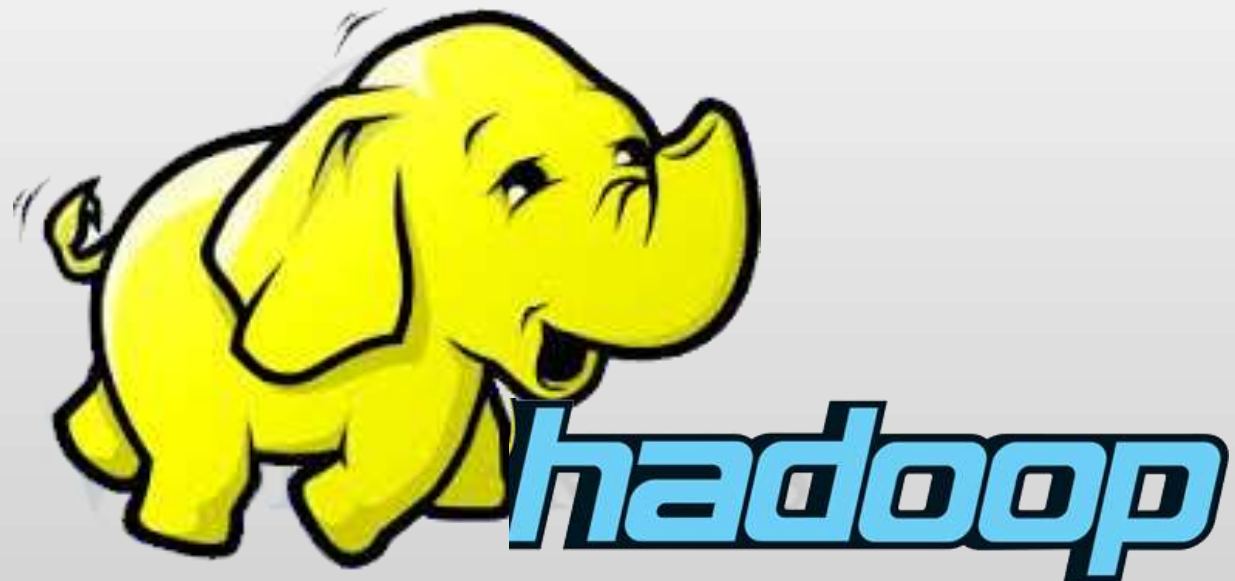


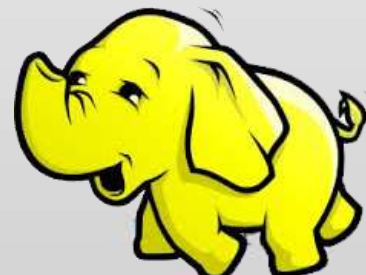
```
<?php:
<?php
    * @package WordPress
    * @subpackage DefaultTheme
    */
?>
<!DOCTYPE html PUBLIC "-//W3C
<html xmlns="http://www.W3.
<head profile="http://
<meta http-equiv="Con
<title><?php wp
<link rel=""
<link rel=""
<sty/
<
<
```



**1. Introduction: Hadoop's history and advantages**

**2. Architecture in detail**

**3. Hadoop in industry**



```
<?php
/* [package WordPress
 * [subpackage Default_Theme
 */
?>
<!DOCTYPE html PUBLIC "-//W3C
<html xmlns="http://www.w3.

<head profile="http://r
<meta http-equiv="C

<title><?php vp

<link rel="
<link rel

<body>
```

## Introduction to Hadoop...

- **Hadoop is an open source framework that allows us to store & process large data sets in a parallel & distributed manner.**
- **Doug Cutting and Mike Cafarella.**
- **Two main components HDFS & MapReduce.**
- **Hadoop Distributed File System (HDFS) is the primary data storage system used by Hadoop applications.**
- **MapReduce is the processing unit of Hadoop.**



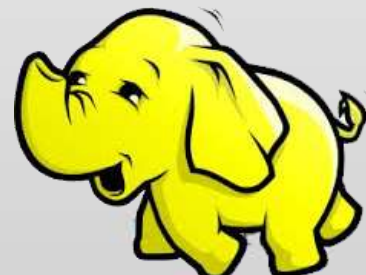
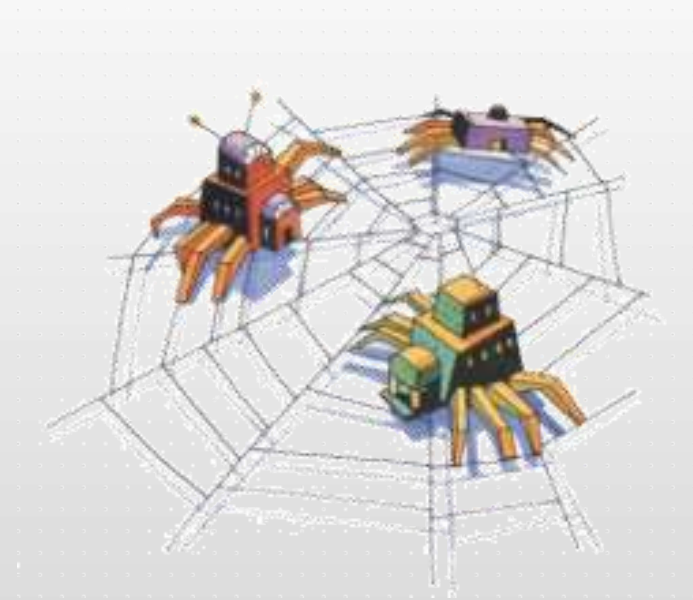
```
<?php
<!-- [package WordPress
<!-- [package Default_Theme
<?
<!DOCTYPE html PUBLIC "-//W3C
<html xmlns="http://www.w3.
<head profile="http://
<meta http-equiv="Con
<title><?php echo
<link rel="
<link rel="
<script>
```



- Apache top level project, open-source implementation of frameworks for reliable, scalable, distributed computing and data storage.
- It is a flexible and highly-available architecture for large scale computation and data processing on a network of commodity hardware.

# Brief History of Hadoop

- Designed to answer the question:  
**“How to process big data with reasonable cost and time?”**





# Search engines in 1990s



## MetaCrawler Parallel Web Search Service

by [Erik Selberg](#) and [Oren Etzioni](#)

Try the new [MetaCrawler Beta!](#)  
If you're searching for a person's home page, try [Ahoy!](#)

[Examples](#) [Beta Site](#) [Add Site](#) [About](#)

Search for:   
☐ as a Phrase ☒ All of these words ☐ Any of these words

For better results, please specify:  
Search Region:  Search Sites:

Performance parameters:  
Max wait:  minutes Match type:

[About](#) [Help](#) [Problems](#) [Add Site](#) [Search](#)  
[webmaster@metacrawler.com](mailto:webmaster@metacrawler.com)  
© Copyright 1995, 1996 Erik Selberg and Oren Etzioni

1996

Serious Sports Fans Only \$1,000,000 in Cash and Prizes!  
For serious sports fans only! Play Fantasy Football!



It's amazing where  
Go Get It will get you.

