

Julian Kunkel

The Apache Ecosystem and Beyond



Outline

- 1 Hadoop Ecosystem
- 2 User/Admin Interfaces
- 3 Workflows
- 4 SQL Tools
- 5 Other BigData Tools
- 6 Machine Learning
- 7 Summary

Hortonworks (2019 bought by Cloudera)

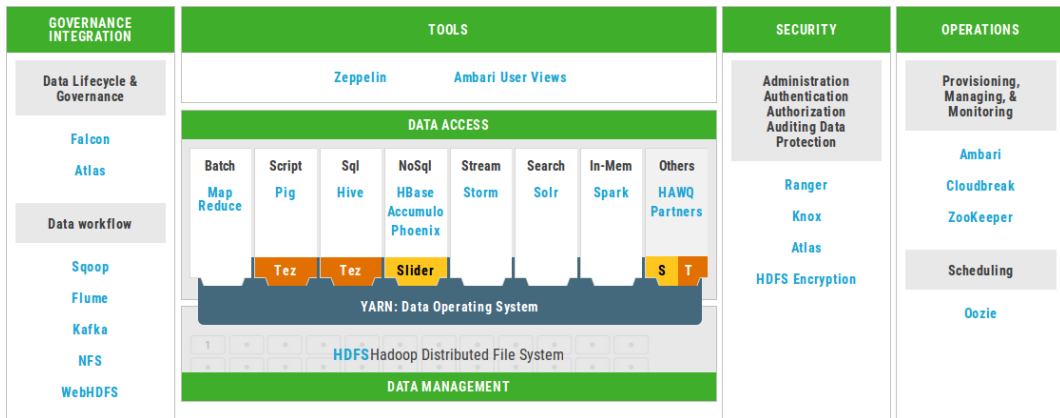


Figure: Screenshot from [40]

- Additionally: Hortonworks offers support, service
- Build with open-source

Cloudera Enterprise Data Hub [25]

- Cloudera offers support, services and tools around Hadoop
- Unified architecture: common infrastructure and data pool for tools
- Build with open-source tools, some own tools for management
- Cloudera has various other products (e.g., cloud-ready available)

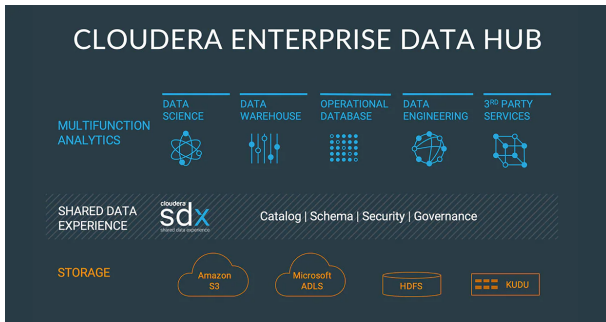


Figure: Source: [25]

Supporting Tools¹

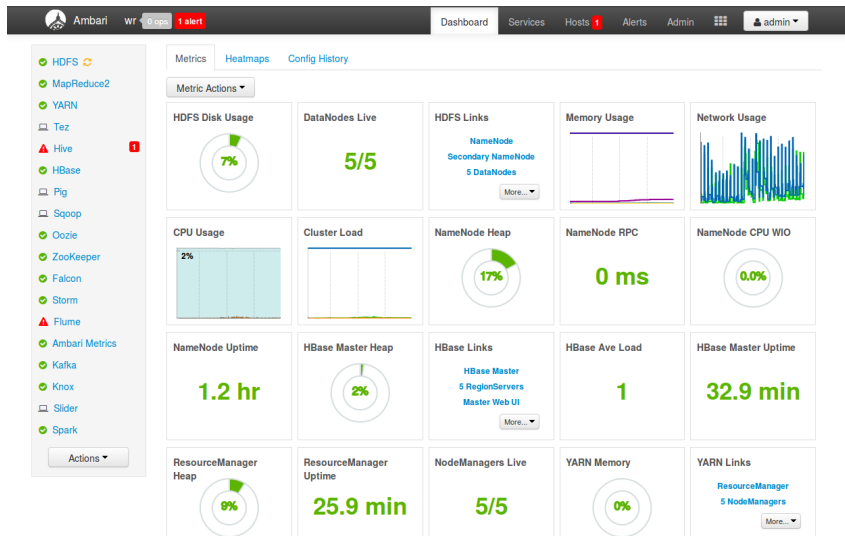
- Ambari: A Tool for Managing Hadoop Clusters
- Hue: Manage „BigData“ projects in a browser
- ZooKeeper: coordination/configuration service for services
- Oozie: Workflow scheduler (schedules/triggers workflows)
- Falcon: Data governance engine for data pipelines
- Flume: collecting, aggregating and moving large streaming event data
- Kafka: publish-subscribe distributed messaging system
- Knox: REST API gateway (for all services)
- Ranger: Integrate ACL permissions into Hadoop (ecosystem)
- Slider: YARN application supporting monitoring and dynamic scaling of non-YARN apps

¹ <https://hadoop.apache.org/>

Ambari: A Tool for Managing Hadoop Clusters

- Convenient tool managing 10+ Apache tools
- Supports installation and management
 - ▶ Dealing with data dependencies
 - ▶ Service startup
 - ▶ Monitoring of health and performance
 - ▶ (Re)configuration of services
 - ▶ Web UI with REST API

Management with Ambari: Dashboard



Management with Ambari: Configuration

[Summary](#) [Configs](#) [Quick Links](#) [Service Actions](#)

[Restart Required: 1 Component on 1 Host](#) [Restart](#)

Group

HDFS Default (5)

Manage Config Groups

Filter...

< >

V2

admin

2 months ago

Current

V1

admin

2 months ago

⌵

V2

Current

admin authored on Tue, Jul 07, 2015 19:05

Discard

Save

▼

NameNode

NameNode hosts

abu1.cluster

NameNode directories

/tmp/hadoop/hdfs/namenode,/bigdata/hdfs/namenode

🔒

NameNode Java heap size

95744

MB

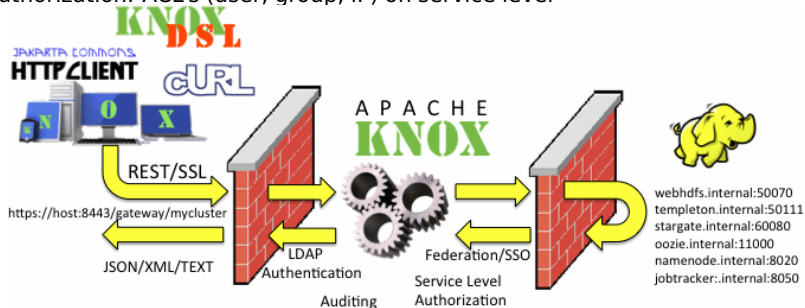
NameNode new generation size

23936

MB

Knox: Security for Hadoop [22]

