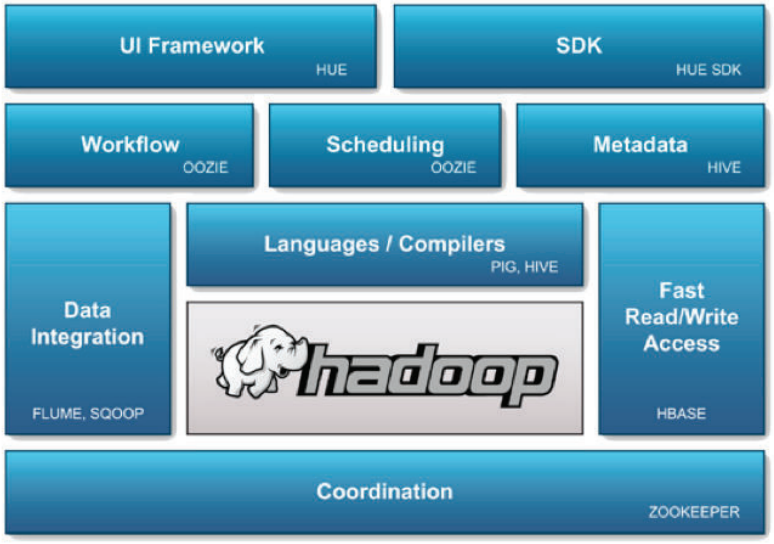
What is Oozie?

- Oozie is a workflow scheduler for Hadoop
- Originally, designed at Yahoo! for their complex search engine workflows
- Now it is an open-source Apache incubator project



Apache Hadoop - Reliable, scalable distributed storage and computing

Apache Hive - SQL-like language and metadata repository

Apache Pig - High-level language for expressing data analysis programs

Apache HBase - Hadoop database for random, real-time read/write access

Apache Zookeeper - Highly-reliable distributed coordination service

Apache Whirr - Library for running Hadoop in the cloud

Apache Flume - Distributed service for collecting and aggregating log and event data

Apache Sqoop - Integrating Hadoop with RDBMS

Hue - Browser-based desktop interface for interacting with Hadoop

Oozie - Server-based workflow engine for Hadoop activities

What is Oozie?

- Oozie allows a user to create Directed Acyclic Graphs of workflows and these can be ran in parallel and sequential in Hadoop
- Oozie can also run plain java classes, Pig workflows, and interact with the HDFS
  - Nice if you need to delete or move files before a job runs
- Oozie can run job's sequentially (one after the other) and in parallel (multiple at a time)

Why use Oozie instead of just cascading a jobs one after another?

- Major flexibility
  - Start, Stop, Suspend, and re-run jobs
- Oozie allows you to restart from a failure
  - You can tell Oozie to restart a job from a specific node in the graph or to skip specific failed nodes

Other Features

- Java Client API / Command Line Interface
  - Launch, control, and monitor jobs from your Java Apps
- Web Service API
  - You can control jobs from anywhere
- Run Periodic jobs
  - Have jobs that you need to run every hour, day, week? Have Oozie run the jobs for you
- Receive an email when a job is complete

How do you make a workflow?

- First make a Hadoop job and make sure that it works using the jar command in Hadoop
  - This ensures that the configuration is correct for your job
- Make a jar out of your classes
- Then make a workflow.xml file and copy all of the job configuration properties into the xml file. These include:
  - Input files
  - Output files
  - Input readers and writers
  - Mappers and reducers
  - Job specific arguments

How do you make a workflow?

- You also need a job.properties file. This file defines the Name node, Job tracker, etc.
- It also gives the location of the shared jars and other files
- When you have these files ready, you need to copy them into the HDFS and then you can run them from the command line

Oozie Start, End, Error Nodes

- Start Node
  - Tells the application where to start

    <start to="[NODE-NAME]" />
- End Node
  - Signals the end of a Oozie Job

    <end name="[NODE-NAME]"/>
- Error Node
  - Signals that an error occurred and a message describing the error should be printed out

    <error name="[NODE-NAME]" />

    <message>"[A custom message]"</message>

    </error>

Oozie Action Node

- Action Nodes
  - Action Nodes specify the Map/Reduce, Pig, or java class to run
  - All Nodes have ok and error tags
    - Ok transitions to the next node
    - Error goes to the error node and should print an error message

```
<action name="[NODE-NAME]">
    <ok to="[NODE-NAME]" />
    <error to="[NODE-NAME]" />
</action>
```

