Oozie

Definition

- Oozie is a server based Workflow Engine specialized in running workflow jobs with actions that run Hadoop Map/Reduce, Pig, Hive and other jobs on a Cluster.
- The workflow engine has options to schedule jobs (via the Coordinator), notify users etc…
- Job details are defined in an XML, called as Workflow XML.
- Oozie is a Java Web-Application that runs in a Java servlet-container.

Job Definition

- Oozie jobs are defined using Workflows
- A workflow is a collection of actions (i.e. Hadoop Map/Reduce jobs, Pig jobs etc…) arranged in a control dependency DAG (Direct Acyclic Graph). "control dependency" from one action to another means that the second action can't run until the first action has completed.
- Oozie workflows definitions are written in hPDL, an XML vocabulary which models typical workflows.
- Oozie workflows contain control flow nodes and action nodes.

Workflow nodes

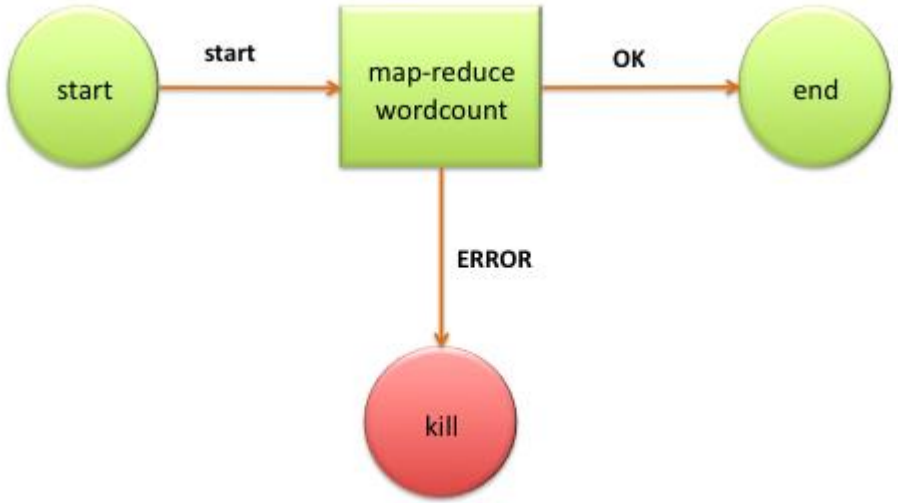
- Control flow nodes define the beginning and the end of a workflow (start , end and fail nodes) and provide a mechanism to control the workflow execution path (decision , fork and join nodes).

- Action nodes are the mechanism by which a workflow triggers the execution of a computation/processing task. Oozie provides support for different types of actions: Hadoop map-reduce, Hadoop file system, Pig, Hive, SSH, HTTP, eMail and Oozie sub-workflow.
- Oozie can be extended to support additional type of actions.

Workflow

- Oozie workflows can be parameterized (using variables like ${inputDir} within the workflow definition).
- When submitting a workflow job values for the parameters must be provided.

A Sample workflow




```
<workflow-app name='wordcount-wf' xmlns="uri:oozie:workflow:0.1">
    <start to='wordcount'/>
    <action name='wordcount'>
        <map-reduce>
            <job-tracker>${jobTracker}</job-tracker>
            <name-node>${nameNode}</name-node>
            <configuration>
                <property>
                    <name>mapred.mapper.class</name>
                    <value>org.myorg.WordCount.Map</value>
                </property>
                <property>
                    <name>mapred.reducer.class</name>
                    <value>org.myorg.WordCount.Reduce</value>
                </property>
                <property>
                    <name>mapred.input.dir</name>
                    <value>${inputDir}</value>
                </property>
                <property>
                    <name>mapred.output.dir</name>
                    <value>${outputDir}</value>
                </property>
            </configuration>
        </map-reduce>
        <ok to='end'/>
        <error to='kill'/>
    </action>
    <kill name='kill'>
        <message>Something went wrong: ${wf:errorCode('wordcount')}</message>
    </kill/>
    <end name='end'/>
</workflow-app>
```

Building Oozie

- Download from http://www-eu.apache.org/dist/oozie/4.2.0/oozie-4.2.0.tar.gz
- tar xvfz oozie-4.2.0.tar.gz
- mv oozie-4.2.0 ooziesrc
- cd ooziesrc
- Build using maven
  - mvn clean package assembly:single -Puber -Phadoop-2 -DskipTests
- cd ..; mkdir oozie; cd oozie
- cp -R ../ooziesrc/distro/target/oozie-4.2.0-distro/oozie-4.2.0/* .

Configuring Oozie

- Pickup a prebuilt distro (or) build one
- mkdir libext
- wget -P libext http://extjs.com/deploy/ext-2.2.zip
- ./bin/oozie-setup.sh prepare-war

Start from here if you are using a prebuilt distro

- Add the following to core-site.xml of your Hadoop cluster (replace proxyuser.* with more specific user name, such as proxyuser.hdtester etc…

```
<property>
<name>hadoop.proxyuser.*.hosts</name>
<value>*</value>
</property>
<property>
<name>hadoop.proxyuser.*.groups</name>
<value>*</value>
</property>
```

- ./bin/ooziedb.sh create -sqlfile oozie.sql –run
- --start hdfs and yarn (incl history server)
- Modify conf/oozie-site.xml and include the below config

```
<property>
      <name>oozie.service.HadoopAccessorService.hadoop.configurations</name>
      <value>*=/home/hdtester/hadoop/etc/hadoop/</value>
</property>
```

- ./bin/oozie-setup.sh sharelib create -fs hdfs://<namenodeendpoint>

- ./bin/oozied.sh start
- Verify by Navigating to http://localhost:11000/oozie/

Running a sample

- Modify examples/apps/map-reduce/job.properties to suite your cluster needs
- Upload the examples dir of oozie to HDFS hdfs dfs -put examples examples

- bin/oozie job -oozie http://localhost:11000/oozie/ -config examples/apps/map-reduce/job.properties -run

Turn on to CDH

- You have oozie preinstalled, but it runs on a user named oozie
- The user that you use to login and to create data in HDFS is cloudera
- Ensure you have permissions set properly so that jobs that are run as oozie user can read your files and write to the output directory.
- Run the pig workflow for phones data processing