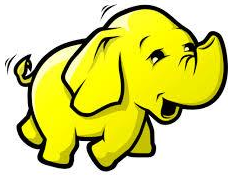
Universitatea “Alexandru Ioan Cuza”
Facultatea de Informatică

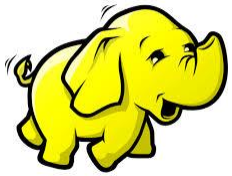


Hadoop

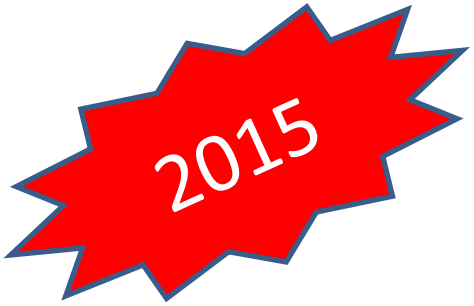
Lenuța Alboaie
adria@info.uaic.ro

Cuprins

- Context
- Hadoop – imagine generala
  - Componente
- **HDFS - Hadoop Distributed Filesystem**
  - Caracteristici
  - Concepte
  - Arhitectura
  - Map Reduce … YARN



Context



Context:

- **!DATA**
- Estimari: 0.18 zetabytes in 2006 -> 1.8 zettabytes in 2011-> … 2015

(1 zetabytes = 10^{21} bytes)

- Surse:

- The New York Stock Exchange generates about one terabyte of new trade data per day.
- Facebook hosts approximately 10 billion photos, taking up one petabyte of storage.
- Ancestry.com, the genealogy site, stores around 2.5 petabytes of data.
- The Internet Archive stores around 2 petabytes of data, and is growing at a rate of 20 terabytes per month.
- The Large Hadron Collider near Geneva, Switzerland, will produce about 15 petabytes of data per year.

[Tom White, Hadoop-The definitive Guide, 2011]

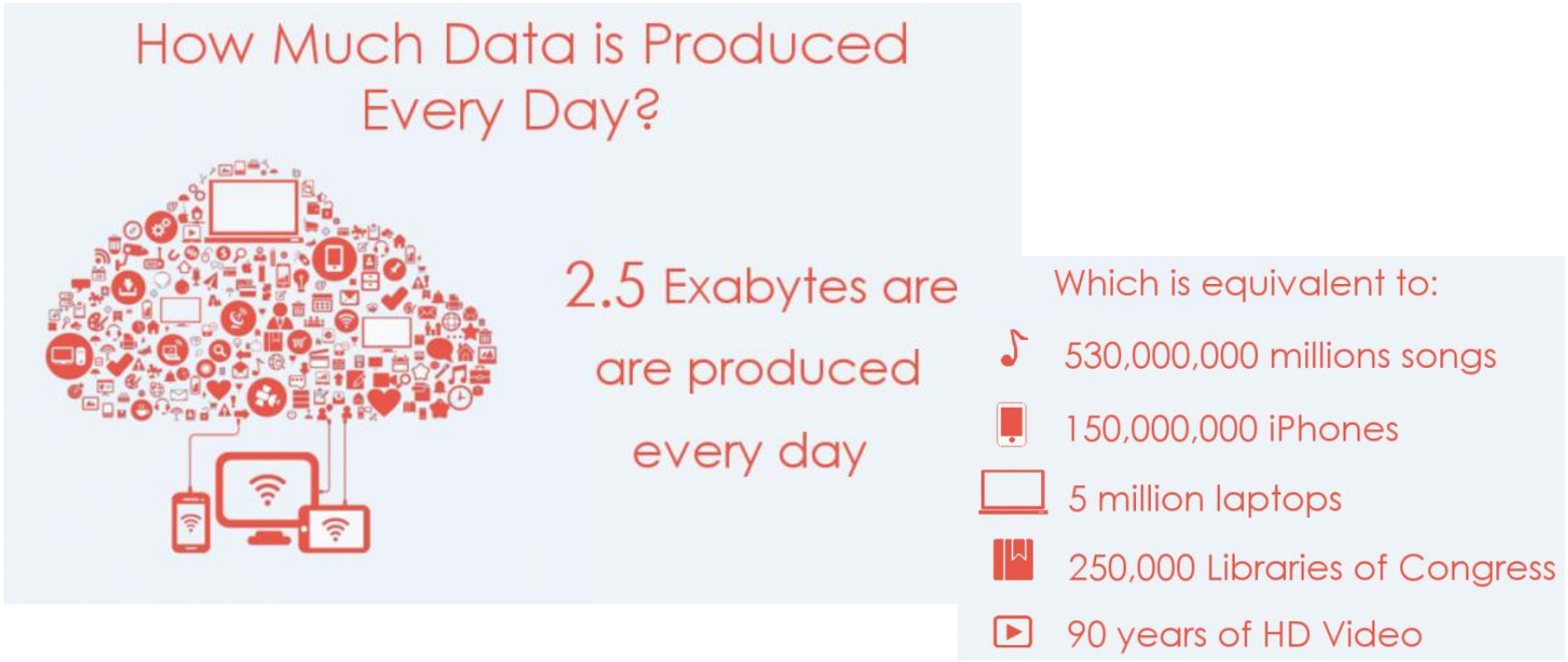
- ? Succesul => capacitatea de analiza a datelor diferitelor organizatii (e.g. initiative Public Data Sets – *Amazon, Infochimps.org, theinfo.org, Google,* …)

Context

2017

Context:

- **!DATA**

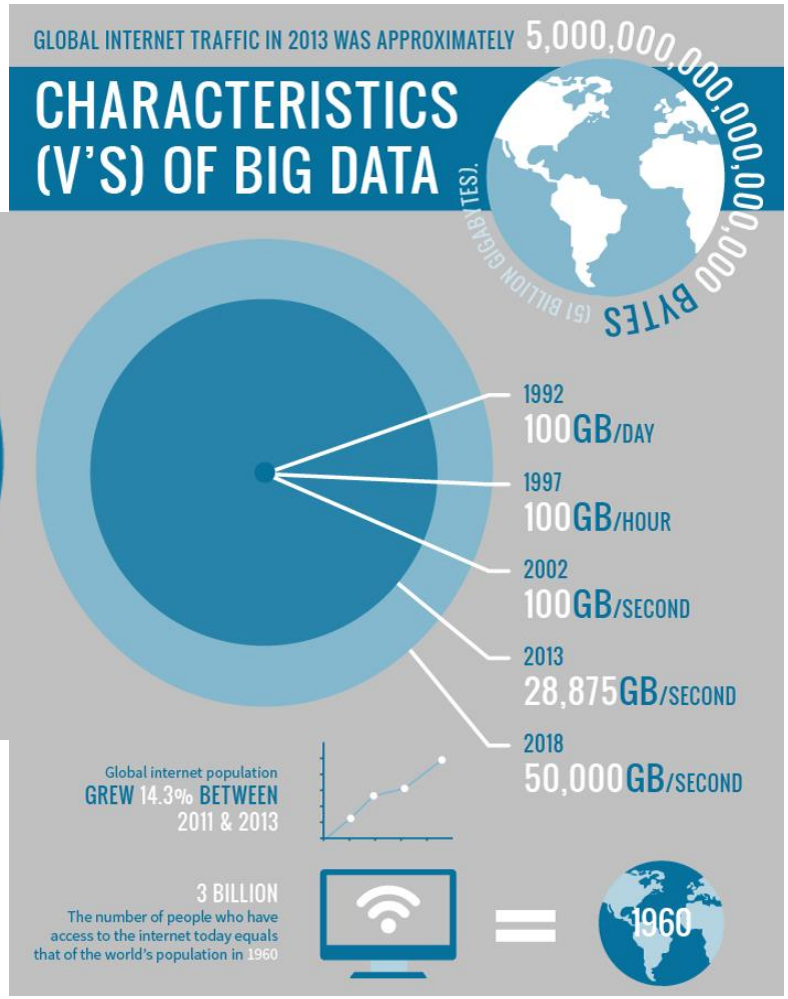


- ? Succesul => capacitatea de analiza a datelor diferitelor organizatii (e.g. initiative Public Data Sets – *Amazon, Infochimps.org, theinfo.org, Google, …*)

Context

Context:

- **!DATA**



Context

AWS Public Datasets

Sign up now

AWS hosts a variety of public datasets that anyone can access for free.

Previously, large datasets such as satellite imagery or genomic data have required hours or days to locate, download, customize, and analyze. When data is made publicly available on AWS, anyone can analyze any volume of data without needing to download or store it themselves. These datasets can be analyzed using AWS compute and data analytics products, including Amazon EC2, Amazon Athena, AWS Lambda and Amazon EMR.

