
hadoop
OOZIE

APACHE OOZIE

Agenda

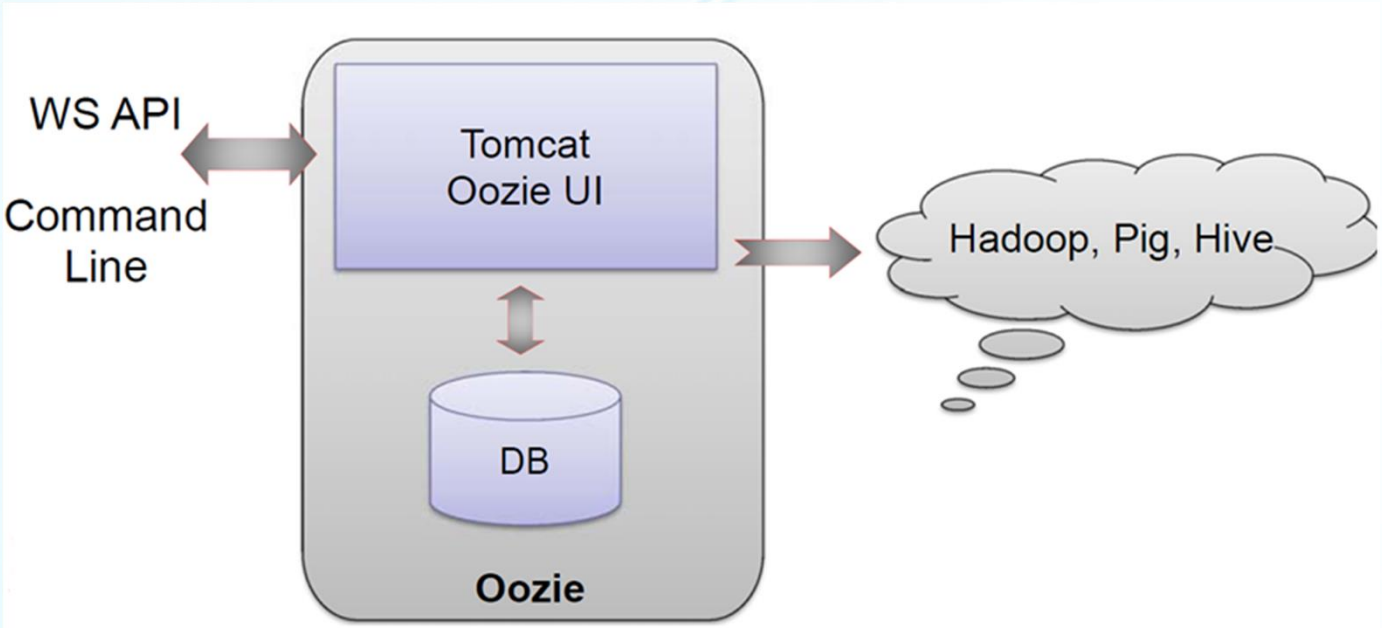
- ✓ What is Oozie?
- ✓ Oozie Workflow
- ✓ Workflow & Property files
- ✓ Running Oozie Workflows
- ✓ Action Nodes & Control Nodes
- ✓ Oozie Coordinators
- ✓ Oozie Bundles

What is Oozie ?

Oozie is a server based Workflow Engine specialized in running workflow jobs with actions that run Hadoop Map/Reduce and Pig jobs.

- Workflow Engine
- Java Web App that runs in a Java servlet-container
- Workflows definitions are written in hPDL
- Oozie workflows contain control flow nodes and action nodes.

What is Oozie ?



