

Julian Kunkel

The Apache Ecosystem and Beyond



Hortonworks (2019 bought by Cloudera)

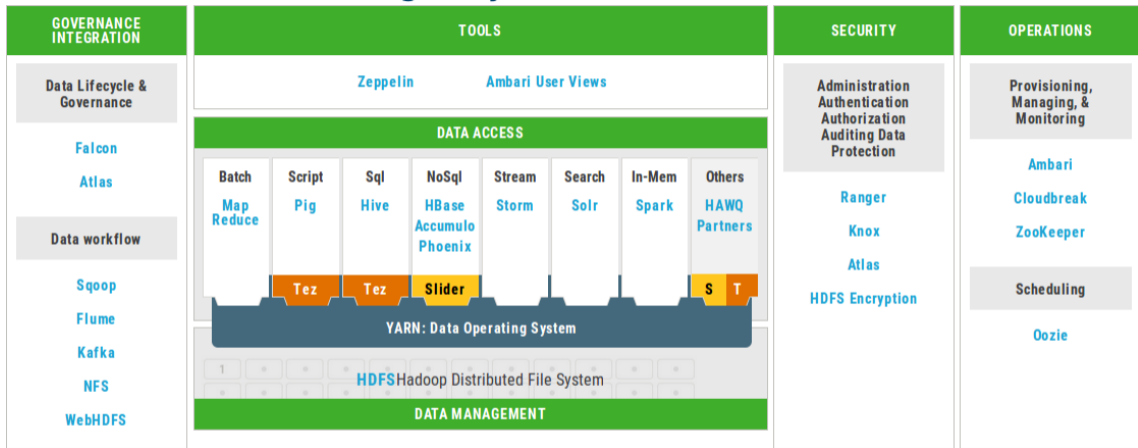


Figure: Screenshot from [40]

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

Management with Ambari: Dashboard

Ambari wr 0 ops 1 alert Dashboard Services Hosts 1 Alerts Admin admin

Metrics Heatmaps Config History

Metric Actions

HDFS Disk Usage 	DataNodes Live 5/5	HDFS Links NameNode Secondary NameNode 5 DataNodes More...	Memory Usage 	Network Usage
CPU Usage 	Cluster Load 	NameNode Heap 	NameNode RPC 0 ms	NameNode CPU WIO
NameNode Uptime 1.2 hr	HBase Master Heap 	HBase Links HBase Master 5 RegionServers Master Web UI More...	HBase Ave Load 1	HBase Master Uptime 32.9 min
ResourceManager Heap 	ResourceManager Uptime 25.9 min	NodeManagers Live 5/5	YARN Memory 	YARN Links ResourceManager 5 NodeManagers More...

HDFS
 MapReduce2
 YARN
 Tez
 Hive 1
 HBase
 Pig
 Sqoop
 Oozie
 ZooKeeper
 Falcon
 Storm
 Flume
 Ambari Metrics
 Kafka
 Knox
 Slider
 Spark

Actions

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 23936

MB

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: Lightweight Web Server for Hadoop

HUE Home Query Editors Data Browsers Workflows Suche Security Datel-Browser Job-Browser 51emffu

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

Abgeschlossen

Fertigstellung	Status	Name	Dauer	Sender	Häufigkeit	Startzeit	ID
Wed, 30 Sep 2015 22:41:00	KILLED	My_Coordinator	93d:14h:56m:0s	dtto9j	*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	SUCCEEDED	My_Coordinator	7d:0h:0m:0s	9jaipv9	1 0,6***	Tue, 18 Aug 2015	0000504-150730175918991-oozie-oozi-C

Feedback

Hue: Lightweight Web Server for Hadoop

The screenshot shows the Hue web interface. At the top, there is a navigation bar with the Hue logo and various menu items: Query Editors, Data Browsers, Workflows, Suche, Security, Data-Browser, Job-Browser, and a user profile for '51emffu'. Below this is a sub-navigation bar for the 'Hive Editor' with options like 'Abfrage-Editor', 'Meine Abfragen', 'Gespeicherte Abfragen', and 'Verlauf'.