Find:

[Enhance your search](#)



[New Search](#) [TopNews](#) [Sites by Subject](#) [Top 5% Sites](#) [City Guide](#) [Pictures & Sounds](#)  
[PeopleFind](#) [Point Review](#) [Road Maps](#) [Software](#) [About Lycos](#) [Club Lycos](#) [Help](#)

[Add Your Site to Lycos](#)

Copyright © 1996 Lycos™, Inc. All Rights Reserved.  
Lycos is a trademark of Carnegie Mellon University.  
[Questions & Comments](#)

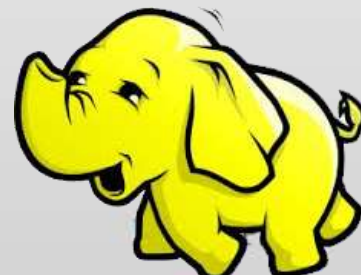
1996

The Excite search engine interface features a red header with the 'excite' logo. Navigation links include 'search', 'reviews', 'city.net', 'new lives', and 'reference?'. Below these are 'excite home', 'maps', 'news', and 'people finder'. The main search area has a 'What:' text input, a 'Where:' dropdown menu set to 'World Wide Web', and a 'search' button. A 'Help' link and an '[Add URL]' button are also present. A section titled 'Excite Search: twice the power of the competition.' is followed by a list of categories: Arts, Business, Computing, Education, Entertainment, Health, Hobbies, Life & Style, Money, News & Reference, Personal Pages, Politics & Law, Regional, Science, Shopping, and Sports. A sidebar on the left mentions 'Researching stocks?', 'Buying a car?', 'Planning a wedding?', and 'Check out Excite Seeing Tours.' with a link to 'Bill Mitchell: Notice that clicks!'.

1996

The Wired Search Center interface has a purple header with 'Wired' and 'The WIRED Search Center'. A search bar contains 'all the words' and a 'SEARCH' button. Below the search bar are filters for 'Date' (set to 'in the last week') and 'Geocode' (set to 'North America (.com)'). There are checkboxes for 'Include media files' (Image, Audio, Video, Compression) and 'Return Results' (set to '10 full descriptions'). A sidebar on the left lists 'Search: The Web' with links to Usenet, Free Public Sites, Classic Web, Online Movies, Stocks, Electronic Groups, SearchWare, Find, Resources, People, and Email Address. A sidebar on the right features 'Sandbox Entertainment', 'Shop WIRED Holiday Gift Guide', 'SOMETHING HAS SURVIVED.', and links to 'Find more deals', 'To Do', 'Cybernet Outpost', 'Microsoft Expedia Travel', and 'ON SALE'.

1997



# Google search engines



Search the web using Google!

Google Search

I'm feeling lucky

Special Searches  
[Stanford Search](#)  
[Linux Search](#)

Help!  
[About Google!](#)  
[Company Info](#)  
[Google! Legal](#)

Get Google!  
updates monthly

your e-mail

Subscribe

[Archive](#)

Copyright ©1998 Google Inc.

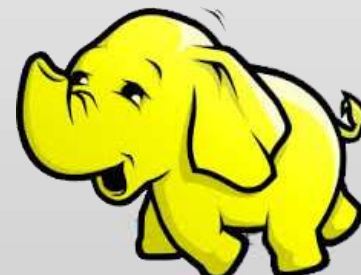
1998



Google Search

I'm Feeling Lucky

2013



# Hadoop's Developers



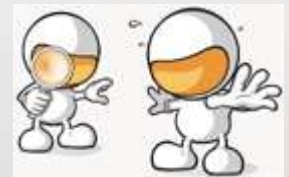
Doug Cutting



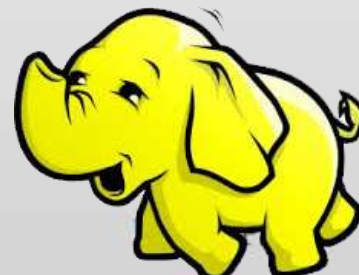
**2005:** Doug Cutting and Michael J. Cafarella developed Hadoop to support distribution for the [Nutch](#) search engine project.



The project was funded by Yahoo.



**2006:** Yahoo gave the project to Apache Software Foundation.





# Google Origins

2003

## The Google File System

Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung  
Google\*



2004

## MapReduce: Simplified Data Processing on Large Clusters

Jeffrey Dean and Sanjay Ghemawat

jeff@google.com, sanjay@google.com

Google, Inc.



2006

## Bigtable: A Distributed Storage System for Structured Data

Fay Chang, Jeffrey Dean, Sanjay Ghemawat, Wilson C. Hsieh, Deborah A. Wallach,  
Mike Burrows, Tushar Chandra, Andrew Fikes, Robert E. Gruber  
(fay.jeff.sanjay.wilson@cs.cmu.edu, tushar.chandra@cs.cmu.edu, robert@cs.cmu.edu)

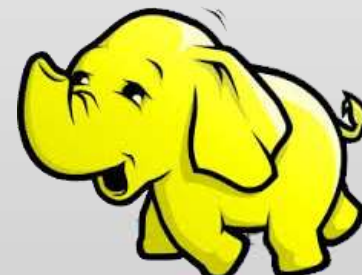
Google, Inc.

### Abstract

Bigtable is a distributed storage system for managing structured data that is designed to scale to a very large number of nodes. It is designed to store and manage data in a distributed manner. Many projects at Google store data in Bigtable, including web indexing, Google Earth, and Google Maps. These applications place very different demands on Bigtable, both in terms of data size (from URLs to

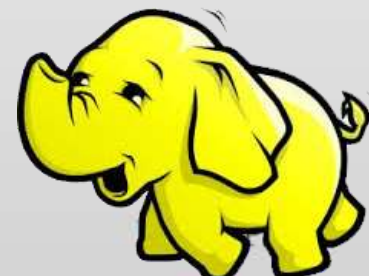
achieved scalability and high performance, but Bigtable provides a different interface than such systems. Bigtable does not support a full relational data model; instead, it provides clients with a simple data model that supports dynamic control over data layout and format, allows clients to reason about the locality properties of data represented in the underlying storage. Data is accessed using row and column names that can be arbitrary strings. Bigtable also treats data as uninterpreted bit

APACHE  
HBASE



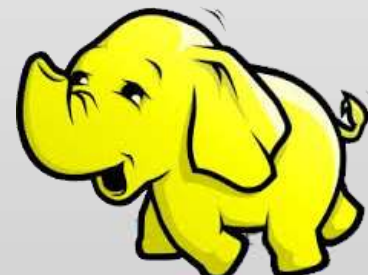
# Some Hadoop Milestones

- **2008 - Hadoop Wins Terabyte Sort Benchmark** (sorted 1 terabyte of data in 209 seconds, compared to previous record of 297 seconds)
- 2009 - Avro and Chukwa became new members of Hadoop Framework family
- 2010 - Hadoop's Hbase, Hive and Pig subprojects completed, adding more computational power to Hadoop framework
- **2011 - ZooKeeper Completed**
- **2013 - Hadoop 1.1.2 and Hadoop 2.0.3 alpha.**
  - Ambari, Cassandra, Mahout have been added