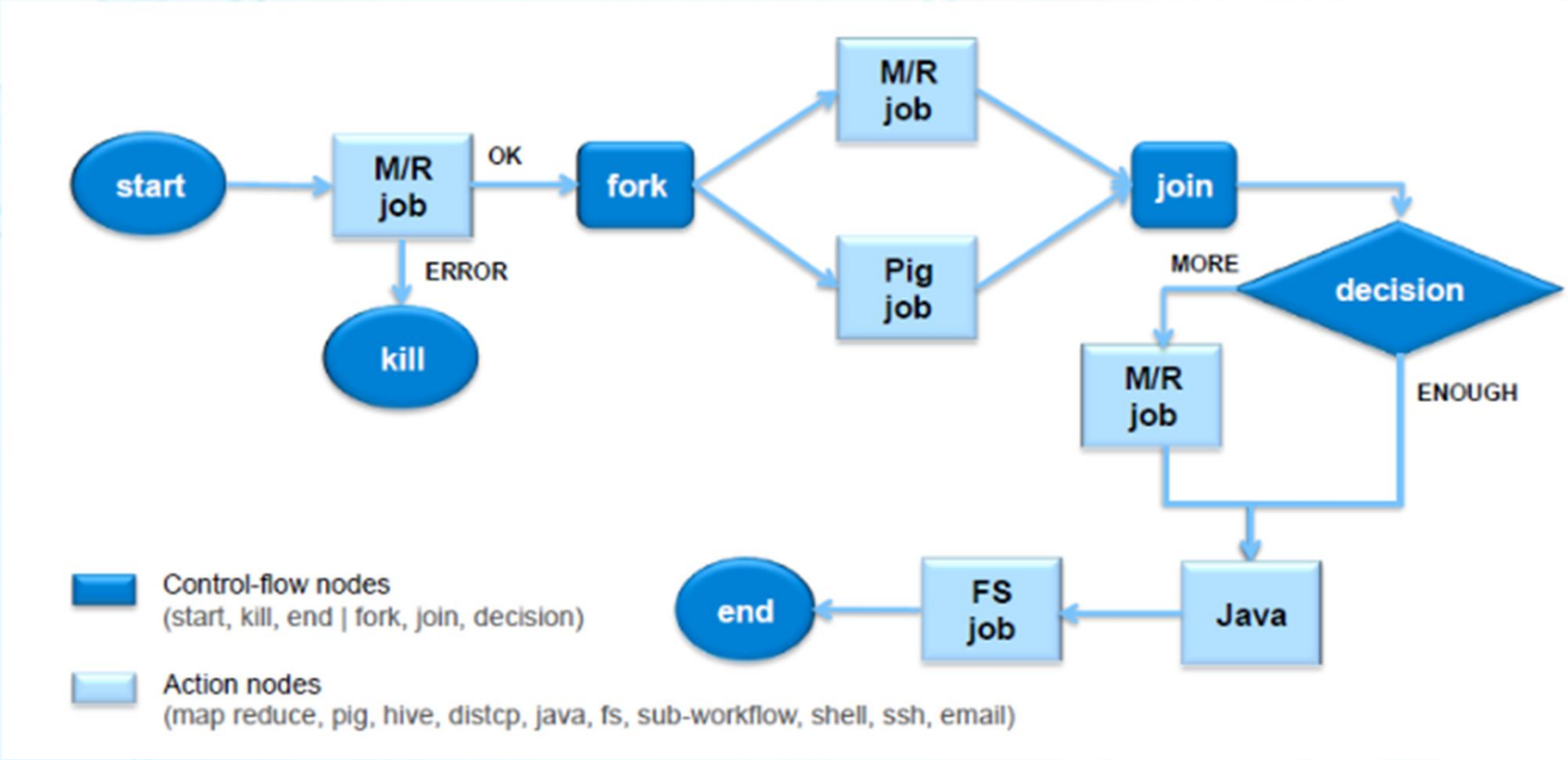
What can we do with Oozie ?

