Apache Oozie

BAS Academy

Agenda

- About Oozie
- Control Flow Nodes
- Action Nodes
- Functions and Execution
- Hands On

About Oozie

What is Oozie

- Oozie is an open-source Apache project
- Provides a framework for coordinating and scheduling Hadoop jobs
- Oozie can be used to schedule MapReduce jobs, Pig, Hive, Sqoop, Streaming jobs, and even Java programs
- Oozie is a Java web application that runs in a Tomcat instance.

Oozie has two main capabilities:

- **Oozie Workflow**

A collection of actions (defined in a workflow.xml file)

- **Oozie Coordinator**

A recurring workflow (defined in a coordinator.xml file)

Oozie Workflow

- A workflow is a collection of actions jobs 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 workflow actions start jobs in remote systems (i.e. Hadoop, Pig).
- Upon action completion, the remote systems callback Oozie to notify the action completion, at this point Oozie proceeds to the next action in the workflow.

Oozie workflows contain

- Control Flow Nodes
- Action Nodes

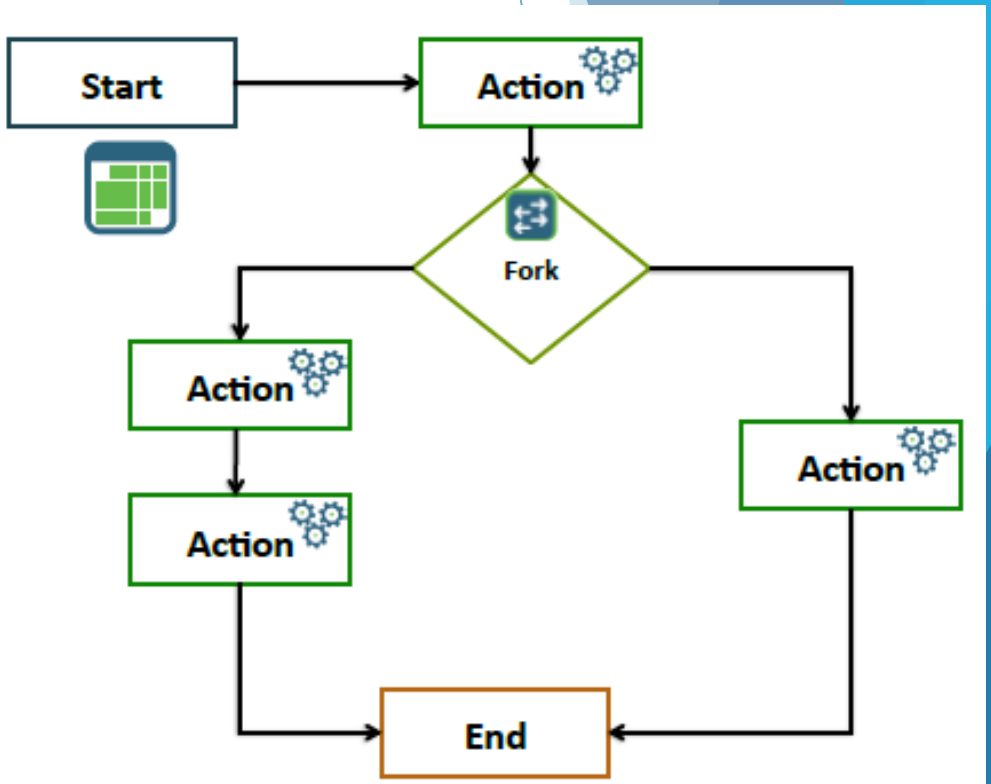
Workflow Nodes

Control Flow Nodes:

- To determine execution path
- Define the beginning and the end of a workflow (start , end and fail nodes)
- Provide a mechanism to control the workflow execution path (decision, fork and join nodes)

Action Nodes:

- For executing a job or task
- Oozie provides support for different types of actions: Hadoop map-reduce, Hadoop file system, Pig etc.,



Control Flow Nodes

Control Flow Nodes

- Start Control Node:

Entry point for a workflow job, it indicates the first workflow node the workflow job

```
<workflow-app name="[WF-DEF-NAME]" xmlns="uri:oozie:workflow:0.1">
  ...
  <start to="[NODE-NAME]"/>
  ...
</workflow-app>
```

- End Control Node:

