Sistemas de Procesamiento Batch Hadoop

FUNDAMENTOS TECNOLÓGICOS PARA BIG DATA

Isaac Esau Rubio Torres

2018

Contenido

- Introducción al Big Data
- Apache Hadoop
- Arquitectura de Hadoop
- Paradigma MapReduce

- Ejemplos

¿Qué es Big Data?

"Big Data is the frontier of a firm's ability to store, process, and access (SPA) all the data it needs to operate effectively, make decisions, reduce risks, and serve customers."

Forrester

¿Qué es Big Data?

"Big Data in general is defined as high volume, velocity and variety information assets that demand cost-effective, innovative forms of information processing for enhancedinsight and decision making.

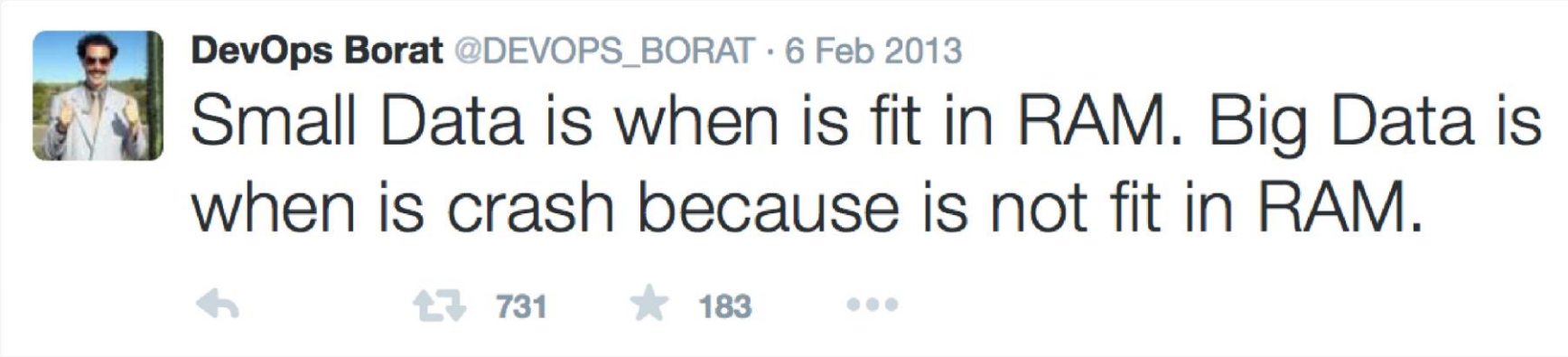
Gartner

¿Qué es Big Data?

"Big Data is data that exceeds the processing capacity of conventionaldatabasesystems. The data is too big, moves too fast, or doesn't fit the strictures of your databasearchitectures. To gain value from this data, you must choose an alternative way to process it."

O'Reilly

¿Qué es Big Data?



¿Qué es Big Data?

¿Qué es el Big Data?

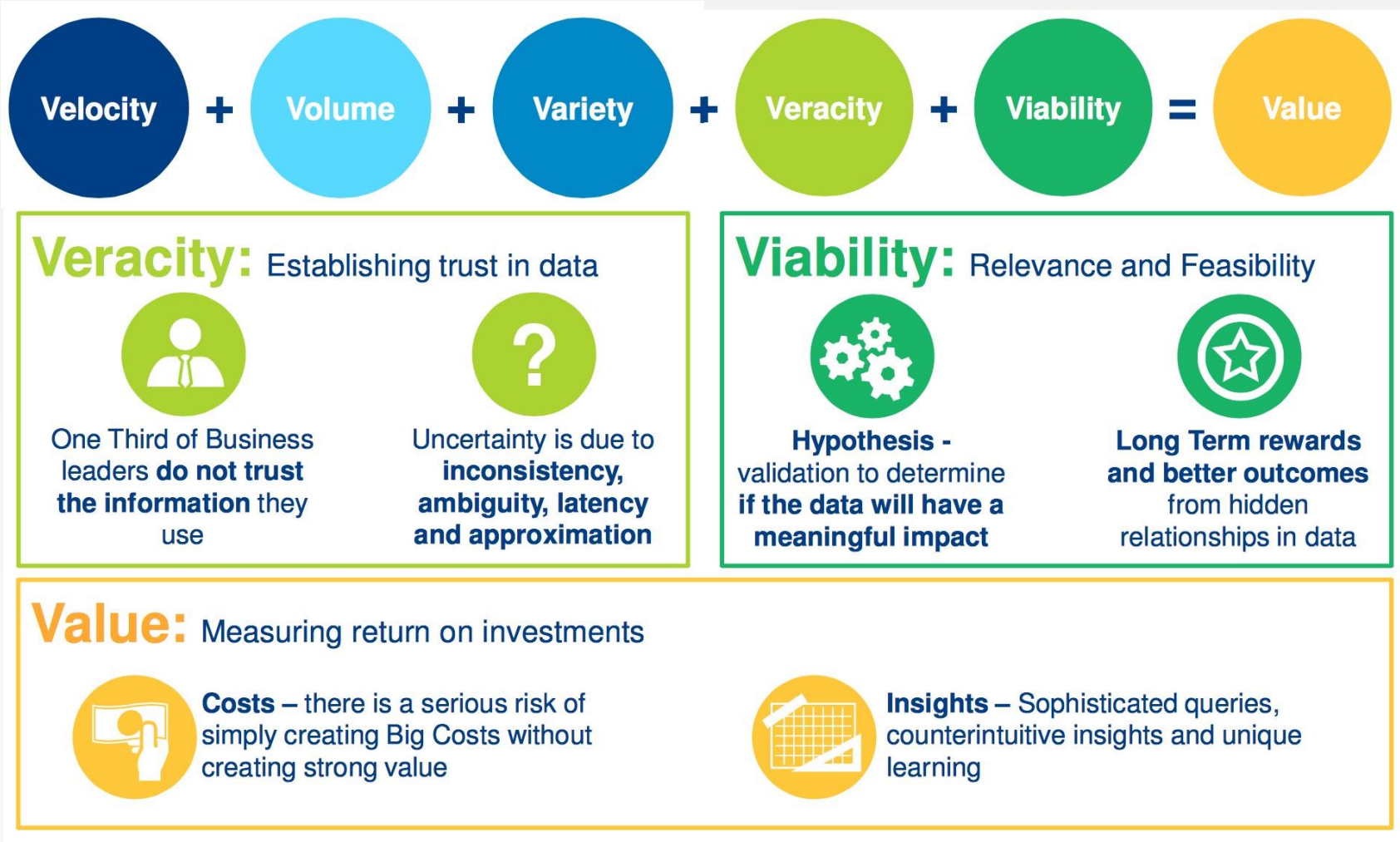
"**Data** is the new **bacon**"

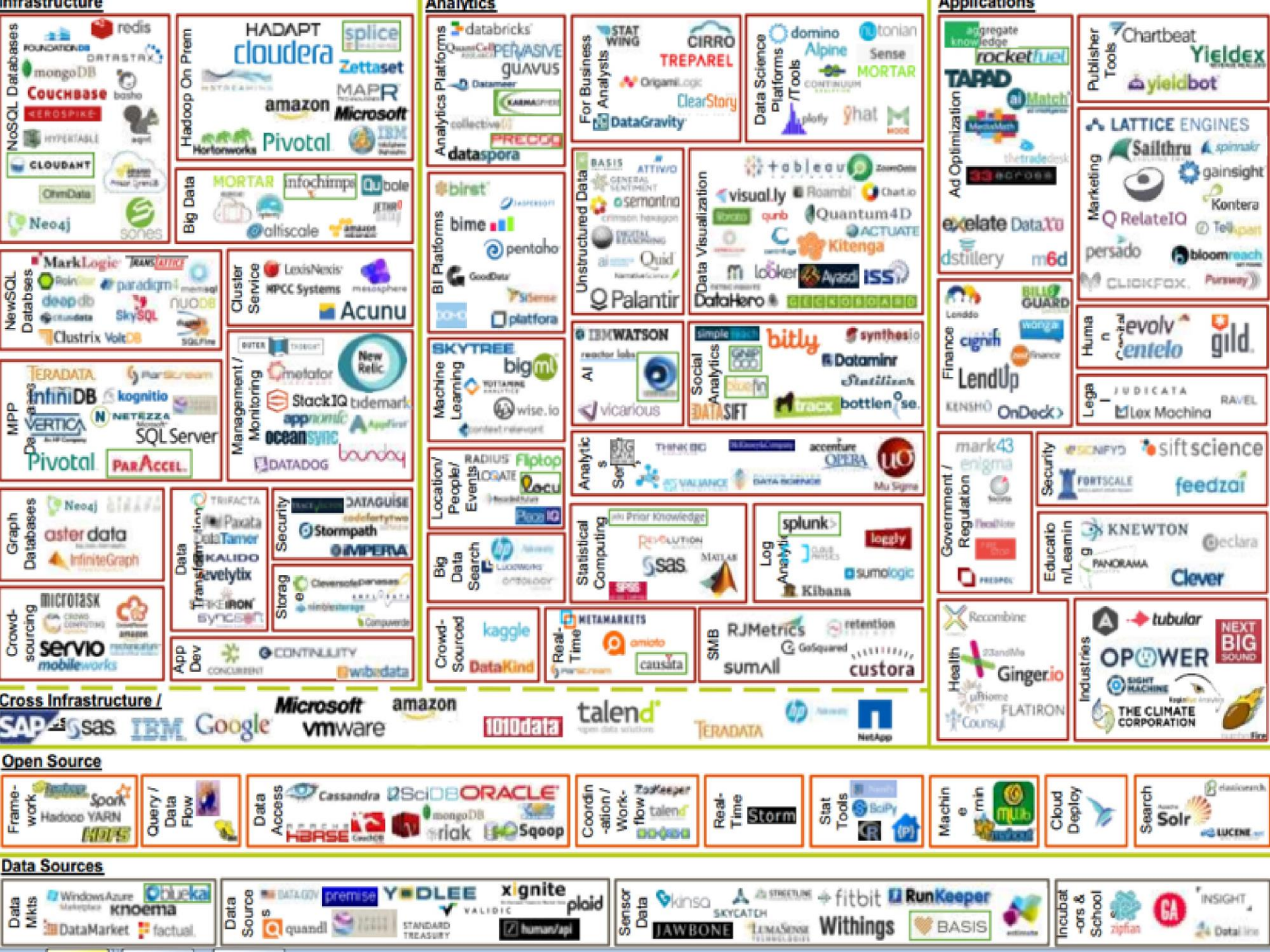
Cloudera

Big Data es…

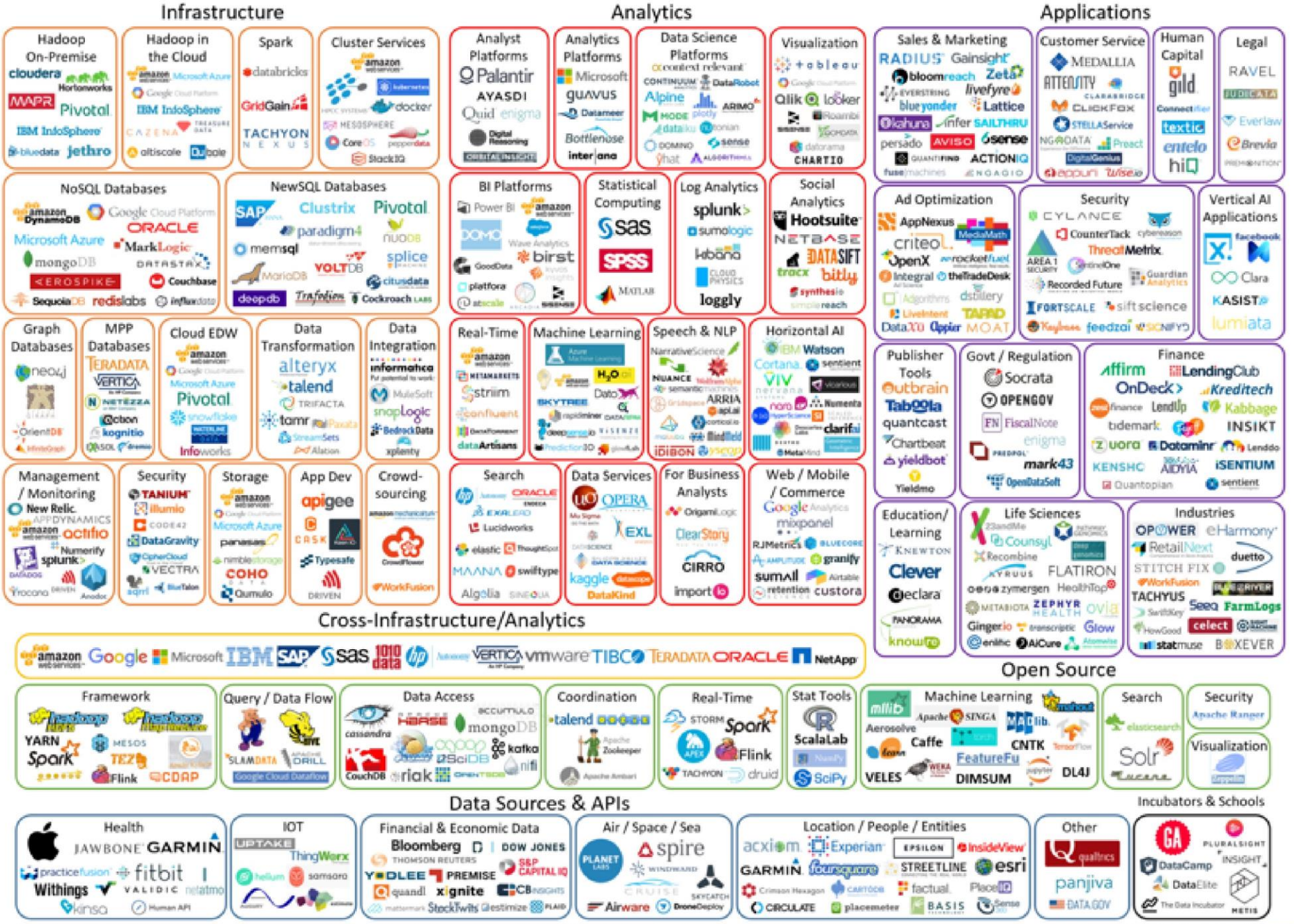
- 3Vs
- Procesar muchos datos
- Son herramientas para redes sociales
- Son datos no estructurados
- Hadoop
- Una moda
- Un problema

