UFPR - Universidade Federal do Paraná
SEPT - Setor de Educação Profissional e Tecnológica

IAA - Especialização em Inteligência Artificial Aplicada

IAA013 - Big Data (Parte 3)

Prof. **João Eugenio** Marynowski – jeugenio@ufpr.br

Conteúdo Anterior

- Fundamentos de Big Data
  - Big Data, Data Lake e Data Science
- Map Reduce e Hadoop
  - Utilização da Sandbox/VM
  - Personalização de aplicações Map Reduce
- Data Engineering (Gerenciamento/Ferramentas para Big Data)
- NoSQL e NewSQL
- **Dados em movimento – Processamento de Streaming**

Atividade 9 – Storm

- Enviar um arquivo PDF respondendo qual a diferença entre os dados manipulados por aplicações Hadoop e pelo Storm?
- Comente brevemente justificando e apresentando uma possível utilização do Storm.

???

Atividade 10 – Estudo de Caso

- Enviar um arquivo PDF contendo uma descrição breve (2 páginas) sobre a implementação de uma aplicação ou estudo de caso envolvendo Big Data e suas ferramentas (NoSQL/Streaming).
  - Caracterizar os dados e seus Vs, e sobre a modelagem

- Também preparar uma apresentação (5 min)

Faremos a Tarde!

Programa

- Fundamentos de Big Data
  - Big Data, Data Lake e Data Science
- Map Reduce e Hadoop
  - Utilização da Sandbox/VM
  - Personalização de aplicações Map Reduce
- Data Engineering (Gerenciamento/Ferramentas para Big Data)
  - NoSQL e NewSQL
  - Dados em movimento – Processamento de Streaming

Hoje

Detalhes sórdidos e ruminar tudo isso aí!

Execução do Wordcount no Storm da HDP 2.1

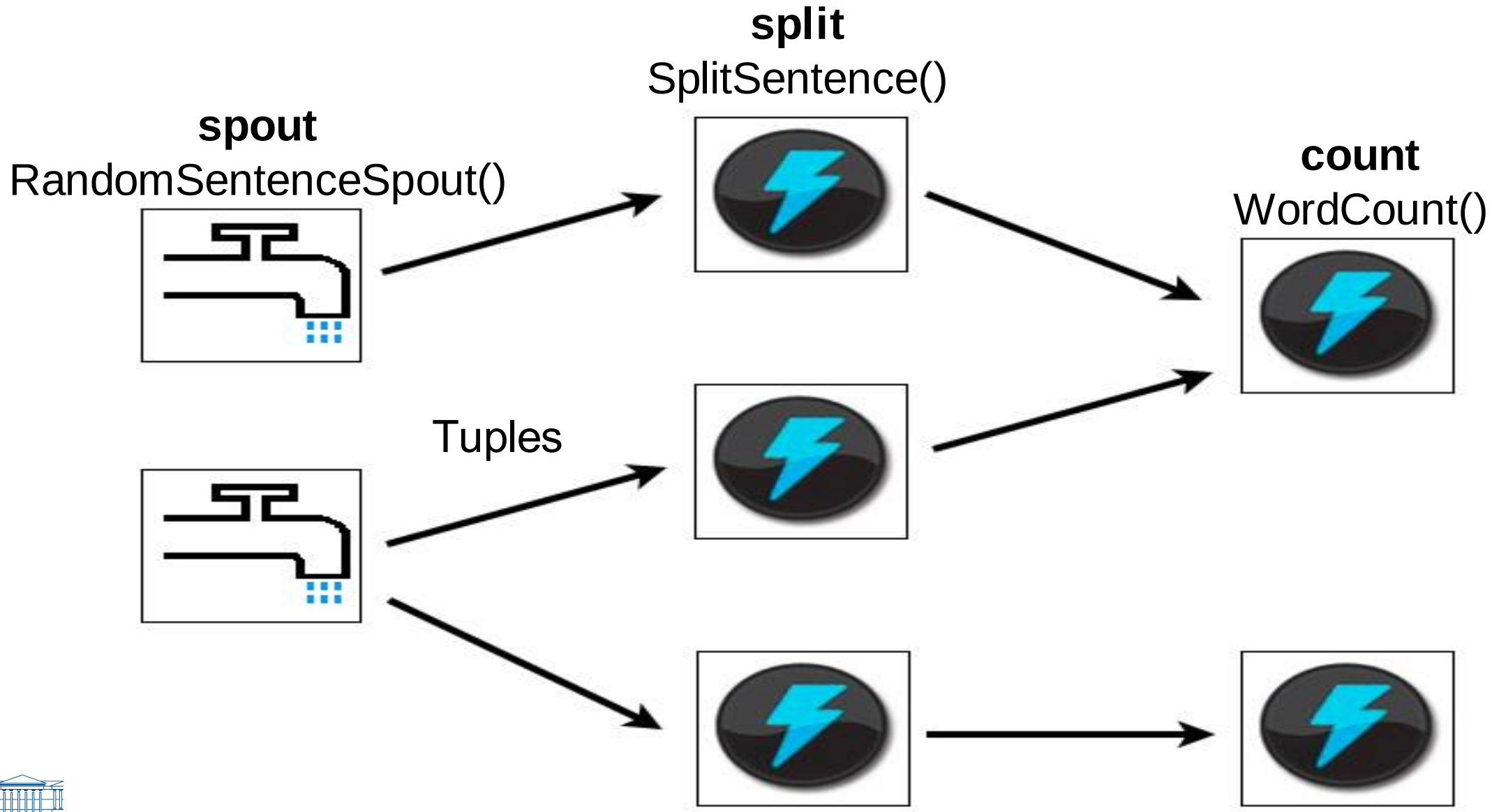
- Start Ambari
  - http://127.0.0.1:8000 (Ambari, Enable)
- Start Storm
  - http://127.0.0.1:8080 (admin:admin)
    (Storm, Service Actions, Start)
- Storm UI
  - http://127.0.0.1:8744
- Submit an wordcount topology to Storm
  - `storm jar /usr/lib/storm/contrib/storm-starter/storm-starter-0.9.1.2.1.1.0-385-jar-with-dependencies.jar storm.starter.WordCountTopology WordCount`

Obs: storm jar storm*.jar <classe para executar> <nome da topologia>

Administração do Storm

- Listar topologias ativas
  - `storm list`
  - `http://127.0.0.1:8744`
- Sumário da topologia
  - `http://127.0.0.1:8744 (clicar no nome da topologia)`
- Detalhes dos Spout e Bolts
  - `http://127.0.0.1:8744 (clicar nos spouts e bolts)`
- Logs (de acordo com os executores/portas)
  - `less /var/log/storm/worker-6700.log`
  - `tail -f /var/log/storm/worker-6701.log`
    `(Crt+C para sair)`
- Finalizar
  - `storm kill WordCount | http://127.0.0.1:8744 (Kill)`

Topologia WordCount



WordCount Topology

```
TopologyBuilder builder = new TopologyBuilder();
builder.setSpout("spout", new RandomSentenceSpout(), 2);

builder.setBolt(”split", new SplitSentence(), 3)
           .shuffleGrouping("spout");
builder.setBolt(”count", new WordCount(), 2)
           .fieldsGrouping(”split", new Fields("word"));

Config conf = new Config();
LocalCluster cluster = new LocalCluster();
cluster.submitTopology(args[0], conf,
builder.createTopology());
```

Detalhes sórdidos...

WordCount Topology

```
TopologyBuilder builder = new TopologyBuilder();
builder.setSpout("spout", new RandomSentenceSpout(), 2);

builder.setBolt(”split", new SplitSentence(), 3)
           .shuffleGrouping("spout");
builder.setBolt(”count", new WordCount(), 2)
           .fieldsGrouping(”split", new Fields("word"));

Config conf = new Config();
LocalCluster cluster = new LocalCluster();
cluster.submitTopology(args[0], conf,
builder.createTopology());
```

```
  public static class RandomSentenceSpout extends BaseRichSpout {
    public RandomSentenceSpout(int ninterval){
      interval = ninterval;
    }
    public void nextTuple() {
      Utils.sleep(interval);
      String[] sentences = new String[]{"the cow jumped over the
moon", "an apple a day keeps the doctor away", "four score and seven
years ago", "snow white and the seven dwarfs", "i am at two with
nature"};
      String sentence = sentences[_rand.nextInt(sentences.length)];
      _collector.emit(new Values(sentence));
    }
    public void declareOutputFields(OutputFieldsDeclarer declarer) {
      declarer.declare(new Fields("word"));
    }
```

WordCount Topology

```
TopologyBuilder builder = new TopologyBuilder();
builder.setSpout("spout", new RandomSentenceSpout(), 2);

builder.setBolt(”split", new SplitSentence(), 3)
           .shuffleGrouping("spout");
builder.setBolt(”count", new WordCount(), 2)
           .fieldsGrouping(”split", new Fields("word"));

Config conf = new Config();
LocalCluster cluster = new LocalCluster();
cluster.submitTopology(args[0], conf,
builder.createTopology());
```

```
  public static class SplitSentence extends ShellBolt
implements IRichBolt {
    public SplitSentence() {
      super("python", "splitsentence.py");
    }
```

```
class SplitSentenceBolt(storm.BasicBolt):
    def process(self, tup):
        words = tup.values[0].split(" ")
        for word in words:
            storm.emit([word])
```

```
    public void declareOutputFields(OutputFieldsDeclarer
declarer) {
      declarer.declare(new Fields("word"));
    }
```

WordCount Topology

```
TopologyBuilder builder = new TopologyBuilder();
builder.setSpout("spout", new RandomSentenceSpout(), 2);

builder.setBolt(”split", new SplitSentence(), 3)
            .shuffleGrouping("spout");
builder.setBolt(”count", new WordCount(), 2)
            .fieldsGrouping(”split", new Fields("word"));

Config conf = new Config();
LocalCluster cluster = new LocalCluster();
cluster.submitTopology(args[0], conf,
builder.createTopology());
```

- **shuffleGrouping**
  - Distribui igualmente as tuplas entre os bolts
  - “Hello world Hello” → “Hello”; “world”; “Hello”
- **fieldGrouping**
  - Agrupa, distribuindo as duplas de acordo com um campo
  - “Hello world Hello” → “Hello” “Hello”; “world”
- globalGrouping
  - Envia as tuplas para um bolt específico
  - “Hello world Hello” → “Hello” “world” “Hello”
- allGrouping
  - Envia todas as tuplas para todos bolts
  - “Hello world Hello” → “Hello” “world” “Hello”; “Hello” “world” “Hello”; “Hello” “world” “Hello”

WordCount Topology

```
TopologyBuilder builder = new TopologyBuilder();
builder.setSpout("spout", new RandomSentenceSpout(), 2);

builder.setBolt(”split", new SplitSentence(), 3)
            .shuffleGrouping("spout");
builder.setBolt(”count", new WordCount(), 2)
            .fieldsGrouping(”split", new Fields("word"));

Config conf = new Config();
LocalCluster cluster = new LocalCluster();
cluster.submitTopology(args[0], conf,
builder.createTopology());
```

WordCount Topology

```
TopologyBuilder builder = new TopologyBuilder();
builder.setSpout("spout", new RandomSentenceSpout(), 2);

builder.setBolt(”split", new SplitSentence(), 3)
           .shuffleGrouping("spout");
builder.setBolt(”count", new WordCount(), 2)
           .fieldsGrouping(”split", new Fields("word"));

Config conf = new Config();
LocalCluster cluster = new LocalCluster();
cluster.submitTopology(args[0], conf,
builder.createTopology());
```

```
  public static class WordCount extends BaseBasicBolt {
    Map<String, Integer> counts = new HashMap<String,
Integer>();
    public void execute(Tuple tuple, BasicOutputCollector
collector) {
      String word = tuple.getString(0);
      Integer count = counts.get(word);
      if (count == null) { count = 0; }
      count++;
      counts.put(word, count);
      collector.emit(new Values(word, count));
    }
    public void declareOutputFields(OutputFieldsDeclarer
declarer) {
      declarer.declare(new Fields("word", "count"));
    }
```

