# BIGDATA SI SCALABILITATE

Hadoop HDFS

# Presupuneri si scopuri in HDFS

# Presupuneri si scopuri in HDFS

## Defectarea hardware

- Defectarea hardware este tratata ca un eveniment normal si nu ca o exceptie
- Datorita numarului mare de masini ( de ordinul miilor ) probabilitatea de defectare hardware este ridicata
- Detectarea defectarii este rapida
- Revenirea automata este unul din scopurile arhitecturale ale HDFS

# Presupuneri si scopuri in HDFS

## Accesul la date ca stream

- HDFS nu este gandit pentru uz general ci pentru procesare in masa
- Optimizeaza troughput si nu latenta ( ca un tren )

# Presupuneri si scopuri in HDFS

## Seturi mari de date

- Un fisier tipic din HDFS stocheaza zeci de GB de informatie
- Este optimizat pentru lucrul cu astfel de fisiere

# Presupuneri si scopuri in HDFS

## Model simplu de access la fisiere

- Scrie o data, citeste de multe ori ( read-once-write-many )
- Dupe ce fisierul a fost creat, scris si inchis el NU se mai pote modifica!
- Sunt planuri pentru a suporta scrieri la sfarsitul unui fisier

# Presupuneri si scopuri in HDFS

Mutarea procesarii este mult mai ieftina decat mutarea datelor

# Presupuneri si scopuri in HDFS

## Portabilitate peste sisteme eterogene

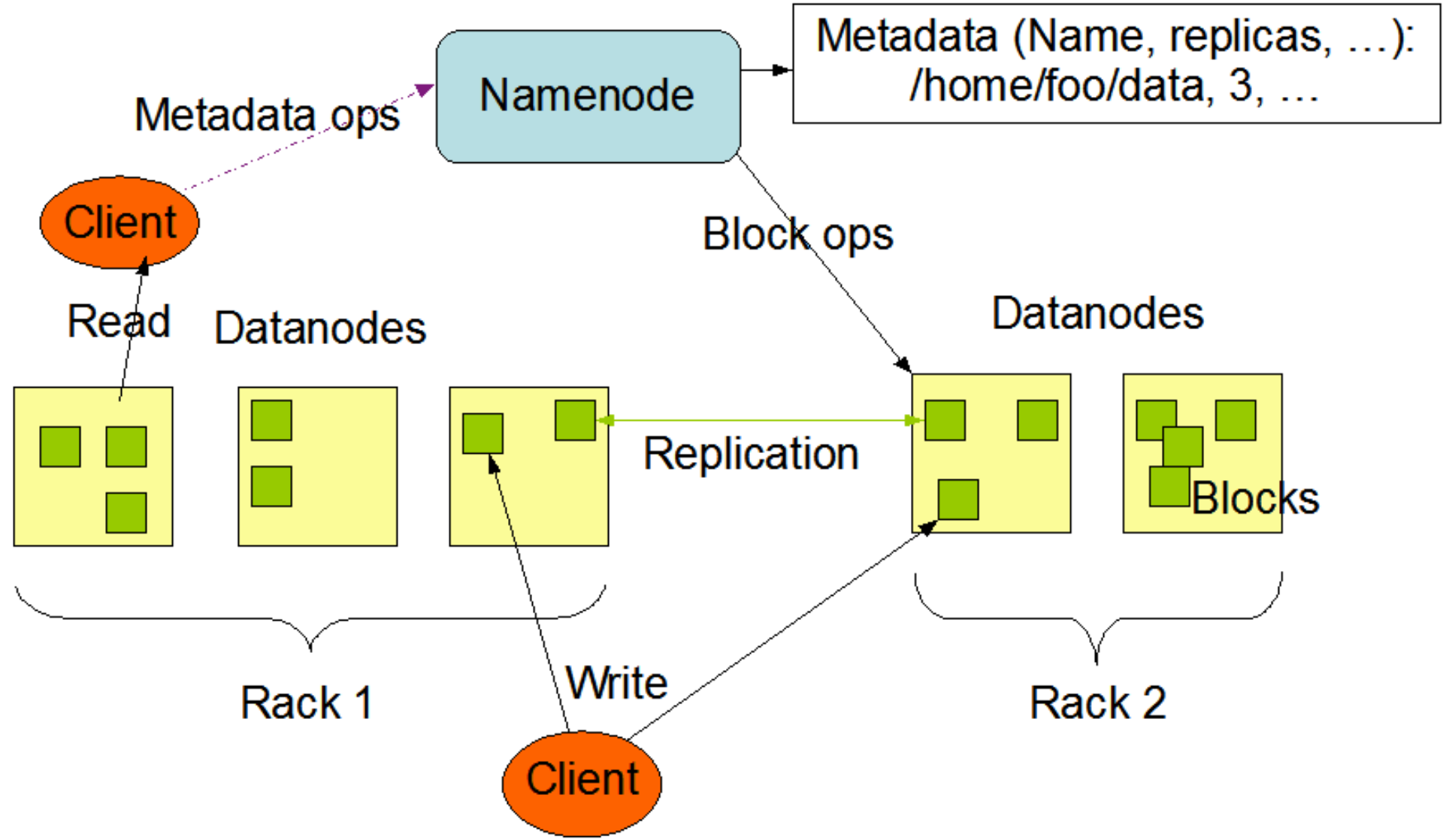
- Nu depinde de sistemul de operare
- Poate rula pe Windows, UNIX, masini de spalat, ceasuri de buzunar, orice combinatie de hardware si software care are o masina virtuala java si un sistem de fisiere local