Muchas Vs




Infrastructure
NoSQL Databases
redis
FOUNDATIONDB
DATASTAX
mongoDB
Couchbase
basho
AEROSPIKE
HYPERTABLE
aspril
CLOUDANT
OrientDB
Neo4j
sones
Hadoop On Prem
HADAPT
cloudera
splice
Zettaset
MAPR
amazon
Microsoft
Hortonworks
Pivotal
IBM
Big Data
MORTAR
infochimps
Qubole
altiscale
NewSQL Databases
MarkLogic
TRANSACTIVE
RAIMA
paradigm4
memsql
deepdb
SkySQL
nuoDB
Clustrix
VoltDB
SQLFire
Cluster Service
LexisNexis
HPCC Systems
mesosphere
Acunu
MPP Data Warehouse
TERADATA
ParStream
InfiniDB
kognitio
VERTICA
NETEZZA
SQL Server
Pivotal
ParAccel
Management / Monitoring
OUTER THOUGHT
metafor
New Relic
StackIQ
tidemark
appnomic
AppFirst
oceansync
boundary
DATADOG
Graph Databases
Neo4j
aster data
InfiniteGraph
Data Transformation
TRIFACTA
Paxata
DataTamer
KALIDO
Pervasive
IRON
SYNCSORT
Security
DATAGUISE
codefortytwo
Stormpath
IMPERVA
Storage
Cleversafe
nimbleStorage
Compuverde
Crowd-sourcing
microtask
CROWD COMPUTING
CrowdFlower
servio
mobileworks
App Dev
CONCURRENT
CONTINUITY
wibidata
Analytics
Analytics Platforms
databricks
QuantCell
PERVASIVE
guavus
Datameer
KARMASPHERE
collective[i]
PRECOG
dataspora
BI Platforms
birst
bime
pentaho
GoodData
SiSense
platfora
SKYTREE
Machine Learning
bigml
TOTAMINE ANALYTICS
wise.io
Location / People / Events
RADIUS
Fliptop
LOCATE
Locu
PlaceIQ
Big Data Search
Lucid Works
ONTOLOGY
Crowd-Sourced
kaggle
DataKind
Real-Time
METAMARKETS
metamarkets
amoto
causata
For Business Analysts
STAT WING
CIRRO
TREPAREL
Origami Logic
ClearStory
DataGravity
Data Science Platforms / Tools
domino
tonian
Alpine
Sense
MORTAR
CONTINUUM
plotly
yhat
MODE
Unstructured Data
BASIS
ATTIVIO
GENERAL SENTIMENT
semantria
crimson hexagon
DIGITAL REASONING
Quid
Palantir
Data Visualization
tableau
ZoomData
visual.ly
Roambi
Chart.io
Quantum4D
ACTUATE
Kitenga
looker
Ayasdi
ISS
DataHero
GEOMERATIVE
AI
IBM WATSON
reactor labs
vicarious
Social Analytics
simple reach
GNIP
bitly
synthesio
Datominr
Statilytics
DATASIFT
tracx
bottlenose
Analytics Services
BIG DATA SPECIALISTS
THINK BIG
accenture
OPERA
VALIANCE
DATA SCIENCE
Mu Sigma
Statistical Computing
Prior Knowledge
REVOLUTION
sas
MATLAB
spss
Log Analytics
splunk
loggly
sumologic
Kibana
SMB
RJMetrics
retention
GoSquared
sumall
custora
Applications
Ad Optimization
aggregate knowledge
rocketfuel
TAPAD
ai Match
MediaMath
thetradedesk
33across
exelate
DataXu
dstillery
m6d
Finance
Londdo
BILLGUARD
cignifi
wonga
zeta finance
LendUp
KENSHO
OnDeck
Government / Regulation
mark43
enigma
RealNote
PREDPOL
Health
Recombine
23andMe
Ginger.io
uBiome
Counsyl
FLATIRON
Publisher Tools
Chartbeat
Yieldex
yieldbot
Marketing
LATTICE ENGINES
Sailthru
spinnakr
gainsight
Kontera
RelateIQ
TellApart
persado
bloomreach
CLICKFOX
Pursway
Human Capital
evolv
centelo
gild
Legal
JUDICATA
Lex Machina
RAVEL
Security
SECNFYD
siftscience
FORTSCALE
feedzai
Education/Learning
KNEWTON
eclara
PANORAMA
Clever
Industries
tubular
NEXT BIG SOUND
OPOWER
SIGHT MACHINE
THE CLIMATE CORPORATION
Cross Infrastructure /
SAP
sas
IBM
Google
Microsoft
vmware
amazon
1010data
talend
TERADATA
NetApp
Open Source
Frame-work
Spark
Hadoop YARN
Query / Data Flow
Data Access
Cassandra
SciDB
ORACLE
mongoDB
riak
Sqoop
HBase
Coordin-ation / Work-flow
ZooKeeper
talend
Real-Time
Storm
Stat Tools
SciPy
Machine Learning
mahout
Cloud Deploy
Search
Solr
LUCENE
Data Sources
Data Mkts
Windows Azure Marketplace
Kaggle
Knoema
DataMarket
factual
Data Sources
DATA.GOV
premise
YODLEE
xignite
plaid
quandl
STANDARD TREASURY
VALIDIC
human/api
Sensor Data
kinsa
SKYCATCH
fitbit
RunKeeper
JAWBONE
LUMOSITY
Withings
BASIS
Incubat-ors & School
zipfian
GA
INSIGHT
Data Incubator

Big Data Landscape 2016 (Version 3.0)



FIRSTMARK

¿Big Data = Hadoop?

Big data is coming…

From a Molehill To a Mountain

A timeline of digital data

Everyone knows that the amount of digital data has exploded in recent years. But it isn't always easy to put today's mountain of data in perspective.

To illustrate the data boom, here's a timeline with snapshots of some of the largest corporate data collections from each decade since the Univac computer was the hot new thing. —*Matthew Kassel*

2010s | **100 Petabytes**
Facebook Is Facebook's data hoard bigger than Google's today? Some say yes, some say no. According to Facebook, its user content makes up more than 100 petabytes of stored photos and video. And analyzing that data generates about 500 terabytes of new information every day—more than 2½ times the size of a '90s Wal-Mart data cache.

2000s | **25 Petabytes**
Google The explosion of the Internet in the '90s set the stage for Web-based companies to emerge in the following decade as the global leaders in big data.

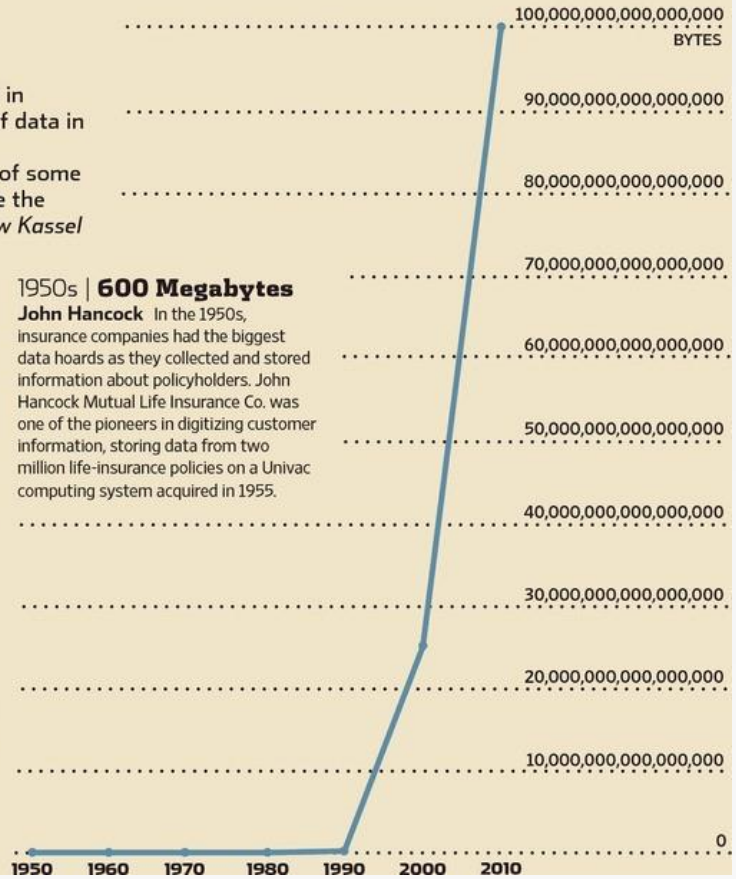
1990s | **180 Terabytes**
Wal Mart Wal-Mart in this decade became the largest American bricks-and-mortar retail operation, and, it is believed, had the biggest commercial data warehouse in the world. To put this number in perspective, experts have estimated that Amazon.com in the late '90s had single-digit terabytes of stored data.

1980s | **450 Gigabytes**
CitiCorp's NAIB In the 1980s, banks were at the forefront of data growth, with ATMs coming into vogue and banks focusing on collecting and analyzing transaction data for all their businesses.

1970s | **80 Gigabytes**
Federal Express Cosmos FedEx's Cosmos system, introduced in the 1970s, allowed the company to scan and track its huge volume of packages being shipped around the world.

1960s | **807 Megabytes**
American Airlines Sabre In the 1960s, American Airlines developed Sabre, a flight-reservation system built around one of the largest IBM computing systems available. Sabre was one of the first online computing systems, allowing the airline to keep track of an immense matrix of reservations, flight schedules and seat inventories.

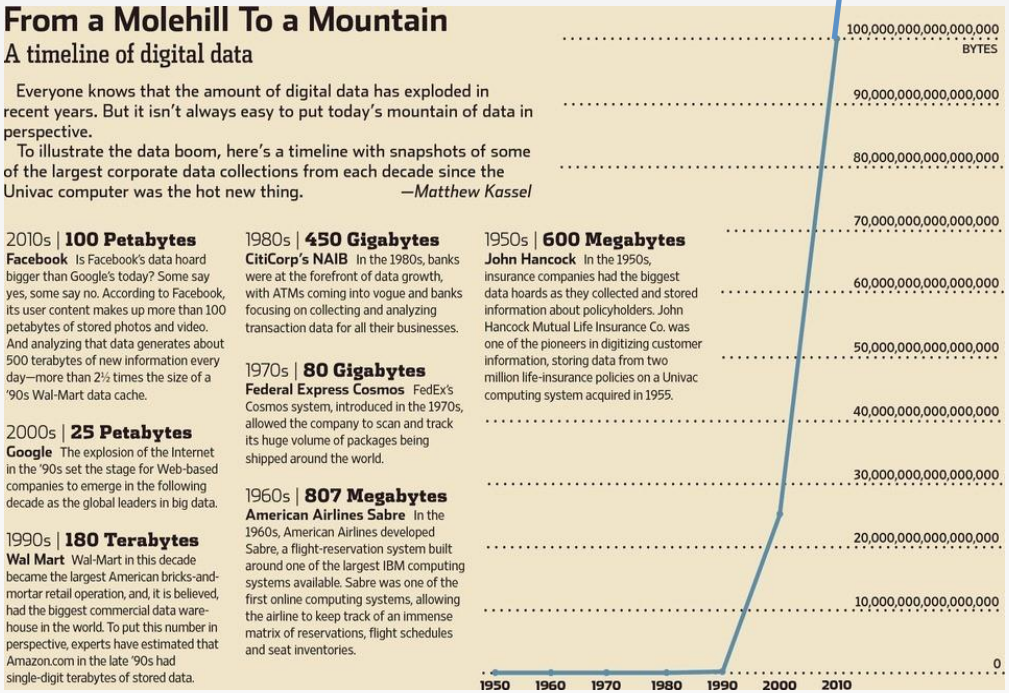
1950s | **600 Megabytes**
John Hancock In the 1950s, insurance companies had the biggest data hoards as they collected and stored information about policyholders. John Hancock Mutual Life Insurance Co. was one of the pioneers in digitizing customer information, storing data from two million life-insurance policies on a Univac computing system acquired in 1955.



http://online.wsj.com/news/articles/SB10001424127887324178904578340071261396666

Big data is coming…

Ciencia
Industria
Logs de sistema
…



From a Molehill To a Mountain

A timeline of digital data

Everyone knows that the amount of digital data has exploded in recent years. But it isn't always easy to put today's mountain of data in perspective.

To illustrate the data boom, here's a timeline with snapshots of some of the largest corporate data collections from each decade since the Univac computer was the hot new thing. —*Matthew Kassel*

2010s | **100 Petabytes**
Facebook Is Facebook's data hoard bigger than Google's today? Some say yes, some say no. According to Facebook, its user content makes up more than 100 petabytes of stored photos and video. And analyzing that data generates about 500 terabytes of new information every day—more than 2½ times the size of a '90s Wal-Mart data cache.

2000s | **25 Petabytes**
Google The explosion of the Internet in the '90s set the stage for Web-based companies to emerge in the following decade as the global leaders in big data.

1990s | **180 Terabytes**
Wal Mart Wal-Mart in this decade became the largest American bricks-and-mortar retail operation, and, it is believed, had the biggest commercial data warehouse in the world. To put this number in perspective, experts have estimated that Amazon.com in the late '90s had single-digit terabytes of stored data.

1980s | **450 Gigabytes**
CitiCorp's NAIB In the 1980s, banks were at the forefront of data growth, with ATMs coming into vogue and banks focusing on collecting and analyzing transaction data for all their businesses.

1970s | **80 Gigabytes**
Federal Express Cosmos FedEx's Cosmos system, introduced in the 1970s, allowed the company to scan and track its huge volume of packages being shipped around the world.

1960s | **807 Megabytes**
American Airlines Sabre In the 1960s, American Airlines developed Sabre, a flight-reservation system built around one of the largest IBM computing systems available. Sabre was one of the first online computing systems, allowing the airline to keep track of an immense matrix of reservations, flight schedules and seat inventories.

1950s | **600 Megabytes**
John Hancock In the 1950s, insurance companies had the biggest data hoards as they collected and stored information about policyholders. John Hancock Mutual Life Insurance Co. was one of the pioneers in digitizing customer information, storing data from two million life-insurance policies on a Univac computing system acquired in 1955.

http://online.wsj.com/news/articles/SB10001424127887324178904578340071261396666

Buscar/crear la herramienta adecuada…

Breve historia

- Actualmente es un proyecto de código abierto bajo licencia Apache.
  - Su licencia ha facilitado que sea adoptado por un importante número de empresas.
- El apoyo de IBM, Oracle, EMC, HP, etc. ha acelerado su implantación y su mejora en prestaciones.
  - Gran uso en ñproyectos tipo *big data*.
  - JPMorgan Chase: "We're hiring, and we're paying 10% more Than the otherguys."

http://www.informationweek.com/software/information--management/its--next--hot--job--hadoop--guru/d/d--id/1101209?

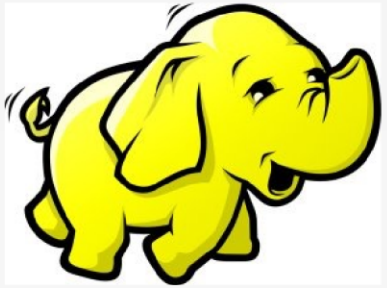
http://hadoop.apache.org

- Apache Hadoop es un proyecto software *open source* para computación distribuida escalable y de confianza (*reliable*).
- Ofrece un frameworkque permite la computación distribuida de grandes conjuntos de datos mediante clusters de ordenadores usandomodelos de programación simples.
  - Diseñado para poder pasar de un solo servidor a miles de máquinas (scale--up), donde cada una ofrece tanto computación como almacenamiento.
  - El framework está diseñado para detectar y tratar fallos a nivel de aplicación. Esto permite ofrecer servicios con alta disponibilidad sobre un cluster de ordenadores (aunque estos tenganfallos).

http://hadoop.apache.org/

Características generales

- Open Source
- Desarrollado originalmente por Yahoo
- Administrado por Apache Software Foundation
- Diseñado para trabajar con petabytes de datos
- Pensado para implementarse con hardware económico
- Ofrece alta disponibilidad
- Escala horizontalmente
- Muchas tecnologías de desarrollo están basadas en Hadoop
- Bueno aceptación en el mercado
- Curva de aprendizaje elevada
- No es una base de datos
- No es real time



¿Quiénes usan Hadoop?