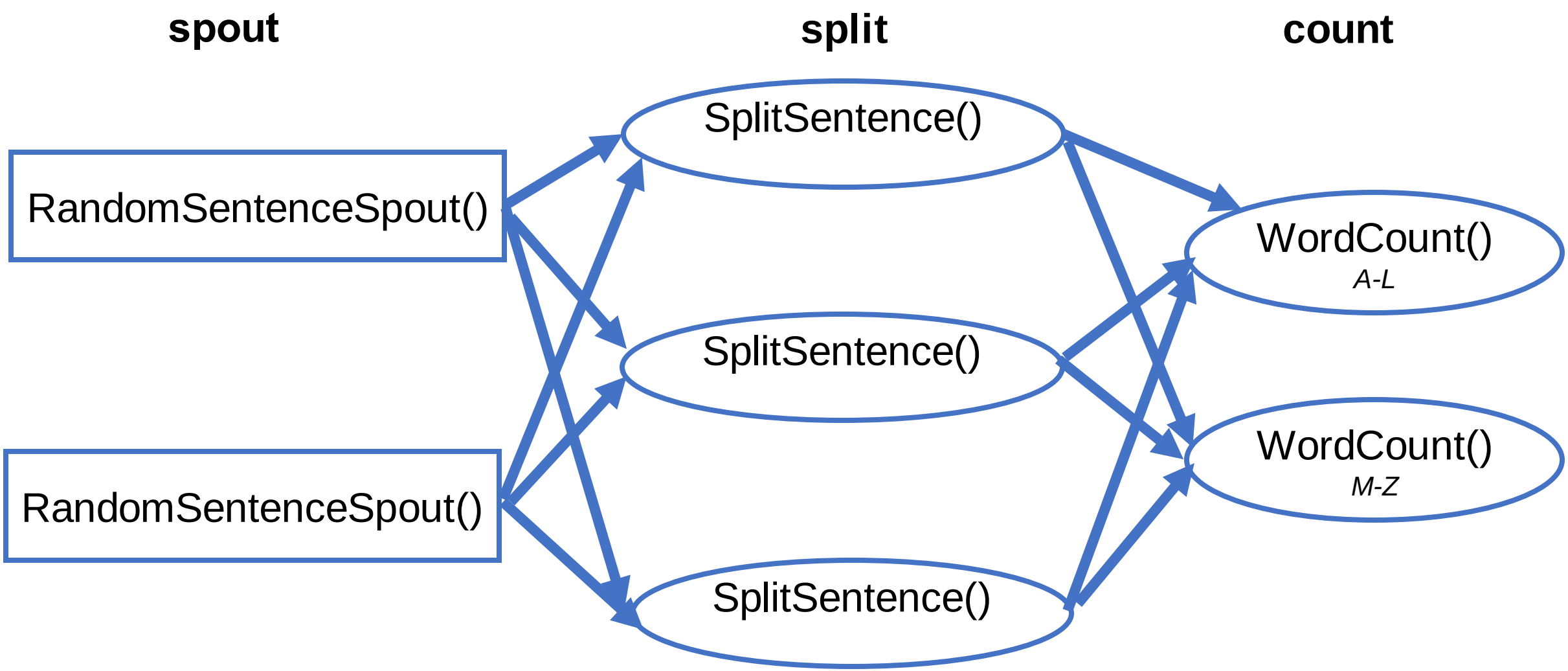
WordCount Topology

```
TopologyBuilder builder = new TopologyBuilder();
builder.setSpout("spout", new RandomSentenceSpout(), 2);

builder.setBolt(”split", new SplitSentence(), 3)
           .shuffleGrouping("spout");
builder.setBolt(”count", new WordCount(), 2)
           .fieldsGrouping(”split", new Fields("word"));

Config conf = new Config();
LocalCluster cluster = new LocalCluster();
cluster.submitTopology(args[0], conf,
builder.createTopology());
```

WordCount Storm Topology



Como faço a minha aplicação Storm?

Implementação e Implantação de uma Aplicação Storm

- Gerar o jar com a aplicação Storm no desktop
  - Seu ambiente de desenvolvimento (Sandbox) não tem JDK e fonte do Storm
- Enviar o jar para a sandbox ou cluster Storm
  - Winscp, scp ou [filebrowser]! :)
- Executar na sandbox ou cluster
- `$storm jar seu.jar seupacote.WordCount topology-name`

- *** *Equivalmente ao Hadoop!*

- Compilar com: $ mvn clean install -DskipTests=true

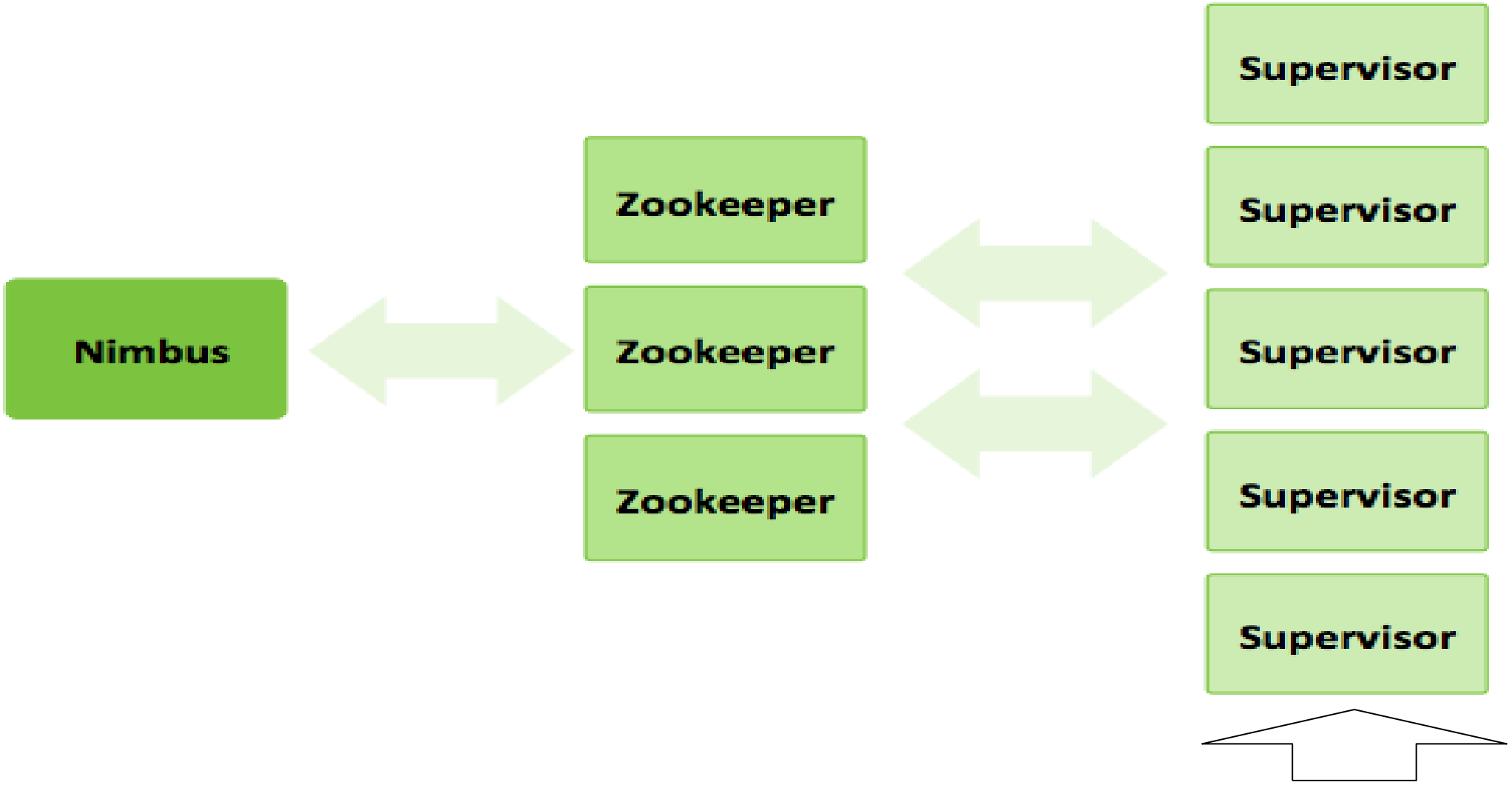
- ou configurar IDE para adicionar a opção skipTests=true na construção.

  * Netbeans: Properties, Actions, [**Build project, Clean and Build project, Build with Dependencies**], Set Properties: skipTests=true

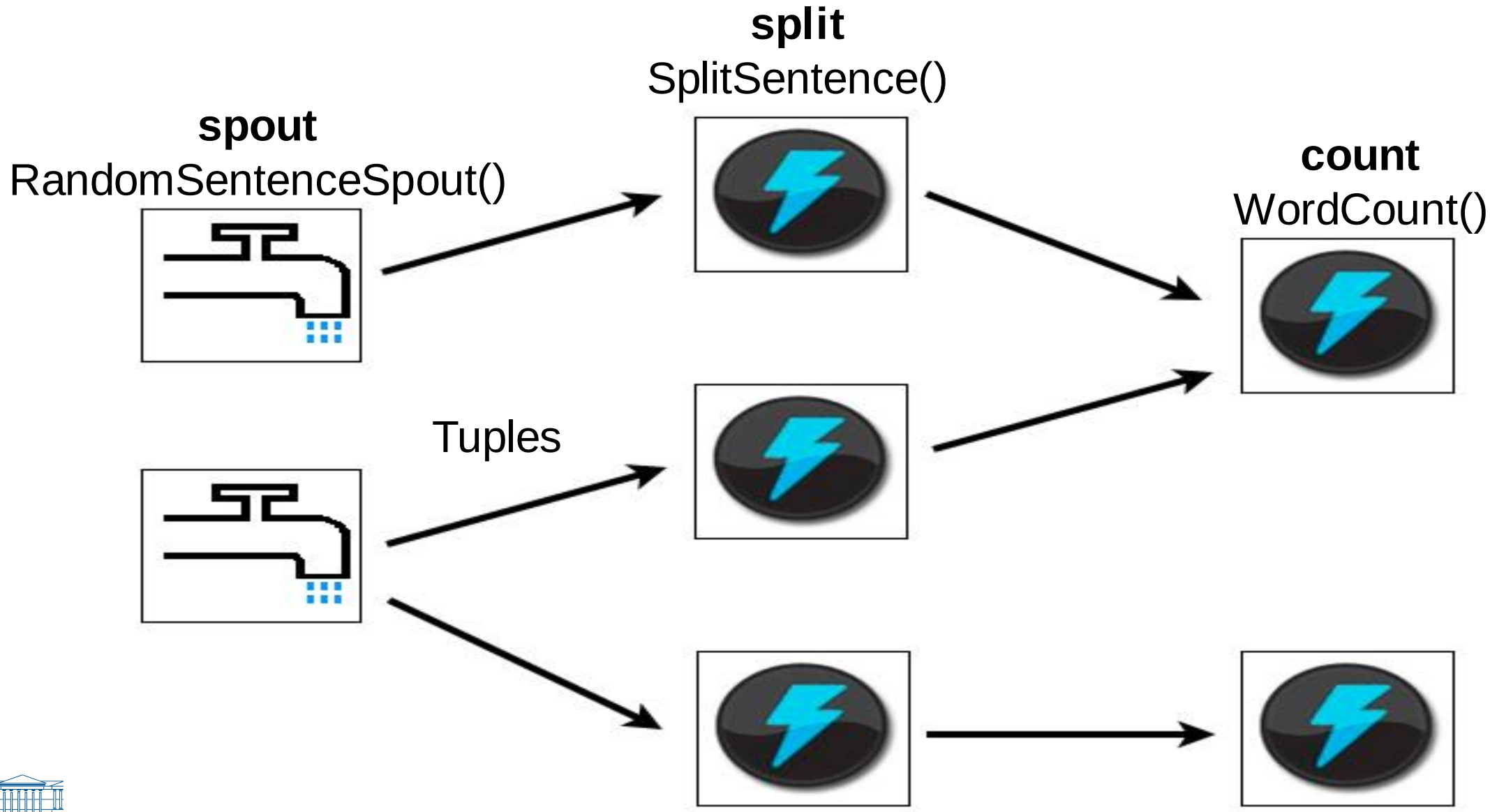
  * POM: <version> 0.9.1.2.1.1.0-385 </version>

https://archive.apache.org/dist/storm/apache-storm-0.9.1-incubating/

Componentes do Storm



Topologia WordCount



Storm Components

- Nimbus node (master, similar to JobTracker)
  - Uploads computations (jobs)
  - Distributes code
  - Launches workers
  - Monitors computation and reallocates workers
- ZooKeeper nodes
  - Coordinates the Storm cluster
- Supervisor nodes
  - Communicates with Nimbus through Zookeeper
  - Starts and stops workers according to signals from Nimbus

