Contents

[Hadoop  Ecosystem 1](#_Toc457312779)

[1. Storage 2](#_Toc457312780)

[**﻿**HDFS 2](#_Toc457312781)

[HBase 2](#_Toc457312782)

[HCatalog 3](#_Toc457312783)

[2. Processing 3](#_Toc457312784)

[MapReduce 3](#_Toc457312785)

[Pig 4](#_Toc457312786)

[3. Querying 4](#_Toc457312787)

[HIVE 5](#_Toc457312788)

[4. External integration 5](#_Toc457312789)

[Flume 5](#_Toc457312790)

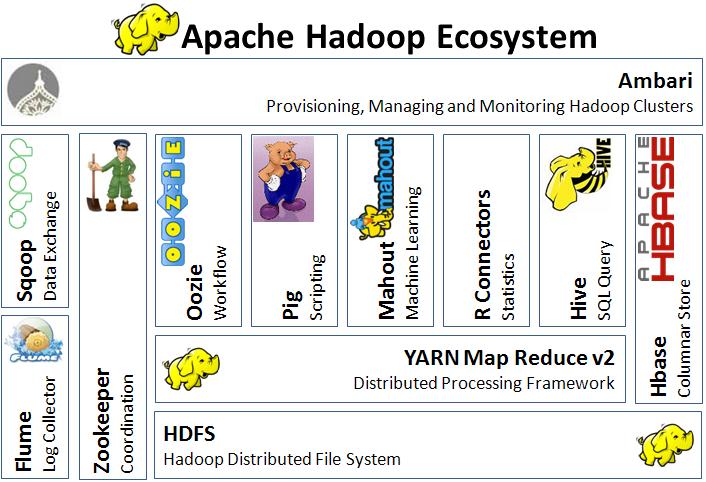
[Other 6](#_Toc457312791)

[Cassandra 6](#_Toc457312792)

[Oozie: Workflow 6](#_Toc457312793)

[Sqoop: Data Exchange 6](#_Toc457312794)

[Hadoop  Ecosystem](http://thebigdatablog.weebly.com/blog/the-hadoop-ecosystem-overview)



Before you can traverse through the Hadoop environment it is important to identify and learn about the key players. In this post I will provide an overview of the applications, tools and interfaces currently available in the Hadoop ecosystem. I will categorize each product by its functionality (Storage, Processing, Querying, External Integration & Coordination) and provide a description along with architecture, uses and resource likes. It is important to note that in a few months or yea**rs these products might become obsolete and replaced by another.**

## 1. Storage

### **﻿**HDFS

The primary distributed file system used by Hadoop applications which runs on large clusters of commodity machines. HDFS clusters consist of a NameNode that manages the file system metadata and DataNodes that store the actual data.

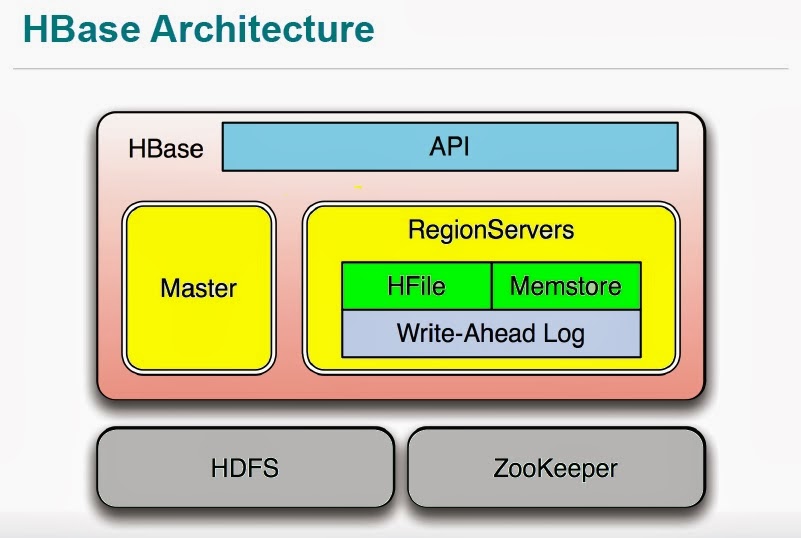
**Architecture:** [HDFS Architecture](http://thebigdatablog.weebly.com/uploads/3/2/3/2/32326475/hdfs_architecture.gif)



**Uses**:  
-Storage of large imported files from applications outside of the Hadoop ecosystem  
-Staging of imported files to be processed by Hadoop applications  
  
**Resource:**  
<http://hadoop.apache.org/docs/r2.3.0/hadoop-project-dist/hadoop-hdfs/HdfsUserGuide.html>

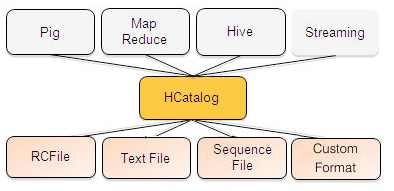
### HBase

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).  
  