- REST API Gateway for Hadoop ecosystem services
 - ▶ Supports: HDFS, Hcatalog, HBase, Oozie, Hive, Yarn, Storm
 - ▶ Supports multiple clusters
- Provides authentication, federation/SSO, authorization, auditing
- Enhances security providing central control and protection
 - ▶ SSL encryption
 - ▶ Authentication: LDAP, Active Directory, Kerberos
 - ▶ Authorization: ACL's (user, group, IP) on service level²



Example Accesses via the API [22]

List a HDFS directory

```
1 curl -i -k -u guest:guest-password -X GET  
   ↪ 'https://localhost:8443/gateway/sandbox/webhdfs/v1/?op=LISTSTATUS'
```

Example response

```
1 HTTP/1.1 200 OK  
2 Content-Type: application/json  
3 Content-Length: 450  
4 Server: Jetty(6.1.26)  
5  
6 {"FileStatuses":{"FileStatus": [  
7   {"accessTime":0,"blockSize":0,"group":"hdfs","length":0,"modificationTime":1350595859762,  
   ↪ "owner":"hdfs","pathSuffix":"apps","permission":"755","replication":0,"type":"DIRECTORY"},  
8   {"accessTime":0,"blockSize":0,"group":"mapred","length":0,"modificationTime":1350595874024,  
   ↪ "owner":"mapred","pathSuffix":"mapred","permission":"755",  
   ↪ "replication":0,"type":"DIRECTORY"},  
9 ]}}
```

Hue [12]: Lightweight Web Server for Hadoop

- Manage BigData projects in a browser
- Supports: Hadoop ecosystem
 - ▶ HDFS, Pig, Hive, MapReduce, Spark, Flink, ...

Features

- Data upload/download
- Management of HCatalog tables
- Query editor (Hive, Pig, Impala)
- Starting and monitoring of jobs

Hue: Lightweight Web Server for Hadoop

HUE

Query Editors ▾ Data Browsers ▾ Workflows ▾ Suche ▾ Security ▾ Datei-Browser Job-Browser 51emfu ▾ ?

Oozie-Dashboard Workflows **Coordinators** Bundles SLA Oozie

Nach Benutzernamen, Namen usw. suchen **Fortsetzen** **Unterbrechen** **Beenden** Nur Folgende anzeigen 1 7 15 30 Tage mit Status **Erfolgreich** **Aktiv** **Beendet**

Aktiv

☐ Nächste Übermittlung ☐ Status ☐ Name ☐ Fortschritt ☐ Sender ☐ Häufigkeit ☐ Startzeit ☐ ID

Keine Daten verfügbar

0 bis 0 von 0 Einträgen werden angezeigt ← Zurück Weiter →

Abgeschlossen

Fertigtstellung	Status	Name	Dauer	Sender	Häufigkeit	Startzeit	ID
Wed, 30 Sep 2015 22:41:00	KILLED	My_Coordinator	93d:14h:56m:0s	dtlqo9j	* / 1 * * * *	Mon, 29 Jun 2015 07:45:00	0000304-150621143055208-oozie-oozi-C
Mon, 07 Sep 2015 17:05:00	KILLED	My_Coordinator	7d:0h:0m:0s	a309ve7	30 1 * * *	Mon, 31 Aug 2015 17:05:00	0000094-150828163545629-oozie-oozi-C
Tue, 25 Aug 2015 13:15:00	KILLED	My_Coordinator	7d:0h:0m:0s	4k0susv	17 0 * * *	Tue, 18 Aug 2015 13:15:00	0000507-150730175918991-oozie-oozi-C
Tue, 25 Aug 2015 13:13:00	SUCCEEDED	My_Coordinator	7d:0h:0m:0s	9jalpv9	1 0,6 * * *	Tue, 18 Aug 2015 13:13:00	0000504-150730175918991-oozie-oozi-C

Hue: Lightweight Web Server for Hadoop

The screenshot displays the Hue web interface's 'Data Browser' view. At the top, there's a navigation bar with the Hue logo and links to 'Datei-Browser', 'Job-Browser', and a user profile '51emffu'. Below this, a sub-navigation bar shows 'Data Browser' as the active section, with other options like 'Data Browsers', 'Workflows', 'Suche', and 'Security'. The main interface features a search bar 'Nach Dateinamen suchen', an 'Aktionen' dropdown, and a button 'In Papierkorb verschieben'. The breadcrumb path is 'Startseite / user / 51emffu'. To the right, there are buttons for 'Verlauf' and 'Papierkorb'. The central part of the interface is a table listing files and directories. The table has columns for 'Name', 'Größe', 'Benutzer', 'Gruppe', 'Berechtigungen', and 'Datum'. The listed items are a directory '.', a directory '.staging', and a directory 'oozie-oozi'. At the bottom, there's a pagination bar showing 'Anzeigen 45 von 2 Elemente' and a page navigation section with 'Seite 1 of 1' and navigation arrows. A vertical 'Feedback' button is located on the right edge of the interface.

☐	Name	Größe	Benutzer	Gruppe	Berechtigungen	Datum
☐	↓		hdfs	supergroup	drwxr-xr-x	September 17, 2015 02:46 AM
☐	.		51emffu	51emffu	drwxr-xr-x	September 17, 2015 02:39 AM
☐	.staging		51emffu	51emffu	drwx---	September 17, 2015 02:46 AM
☐	oozie-oozi		51emffu	51emffu	drwxr-xr-x	September 17, 2015 02:40 AM

Figure: File browser (Live system on gethue.com)

Hue: Lightweight Web Server for Hadoop

The screenshot displays the Hue web interface for editing Hive queries. The top navigation bar includes links for Query Editors, Data Browsers, Workflows, Suche, Security, and various browsers. The main area is titled 'Hive Editor' and 'Abfrage-Editor'. A sidebar on the left shows the 'DATENBANK' (default) and a list of tables including sample_07, sample_08, and their columns. The central query editor contains the following SQL:

```
1 SELECT * FROM sample_08
2 WHERE salary < 100000
```

Below the query editor are buttons for 'Ausführen', 'Speichern unter...', 'Erklären', and 'Neue Abfrage'. The results tab is active, showing a table with the following data:

	sample_08.code	sample_08.description	sample_08.total_emp	sample_08.salary
0	00-0000	All Occupations	135185230	42270
1	11-1031	Legislators	64650	37980
2	11-2011	Advertising and promotions managers	36100	94720
3	11-3011	Administrative services managers	246930	79500
4	11-3041	Compensation and benefits managers	38810	93410
5	11-3042	Training and development managers	29350	93830
6	11-3051	Industrial production managers	154030	91200