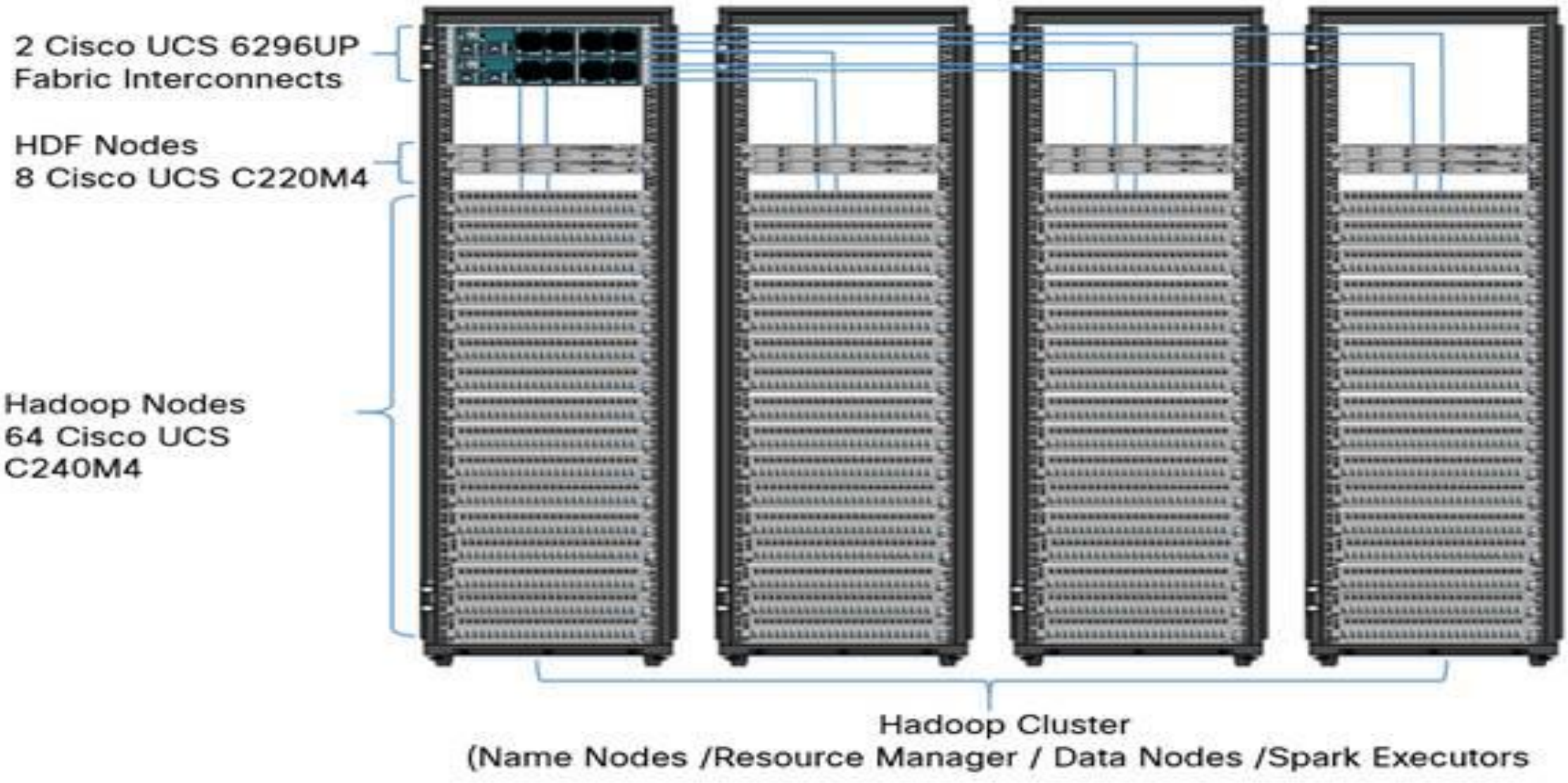
Ver os componentes no Ambari ...
Onde ficam os spouts e bolts... workers

Referencias para o Storm

- JAIN, A.; NALYA, A. Learning Storm. [s.l.] Packt Publishing Ltd., 2014.
- http://www.michael-noll.com/tutorials/running-multi-node-storm-cluster/
- http://br.hortonworks.com/hadoop-tutorial/processing-streaming-data-near-real-time-apache-storm/
- https://github.com/apache/storm/tree/master/examples/storm-starter
- ...

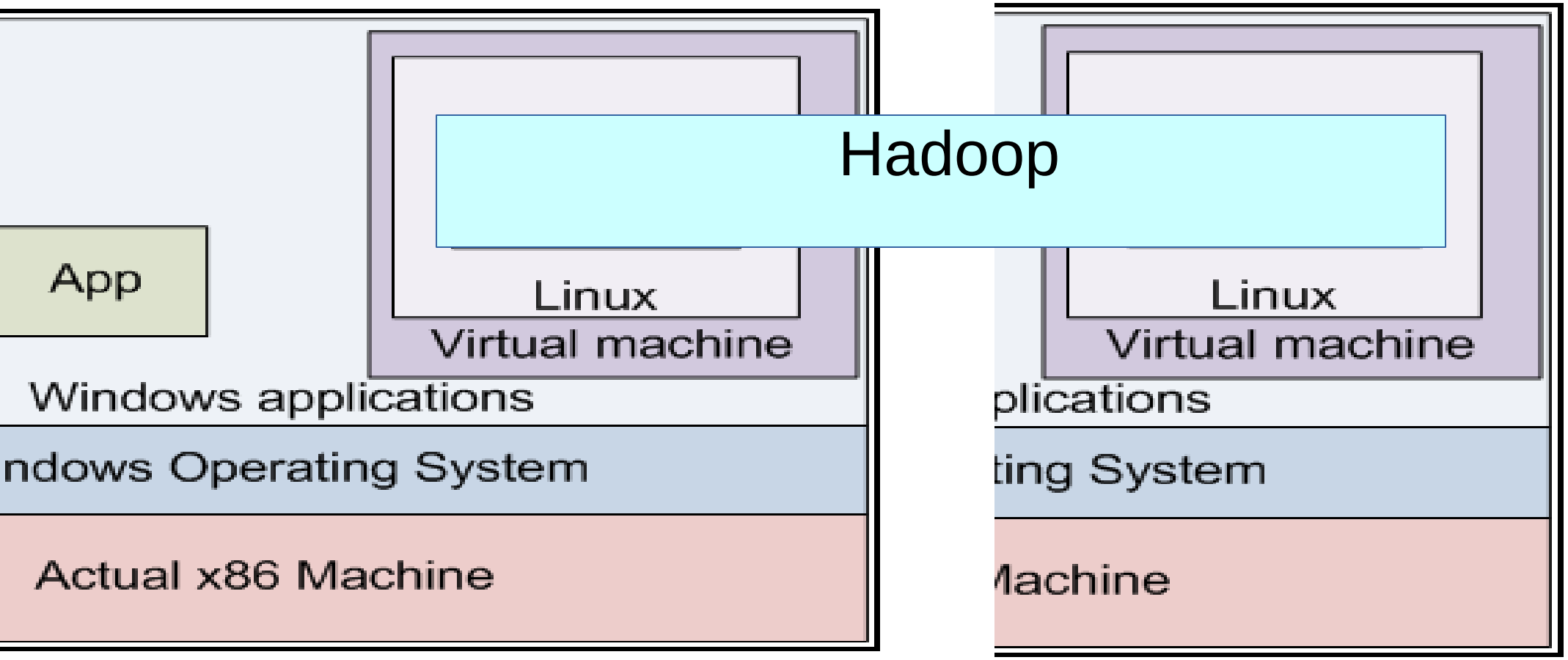
Sistemas Big Data

Sistemas

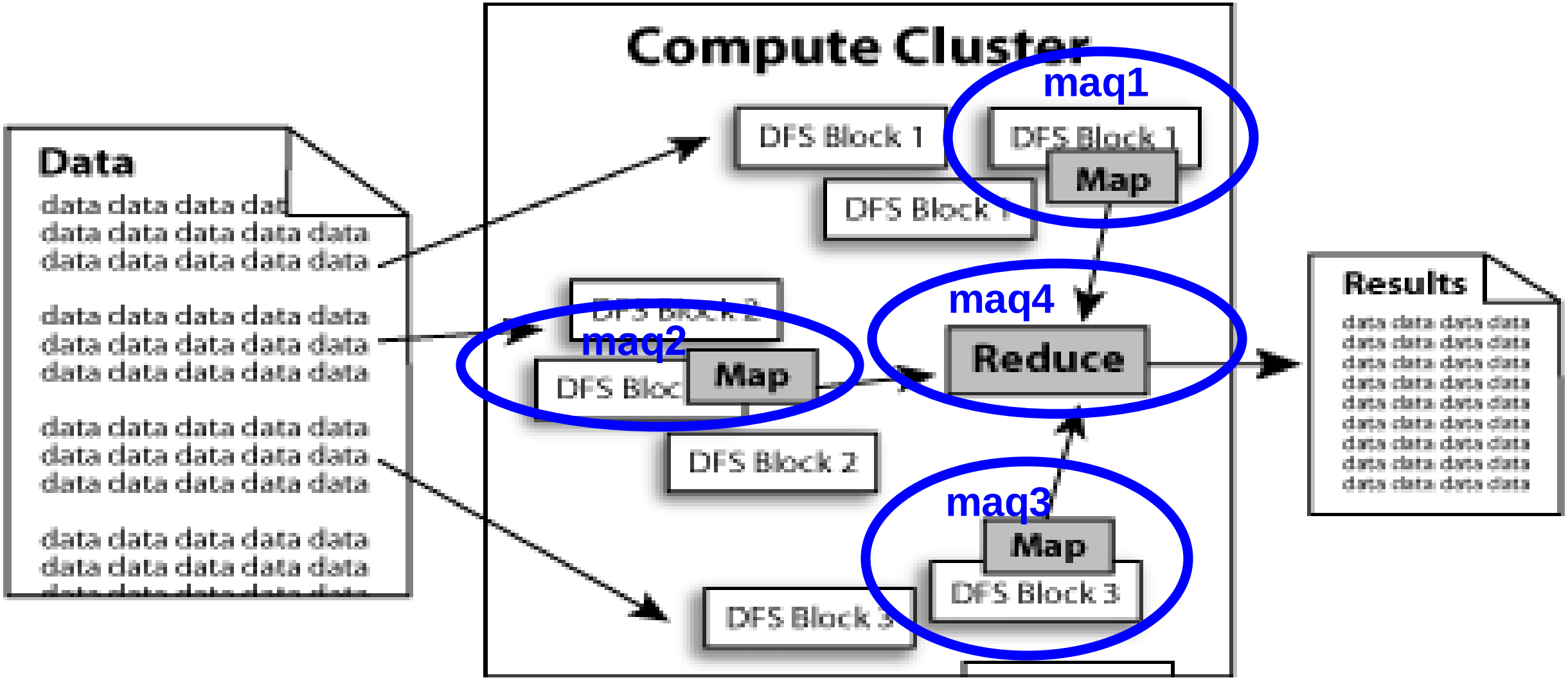


https://www.cisco.com/c/en/us/td/docs/unified_computing/ucs/UCS_CVDs/Cisco_UCS_Integrated_Infrastructure_for_Big_Data_and_Analytics_with_Hortonworks_and_HDF.html