**Architecture:** [HBase Architecture](http://thebigdatablog.weebly.com/uploads/3/2/3/2/32326475/hbase_architecture.jpg)  
**Uses:**  
-Storage of large data volumes (billions of rows) atop clusters of commodity hardware  
-Bulk storage of logs, documents, real-time activity feeds and raw imported data  
-Consistent performance of reads/writes to data used by Hadoop applications  
-Data Store than can be aggregated or processed using MapReduce functionality  
-Data platform for Analytics and Machine Learning  
  
**Resource:** <http://hbase.apache.org/book/architecture.html>



### HCatalog

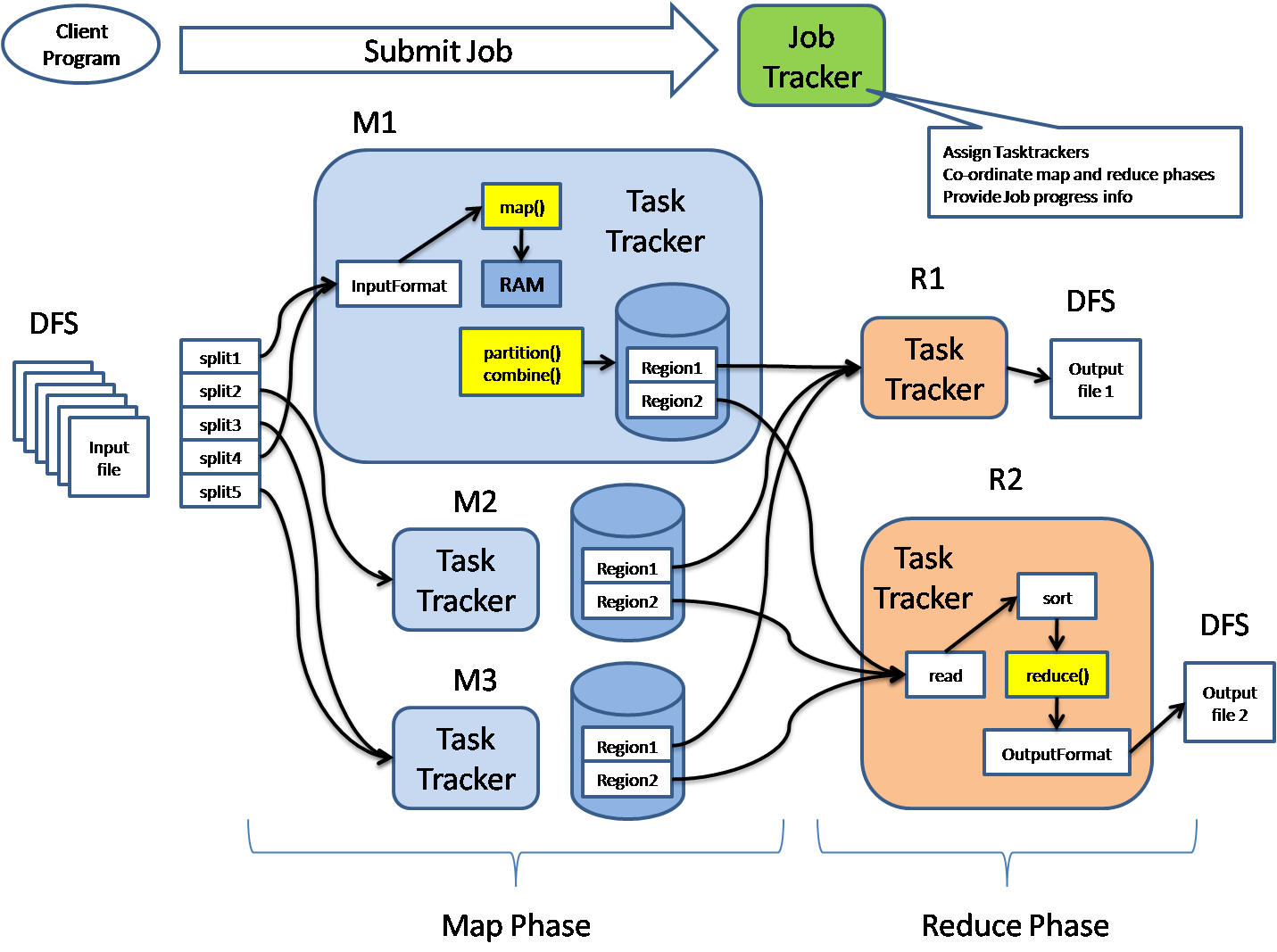
A table and storage management layer for Hadoop that enables Hadoop applications (Pig, MapReduce, and Hive) to read and write data to a tabular form as opposed to files. It also provides REST APIs so that external systems can access these tables’ metadata.  
  