Figure: Query editor (Live system on gethue.com)

Hue: Lightweight Web Server for Hadoop

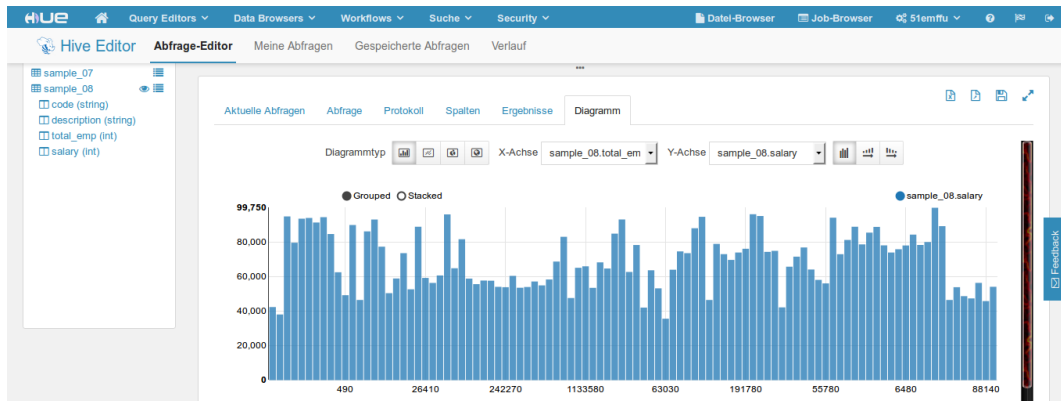


Figure: Visualizing query results in diagrams (Live system on gethue.com)

Zeppelin [39]

- Web-based notebook for interactive data analytics
 - ▶ Add code snippets
 - ▶ Arrange them
 - ▶ Execute them
 - ▶ Visualizes results
- Supports Spark, Scala, Pig, SQL, Python, R, Hive, Shell, ...
- Collaborative environment for multiple users
- Can export paragraph links for embedding into a webpage
- Similarities to Jupyter but has some improvements



Notebook ▾

Interpreter

Connected

Zeppelin Tutorial



? ⚙ default ▾

```
%md
## Welcome to Zeppelin.
#### This is a live tutorial, you can run the code yourself. (Shift-Enter to Run)
```

Took 1 seconds (outdated)

FINISHED ▶ ⌵ ⌲ ⌳ ⌴

Load Data Into Table

ERROR ▶ ⌵ ⌲ ⌳ ⌴

```
import org.apache.commons.io.IOUtils
import java.net.URL
import java.nio.charset.Charset

// Zeppelin creates and injects sc (SparkContext) and sqlContext (HiveContext or SqlContext)
// So you don't need create them manually

// load bank data
val bankText = sc.parallelize(
  IOUtils.toString(
    new URL("https://s3.amazonaws.com/apache-zeppelin/tutorial/bank/bank.csv"),
    Charset.forName("utf8")).split("\n"))

case class Bank(age: Integer, job: String, marital: String, education: String, balance: Integer)

val bank = bankText.map(s => s.split(";")).filter(s => s(0) != "\age\").map(
  s => Bank(s(0).toInt,
    s(1).replaceAll("\", ""),
    s(2).replaceAll("\", ""),
    s(3).replaceAll("\", ""),
    s(5).replaceAll("\", "").toInt
  )
).toDF()
bank.registerTempTable("bank")
```

Too many open files

Took 0 seconds (outdated)

```
%sql
select age, count(1) value
from bank
where age < 30
group by age
order by age
```

ERROR ▶ ⌵ ⌲ ⌳ ⌴

```
%sql
select age, count(1) value
from bank
where age < ${maxAge=30}
group by age
order by age
```

ERROR ▶ ⌵ ⌲ ⌳ ⌴

```
%sql
select age, count(1) value
from bank
where marital=${marital=single,single|divorced|married}
group by age
order by age
```

FINISHED ▶ ⌵ ⌲ ⌳ ⌴

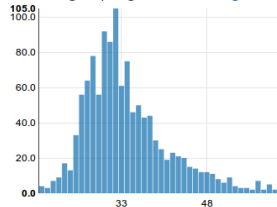
marital

single ▾



settings ▾

● Grouped ○ Stacked ● value



Outline

- 1 Hadoop Ecosystem
- 2 User/Admin Interfaces
- 3 Workflows**
 - Oozie
 - Falcon
 - Atlas
 - Sqoop
 - Slider
- 4 SQL Tools
- 5 Other BigData Tools

Oozie [15, 16]

- Scalable, reliable and extensible workflow scheduler
- Jobs are DAGs of actions specified in XML workflows
- Actions: Map-reduce, Pig, Hive, Spark, Shell actions
- Workflows can be parameterized
 - ▶ Triggers notifications via HTTP GET upon start/end of a node/job
 - ▶ Automatic user-retry to repeat actions when fixable errors occur
 - ▶ Monitors a few runtime metrics upon execution
- Interfaces: command line tools, web-service and Java APIs
- Integrates with HCatalog
- Coordinator jobs trigger start of jobs
 - ▶ By time schedules
 - ▶ When data becomes available
 - Requires polling of HDFS (1-10 min intervals)
 - With HCatalog's publish-subscribe, jobs can be started immediately
 - ▶ Can record events for service level agreement

Workflows [16]

- A workflow application is a ZIP file to be uploaded
 - ▶ Includes workflow definition and coordinator job
 - ▶ Bundles scripts, JARs, libraries needed for execution
- Workflow definition is a DAG with control flow and action nodes
 - ▶ Control flow: start, end, decision, fork, join
 - ▶ Action nodes: whatever to execute
- Variables/Parameters³
 - ▶ Default values can be defined in a config-default.xml in the ZIP
- Expression language functions help in parameterization¹
 - ▶ Basic functions: `timestamp()`, `trim()`, `concat(s1, s2)`
 - ▶ Workflow functions: `wf:errorCode(< action node >)`
 - ▶ Action specific functions: `hadoop:counters("mr-node")["FileSystemCounters"]["FILE_BYTES_READ"]`
- Coordinator job is also an XML file

³ They are used with with `${NAME/FUNCTION}`, e.g., `${timestamp()}`

Coordinator Jobs [17]

App which periodically starts a workflow (every 60 min)

```

1 <coordinator-app name="MY_APP" frequency="60" start="2009-01-01T05:00Z" end="2009-01-01T06:00Z" timezone="UTC"
  ↳ xmlns="uri:oozie:coordinator:0.1">
2   <action>
3     <workflow> <!-- here the workflow is not further defined -->
4       <app-path>hdfs://localhost:9000/tmp/workflows</app-path>
5     </workflow>
6   </action>
7 </coordinator-app>

