The workflow of Oozie and its Benefits:

Apache Oozie is a Java Web application used to schedule Apache Hadoop jobs. Oozie combines multiple jobs sequentially into one logical unit of work. It is integrated with the Hadoop stack, with YARN as its architectural center, and supports Hadoop jobs for Apache MapReduce, Apache Pig, Apache Hive, and Apache Sqoop. Oozie can also schedule jobs specific to a system, like Java programs or shell scripts.Apache Oozie is a tool for Hadoop operations that allows cluster administrators to build complex data transformations out of multiple component tasks. This provides greater control over jobs and also makes it easier to repeat those jobs at predetermined intervals. At its core, Oozie helps administrators derive more value from Hadoop.

There are two basic types of Oozie jobs:

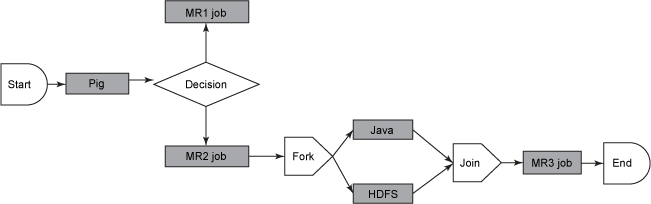
* **Oozie Workflow** jobs are Directed Acyclical Graphs (DAGs), specifying a sequence of actions to execute. The Workflow job has to wait
* **Oozie Coordinator** jobs are recurrent Oozie Workflow jobs that are triggered by time and data availability.

**Oozie Bundle** provides a way to package multiple coordinator and workflow jobs and to manage the lifecycle of those jobs

An Oozie workflow is a collection of actions arranged in a directed acyclic graph (DAG). This graph can contain two types of nodes: control nodes and action nodes. *Control nodes*, which are used to define job chronology, provide the rules for beginning and ending a workflow and control the workflow execution path with possible decision points known as fork and join nodes. *Action nodes* are used to trigger the execution of tasks. In particular, an action node can be a MapReduce job, a Pig application, a file system task, or a Java application. (The shell and ssh actions have been deprecated).

Oozie is a native Hadoop stack integration that supports all types of Hadoop jobs and is integrated with the Hadoop stack. In particular, Oozie is responsible for triggering the workflow actions, while the actual execution of the tasks is done using Hadoop MapReduce. Therefore, Oozie becomes able to leverage existing Hadoop machinery for load balancing, fail-over, etc. Oozie detects completion of tasks through callback and polling. When Oozie starts a task, it provides a unique callback HTTP URL to the task, and notifies that URL when it is complete. If the task fails to invoke the callback URL, Oozie can poll the task for completion. Figure 1 illustrates a sample Oozie workflow that combines six action nodes (Pig script, MapReduce jobs, Java code, and HDFS task) and five control nodes (Start, Decision control, Fork, Join, and End). Oozie workflows can be also parameterized. When submitting a workflow job, values for the parameters must be provided. If the appropriate parameters are used, several identical workflow jobs can occur concurrently.

Figure 1. Sample Oozie workflow



In practice, it is sometimes necessary to run Oozie workflows on regular time intervals, but in coordination with other conditions, such as the availability of specific data or the completion of any other events or tasks. In these situations, Oozie Coordinator jobs allow the user to model workflow execution triggers in the form of the data, time, or event predicates where the workflow job is started after those predicates get satisfied. The Oozie Coordinator can also manage multiple workflows that are dependent on the outcome of subsequent workflows. The outputs of subsequent workflows become the input to the next workflow. This chain is called a *data application pipeline*.

Oozie workflow definition language is XML-based and it is called the *Hadoop Process Definition Language.*Oozie comes with a command-line program for submitting jobs. This command-line program interacts with the Oozie server using REST. To submit or run a job using the Oozie client, give Oozie the full path to your workflow.xml file in HDFS as a parameter to the client. Oozie does not have a notion of global properties. All properties, including the *jobtracker* and the *namenode*, must be submitted as part of every job run. Oozie uses an RDBMS for storing state.

Benefits are

1. Oozie is designed to scale in a Hadoop cluster. Each job will be launched from a different datanode. This means that the workflow load will be balanced and no single machine will become overburdened by launching workflows. This also means that the capacity to launch workflows will grow as the cluster grows.
2. Oozie is well integrated with Hadoop security. Oozie knows which user submitted the job and will launch all actions as that user, with the proper privileges. It will handle all the authentication details for the user as well.
3. Oozie is the only workflow manager with built-in Hadoop actions, making workflow development, maintenance and troubleshooting easier.
4. Oozie UI makes it easier to drill down to specific errors in the data nodes. Other systems would require significantly more work to correlate jobtracker jobs with the workflow actions.
5. Oozie is proven to scale in some of the world’s largest clusters.
6. Oozie gets callbacks from MapReduce jobs so it knows when they finish and whether they hang without expensive polling. No other workflow manager can do this.
7. Oozie Coordinator allows triggering actions when files arrive at HDFS.
8. Oozie is supported by Hadoop vendors. If there is ever an issue with how the workflow manager integrates with Hadoop

**Sqoop :**

**Control Flow nodes:**

Control flow nodes define the beginning and the end of a workflow (the start , end and kill nodes) and provide a mechanism to control the workflow execution path (the decision , fork and join nodes).

The start node is the entry point for a workflow job, it indicates the first workflow node the workflow job must transition to.

When a workflow is started, it automatically transitions to the node specified in the start .

A workflow definition must have one start node.

The end node is the end for a workflow job, it indicates that the workflow job has completed successfully.

When a workflow job reaches the end it finishes successfully (SUCCEEDED).

If one or more actions started by the workflow job are executing when the end node is reached, the actions will be killed. In this scenario the workflow job is still considered as successfully run.

A workflow definition must have one end node.

The kill node allows a workflow job to kill itself.

When a workflow job reaches the kill it finishes in error (KILLED).

If one or more actions started by the workflow job are executing when the kill node is reached, the actions will be killed.

A workflow definition may have zero or more kill nodes.

**Action Nodes:**

Action nodes are the mechanism by which a workflow triggers the execution of a computation/processing task.

All computation/processing tasks triggered by an action node are executed asynchronously by Oozie.

For most types of computation/processing tasks triggered by workflow action, the workflow job has to wait until the computation/processing task completes before transitioning to the following node in the workflow.