Big Data Primer For IT Professional

Sujee Maniyam
Founder / Principal @ Elephant Scale

SNIA Legal Notice

- The material contained in this tutorial is copyrighted by the SNIA unless otherwise noted.
- Member companies and individual members may use this material in presentations and literature under the following conditions:
  - Any slide or slides used must be reproduced in their entirety without modification
  - The SNIA must be acknowledged as the source of any material used in the body of any document containing material from these presentations.
- This presentation is a project of the SNIA Education Committee.
- Neither the author nor the presenter is an attorney and nothing in this presentation is intended to be, or should be construed as legal advice or an opinion of counsel. If you need legal advice or a legal opinion please contact your attorney.
- The information presented herein represents the author's personal opinion and current understanding of the relevant issues involved. The author, the presenter, and the SNIA do not assume any responsibility or liability for damages arising out of any reliance on or use of this information.

NO WARRANTIES, EXPRESS OR IMPLIED. USE AT YOUR OWN RISK.

Abstract

- Big Data Primer For IT Professionals
  - This session will highlight some Big Data technologies that an aspiring Big Data developers should learn.
    This talk will appeal to developers / engineers who want to learn Big Data technologies.

A look at Big data eco system

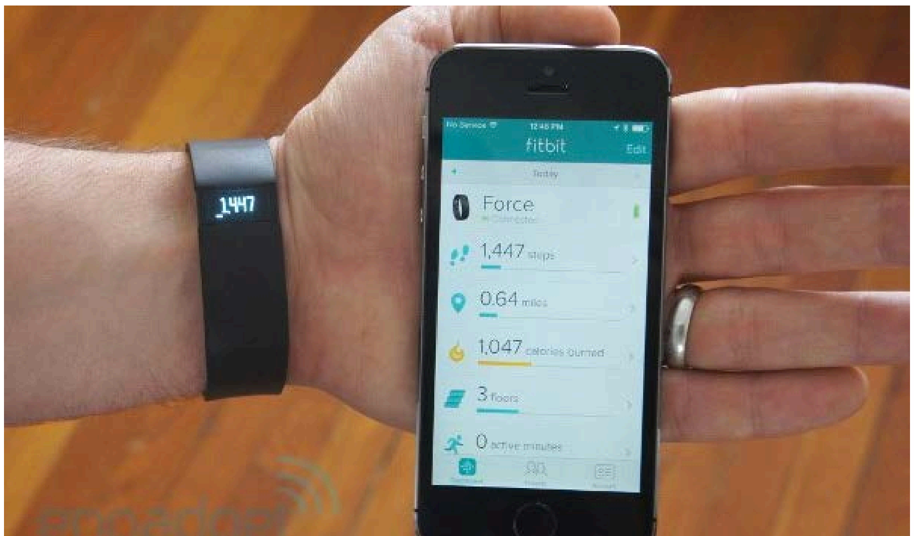
SNIA Global Education

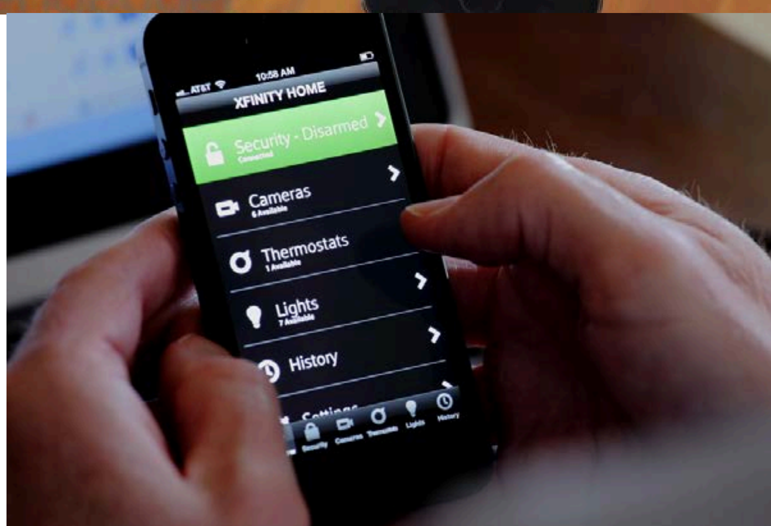
Source : datafloq.com

Which One ?

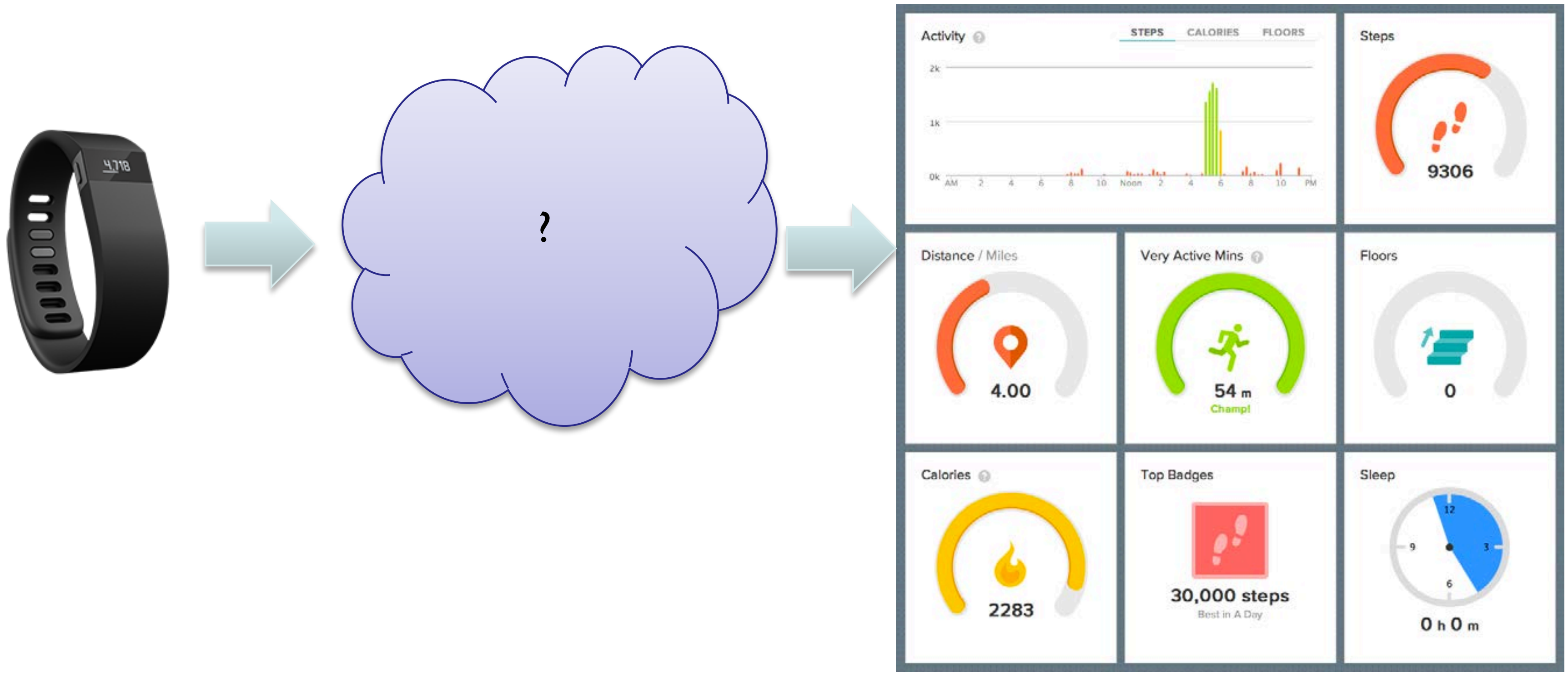
- Let’s take ‘design driven approach’

Internet of Things – A reality





Data infrastructure



Data Volume ?
A Napkin calculation

- Say we have
- Million sensors
- Each sensor reports every minute
- data size 1KB

- This will result in :
- 1.44 Billions events / day !
- 1.44 TB / day !!

Sensor Data : Texas utilities smart meter data

Texas Smart Meter Projections

variables	**description**				
sensors	10 million customers	1.00E+07	10 million		
signal frequency	every 15 mins	900	secs		
event size	1.4 K	1400	bytes		
events per day per sensor		**96**			
total events per day		**9.60E+08**	**960**	**millions**	0.96 billion
total events / sec		**1.11E+04**			11,111.11
total data size per day		**1.34E+12**	**1344**	GB	1.344 TB

Data Velocity

- Say we have
- Million sensors
- Each sensor reports every minute
- data size 1KB