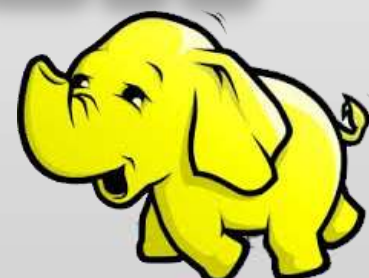
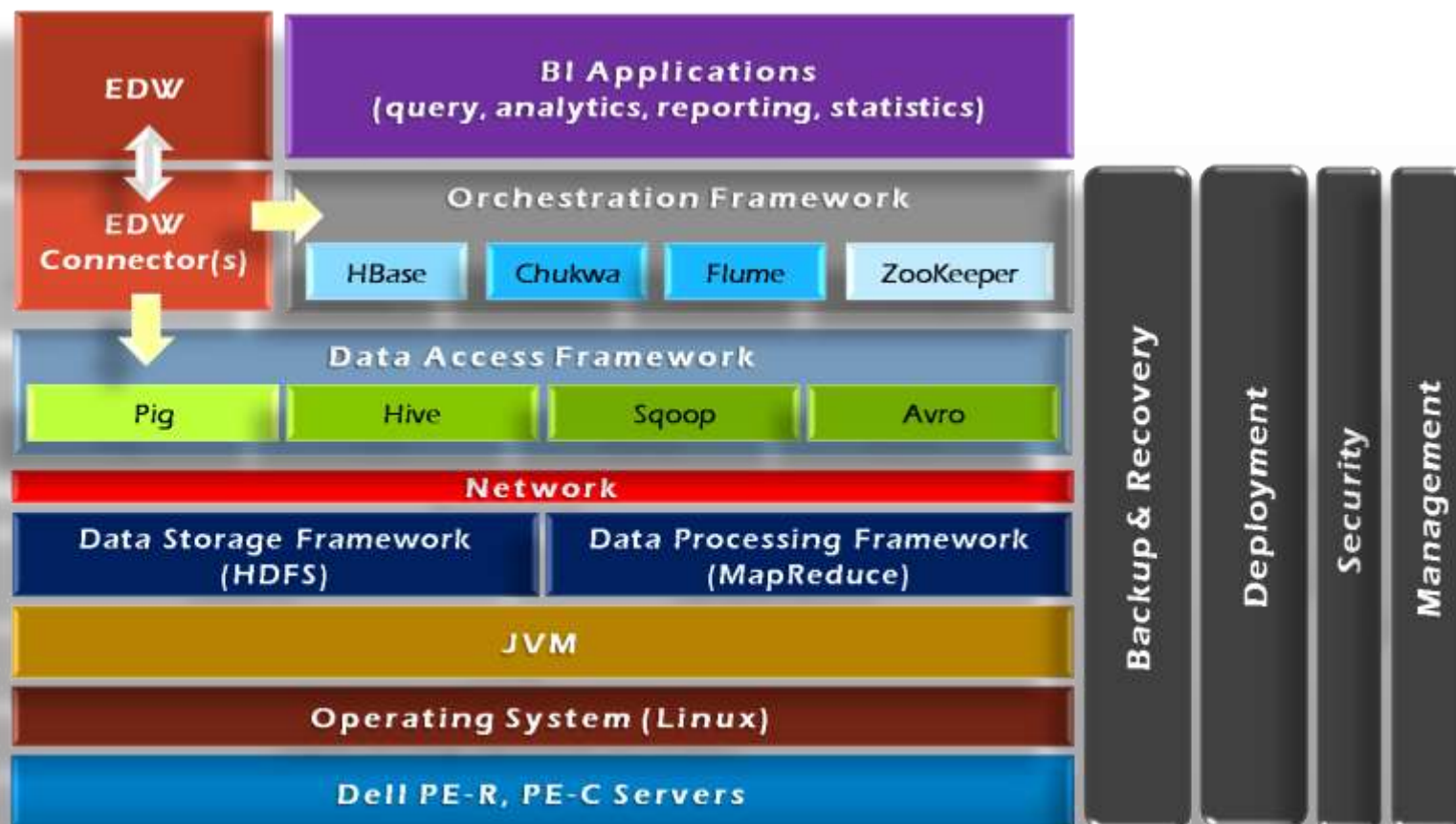


# What is Hadoop?

- **Hadoop:**
  - an open-source software framework that supports data-intensive distributed applications, licensed under the Apache v2 license.
- **Goals / Requirements:**
  - Abstract and facilitate the storage and processing of large and/or rapidly growing data sets
    - Structured and non-structured data
    - Simple programming models
  - High scalability and availability
  - Use commodity (cheap!) hardware with little redundancy
  - Fault-tolerance
  - Move computation rather than data

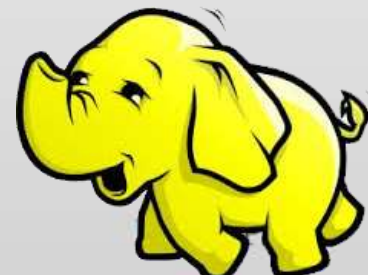


# Hadoop Framework Tools



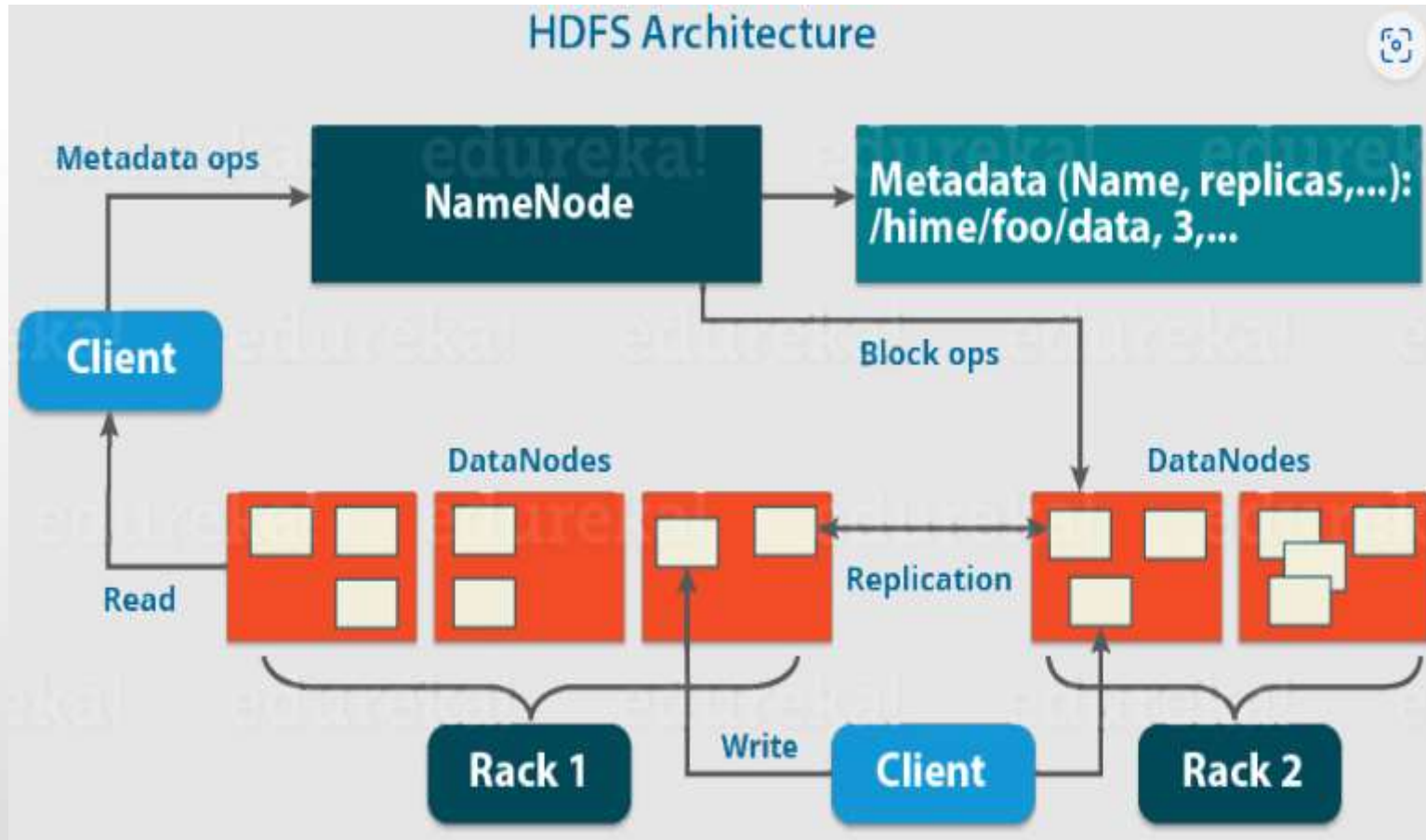
# Hadoop's Architecture

- Distributed, with some centralization
- Main nodes of cluster are where most of the computational power and storage of the system lies
- Main nodes run TaskTracker to accept and reply to MapReduce tasks, and also DataNode to store needed blocks closely as possible
- Central control node runs NameNode to keep track of HDFS directories & files, and JobTracker to dispatch compute tasks to TaskTracker
- Written in Java, also supports Python and Ruby