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

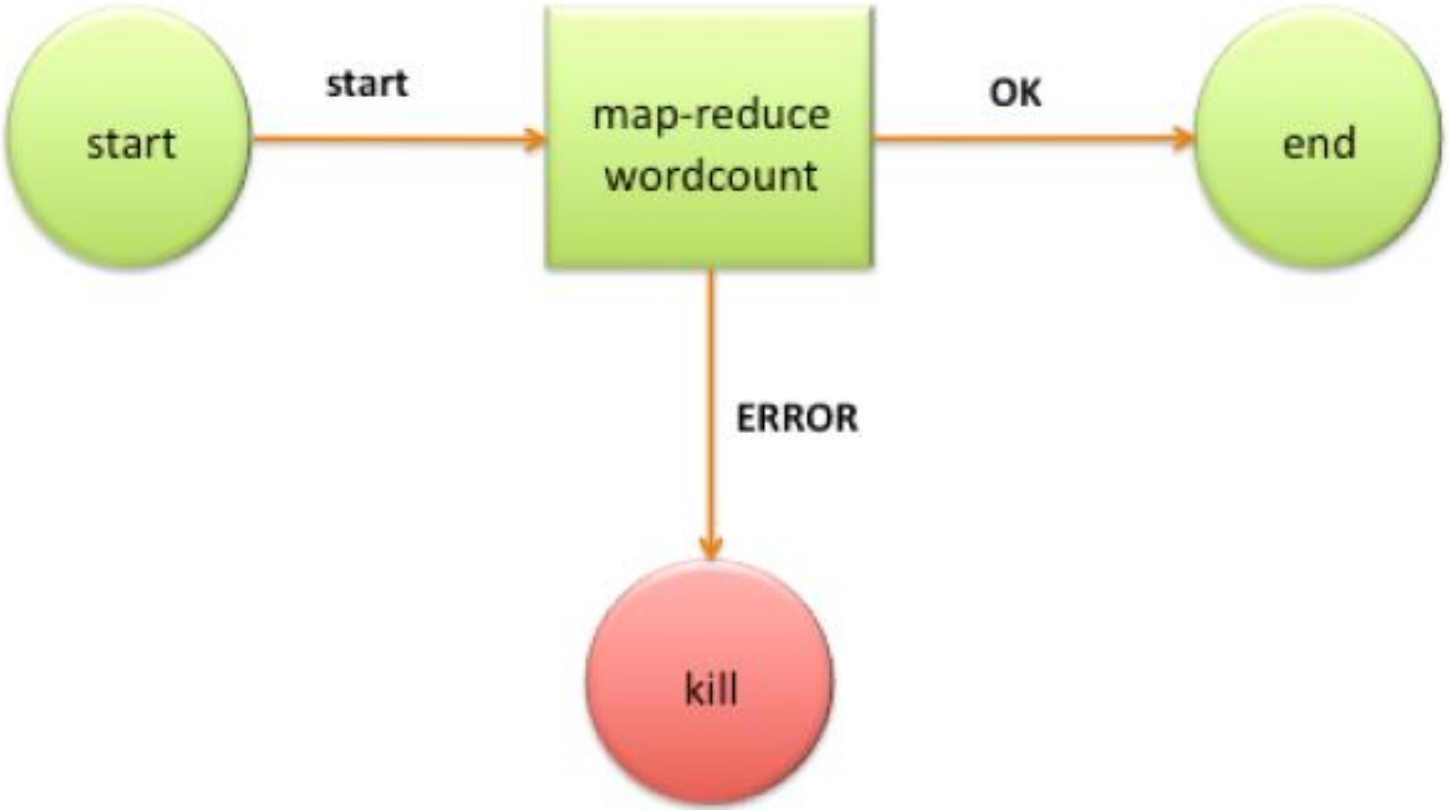
```
<workflow-app name="[WF-DEF-NAME]" xmlns="uri:oozie:workflow:0.1">
    ...
    <end name="[NODE-NAME]"/>
    ...
</workflow-app>
```

- Kill Control Node:

The kill node allows a workflow job to kill itself. (On Error)

```
<workflow-app name="[WF-DEF-NAME]" xmlns="uri:oozie:workflow:0.1">
    ...
    <kill name="[NODE-NAME]">
        <message>[MESSAGE-TO-LOG]</message>
    </kill>
    ...
</workflow-app>
```

Example - DAG



Control Flow Node – Decision Control

- Decision Control Node:

A decision node enables a workflow to make a selection on the execution path to follow. Ex: Switch Case statement

```
<workflow-app name="[WF-DEF-NAME]" xmlns="uri:oozie:workflow:0.1">
    ...
    <decision name="[NODE-NAME]">
        <switch>
            <case to="[NODE_NAME]">[PREDICATE]</case>
            ...
            <case to="[NODE_NAME]">[PREDICATE]</case>
            <default to="[NODE_NAME]"/>
        </switch>
    </decision>
    ...
</workflow-app>
```

Fork and Join Node

- Fork Node

A fork node splits one path of execution into multiple concurrent paths of execution.

