Explain in brief:

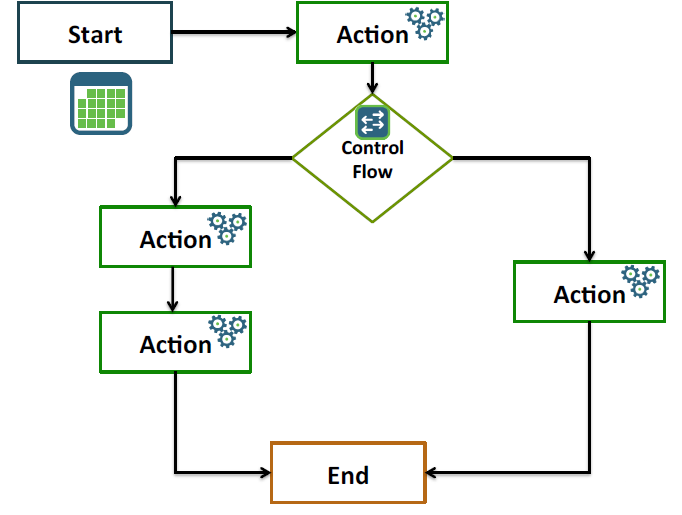
● The complete structure and the working of “Oozie Workflow scheduler “

**Solution:**

* Oozie is an open-source Apache project that provides a framework for coordinating and scheduling Hadoop jobs. Oozie is not restricted to just MapReduce jobs; you can use Oozie to schedule Pig, Hive, Sqoop, Streaming jobs, and even Java programs.
* Oozie is a Java web application that runs in a Tomcat instance.
* Oozie can be run as a service, and then start workflows using the oozie command.
* Oozie has two main capabilities:

1. Oozie Workflow: a collection of actions (defined in a **workflow.xml** file).
   * + - Pig Actions
       - Hive Actions
       - MapReduce Actions
2. Oozie Coordinator: a recurring workflow (defined in a **coordinator.xml** file).
   * + - Schedule a Job Based on Time
       - Schedule a Job Based on Data Availability

Defining an Oozie Workflow:



**Oozie Workflow:**

* An Oozie workflow consists of a workflow.xml file and the necessary files required by the workflow. The workflow is put into HDFS with the following directory structure:
  + - * + /appdir/workflow.xml
        + /appdir/config-default.xml
        + /appdir/lib/files.jar
        + The workflow.xml, a workflow definition consists of two main entries:
    - Control flow nodes: for determining the execution path.
    - Action nodes: for executing a job or task.
* The config-default.xml file is optional and contains properties shared by all workflows.
* Each workflow can also have a job.properties file (not put into HDFS) for job-specific properties.

**PIG ACTION:**

<workflow-app xmlns=*"uri:oozie:workflow:0.2" name=“transform-workflow">*

<start to=*"transform\_data" />*

<action name=*" transform\_data">*

<pig>

<job-tracker>${resourceManager}</job-tracker>

<name-node>${nameNode}</name-node>

<prepare>

<delete path=*“old\_data\_path" />*

</prepare>

<script>transform.pig</script>

</pig>

<ok to=*"end" />*

<error to=*"fail" />*

</action>

<kill name=*"fail">*

<message>Job failed, error message[${wf:errorMessage(wf:lastErrorNode())}]</message>

</kill>

<end name=*"end" />*

</workflow-app>

**Hive Action:**

<action name=*"find\_count\_visit">*

<hive xmlns=*"uri:oozie:hive-action:0.5">*

<job-tracker>${resourceManager}</job-tracker>

<name-node>${nameNode}</name-node>

<prepare>

<delete path=*“old\_data\_path" />*

</prepare>

<job-xml>hive-site.xml</job-xml>

<configuration>

<property>

<name>mapreduce.map.output.compress</name>

<value>true</value>

</property>

</configuration>

<script>findcount.hive</script>

</hive>

<ok to=*"end" />*

<error to=*"fail" />*

</action>

**Oozie Coordinator:**

Oozie Coordinator is a collection of predicates (conditional statements based on timefrequency and data availability) and actions (i.e. Hadoop Map/Reduce jobs, Hadoop file system, Hadoop Streaming, Pig, Java and Oozie sub-workflow).

Actions are recurrent workflow jobs invoked each time predicate returns true.

Coordinator applications allow users to schedule complex workflows, including workflows that are scheduled regularly.

Oozie Coordinator models the workflow execution triggers in the form of time, data or event predicates. The workflow job mentioned inside the Coordinator is started only after the given conditions are 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**.**

**Schedule a Job Based on Time**

<coordinator-app name=*"tf-idf" frequency="1440"*

start=*"2013-01-01T00:00Z" end="2013-12-31T00:00Z" timezone="UTC"*

xmlns=*"uri:oozie:coordinator:0.1">*

<action>

<workflow>

<app-path> hdfs://node:8020/home/train/tfidf/workflow

</app-path>

</workflow>

</action>

</coordinator-app>

**Schedule a Job Based on Data Availability**

<coordinator-app name=*"file\_check" frequency="1440"*

start=*"2012-01-01T00:00Z" end="2015-12-31T00:00Z" timezone="UTC"*

xmlns=*"uri:oozie:coordinator:0.1">*

<datasets>

<dataset name=*"input1">*

<uri-template>

hdfs://node:8020/job/result/

</uri-template>

</dataset>

</datasets>

<action>

<workflow>

<app-path>hdfs://node:8020/myapp/</app-path>

</workflow>

</action>

</coordinator-app>