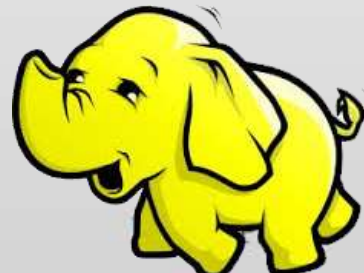
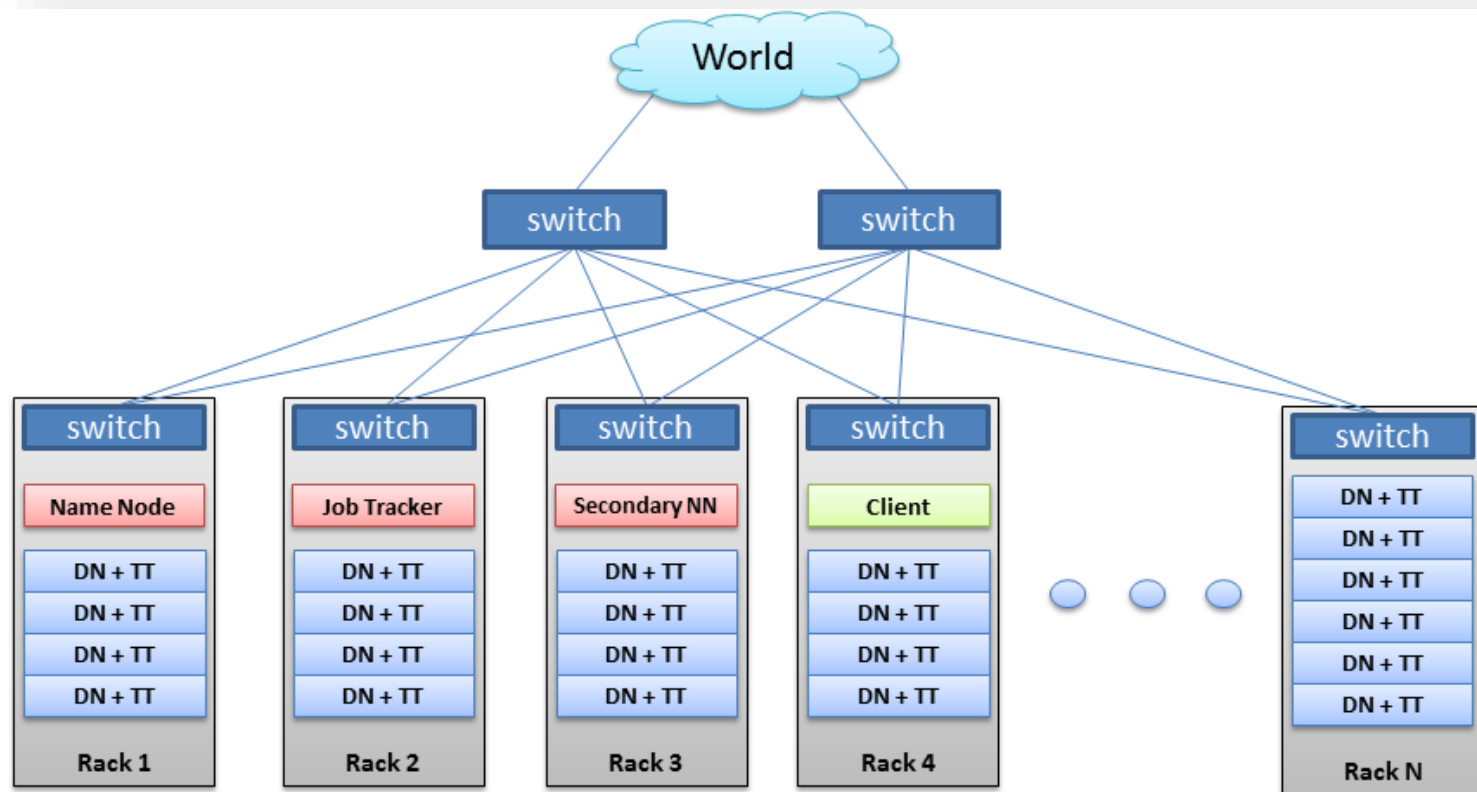