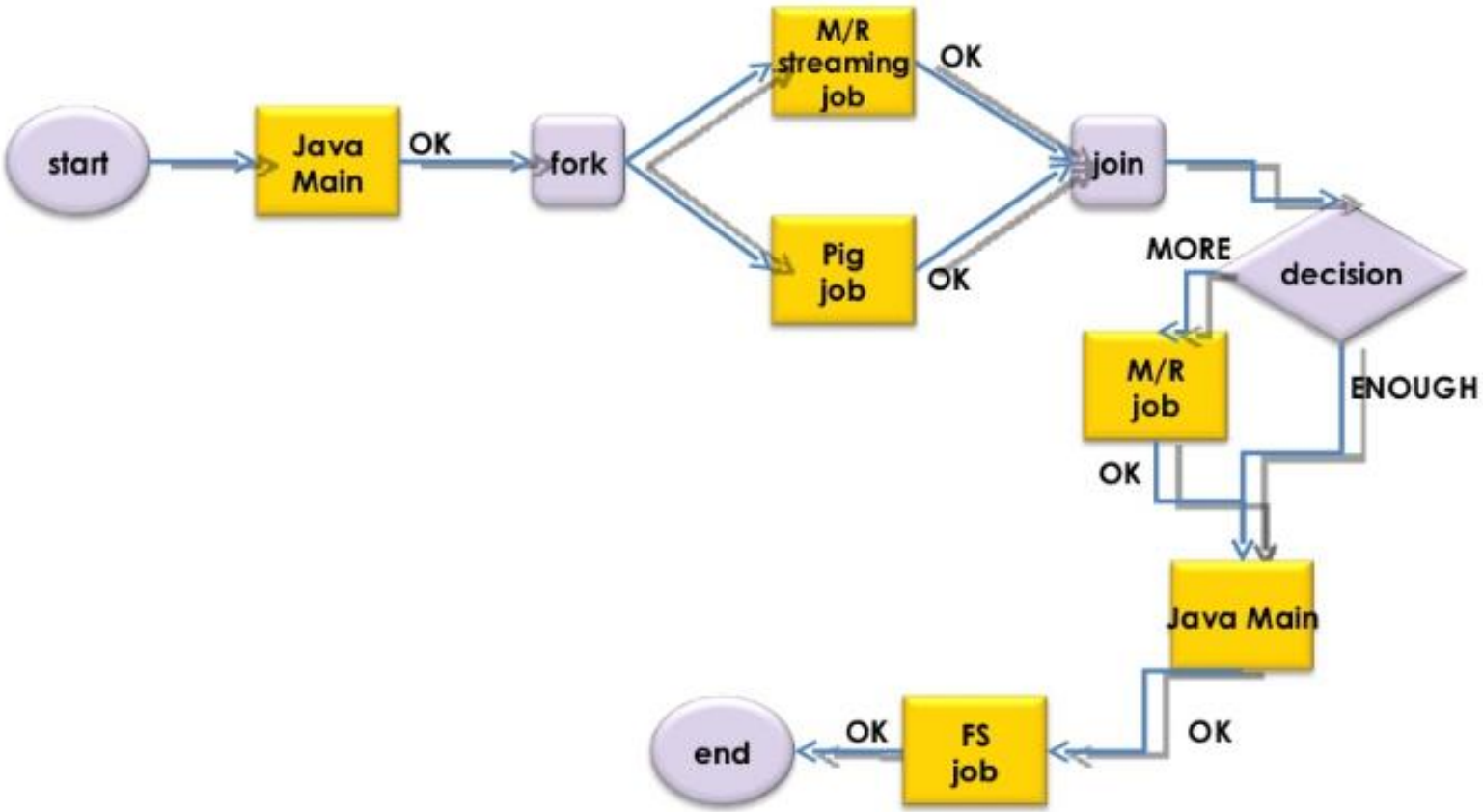
- Join Node

A join node waits until every concurrent execution path of a previous fork node arrives to it.