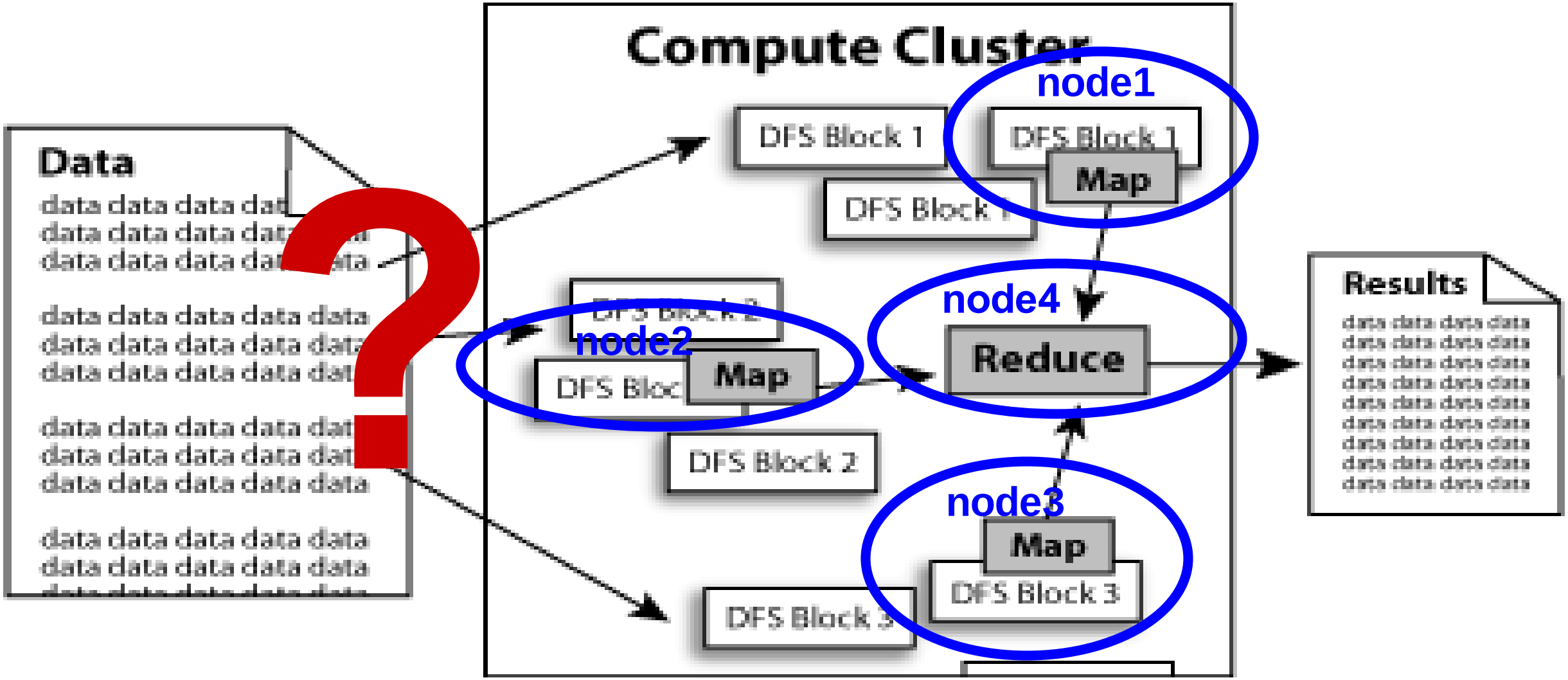
Sandbox HDP x2



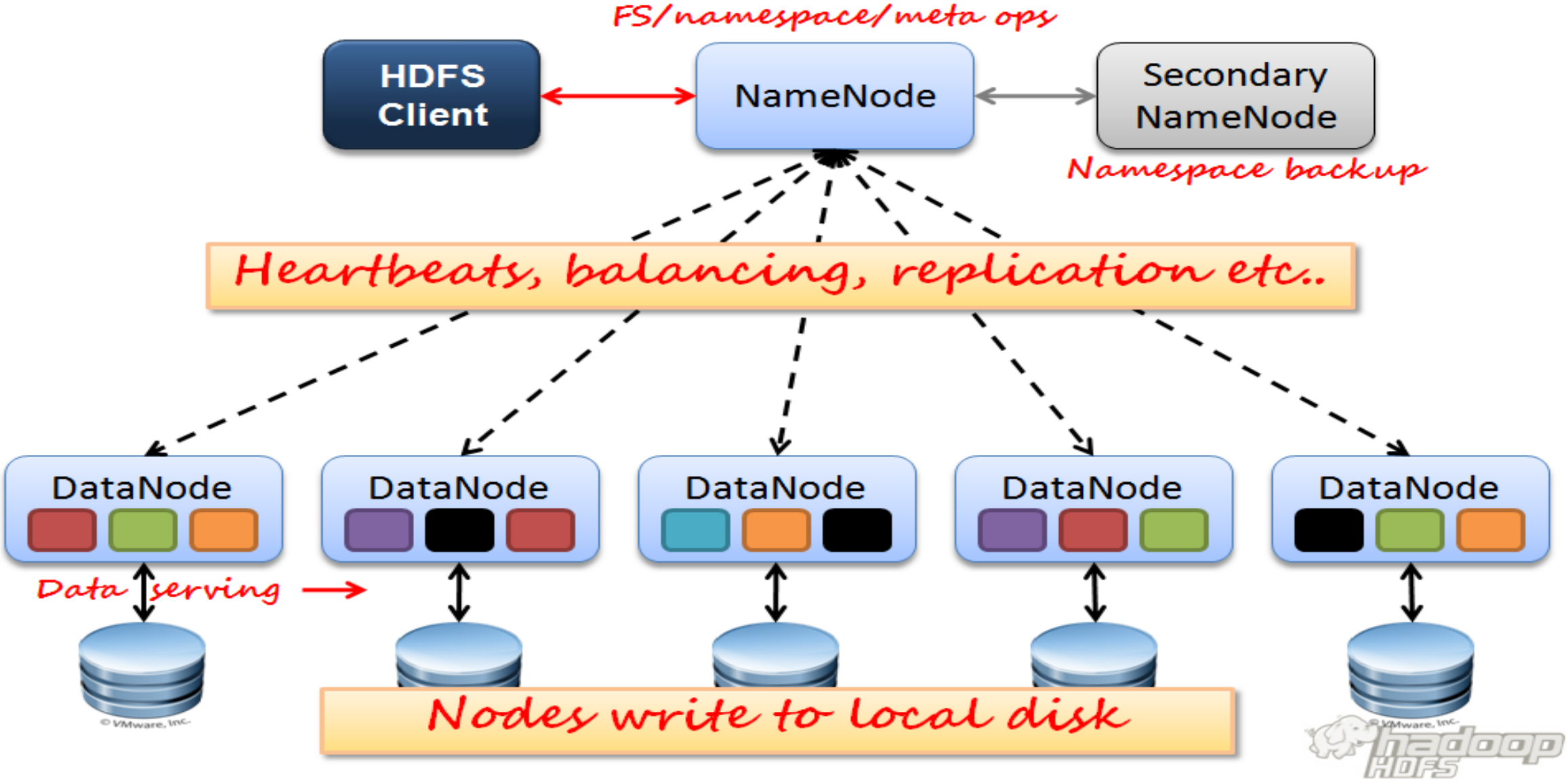
Distributed File System (DFS) + MapReduce

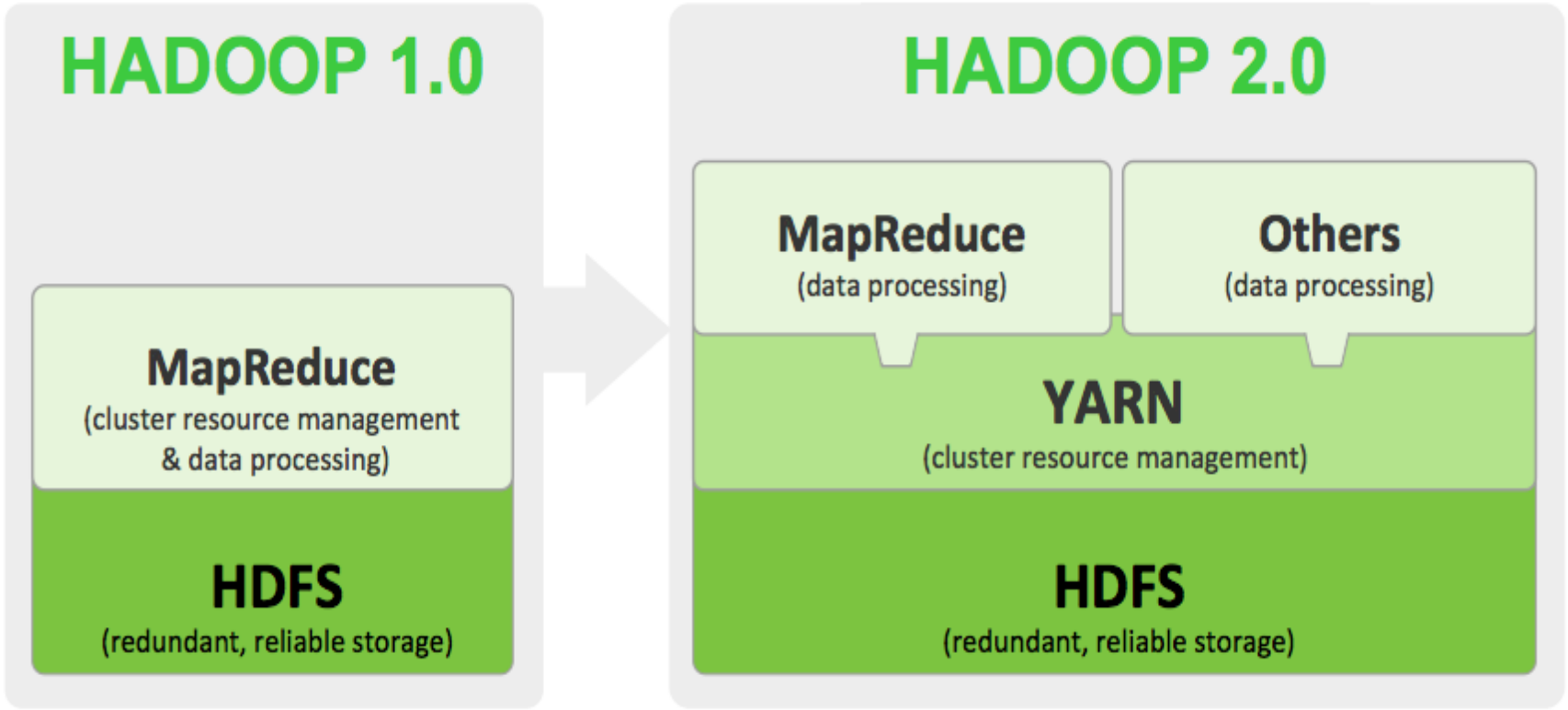


Como Cria um Ambiente Distribuído?