**Architecture:** [HCatalog Architecture](http://thebigdatablog.weebly.com/uploads/3/2/3/2/32326475/hcatalog_architecture.jpg)  
  
**Uses**:  
-Centralized location of storage for data used by Hadoop applications  
-Reusable data store for sequenced and iterated Hadoop processes (ex: ETL)  
-Storage of data in a relational abstraction  
  
**Resource:** <https://cwiki.apache.org/confluence/display/Hive/HCatalog>



## 2. Processing

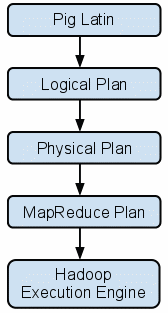
### MapReduce

A distributed data processing model and execution environment that runs on large clusters of commodity machines. It uses the MapReduce algorithm which breaks down all operations into Map or Reduce functions.  
  
**Architecture:** [MapReduce Architecture](http://thebigdatablog.weebly.com/uploads/3/2/3/2/32326475/mapreduce_architecture.png)  
  
**Uses:**  
-Aggregation (Counting, Sorting, Filtering, Stitching) on large and desperate data sets  
-Scalable parallelism of Map or Reduce tasks  
-Distributed task execution  
-Machine learning  
  
**Resource:** <http://hadoop.apache.org/docs/r1.0.4/mapred_tutorial.html#Purpose>



### Pig

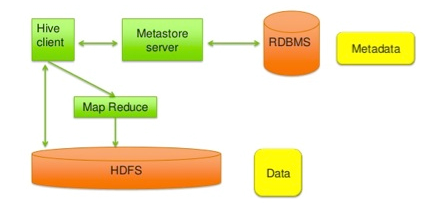
A scripting SQL based language and execution environment for creating complex MapReduce transformations. Functions are written in Pig Latin (the language) and translated into executable MapReduce jobs. Pig also allows the user to create extended functions (UDFs) using Java.  
  
**Architecture:** [Pig Architecture](http://thebigdatablog.weebly.com/uploads/3/2/3/2/32326475/pig_architecture.gif)  
  
**Uses:**  
-Scripting environment to execute ETL tasks/procedures on raw data in HDFS  
-SQL based language for creating and running complex MapReduce functions  
-Data processing, stitching, schematizing on large and desperate data sets  
  
**Resource:** [**http://pig.apache.org/docs/r0.12.1/index.html**](http://pig.apache.org/docs/r0.12.1/index.html)



## 3. Querying

### HIVE

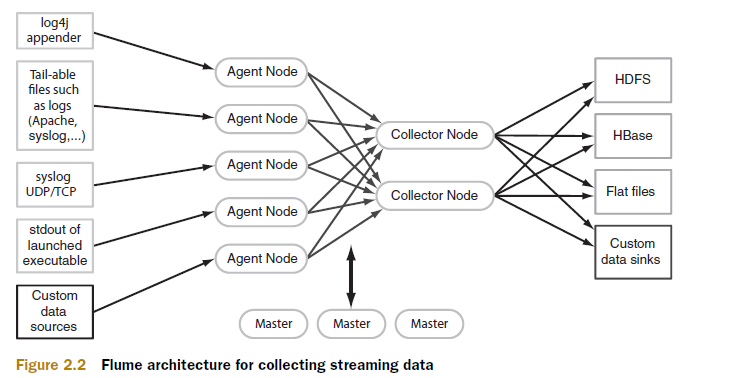
A distributed data warehouse built on top of HDFS to manage and organize large amounts of data. Hive provides a query language based on SQL semantics (HiveQL) which is translated by the runtime engine to MapReduce jobs for querying the data.  
  