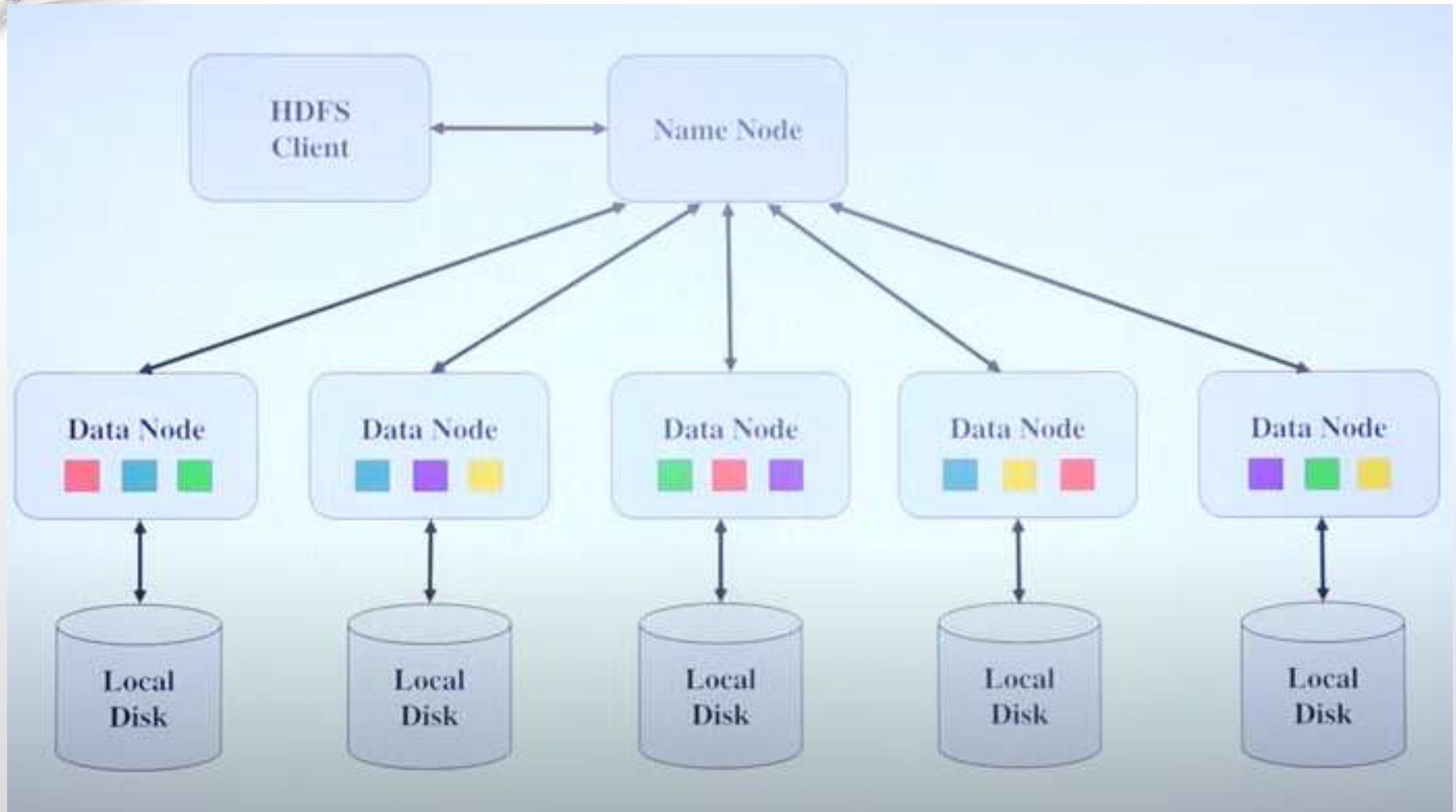
# Hadoop's Architecture



# Hadoop's Architecture

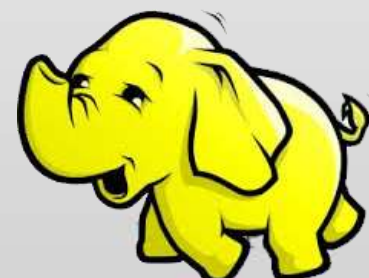


# HDFS



# Hadoop's Architecture

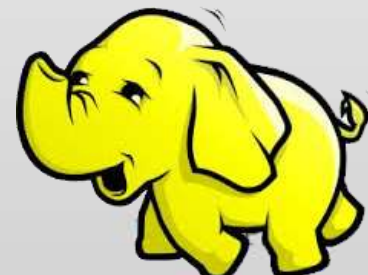
- Hadoop Distributed Filesystem
- Tailored to needs of MapReduce
- Targeted towards many reads of filestreams
- Writes are more costly
- High degree of data replication (3x by default)
- No need for RAID on normal nodes
- Large blocksize (64MB)
- Location awareness of DataNodes in network



# Hadoop's Architecture

## NameNode:

- Stores metadata for the files, like the directory structure of a typical FS.
- The server holding the NameNode instance is quite crucial, as there is only one.
- Transaction log for file deletes/adds, etc. Does not use transactions for whole blocks or file-streams, only metadata.
- Handles creation of more replica blocks when necessary after a DataNode failure

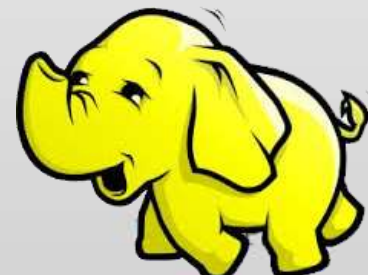




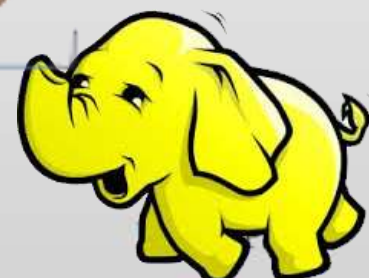
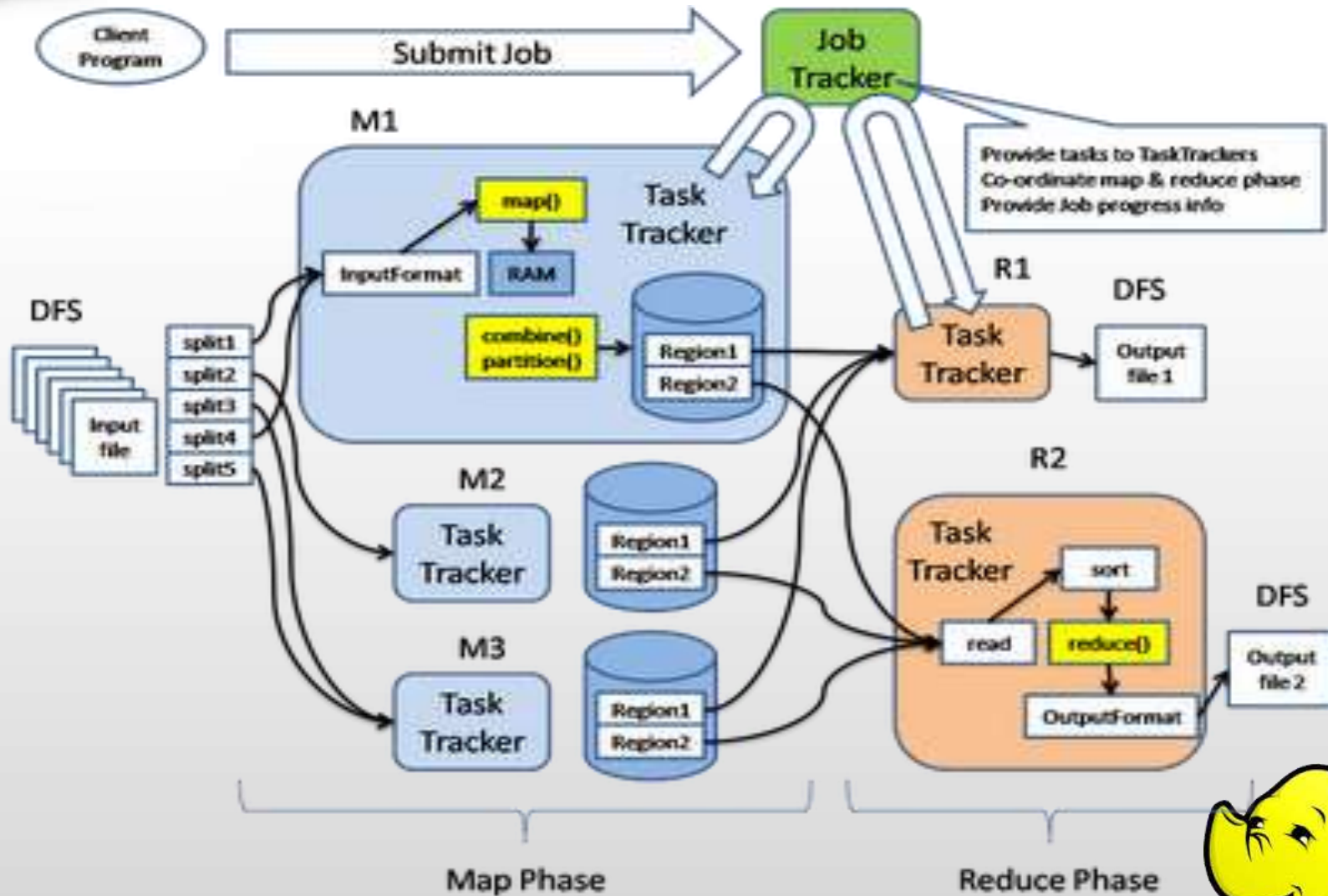
# Hadoop's Architecture