- →
- Millions events / minute
- ~ 17,000 events / sec

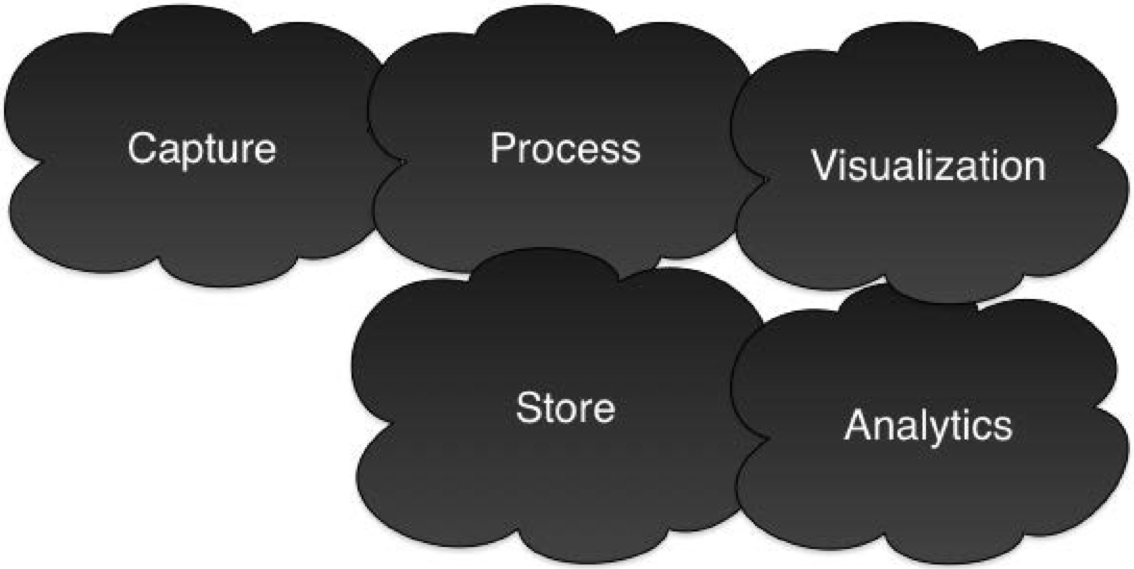
Data processing speed

- Need (near) real time processing most of the time
  - E.g. Need to alert if temperature suddenly spikes

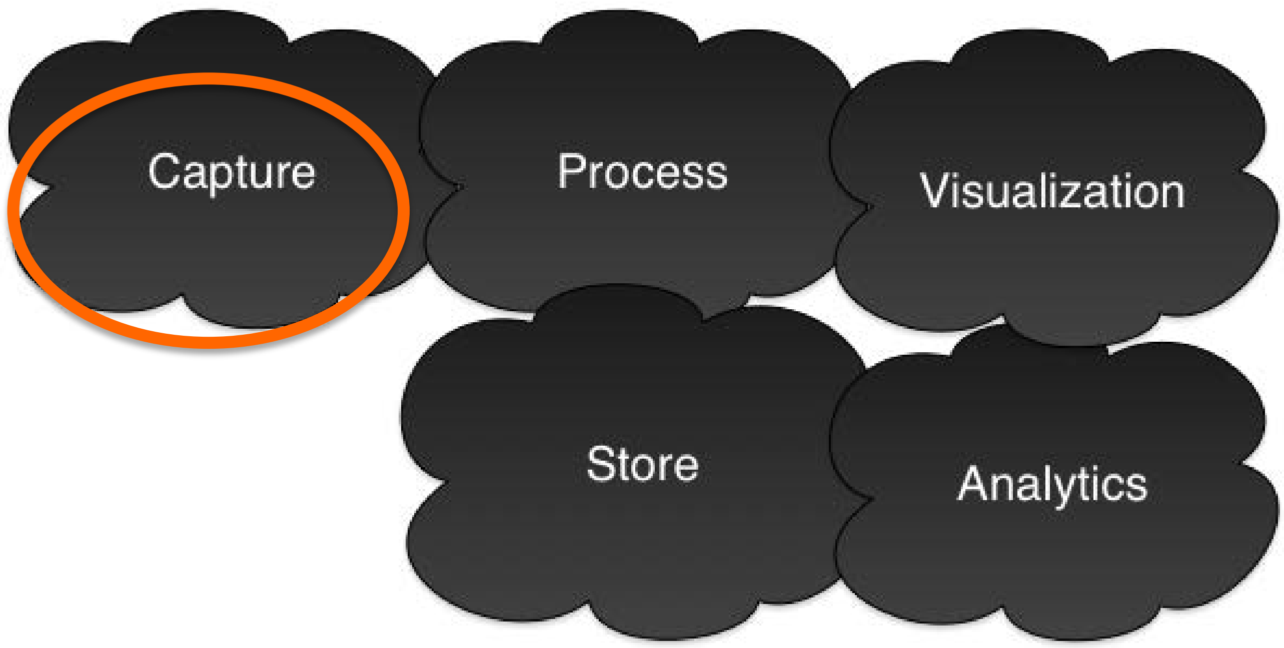
Challenge = big data + real time

- Don’t loose events !
  - Any event could be important
  - Most events are mundane (e.g. temperature stays between 68’F – 72’ F)
- Process them in near real time
- Store the events for a long time
  - Audit
  - Diagnose
- Support various queries
  - Real time (what is the latest temperature for sensor id 123?)
  - Aggregate (what is the avg. temp in zipcode 12345)

High Level Architecture



Capture



(1) Capture
Requirements

- Requirements:
- Capture events coming at high speed
  - Tens of thousands events / sec (some times millions / sec)
- Don't loose events
  - Tolerate hardware / software failure
  - Tolerate intermittent connectivity issues
- Scale 'easily'

(1) Capture
Choices

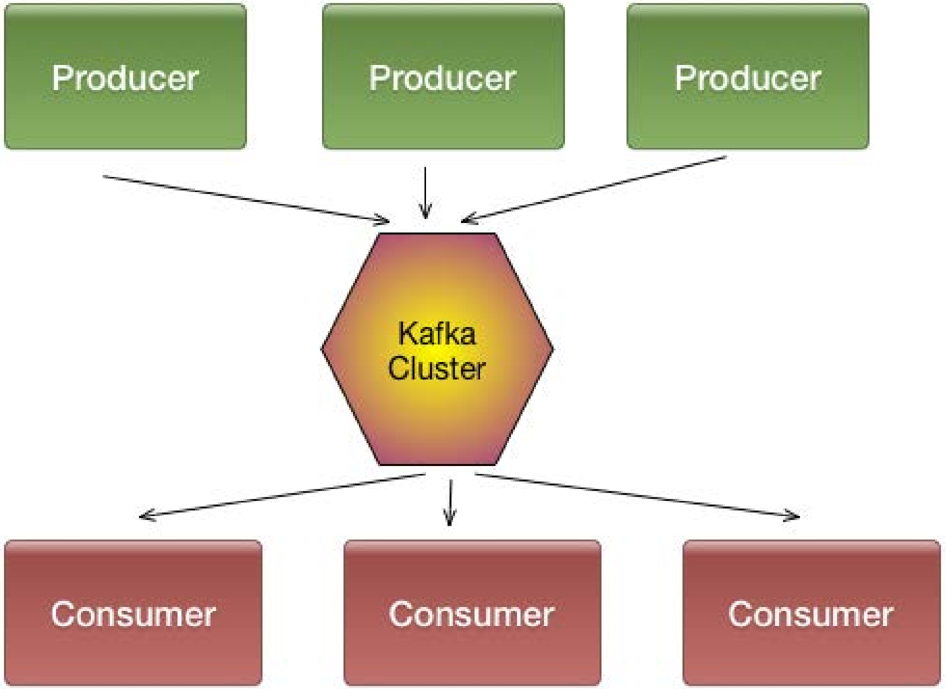
- MQ (RabbitMQ ..etc)
  - Good adoption in enterprises / durable
- FluentD
  - Data collector for various sources
- Flume
  - Part of Hadoop eco system
  - Good for collecting logs from many sources
- AWS Kinesis
  - Queue system in Amazon Cloud
- Kafka
  - Distributed queue

(1) Capture
Meet kafka

- Apache Kafka is a distributed messaging system
- Came out of LinkedIn… open sourced in 2011
- Built to tolerate hardware / software / network failures
- Built for high throughput and scale
  - LinkedIn : 220 Billion messages / day
  - At peak : 3+ million messages / sec