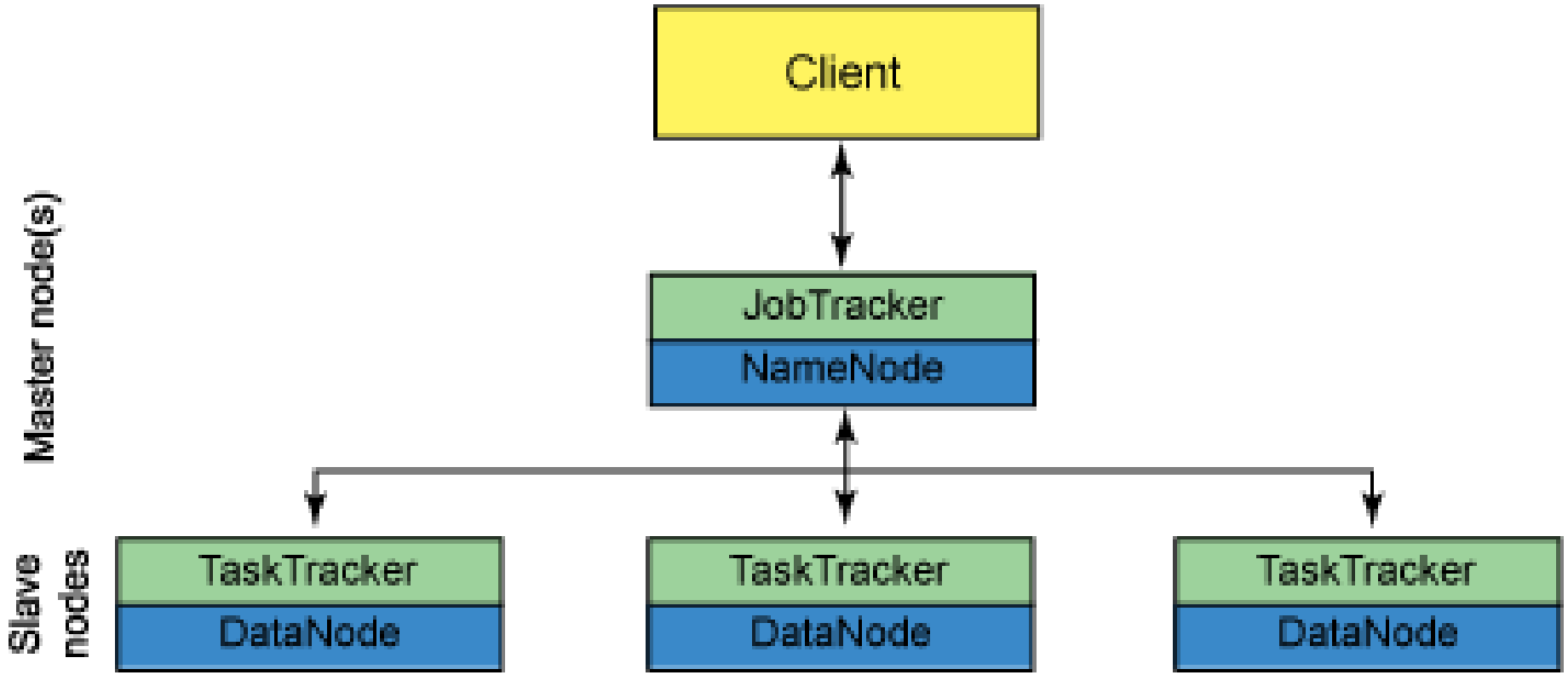


Arquitetura do HDFS

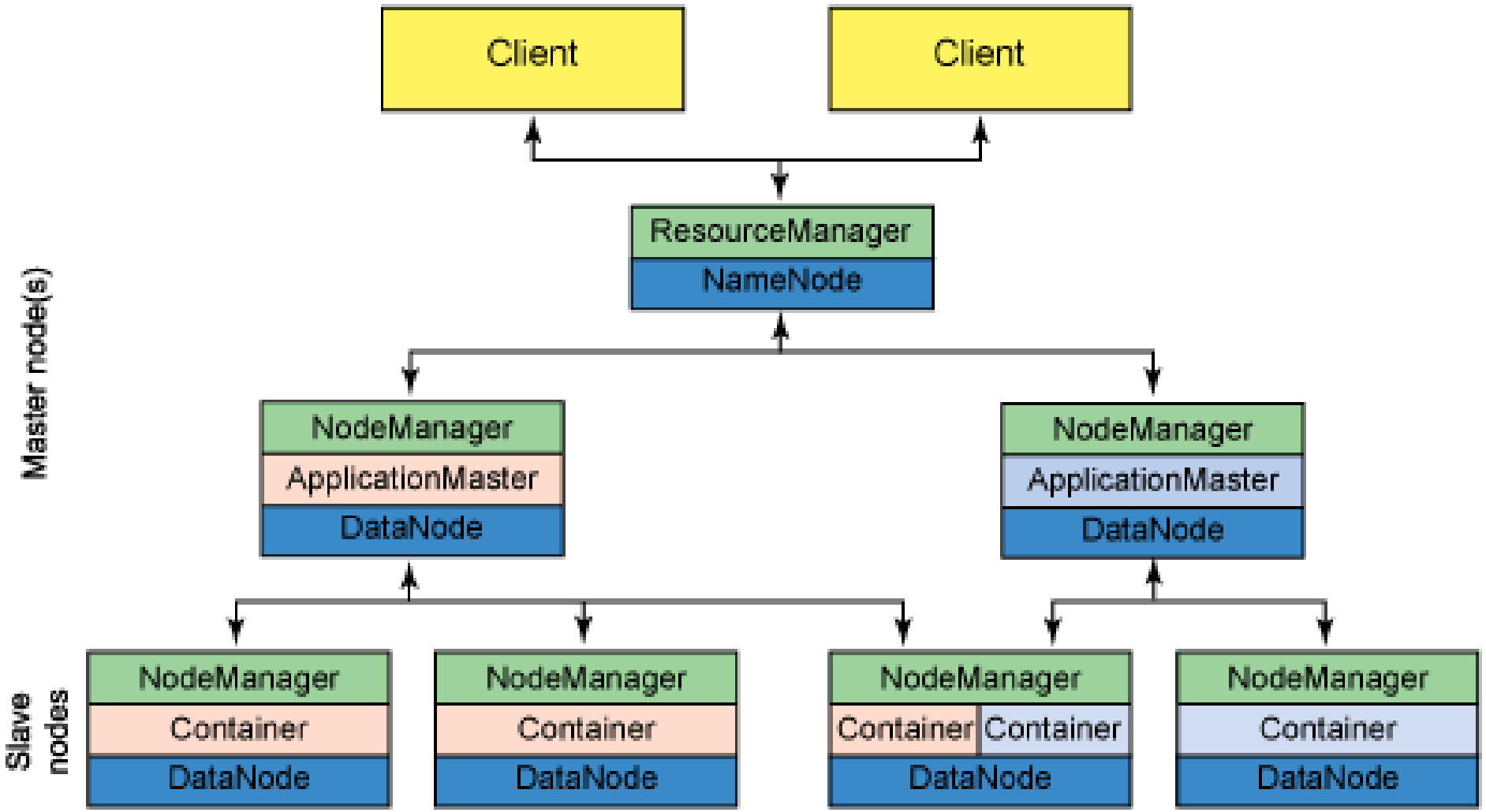



HADOOP 1.0
MapReduce
(cluster resource management
& data processing)
HDFS
(redundant, reliable storage)
HADOOP 2.0
MapReduce
(data processing)
Others
(data processing)
YARN
(cluster resource management)
HDFS
(redundant, reliable storage)

Hadoop 1 (HDFS, MapReduce)



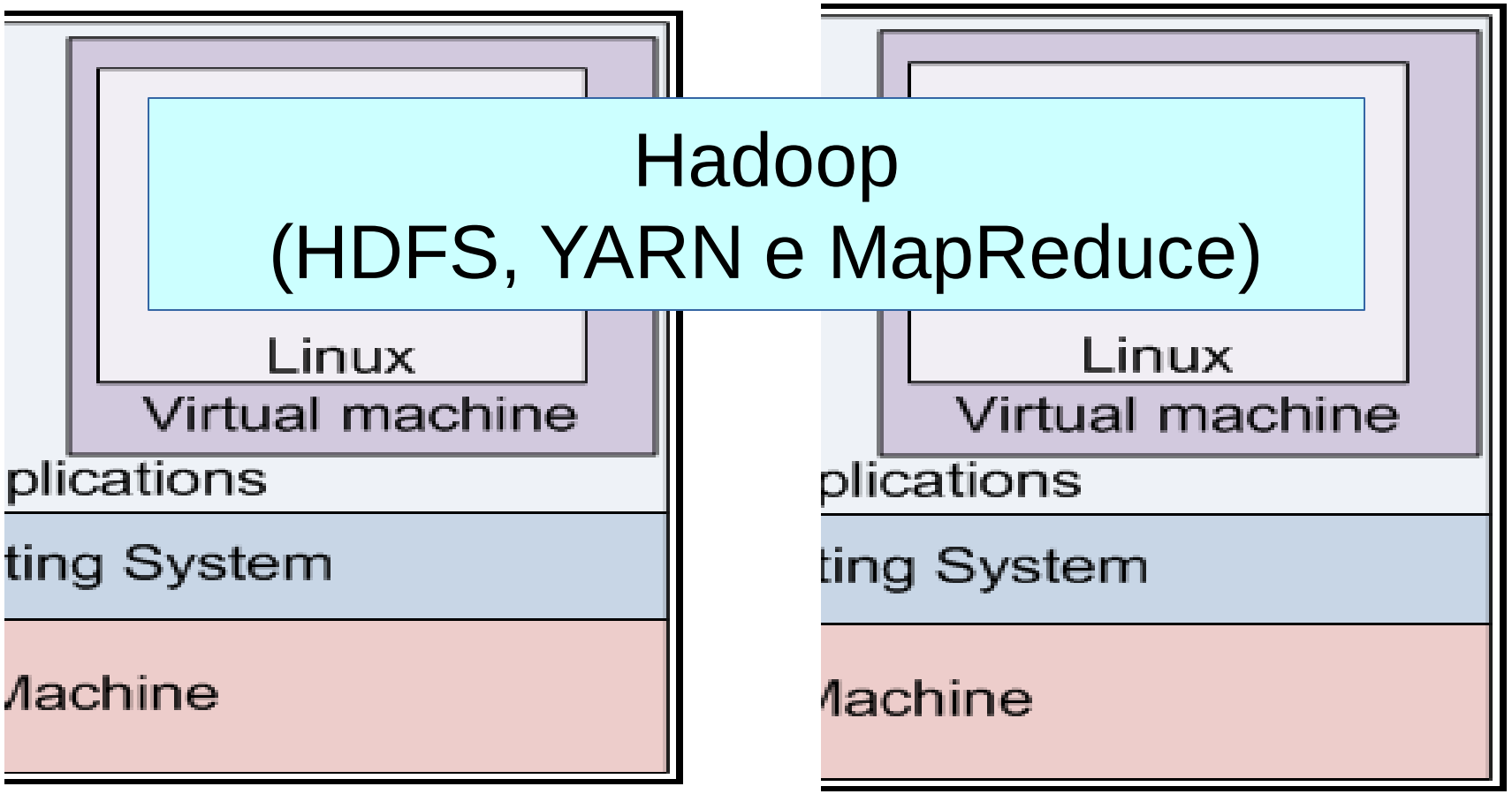
Hadoop 2 (HDFS, YARN, MapReduce)



MapReduce e os Componentes do YARN

- Client – submete um Job MapReduce
- **HDFS NameNode, Secondary NameNode** e **DataNodes**
  - Compartilha artefatos de recursos e Jobs
- **Resource Manager (RM)**
  - Controla o uso dos recursos
- **Node Manager (NM)**
  - Cria container p/ execução e monitora o seu uso
- **Application Master (AM)** – (MapReduce JobTracker)
  - Coordena e gerencia Jobs e tarefas MR
  - Negocia com o RM para escalonar tarefas MR
  - Tarefas MR são iniciadas pelos NM(s)
- **YarnChild** – (MapReduce TaskTracker)
  - Iniciados pelos NMs para executar tarefas MR

Sandbox HDP x2,...



Ecosistema Hadoop HDP 2.1

- Hue – ecosistema Hadoop UI
- Beeswax – Hive UI (interface BDR → SQL)
- Pig (PigLatin – Bash/SQL)
- Hcatalog – Catálogo de bases de dados
- Filebrowser – HDFS UI
- Job Browser – Hadoop Jobs
- Job Designer – aplicações Hadoop
- Oozie designer – Diversos sistemas/aplicações
- **Ambari – Gerência do cluster e aplicações http://<ip>:8080 (admin:admin)**
- Hbase – BD orientado a columa
- Knox – Segurança
- Storm – Stream ...