- The fork and join nodes must be used in pairs.
- The join node assumes concurrent execution paths are children of the same fork node.

```
<workflow-app name="[WF-DEF-NAME]" xmlns="uri:oozie:workflow:0.1">
    ...
    <fork name="[FORK-NODE-NAME]">
        <path start="[NODE-NAME]" />
        ...
        <path start="[NODE-NAME]" />
    </fork>
    ...
    <join name="[JOIN-NODE-NAME]" to="[NODE-NAME]" />
    ...
</workflow-app>
```

Example – DAG (Decision and Fork)



Action Nodes

Action Node - Recovery

- All computation/processing tasks triggered by an action node are remote to Oozie.
- Depending on the nature of the failure, Oozie will have different recovery strategies
- If the failure is of transient nature, Oozie will perform retries after a pre-defined time interval. Ex: network problems or a remote system temporary unavailable
- The number of retries and timer interval for a type of action must be pre-configured at Oozie level.
- If the failure is of non-transient nature, Oozie will suspend the workflow job until an manual or programmatic \ intervention resumes the workflow job

Action – MapReduce and Pig

Map-Reduce Action:

- The map-reduce action starts a Hadoop map/reduce job from a workflow.

Pig Action:

- The pig action starts a Pig job.
- The workflow job will wait until the pig job completes before continuing to the next action.
- The pig action has to be configured with the job-tracker, name-node, pig script and the necessary parameters and configuration to run the Pig job.
- Map Reduce and pig action can be configured to perform HDFS files/directories cleanup before starting the Pig job.
- This capability enables Oozie to retry a Pig/MR job in the situation of a transient failure (Pig creates temporary directories for intermediate data, thus a retry without cleanup would fail).

Action - Fs (HDFS)

Fs (HDFS) action:

- The fs action allows to manipulate files and directories in HDFS from a workflow application.
- The supported commands are move , delete and mkdir.
- The FS commands are executed synchronously from within the FS action, the workflow job will wait until the specified file commands are completed before continuing to the next action.

```
<workflow-app name="[WF-DEF-NAME]" xmlns="uri:oozie:workflow:0.1">
    ...
    <action name="[NODE-NAME]">
        <fs>
            <delete path='[PATH]'/>
            ...
            <mkdir path='[PATH]'/>
            ...
            <move source='[SOURCE-PATH]' target='[TARGET-PATH]'/>
            ...
            <chmod path='[PATH]' permissions='[PERMISSIONS]' dir-files='false' />
            ...
        </fs>
        <ok to="[NODE-NAME]"/>
        <error to="[NODE-NAME]"/>
    </action>
    ...
</workflow-app>
```

- <ok> tag determines where the flow should go if the job completes successfully. The
- <error> tag defines where to go if the job fails

Other Actions

Ssh Action:

- The ssh action starts a shell command on a remote machine as a remote secure shell in background.

Java Action:

- The java action will execute the public static void main(String[] args) method of the specified main Java class.

Hive Action:

- The hive action runs a Hive job
- The hive-site.xml file needs to be packaged in the workflow and needs to contain the various properties for connecting to Hive

Functions and Execution

Expression Language Functions

The Oozie framework provides set of built in EL functions

- wf:user(), which returns the name of the user running the job
- wf:lastErrorNode(), which returns the DataNode where the most recent error occurred
- wf:name(), which returns the workflow application name for the current workflow job.
- wf:appPath(), which returns the workflow application path for the current workflow job.
- wf:conf(String name), which returns the value of the workflow job configuration property for the current workflow job, or an empty string if undefined.

Deploying Workflow Application

- A workflow application is made up of the workflow definition(xml file) and all the associated resources (such as MapReduce JAR files, Pig scripts, and so on) needed to run it.

- Possible states for a workflow jobs are: PREP , RUNNING , SUSPENDED , SUCCEEDED , KILLED and FAILED .