```

Every 24h check if dependencies for a workflow are met, then run it

```

1 <coordinator-app name="MY_APP" frequency="1440" start="2009-02-01T00:00Z" end="2009-02-07T00:00Z" ...>
2   <datasets> <-- define a dataset, that is checked for existence -->
3     <dataset name="input1" frequency="60" initial-instance="2009-01-01T00:00Z" timezone="UTC">
4       <uri-template>hdfs://localhost:9000/tmp/revenue_feed/${YEAR}/${MONTH}/${DAY}/${HOUR}</uri-template>
5     </dataset>
6   </datasets>
7   <input-events> <-- we depend on the last 24 hours input data -->
8     <data-in name="coordInput1" dataset="input1">
9       <start-instance>${coord:current(-23)}</start-instance>
10      <end-instance>${coord:current(0)}</end-instance>
11    </data-in>
12  </input-events>
13  <action>
14    <workflow>
15      <app-path>hdfs://localhost:9000/tmp/workflows</app-path>
16    </workflow>
17  </action>
18 </coordinator-app>

```

Example Oozie Workflow [13]

Three actions: Execute pig script, concatenate reducer files, upload files remotely via ssh

```

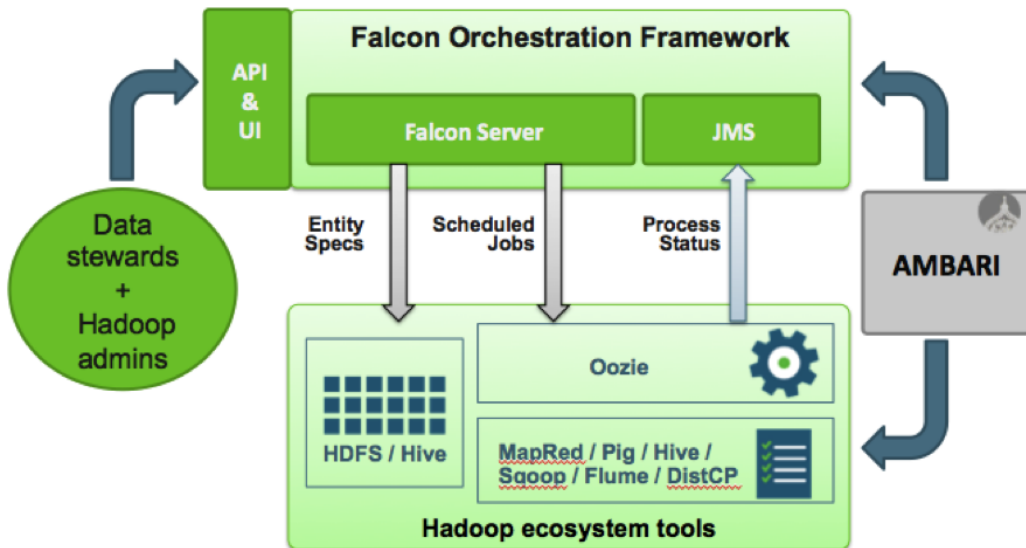
1 <workflow-app xmlns="uri:oozie:workflow:0.2" name="sample-wf">
2   <start to="pig" />
3   <action name="pig">
4     <pig><job-tracker>${jobTracker}</job-tracker>
5       <name-node>${nameNode}</name-node>
6       <prepare><delete path="${output}" /></prepare>
7       <configuration>
8         <property> <name>mapred.job.queue.name</name><value>${queueName}</value></property>
9         <property> <name>mapreduce.fileoutputcommitter.marksuccessfuljobs</name><value>true</value></property>
10      </configuration>
11      <script>${nameNode}/projects/bootcamp/workflow/script.pig</script>
12      <param>input=${input}</param>
13      <param>output=${output}</param>
14      <file>lib/dependent.jar</file>
15    </pig><ok to="concatenator" /><error to="fail" /> <-- the concatenator action is not shown here -->
16  </action>
17
18  <action name="fileupload">
19    <ssh><host>localhost</host>
20      <command>tmp/fileupload.sh</command>
21      <args>${nameNode}/projects/bootcamp/concat/data-${fileTimestamp}.csv</args><args>${wf:conf("ssh.host")}</args>
22      <capture-output></ssh>
23    <ok to="fileUploadDecision" /><error to="fail" />
24  </action>
25
26  <decision name="fileUploadDecision"> <-- check the exit status of the file upload -->
27    <switch><case to="end">${wf:actionData('fileupload')['output']} == '0'</case><default to="fail" /> </switch>
28  </decision>
29  <kill name="fail"><message>Workflow failed, error message[${wf:errorMessage(wf:lastErrorNode())}]</message></kill>
30  <end name="end" />
31 </workflow-app>

```

Falcon [11,13]

- Feed (data set) management and processing system
- Simplifies dealing with many Oozie jobs
- Supports data governance
 - ▶ Define and run data pipelines (management policies)
 - ▶ Monitor data pipelines
 - ▶ Trace pipelines to identify dependencies and perform audits
- Data model defines entities describing policies and pipelines
 - ▶ Clusters define resources and interfaces to use
 - ▶ Feeds define frequency, data retention, input, outputs, retry and use clusters (multiple for replication)
 - ▶ Process: processing task, i.e., Oozie workflow, Hive or Pig script
- Features
 - ▶ Supports reuse of entities for different workflows
 - ▶ Enables replication across clusters and data archival
 - ▶ Supports HCatalog
 - ▶ Notification of users upon availability of feed groups

Falcon: High-level Architecture



Falcon: Example Pipeline

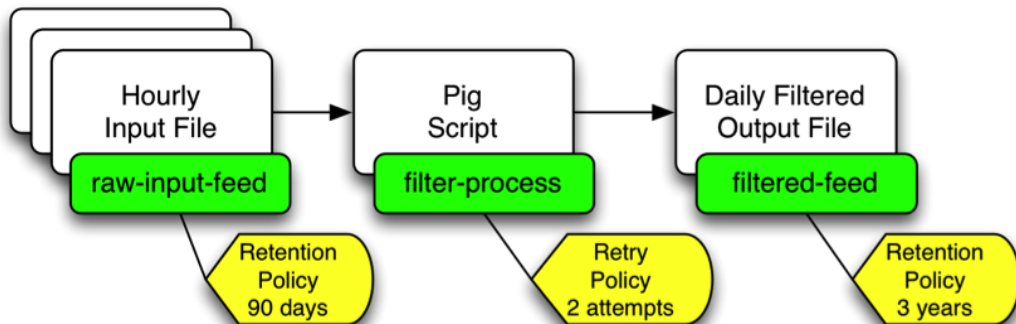


Figure: Source: [11]