Adicionar um Slave com HDP

- Desligar as VMs e configure-as para modo **Bridge** (mesma rede), gere **endereço MAC aleatório** e religue as VMs
- Eleger uma máquina ***master*** e configurar os nomes adequadamente em todas as máquinas ajustando seus arquivos ***/etc/hosts***, por exemplo:
  - `192.168.1.16 sandbox.hortonworks.com`
  - `192.168.1.17 slave1.sandbox.hortonworks.com`
- Em cada slave, ativar o seu novo nome: `hostname <nome slaves>` e reiniciar agente: `ambari-agent restart`
- http://<server ip>:8080 (admin:admin)
  - Hosts, Actions, Add New Hosts, Target hosts
    - *Perform manual registration… do not use SSH*
    - Slaves já possuem *ambari agent*
    - Adicione os componentes [DataNode e NodeManager]

***Resolver os warnings conforme informado ajuda a eliminar os serviços em execução no slave e que não serão utilizados.*

UFPR

Especialização em Inteligência Artificial

SEPT

Comandos VI

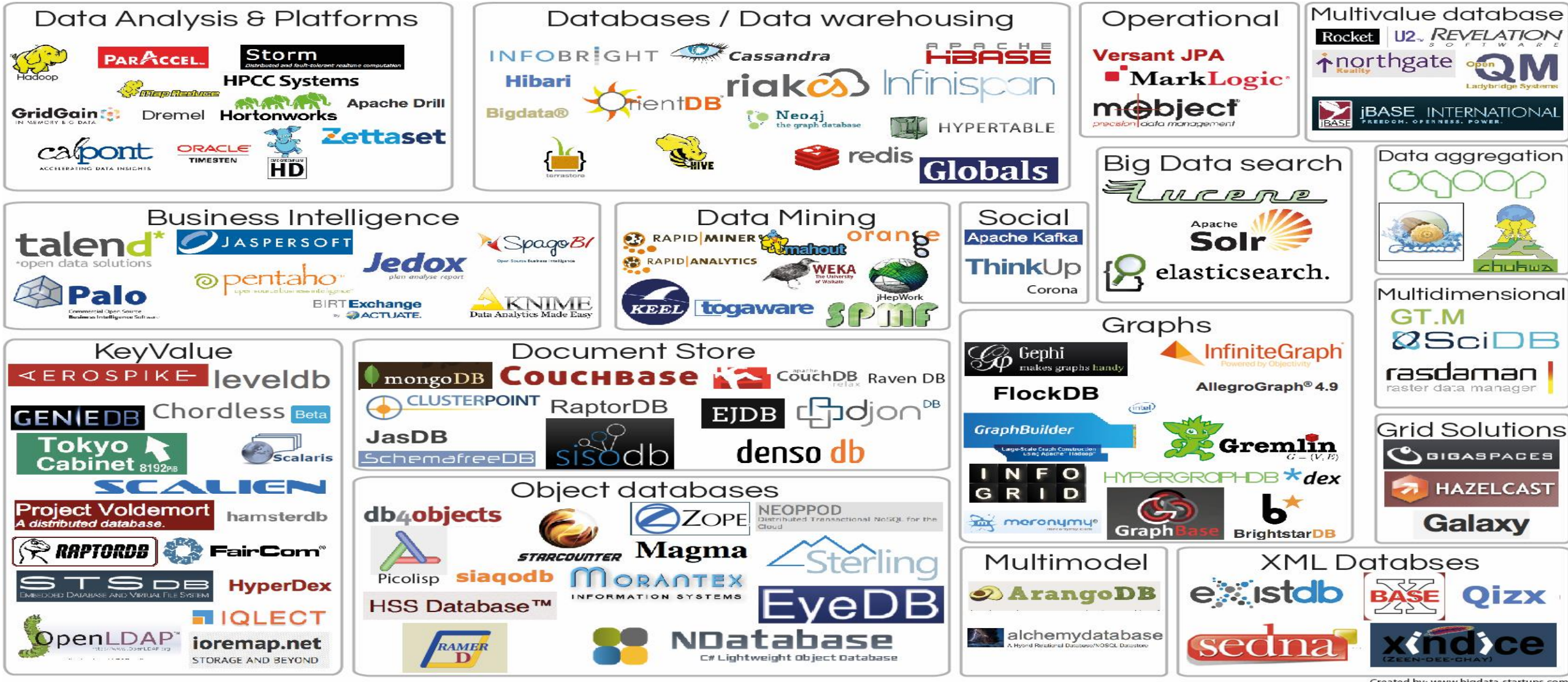
- Não tem vim na Sandbox
- vi
  - <esc> - modo comando
  - i – insere
  - a – insere depois
  - s – substitui
  - yy – copia linha
  - p – cola
  - dd – apaga linha (d apaga caracter)
  - cw – substitui palavra
  - wq – salva e sai

- :8088 – ResourceManager Web UI
  - nodes slaves ativos, senão reinicar NodeManager nos slaves
- :19888 – jobhistory
  - quais máquinas participaram da execução (Job ID, Map|Reduce, task*, Node)