Big Opportunities…

http://www.indeed.com/jobtrends?q=Big--data&relative=1

Hadoop

En Google:

- Construcción de índices para el buscador (pagerank)
- Clustering de artículos en Google News
- Búsqueda de rutas en Google Maps
- Traducción estadística

En Facebook:

- Minería de datos
- Optimización de ads
- Detección de spam
- Gestión de logs

En investigación:

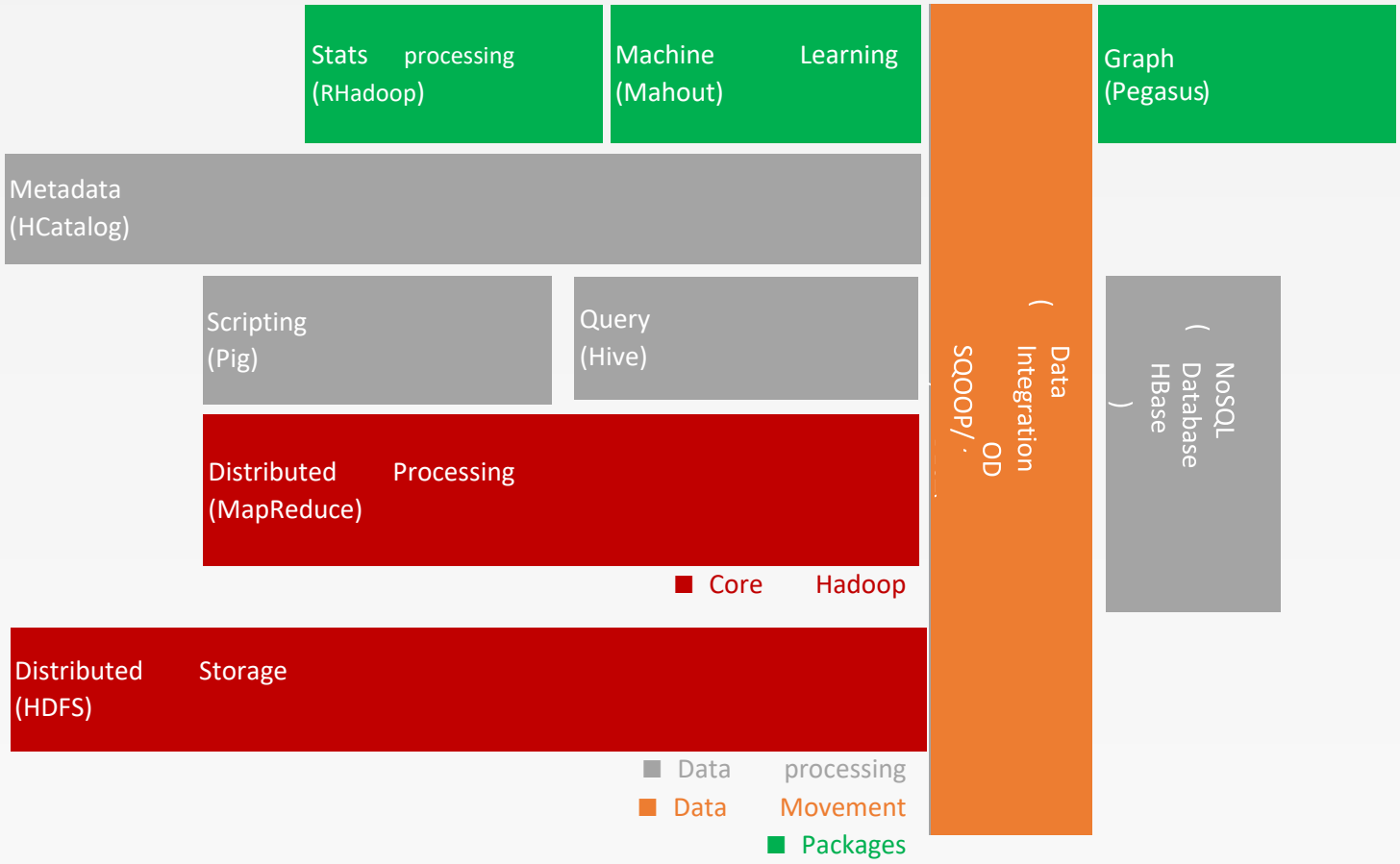
- Análisis astronómico, bioinformática, física de partículas, simulación climática, procesamiento del lenguaje natural, ...

Componentes principales

Hadoop se compone por tres elementos principales

- HDFS
- MapReduce
- Hadoop Common

Componentes principales



Map Reduce

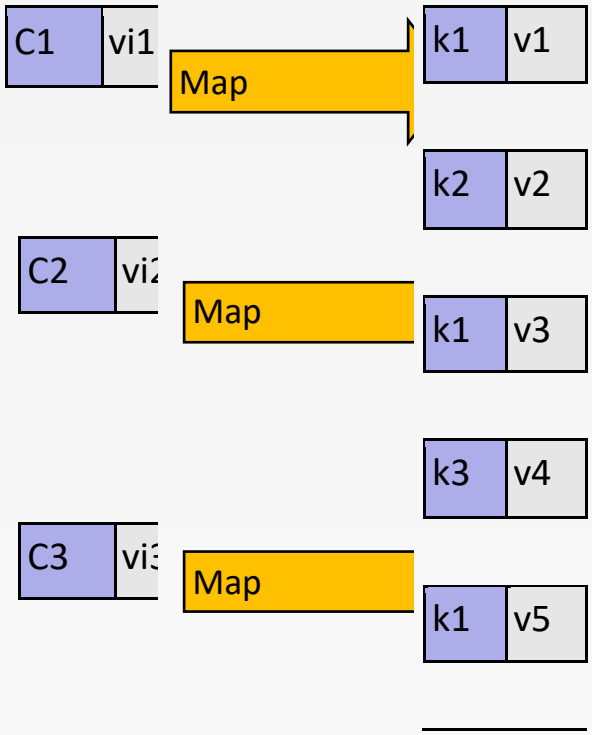
- Creado por Doug Cutting Google lo introdujo en 2004 Consiste en la ejecución de dos procesos separados,
- Map y Reduce
- Paralelismo
- Escalabilidad
- Tolerancia a fallos
- Curva de aprendizaje elevada

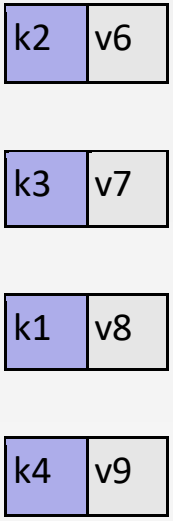
Map Reduce

- Hadoop proporciona un entorno de ejecución orientado a aplicaciones desarrolladas bajo el modelo de programación MapReduce. Bajo este modelo, la ejecución de una aplicación presenta dos etapas:
  - Map: donde se realiza la ingestión y la transformación de los datos de entrada, en la cual los registros de entrada pueden ser son procesados en paralelo.
  - Reduce: fase de agregación o resumen, donde todos los registros asociados entre sí deben ser procesados juntos por una misma entidad.

MapReduce - Map

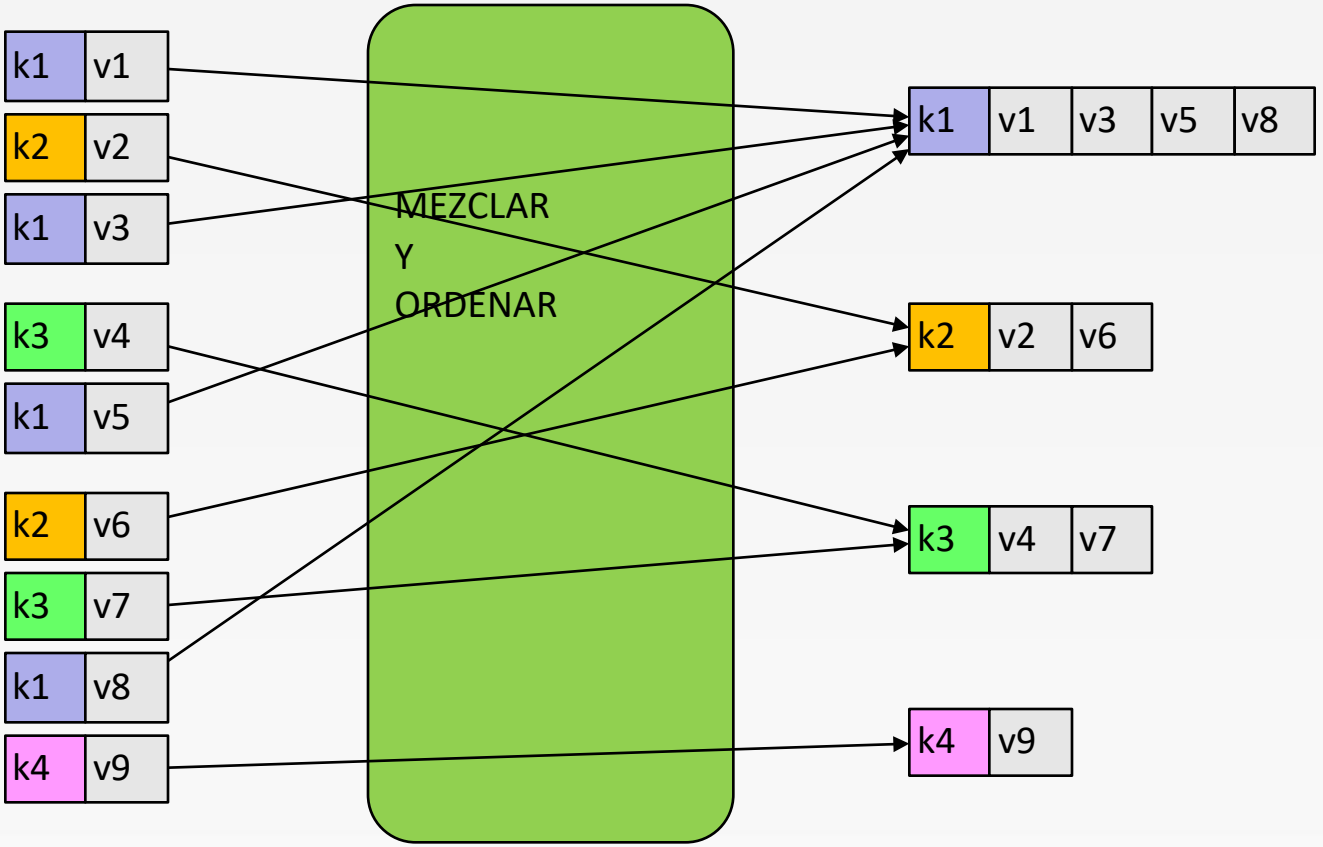
- Recibe como entrada un par (clave,valor) y recupera como salida uno o varios pares (clave-i, valor-i)




k2
v6
k3
v7
k1
v8
k4
v9

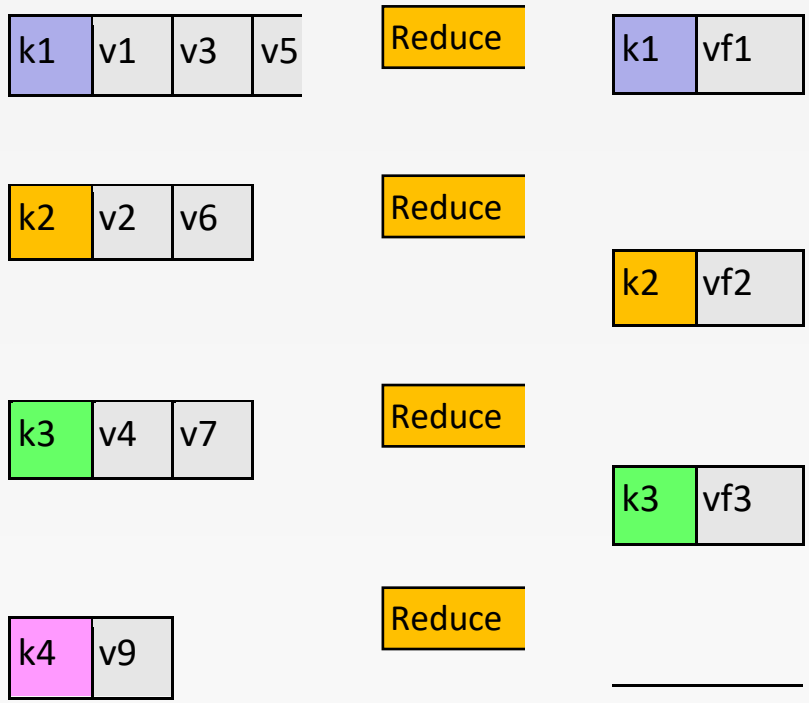
Map-reduce

- Para cada (clave1, valor1) de entrada recupera una lista de (clave2, valor2)



MapReduce - Reduce

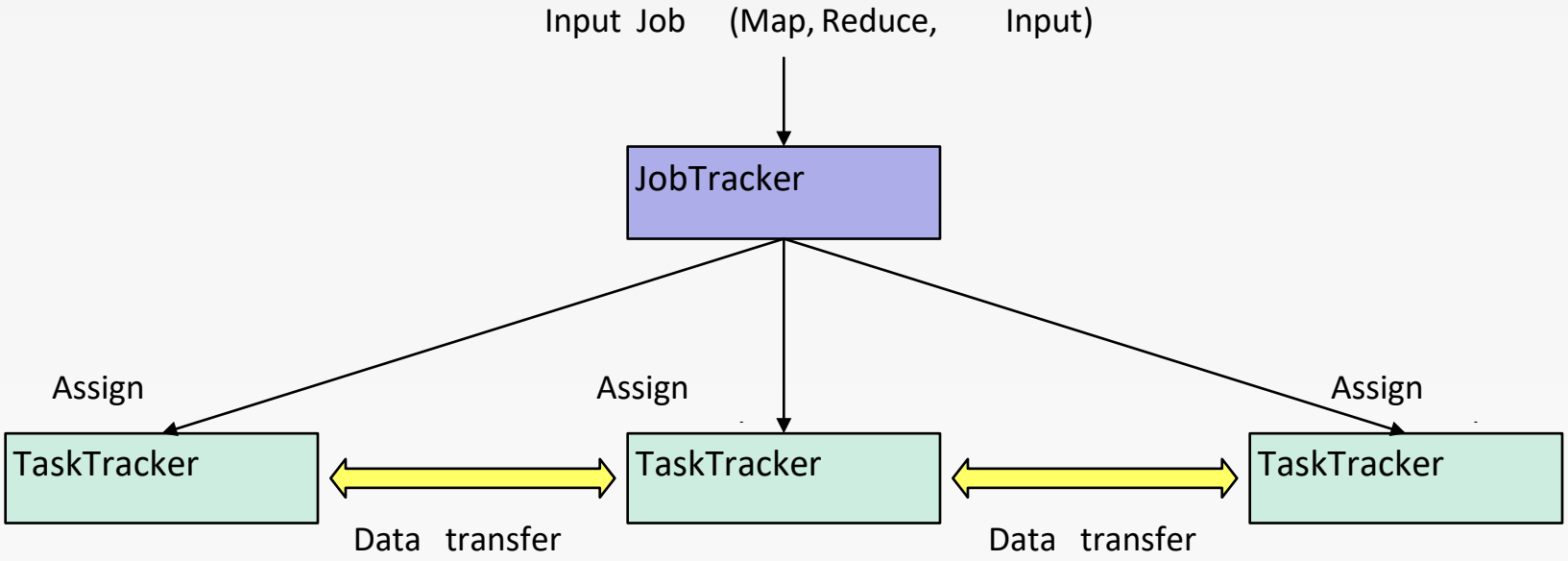
- Recibe como entrada un par (clave,lista de valores) y recupera como salida un **único** par (clave,valor)



k4	vf4

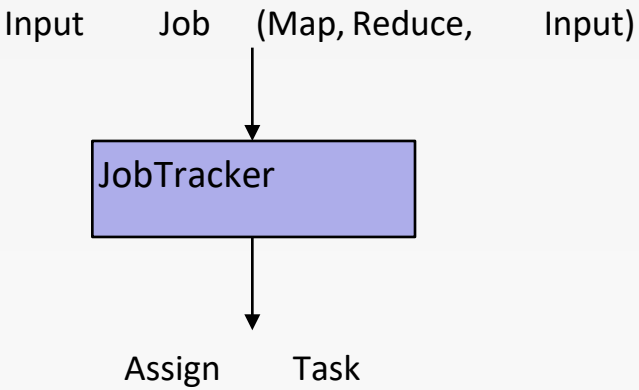
Hadoop - Arquitectura

- Es un componente de Hadoop Lee y escribe
- sobre el sistema de archivos de Hadoop (HDFS)