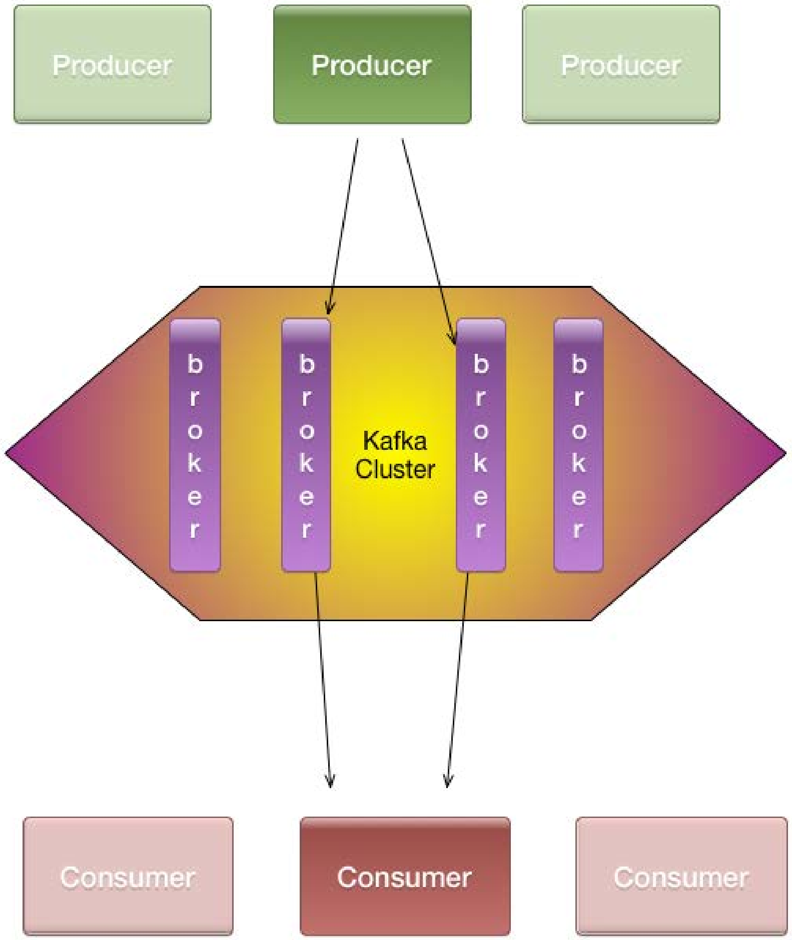
(1) Capture
Kafka architecture

- Publisher - subscriber / producer – consumer model

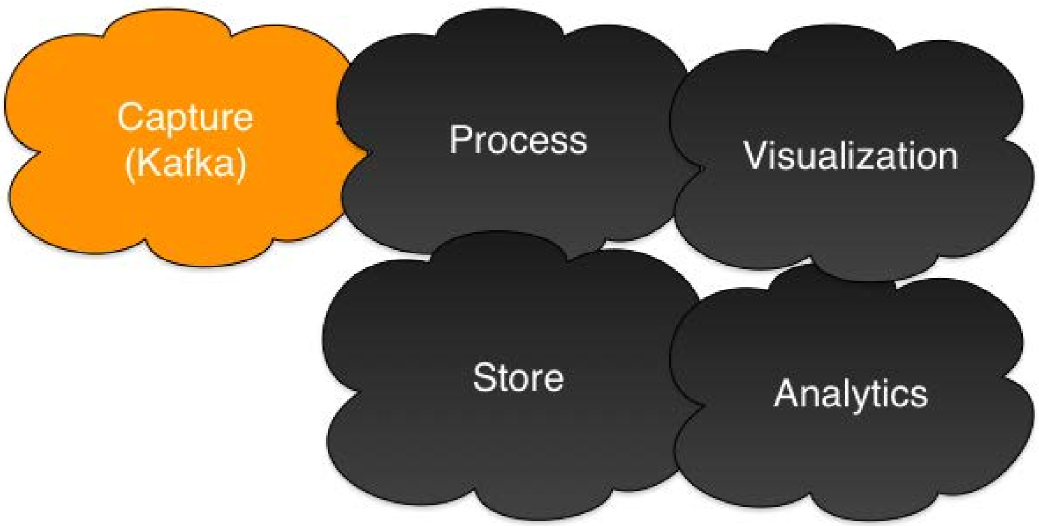


(1) Capture
Kafka architecture

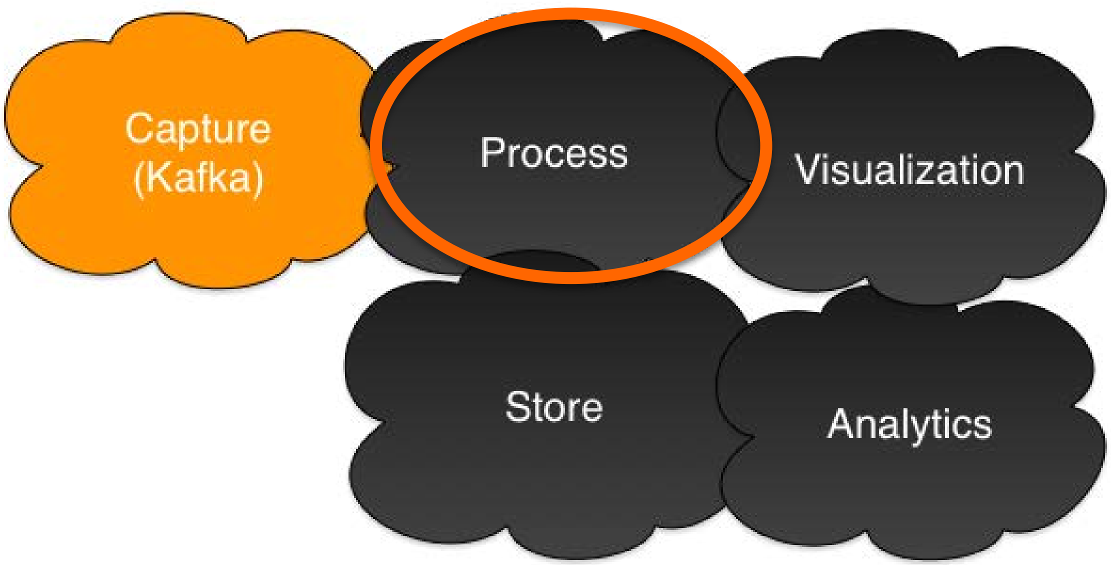
- Producers write data to brokers
- Consumers read data from brokers
- All of this is distributed / parallel
- Failure tolerant
- Data is stored as topics
  - “sensor_data”
  - “alerts”
  - “emails”



Capture



Next : (2) Processing



(2) Processing requirements

- Process events in real time or near real time
- High velocity
  - Tens of thousands → millions of events / sec
- Guaranteed processing
  - Process an event at-least-once
  - Exactly-once (harder to achieve)
- Failure tolerant
- Scale 'easily'

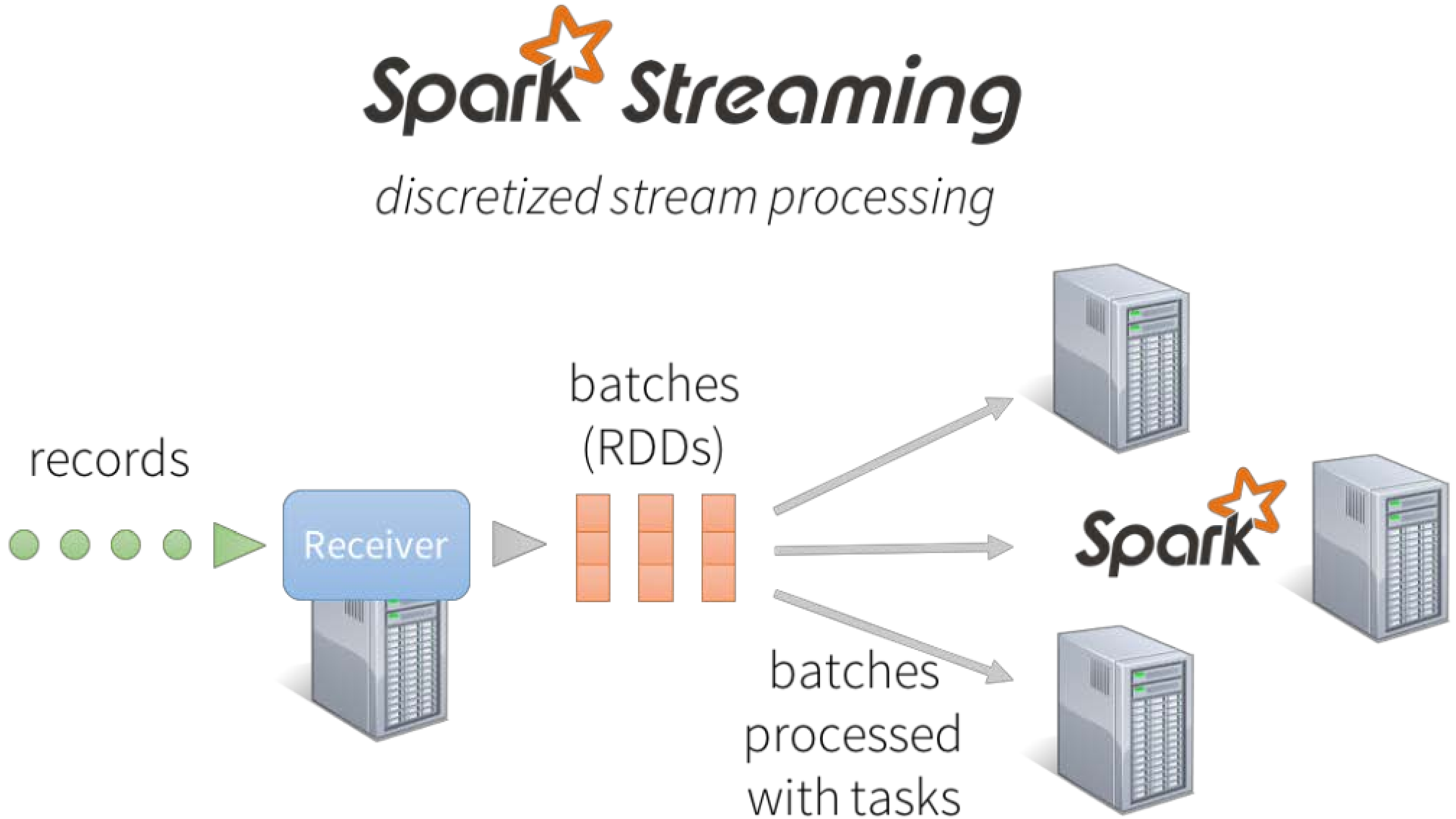
(2) Processing
Choices

- Storm
  - ’Original’ stream processing
- Apache Samza
  - Stream processing framework based on Kafka + Hadoop YARN
- Apache NiFi
  - Data flow
- Flink
  - New framework
- Spark Streaming
  - Cool framework