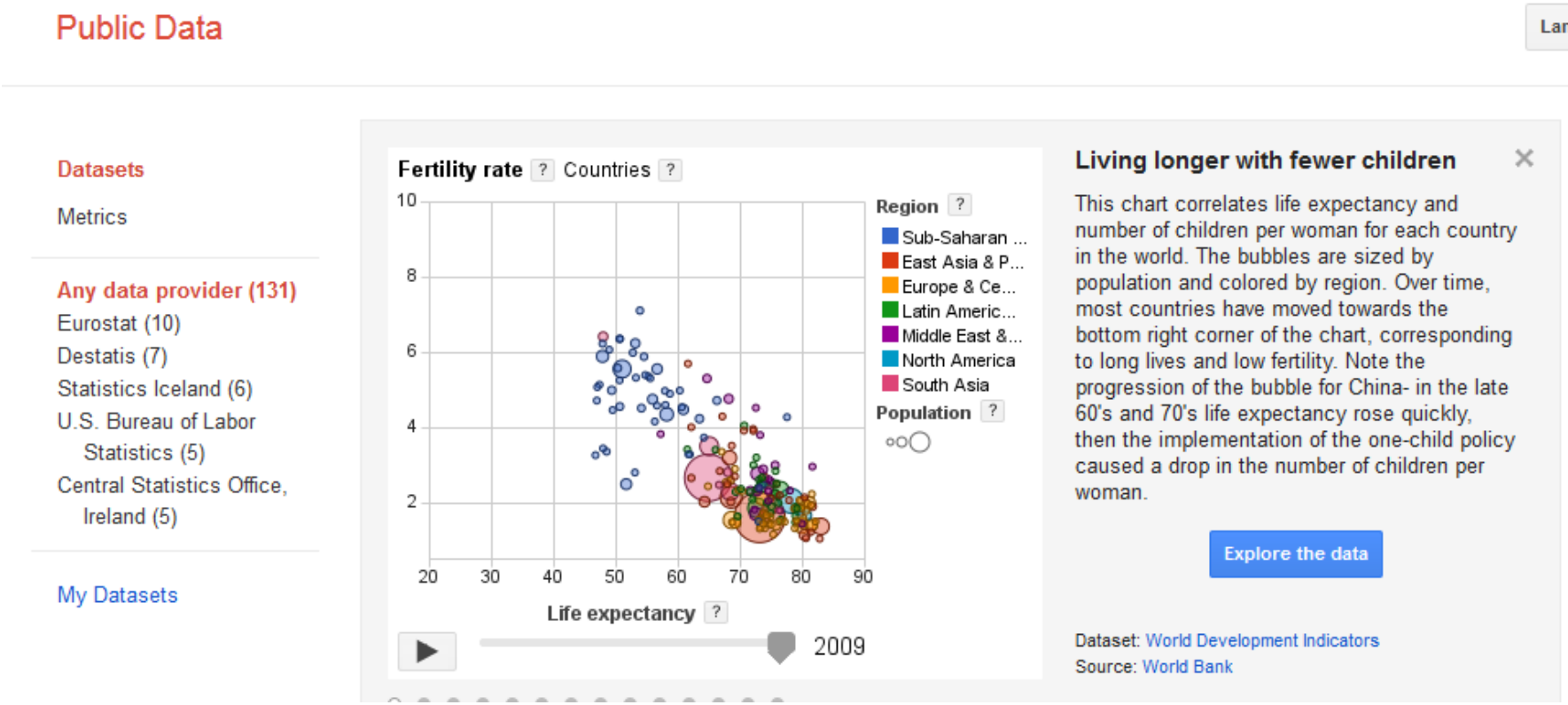
Available Public Datasets on AWS

Geospatial and Environmental Datasets

Learn more about working with geospatial data on AWS at Earth on AWS

[https://aws.amazon.com/public-datasets/]

Context


Public Data
Languag
Datasets
Metrics
Any data provider (131)
Eurostat (10)
Destatis (7)
Statistics Iceland (6)
U.S. Bureau of Labor Statistics (5)
Central Statistics Office, Ireland (5)
My Datasets
Fertility rate
Countries
10
8
6
4
2
20
30
40
50
60
70
80
90
Life expectancy
2009
Region
Sub-Saharan ...
East Asia & P...
Europe & Ce...
Latin Americ...
Middle East &...
North America
South Asia
Population
Living longer with fewer children
This chart correlates life expectancy and number of children per woman for each country in the world. The bubbles are sized by population and colored by region. Over time, most countries have moved towards the bottom right corner of the chart, corresponding to long lives and low fertility. Note the progression of the bubble for China- in the late 60's and 70's life expectancy rose quickly, then the implementation of the one-child policy caused a drop in the number of children per woman.
Explore the data
Dataset: World Development Indicators
Source: World Bank

Context

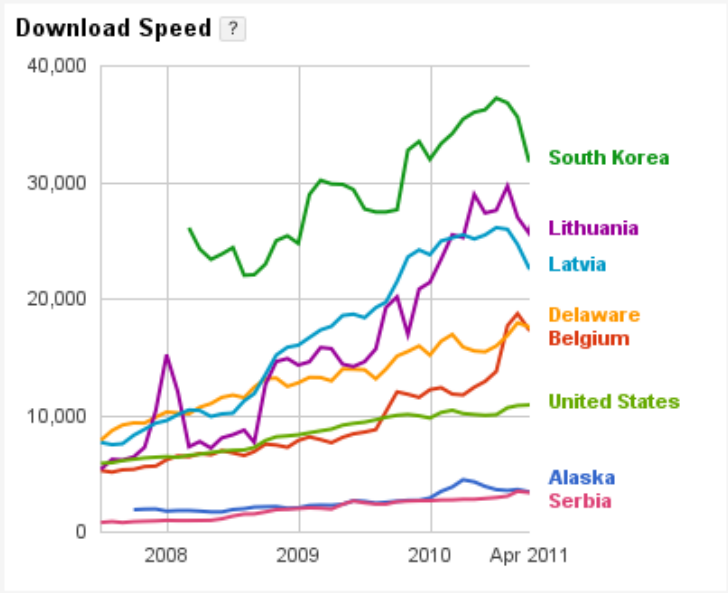
Public Data

Datasets

Metrics

Any data provider (131)
Eurostat (10)
Destatis (7)
Statistics Iceland (6)
U.S. Bureau of Labor Statistics (5)
Central Statistics Office, Ireland (5)

My Datasets



Who enjoys the fastest internet?

South Koreans do, according to Ookla- the average South Korean Internet connection is more than 3x faster than the average connection in the US. Eastern European countries like Latvia and Lithuania are also at the top of the pool. Within the US, there is tremendous variation by state. Delaware, a very small state, has the fastest connections (comparable to those in Belgium), whereas Alaska, the biggest state, has the slowest connections (comparable to those in Serbia).

Explore the data

Dataset: Global Broadband Performance
Source: Net Index by Ookla

Context

!Stocarea Datelor si Analiza

- Capacitatea de stocare a crescut dar viteza de acces a cunoscut o crestere mai mica
- (e.g. 1990, dispozitiv stoca 1370 MB, viteza de transfer: 4.4 MB/s, ~5 minute; 2010 dispozitiv de 1 terabytes, rata de transfer: 100 MB/s ~2.30 minute pentru citirea datelor)

=>citirea in paralel de pe disk-uri multiple

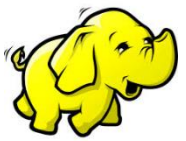
Probleme:

- Esecuri hardware
  - Solutie: replicare
  - Implementari: RAID, HDFS (Hadoop Distributed Filesystems),….
- In analiza datelor, este nevoie de combinare a datelor (in mod coerent) din surse diverse
  - O solutie: MapReduce
    - Model de programare care abstractizeaza nivelul operatiilor de citire/scriere in calcul asupra seturilor cheie-valoare

Hadoop

- “Hadoop provides: a reliable shared storage and analysis system”
- Hadoop Kernel
    - HDFS -> storage
    - MapReduce Software Framework -> analiza
- “Hadoop is designed to efficiently process large volumes of information by connecting many commodity computers together to work in parallel.”

- Creatorul: Doug Cutting, 2008
- Sursa:
  - GFS in perioada 2000
  - Apache Nutch – un motor de cautare opensource (inceput in 2002)
- Denumirea: “*The name my kid gave a stuffed yellow elephant. Short, relatively easy to spell and pronounce, meaningless, and not used elsewhere: those are my naming criteria*.” (Doug Cutting) ☺



Hadoop

- Aprilie 2008: Hadoop a devenit cel mai rapid sistem de sortare a datelor

"Hadoop sorted one terabyte in 209 seconds (just under 3½ minutes), beating the previous year's winner of 297 seconds (described in detail in "TeraByte Sort on Apache Hadoop" on page 553). In November of the same year, Google reported that its MapReduce implementation sorted one terabyte in 68 seconds. ……. (May 2009), it was announced that a team at Yahoo! used Hadoop to sort one terabyte in 62 seconds."

- Utilizari: Yahoo, Facebook,…

- …dar….(curs viitor)

Hadoop

- Comparatie cu sisteme existente
  - Condor – realizeaza procesarea intr-o infrastructura de tip grid
    - Nu permite distribuirea automata a datelor (un SAN separat trebuie administrat separat pentru un cluster)
    - Colaborarea intre noduri multiple se face apeland la un sistem de tip MPI
  - Hadoop – simplifica modelul de programare si permite scrierea rapida de cod si testarea sistemului distribuit
    - *Flat scalability*

Hadoop

Ecosistemul Hadoop:

Common

- *A set of components and interfaces for distributed filesystems and general I/O (serialization, Java RPC, persistent data structures).*

Avro

- *A serialization system for efficient, cross-language RPC, and persistent data storage.*

MapReduce

- ***A distributed data processing model and execution environment that runs on large clusters of commodity machines*.**

HDFS

- ***A distributed filesystem that runs on large clusters of commodity machines.***

Pig

- *A data flow language and execution environment for exploring very large datasets. Pig runs on HDFS and MapReduce clusters.*

Hadoop

Ecosistemul Hadoop:

Hive

- *A distributed data warehouse. Hive manages data stored in HDFS and provides a query language based on SQL (and which is translated by the runtime engine to MapReduce jobs) for querying the data.*

HBase

- *A distributed, column-oriented database. HBase uses HDFS for its underlying storage, and supports both batch-style computations using MapReduce and point queries (random reads).*

ZooKeeper

- *A distributed, highly available coordination service. ZooKeeper provides primitives such as distributed locks that can be used for building distributed applications.*

Sqoop

- *A tool for efficiently moving data between relational databases and HDFS.*

HDFS - Hadoop Distributed Filesystem

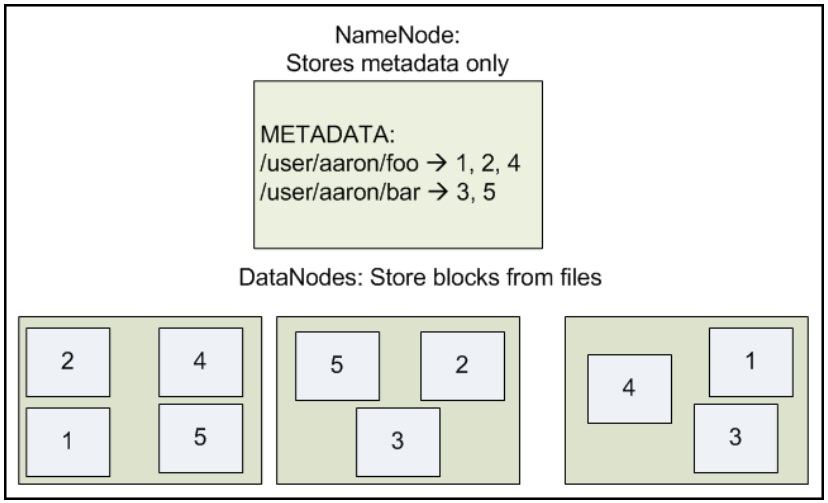
- "*HDFS is a filesystem designed for storing very large files with streaming data access patterns, running on clusters of commodity hardware*" (T. White)
  - *Very large files*
    - Fisiere cu dimensiuni de ordinul sutelor de megabytes, gigabytes sau terabytes
    - Suporta fisiere de dimensiune mai mare decat NFS
  - *Streaming data access*
    - HDFS a fost proiectat cu presupunerea ca patternul de procesare a datelor este: *write-once, read many times*