 **Architecture:** [Hive Architecture](http://thebigdatablog.weebly.com/uploads/3/2/3/2/32326475/hive_architecture_2.png)  
  
**Uses:**  
-Schematized data store for housing large amounts of raw data  
-SQL-like Environment to execute analysis and querying tasks on raw data in HDFS  
-Integration with outside RDBMS applications   
  
 **Resource:** <https://cwiki.apache.org/confluence/display/Hive/Home#Home-ApacheHive>



## 4. External integration

### Flume

A distributed, reliable, and available service for efficiently collecting, aggregating, and moving large amounts of log data into HDFS. Flume's transports large quantities of event data using a steaming data flow architecture that is fault tolerant and failover recovery ready.  
  
**Architecture:** [Flume Architecture](http://thebigdatablog.weebly.com/uploads/3/2/3/2/32326475/flume_architecture_2.png)  
  
**Uses:**  
-Transportation of large amounts of event data (network traffic, logs, email messages)  
-Stream data from multiple sources into HDFS  
-Guaranteed and reliable real-time data streaming to Hadoop applications  
  
**Resource:** <http://flume.apache.org/FlumeUserGuide.html>



## Other

### apache Ambari

Apache Ambari is an open source software framework for provisioning, managing, and monitoring Hadoop clusters. The following are few highlights of this project:

* Ambari is useful for installing Hadoop services across different nodes of the cluster and handling the configuration of Hadoop Services on the cluster.
* Ambari offers centralized management of the cluster including configuration and re-configuration of services, starting and stopping of cluster and a lot more.
* Ambari offers a dashboard for monitoring the overall health of the cluster.
* Ambari offers alerting and email mechanism to get the required attention when required.
* Ambari offers [REST APIs](https://github.com/apache/ambari/blob/trunk/ambari-server/docs/api/v1/index.md) to developers for application integration.

### Cassandra

A Distributed Non-SQL Big Data Database

The Apache Cassandra database is the right choice when you need scalability and high availability without compromising performance. [Linear scalability](http://techblog.netflix.com/2011/11/benchmarking-cassandra-scalability-on.html) and proven fault-tolerance on commodity hardware or cloud infrastructure make it the perfect platform for mission-critical data. Cassandra's support for replicating across multiple datacenters is best-in-class, providing lower latency for your users and the peace of mind of knowing that you can survive regional outages.

Cassandra's data model offers the convenience of [column indexes](http://www.datastax.com/documentation/cql/3.1/cql/ddl/ddl_primary_index_c.html) with the performance of log-structured updates, strong support for [denormalization](http://www.datastax.com/documentation/cql/3.1/cql/ddl/ddl_intro_c.html) and [materialized views](http://www.datastax.com/dev/blog/new-in-cassandra-3-0-materialized-views), and powerful built-in caching.

### Oozie: Workflow

Apache Oozie is a job workflow scheduling and coordination manager for managing the jobs executed on Hadoop. The following are a few highlights of this project:

* Oozie can include both MapReduce as well as Non-MapReduce jobs.
* Oozie is integrated with Hadoop and is an integral part of the Hadoop Ecosystem.
* Oozie supports various jobs out of the box including MapReduce, Pig, Hive, Sqoop, etc.
* Oozie jobs are scheduled/recurring jobs and are executed based on scheduled frequency and availability of data.
* Oozie jobs are organized/arranged in a [Directed Acyclic Graph (DAG)](http://en.wikipedia.org/wiki/Directed_acyclic_graph) fashion.

### Sqoop: Data Exchange

Apache Sqoop is a tool designed for efficiently transferring the data between Hadoop and Relational Databases (RDBMS). The following are a few highlights of this project:

* Sqoop can efficiently transfer bulk data between HDFS and Relational Databases.
* Sqoop allows importing the data into HDFS in an incremental fashion.
* Sqoop can import and export data to and from HDFS, Hive, Relational Databases and Data Warehouses.
* Sqoop uses MapReduce to import and export of data thereby effectively utilizing the parallelism and fault tolerance features of Hadoop.
* Sqoop offers a command line commonly referred to as Sqoop command line.