- Instalação Hadoop via Ambari
  - Ssh ou não

Componentes Extras

Componentes de Sistemas Big Data



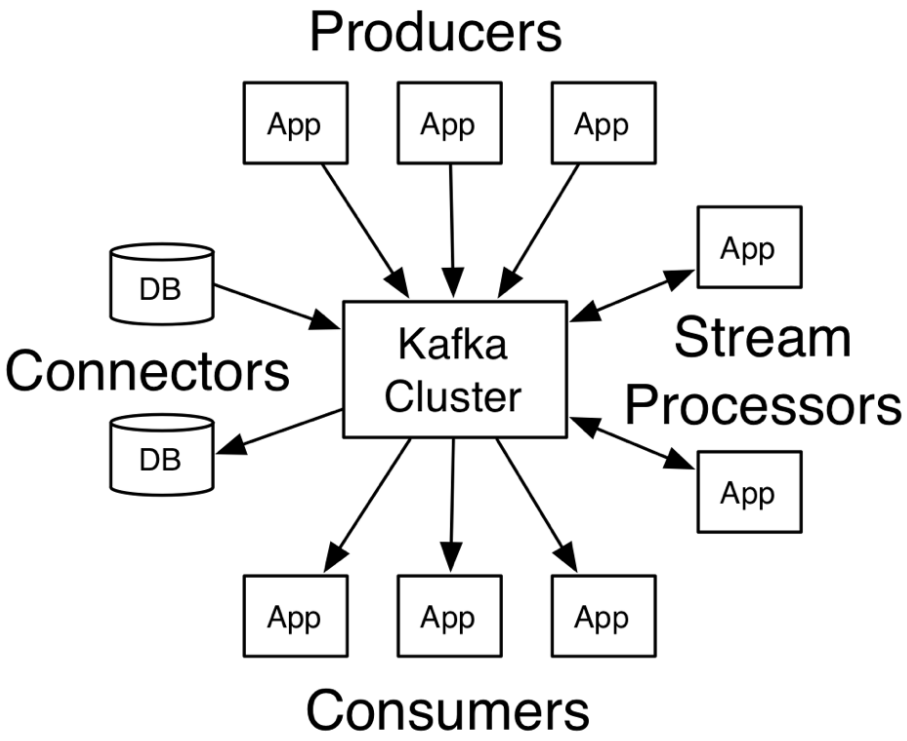
UFPR

Especialização em Inteligência Artificial

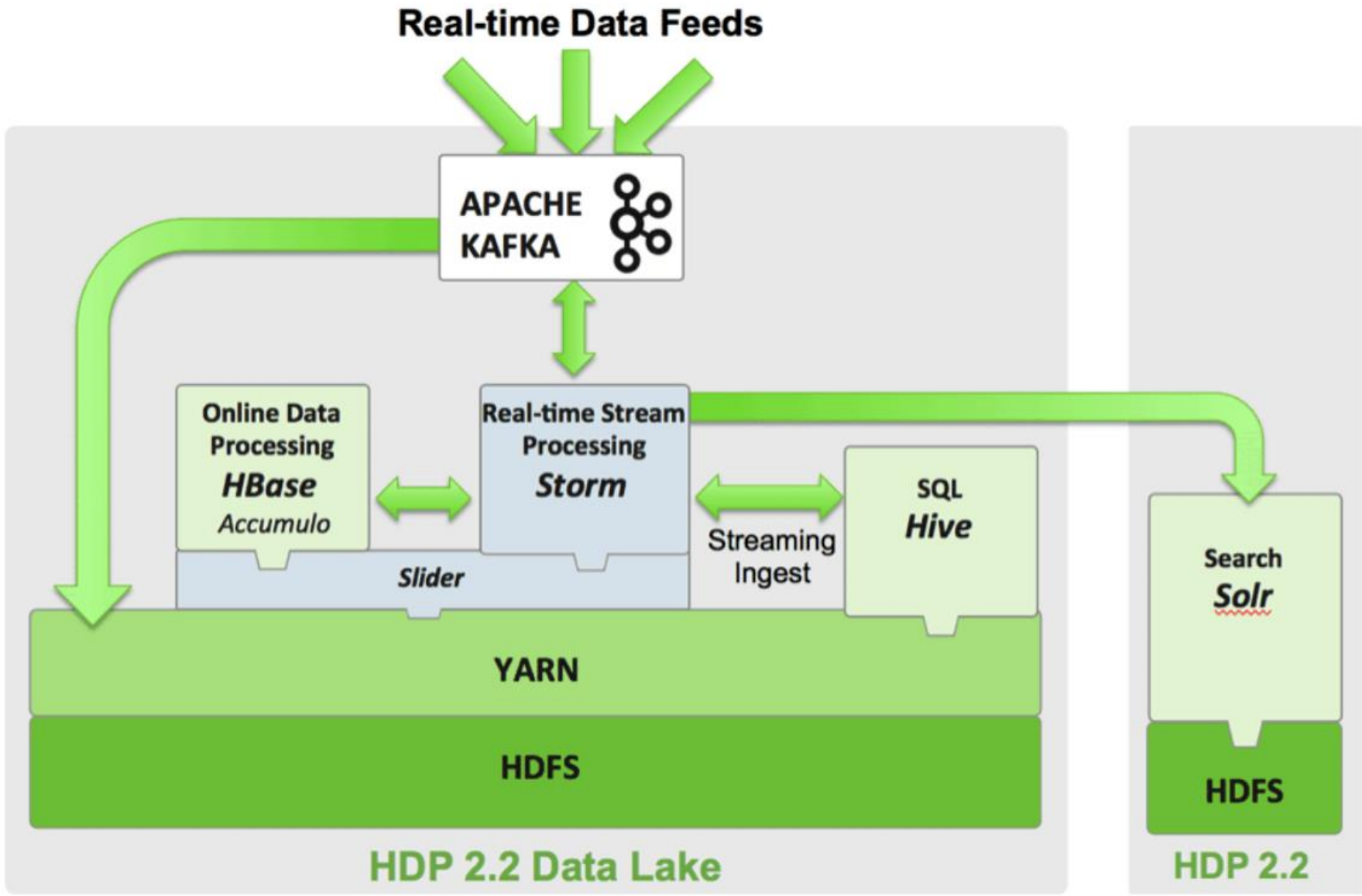
Created by: www.bigdata-startups.com

SEPT

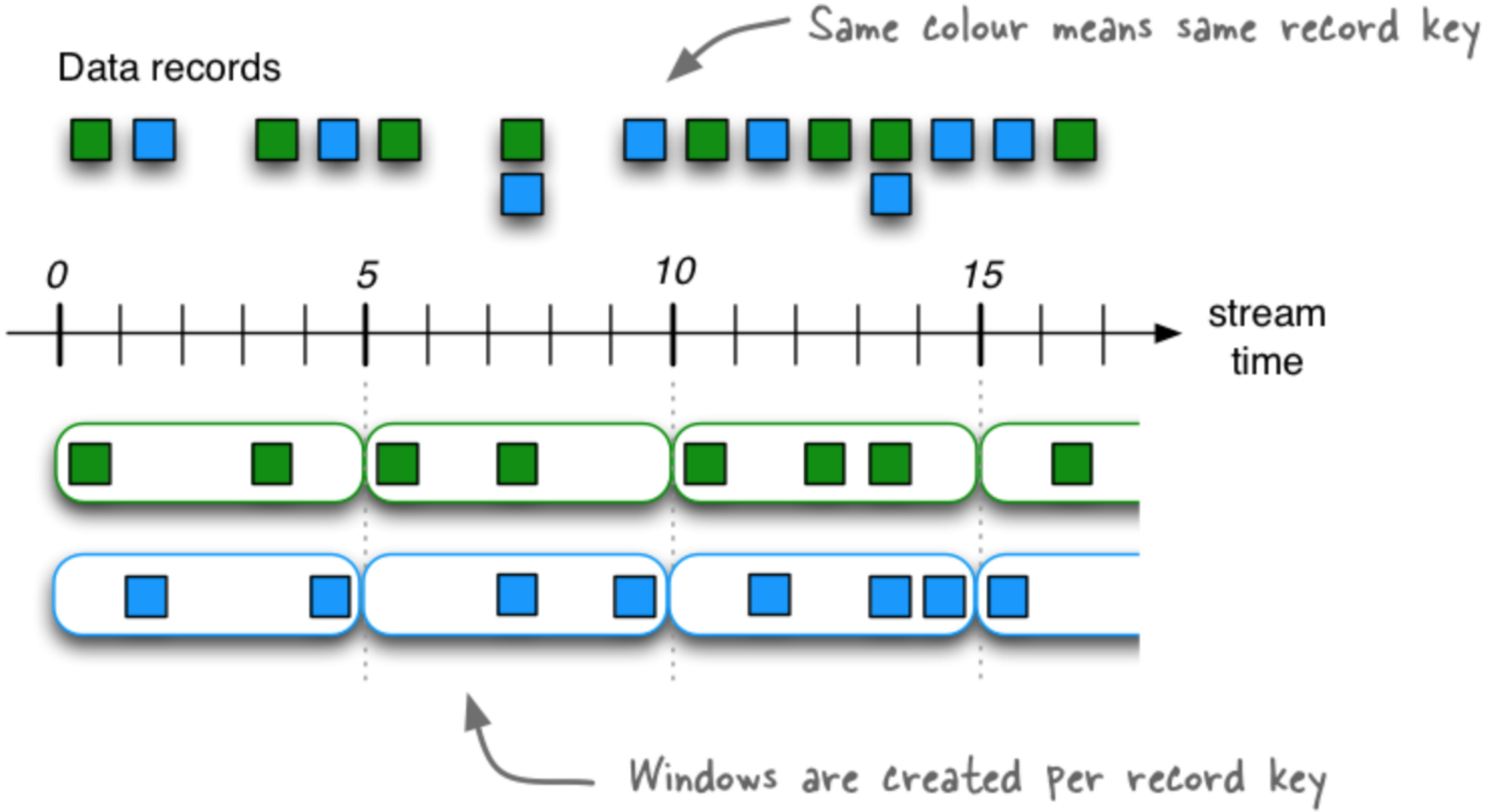
Apache Kafka



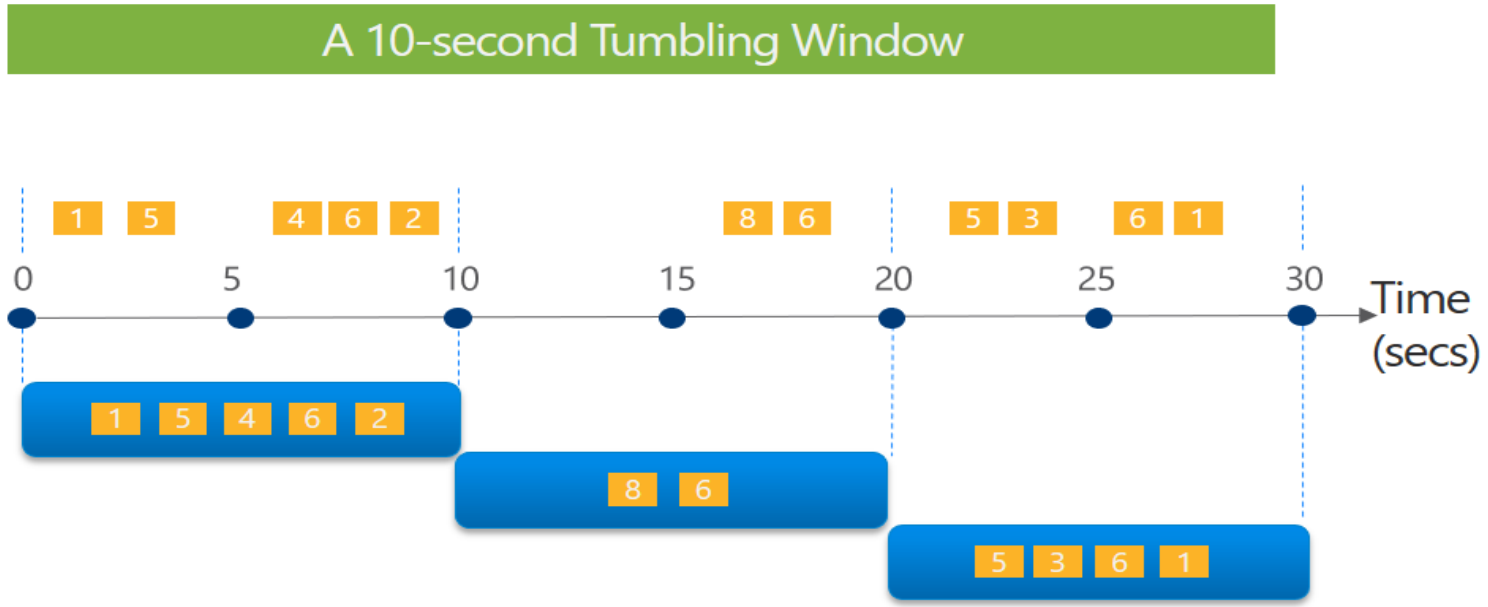
- https://kafka.apache.org
- Read and write streams of data like a messaging system
- Streaming data pipelines that reliably get data between systems or applications
- Streaming applications that transform or react to the streams of data
- Write scalable stream processing applications that react to events in real-time
- Store streams of data safely in a distributed, replicated, fault-tolerant cluster


Real-time Data Feeds
APACHE KAFKA
Online Data Processing
HBase
Accumulo
Real-time Stream Processing
Storm
Slider
Streaming Ingest
SQL
Hive
YARN
HDFS
HDP 2.2 Data Lake
Search
Solr
HDFS
HDP 2.2

Stream Windowing

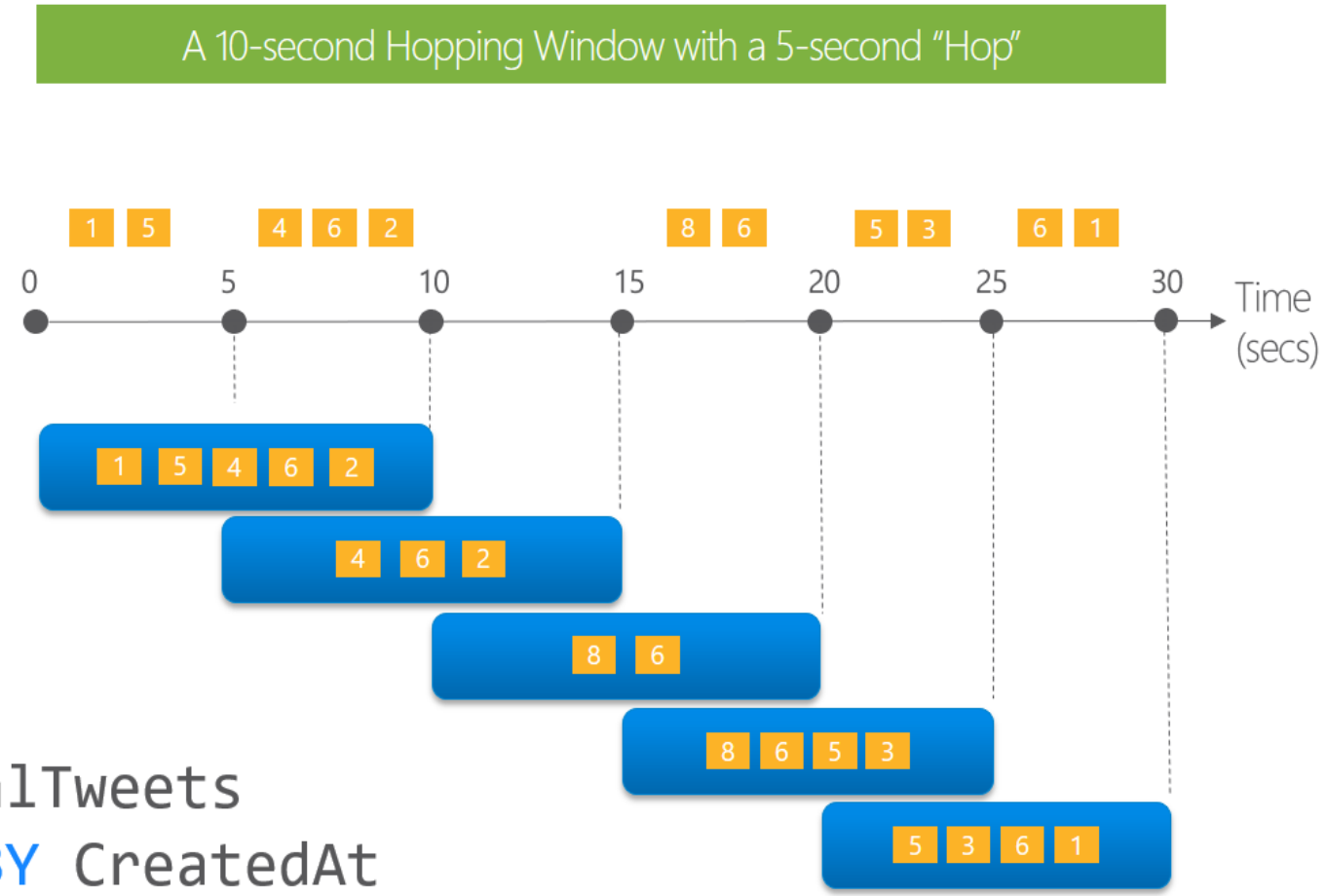


Tell me the count of tweets per time zone every 10 seconds




```
SELECT TimeZone, COUNT(*) AS Count
FROM TwitterStream TIMESTAMP BY CreatedAt
GROUP BY TimeZone, TumblingWindow(second,10)
```

Every 5 seconds give me the count of tweets over the last 10 seconds

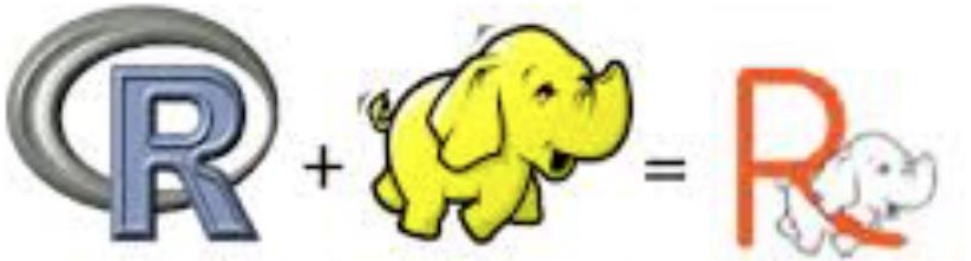



```
SELECT Topic, COUNT(*) AS TotalTweets
FROM TwitterStream TIMESTAMP BY CreatedAt
GROUP BY Topic, HoppingWindow(second, 10 , 5)
```

Apache Cassandra

- http://cassandra.apache.org
- https://www.youtube.com/watch?v=5qEoEAfAer8
- Orientado à família de colunas
- Tempo real e aplicações transacionais (online)
- Leitura intensiva para BI de grande escala
- Google BigTable + Amazon Dynamo → Facebook Cassandra
  - 2008/2010 – Apache Cassandra
- Nós Peer-to-peer simétricos
- Particionamento em todos nós (75k na Apple)
- Replicação customizada para garantir tolerância à falhas
- Projeto para leitura e escrita dinâmica (anywhere)
- Netflix, Adobe, Twitter, HP, IBM, Rackspace, Cisco,...
- Cassandra Query Language (CQL): SQL-like

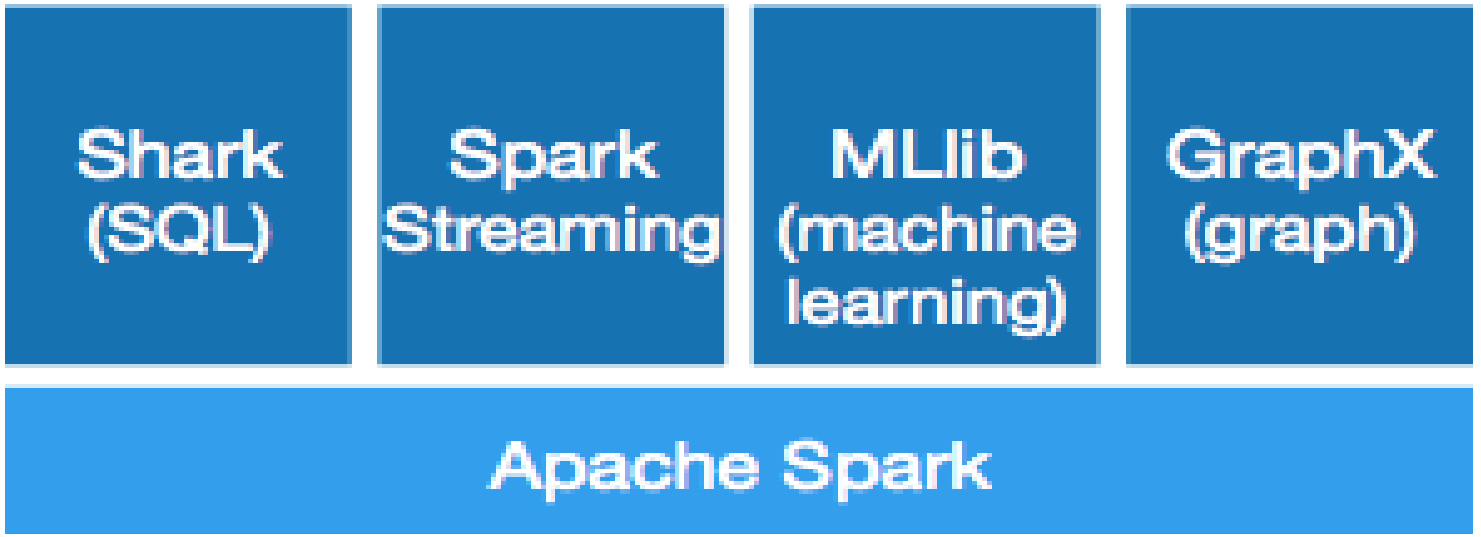
Estatística e Aprendizado de Máquina



R + Hadoop = RHadoop

Spark

- http://spark.apache.org/
- Execução em memória (100x mais rápido que o Hadoop)
- Aplicações em Java, Scala, Python e Spark shell