## DataNode:

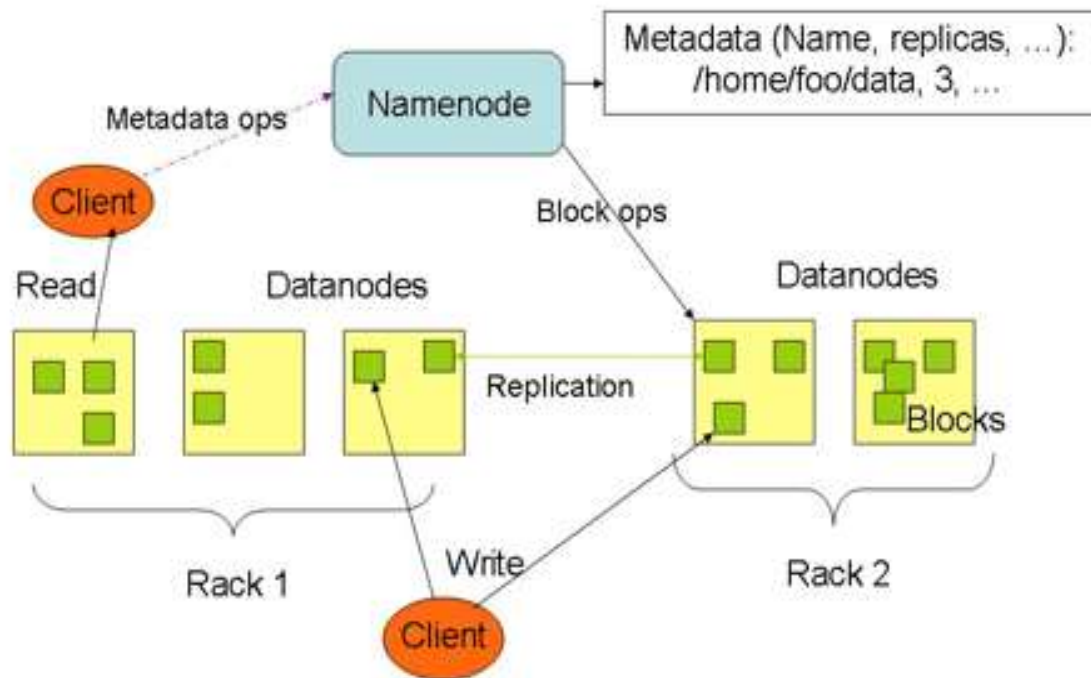
- Stores the actual data in HDFS
- Can run on any underlying filesystem (ext3/4, NTFS, etc)
- Notifies NameNode of what blocks it has
- NameNode replicates blocks 2x in local rack, 1x elsewhere



# Hadoop's Architecture: MapReduce Engine



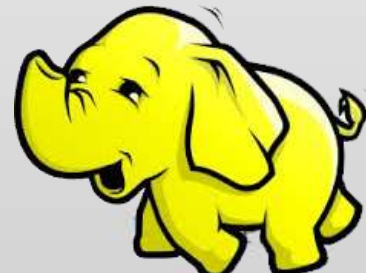
```
<?php  
/*  
 * [package WordPress  
 * [subpackage Default_Theme  
 */  
?  
<!DOCTYPE html PUBLIC "-//W3C  
-html 4.01 Transitional  
->  
<html profile="http://  
-meta http-equiv="C  
<title><?php  
<link rel=""  
<link rel=""  
<body>
```



# Hadoop's Architecture

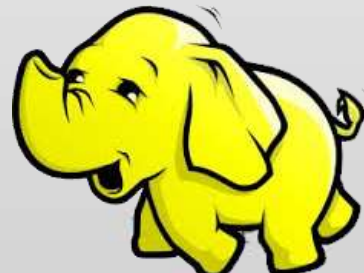
## MapReduce Engine:

- JobTracker & TaskTracker
- JobTracker splits up data into smaller tasks("Map") and sends it to the TaskTracker process in each node
- TaskTracker reports back to the JobTracker node and reports on job progress, sends data ("Reduce") or requests new jobs



# Hadoop's Architecture

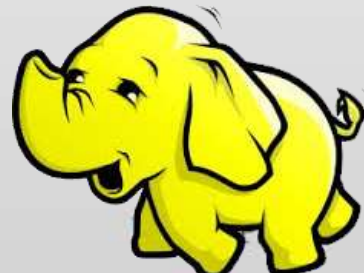
- None of these components are necessarily limited to using HDFS
- Many other distributed file-systems with quite different architectures work
- Many other software packages besides Hadoop's MapReduce platform make use of HDFS





# Hadoop in the Wild

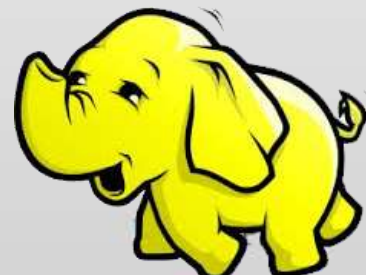
- Hadoop is in use at most organizations that handle big data:
  - Yahoo!
  - Facebook
  - Amazon
  - Netflix
  - Etc...
- Some examples of scale:
  - Yahoo!'s Search Webmap runs on 10,000 core Linux cluster and powers Yahoo! Web search
  - FB's Hadoop cluster hosts 100+ PB of data (July, 2012) & growing at ½ PB/day (Nov, 2012)



# Hadoop in the Wild

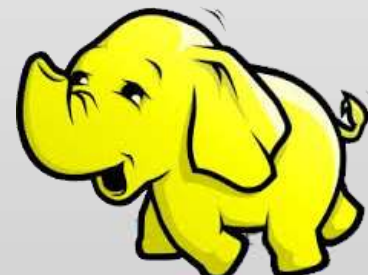
## Three main applications of Hadoop:

- Advertisement (Mining user behavior to generate recommendations)
- Searches (group related documents)
- Security (search for uncommon patterns)



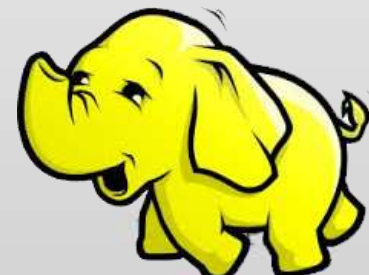
# Hadoop in the Wild

- Non-realtime large dataset computing:
  - NY Times was dynamically generating PDFs of articles from 1851-1922
  - Wanted to pre-generate & statically serve articles to improve performance
  - Using Hadoop + MapReduce running on EC2 / S3, converted 4TB of TIFFs into 11 million PDF articles in 24 hrs



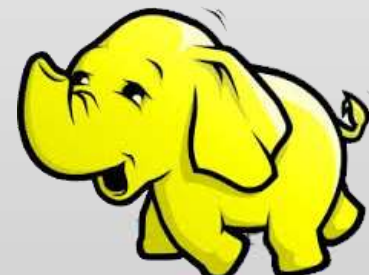
# Hadoop in the Wild: Facebook Messages

- Design requirements:
  - Integrate display of email, SMS and chat messages between pairs and groups of users
  - Strong control over who users receive messages from
  - Suited for production use between 500 million people immediately after launch
  - Stringent latency & uptime requirements



# Hadoop in the Wild

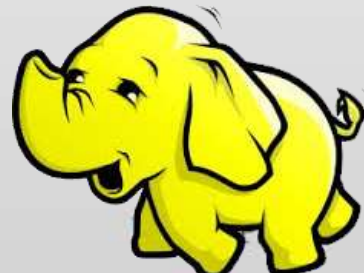
- System requirements
  - High write throughput
  - Cheap, elastic storage
  - Low latency
  - High consistency (within a single data center good enough)
  - Disk-efficient sequential and random read performance





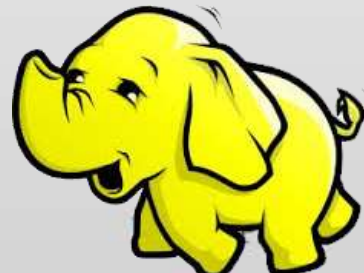
# Hadoop in the Wild

- Classic alternatives
  - These requirements typically met using large MySQL cluster & caching tiers using Memcached
  - Content on HDFS could be loaded into MySQL or Memcached if needed by web tier
- Problems with previous solutions
  - MySQL has low random write throughput... BIG problem for messaging!
  - Difficult to scale MySQL clusters rapidly while maintaining performance
  - MySQL clusters have high management overhead, require more expensive hardware