# Anatomia HDFS

# NameNode si DataNode-uri

- Un singur NameNode care gestioneaza spatiul fisierelor si accesul clientilor la fisiere
- Multe DataNode-uri care gestioneaza stocarea atasata masinilor pe care ruleaza. Sunt cele care stocheaza datele din sistem.
- Fisierele sunt sparte in bucati si salvate in DataNode-uri
- NameNode-ul executa operatii pe spatiul de fisiere: deschidere, creare, redenumire, mutare etc.
- DataNode-urile servesc cererile de citire/scriere

# Arhitectura HDFS



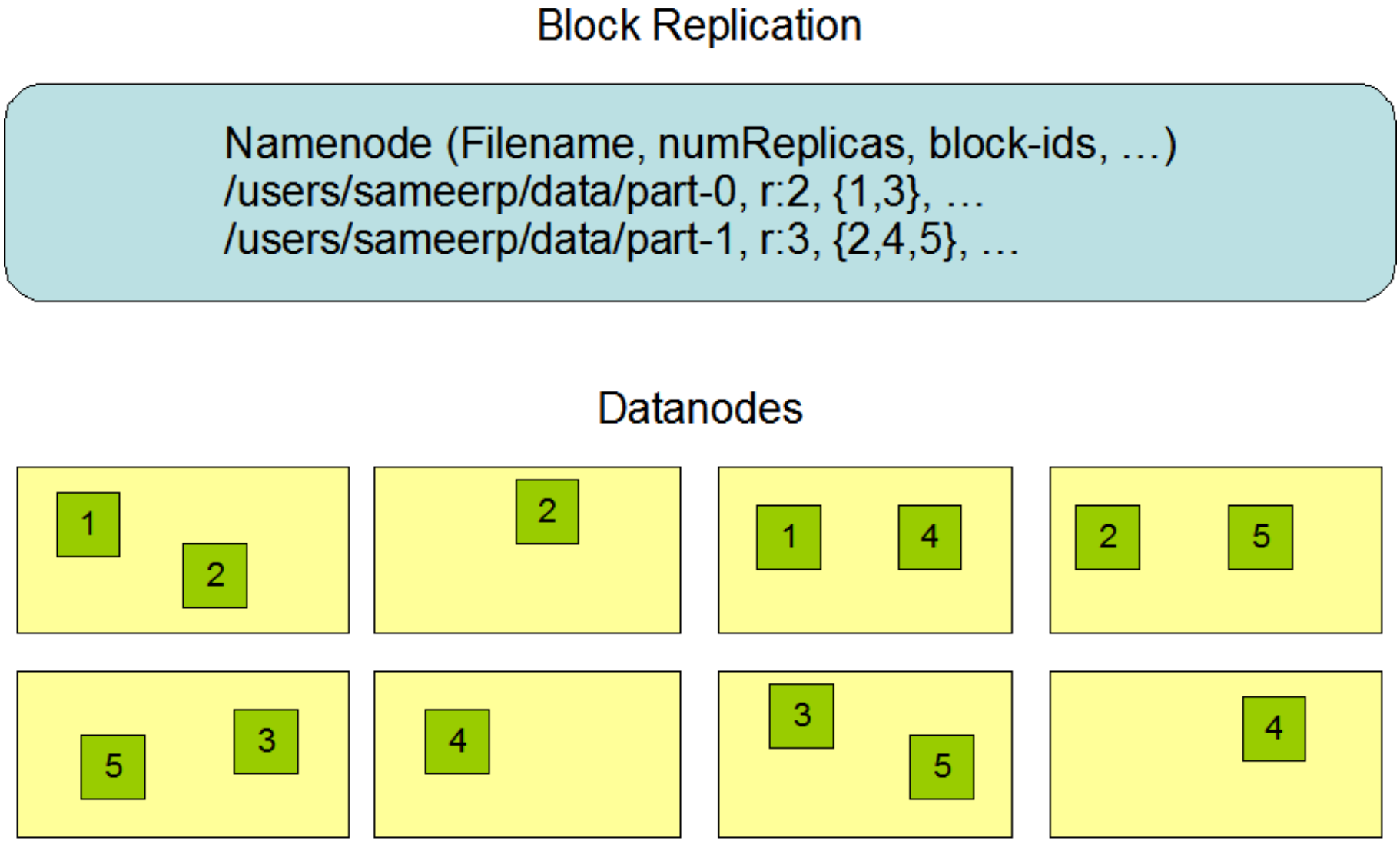
# Spatiul numelor fisierelor

- HDFS suporta modelul traditional de organizare ierarhica
- Un utilizator poate crea directoare si stoca fisiere in directoare. Poate muta, redenumi, sterge fisiere.
- NU suporta limitarea spatiului per utilizator (user quota) si nici legaturi simbolice/hardware.

# Replicare

- Fiecare fisier e stocat ca o secventa de blocuri
- Toate blocurile cu exceptia ultimului au aceeasi dimensiune
- Dimensiunea blocurilor si factorul de replicare este configurabil per fisier
- Factorul de replicare este specificat la creare si poate fi schimbat ulterior
- NameNode-ul ia toate deciziile legate de replicare si primeste un hearbeat de la fiecare DataNode. Heartbeat-ul presupune functionarea corecta da DataNode-ului

# Replicare


Block Replication
Namenode (Filename, numReplicas, block-ids, …)
/users/sameerp/data/part-0, r:2, {1,3}, …
/users/sameerp/data/part-1, r:3, {2,4,5}, …
Datanodes
1
2
2
1
4
2
5
5
3
4
3
5
4

# Replicate: Rack awareness

- Trebuie activata ( un flag in fisier de configurare )