On the left side, there is a sidebar with 'Unterstützung' and 'Einstellungen'. The 'DATENBANK' section shows a dropdown menu set to 'default' and a text input for 'Tabellenname...'. Below that, a list of tables is shown, including 'sample_07' and 'sample_08'. The 'sample_08' table is expanded, showing columns: 'code (string)', 'description (string)', 'total_emp (int)', and 'salary (int)'.

The main area is the 'Abfrage-Editor' (Query Editor). It contains a text area with the following SQL query:

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

Below the query editor are several buttons: 'Ausführen' (Execute), 'Speichern unter...' (Save as...), 'Erklären' (Explain), 'oder erstellen Sie eine' (or create a), and 'Neue Abfrage' (New Query).

Below the buttons, there are tabs for 'Aktuelle Abfragen', 'Abfrage', 'Protokoll', 'Spalten', 'Ergebnisse' (Results), and 'Diagramm'. The 'Ergebnisse' tab is active, displaying a table of results:

	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	93630
6	11-3051	Industrial production managers	154030	91200

On the right side of the interface, there is a vertical 'Feedback' button.

Hue: Lightweight Web Server for Hadoop

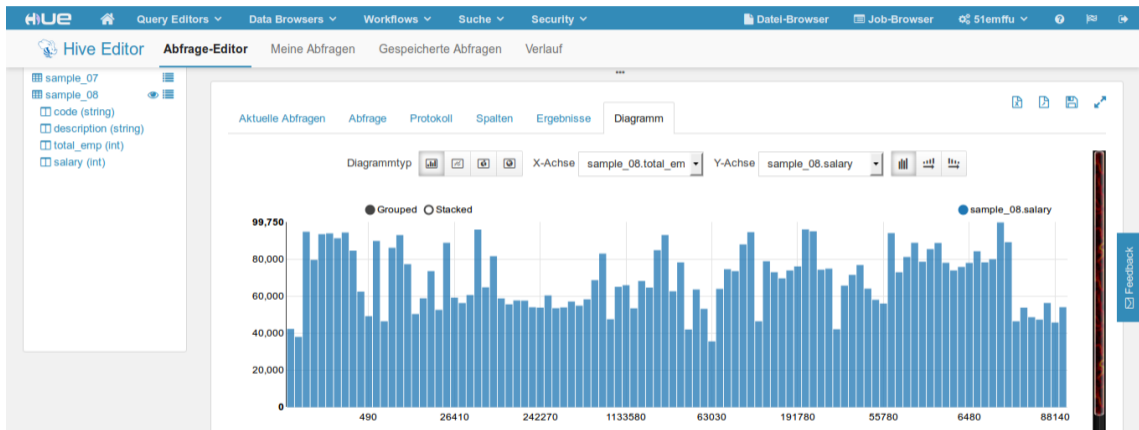


Figure: Visualizing query results in diagrams (Live system on gethue.com)



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 marital=${marital=single,single|divorced|married}
group by age
order by age
```

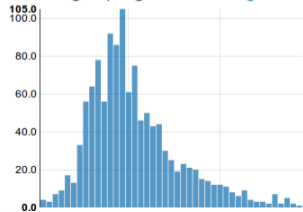
FINISHED ▶ ⌘ 📄 ⚙

marital

single ▾

📄 📊 📉 📈 📉 📈 settings ▾

● Grouped ○ Stacked ● value



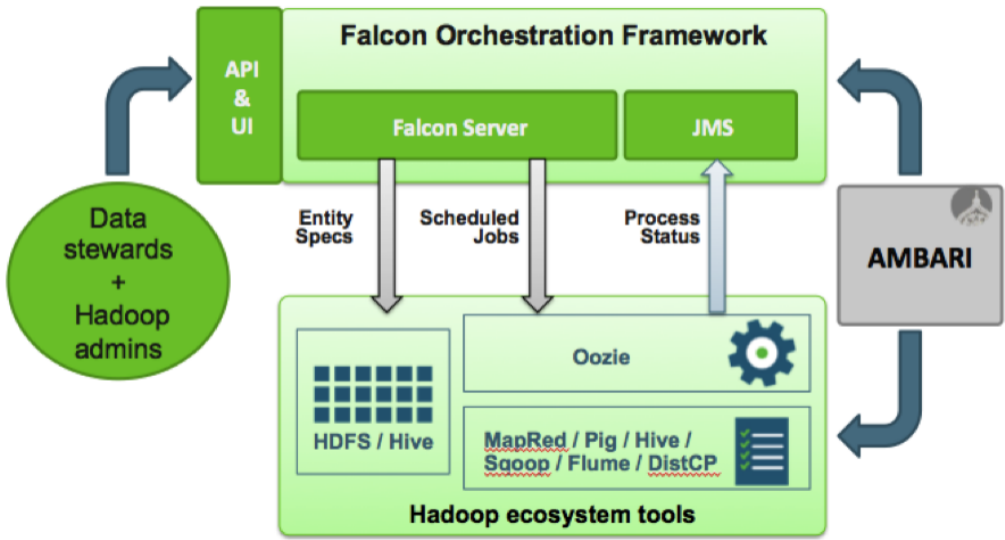
```
%sql
select age, count(1) value
from bank
```

ERROR ▶ ⌘ 📄 ⚙