Falcon: Example Process Definition [11, 14]

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!-- Sample process. Runs at 6th hour every day. Input: last day hourly data. Output: for yesterday -->
3 <process name="SampleProcess">
4   <cluster name="wr" />
5   <frequency>days(1)</frequency>
6
7   <validity start="2015-04-03T06:00Z" end="2022-12-30T00:00Z" timezone="UTC" />
8
9   <inputs>
10     <input name="input" feed="SampleInput" start="yesterday(0,0)" end="today(-1,0)" />
11   </inputs>
12
13   <outputs>
14     <output name="output" feed="SampleOutput" instance="yesterday(0,0)" />
15   </outputs>
16
17   <properties>
18     <property name="queueName" value="reports" />
19     <property name="ssh.host" value="host.com" />
20     <property name="fileTimestamp" value="{\coord:formatTime(coord:nominalTime(), 'yyyy-MM-dd')}" />
21   </properties>
22
23   <workflow engine="oozie" path="/projects/bootcamp/workflow" />
24
25   <retry policy="backoff" delay="minutes(5)" attempts="3" />
26
27   <-- How to check and handle late arrival of input data-->
28   <late-process policy="exp-backoff" delay="hours(1)">
29     <late-input input="input" workflow-path="/projects/bootcamp/workflow/lateinput" />
30   </late-process>
31 </process>
```

Atlas [23]

- A framework for platform-agnostic data governance and metadata management
- Gather, process and preserve metadata; exchange with other tools
- Dynamic creation of tags for classification
- Audit operations, explore data lineage (history) of data and metadata
- Support lifecycle management workflows (Falcon) and access control (Ranger)

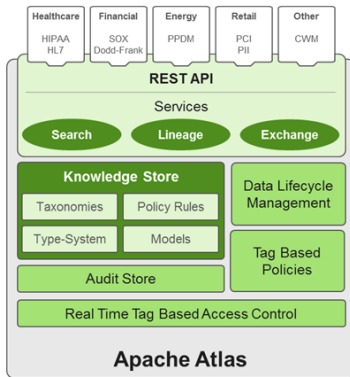


Figure: Source: [48]

Atlas: Metadata Fields

Apache Atlas

SEARCH CLASSIFICATION GLOSSARY

Basic ☒ Advanced ⓘ

Search By Type
DataFile_7 (3)

Search By Classification
Select Classification

Search By Term
Search Term

Search By Text
Search by text

Clear Search

Favorite Searches Save Save As

You don't have any favorite search.

Back To Results

campaign_file (DataFile_7)

Classifications: A_GNMOT

Term:

Properties Lineage Relationships Classifications Audits

Key	Value
Schema	moat_nielsen__instagram_stories_display
business_tags	<pre>{ tenant_id: "r09e2", market: "US", subtenant_id: "y5f68", file_type: "instagram_stories_display", link_id: "111" }</pre>
file_name	campaign_file
name	campaign_file
qualifiedName	hdfs://A_GNMOT/US/current_transformers/moat/18ce53vrr8e/campaign_management_accounts/2020-04-23-17-21-34.337645/test_file_111.csv
technical_tags	<pre>{ date: "2020-04-19T00:00:00Z", update_type: { incremental: "{}" }, file_ext: ".csv" }</pre>

Show Empty Values

Figure: Source: [49]

Atlas: Metadata Search

The screenshot shows the Apache Atlas Metadata Search interface. On the left is a dark sidebar with navigation links: SEARCH (active), CLASSIFICATION, and GLOSSARY. Below these are tabs for Basic (selected) and Advanced. The Basic search section includes filters for Search By Type (DataFile), Search By Classification (DCM), Search By Term, and Search By Text (adver*). There are buttons for Clear and Search, and a Favorite Searches section at the bottom. The main content area displays search results for the query: (Type: DataFile) AND (Classification: DCM) AND (Query: adver*). It shows 1 record from 1 to 1. A table lists the results with columns: Name, Owner, Description, Type, Classifications, and Term. The first result is dcm_mathctable_advertiser_2019_03_13.csv, which is a DataFile. Annotations with callout boxes highlight specific features: 'Filter by Data Asset Type' points to the Search By Type dropdown; 'Filter by Classification' points to the Search By Classification dropdown; 'Search by Text Wildcards: adver*, placem*' points to the Search By Text input field. A large box at the top right of the main area is titled 'Metadata Catalog Search: Free Text'. At the bottom right, there is a 'Page Limit' dropdown set to 25.

Apache Atlas

SEARCH CLASSIFICATION GLOSSARY

Basic Advanced

Search By Type
DataFile

Search By Classification
DCM

Search By Term
Search Term

Search By Text
adver*

Clear Search

Favorite Searches
Save Save As

You don't have any favorite search.

Results for: (Type: DataFile) AND (Classification: DCM) AND (Query: adver*)
If you do not find the entity in search result below then you can [create new entity](#)

Showing 1 records From 1 - 1

Filter by Data Asset Type

Filter by Classification

Search by Text
Wildcards: adver*, placem*

Metadata Catalog Search: Free Text

Exclude sub-types Exclude sub-classifications Show historical entities Columns

Name	Owner	Description	Type	Classifications	Term
dcm_mathctable_advertiser_2019_03_13.csv			DataFile	Expires On + -	+

Page Limit: 25

Figure: Source: [49]

Atlas: Data Lineage Visualization

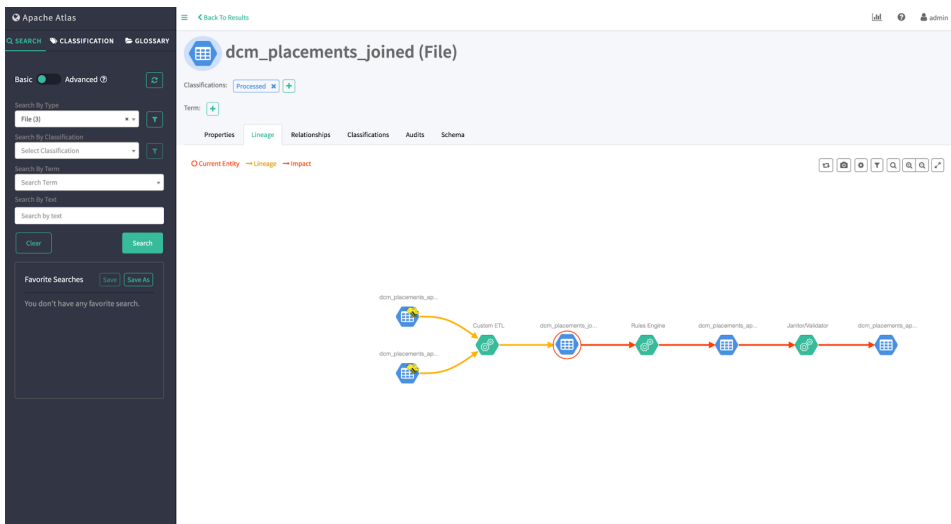


Figure: Source: [49]

Sqoop (A retired tool replaced by Flume...) [18, 19]

- Transfers bulk data between Hadoop and RDBMS, either
 - ▶ One/multiple tables (preserving their schema)
 - ▶ Results of a free-form SQL query
- Uses MapReduce to execute import/export jobs
 - ▶ Parallelism is based on splitting one column's value
- Validate data transfer (comparing row counts) for full tables
- Save jobs for repeated invocation
- Main command line tool sqoop, more specific tools sqoop*

Features [19]

Import Features

- Incremental import (scan and add only newer rows)
- File formats: CSV, SequenceFiles, Avro, Parquet
 - ▶ Compression support
- Outsource large BLOBS/TEXT into additional files
- Import into Hive (and HBase)
- Can create the table schema in HCatalog automatically
 - ▶ With HCatalog, only CSV can be imported

Export Features

- Bulk insert: 100 records per statement
- Periodic commit after 100 statements

Import Process [19]

- Read the schema of the source table
- Create a Java class representing a row of the table
 - ▶ This class can be used later to work with the data
- Start MapReduce to load data parallel into multiple files
 - ▶ The number of mappers can be configured
 - ▶ Mappers work on different values of the splitting column
 - ▶ The default splitting column is the primary key
 - Determines min and max value of the key
 - Distributes fixed chunks to mappers
- Output status information to the MapReduce job tracker

Example Imports [19]