- Pe fiecare masina care ruleaza DataNode se ruleaza un script specificat de utilizator care intoarce un id. Id-ul este considerat id-ul rack-ului sub care este masina
- Ca exemplu, se poate folosi ip-ul unei masini pentru a determina rack-ul din care face parte

# Locatia replicilor

- Locatia replicilor e critica pentru performanta si disponibilitate
- In general latimea de banda intre 2 masini sub acelasi switch este mai mare decat intre 2 masini sub switch-uri diferite
- Politica pentru determinarea locatie replicilor nu este finala in hadoop. Este inca in stare experimentala!

# Locatia replicilor

O politica simpla dar neoptima este sa plasezi fiecare replica in alt rack

- Avantaje:
  - Nu se pierd date cand un intreg rack se defecteaza
  - La citirea datelor se foloseste latime de banda din mai multe rack-uri
  - Replicile sunt distribuite in mod egal
- Dezavantaje
  - Costul de scriere este marit pentru ca datele sunt trimise catre rack-uri diferite

# Locatia replicilor: replicare 3

- Replica 1: stocata pe o masina din rack-ul local
- Replica 2: stocata pe o masina din alt rack
- Replica 3: stocata in acelasi rack cu replica 2 dar pe o masina diferita

# Locatia replicilor: replicare 3

Observatii

- Reduce traficul intre rack-uri si in cele mai multe cazuri creste performanta la scriere
- Probabilitatea de defectare a unui rack este mult mai mica decat a unui nod, deci nu se pierde siguranta disponibilitatii
- Replicile NU sunt distribuite in mod egal
- Politica este inca in dezvoltare!