Streaming Systems Feature Comparison

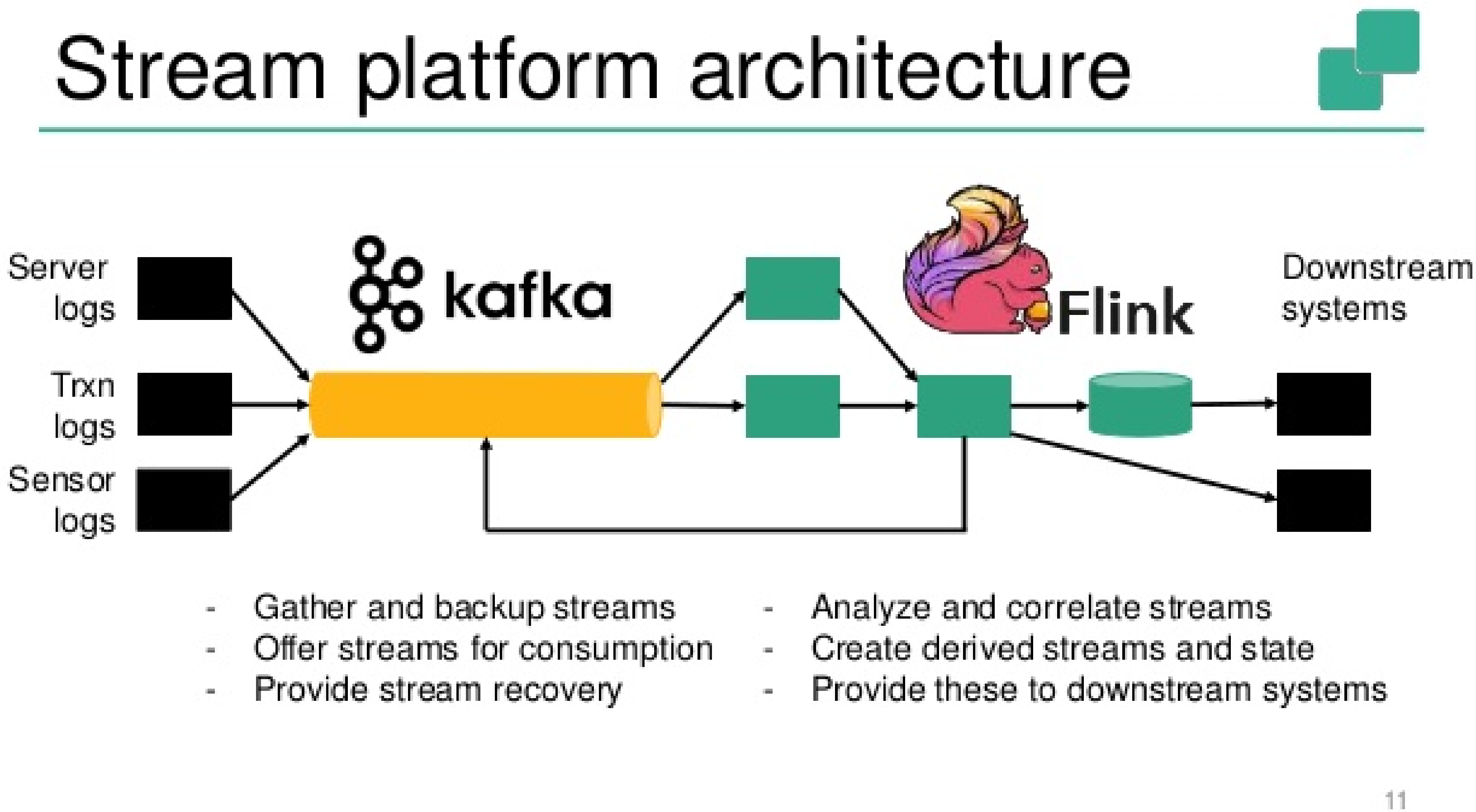
Feature	Storm	Spark Streaming	Flink	NiFi
Processing Model	Event based by default (micro batch using Trident)	Micro Batch	Event based + Micro Batch based	Event Based (?)
Windowing operations	Supported by Trident	Yes	Yes	?
Latency	Milliseconds	Seconds	Milliseconds	Milliseconds
At-least-once	YES	YES	YES	YES
At-most-once	YES	NO	YES	?
Exactly-once	YES with Trident	YES	YES	?

Spark Streaming Architecture

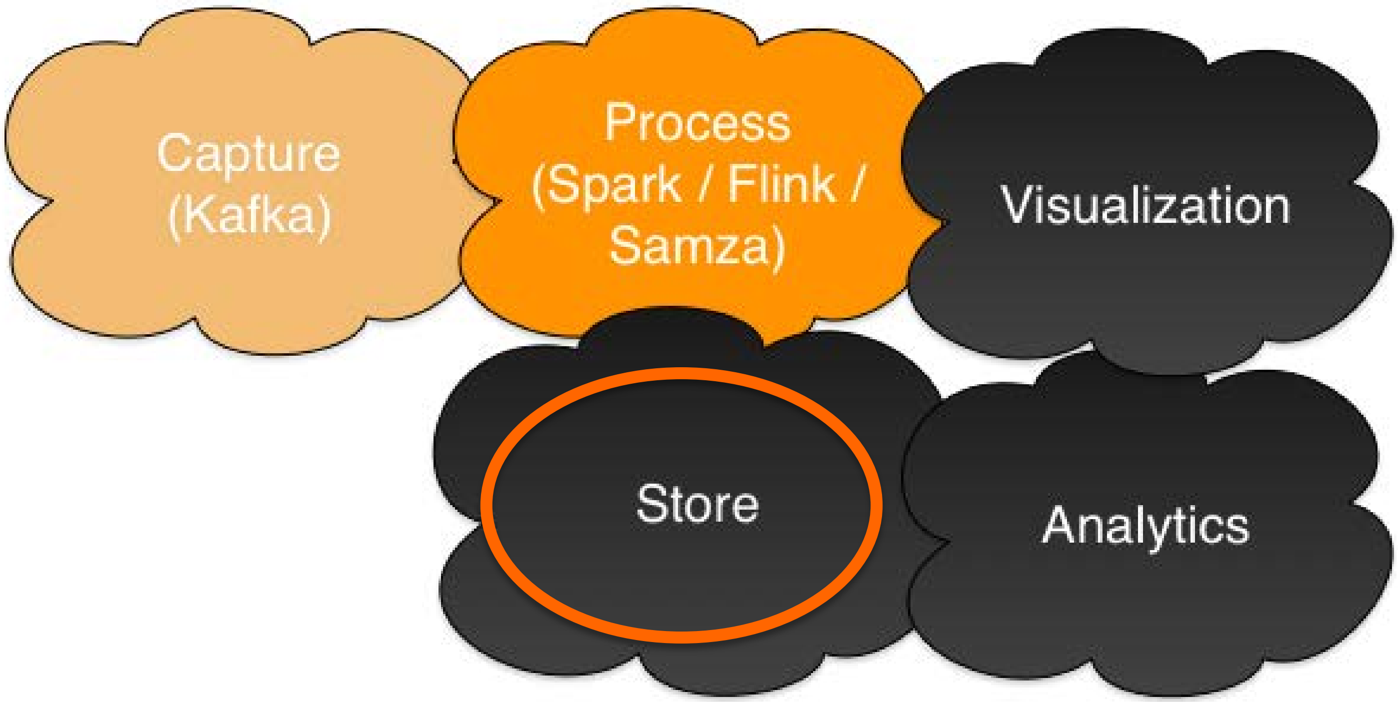


records processed in batches with short tasks
each batch is a RDD (partitioned dataset)

Flink Streaming Architecture



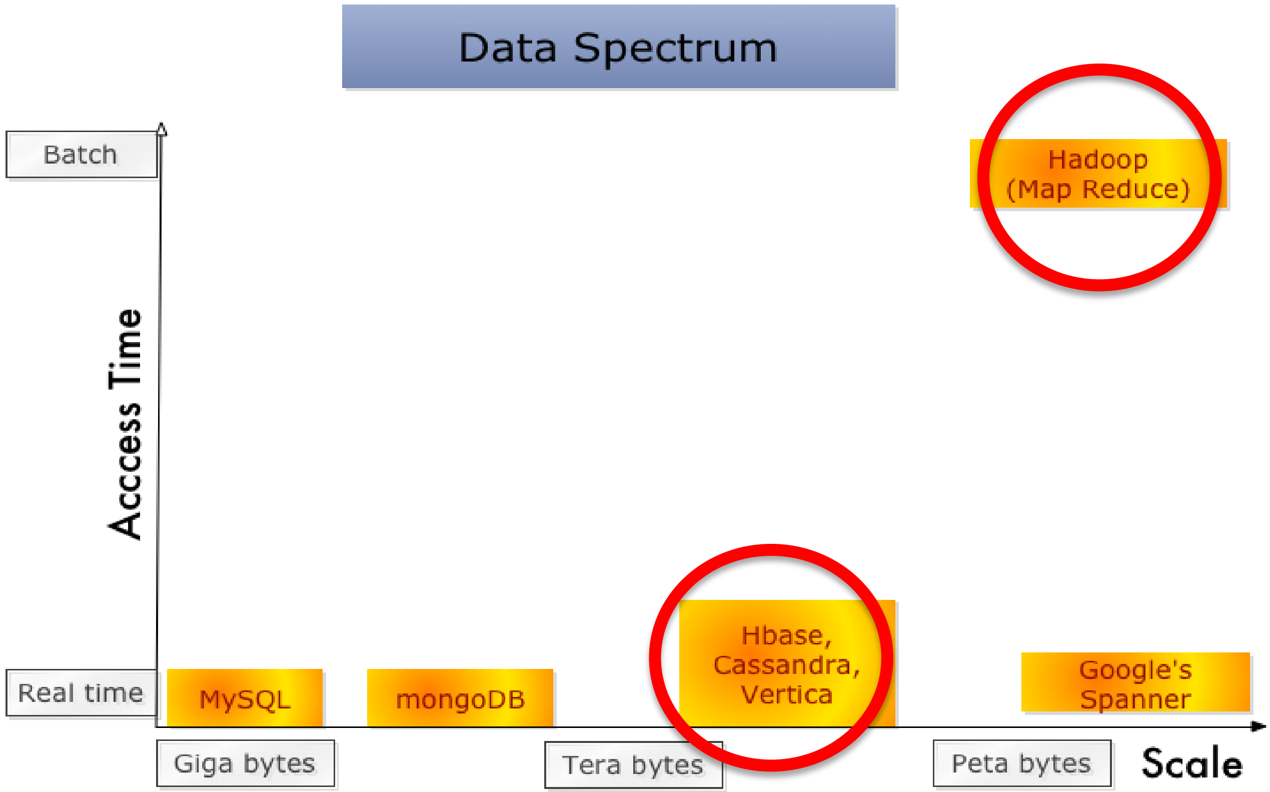
Stream Processing



(3) storage
Requirements

- Handle ‘Big Data’ (1 TB / day !)
- Traditional storages are not effective (or too expensive)
- Need two types of storage
  1. ‘forever’ storage
     - Store multi terabytes of data for a long periods
     - Support Batch queries
  2. ‘fast / real-time lookup’ storage
     - Query in real time (milliseconds)
       “what is the latest reading for sensor-123 ?”
     - Store latest / new data (e.g. last 3 months)
     - Flexible schema for semi-structured data
- Both need to scale