HDFS - Hadoop Distributed Filesystem

- Concepte si termeni
  - Block
    - Fiecare fisier este stocat ca o secventa de block-uri, de aceeasi dimensiune cu exceptia ultimului
    - Block-urile sunt replicate => *fault tolerance*
    - Dimensiunea block-urilor (implicit 64MB) si replicarea sunt parametrii configurabili
    - Avantajele aduse unui sistem de fisiere distribuit de abstractizarea cu block-uri:
      - Un fisier poate fi mai mare decat orice disk din retea
      - Rezistenta la erori si disponibilitatea

HDFS - Hadoop Distributed Filesystem

- Concepte si termeni
  - Metadatele sistemului de fisiere si datele sunt stocate separat
    - Metadatele sunt stocate pe ***NameNode***
    - Datele aplicatiilor sunt stocate pe servere numite ***DataNodes***
  - Serverele sunt conectate intre ele si comunica folosind protocoale TCP-based



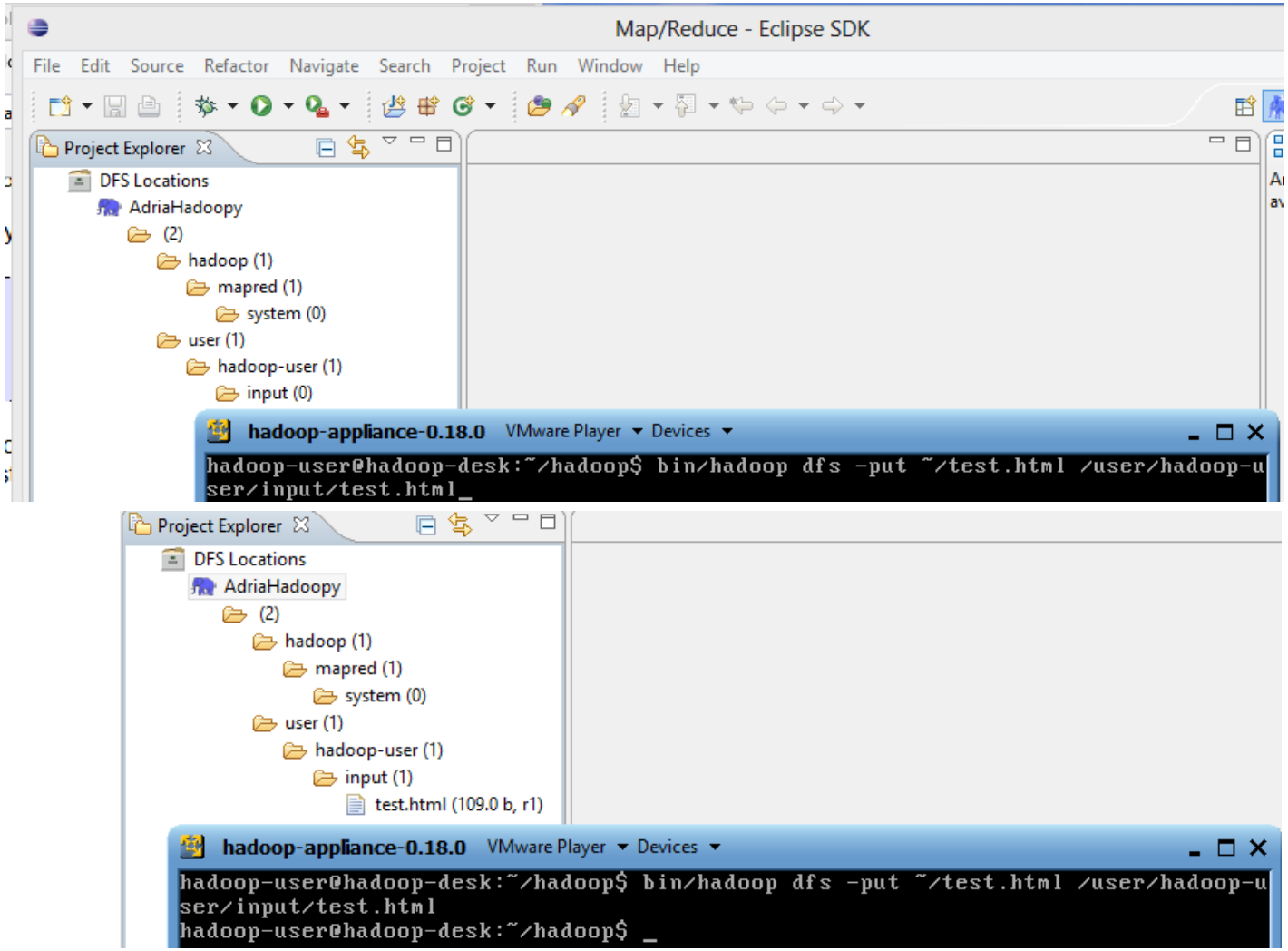
  - ls ? cp? mv? => ?
    - HDFS ruleaza intr-un **spatiu de nume** izolat de continutul sistemului de fisiere gazda

HDFS - Hadoop Distributed Filesystem

- Accesarea HDFS
  - Java API
  - wrapper C pentru Java API
  - FileSystem (FS) shell
  - DFSAdmin – un set de comenzi de administrare a clusterului HDFS
  - fsck – comanda folosita pentru verificarea inconsistentelor in HDFS
  - Eclipse plugin
  - ….

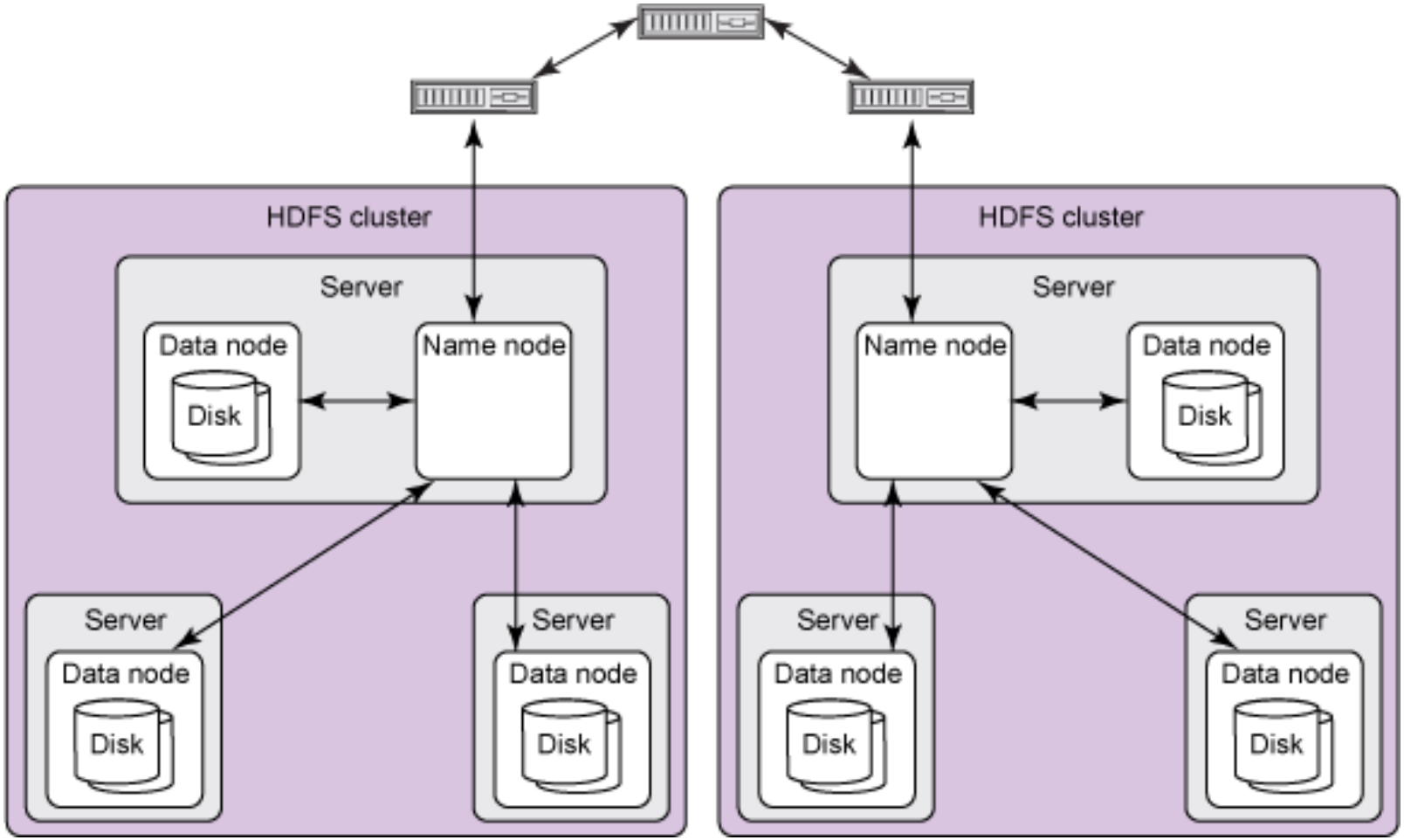
HDFS - Hadoop Distributed Filesystem

- Lucrul cu HDFS - exemplu



HDFS - Hadoop Distributed Filesystem

- Arhitectura



[http://www.ibm.com/developerworks/library/wa-introhdfs/]

HDFS - Hadoop Distributed Filesystem

- Arhitectura
  - ***NameNode*** – management:
    - Metadata formata din inoduri si lista de block-uri apartinand fiecarui fisier poarta numele de *Image*
    - Modificarile asupra *Image* sunt referite intr-un *Journal*
    - In timpul repornirii, *NameNode* restaureaza spatiul de nume pe baza *Journal*
    - Checkpoint-urile -> sunt inregistrari persistente ale imaginilor, stocate in sistemul de fisiere nativ local
    - Operatii:
      - deschidere, inchidere, redenumire fisiere si directoare
      - maparea blocurilor la *DataNode*-urile corespunzatoare

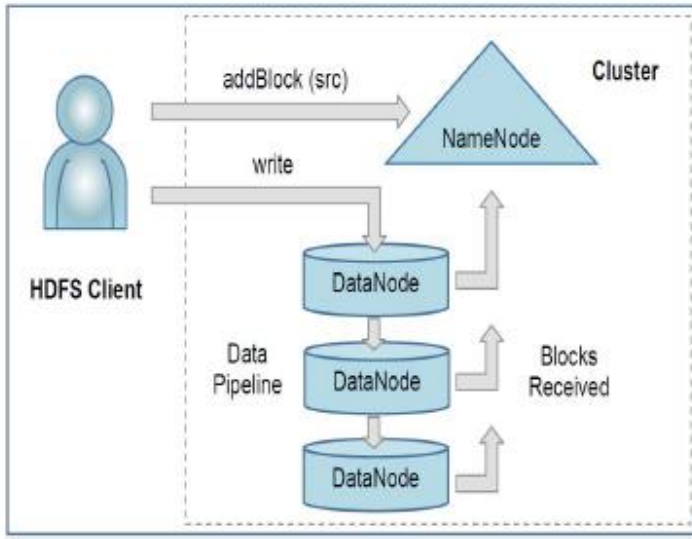
HDFS - Hadoop Distributed Filesystem

- Arhitectura
  - ***DataNode***
    - Operatii:
      - Creare, stergere si replicare a blocurilor de date conform instructiunilor *NameNode*
    - La pornire, un *DataNode* se conecteaza la un NameNode (*handshake*)
      - Verifica ID-ul spatiului de nume, versiunea de software a DataNode-ului
        - » In caz de nepotrivire, DataNode se inchide automat
      - *DataNode* identifica block-urile aflate in posesia sa, si trimite un raport (*block report*) la *NameNode*
        - » Raportul contine block ID, dimensiunea, *generation stamp*
      - Aceste rapoarte sunt trimise periodic, si asigura nodului *NameNode* o viziune actualizata asupra replicilor din cluster