# Hadoop in the Wild

- Facebook's solution
  - Hadoop + HBase as foundations
  - Improve & adapt HDFS and HBase to scale to FB's workload and operational considerations
    - Major concern was availability: NameNode is SPOF & failover times are at least 20 minutes
    - Proprietary "AvatarNode": eliminates SPOF, makes HDFS safe to deploy even with 24/7 uptime requirement
    - Performance improvements for realtime workload: RPC timeout. Rather fail fast and try a different DataNode



```
<?php
/*
 * [package WordPress
 * [subpackage Default_Theme
 */
?>
<!DOCTYPE html PUBLIC "-//W3C
<html xmlns="http://www.w3.

<head profile="http://r
<meta http-equiv="Co

<title><?php echo

<link rel="
<link rel=

<?php
```

- Distributed File System
- Fault Tolerance
- Open Data Format
- Flexible Schema
- Queryable Database

```
<?php
<!-- [package WordPress]
<!-- [subpackage Default_Theme]
-->
<!DOCTYPE html PUBLIC "-//W3C//
<html xmlns="http://www.w3.
<head profile="http://
<meta http-equiv="Co
<title><?php
<link rel="
<link rel
</?php
-->
```

- Need to process Multi Petabyte Datasets
- Data may not have strict schema
- Expensive to build reliability in each application
- Nodes fails everyday
- Need common infrastructure
- Very Large Distributed File System
- Assumes Commodity Hardware
- Optimized for Batch Processing
- Runs on heterogeneous OS

```
<?php
/*
 * [package WordPress]
 * [subpackage Default_Theme]
 */
?>
<!DOCTYPE html PUBLIC "-//W3C//
<html xmlns="http://www.w3.

<head profile="http://r
<meta http-equiv="C

<title><?php echo

<link rel="
<link rel=

<body>
```

- A Block Server
  - Stores data in local file system
  - Stores meta-data of a block - checksum
  - Serves data and meta-data to clients
- Block Report
  - Periodically sends a report of all existing blocks to NameNode
- Facilitate Pipelining of Data
  - Forwards data to other specified DataNodes

```
<?php
/*
 * [package WordPress]
 * [subpackage Default_Theme]
 */
?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head profile="http://gmpg.org/xfn/11">
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title><?php echo get_bloginfo( 'name' );>
</title>
<link rel="stylesheet" type="text/css" href="http://www.wordpress.org/wp-content/themes/default/css/style.css" />
</head>
<body>
```

- Replication Strategy
  - One replica on local node
  - Second replica on a remote rack
  - Third replica on same remote rack
  - Additional replicas are randomly placed
- Clients read from nearest replica

```
<?php
/*
 * [package WordPress]
 * [subpackage Default_Theme]
 */
?>
<!DOCTYPE html PUBLIC "-//W3C//
<html xmlns="http://www.w3.

<head profile="http://e
<meta http-equiv="Con

<title><?php echo

<link rel="
<link rel=

<body>
```

- Use Checksums to validate data – CRC32
- File Creation
  - Client computes checksum per 512 byte
  - DataNode stores the checksum
- File Access
  - Client retrieves the data and checksum from DataNode
  - If validation fails, client tries other replicas



```
<?php  
/* [package WordPress  
/* [subpackage Default_Theme  
*/  
*/  
?>  
<!DOCTYPE html PUBLIC "-//W3C  
<html xmlns="http://www.w3.  
  
<head profile="http://  
<meta http-equiv="C  
  
<title><?php echo  
  
<link rel=""  
<link rel=""  
  
<body>
```

- Client retrieves a list of DataNodes on which to place replicas of a block
- Client writes block to the first DataNode
- The first DataNode forwards the data to the next DataNode in the Pipeline
- When all replicas are written, the client moves on to write the next block in file

```
<?php
/* [package WordPress
 * [subpackage Default_Theme
 */
?>
<!DOCTYPE html PUBLIC "-//W3C
<html xmlns="http://www.w3.

<head profile="http://r
<meta http-equiv="Co

<title><?php echo

<link rel="
<link rel=

<?php
```

- MapReduce programming model
  - Framework for distributed processing of large data sets
  - Pluggable user code runs in generic framework
- Common design pattern in data processing
  - `cat * | grep | sort | uniq -c | cat > file`
  - `input | map | shuffle | reduce | output`

```
<?php
/*
 * [package WordPress]
 * [subpackage Default_Theme]
 */
?>
<!DOCTYPE html PUBLIC "-//W3C//
<html xmlns="http://www.w3.

<head profile="http://r
<meta http-equiv="C

<title><?php echo

<link rel="
<link rel=

<script>
```

- Log processing
- Web search indexing
- Ad-hoc queries

```
<?php
/*
 * @package WordPress
 * @subpackage Default_Theme
 */
<?
<!DOCTYPE html PUBLIC "-//W3C
<html xmlns="http://www.w3.
<head profile="http://
<meta http-equiv="Co
<title><?php echo
<link rel="
<link rel
<script>
```

- MapReduce Component
  - JobClient
  - JobTracker
  - TaskTracker
  - Child
- Job Creation/Execution Process

```
<?php
/*
 * [package WordPress]
 * [subpackage Default_Theme]
 */
?>
<!DOCTYPE html PUBLIC "-//W3C//
<html xmlns="http://www.w3.

<head profile="http://r
<meta http-equiv="Co

<title><?php echo

<link rel="
<link rel=

<?php
```

- JobClient
  - Submit job
- JobTracker
  - Manage and schedule job, split job into tasks
- TaskTracker
  - Start and monitor the task execution
- Child
  - The process that really execute the task

```

<?php
/*
 * [package WordPress]
 * [subpackage Default_Theme]
 */
?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
  <head profile="http://www.w3.org/TR/xhtml1/profile/xhtml1-strict.dtd">
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
  <title><?php echo $page_title; ?></title>
  <link rel="stylesheet" type="text/css" href="style.css">
  <link rel="stylesheet" type="text/css" href="wp-content/themes/default/images/css/print.css" media="print">
  </head>
  <body>

```

- Protocol
  - JobClient  $\xleftrightarrow{\text{JobSubmissionProtocol}}$  JobTracker
  - TaskTracker  $\xleftrightarrow{\text{JobTrackerProtocol}}$  JobTracker
  - TaskTracker  $\xleftrightarrow{\text{TaskUmbilicalProtocol}}$  Child
- JobTracker implements both protocol and works as server in both IPC
- TaskTracker implements the TaskUmbilicalProtocol; Child gets task information and reports task status through it.

```
<?php
<!-- [package WordPress]
--> [namespace DefaultNamespace]
-->
?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head profile="http://gmpg.org/xfn/11">
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title><?php echo $bloginfo['name']></?php>
<link rel="stylesheet" href="http://www.wordpress.org/wp-content/themes/default/css/style.css" type="text/css">
</head>
<body>
```

- Check input and output, e.g. check if the output directory is already existing
  - `job.getInputFormat().validateInput(job);`
  - `job.getOutputFormat().checkOutputSpecs(fs, job);`
- Get InputSplits, sort, and write output to HDFS
  - `InputSplit[] splits = job.getInputFormat().getSplits(job, job.getNumMapTasks());`
  - `writeSplitsFile(splits, out); // out is $SYSTEMDIR/$JOBID/job.split`



```
<?php
<!-- [package WordPress]
<!-- [package DefaultTheme]
-->
<!DOCTYPE html PUBLIC "-//W3C//
<html xmlns="http://www.w3.
<head profile="http://e
<meta http-equiv="Co
<title><?php echo
<link rel="
<link rel="
<script>
```

- The jar file and configuration file will be uploaded to HDFS system directory
  - `job.write(out);` // out is `$SYSTEMDIR/$JOBID/job.xml`
- `JobStatus status =`  
`jobSubmitClient.submitJob(jobId);`
  - This is an RPC invocation, `jobSubmitClient` is a proxy created in the initialization

```
<?php
/* [package WordPress
 * [subpackage Default_Theme
 */
<?php
<!DOCTYPE html PUBLIC "-//W3C
<html xmlns="http://www.w3.
<head profile="http://
<meta http-equiv="Co
<title><?php echo
<link rel="
<link rel=
<script>
```