(3) Storage Requirements



adapted from : http://www.slideshare.net/medriscoll/driscoll-strata-buildingdatastartups25may2011clean

(3) storage
Choices

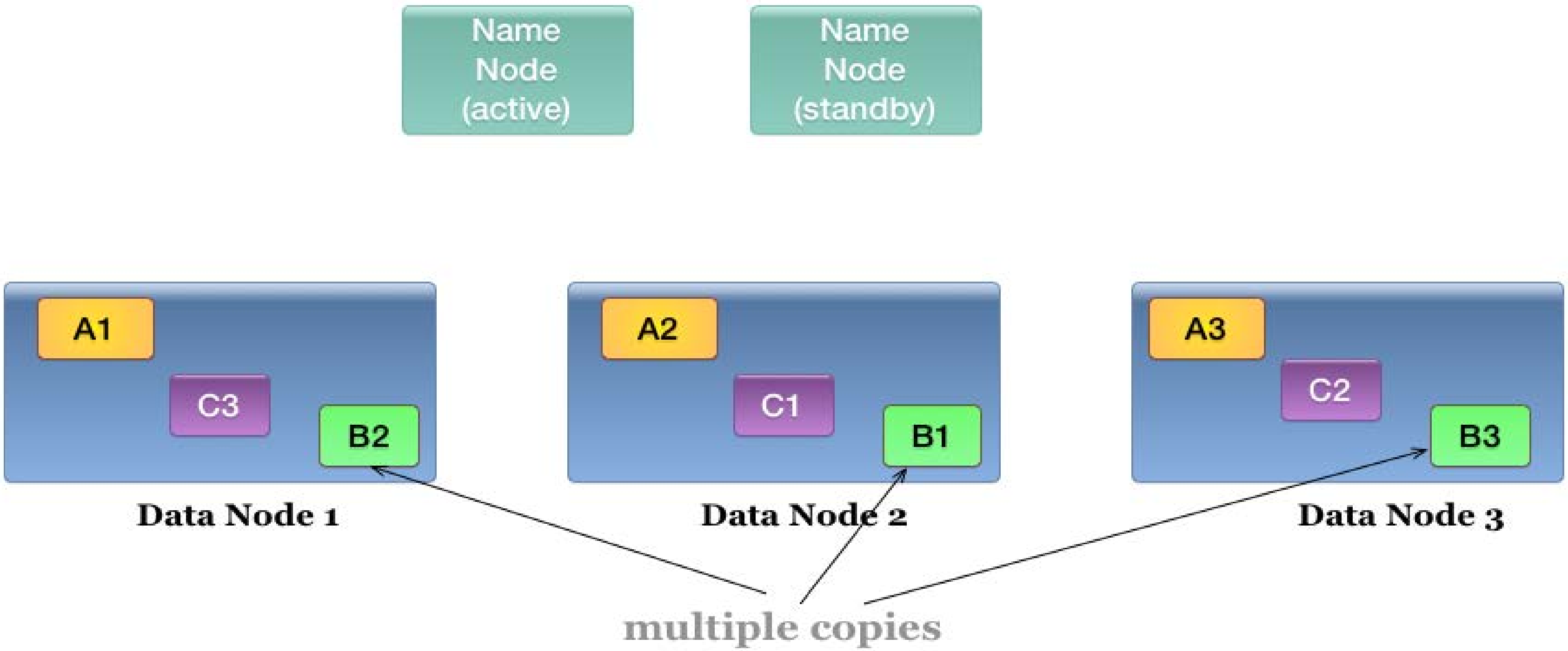
- ‘forever’ storage
  - Scalable distributed file systems
  - Hadoop ! (HDFS actually)

- ‘real time store’
  - Traditional RDBMS won’t work
    - Don’t scale well (or too expensive)
    - Rigid schema layout
  - NoSQL !

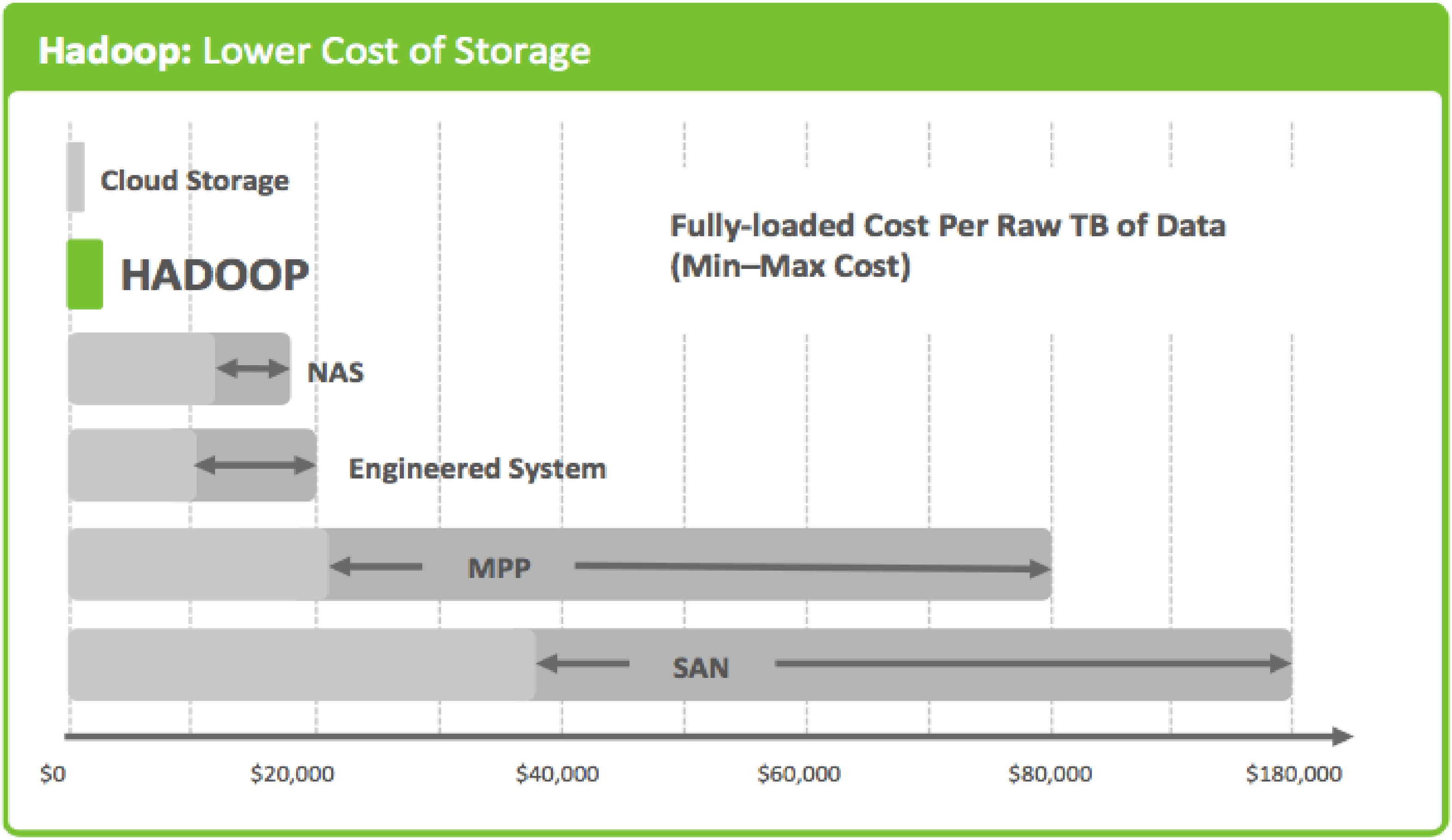
(3) Storage
HDFS (in 20 secs)

- Distributed file system
- Runs on commodity servers
  - → high ROI
- Can keep ticking even when nodes go down
  - → fault tolerant
- Replicates data to prevent data loss in case of node failures
  - → built in backup ☺
- Scales to Peta bytes (horizontal scalability)
- Proven in the field