HDFS - Hadoop Distributed Filesystem

Metode de acces

- *Read*
  - Cand o aplicatie citeste un fisier, clientul HDFS intreaba *NameNode* de lista de *DataNode* care contin replici ale block-urilor fisierului
  - Apoi comunicarea se face direct cu *DataNode*
- *Write*
  - Cand o aplicatie client doreste sa scrie, clientul HDFS intreaba NameNode sa ii furnizeze acele *DataNode* care sa contina replici ale primului block al fisierului
  - Clientul HDFS organizeaza un pipeline din nod-in-nod si trimite datele
  - Cand primul block este umplut, clientul cere noi DataNodes pentru a fi alesi sa gazduiasca replici ale urmatorului block (un nou pipeline este organizat, si clientul trimite urmatorii octeti ai fisierului)



[Mahesh Bharath Keerthivasan, Review of Distributed File Systems]

HDFS - Hadoop Distributed Filesystem

- **Sincronizarea**
  - HDFS implementeaza modelul *single-writer, multiple-reader*
  - Un client Hadoop, care deschide fisierul pentru operatia de *write*, are asigurata o perioada de *lease;*
    - aceasta perioada se reinnoieste periodic
    - La inchiderea fisierului *lease* este revocata
    - Operatia de citire este permisa
- **Replicarea**
  - Numarul de replici implicit este 3
  - Un NameNode detecteaza (si creste sau scade numarul de replici) daca se intimpla *under- sau* over-replica pe baza rapoartelor nodurilor DataNode
- **Consistenta**
  - Se face apel la sume de control pentru fiecare block
  - Aceste sume de control sunt verificate de client

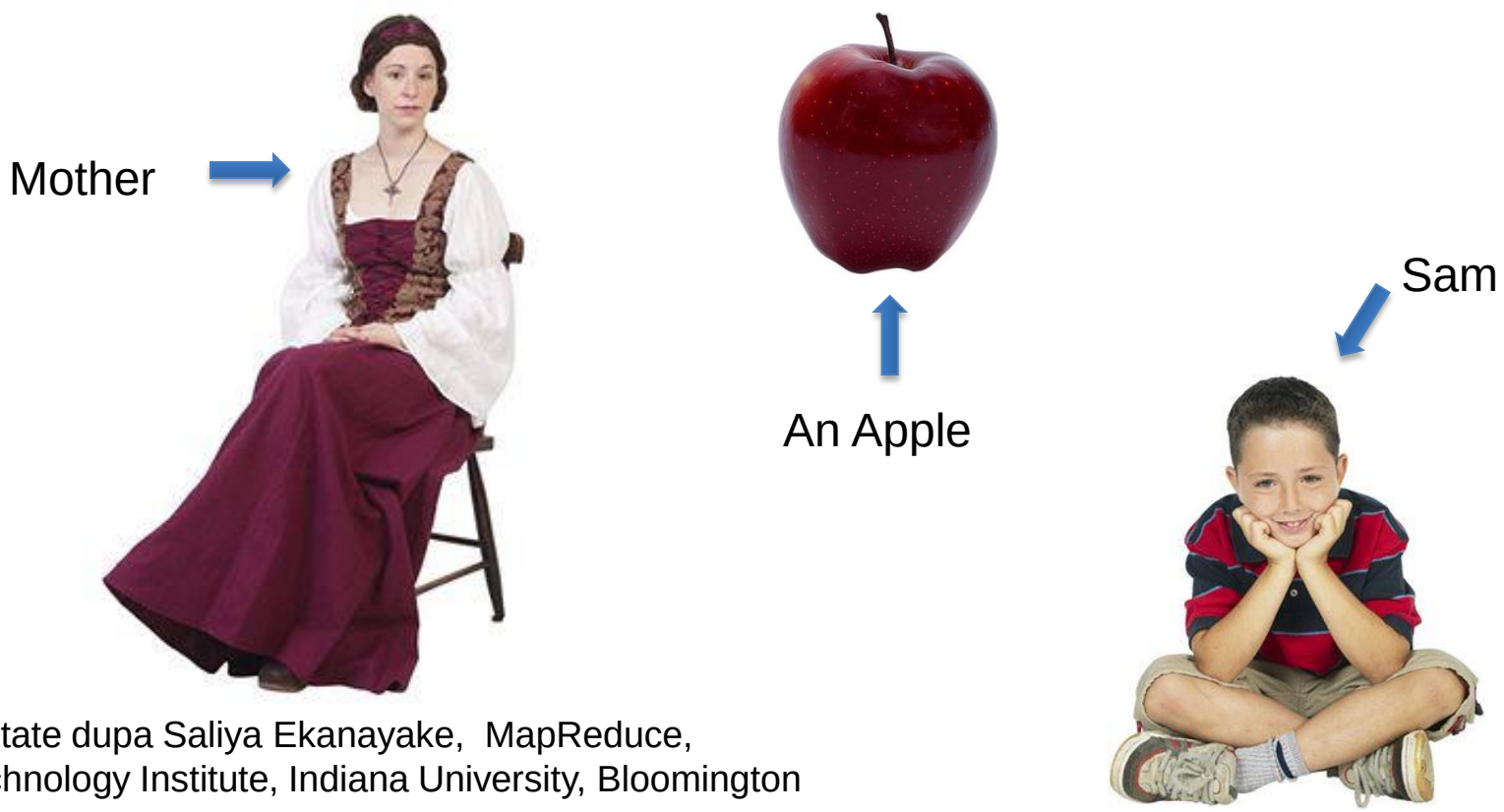
HDFS - Hadoop Distributed Filesystem

- Din modul de proiectare, HDFS este scalabil dar este destinat unei categorii mai restranse de aplicatii
  - *Low-latency data access*
    - Aplicatii care necesita un minim de latenta in accesul datelor (la nivel de zeci de milisecunde)
    - Obs: HDFS este optimizat pentru livrarea unei cantitati mari de date, iar acest lucru poate fi in detrimentul latentei
  - *Lots of small files*
    - Deoarece *namenode* tine metadatele asociate sistemului de fisiere in memorie, limita numarului de fisiere este guvernata de cantitatea de memorie a nodului; (e.g. stocarea de milioane de fisiere este fezabila, dar stocarea de bilioane depaseste capabilitatile hardware-ului curent)
  - *Multiple writers, arbitrary file modifications*
    - Fisierele in HDFS pot fi modificate de un singur *writer*; nu exista suport pentru scrieri multiple

Map Reduce

Sam's Mother

- Believed "an apple a day keeps a doctor away"

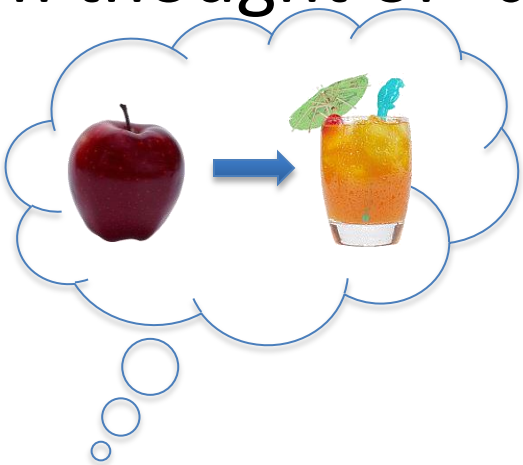


Slide-uri adaptate dupa Saliya Ekanayake, MapReduce, Pervasive Technology Institute, Indiana University, Bloomington

Map Reduce

One day

- Sam thought of “drinking” the apple



- He used a  to cut the 

and a  to make juice.



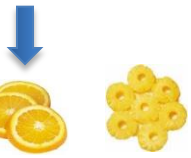


Map Reduce

Next Day

- Sam applied his invention to all the fruits he could find in the *fruit basket*

- (map  ‘(  



- (reduce  ‘(  

A *list of values* mapped into another *list of values*, which gets reduced into a *single value*

Classical Notion of MapReduce in Functional Programming

Slide-uri adaptate dupa Saliya Ekanayake, MapReduce, Pervasive Technology Institute, Indiana University, Bloomington

Map Reduce

18 Years Later

- Sam got his first job in JuiceRUs for his talent in making juice

Wait!

- Now, it's not just one basket but a whole *container* of fruits

- Also, they produce a *list* of juice types separately

Large data and list of values for output

- But, Sam had just ONE  and ONE 

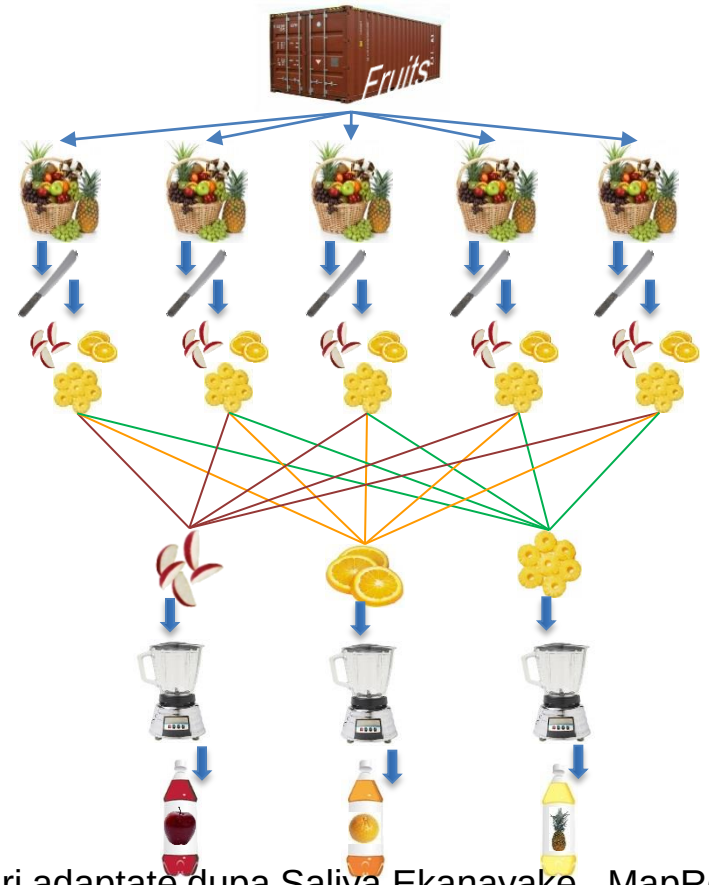
NOT ENOUGH !!