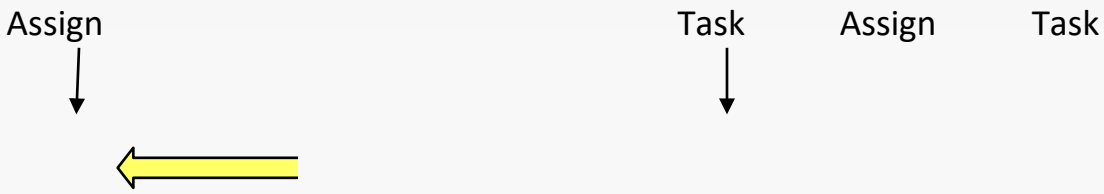
Hadoop - Arquitectura

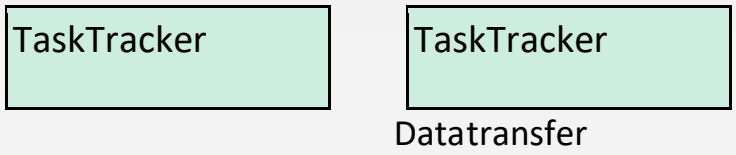
- JobTracker: Planificador de tareas
  - Registra los trabajos pendientes
  - Asigna las tareas a los nodos
  - Mantiene los trabajos cerca de los nodos
  - Si falla el JobTracker los trabajos pendientes de ejecución se pierden



Hadoop - Arquitectura

- TaskTracker
  - Se llaman TaskTrackers a los nodos
  - Atienden operaciones de Map y Reduce
  - Tienen slots asignados para Map y para Reduce
  - Controla las tareas en ejecución
  - Notifica al JobTracker acerca del estado del nodo y las tareas
  - Si un TaskTracker falla o se produce un timeout, esa parte del trabajo se planifica nuevamente

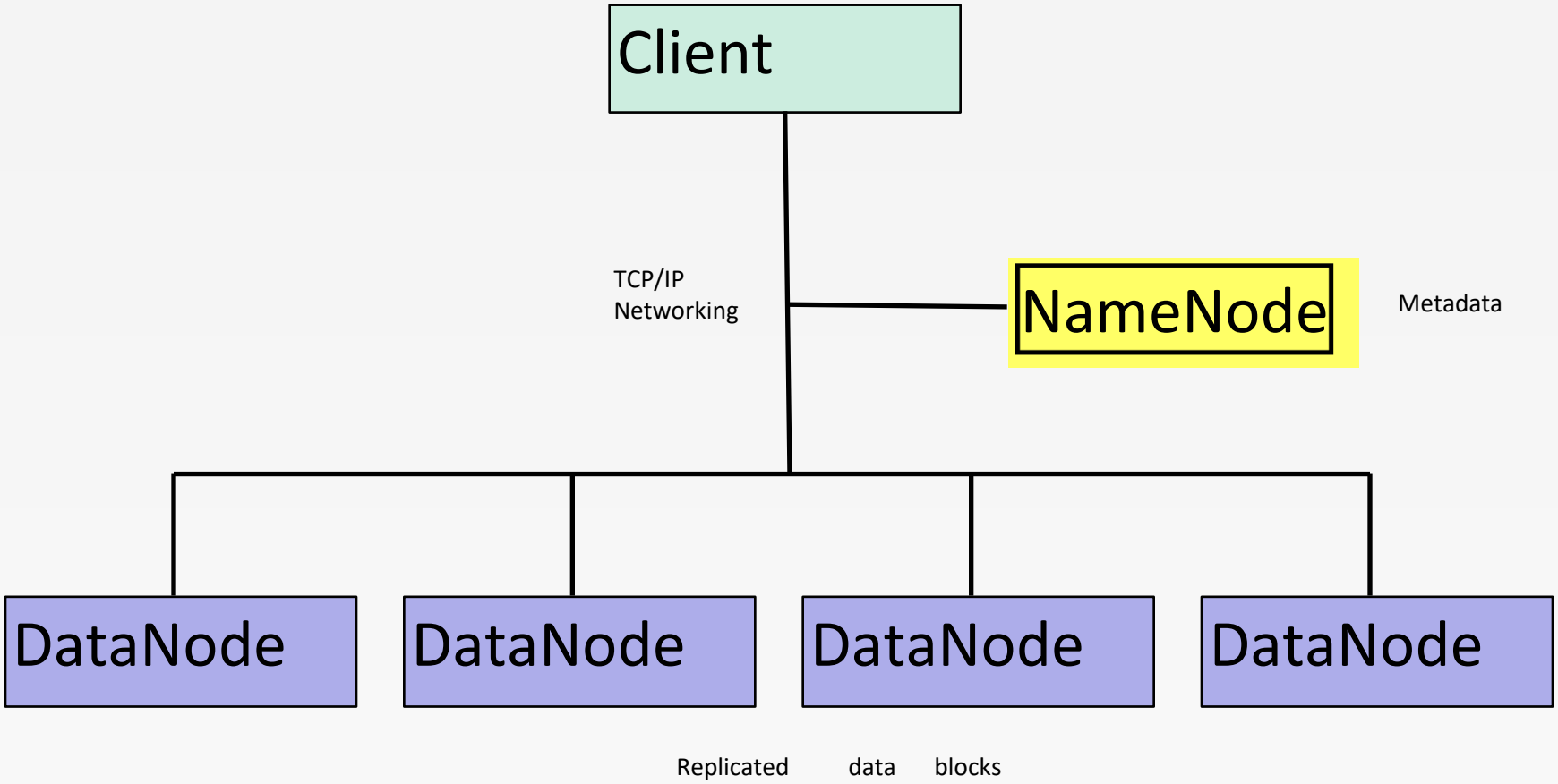




HDFS

- Significa Hadoop Distributed File System Es el sistema de archivos por defecto de Hadoop
- Inspirado en GFS Estructurado en bloques (típicamente 64 MB o 128 MB por bloque)
- Rebalanceo de bloques
- Escalabilidad
- Disponibilidad
- Modelo de seguridadPOSIX

HDFS - Arquitectura



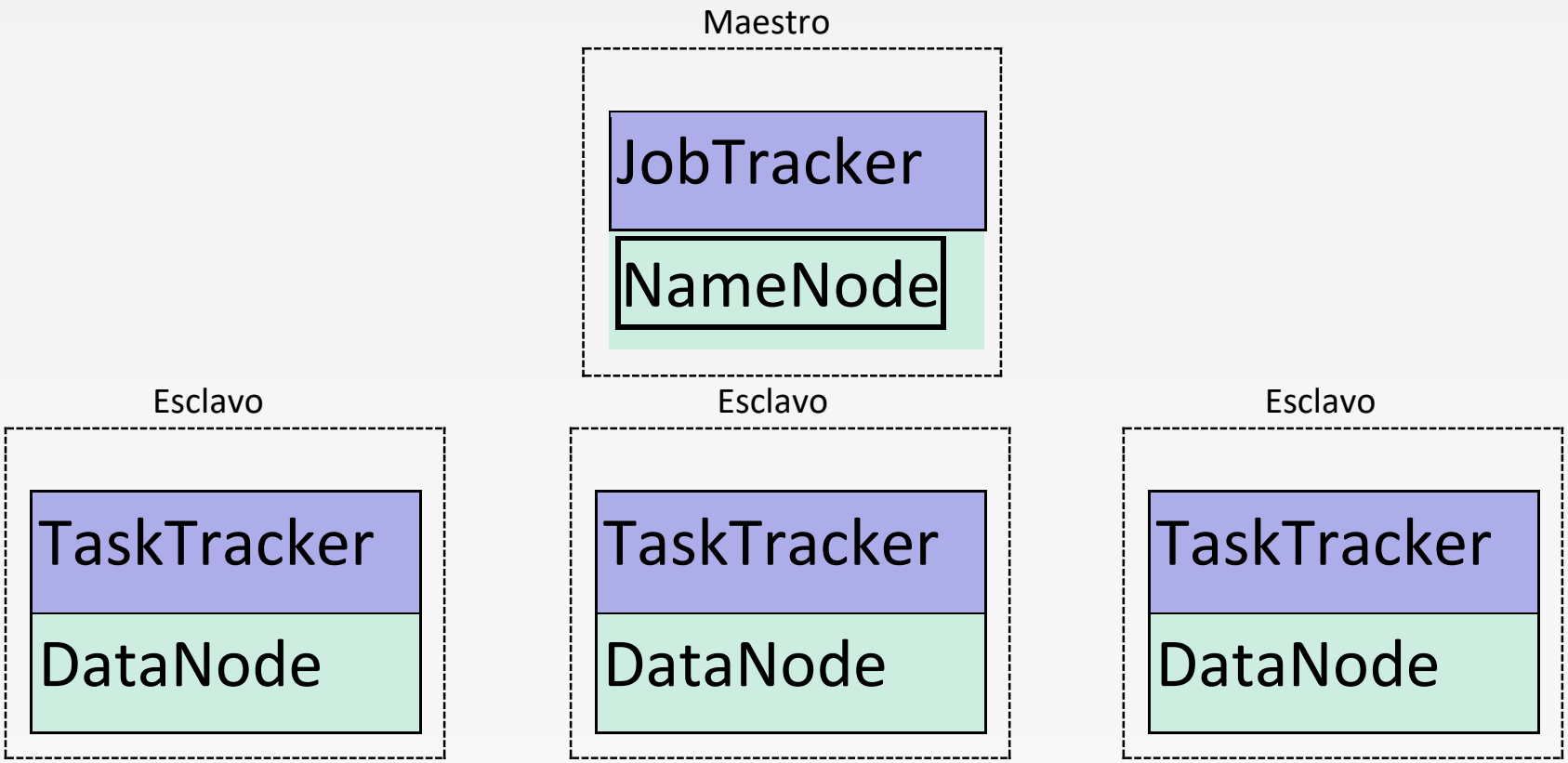
HDFS - Arquitectura

- NameNode
  - Es la pieza central del HDFS
  - Administra el almacenamiento de datos
  - No almacena datos en si mismo
  - Las operaciones de Entrada/Salida no pasana través de él
  - Hacede intermediario entre el cliente y los DataNodes
  - Es un Single Point of Failure

HDFS - Arquitectura

- DataNode
  - Cientos o miles de DataNodes por cluster
  - Organizados en racks
  - Operaciones de Entrada/Salida ocurren sobre el DataNode
  - Contienen información replicada
  - Alta tolerancia a fallas

HDFS - Arquitectura



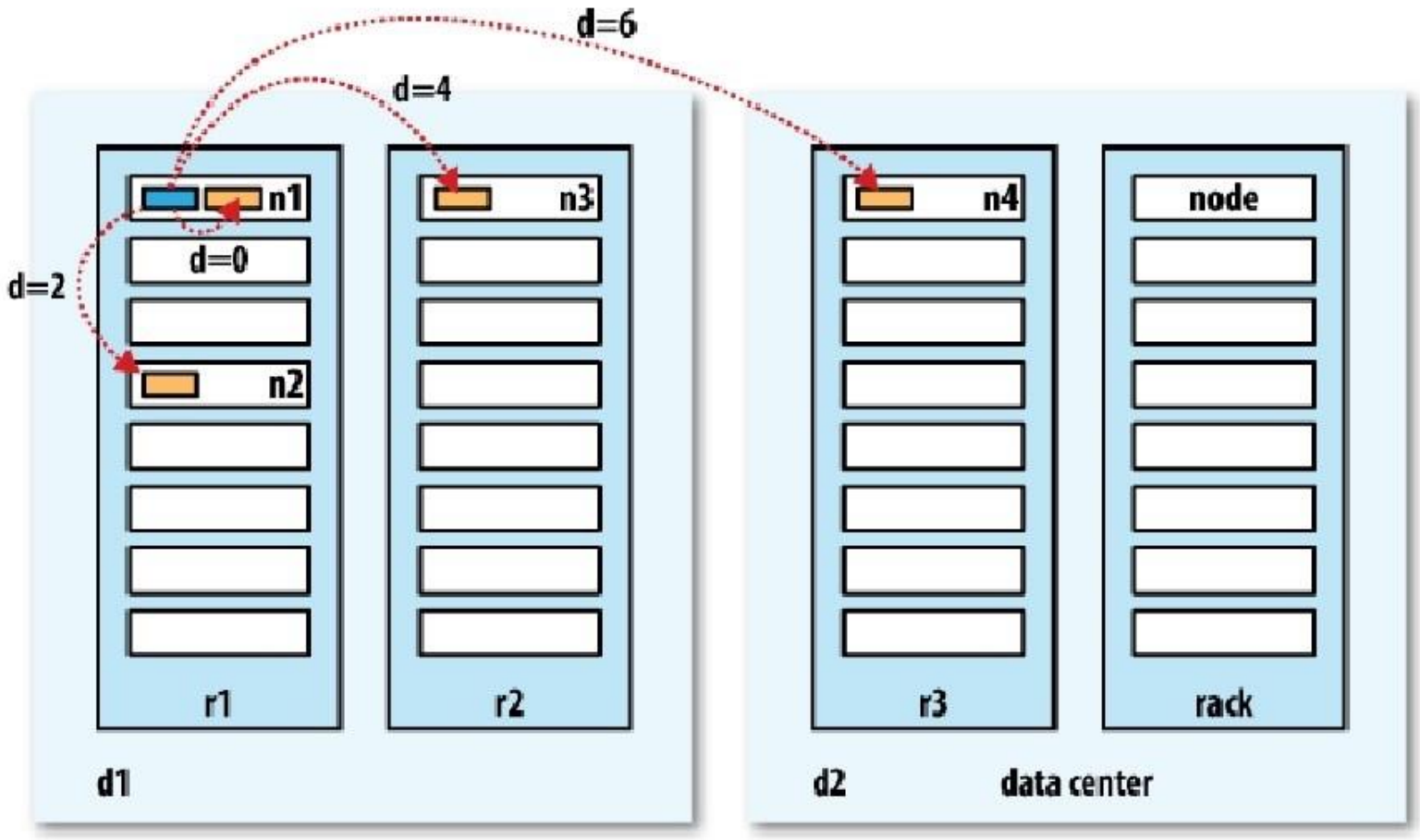
¿Cuándo usar HDFS?