```
%sql
select age, count(1) value
from bank
```

ERROR ▶ ⌘ 📄 ⚙

Falcon: High-level Architecture



Falcon: Example Pipeline

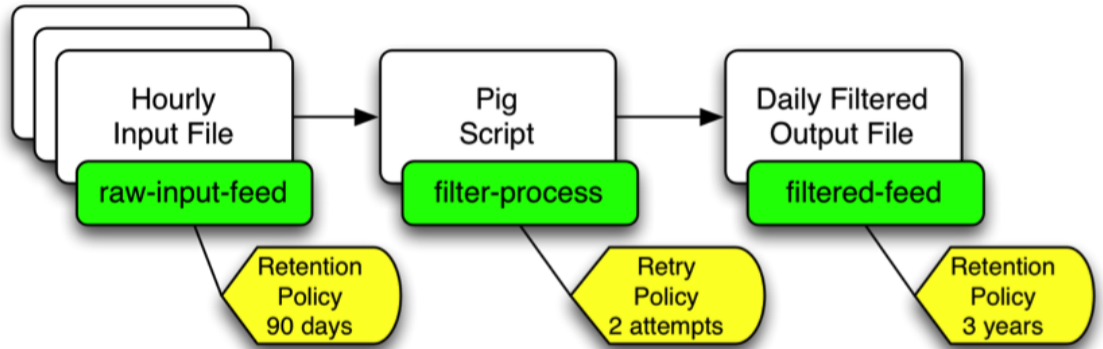
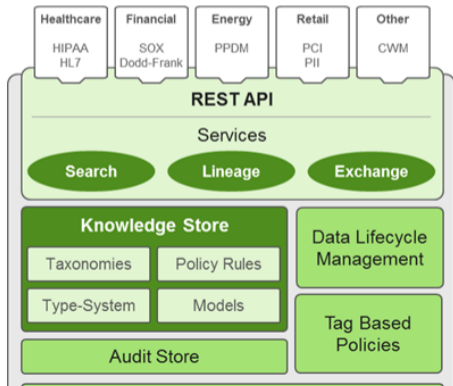


Figure: Source: [11]

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)



Source: [48]

Atlas: Metadata Fields

The screenshot shows the Apache Atlas web interface. On the left is a dark sidebar with search filters. The main area displays the metadata for 'campaign_file (DataFile_7)'. The 'Classifications' section shows 'A_GNMOT'. The 'Term' field is empty. Below are tabs for Properties, Lineage, Relationships, Classifications, and Audits. The 'Properties' tab is active, showing a table of key-value pairs.

Search Filters (Left Sidebar):

- Basic (selected) / Advanced
- Search By Type: DataFile_7 (3)
- Search By Classification: Select Classification
- Search By Term: Search Term
- Search By Text: Search by text
- Buttons: Clear, Search
- Favorite Searches: Save, Save As
- Message: You don't have any favorite search.

Metadata Details (Main Content):

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", client: "A_GNMOT" }</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>

Figure: Source: [49]

Atlas: Metadata Search

The screenshot shows the Apache Atlas Metadata Search interface. On the left is a dark sidebar with search filters, and on the right is the main search results area.

Search Filters (Left Sidebar):

- Mode: Basic (selected), Advanced
- Search By Type: DataFile
- Search By Classification: DCM
- Search By Term: Search Term
- Search By Text: adver*
- Buttons: Clear, Search
- Favorite Searches: Save, Save As
- Message: You don't have any favorite search.

Search Results (Main Area):

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

Buttons: Exclude sub-types, Exclude sub-classifications, Show historical entities, Columns

Name	Owner	Description	Type	Classifications	Term
dcn_mathctable_advertiser_2019_03_13.csv			DataFile	Expires On	

Page Limit: 25

Callouts:

- Filter by Data Asset Type:** Points to the 'DataFile' dropdown in the 'Search By Type' filter.
- Filter by Classification:** Points to the 'DCM' dropdown in the 'Search By Classification' filter.
- Search by Text Wildcards: adver*, placem*:** Points to the 'adver*' text input in the 'Search By Text' filter.

Atlas: Data Lineage Visualization

Apache Atlas

SEARCH CLASSIFICATION GLOSSARY

Basic Advanced

Search By Type: File (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

dcm_placements_joined (File)

Classifications: Processed

Term: +

Properties Lineage Relationships Classifications Audits Schema

Current Entity → Lineage → Impact