- `JobTracker.submitJob(jobID) <--` receive RPC invocation request
- `JobInProgress job = new JobInProgress(jobId, this, this.conf)`
- Add the job into Job Queue
  - `jobs.put(job.getProfile().getJobId(), job);`
  - `jobsByPriority.add(job);`
  - `jobInitQueue.add(job);`

- Sort by priority
  - resortPriority();
  - compare the JobPriority first, then compare the JobSubmissionTime
- Wake JobInitThread
  - jobInitQueue.notifyall();
  - job = jobInitQueue.remove(0);
  - job.initTasks();

```
<?php
/* [package WordPress
 * [subpackage Default_Theme
 *
 *
 */
<!DOCTYPE html PUBLIC "-//W3C
<html xmlns="http://www.w3.

<head profile="http://r
<meta http-equiv="C

<title><?php echo

<link rel="
<link rel=

<script>
```

- `JobInProgress(String jobid, JobTracker jobtracker, JobConf default_conf);`
- `JobInProgress.initTasks()`
  - `DataInputStream splitFile = fs.open(new Path(conf.get("mapred.job.split.file")));`  
`// mapred.job.split.file -->`  
`$SYSTEMDIR/$JOBID/job.split`

```
<?php
<!-- [package WordPress]
--> [package DefaultTheme]
-->
<?php
<!--DOCTYPE html PUBLIC "-//W3C
-->html xmlns="http://www.w3.
-->
<head profile="http://
-->meta http-equiv="C
-->
<title><?php
-->
<link rel="
-->
<link rel="
-->
<?php
-->
```

- `splits = JobClient.readSplitFile(splitFile);`
- `numMapTasks = splits.length;`
- `maps[i] = new TaskInProgress(jobId, jobFile, splits[i], jobtracker, conf, this, i);`
- `reduces[i] = new TaskInProgress(jobId, jobFile, splits[i], jobtracker, conf, this, i);`
- `JobStatus --> JobStatus.RUNNING`

- Task `getNewTaskForTaskTracker(String taskTracker)`
- Compute the maximum tasks that can be running on taskTracker
  - `int maxCurrentMapTasks = tts.getMaxMapTasks();`
  - `int maxMapLoad = Math.min(maxCurrentMapTasks, (int) Math.ceil(double) remainingMapLoad/numTaskTrackers);`

- `int numMaps = tts.countMapTasks(); // running tasks number`
- If `numMaps < maxMapLoad`, then more tasks can be allocated, then based on priority, pick the first job from the `jobsByPriority Queue`, create a task, and return to `TaskTracker`
  - `Task t = job.obtainNewMapTask(tts, numTaskTrackers);`



```
<?php
/*
 * [package WordPress]
 * [subpackage Default_Theme]
 */
?>
<!DOCTYPE html PUBLIC "-//W3C//
<html xmlns="http://www.w3.

<head profile="http://
<meta http-equiv="C

<title><?php echo

<link rel="
<link rel=

<?php
```

- initialize()
  - Remove original local directory
  - RPC initialization
    - TaskReportServer = RPC.getServer(this, bindAddress, tmpPort, max, false, this, fConf);
    - InterTrackerProtocol jobClient = (InterTrackerProtocol) RPC.waitForProxy(InterTrackerProtocol.class, InterTrackerProtocol.versionID, jobTrackAddr, this.fConf);

```
<?php
/*
 * [package WordPress]
 * [subpackage Default_Theme]
 */
?>
<!DOCTYPE html PUBLIC "-//W3C//
<html xmlns="http://www.w3.

<head profile="http://r
<meta http-equiv="Co

<title><?php echo

<link rel="
<link rel=

<script>
```

- `run();`
- `offerService();`
- TaskTracker talks to JobTracker with HeartBeat message periodically
  - `HeartbeatResponse heartbeatResponse = transmitHeartBeat();`

```
<?php
/* [package WordPress
 * [subpackage Default_Theme
 *
 */
?>
<!DOCTYPE html PUBLIC "-//W3C
<html xmlns="http://www.w3.

<head profile="http://r
<meta http-equiv="Co

<title><?php echo

<link rel="
<link rel=

<body>
```

- TaskTracker.localizeJob(TaskInProgress tip);
- launchTasksForJob(tip, new JobConf(rjob.jobFile));
  - tip.launchTask(); // TaskTracker.TaskInProgress
  - tip.localizeTask(task); // create folder, symbol link
  - runner = task.createRunner(TaskTracker.this);
  - runner.start(); // start TaskRunner thread

```
<?php
/*
 * [package WordPress
 * [subpackage Default_Theme
 */
?>
<!DOCTYPE html PUBLIC "-//W3C
<html xmlns="http://www.w3.

<head profile="http://
<meta http-equiv="Co

<title><?php echo

<link rel="
<link rel=

<?php
```

- TaskRunner.run();
  - Configure child process' jvm parameters, i.e. classpath, taskid, taskReportServer's address & port
  - Start Child Process
    - runChild(wrappedCommand, workDir, taskid);

```
<?php
/*
 * [package WordPress]
 * [subpackage Default_Theme]
 */
?>
<!DOCTYPE html PUBLIC "-//W3C//
<html xmlns="http://www.w3.

<head profile="http://r
<meta http-equiv="Con

<title><?php echo

<link rel="
<link rel=

<body>
```

- Create RPC Proxy, and execute RPC invocation
  - TaskUmbilicalProtocol umbilical =  
(TaskUmbilicalProtocol)  
RPC.getProxy(TaskUmbilicalProtocol.class,  
TaskUmbilicalProtocol.versionID, address,  
defaultConf);
  - Task task = umbilical.getTask(taskid);
- task.run(); // mapTask / reduceTask.run

```
<?php
/*
 * [package WordPress]
 * [subpackage Default_Theme]
 */
?>
<!DOCTYPE html PUBLIC "-//W3C//
<html xmlns="http://www.w3.

<head profile="http://r
<meta http-equiv="C

<title><?php echo

<link rel="
<link rel=

<body>
```

- Child
  - task.done(umbilical);
    - RPC call: umbilical.done(taskId, shouldBePromoted)
- TaskTracker
  - done(taskId, shouldPromote)
    - TaskInProgress tip = tasks.get(taskid);
    - tip.reportDone(shouldPromote);
      - taskStatus.setRunState(TaskStatus.State.SUCCEEDED)

```
<?php
/*
 * [package WordPress
 * [subpackage Default_Theme
 */
?>
<!DOCTYPE html PUBLIC "-//W3C
<html xmlns="http://www.w3.

<head profile="http://
<meta http-equiv="Co

<title><?php echo

<link rel="
<link rel=

<body>
```

- JobTracker

- TaskStatus report: `status.getTaskReports();`
- TaskInProgress tip = `taskIdToTIPMap.get(taskId);`
- JobInProgress update JobStatus
  - `tip.getJob().updateTaskStatus(tip, report, myMetrics);`
    - One task of current job is finished
    - `completedTask(tip, taskStatus, metrics);`
    - `If (this.status.getRunState() == JobStatus.RUNNING && allDone) {this.status.setRunState(JobStatus.SUCCEEDED)}`



```
<?php
/*
 * [package WordPress
 * [subpackage Default_Theme
 */
?>
<!DOCTYPE html PUBLIC "-//W3C
<html xmlns="http://www.w3.

<head profile="http://r
<meta http-equiv="Co

<title><?php echo

<link rel="
<link rel=

<?php
```

- Word Count
  - `hadoop jar hadoop-0.20.2-examples.jar wordcount <input dir> <output dir>`
- Hive
  - `hive -f pagerank.hive`