Slide-uri adaptate dupa Saliya Ekanayake, MapReduce, Pervasive Technology Institute, Indiana University, Bloomington

Map Reduce

Brave Sam

- Implemented a *parallel* version of his innovation



Each input to a map is a list of <key, value> pairs

(<a, > , <o, > , <p, > , …)

Each output of a map is a list of <key, value> pairs

(<a’, > , <o’, > , <p’, > , …)

Grouped by key

Each input to a reduce is a <key, value-list>
(possibly a list of these, depending on the grouping/hashing mechanism)
e.g. <a’, (…)>

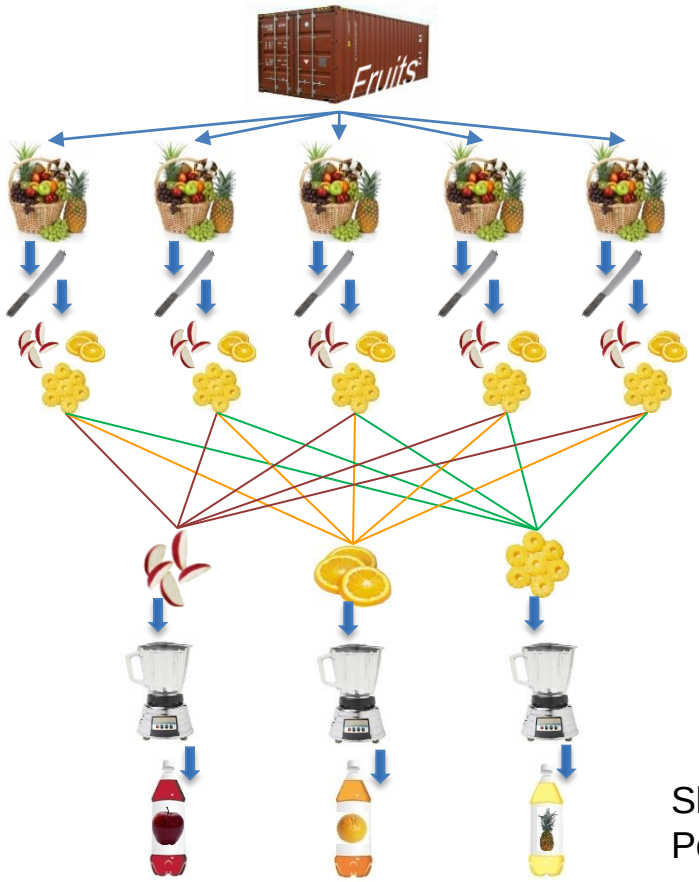
Reduced into a list of values

Slide-uri adaptate dupa Saliya Ekanayake, MapReduce, Pervasive Technology Institute, Indiana University, Bloomington

Map Reduce

Brave Sam

- Implemented a *parallel* version of his innovation



A *list of <key, value>* pairs mapped into another *list of <key, value>* pairs which gets grouped by the key and reduced into a *list of values*

The idea of MapReduce in Data Intensive Computing

Slide-uri adaptate dupa Saliya Ekanayake, MapReduce, Pervasive Technology Institute, Indiana University, Bloomington

Map Reduce

Afterwards

- Sam realized,
  - To create his favorite mix fruit juice he can use a *combiner* after the reducers
  - If several <key, value-list> fall into the same group (based on the grouping/hashing algorithm) then use the blender (reducer) separately on each of them
  - The knife (mapper) and blender (reducer) should not contain residue after use – *Side Effect Free*
  - In general reducer should be *associative* and *commutative*

We think Sam was you ☺

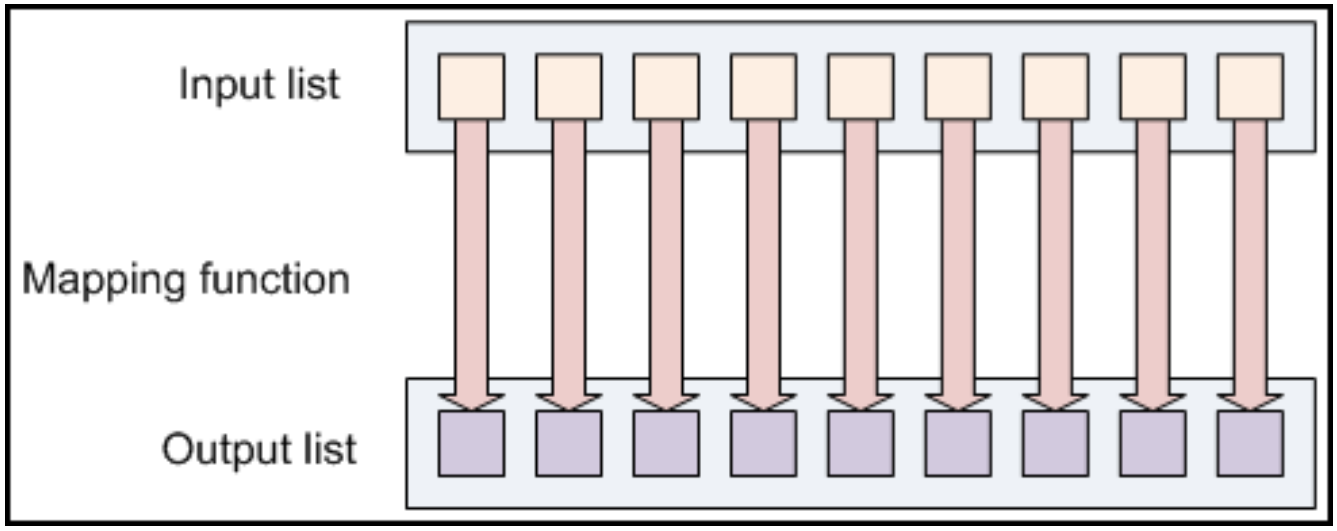
Slide-uri adaptate dupa Saliya Ekanayake, MapReduce, Pervasive Technology Institute, Indiana University, Bloomington

Map Reduce

- Este o metode de distribuite a taskurilor la noduri multiple
- Fiecare nod proceseaza date stocata pe acel nod => se evita crearea de trafic in retea
  - Atunci cand este posibil
- Caracteristici:
  - Distribuirea si paralelizarea automata
  - Rezistenta la erori
  - Instrumente de monitorizare
  - Oferirea unui nivel de abstractizare pentru programatori
    - Programatorul se concentreaza pe scrierea functiilor de Map si Reduce
- Consta din doua faze
  - ***Map***
  - ***Reduce***

Map Reduce

- ***Faza Map***
  - Transforma individual fiecare element de intrare intr-un element de iesire



Exemplu: *toUpper(str)* returneaza forma *uppercase* a unui string primit la intrare

Obs. nu are loc modificarea stringului de intrare, ci se returneaza un nou string care va face parte dintr-o lista de iesire

Map Reduce

- ***Faza Map***
  - Citeste datele in perechi *key/value*
  - Returneaza zero sau mai multe perechi *key/value*

map(in_key, in_value) -> (inter_key, inter_value) list

Obs. Mapper-ul poate ignora cheia de intrare, dar la iesire se obtin perechi *key/value*

Exemplu: citirea a cate unei linii dintr-un fisier (*key* = offset-ul byte-ului din fisier la care incepe linia, valoarea = continutul liniei; In acest caz cheia este irelevanta)

Map Reduce

- ***Faza Map***

Exemplu: WordCount – contorizeaza numarul de aparitii a unui cuvant in datele de intrare

Map(input_key, input_value)

foreach word w in input_value:

emit(w, 1)

Input pentru Mapper

(3414, 'the cat sat on the mat ')

(3437, 'the aardvark sat on the sofa‘)

Output de la Mapper

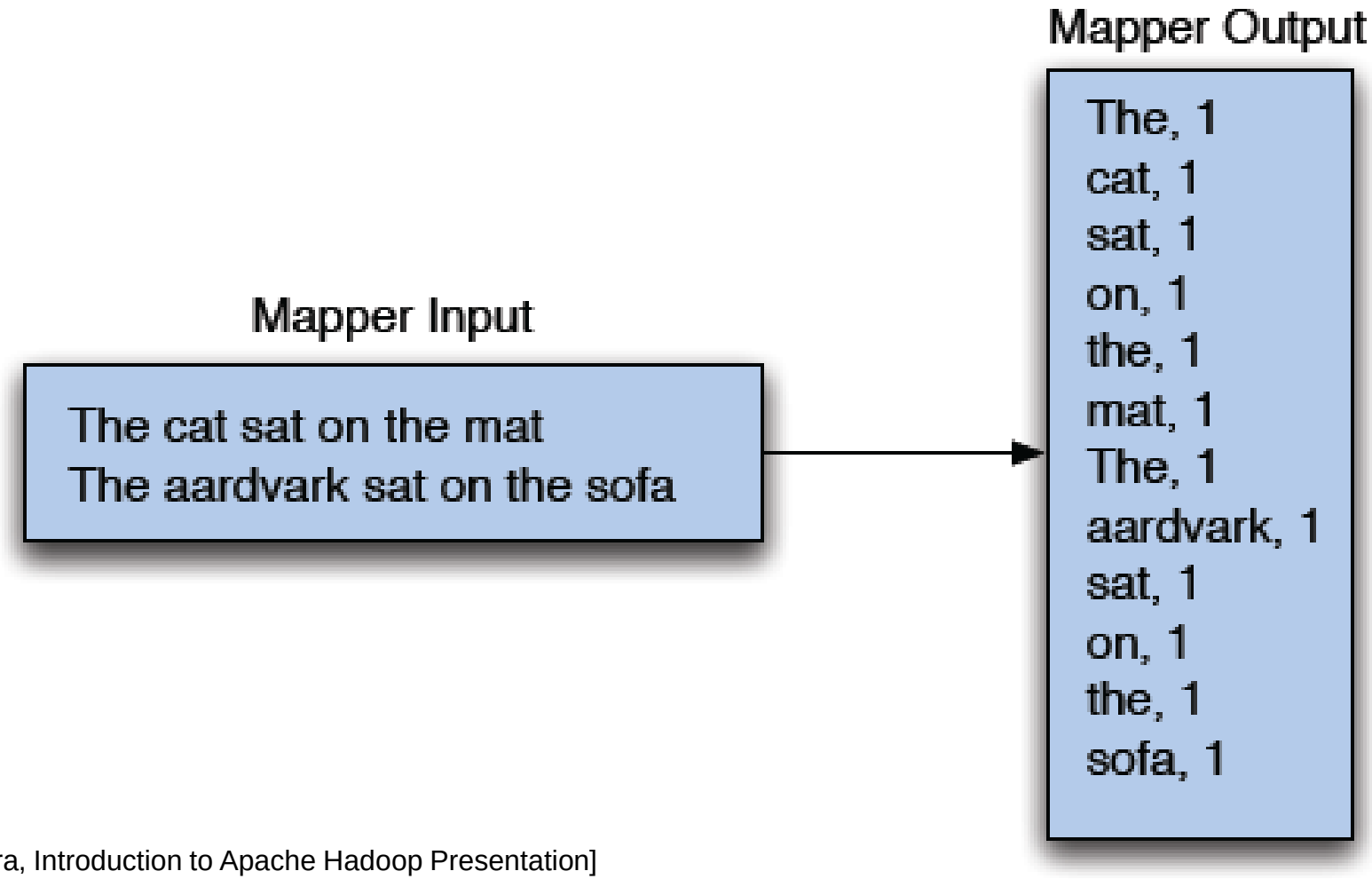
('the', 1), ('cat', 1), ('sat', 1), ('on', 1), ('the', 1), ('mat', 1),