```
1 # Import columns from "foo" into HDFS to /home/x/foo (table name is appended)
2 # When not specifying any columns, all columns will be imported.
3 $ sqoop import --connect jdbc:mysql://localhost/db --username foo --table TEST --columns
   ↪ "matrikel,name" --warehouse-dir /home/x --validate
4
5 # We'll use a free-form query, it is parallelized on the split-by column
6 # The value is set into the magic $CONDITIONS variable
7 $ sqoop import --query 'SELECT a.*, b.* FROM a JOIN b on (a.id == b.id) WHERE $CONDITIONS'
   ↪ --split-by a.id --target-dir /user/foo/joinresults
8
9 # To create the HCatalog table use --hcatalog-table or --hive-import
10 # See [19] for details of the available options
```

Slider [20]

- Is a YARN application that manages non-YARN apps on a cluster
- ⇒ Utilize YARN for resource management
- Enables installation, execution, monitoring and dynamic scaling
- Command line tool `slider`
- Apps are installed and run from a package
 - ▶ Tarball with well-defined structure [21]
 - ▶ Scripts for installing, starting, status, ...
- Example packages: `jmemcached`, `HBase`
- Slider is currently extended to deploy Docker images (Tech preview)

Outline

- 1 Hadoop Ecosystem
- 2 User/Admin Interfaces
- 3 Workflows
- 4 SQL Tools**
 - Drill
 - Impala
- 5 Other BigData Tools
- 6 Machine Learning
- 7 Summary

Drill [10, 29, 30]

- Software framework for data-intensive distributed applications
- Data model: relational (ANSI SQL !) + schema-free JSON
- Analyse data in-situ without data movement
 - ▶ Execute one query against multiple NoSQL datastores
 - ▶ Datastores: HBase, MongoDB, HDFS, S3, Swift, local files
- Features
 - ▶ REST APIs
 - ▶ Columnar execution engine supporting complex data
 - ▶ Locality-aware execution
 - ▶ Cost-based optimizer pushing processing into datastore
 - ▶ Runtime compilation of queries

```
1 # Different datastores, localstorage, mongodb and s3
2 SELECT * FROM dfs.root.'/logs';
3 SELECT country, count(*) FROM mongodb.web.users GROUP BY country;
4 SELECT timestamp FROM s3.root.'users.json' WHERE user_id = 'max';
5
6 # Query JSON: access the first students age from private data (a map)
7 SELECT student[0].private.AGE, FROM dfs.'students.json';
```

Cloudera Impala [25, 26]

- Enterprise analytic database
 - ▶ Utilizes HDFS, HBase and Amazon S3
 - ▶ Based on Google Dremel like Apache Drill
- Written in C++, Java
- Massively-parallel SQL engine
 - ▶ Supports HiveQL and subset of ANSI-92 SQL
- Uses LLVM to generate efficient code for queries

Apache Metamodel [43]

- Provides a Java based SQL-alike interface to various data sources
 - ▶ CSV, SQL dbs, JSON, HBase, MongoDB

Query [43]

```
1 DataContext dataContext = DataContextFactory.create[TypeOfDatastore](...);
2 DataSet dataSet = dataContext.query()
  ↳ .from("libraries").select("name").where("language").eq("Java").and("enhances_data_access").eq(true).execute();
```

Update [43]

```
1 DataContext.executeUpdate(new UpdateScript() {
2     public void run(UpdateCallback callback) {
3         // CREATE a table
4         Table table = callback.createTable("contributors")
5             .withColumn("id").ofType(INTEGER).withColumn("name").ofType(VARCHAR).execute();
6
7         // INSERT INTO table
8         callback.insertInto(table).value("id", 1).value("name", "John Doe").execute();
9         callback.insertInto(table).value("name", "Jane D.").execute();
10
11        // UPDATE table
12        callback.update(table).value("name", "Jane Doe").where("id").eq(2).execute();
13
14        // DELETE FROM table
15        callback.deleteFrom(table).where("id").eq(1).execute();
16    }
17 });
```

Outline

- 1 Hadoop Ecosystem
- 2 User/Admin Interfaces
- 3 Workflows
- 4 SQL Tools
- 5 Other BigData Tools**
 - Kafka
 - Solr
- 6 Machine Learning
- 7 Summary

Kafka [41]

■ Publish-subscribe distributed messaging system

- ▶ Producer publishes message for a given **topic**
- ▶ Consumer subscribes to topic and receives data
- ▶ Simple: Consumer has to remember its read position (**offset**)

■ A data source for Storm, HBase, Spark, ...

■ Use cases – support data ingestion:

- ▶ GPS data from truck fleet, sensor data
- ▶ Error logs from cluster nodes, web server activity

■ Features

- ▶ Parallel, fault-tolerant server system (a server is called **broker**)

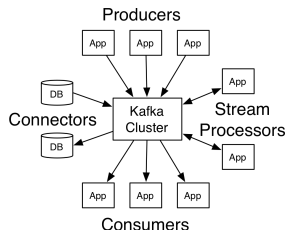


Figure: Source: [42]

Solr [10, 31]

- Full-text search and indexing platform
- REST API: index documents and query via HTTP
 - ▶ Query response in JSON, XML, CSV, binary
- Features
 - ▶ Data can be stored on HDFS
 - ▶ High-availability, scalable and fault tolerant
 - ▶ Distributed search
 - ▶ Faceted classification: organize knowledge into a systematic order using (general or subject-specific) semantic categories that can be combined per classification entry [10]
 - Examples: country, color, size, product type
 - ▶ Geo-spatial search
 - ▶ Caching of queries, filters and documents
- Uses lucene library for search
- Very similar: Elasticsearch [33], <http://solr-vs-elasticsearch.com/>

Example Query [32]

Identifying available facets terms and number of docs for each

