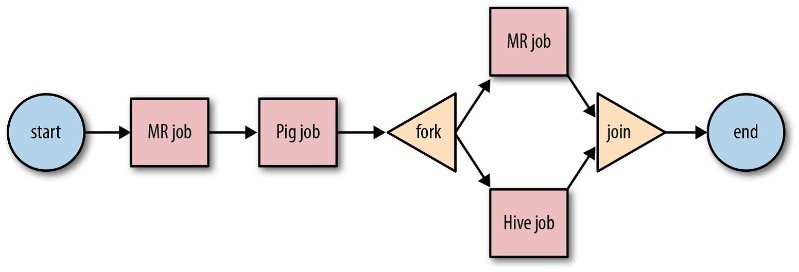
***ASSIGNMENT 34.1***

***The complete structure and the working of “Oozie Workflow scheduler”***

***Apache oozie:***

* *Apache Oozie is an open source project based on Java technology that simplifies the process of creating workflows and managing coordination among jobs.*
* *In principle, Oozie offers the ability to combine multiple jobs sequentially into one logical unit of work.*
* *In addition, it can be used to schedule jobs specific to a system, such as Java programs. Therefore, using Oozie, Hadoop administrators are able to build complex data transformations that can combine the processing of different individual tasks and even sub-workflows.*
* *This ability allows for greater control over complex jobs and makes it easier to repeat those jobs at predetermined periods.*

**

***Types of Oozie Jobs :-*** *Oozie supports job scheduling for the full Hadoop stack like Apache MapReduce, Apache Hive, Apache Sqoop and Apache Pig.*

***1) Periodical/Coordinator Job***

*These are recurrent jobs which run based on a particular time or they can be configured to run when data is available. Coordinator jobs can manage multiple workflow based jobs as well as where the output of one workflow can be the input for another workflow.*

***2) Oozie Hadoop Workflow***

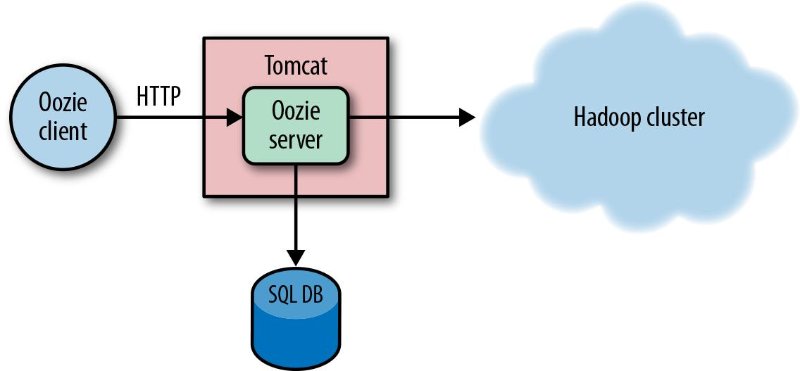
*It is Directed Acyclic Graph (DAG) which consists of collection of actions. The Control nodes decide the chronological order, setting of rules, execution path decision, joining the nodes and fork. Whereas, Action node triggers the execution.*

***3) Oozie Bundle***

*An Oozie bundle is collection of many coordinator jobs which can be started, suspended and stopped periodically.*

***Oozie Architecture :-***

*Oozie Architecture has a Web Server and a database for storing all the jobs. The default web server is Apache Tomcat, which is the open source implementation of Java Servlet Technology. Oozie server is a stateless web application and does not store any information regarding the user and job in-memory. The users or Oozie clients can interact with the server, using either the command line tool, Java Client API or, HTTP REST API.*

**

*This type of design pattern helps Oozie support thousands of jobs with low configuration hardware.*

*Oozie itself has two main components which do all the work, the Command and the ActionExecutor classes.*

***Use Cases for Oozie***

*Yahoo has around 40,000 nodes across multiple Hadoop clusters and Oozie is the primary Hadoop workflow engine. The largest Hadoop cluster at Yahoo processes 60 bundles and 1600 coordinators totaling to 80,000 daily workflows on 3 million workflow nodes.*

***Sample Workflow Of Wordcount Program:***

*<workflow-app xmlns='uri:oozie:workflow:0.1' name='java-main-wf'>  
<start to='mapreduce-wordcount-example' />  
<action name='mapreduce-wordcount-example'>  
<java>  
<job-tracker>${jobTracker}</job-tracker>  
<name-node>${nameNode}</name-node>  
<configuration>  
<property>  
<name>mapred.job.queue.name</name>  
<value>default</value>  
</property>  
</configuration>  
<main-class>org.apache.hadoop.examples.ExampleDriver</main-class>  
<arg>wordcount</arg>  
<arg>${inputDir}</arg>  
<arg>${outputDir}</arg>  
</java>  
<ok to="end" />  
<error to="fail" />  
</action>  
<kill name="fail">  
<message>Java failed, error message[${wf:errorMessage(wf:lastErrorNode())}]</message>  
</kill>  
<end name='end' />  
</workflow-app>*