# Persistenta spatiului de nume

- Se folosete de o combinatie de 2 sisteme de backup
  - Unul bazat pe snapshot-uri/checkpoint-uri (FsImage)
  - Altul bazat pe loguri/diff-uri (EditLog)
- La un interval de timp se construieste un checkpoint din cel precedent si lista de log-uri

# Robust

- Heartbeat
- Re-Replication
- Cluster rebalancing
- Checksum pentru integritate

# Permisii pentru fisiere

- hdfs foloseste user-ul de pe masina pe care ruleaza clientul
- Daca in sesiunea shell esti autentificat ca user-ul ‘xulescu’ si executi o comanda hdfs pe fisierul ‘X’, user-ul ‘xulescu’ trebuie sa aiba permisii pe fisierul ‘x’
- Hdfs nu creaza/gestioneaza useri
- Utilizatorul ‘root’ este cal care a pornit NameNode-ul.

# Lucrul cu HDFS

# Interfete web

- Serviciile hdfs ruleaza cate un server http care ofera date statistice si comenzi uzuale
- http://namenode-name:50070/

# Interfete web

## NameNode 'master:9000'

**Started:** Wed Oct 23 11:43:26 UTC 2013
**Version:** 1.2.1, r1503152
**Compiled:** Mon Jul 22 15:23:09 PDT 2013 by mattf
**Upgrades:** There are no upgrades in progress.

Browse the filesystem
Namenode Logs

---

### Cluster Summary

**6 files and directories, 1 blocks = 7 total. Heap Size is 46.11 MB / 966.69 MB (4%)**

| | | |
|---|---|---|
| **Configured Capacity** | : | 236.64 GB |
| **DFS Used** | : | 120 KB |
| **Non DFS Used** | : | 19.55 GB |
| **DFS Remaining** | : | 217.09 GB |
| **DFS Used%** | : | 0 % |
| **DFS Remaining%** | : | 91.74 % |
| **Live Nodes** | : | 3 |
| **Dead Nodes** | : | 0 |
| **Decommissioning Nodes** | : | 0 |
| **Number of Under-Replicated Blocks** | : | 0 |

---

### NameNode Storage:

| Storage Directory | Type | State |
|---|---|---|
| /tmp/hadoop-root/dfs/name | IMAGE_AND_EDITS | Active |

# Interfete web

## NameNode 'master:9000'

**Started:** Wed Oct 23 11:43:26 UTC 2013
**Version:** 1.2.1, r1503152
**Compiled:** Mon Jul 22 15:23:09 PDT 2013 by mattf
**Upgrades:** There are no upgrades in progress.

Browse the filesystem
Namenode Logs
Go back to DFS home

### Live Datanodes : 3

| Node | Last Contact | Admin State | Configured Capacity (GB) | Used (GB) | Non DFS Used (GB) | Remaining (GB) | Used (%) | Used (%) | Remaining (%) | Blocks |
|---|---|---|---|---|---|---|---|---|---|---|
| hadoop1 | 1 | In Service | 78.88 | 0 | 6.52 | 72.36 | 0 | | 91.74 | 1 |
| hadoop2 | 2 | In Service | 78.88 | 0 | 6.52 | 72.36 | 0 | | 91.74 | 1 |
| hadoop3 | 1 | In Service | 78.88 | 0 | 6.52 | 72.36 | 0 | | 91.74 | 1 |

# Linia de comanda dfs

- Un client dfs care inetractioneaza cu hdfs
- bin/hadoop dfs <comanda> <argumente>
- Seamana cu comenzile uzuale bash
- Documentatie: http://hadoop.apache.org/docs/r1.2.1/file_system_shell.html

# Linia de comanda dfs

- cat
- chgrp
- chmod
- chown
- copyFromLocal
- copyToLocal
- count
- cp
- du
- dus
- expunge
- get
- getmerge
- ls
- lsr
- mkdir
- moveFromLocal
- moveToLocal
- mv
- put
- rm
- rmr
- setrep
- stat
- tail
- test
- text
- touchz

# Link-uri utile

- HDFS user guide: http://hadoop.apache.org/docs/r1.2.1/hdfs_user_guide.htm
- HDFS architecture: http://hadoop.apache.org/docs/r1.2.1/hdfs_design.html
- File system shell: http://hadoop.apache.org/docs/r1.2.1/file_system_shell.html