```

    graph LR
      A[dcm_placements_ap...] --> B[Custom ETL]
      C[dcm_placements_ap...] --> B
      B --> D[dcm_placements_joined]
      D --> E[Rules Engine]
      E --> F[dcm_placements_ap...]
      F --> G[Jaritor/Validator]
      G --> H[dcm_placements_ap...]
  
```

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)

Apache Airflow™ : What is Airflow?



- An open-source platform for batch-oriented workflow orchestration.
- Workflow is referred as DAGs (Direct Acyclic Graphs)
- Mainly, used for:
 - ▶ Developing,
 - ▶ Scheduling, and
 - ▶ Monitoring DAGs
- It requires a Linux server and has its own backend interface.
- Its user interface provides in-depth views of two things:
 - ▶ Pipelines, and
 - ▶ Tasks

Apache Airflow™ : Graphical View

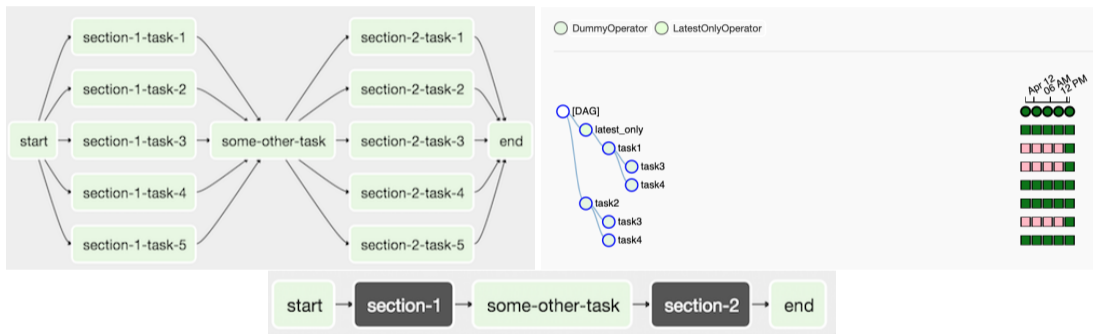


Figure: This example demonstrates different aspects of the Airflow DAG.⁴⁶

⁴⁶ Source: <https://airflow.apache.org/docs/apache-airflow/stable/index.html>

Apache Airflow™ : Purpose

Airflow is for?

■ Programmable workflows as:

▶ Dynamic:

- Pipelines are configured as Python code, allowing for dynamic generation.

▶ Extensible:

- The framework contains operators to connect with numerous technologies, and
- Are easily adjustable to custom environments.

▶ Flexible:

- Workflow parameterization is built-in leveraging the Jinja templating engine.
- Workflows built for finite batch and are triggerable through CLI and REST API.

Airflow is not for?

■ It is not a streaming solution such as Apache Kafka.

■ It was not built for infinitely running event-based workflows.

Apache Airflow™ : Architecture

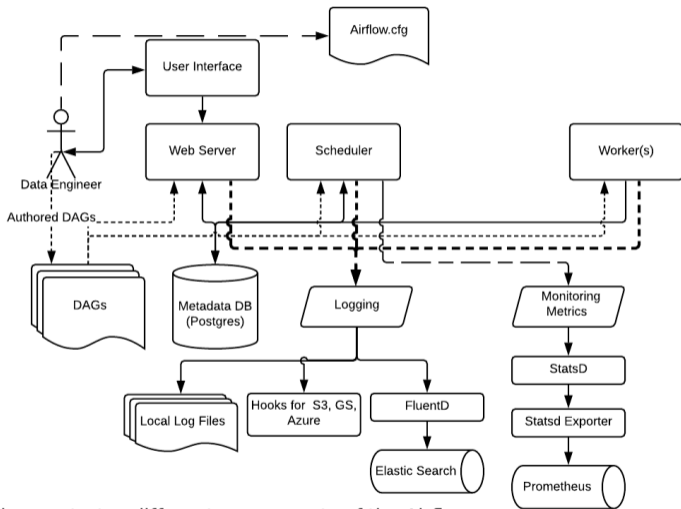


Figure: This example demonstrates different components of the Airflow.

Source: <https://airflow.apache.org/docs/apache-airflow/stable/index.html>

Apache Airflow™ : Hello World