- Archivos muy, muy grandes (GB o más)
- Necesidad de particionar archivos
- Fallo de nodos sin perder información
- Una escritura, muchas lecturas

¿Cuándo NO usar HDFS?

- Alta latenciade red
- Muchos archivos pequeños
- Multiples "escritores"
- Modificaciones arbitrarias en los archivos

Optimización por distancia física



HDFS - API

- Permite interactuar con el HDFS a través de CLI

Ej: `$ hadoop fs –copyFromLocal miArchivo /miHDFSDir`

Algunos comandos son:

- cat
- copyFromLocal
- copyToLocal
- du
- dus
- cp
- rmr
- mkdir

Componentes de Hadoop

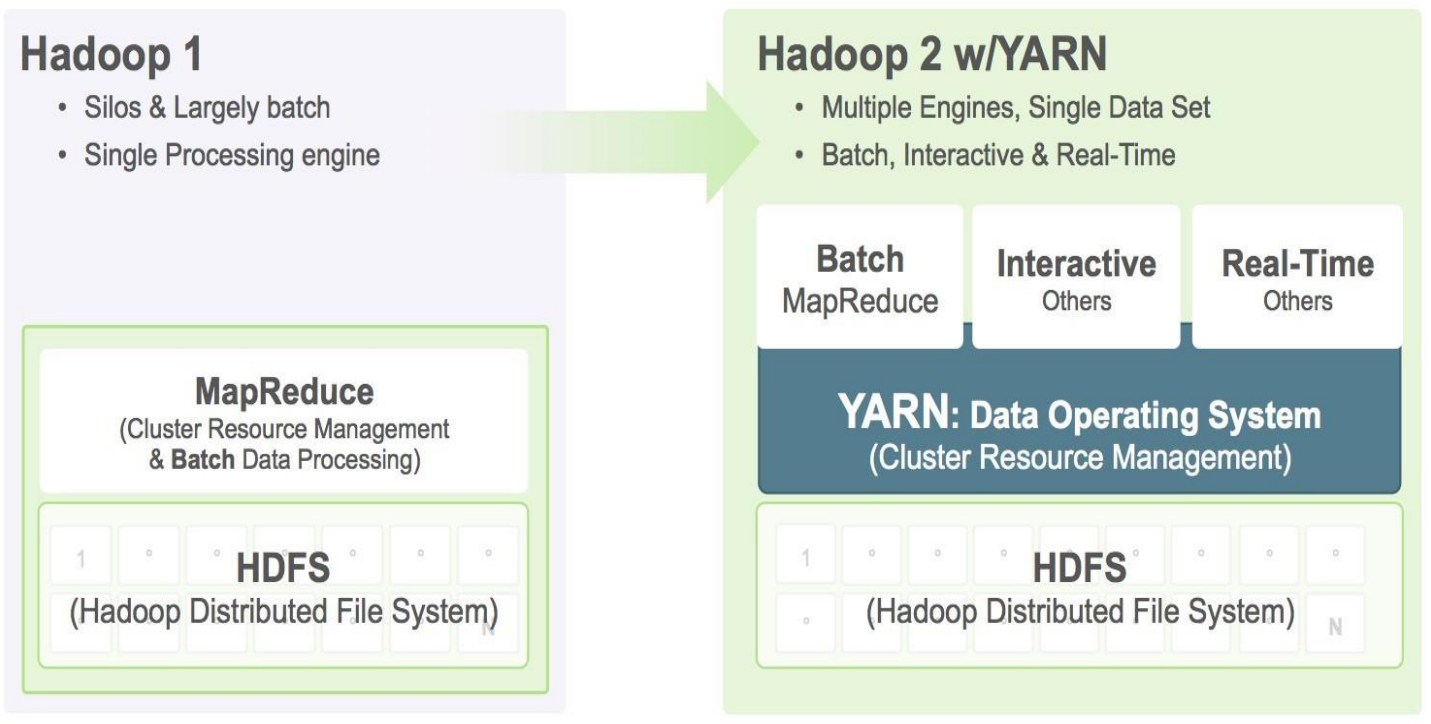
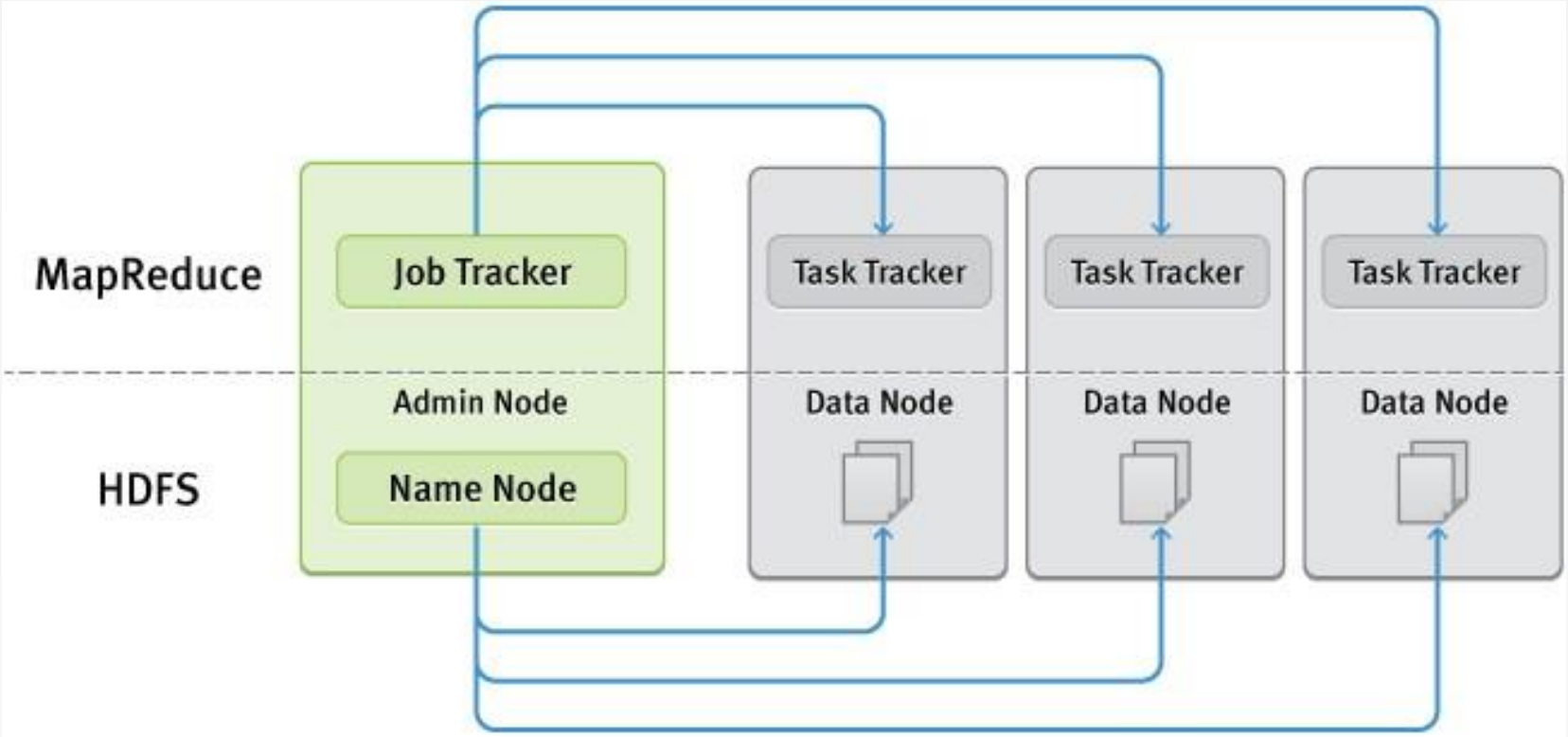


Imagen tomada de: http://hortonworks.com/hadoop/yarn/

Hadoop 1.0


MapReduce
HDFS
Job Tracker
Admin Node
Name Node
Task Tracker
Data Node
Task Tracker
Data Node
Task Tracker
Data Node

Hadoop 2.0

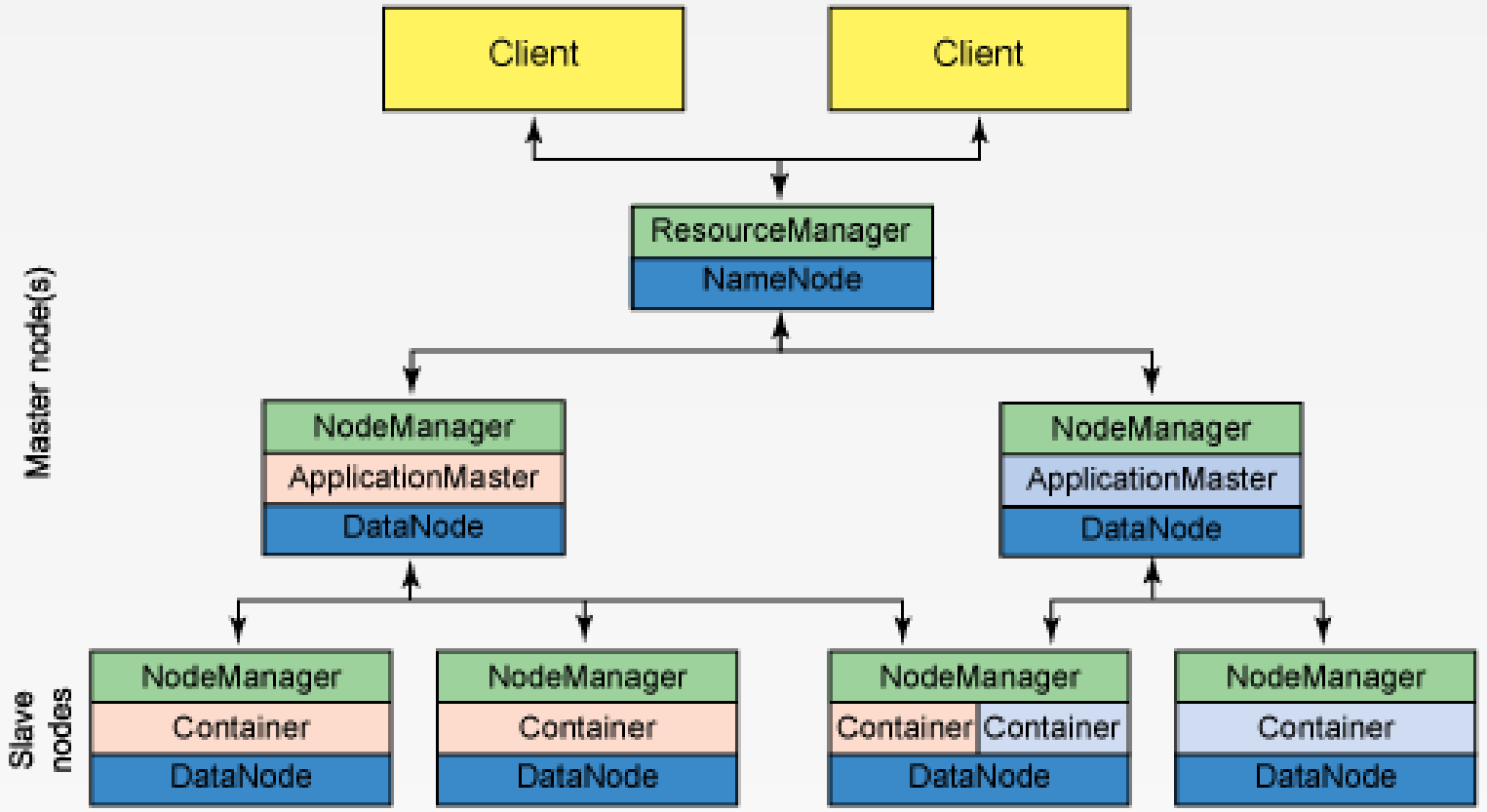


Imagen tomada de: http://www.ibm.com/developerworks/library/bd-hadoopyarn/

Word Count en Hadoop

Esto es una linea
Esto también

Map

map(“Esto es una linea”) =
esto, 1 es, 1 una, 1
linea, 1
map(“Esto también”) =
esto, 1 también, 1

Reduce

reduce(es, {1}) = es, 1
reduce(esto, {1, 1}) = esto, 2
reduce(linea, {1}) = linea, 1
reduce(también, {1}) = también, 1
reduce(una, {1}) = una, 1

Resultado:

es, 1
esto, 2 linea, 1 también,
1 una, 1

Word Count en Hadoop

```
public    class WordCountHadoop          extends  Configured  implements  Tool  {  class  TokenizerMapper  extends            InterruptedException {

                        public static Mapper<Object, Text, Text, IntWritable> {

                                     private final static IntWritable one = new IntWritable(1); private Text
                                     word = new Text();

                                    public                   void map(Object key, Text value, Context context) throws IOException,
                                                     StringTokenizer   itr   =   new   StringTokenizer(value.toString());
                                                     while(itr.hasMoreTokens())        {        word.set(itr.nextToken());
                                                     context.write(word, one);

                                                     }
                                     }
                                                                                                                         IOException,
                       }


                                         class IntSumReducer extends Reducer<Text, IntWritable, Text, IntWritable> {
                          public sta private IntWritable result = new           IntWritable();

                                    public           void reduce(Text key, Iterable<IntWritable> values, Context context) throws
                                         InterruptedException {
                                                     int    sum = 0;
                                                     for(IntWritable val : values) {
                                                                 sum    += val.get();


                                    ul set(sum);
                                    ntext.write(key, result);
ride
          public int run(String[]

                                    args) throws Exception {

                                     if(args.length != 2) {
                                                     System.err.println("Usage: wordcount-hadoop <in> <out>");
                                                     System.exit(2);
                                     }

                                     Path output = new Path(args[1]);
                                     HadoopUtils.deleteIfExists(FileSystem.get(output.toUri(), conf), output);

                                     Job job = new Job(getConf(), "word count hadoop");
                                     job.setJarByClass(WordCountHadoop.class);
                                     job.setMapperClass(TokenizerMapper.class);
                                     job.setCombinerClass(IntSumReducer.class);
                                     job.setReducerClass(IntSumReducer.class); job.setOutputKeyClass(Text.class);

                                     job.setOutputValueClass(IntWritable.class);
                                     FileInputFormat.addInputPath(job, new Path(args[0]));

                                     FileOutputFormat.setOutputPath(job,   new   Path(args[1]));
                                     job.waitForCompletion(true); return 0;
         }

                       public static void main(String[] args) throws Exception {
                                       ToolRunner.run(new SortJobHadoop(), args);
                       }
      }
```

¡Mejor Vamos por partes!