(3) Storage
HDFS Architecture



(3) Storage
Cost of Big data



Source : hortonworks

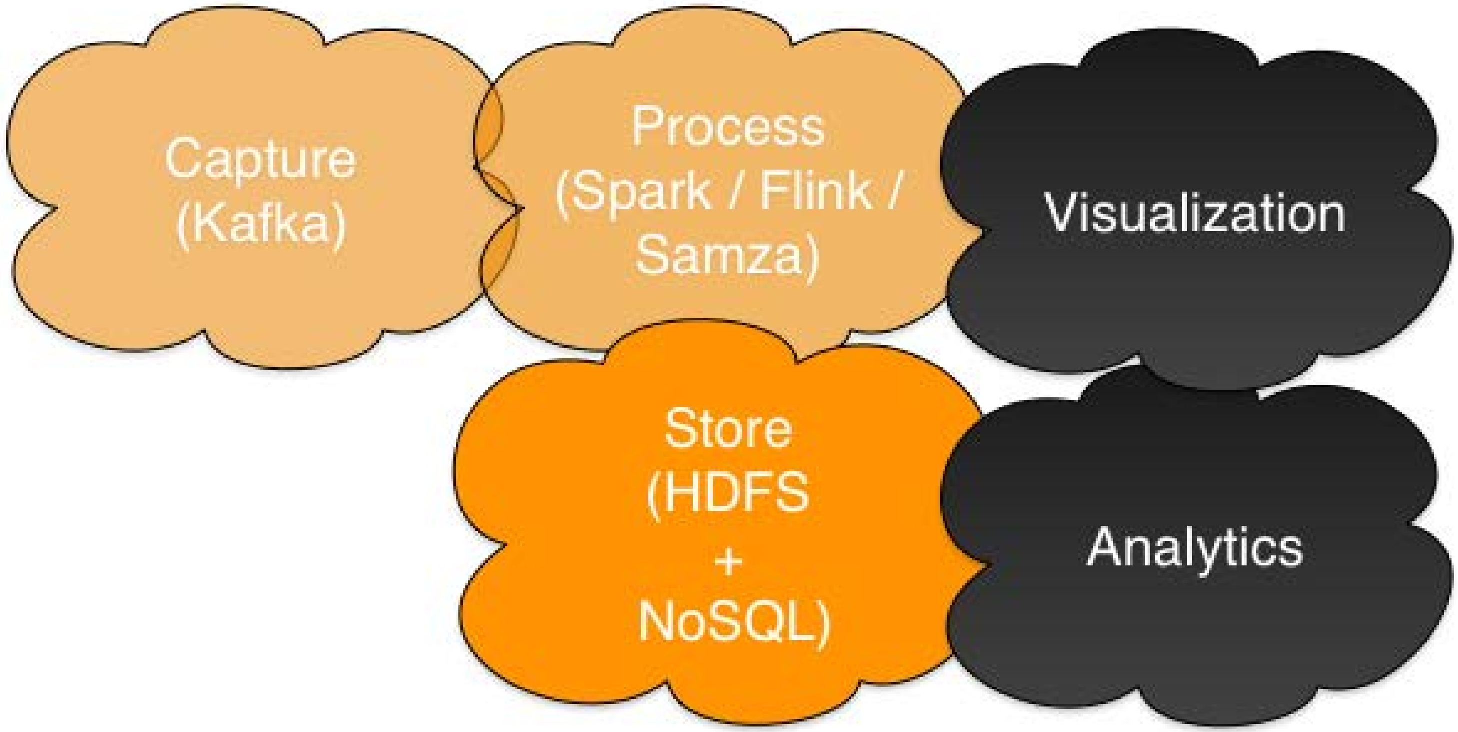
(3) Storage
HDFS

- Can handle big data
- Scales easily
- Cost effective
- “Source of Truth”
  - Files are immutable within HDFS (new data is ‘appended’)
  - Audit friendly

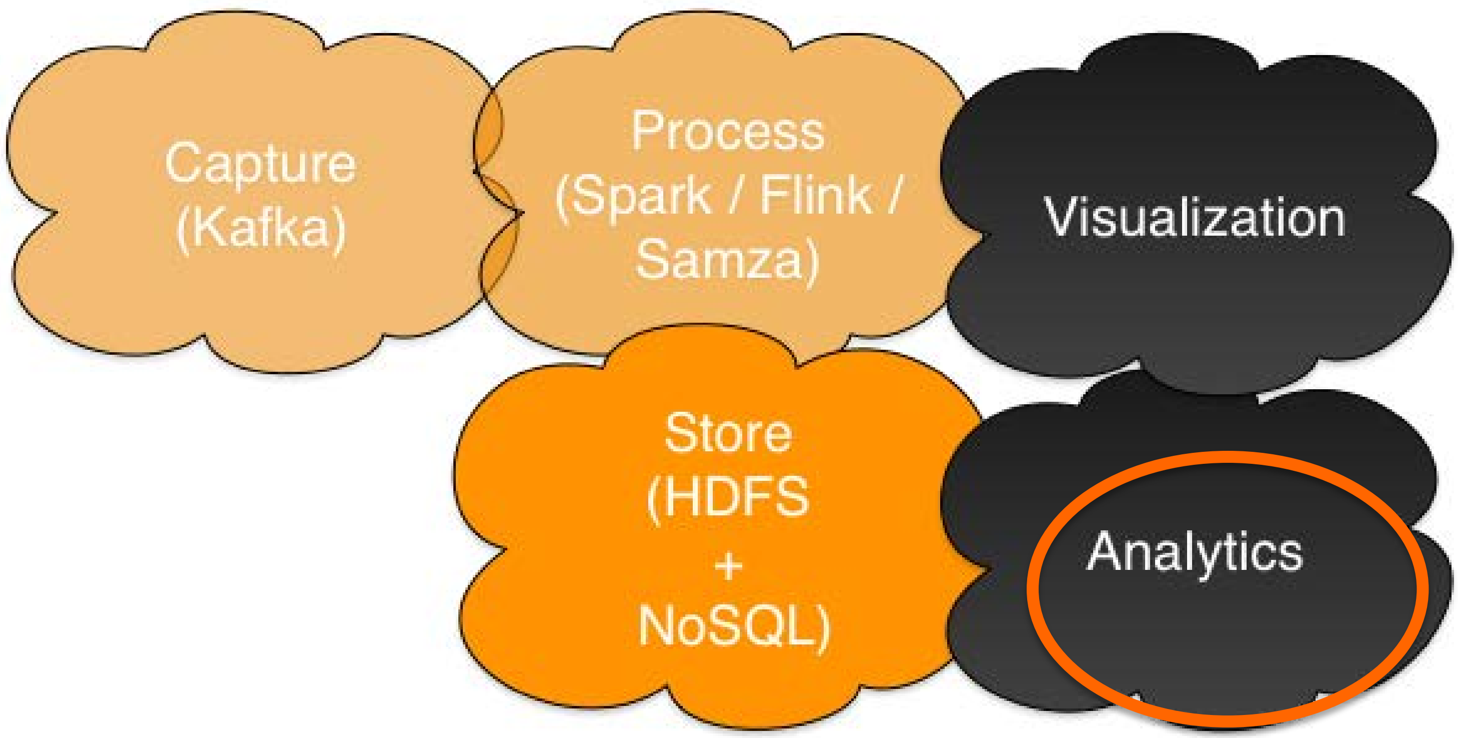
(3) Storage (real time)
Choices for NOSQL

- Too many ! ☺
- HBase
  - Part of Hadoop eco system
  - Uses HDFS for storage
  - Provides consistent view of data
- Cassandra
  - Popular NoSQL store
  - No Single Point of Failure (SPOF) – ring architecture
  - No dependency on Hadoop
- Druid
  - Sub second OLAP queries / fast aggregations

Storage


Capture
(Kafka)
Process
(Spark / Flink /
Samza)
Visualization
Store
(HDFS
+
NoSQL)
Analytics

Next : Analytics



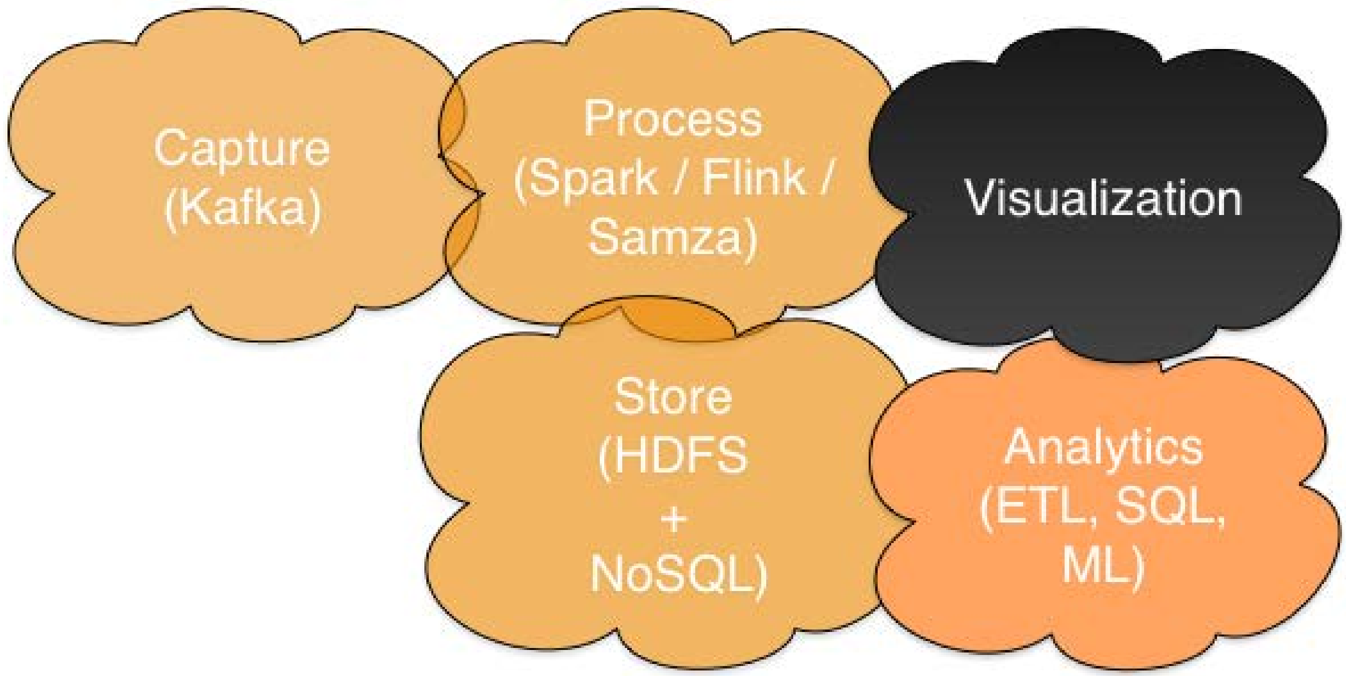
Next : Analytics

- Must scale to peta bytes of data size
- Large queries
  - Popular #hashtags in 2015
- ETL
  - Shape / clean data
- Data warehousing
  - Batch queries
- Machine Learning
  - Model building (credit scoring ..etc)