Oozie Map-Reduce Node

- Map/Reduce tags
  - Action tag used to run a map/reduce process. You need to supply the job tracker, name node, and your Hadoop job configuration details

```
<action name="[NODE-NAME]">
<map-reduce>
    <job-tracker>[JOB-TRACKER ADDRESS]</job-tracker>
    <name-node>[NAME-NODE ADDRESS]</name-node>
    <configuration>
        [YOUR HADOOP CONFIGURATION]
    </configuration>
</map-reduce>
    <ok to="[NODE-NAME]" />
    <error to="[NODE-NAME]" />
</action>
```

Oozie Java Job Tag

- Java Job tags
  - Runs the main() function of a java class

```
<action name="[NODE-NAME]">
<java>
        <job-tracker>[JOB-TRACKER ADDRESS]</job-tracker>
        <name-node>[NAME-NODE ADDRESS]</name-node>
        <configuration>
              [OTHER HADOOP CONFIGURATION ITEMS]
        </configuration>
        <main-class>[MAIN-CLASS PATH]</main-class>
        <java-opts>[ANY –D JAVA ARGUMENTS]</java-opts>
        <arg>[COMMAND LINE ARGUMENTS]</arg>
</java>
        <ok to="[NODE-NAME]" />
        <error to="[NODE-NAME]" />
</action>
```

Oozie File System Tag

- Fs tag
  - Interact with the HDFS

```
<action name="[NODE-NAME]">
    <fs>
        <delete path='[PATH]'/>
        <mkdir path='[PATH]'/>
        <move source='[PATH]' target='[PATH]'/>
        <chmod path='[PATH]' permissions='[PERMISSIONS]' dir-file='false/
    true' />
    </fs>
    <ok to="[NODE-NAME]" />
    <error to="[NODE-NAME]" />
</action>
```

Oozie Sub workflow tag

- Sub-workflow
  - Most likely the most important node in Oozie
  - Allows you to run a sub-workflow (another separate workflow) in a job. Good for specific jobs that you will run all the time or long chains of jobs

```
<action name="[NODE-NAME]">
<sub-workflow>
        <app-path>[CHILD-WORKFLOW-PATH]</app-path>
        <configuration>
                [Propagated configuration]
        </configuration>
</sub-workflow>
        <ok to="[NODE-NAME]" />
        <error to="[NODE-NAME]" />
</action>
```

Oozie Fork/Join Node

- Parallel Fork/Join Nodes
- Fork – Starts the parallel jobs

```
<fork>
    <path start="firstjob">
    [OTHER JOBS]
</fork>
```

- Join – Parallel jobs re-join at this node. All forked jobs must be completed to continue the workflow

```
<join name="[NAME JOBS]" to="[NEXT-NODE]"/>
```

Oozie Decision Nodes

- Need to make a decision?
- Decision nodes are a switch statements that will run different jobs based on the outcome of an expression

```
<decision name="[NODE-NAME]" >
    <switch>
     <case to="singlethreadedJob">
          ${fs:fileSize(lastJob) lt 1 *GB}
     </case>
     <case to="MRJob">
          ${fs:fileSize(lastJob) ge 1 *GB}
     </case>
    </switch>
</decision>
```

- Other decision points include Hadoop counters, HDFS operations, string manipulations

Oozie Parameterization

- Parameterization helps make flexible code
- Items like job-trackers, name-nodes, input paths, output table, table names, other constants should be in the job.properties file
- If you want to use a parameter just put it in a ${}

Oozie Parameterization Example

```
<action name="[NODE-NAME]">
<map-reduce>
        <job-tracker>${JOBTRACKER}</job-tracker>
        <name-node>${NAMENODE}</name-node>
        <configuration>
                        <property>
                                <name>mapred.input.dir</name>
                                <value>${INPUTDIR}</value>
                        </property>
                        <property>
                                <name>mapred.output.dir</name>
                                <value>${OUTPUTDIR}</value>
                        </property>
                [OTHER HADOOP CONFIGURATION PARAMETERS]
        </configuration>
</map-reduce>
        <ok to="[NODE-NAME]" />
        <error to="[NODE-NAME]" />
</action>
```

Re-Running a Failed Job

- Hadoop job failures happen
- But, re-running or finishing your job is easy
  - Two ways of accomplishing this:
    - Skip Nodes (oozie.wf.rerun.skip.nodes)
      - A comma separated list of nodes to skip in the next run
    - ReRun Failed Nodes (oozie.wf.rerun.failnodes)
      - Re Run the job from the failed node (true / false)
  - Only one of these can be defined at a time
    - You can only skip nodes or re run nodes, you can't do both
  - Define these in your job.properties file when you re-run your job

What haven't we talked about?