Word Count en Hadoop - Mapper

```
public static class Map extends Mapper<LongWritable, Text, Text, IntWritable> {
    private final static IntWritable one = new IntWritable(1); private Text word
    = new Text();

    public void map(LongWritable key, Text value, Context context) throws
IOException, InterruptedException {
        String line = value.toString();
        StringTokenizer tokenizer = new StringTokenizer(line);
        while          (tokenizer.hasMoreTokens())          {
        word.set(tokenizer.nextToken());  context.write(word,
        one);
        }
    }
 }
```

Word Count en Hadoop - Reducer

```
public static class Reduce extends Reducer<Text,
IntWritable, Text, IntWritable> {

    public void reduce(Text key, Iterable<IntWritable>
values, Context context) throws IOException,
      InterruptedException{
        int sum = 0;
        for (IntWritable val : values) {
            sum += val.get();
        }
        context.write(key, new IntWritable(sum));
    }
 }
```

Word Count en Hadoop – configuración (main)

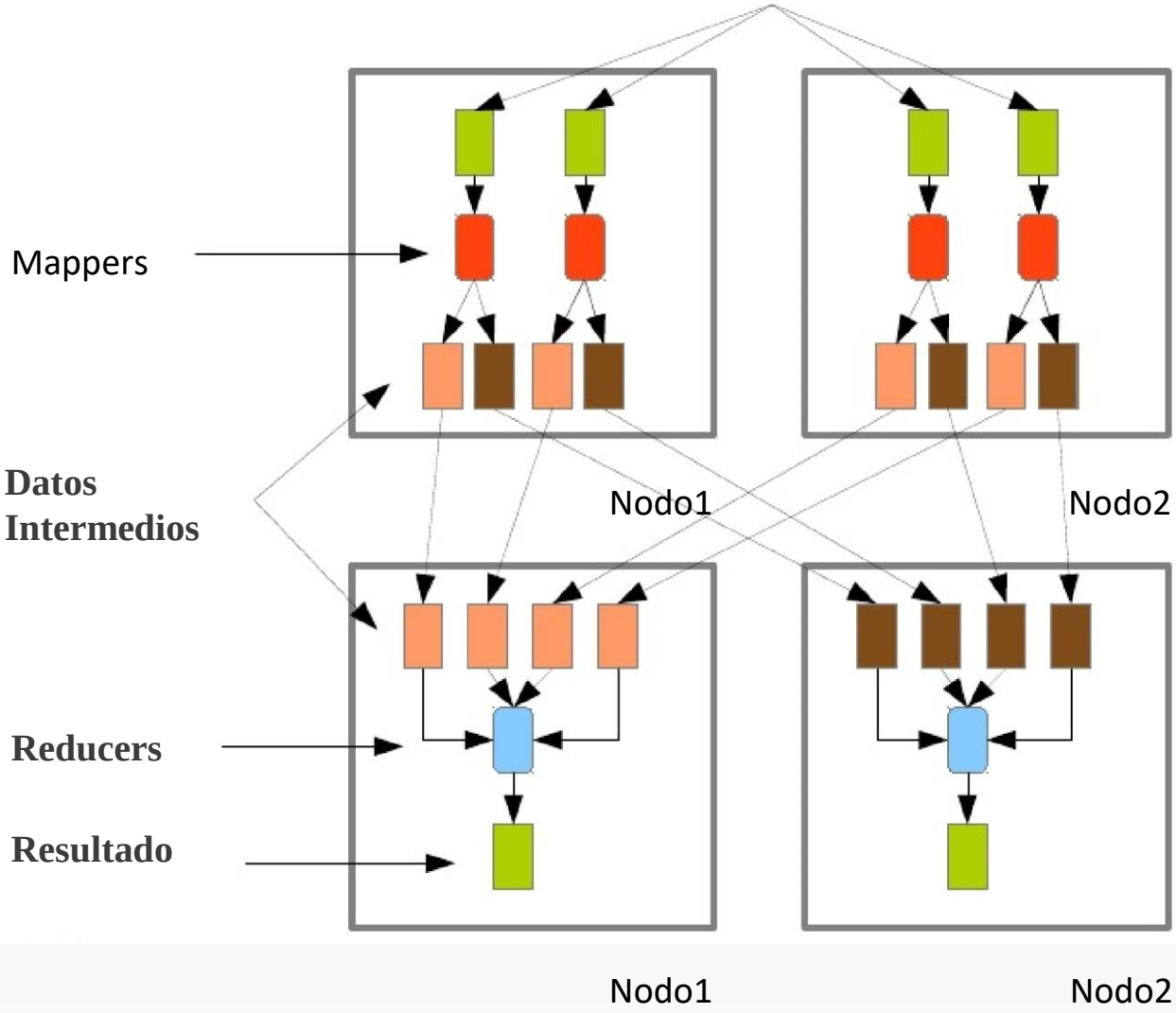
```
public static void main(String[] args) throws Exception {
    Configuration conf = new Configuration();

        Job job = new Job(conf, "wordcount");

    job.setOutputKeyClass(Text.class);
    job.setOutputValueClass(IntWritable.class);

    job.setMapperClass(Map.class);
    job.setReducerClass(Reduce.class);

    job.setInputFormatClass(TextInputFormat.class);
    job.setOutputFormatClass(TextOutputFormat.class);

    FileInputFormat.addInputPath(job,          new          Path(args[0]));
    FileOutputFormat.setOutputPath(job, new Path(args[1]));

    job.waitForCompletion(true);
 }
```

https://hadoop.apache.org/docs/current/hadoop-mapreduce-client/hadoop-mapreduce-client-core/MapReduceTutorial.html
https://hadoop.apache.org/docs/r2.7.0/api/org/apache/hadoop/mapreduce/package-tree.html

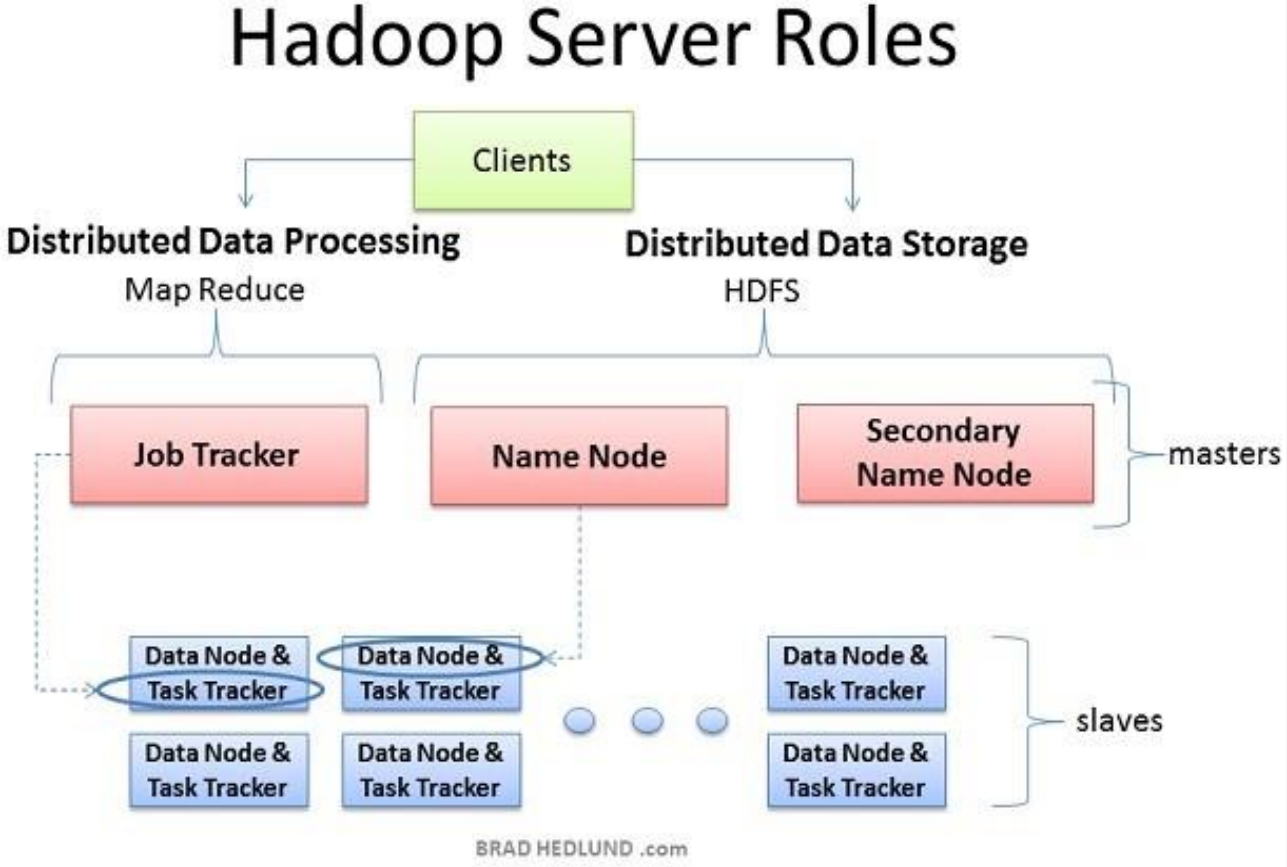
Modos de ejecución

- Hadoop se puede ejecutar de tres formas distintas:
  - **Modo Local / *Standalone***: Por defecto, Hadoop está configurado para ejecutarse en este modo como un proceso de Java aislado. Esto es útil para depuración.
  - **Modo Pseudo-distribuido**: Hadoop puede ejecutarse en este modo, en donde cada tarea se ejecuta en proceso Java diferente.
  - **Modo Distribuido**: Esta es la forma de aprovechar toda la potencia de Hadoop, ya que se maximiza el paralelismo de procesos y se utilizan todos los recursos disponibles del *clúster* en el que se va a configurarHadoop.