```
1 from datetime import datetime
2
3 from airflow import DAG
4 from airflow.decorators import task
5 from airflow.operators.bash import BashOperator
6
7 # A DAG represents a workflow, a collection of tasks
8 with DAG(dag_id="demo", start_date=datetime(2022, 1, 1), schedule="0 0 * * *") as dag:
9     # Tasks are represented as operators
10    hello = BashOperator(task_id="hello", bash_command="echo hello")
11
12 @task()
13 def airflow():
14     print("airflow")
15
16 # Set dependencies between tasks
17 hello >> airflow()
```

Apache Airflow™ : Why Airflow?

■ Airflow is the tool for workflows defined as Python code which means:

- ▶ Workflows can be stored in version control,
- ▶ Workflows can be developed by multiple users simultaneously,
- ▶ Tests can be written to validate the functionality,
- ▶ Its components are reusable and extensible to a wide range.
- ▶ Installable from PyPI:

```
1 pip install "apache-airflow[celery]==2.8.0" --constraint  
2 "https://raw.githubusercontent.com/apache/airflow/constraints-2.8.0/constraints-3.8.txt"
```

- ▶ Support for Python and Kubernetes versions⁴⁷.

⁴⁷

More details: <https://airflow.apache.org/docs/apache-airflow/stable/installation/supported-versions.html>

Apache Airflow™ : Security

■ Airflow Security Model

- ▶ Model to help users make decisions on how to deploy and manage Airflow.

■ Airflow's Security Policy⁴⁸

- ▶ To know how to report security vulnerabilities and how to handle them.

■ Secrets⁴⁹

- ▶ Variables configurations used that contain particularly sensitive information,
- ▶ These variable and connection configurations are kept secret during operation.
- ▶ These follows:
 - Encryption at rest,
 - Using external secret stores, and
 - Masking of sensitive data.

48 More details: <https://github.com/apache/airflow/security/policy>

49 More details: <https://airflow.apache.org/docs/apache-airflow/stable/security/secrets/index.html>

Apache Airflow™ : How to define a DAG?

- 1 Create a DAG definition file.
 - ▶ Import modules⁵⁰ (same as in Python program),
- 2 Default arguments "default_args"⁵¹
 - ▶ A dictionary of default parameters that is used when creating tasks.
- 3 Instantiate a DAG
 - ▶ It is a DAG object to nest tasks into.
- 4 Operators
 - ▶ An operator defines a unit of work for Airflow to complete.
 - ▶ This helps to visualize task dependencies in DAG code.
- 5 Tasks
 - ▶ Tasks determines how to execute the operator's work within a DAG.
 - ▶ A task must be instantiated to use an operator in a DAG.

50 Modules: https://airflow.apache.org/docs/apache-airflow/stable/administration-and-deployment/modules_management.html

51 Arguments: https://airflow.apache.org/docs/apache-airflow/stable/_modules/airflow/example_dags/tutorial.html

Apache Airflow™ : Example of a DAG?