- Coordinator (Periodic) jobs
- SLA monitoring
- Custom Bundled Actions
- Integrating PIG
- Local Oozie Runner
  - Test a workflow locally to ensure that it runs

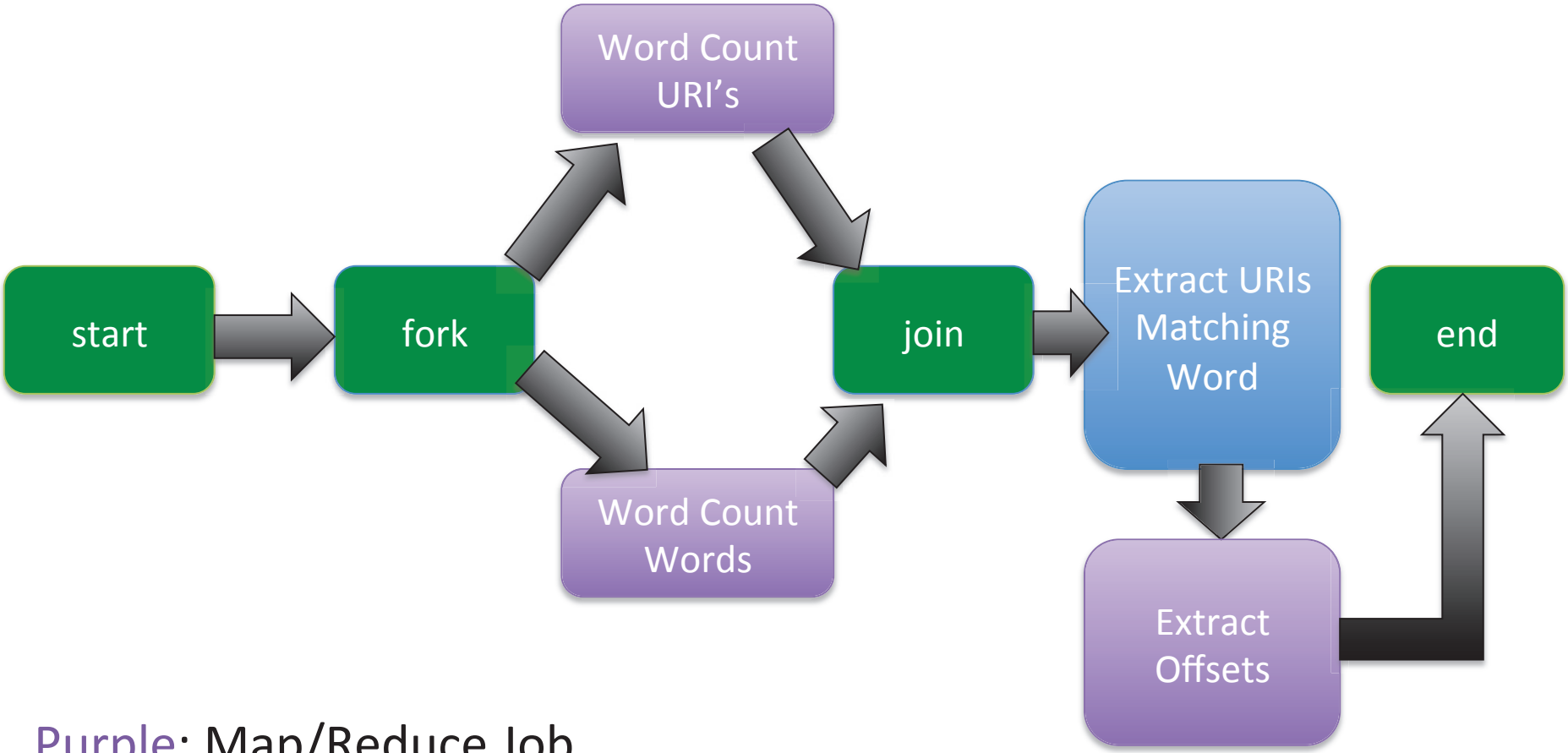
Notes

- All workflow items will start up a Map/Reduce Job.
  - Includes file system manipulations and running Java main classes
- If your jobs have a preparation phase, you need to separate the preparation phase from the execution of the job
- If a job fails, you can re-run it or start it back up from where the job fails

Notes

- Oozie still thinks that you are using the old Hadoop JobConf object. We should not be using this object as it is depreciated. To fix this, two properties can be added to force Oozie to use the new configuration object
- Sometimes you may see more jobs start then are listed in your workflow.xml file. This is fine, there may be some prep work that Oozie is running before a job runs

Example Workflow



Purple: Map/Reduce Job

Blue: Java Job

Want to find out more?

- http://yahoo.github.com/oozie/
- http://yahoo.github.com/oozie/releases/3.1.0/
- http://incubator.apache.org/oozie/