- Each workflow can also have a job.properties file for job-specific properties
- It can be placed outside the HDFS Cluster
- job.properties contains the properties passed in to the workflow

Job.Properties

Here is an example of a `job.properties` file:

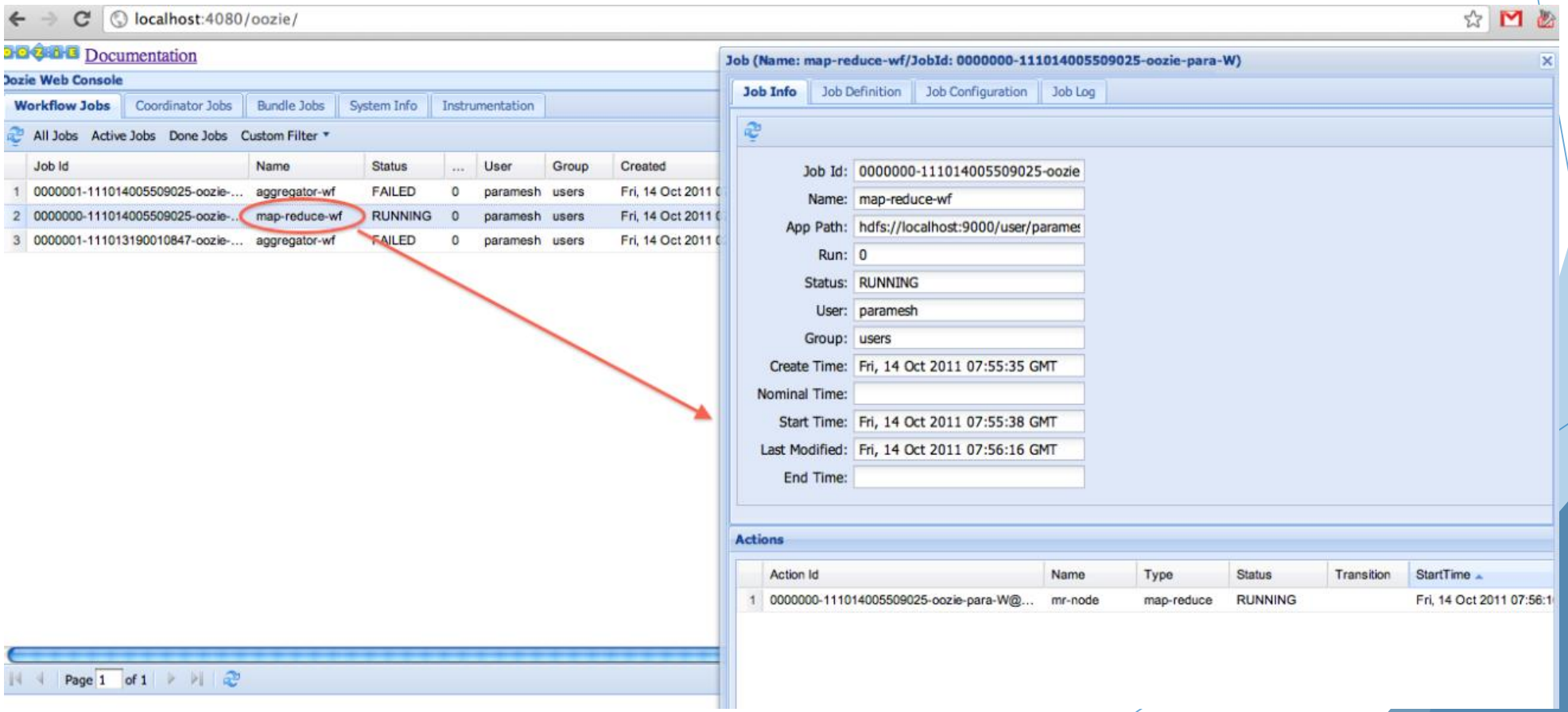
```
oozie.wf.application.path=hdfs://node:8020/path/to/app

#Hadoop ResourceManager
resourceManager=node:8050

#Hadoop fs.default.name
nameNode=hdfs://node:8020/

#Hadoop mapred.queue.name
queueName=default

#Application-specific properties
taxCode=2012
```

- The resourceManager property is for the <job-tracker> value.
- The nameNode property for the <name-node> value
- The queueName property as the value of mapreduce.job.queuename in workflow.xml.

Oozie URL

- To perform job and admin tasks. It is a web console.
- It expects **-oozie OOZIE_URL** option indicating the URL of the Oozie system

Ex: oozie job -oozie http://localhost:8080/oozie -config job.properties -run



Oozie Coordinator

- Oozie Coordinator is a component of Oozie that allows you to define jobs that are recurring Oozie workflows.

These recurring jobs can be triggered by two types of events:

- **Time**
- **Data Availability -** The job triggers when a specified directory is created

```
<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>
```

Hands On

Thank You

Keerthiga Barathan