```
1 curl http://localhost:8983/solr/gettingstarted/select?wt=json&indent=true&q=*&rows=0&facet=true& facet.field=manu_id_s
```

Response

```
1 {
2   "responseHeader":{
3     "status":0,
4     "QTime":3,
5     "params":{ /* Parameters of the query */
6       "facet":"true", "indent":"true", "q":"*:*", "facet.field":"manu_id_s", "wt":"json",
7       "rows":"0"}},
8   "response":{"numFound":2990,"start":0,"docs":[]}, /* number of documents found */
9   "facet_counts":{"
10     "facet_queries":{},
11     "facet_fields":{" /* the available facets and number of documents */
12       "manu_id_s":["corsair",3, "belkin",2, "canon",2, "apple",1, "asus",1, "ati",1, "boa",1, "dell",1, "eu",1, "maxtor",1, "nor",1, "uk",1,
13         ↪ "viewsonic",1, "samsung",0]},
14     "facet_dates":{},
15     "facet_ranges":{},
16     "facet_intervals":{}}
```

Outline

- 1 Hadoop Ecosystem
- 2 User/Admin Interfaces
- 3 Workflows
- 4 SQL Tools
- 5 Other BigData Tools
- 6 Machine Learning**
 - Mahout
 - TensorFlow
- 7 Summary

Mahout [34]

■ Framework for scalable machine learning

- ▶ Collaborative filtering
- ▶ Classification
- ▶ Clustering
- ▶ Dimensionality reduction
- ▶ Recommender
 - history: user purchases + all purchases $\Rightarrow$ recommendations (user)

■ Computation on Spark, MapReduce, H2O engines [36]

- ▶ Can also use a single machine without Hadoop
- ▶ Algorithm availability depends on the backend

■ Bindings for Scala language [35]

- ▶ Provide distributed BLAS, Row Matrix (DRM)
- ▶ R-like DSL embedded in Scala
- ▶ Algebraic optimizer

Recommender Architecture

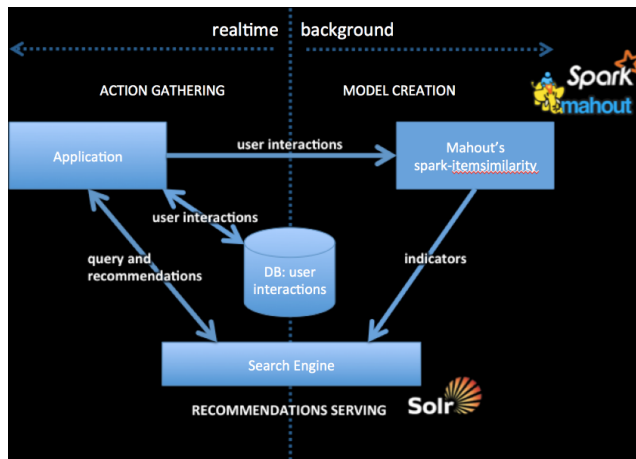


Figure: Source: [36]

- 1 Collect user interactions
 $n \times (\text{user-id, item-id})$
- 2 Learning:
 1. Itemsimilarity creates item, list-of-similar-items
 2. Store those tuples in the search engine
- 3 Query search engine with n latest user interactions
- 4 If they occur in the list-of-similar-items, recommend item

TensorFlow [44]

- Platform independent opens-source library for machine learning
- GPU accelerated
- Developed by GoogleBrain
- Tensor: Multidimensional array with rank dimensions
- Workflow
 - ▶ Define tensors (or placeholders)
 - ▶ Build execution graph based on equations and optimizers (math)
 - ▶ Start execution (bind dummy variables)
- Big community
- PyTorch alternative from Facebook with dynamic graph creation
- Nowadays typically used with Keras that provides high-level NN-architectures

TensorFlow 1.X: Example for MNIST Dataset [47]