Analytics Tools

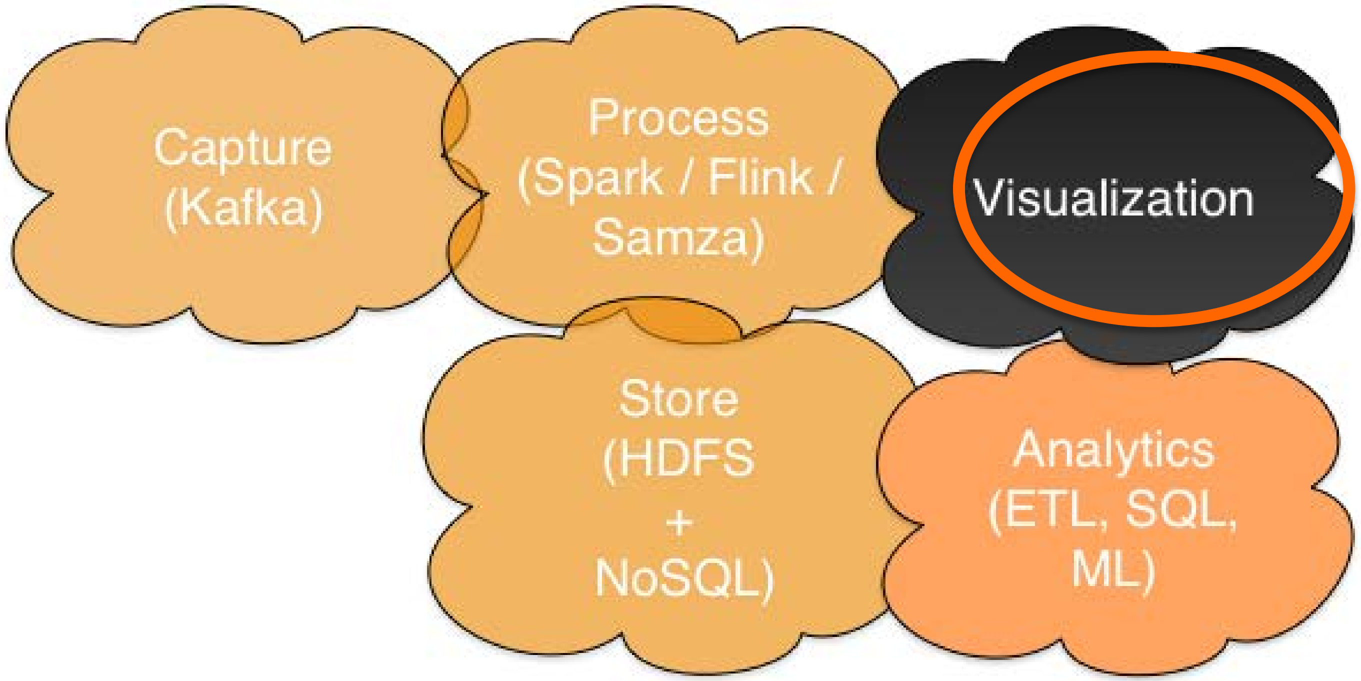
- ETL
  - Pig
  - Spark
  - Flink
- SQL queries
  - Hive / Impala
  - Drill
  - Spark SQL
  - Flink SQL
- Machine Learning
  - Spark ML
  - Flink ML

Analytics



Capture
(Kafka)
Process
(Spark / Flink /
Samza)
Visualization
Store
(HDFS
+
NoSQL)
Analytics
(ETL, SQL,
ML)

Next : Visualization



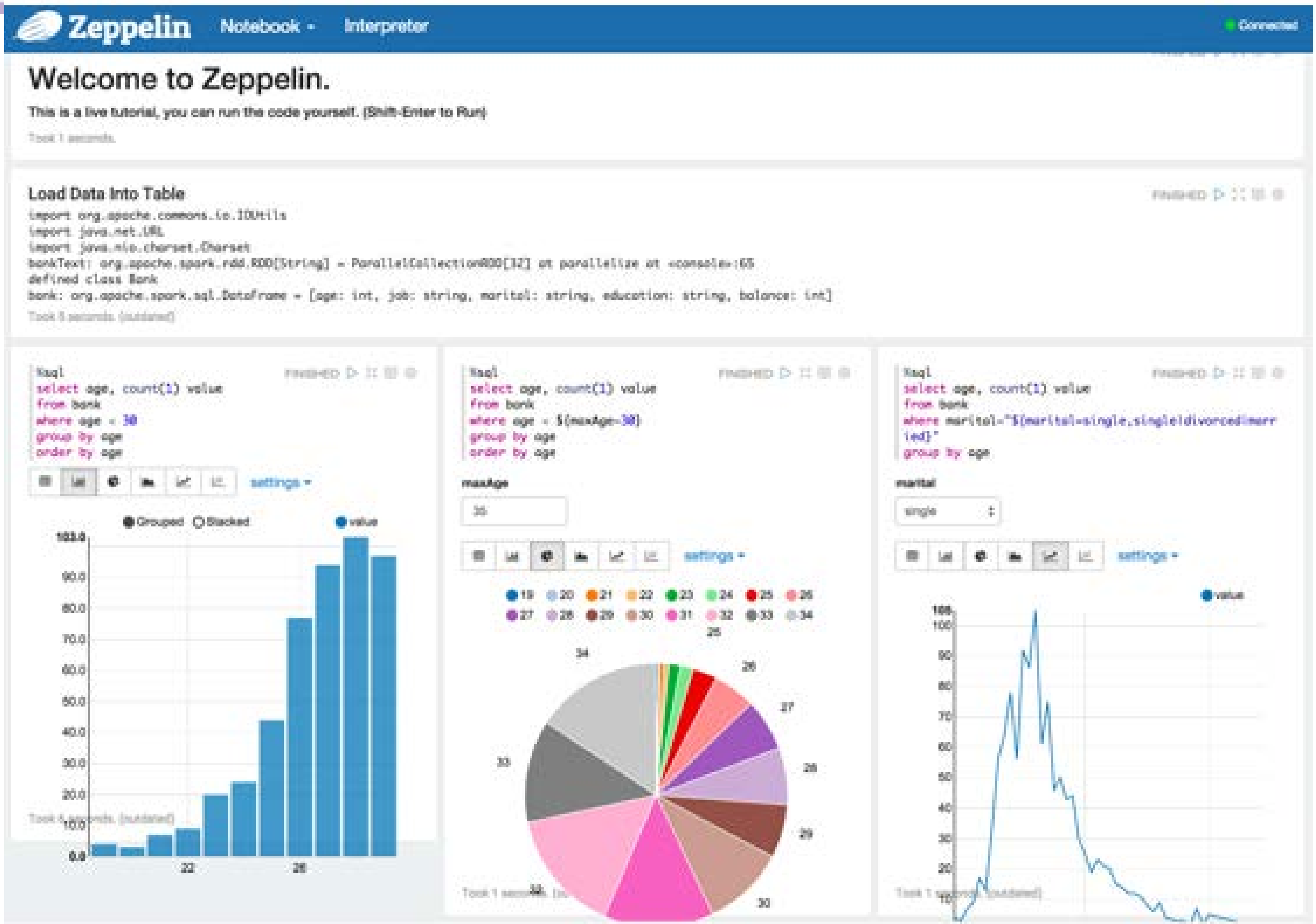
Visualization Tools

- Ready made for enterprises
  - Tableau
  - SiSense
  - Pentaho
- Roll your own
  - Notebooks
  - D3.js
  - R
- And don't forget…
  - Excel !!