```
1 import textwrap; from datetime import datetime, timedelta; ## Modules
2
3 from airflow.models.dag import DAG ## The DAG object; needed to instantiate a DAG
4
5 from airflow.operators.bash import BashOperator ## Operators; needed this to operate!
6
7 with DAG(
8     "tutorial", ## DAG name
9     # These args get passed on to each operator(Op) \& can be overridden on a per-task basis during Op
10     ↪ initialization
11     default_args={"depends_on_past": False, "email": ["airflow@example.com"], "email_on_failure": False, ...},
12     description="A simple tutorial DAG", schedule=timedelta(days=1), start_date=datetime(2021, 1, 1),
13     ↪ catchup=False,
14     tags=["example"],
15 ) as dag:
16
17     ## t1, t2 and t3 are examples of tasks created by instantiating operators
18     t1 = BashOperator( task_id="print_date", bash_command="date", )
19     t2 = BashOperator( task_id="sleep", depends_on_past=False, bash_command="sleep 5", retries=3, )
20     t3 = BashOperator( task_id="templated", depends_on_past=False, bash_command=templated_command, )
21
22     # Task dependency
23     t1 >> [t2, t3]
```

Apache Airflow™ : Templating with Jinja

```
1 templated_command = textwrap.dedent(  
2     """  
3     {% for i in range(5) %}  
4     echo "{{ ds }}"  
5     echo "{{ macros.ds_add(ds, 7)}}"  
6     {% endfor %}  
7     """  
8 )  
9  
10 t3 = BashOperator(  
11     task_id="templated",  
12     depends_on_past=False,  
13     bash_command=templated_command,  
14 )
```

Apache Airflow™ : Adding documentation

```
1 t1.doc_md = textwrap.dedent(  
2     """\br/>3     #### Task Documentation  
4     You can document your task using the attributes 'doc_md' (markdown),  
5     'doc' (plain text), 'doc_rst', 'doc_json', 'doc_yaml' which gets  
6     rendered in the UI's Task Instance Details page.  
7     ![[img]](http://montcs.bloomu.edu/~bobmon/Semesters/2012-01/491/import%20soul.png)  
8     **Image Credit:** Randall Munroe, [XKCD](https://xkcd.com/license.html)  
9     """)  
10 )  
11  
12 dag.doc_md = __doc__ # providing that you have a docstring at the beginning of the DAG; OR  
13 dag.doc_md = ""  
14 This is a documentation placed anywhere  
15 "" # otherwise, type it like this
```

Extra: Snakemake Workflow

- It is a console based tool designed mainly for HPC clusters.
- Like Apache Airflow, Snakemake is another open sourced workflow management tool.
- Below is a simple example of it, where it needs rule, input and output parameters only.

```
1 rule all:
2   input:
3     "final_output.txt"
4
5 rule process_data:
6   input:
7     "raw_data.txt"
8   output:
9     "processed_data.txt"
10  shell:
11    "python process_data.py {input} > {output}"
```

Extra: Apache Airflow™ Vs Bash Snakemake

A few information about **Apache Airflow Vs Bash Snakemake**⁵² Workflow

1 Workflow Definition

▶ Airflow

- Uses Python code to define workflows,
- Provides more programmatic flexibility.

▶ Snakemake

- Uses a declarative DSL in a Snakefile.
- Designed for HPC clusters.

2 Execution Model

▶ Airflow

- Emphasizes dynamic scheduling, monitoring, and orchestration.

▶ Snakemake

- Focuses on data dependencies and parallelization.

52

source: <https://snakemake.readthedocs.io/en/stable/>

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**)

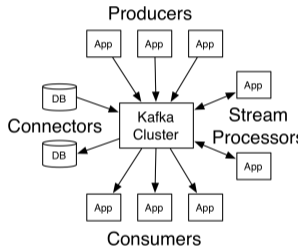


Figure: Source: [42]

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/>

TensorBoard: Result Visualization

TensorBoard SCALARS GRAPHS INACTIVE ↻ ⚙ ?

Show data download links
 Ignore outliers in chart scaling
 Tooltip sorting method: nearest ▼

Smoothing 0,554

Horizontal Axis
 RELATIVE WALL

Runs
 Write a regex to filter runs

logs/

Q.*

Tags matching *.* (all tags) 2

accuracy

0.920
0.900
0.880
0.860
0.840

0.000 4.000k 8.000k 12.00k

loss

0.600
0.400
0.200
0.00

0.000 4.000k 8.000k 12.00k

accuracy 1

accuracy

