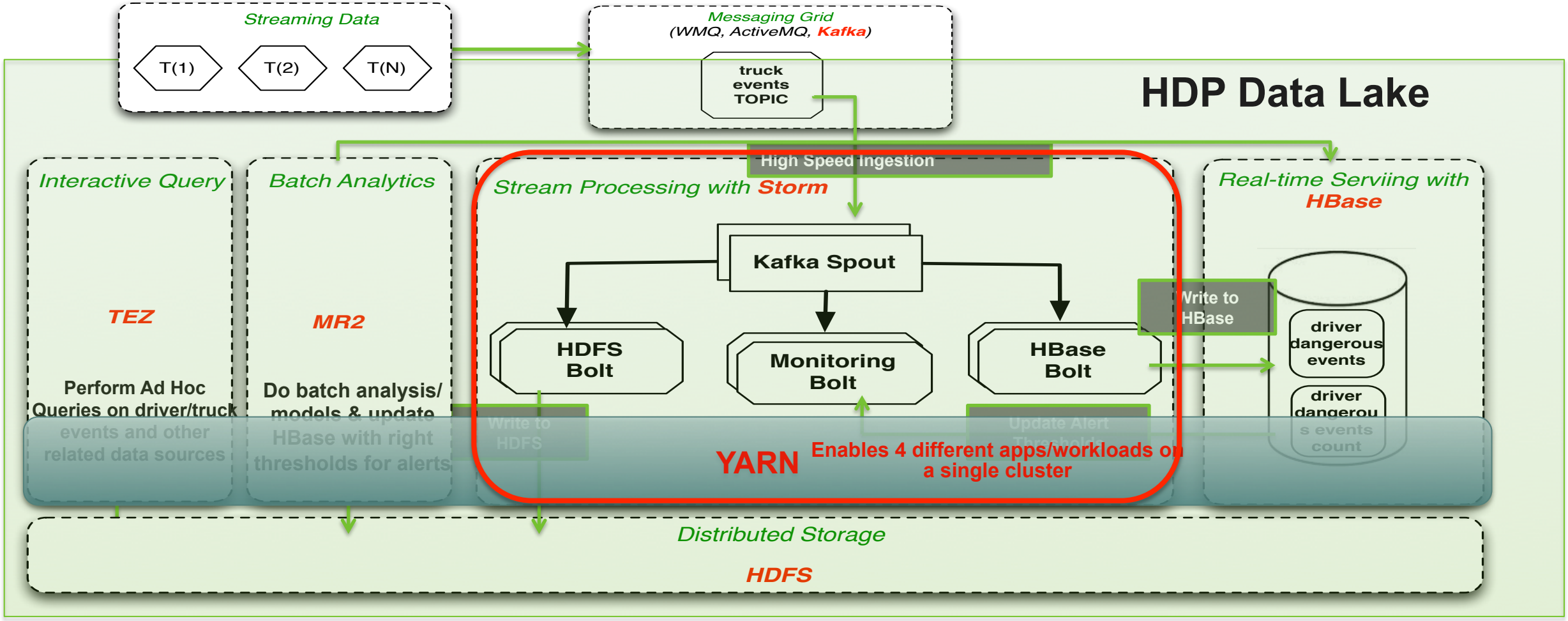
Hortonworks

HDP Provides a Single Data Platform



Hortonworks

HDP Provides a Single Data Platform



Hortonworks

Hortonworks

Q&A