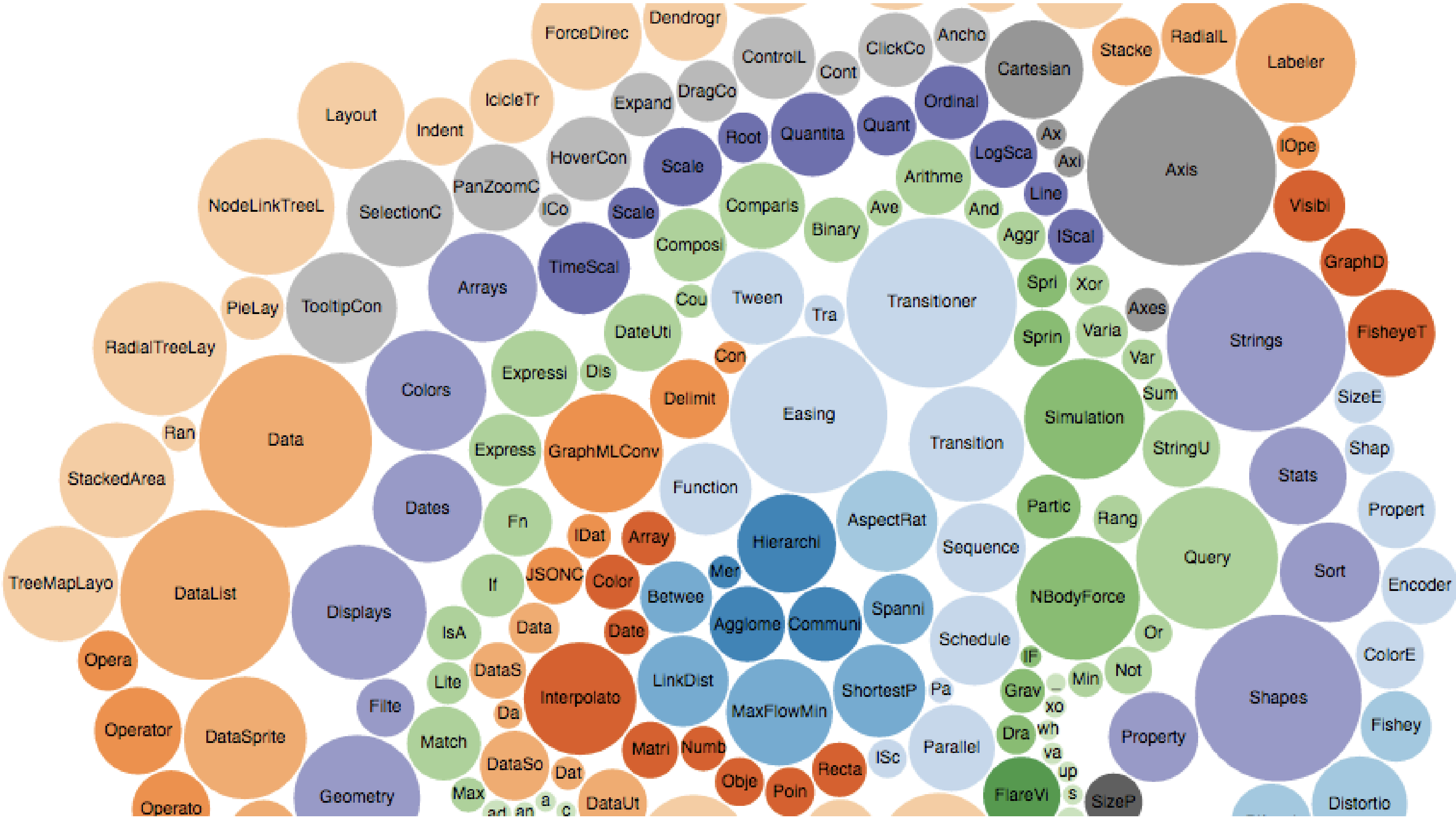
Notebooks

- Zeppelin
- Spark Notebook
- iPython

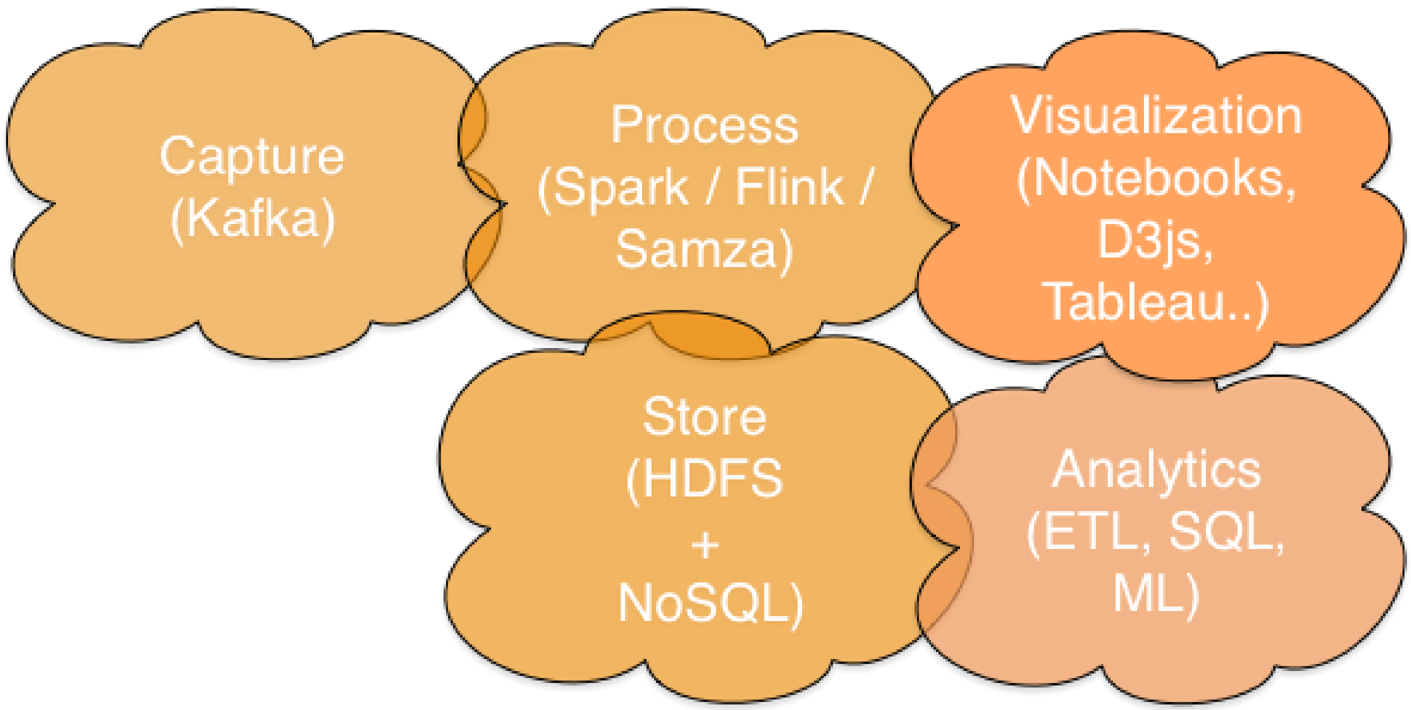
Notebook Example



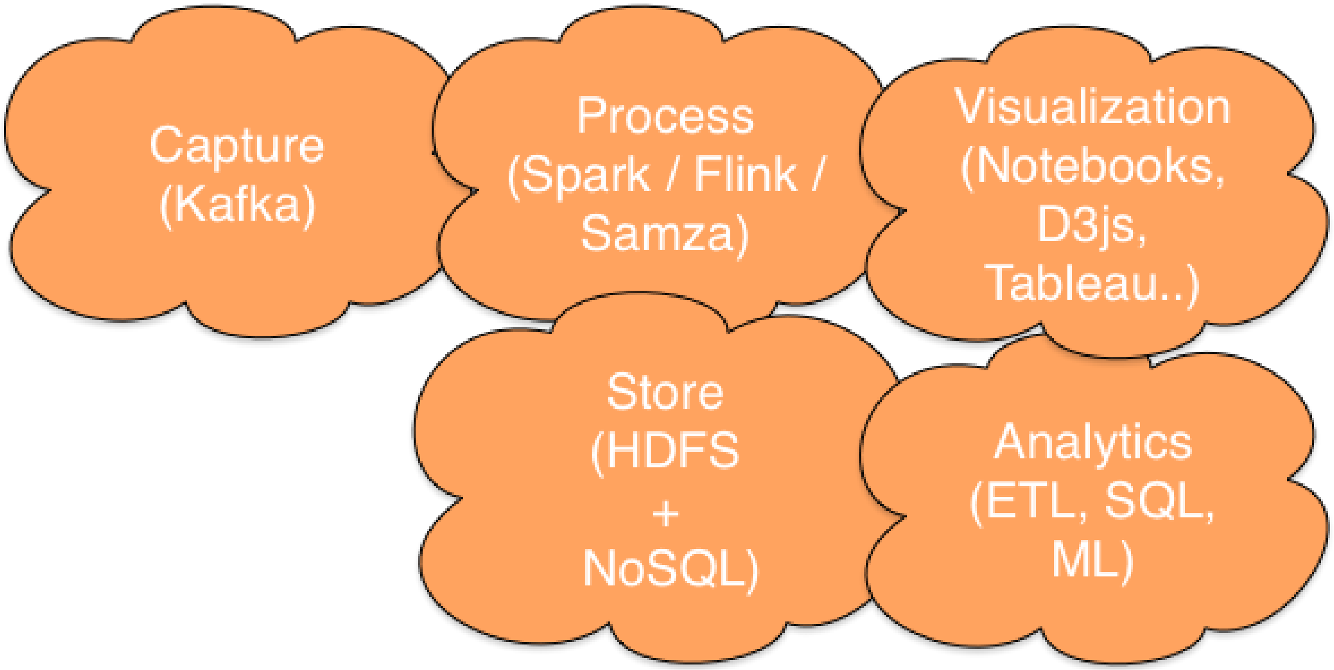
D3.js Example



Visualization



Final Stack


Capture
(Kafka)
Process
(Spark / Flink /
Samza)
Visualization
(Notebooks,
D3js,
Tableau..)
Store
(HDFS
+
NoSQL)
Analytics
(ETL, SQL,
ML)

Final Words

- No one can know every thing !
  - At least get a basic understanding
- Levels of knowledge
  - I haven't heard of it
  - I have heard of it
  - I have played around with it on my laptop
  - I have working knowledge
  - I am an expert / I wrote the damn thing !

Also keep in mind…

≠

At scale nothing works as advertised !

Attribution & Feedback

The SNIA Education Committee thanks the following Individuals for their contributions to this Tutorial.

Authorship History

Sujee Maniyam - May 2016

Additional Contributors

*Please send any questions or comments regarding this SNIA Tutorial to **tracktutorials@snia.org***