Data Frame

```
df=spark.read.format("com.databricks.spark.csv”)\
.option("header","true”).option("delimiter","\t")\
.option("inferSchema","true")\ .load("Diarias_utf8.csv") ;
df.count();
df.select("Nome Órgão Superior","Valor Pagamento").show();
df.filter(df["Nome Órgão Superior"]=="MINISTERIO DA CULTURA").count();
-- Verificar qual órgão teve maior soma de gastos.
df.groupBy("Nome Órgão Superior").sum("Valor Pagamento").show();
…
```

Spark WordCount em Python e Scala

- https://spark.apache.org/examples.html
- **Python**

```
text_file = sc.textFile("hdfs://…")
counts = text_file.flatMap(lambda line: line.split(" ")) \
             .map(lambda word: (word, 1)) \
             .reduceByKey(lambda a, b: a + b)
counts.saveAsTextFile("hdfs://...")
```

- **Scala**

```
file = spark.textFile("hdfs://...")
file.flatMap(line => line.split(" "))
    .map(word => (word, 1))
    .reduceByKey(_ + _)
```

Programming Guides

- Quick Start: a quick introduction to the Spark API; start here!
- RDD Programming Guide: overview of Spark basics - RDDs (core but old API), accumulators, and broadcast variables
- Spark SQL, Datasets, and DataFrames: processing structured data with relational queries (newer API than RDDs)
- Structured Streaming: processing structured data streams with relation queries (using Datasets and DataFrames, newer API than DStreams)
- Spark Streaming: processing data streams using DStreams (old API)
- MLlib: applying machine learning algorithms
- GraphX: processing graphs
- SparkR: processing data with Spark in R
- PySpark: processing data with Spark in Python
- Spark SQL CLI: processing data with SQL on the command line

Spark refs

- https://spark.apache.org/docs/latest/index.html
- https://spark.apache.org/docs/latest/ml-guide.html

- https://www.kaggle.com/code/kkhandekar/apache-spark-beginner-tutorial/notebook
- https://www.kaggle.com/code/tylerx/machine-learning-with-spark
- https://colab.research.google.com/#machine-learning-examples

Neo4J

- Banco de Dados orientado a Grafo, com suporte nativo
- ACID
- Schema
- Free
- Suporta replicação e distribuição
- Suporte a consultas em diversas linguagens de programação

Neo4J – Modelo de Dados

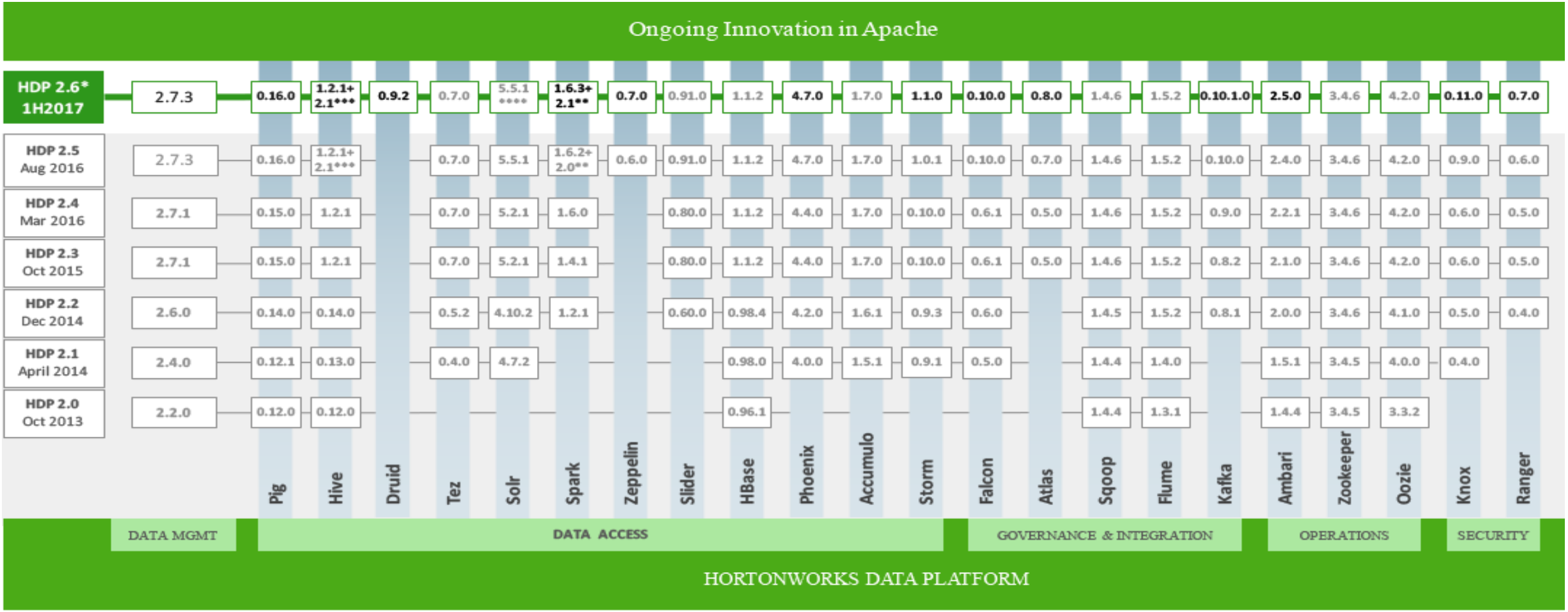
- Nó: representa e armazena uma entidade
  - Label
  - Equivalente a um tipo ou categoria
  - Propriedades
  - Coleção de chave : valor
- Relação: faz a ligação entre nós
  - Label
  - Propriedades
  - Coleção de chave : valor

```
create(
Mauro:Usuario {Nome :
'Mauro', Cidade :
'Curitiba', Filhos : 'Sim'})
Create(Raquel:Usuario{No
me:'Raquel', Cidade:'Prude
ntopolis', Filhos:'Sim'}) ,
(Lorenzo:Usuario{Nome:'Lo
renzo', Cidade:'Curitiba
', Filhos:'Não'}),
(Augusto:Usuario{Nome:'A
ugusto', Cidade:'Curitiba', F
ilhos:'Não'})
```

Onde:◦Mauro : um nome local de variável◦Usuario:Label“tipo”◦{chave:valor, .. } propriedades

Onde:◦Mauro : um nome local de variável◦Usuario:Label“tipo”◦{chave:valor, .. } propriedades

HDP 2.6



* *HDP 2.6 – Shows current Apache branches being used. Final component version subject to change based on Apache release process.*
** *Spark 1.6.3+ Spark 2.1 – HDP 2.6 supports both Spark 1.6.3 and Spark 2.1 as GA.*
*** *Hive 2.1 is GA within HDP 2.6.*
**** *Apache Solr is available as an add-on product HDP Search.*

UFPR

Especialização em Inteligência Artificial

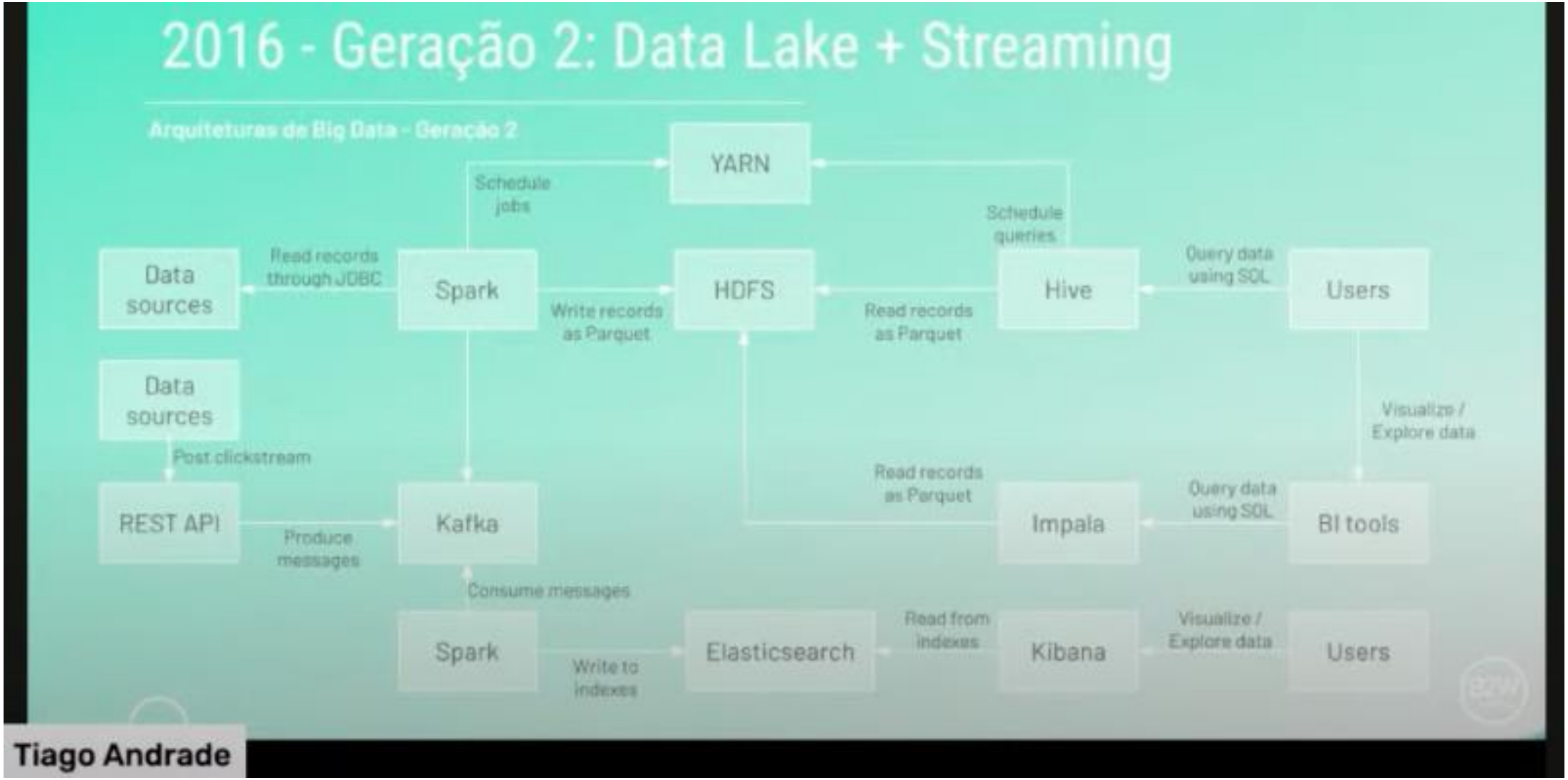
SEPT

2ª CiDWeek - 30/04 à tarde - https://youtu.be/nFYMhG6jPHk

B2W (Americanas, Submarino, Shoptime, SouBarato, Bit, Ame, Let's e Now)

2ª CiDWeek - 30/04 à tarde - https://youtu.be/nFYMhG6jPHk

2ª CiDWeek - 30/04 à tarde - https://youtu.be/nFYMhG6jPHk



Atividade 10 – Estudo de Caso

- Enviar um arquivo PDF contendo uma descrição breve (2 páginas) sobre a implementação de uma aplicação ou estudo de caso envolvendo big data e suas ferramentas (NoSQL/Streaming).
  - Caracterizar os dados e seus Vs, e sobre a modelagem

Apresentação (5 min)

IAA013 - Big Data

- Fundamentos de Big Data
  - Data Lake e Data Science
- Map Reduce e Hadoop
  - Utilização da Sandbox/VM
  - Personalização de aplicações Map Reduce
- Data Engineering (Gerenciamento/Ferramentas para Big Data)
- NoSQL e NewSQL
- Dados em movimento – Processamento de Streaming