Problem Statement:

Explain in brief:

●The complete structure and the working of “Oozie Workflow scheduler “

**Solution:**

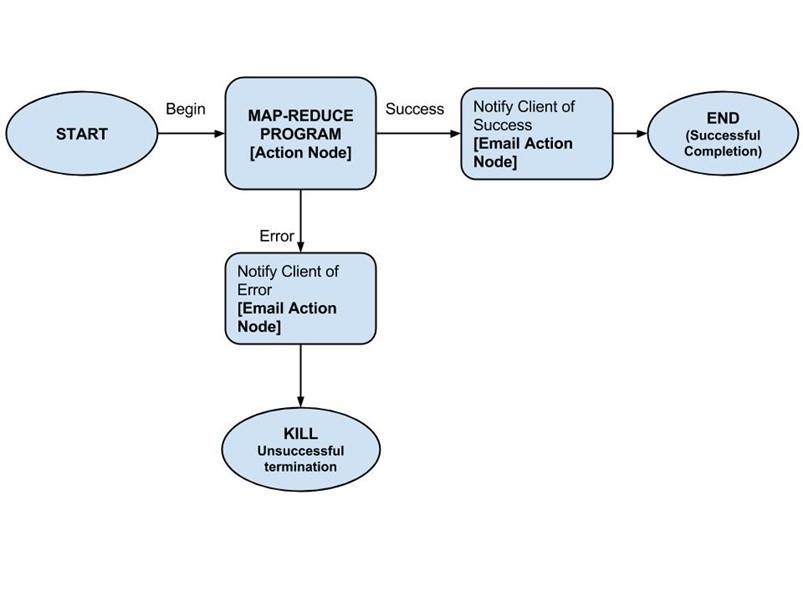
**What is OOZIE?**

Apache Oozie is a workflow scheduler for Hadoop. It is a system which runs workflow of dependent jobs. Here, users are permitted to create **Directed Acyclic Graphs** of workflows, which can be run in parallel and sequentially in Hadoop.

It consists of two parts:

* **Workflow engine**: Responsibility of a workflow engine is to store and run workflows composed of Hadoop jobs e.g., MapReduce, Pig, Hive.
* **Coordinator engine**: It runs workflow jobs based on predefined schedules and availability of data.

Oozie is scalable and can manage timely execution of thousands of workflows (each consisting of dozens of jobs) in a Hadoop cluster.

[](http://cdn.guru99.com/images/Big_Data/061114_1137_LearnOOZIEi1.jpg)

Oozie is very much flexible, as well. One can easily start, stop, suspend and rerun jobs. Oozie makes it very easy to rerun failed workflows. One can easily understand how difficult it can be to catch up missed or failed jobs due to downtime or failure. It is even possible to skip a specific failed node.

**How does OOZIE work?**

Oozie runs as a service in the cluster and clients submit workflow definitions for immediate or later processing.

Oozie workflow consists of **action nodes** and **control-flow nodes**.

An**action node** represents a workflow task, e.g., moving files into HDFS, running a MapReduce, Pig or Hive jobs, importing data using Sqoop or running a shell script of a program written in Java.

A**control-flow node** controls the workflow execution between actions by allowing constructs like conditional logic wherein different branches may be followed depending on the result of earlier action node.

**Start Node**, **End Node** and **Error Node** fall under this category of nodes.

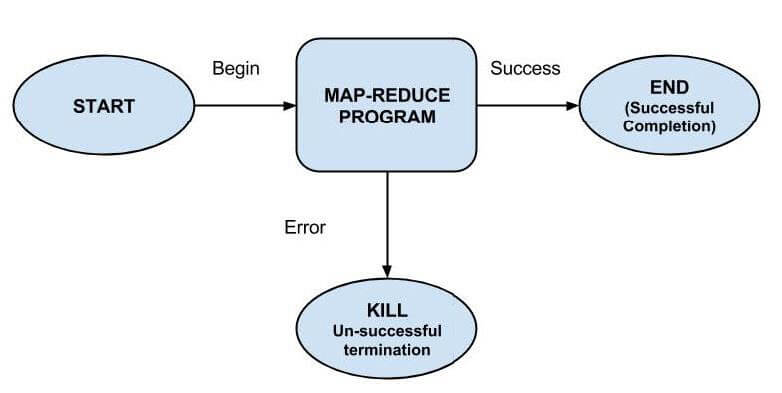
**Start Node,** designates start of the workflow job.

**End Node,** signals end of the job.

**Error Node,** designates an occurrence of error and corresponding error message to be printed.

At the end of execution of workflow, HTTP callback is used by Oozie to update client with the workflow status. Entry-to or exit-from an action node may also trigger callback.

**Example Workflow Diagram**

[](http://cdn.guru99.com/images/pimgpsh_fullsize_distr.jpg)

**Packaging and deploying an Oozie workflow application**

A workflow application consists of the workflow definition and all the associated resources such as MapReduce Jar files, Pig scripts etc. Applications need to follow a simple directory structure and are deployed to HDFS so that Oozie can access them.

An example directory structure is shown below-

/

??? lib/

? ??? hadoop-examples.jar

??? workflow.xml

It is necessary to keep workflow.xml (a workflow definition file) in the top level directory (parent directory with workflow name). Lib directory contains Jar files containing MapReduce classes. Workflow application conforming to this layout can be built with any build tool e.g., Ant or Maven.

Such a build need to be copied to HDFS using command, for example -

% hadoop fs -put hadoop-examples/target/ name of workflow **Steps for Running an Oozie workflow job**

In this section we will see how to run a workflow job. To run this, we will use Oozie command-line tool (a client program which communicates with the Oozie server).

**1. Export OOZIE\_URL environment variable which tells the oozie command which Oozie server to use (here we’re using one running locally):**

% export OOZIE\_URL="http://localhost:11000/oozie"

**2. Run workflow job using-**

% oozie job -config ch05/src/main/resources/max-temp-workflow.properties -run

The -config option refers to a local Java properties file containing definitions for the parameters in the workflow XML file, as well as oozie.wf.application.path, which tells Oozie the location of the workflow application in HDFS.

Example contents of the properties file:

nameNode=hdfs://localhost:8020 jobTracker=localhost:8021 oozie.wf.application.path=${nameNode}/user/${user.name}/

**3. Get the status of workflow job-**

Status of workflow job can be seen using subcommand 'job' with '-info' option and specifying job id after '-info'.

e.g., % oozie job -info

Output shows status which is one of: RUNNING, KILLED, or SUCCEEDED.

**4. Results of successful workflow execution can be seen using Hadoop command like-**

% hadoop fs -cat

**Why use Oozie?**

Main purpose of using Oozie is to manage different type of jobs being processed in Hadoop system.

Dependencies between jobs are specified by user in the form of Directed Acyclic Graphs. Oozie consumes this information and takes care of their execution in correct order as specified in a workflow. That way user's time to manage complete workflow is saved. In addition, Oozie has a provision to specify frequency of execution of a particular job.

**FEATURES OF OOZIE**

* Oozie has client API and command line interface which can be used to launch, control and monitor job from Java application.
* Using its Web Service APIs one can control jobs from anywhere.
* Oozie has provision to execute jobs which are scheduled to run periodically.
* Oozie has provision to send email notifications upon completion of jobs.