Word Count en Hadoop - Ejecución

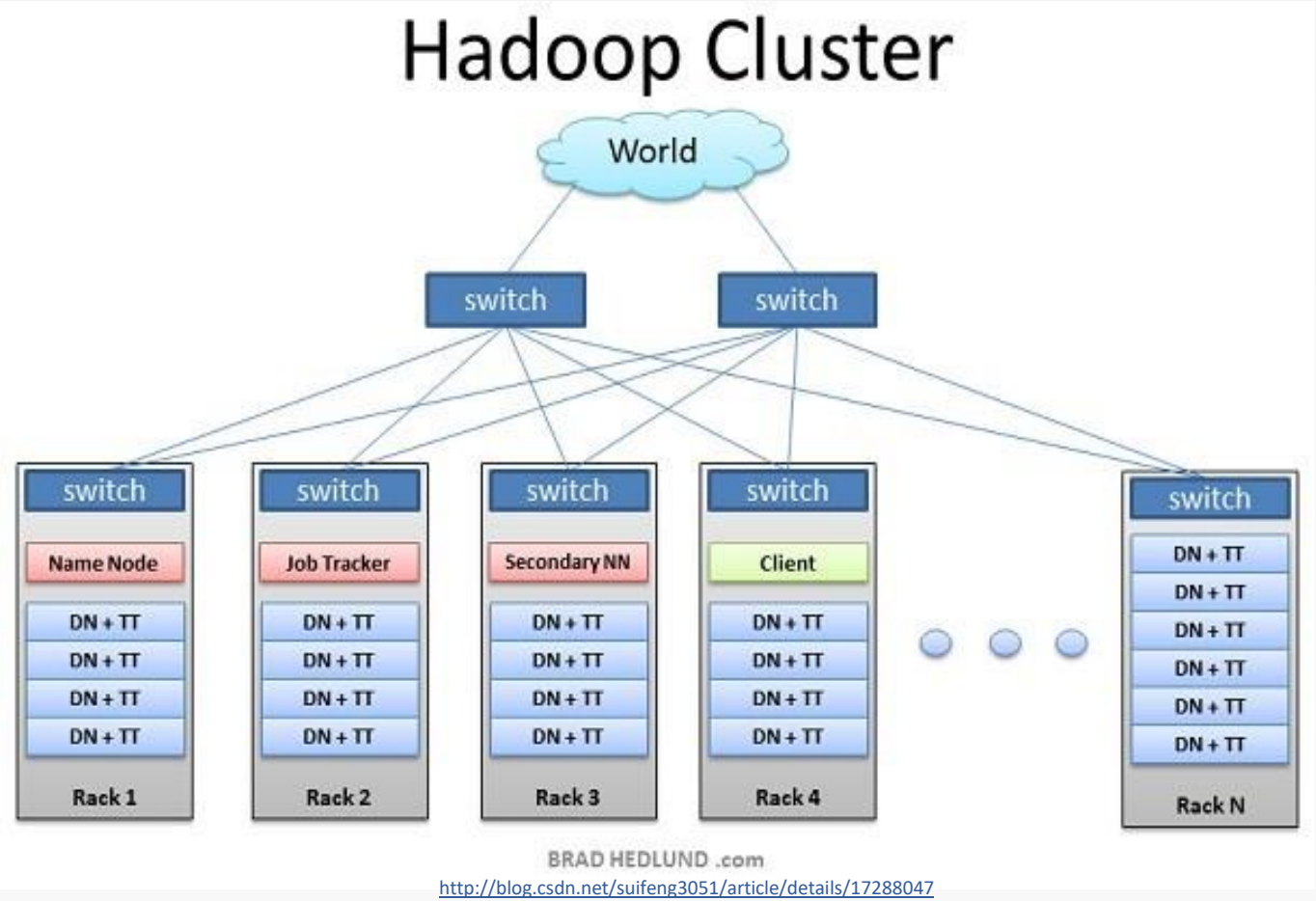


Despliegue



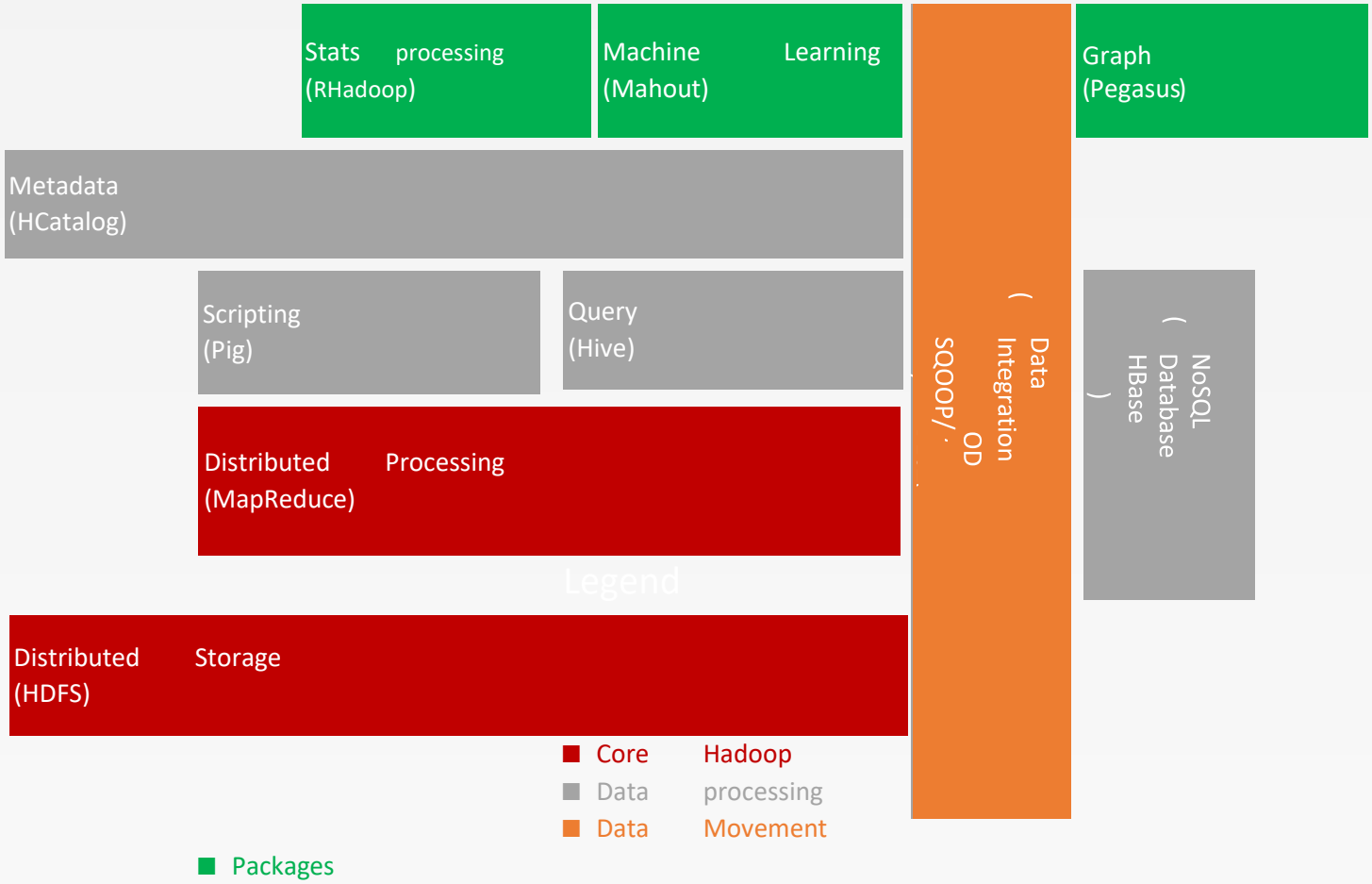
http://blog.csdn.net/suifeng3051/article/details/17288047

Despliegue



http://blog.csdn.net/suifeng3051/article/details/17288047

Hadoop – Componentes principales



Hive

- Consultas en paralelo usando MapReduce
- Lenguaje HiveQL (Símil Sql)
- Permite procesar grandes volúmenes de datos
- Escalabilidad
- Tolerancia a fallos

Ejemplos HiveQL

- Crear una Tabla Externa

```
CREATE EXTERNAL TABLE iislogs(
        sdate string, stime string, ssitename string, csmethod string, csuristem string,
        csuriquery string, sport int, scstatus int, scbytes int, sbytes int, timetaken int)
ROW FORMAT DELIMITED
  FIELDS TERMINATED BY ' '
LOCATION
  'wasb://iis-logs@myapp.blob.core.windows.net/'
```

- Ejecutar una query para obtener datos

```
select sdate, stime, csmethod, scuristem, query_params['api-version'] as api_version,
        query_params['search_value'] as search_value
from (select *, str_to_map(csuriquery, '&', '=') as query_params
from iislogs ) version_logs
where query_params['api-version'] is not null
```

Pig

- Lenguaje script para expresar sentencias MapReduce
- Usa paralelismo para ejecutar las sentencias
- Optimizado para grandes volúmenes de datos
- Lenguaje PigLatin (Símil Sql)

Ejemplos PigLatin

- Carga y Transformación de Datos

```
A = load 'passwd' using PigStorage(':');  -- load the passwd file
B = foreach A generate $0 as id;  -- extract the user IDs store B
into ‘id.out’;  -- write the results to a file name id.out
```

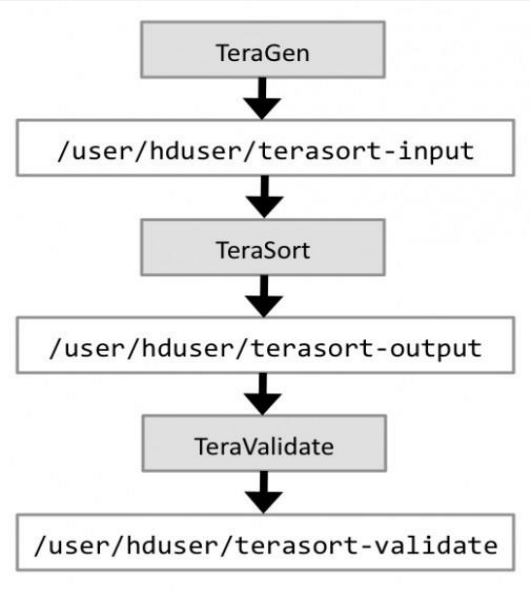
- Ejecutar una query para obtener y procesar datos

```
LOGS = LOAD 'wasb:///example/data/sample.log';
LEVELS = foreach LOGS generate REGEX_EXTRACT($0, '(TRACE|DEBUG|INFO|WARN|ERROR|FATAL)', 1) as
LOGLEVEL;
FILTEREDLEVELS = FILTER LEVELS by LOGLEVEL is not null;
GROUPEDLEVELS = GROUP FILTEREDLEVELS by LOGLEVEL;
FREQUENCIES = foreach GROUPEDLEVELS generate group as LOGLEVEL, COUNT(FILTEREDLEVELS.LOGLEVEL) as
COUNT;
RESULT = order FREQUENCIES by COUNT desc;
DUMP RESULT;
```

Benchmarking

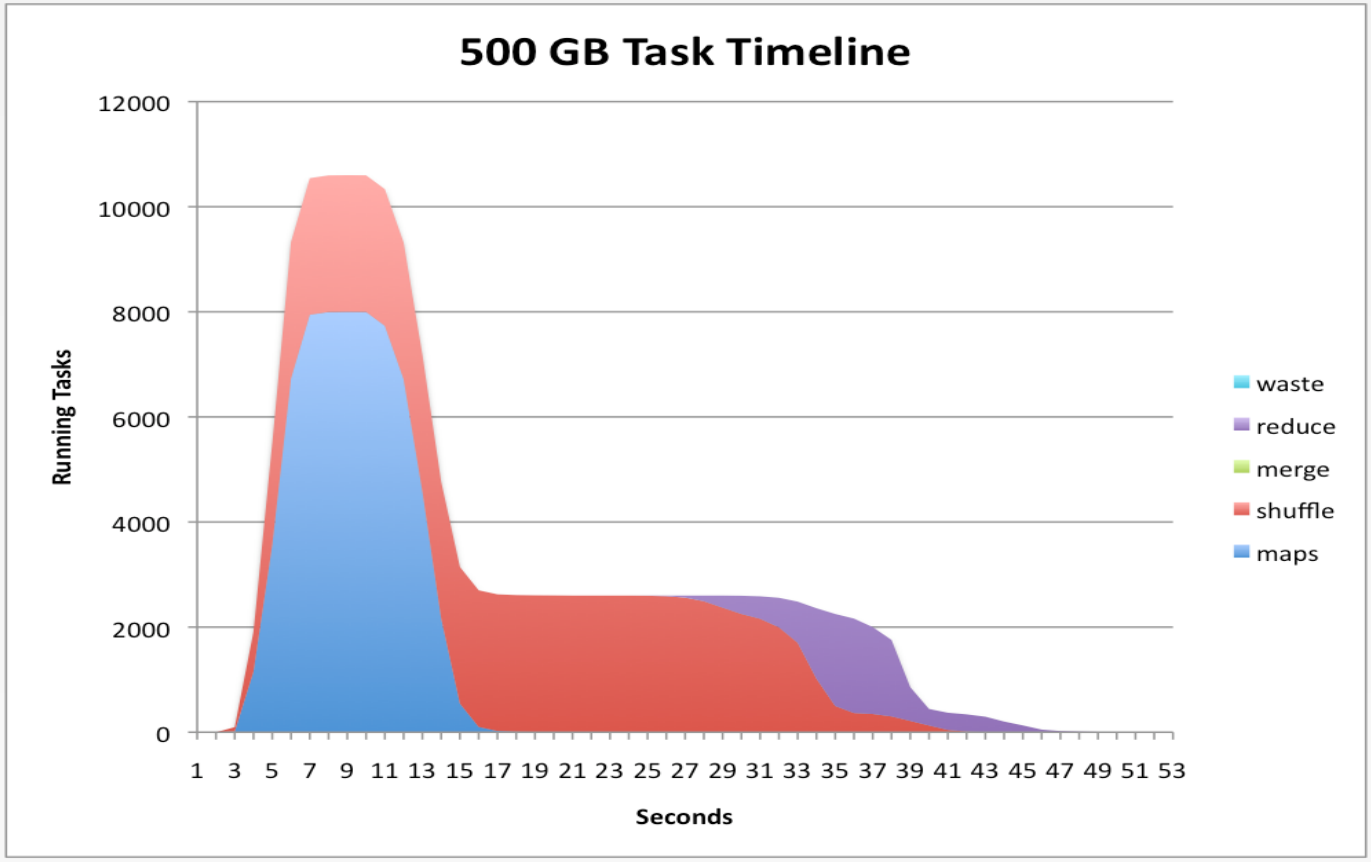
- TestDFSIO

- TeraSort benchmark suite
  - Yahoo! 2009: 1 PB de datos en 16 horas
- NameNode benchmark (nnbench) • MapReduce benchmark (mrbench)



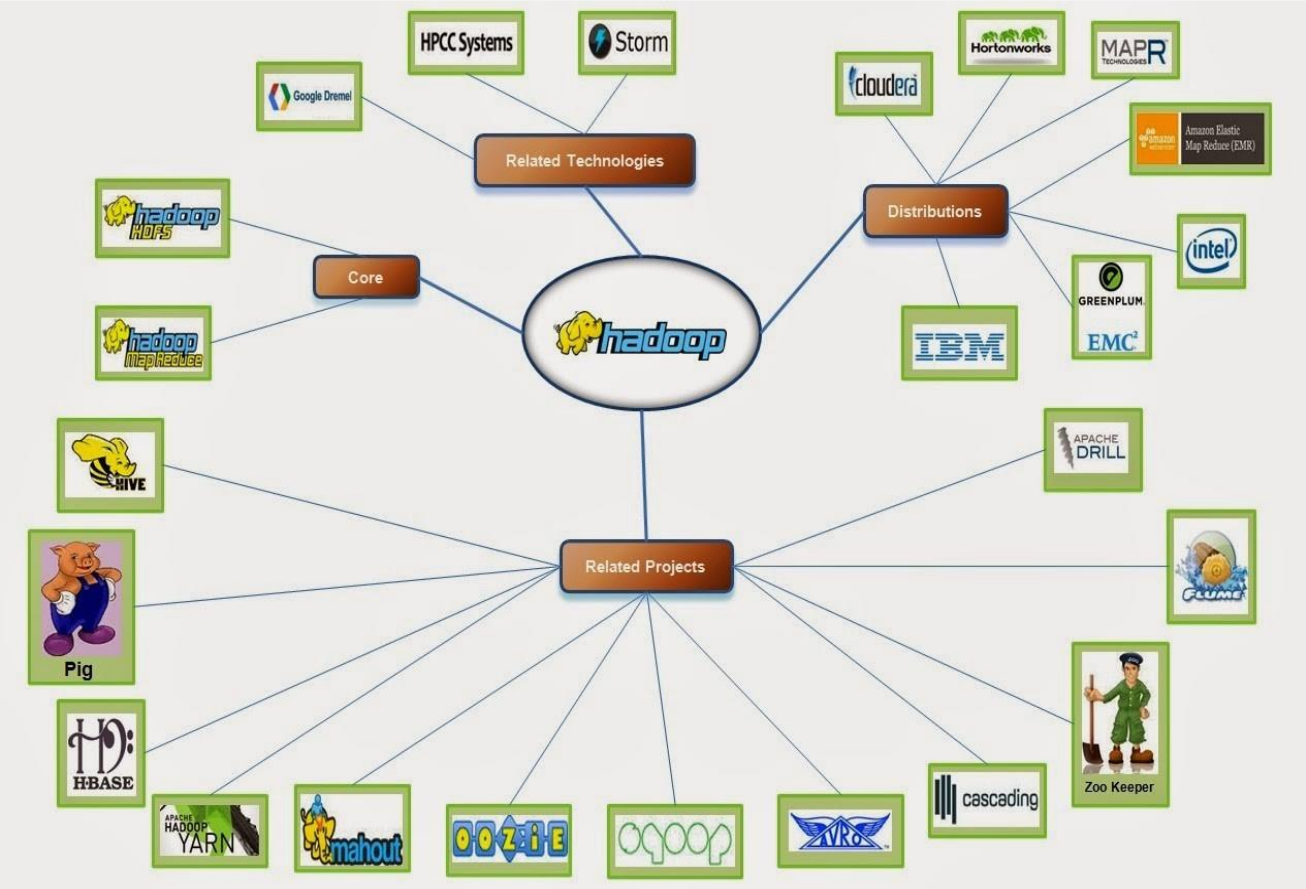
http://m.oschina.net/blog/74201

TeraSort (2009,500GB)



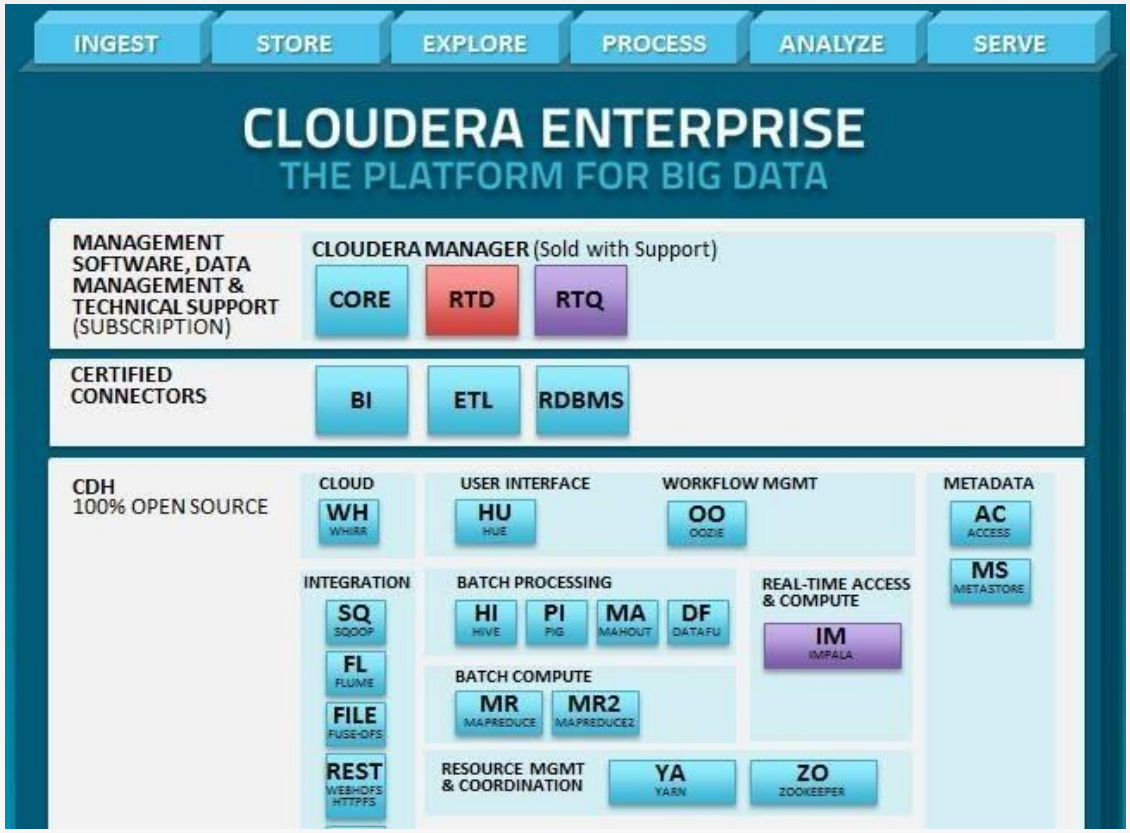
https://developer.yahoo.com/blogs/hadoop/hadoop--sorts--petabyte--16--25--hours--terabyte--62--422.html