('the', 1), ('aardvark', 1), ('sat', 1), ('on', 1), ('the', 1), ('sofa', 1)

[Cloudera, Introduction to Apache Hadoop Presentation]

Map Reduce

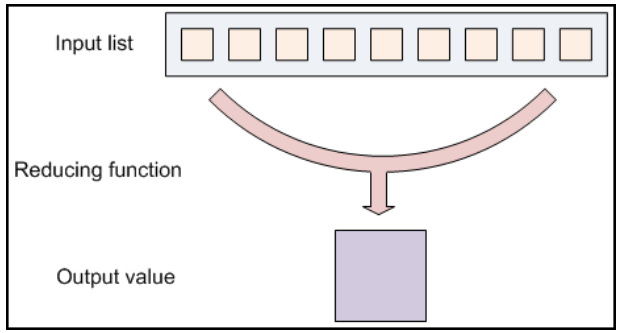
- ***Faza Map***



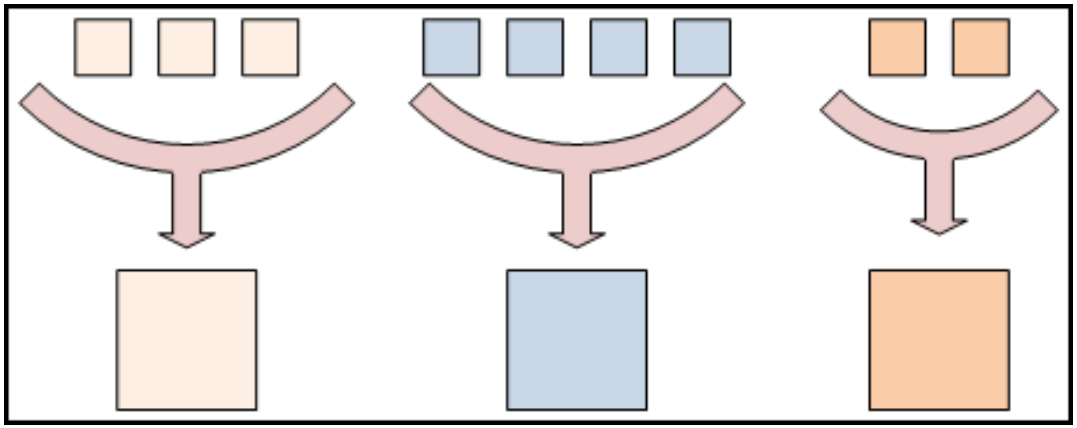
[Cloudera, Introduction to Apache Hadoop Presentation]

Map Reduce

- ***Faza Reduce***
  - Permite agregarea valorilor impreuna



  - Valoarile cu aceeasi *cheie* sunt preluate impreuna de un reducer



Map Reduce

- ***Faza Reduce***
  - Poate exista un singur sau mai multi *Reducers*
  - Valorile asociate unei chei sunt preluate de acelasi *Reducer*
  - Valorile trimise unui reducer sunt sortate dupa cheie
  - Reducer-ul duce la obtinerea a zero sau mai multe perechi finale *key/value*
    - Rezultatele sunt scrise in HDFS
    - Obs. In practica, un Reducer emite o pereche *key/value* pentru fiecare *key* de intrare
  - Pasul poarte si denumirea de "*shuffle and sort*"

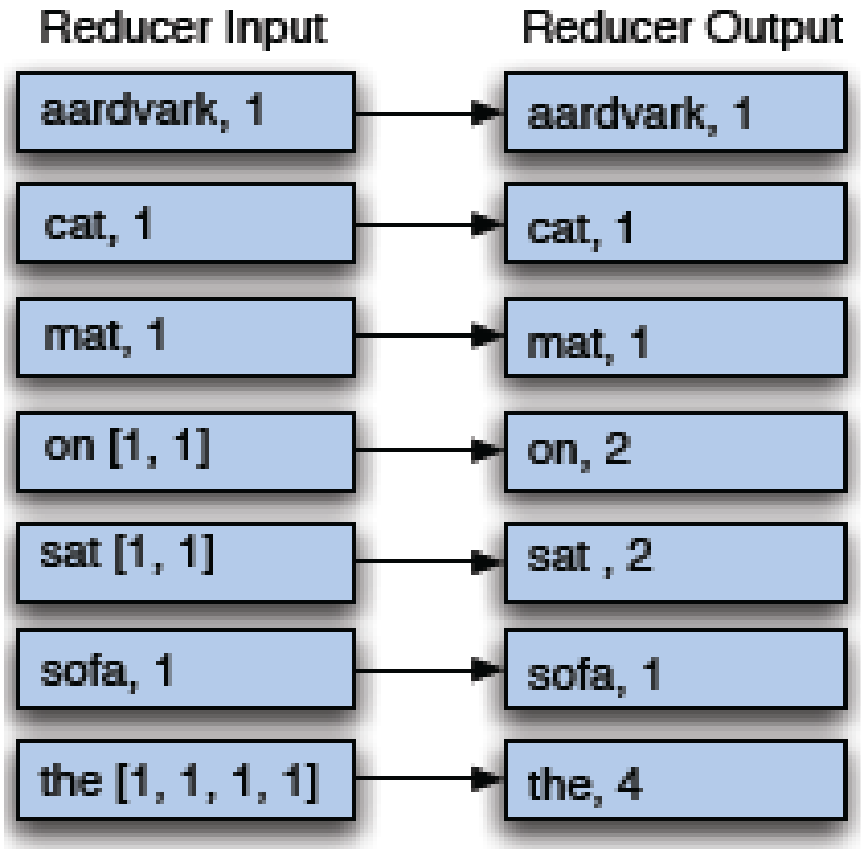
Map Reduce

- ***Faza Reduce***

reduce(output_key, intermediate_vals)
set count = 0
foreach v in intermediate_vals:
count += v
emit(output_key, count)

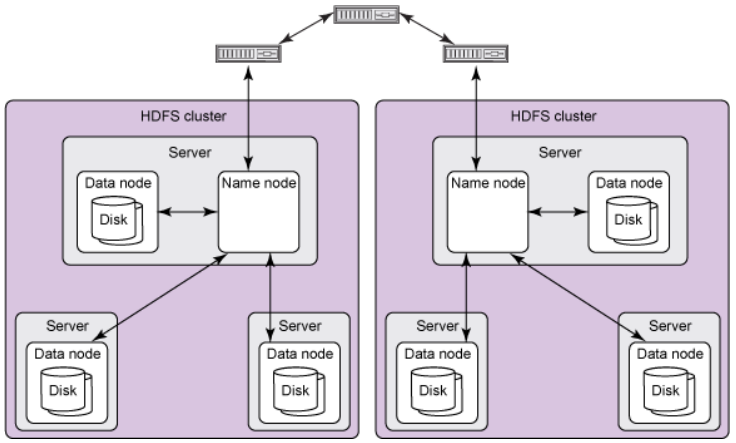
Rezultatul:

```
('aardvark', 1)
('cat', 1)
('mat', 1)
('on', 2)
('sat', 2)
('sofa', 1)
('the', 4)
```



[Cloudera, Introduction to Apache Hadoop Presentation]

Hadoop



Cluster Hadoop (continuare slide 18)

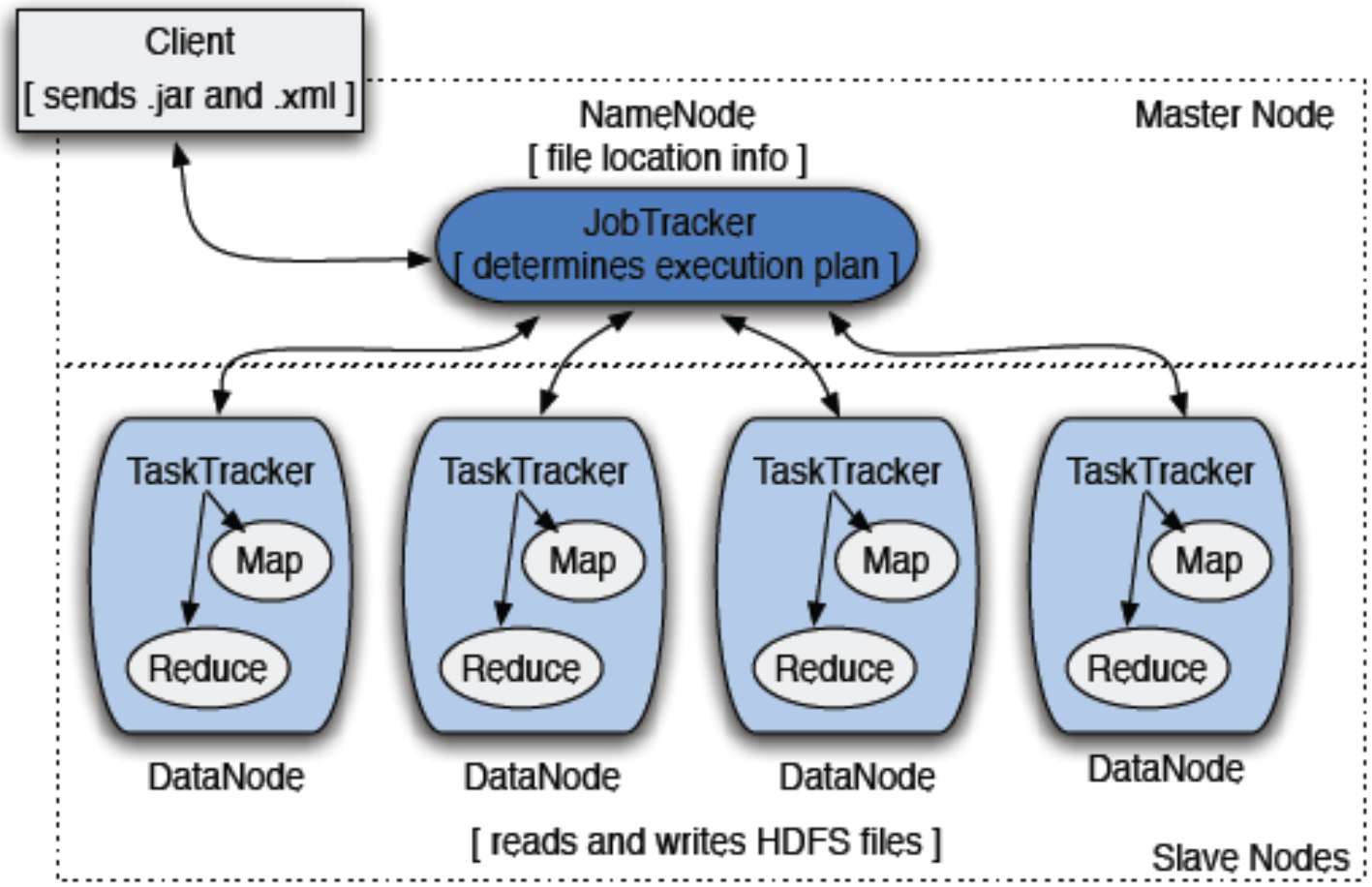
Hadoop este format din:

- **NameNode**
- **Secondary Name Node**
  - Nu este backup sau “hot standby” pentru NameNode
  - Realizeaza “housekeeping functions” pentru NameNode
- **DataNode**
- **JobTracker**
  - Realizeaza managementul job-urilor MapReduce (distribuirea taskurilor…)
- **TaskTracker**
  - Responsabil pentru instantierea si monitorizarea taskurilor individuale de Map si Reduce

[Cloudera, Introduction to Apache Hadoop Presentation]

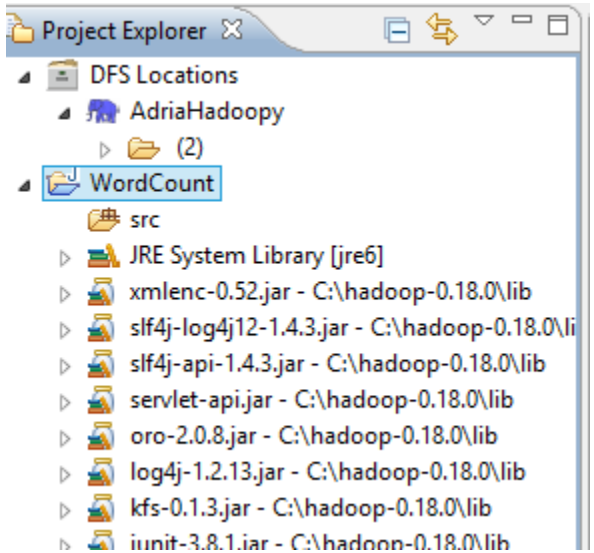
Hadoop

Cluster Hadoop



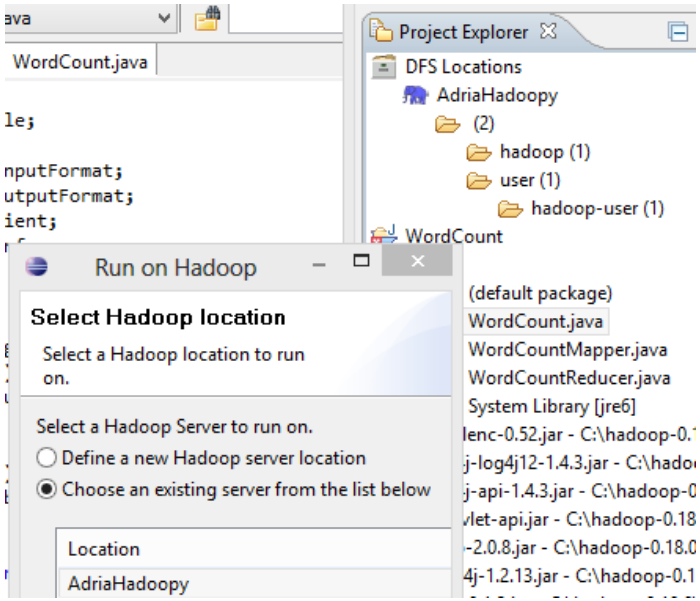
[Cloudera, Introduction to Apache Hadoop Presentation]

Map Reduce



- **Exemplu: WordCount**
- se creaza un proiect Map/Reduce

- Sunt necesare trei clase
  - Mapper si Reducer opereaza asupra datelor
  - Driver – specifica Hadoop cum sunt rulate procesele MapReduce



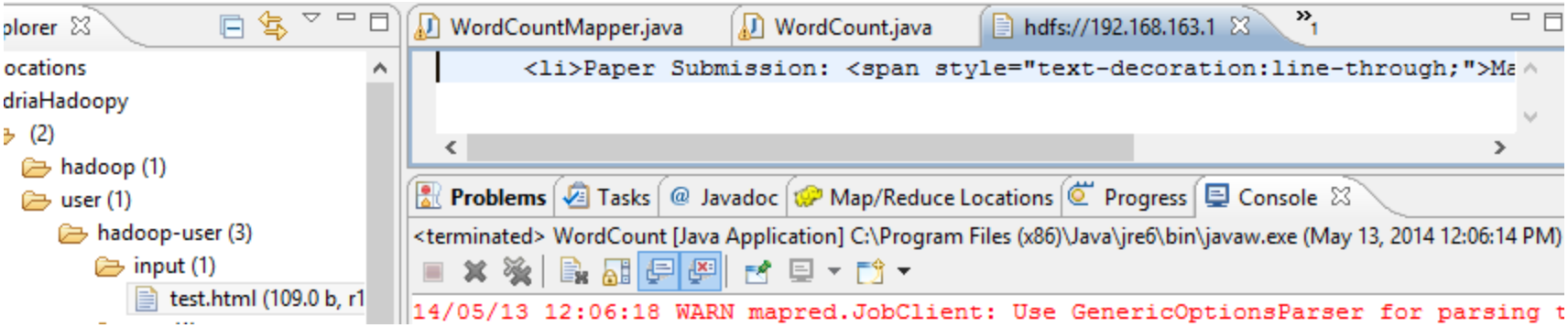
Link-uri utile:

- https://developer.yahoo.com/hadoop/tutorial/module3.html
- http://hadoop.apache.org/docs/r1.2.1/mapred_tutorial.html

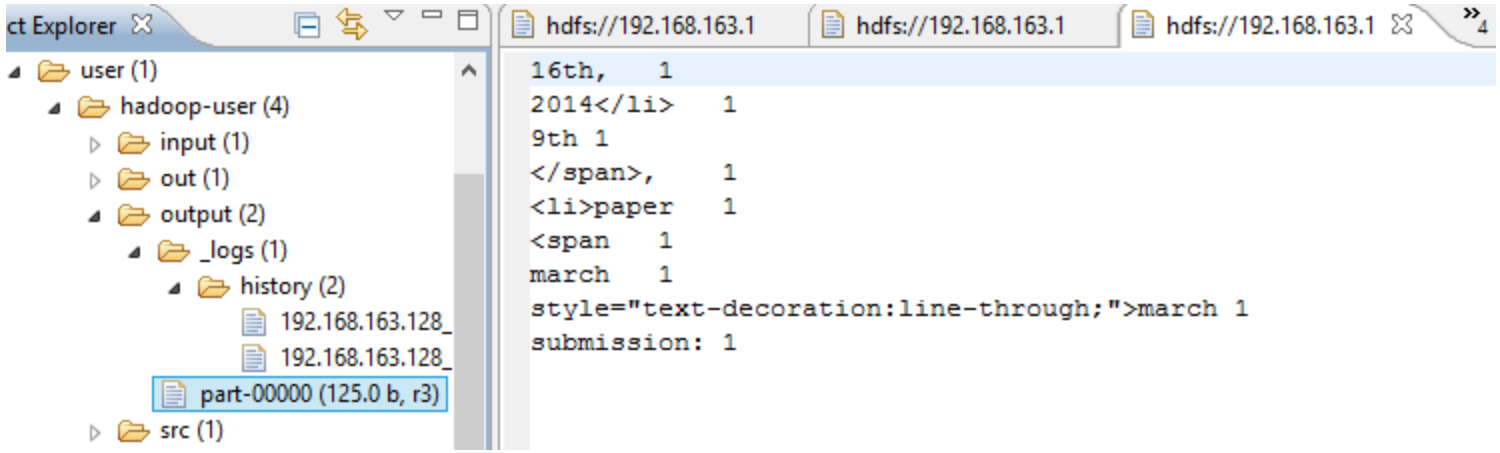
Map Reduce

- **Exemplu: WordCount**

Input:

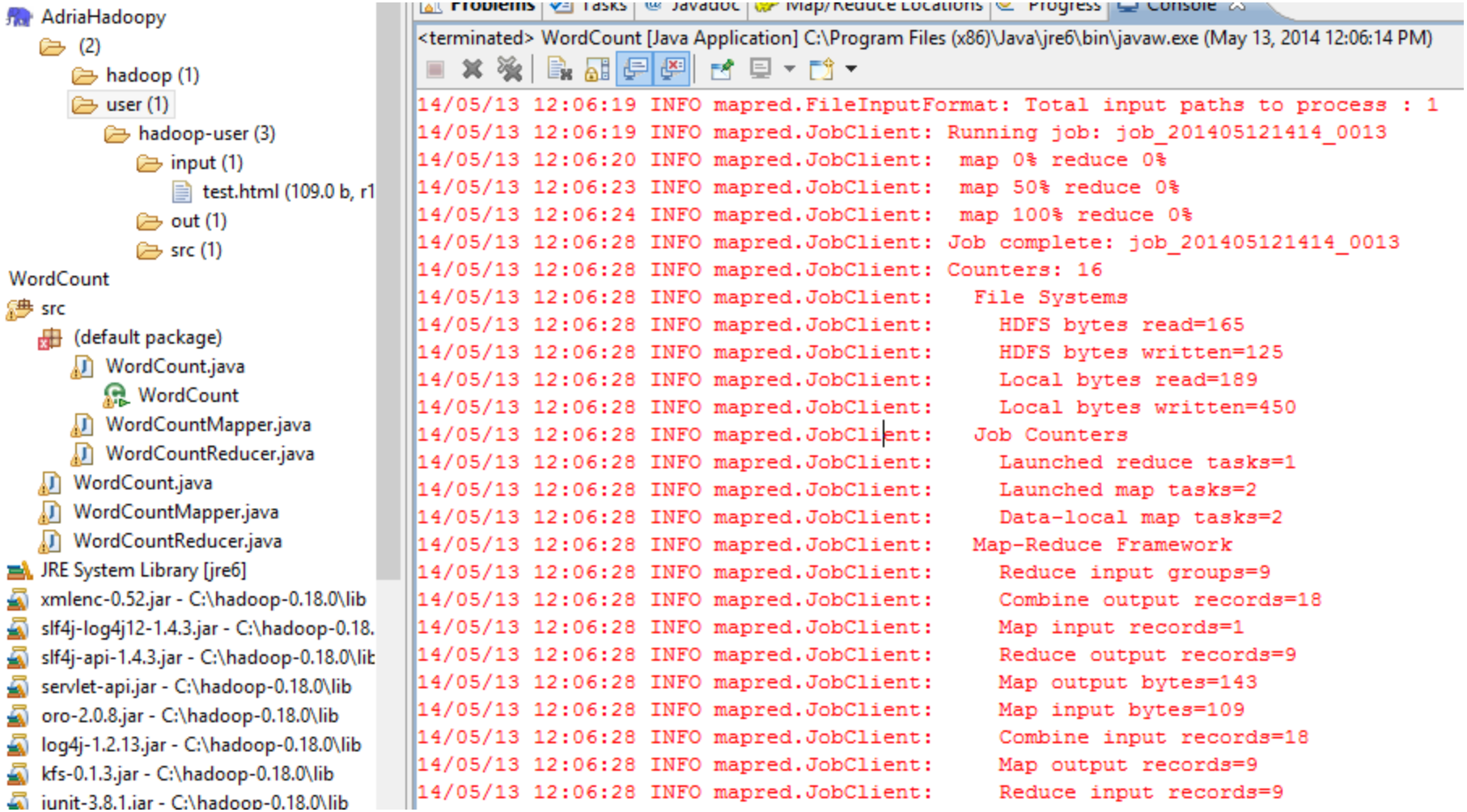


Output:



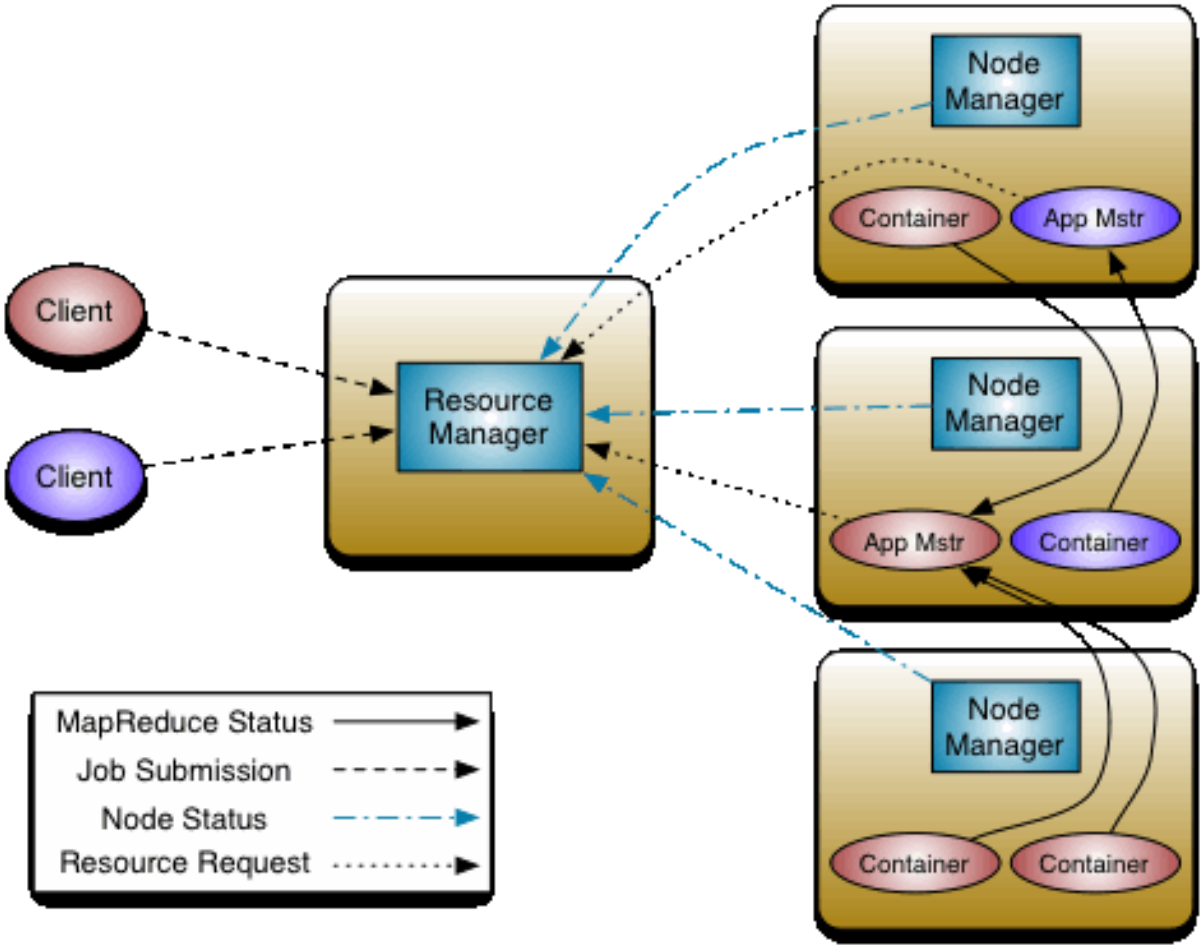
Map Reduce

- **Exemplu: WordCount**



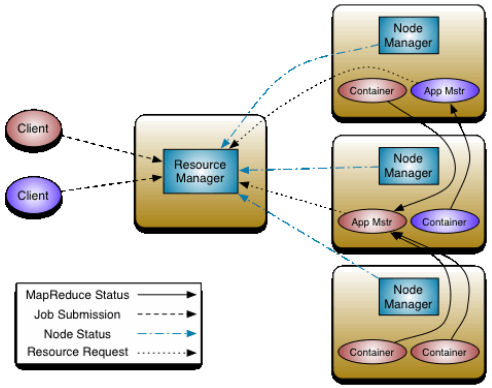
YARN

Cluster Hadoop (continuare slide 18)



[http://hadoop.apache.org/docs/r2.3.0/hadoop-yarn/hadoop-yarn-site/YARN.html]

YARN



- MapReduce 2.0 (MRv2)
- Diferente:
  - Impartirea responsabilitatilor unui JobTracker

    (resource management and job scheduling/monitoring) la demoni separati
  - Exista un Resource Manager (RM) global care mediaza utilizarea resurselor intre toate aplicatiile
    - Impreuna cu NodeManager formeaza *data-computation framework*
  - Exista cate un ApplicationMaster (⇔ o biblioteca specifica) per aplicatie (⇔ un job in versiunea MapReduce 1)
    - Impreuna cu Node Manager executa si monitorizeaza taskurile

[http://hadoop.apache.org/docs/r2.3.0/hadoop-yarn/hadoop-yarn-site/YARN.html]

Bibliografie

- Tom White, Hadoop-The definitive Guide, Second edition, O'Reilly, 2011
- Andrew S. Tanenbaum, Maarten van Steen, Distributed Systems, Principles and Paradigms, Second Edition, 2007
- Ajay D. Kshemkalyani , Mukesh Singhal , Distributed Computing - Principles, Algorithms, and Systems, © Cambridge University Press 2008
- http://www.cs.berkeley.edu/~brewer/cs262b-2004/Lec-AFS-GFS.pdf
- https://wiki.engr.illinois.edu/display/cs598rco/The+Google+File+System+-+Zhijin+Li
- http://www.cs.brown.edu/courses/cs295-11/2006/gfs.pdf
- http://www.ibm.com/developerworks/web/library/wa-introhdfs/index.html?ca=drs-
- *http://hadoop.apache.org/*
- Gantz et al., "The Diverse and Exploding Digital Universe," March 2008 http://www.emc.com/collateral/analyst-reports/diverse-exploding-digital-universe.pdf
- Mahesh Bharath Keerthivasan, Review of Distributed File Systems:Concepts and Case Studies, Dept. of Electrical & Computer Eng., University of Arizona, Tucson
- http://www.intelligententerprise.com/showArticle.jhtml?articleID=207800705, http://mashable.com/2008/10/15/facebook-10-billion-photos/
- http://www.northeastern.edu/levelblog/2016/05/13/how-much-data-produced-every-day/
- http://www.vcloudnews.com/every-day-big-data-statistics-2-5-quintillion-bytes-of-data-created-daily/

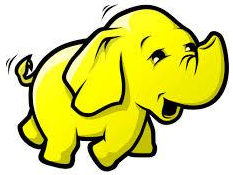
Bibliografie

- http://blog.familytreemagazine.com/insider/Inside+Ancestrycoms+TopSecret+Data+Center.aspx, and http://www.archive.org/about/faqs.php, http://www.interactions.org/cms/?pid=1027032
- Mike Cafarella and Doug Cutting, “Building Nutch: Open Source Search,” *ACM Queue, April 2004, http://*

queue.acm.org/detail.cfm?id=988408

- Zhang S., “Distributed Filesystems Review”,Online Presentation, http://www.slideshare.net/schubertzhang/distributedfilesystems-review
- “The Hadoop Distributed File System” by Konstantin Shvachko, Hairong Kuang, Sanjay Radia, and Robert Chansler (Proceedings of MSST2010, May 2010, *http://storageconference.org/2010/Papers/MSST/Shvachko.pdf).*
- Cloudera, Introduction to Apache Hadoop,
- LustreFile System. http://www.oracle.com/us/products/servers-storage/storage/storage-software/031855.htm
- Saliya Ekanayake, MapReduce, Pervasive Technology Institute, Indiana University, Bloomington

Rezumat

- Context
- Hadoop – imagine generala 
  - Componente
- (II) **HDFS - Hadoop Distributed Filesystem**
  - Caracteristici
  - Concepte
  - Arhitectura
  - HDFS versus GFS (vezi curs anterior)
- **Map Reduce….** YARN

Întrebări?