```
1  # Here build a neural network of a linear classifier to estimate classes
2  import tensorflow as tf
3  # Create two tensors holding features and labels for MNIST
4  # 28x28 images, number is unknown (None)
5  x = tf.placeholder(tf.float32, [None, 784], name='Feature') # name is useful in the GUI
6  # Create tensor to recognize 0-9 => 10 classes
7  y = tf.placeholder(tf.float32, [None, 10], name='Label')
8
9  # Values to compute for a linear classifier, for each pixel compute chance it is class 0-9
10 W = tf.Variable(tf.zeros([784, 10]), name='Weights')
11 b = tf.Variable(tf.zeros([10]), name='Bias')
12
13 # Formulate solution with tensors
14 # Softmax returns a probability for each class that sums up to 1; here 10 classes
15 pred = tf.nn.softmax( tf.matmul(x, W) + b )
16 # Cost function for gradient descent; here is the cross entropy
17 cost = tf.reduce_mean(-tf.reduce_sum(y * tf.log(pred), reduction_indices=1))
18 # Learning algorithm to train the network
19 optimizer = tf.train.GradientDescentOptimizer(learning_rate).minimize(cost)
20
21 # Start execution, assume features and labels have been loaded
22 sess.run([optimizer, cost], feed_dict={x: features, y: labels})
```

TensorBoard [45]

■ Browser visualization for Tensorflow

▶ Model graph

- Operations can be given names
- Name scopes organize operations together

▶ Results: scalar, events, histogram, multi-dimensional, audio, images, ...

▶ e.g., visualize accuracy over training steps

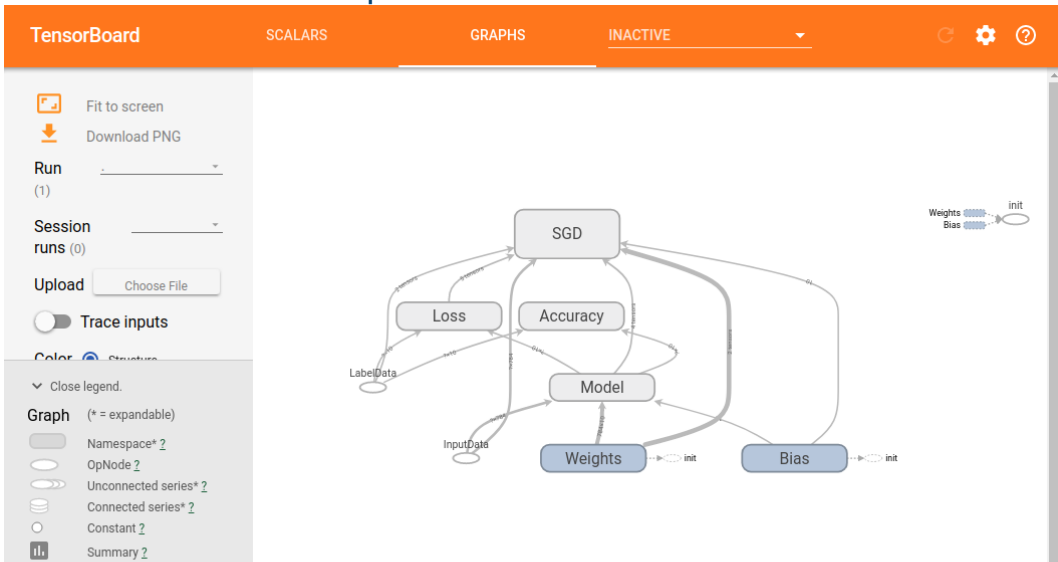
▶ Embeddings: a mapping from discrete objects, e.g., words, to vectors of real numbers

■ Reads (binary) output from a log directory, can be multiple runs!

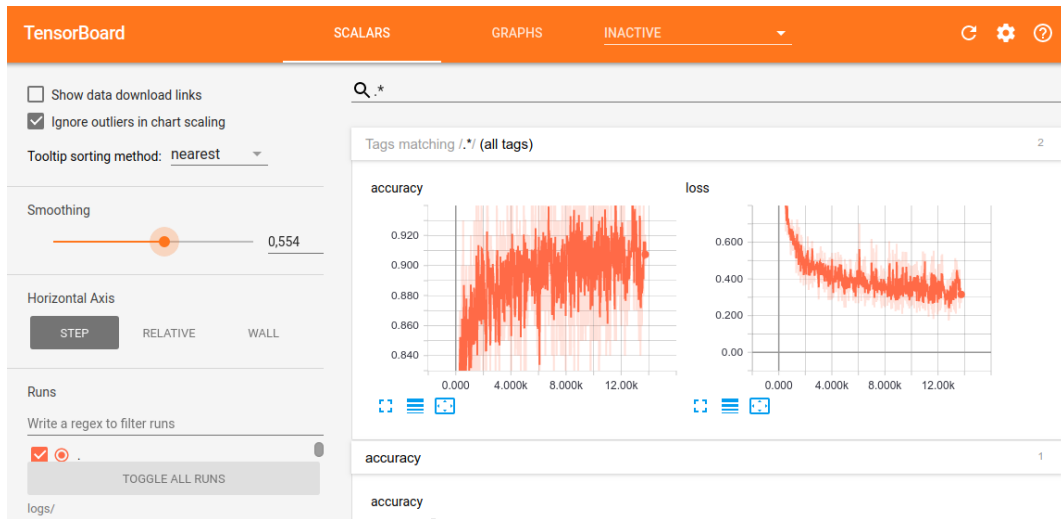
▶ Uses protocol buffers for serialization

▶ Analyze hyperparameter search

TensorBoard: Model Graph Visualization



TensorBoard: Result Visualization



Summary

- The Apache community is very active
- Software responsibilities:
 - ▶ Hadoop deployment and cluster management
 - ▶ Data management and provenance
 - ▶ Security
 - ▶ Analysis
 - ▶ Automation (scheduling, data ingestion)
- One goal: simple usage
 - ▶ Alternative user interfaces
 - ▶ Research of domain-specific languages (XML based or language embedded)
- Many software packages are used but still in Apache incubator (beta)

Bibliography I

- 10 Wikipedia
- 11 <http://hortonworks.com/blog/introduction-apache-falcon-hadoop/>
- 12 <http://gethue.com/>
- 13 <http://falcon.apache.org/>
- 14 <http://falcon.apache.org/0.7/EntitySpecification.html>
- 15 <http://oozie.apache.org/>
- 16 <http://oozie.apache.org/docs/4.2.0/index.html>
- 17 <https://github.com/yahoo/oozie/wiki/Oozie-Coord-Use-Cases>
- 18 <http://sqoop.apache.org/>
- 19 <http://sqoop.apache.org/docs/1.4.6/SqoopUserGuide.html>
- 20 <http://slider.incubator.apache.org/>
- 21 http://slider.incubator.apache.org/docs/slider-specs/application_package.html
- 22 <https://knox.apache.org/>
- 23 <http://hortonworks.com/blog/apache-atlas-project-proposed-for-hadoop-governance/>
- 24 <http://ranger.incubator.apache.org/>
- 25 <https://www.cloudera.com/products/enterprise-data-hub.html>
- 26 <http://www.cloudera.com/content/cloudera/en/products-and-services/cloudera-enterprise.html>
- 27 <https://github.com/cloudera/impala>
- 29 <http://incubator.apache.org/drill/>
- 30 <https://drill.apache.org/docs/json-data-model/>
- 31 <http://lucene.apache.org/solr/features.html>

Bibliography II

```
32 http://lucene.apache.org/solr/quickstart.html
33 http://solr-vs-elasticsearch.com/
34 http://mahout.apache.org/
35 http://www.scala-lang.org/
36 http://h2o.ai/
37 https://mahout.apache.org/users/recommender/quickstart.html
38 Free Ebook: https://www.mapr.com/practical-machine-learning
39 https://zeppelin.incubator.apache.org/
40 http://hortonworks.com/products/data-center/hdp/
41 http://hortonworks.com/apache/kafka/
42 http://kafka.apache.org/intro
43 http://metamodel.apache.org/
44 https://www.tensorflow.org/
45 https://github.com/tensorflow/tensorboard
46 https://www.tensorflow.org/get_started/mnist/beginners
47 https://www.tensorflow.org/get_started/get_started
48 https://docs.cloudera.com/HDPDocuments/HDP3/HDP-3.1.0/atlas-overview/content/apache-atlas-features.html
49 https://blog.dotmodus.com/what-is-apache-atlas-and-why-is-it-important/
```