Ecosistema



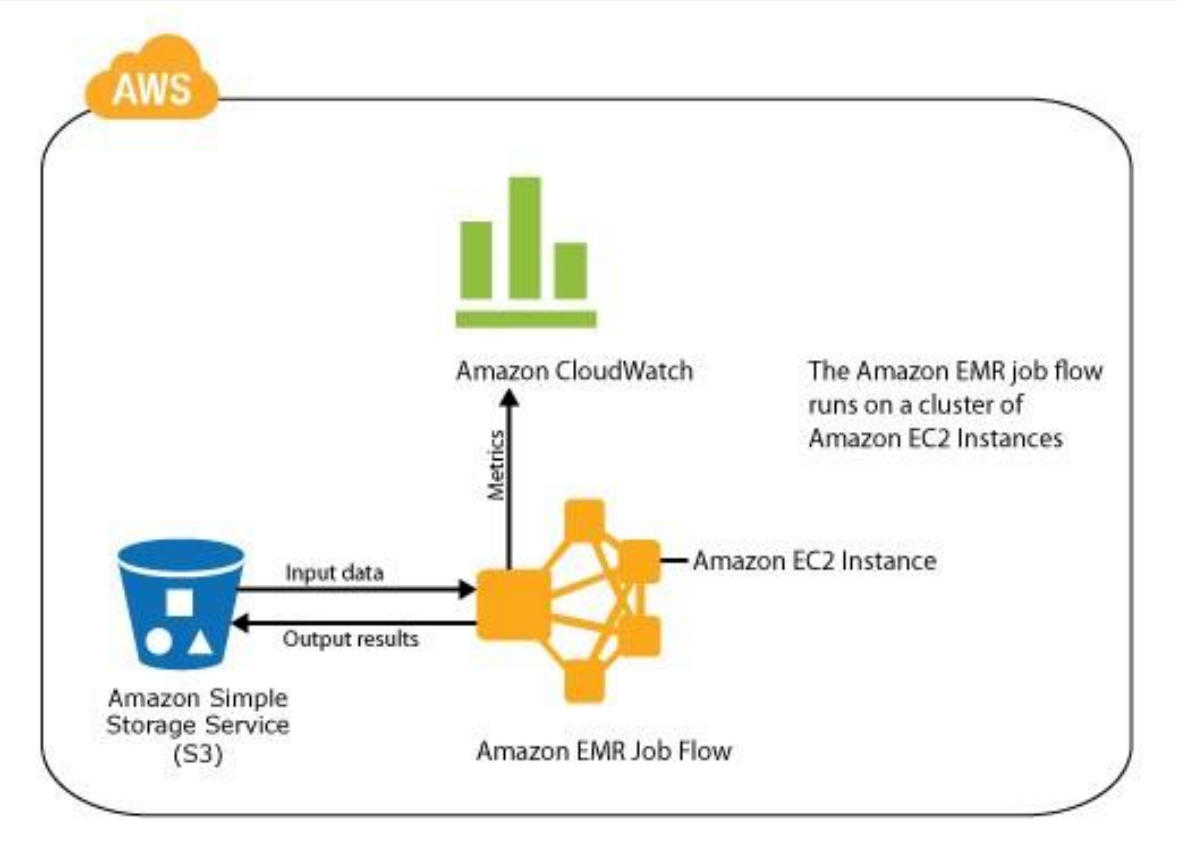
http://ambuj4bigdata.blogspot.com.es/2014_05_01_archive.html

CDH (Cloudera Hadoop Distribution)



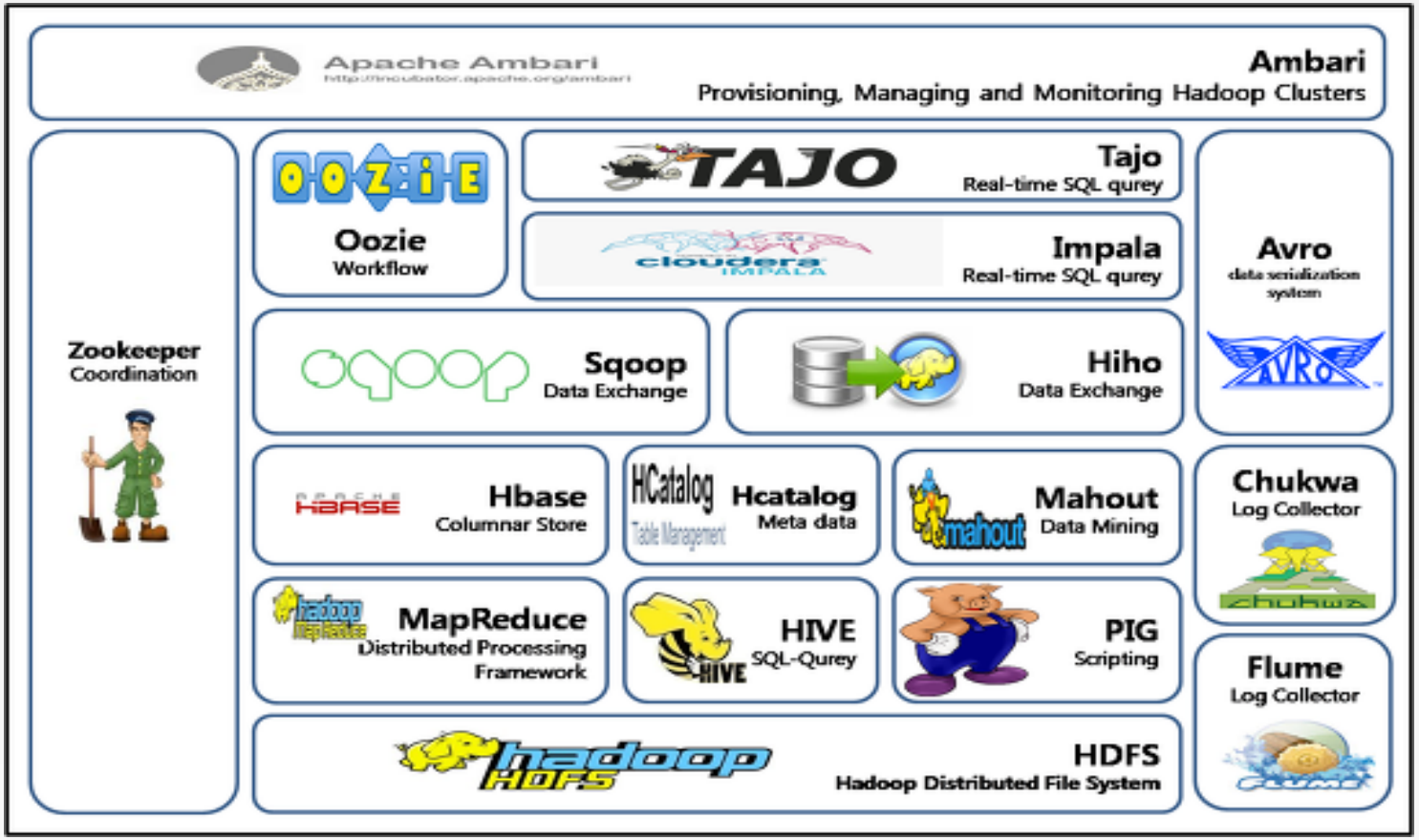
http://www.theregister.co.uk/2012/06/05/cloudera_cdh4_hadoop_stack/

Amazon EMR (Amazon Elastic MapReduce)



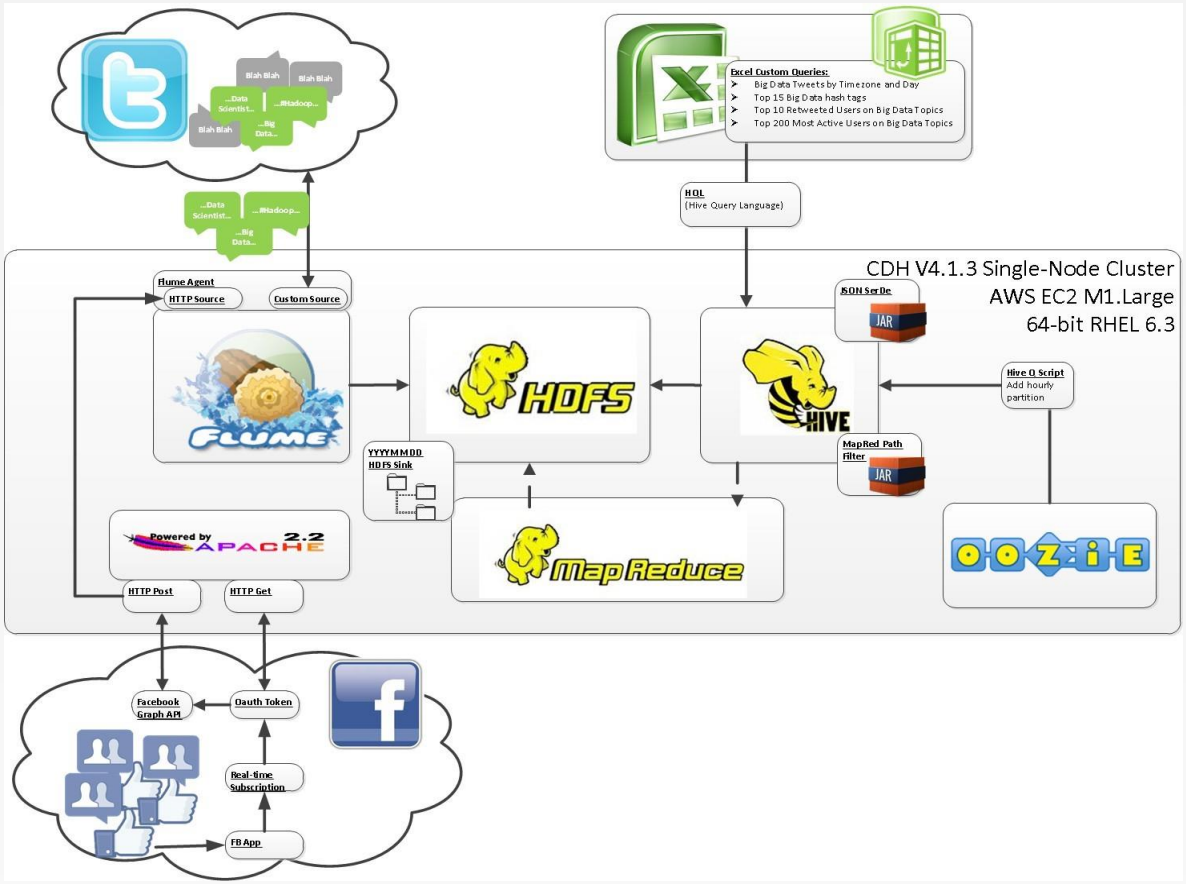
http://aws.amazon.com/es/elasticmapreduce/

Ecosistema



http://www.nextree.co.kr/p2865/

Ej.: Flume--Hive--oozie



http://www.datadansandler.com/2013/03/big--data--and--hadoop--assets--available--on.html

Ejemplo WordCount

- Creamos el fichero WordCount.java con el contenido de
  - https://hadoop.apache.org/docs/current/hadoopmapreduce-client/hadoop-mapreduce-clientcore/MapReduceTutorial.html

- Definimos los siguientes variables de entorno

```
$ export JAVA_HOME=/usr/lib/jvm/java-8-oracle
$ export PATH=${JAVA_HOME}/bin:${PATH}
$ export HADOOP_CLASSPATH=${JAVA_HOME}/lib/tools.jar
```

6

Ejemplo WordCount (Manual)

- Compilamos WordCount y creamos un fichero jar

```
$ hadoop com.sun.tools.javac.Main WordCount.java
$ jar cf wc.jar WordCount*.class
```

- Ejecutamos:

```
$ hadoop jar wc.jar WordCount hdfs:///books
hdfs:///user/miusuario
```

Ejemplo WordCount (Maven) (I)

- Escogemos una carpeta donde crear el proyecto y ejecutamos:

```
$ mvn archetype:generate -DgroupId=com.wordcount -
DartifactId=wordcount -DarchetypeArtifactId=maven-
archetypequickstart -DinteractiveMode=false
```

- Entramos en la carpeta del proyecto que se ha creado, en este caso:

```
$ cd wordcount
```

- Borramos la carpeta para pruebas:

```
$ rm -rf src/test
```

Ejemplo WordCount (Maven) (II)

- Añadimos al fichero de configuración pom.xml el siguiente contenido:

```
<dependencies>
    <dependency>
        <groupId>org.apache.hadoop</groupId>
        <artifactId>hadoop-common</artifactId>
        <version>2.2.0</version>
    </dependency>

    <dependency>
        <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-core</artifactId>
        <version>1.2.1</version>
    </dependency>
  </dependencies>
  <repositories>
    <repository>
        <id>cloudera</id>
        <url>https://repository.cloudera.com/artifactory/cloudera-repos/</url>
    </repository>
</repositories>
```

Ejemplo WordCount (Maven) (III)

- Copiamos nuestros ficheros .java dentro de la carpeta: src/main/java/com/wordcount

```
$ cp ../wordcount.java src/main/java/com/wordcount
```

- Compilamos y empaquetamos nuestro fichero jar

```
$ mvn install
```

- Ejecutamos:

```
$ hadoop jar target/wordcount-1.0-SNAPSHOT.jar WordCount
hdfs:///books hdfs:///user/miusuario
```

Bibliografía: tutoriales

- Página Web oficial:
  - http://hadoop.apache.org/
- Introducción a cómo funcionaHadoop:
  - http://blog.csdn.net/suifeng3051/article/details/17288047
- Tutorial de cómo instalar y usar Hadoop:
  - http://www.bogotobogo.com/Hadoop/ BigData_hadoop_Install_on_ubuntu_single_node_cluster.php
  - http://www.bogotobogo.com/Hadoop/ BigData_hadoop_Running_MapReduce_Job.php

Bibliografía: libro

- Hadoop: The Definitive Guide, 3rd Edition:
  - http://shop.oreilly.com/product/0636920021773.do
  - https://github.com/tomwhite/hadoop-book/