- Oozie Workflow Jobs — DAG of actions
- Oozie Coordinator Jobs — Workflow scheduler
- Oozie Bundle — Defines data pipelines

Oozie Use Cases

- Used by Administrators to run complex log analysis on HDFS.
- Used by Developers for performing ETL operations on data in a sequential order and saving the output in a specified format (Avro, ORC, etc.) in HDFS.
- In an enterprise, Oozie jobs are scheduled as coordinators or bundles.

Oozie Workflow



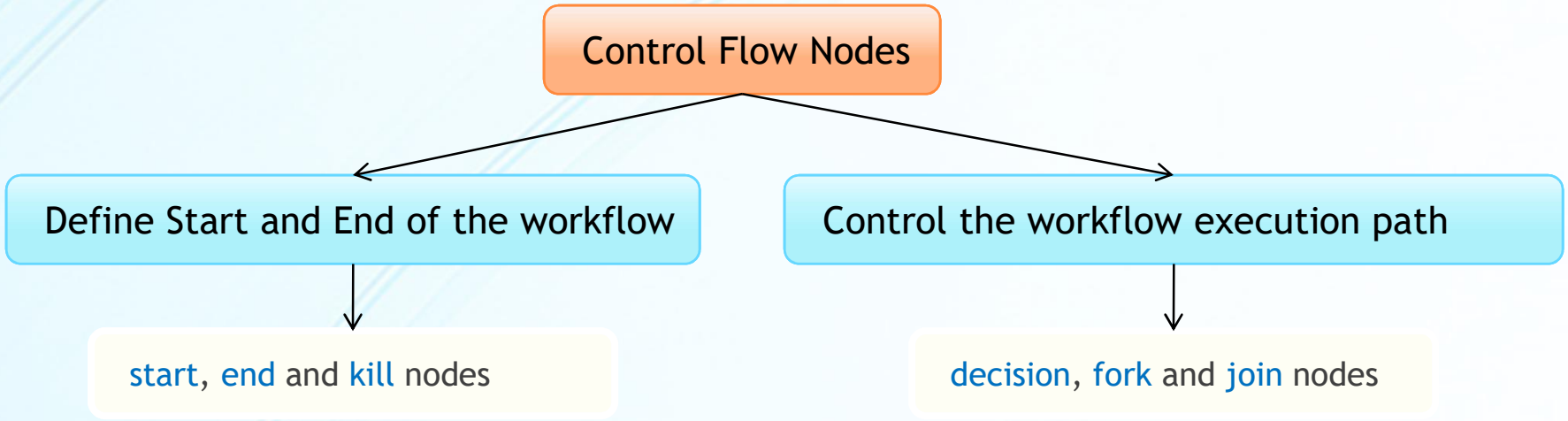
Workflow Example

Oozie workflow definition to run the maximum temperature MapReduce job

```
<workflow-app xmlns="uri:oozie:workflow:0.1" name="max-temp-workflow">
    <start to="max-temp-mr"/>
    <action name="max-temp-mr">
        <map-reduce>
            <job-tracker>${jobTracker}</job-tracker>
            <name-node>${nameNode}</name-node>
            <prepare>
                <delete path="${nameNode}/user/${wf:user()}/output"/>
            </prepare>
            <configuration>
                // MapReduce Properties go here ….
            </configuration>
        </map-reduce>
        <ok to="end"/>
        <error to="fail"/>
    </action>
    <kill name="fail">
        <message>MapReduce failed, error  message</message>
    </kill>
    <end name="end"/>
</workflow-app>
```

Oozie Workflow

Oozie workflows contain control flow nodes and action nodes.



- Represent logical decisions between action nodes
- Execute actions based on conditions or in parallel
- Workflows begin with START node
- Workflows succeed with END node and fails with KILL node
- Several actions support JSP Expression Language (EL)

Map Reduce Node

Action tag used to run a map-reduce process. You have to supply MR related configuration parameters such as Job-tracker, Task-tracker etc.

```
<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>
```

Java Node

Runs Java Jobs. Runs the main() method 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>
```

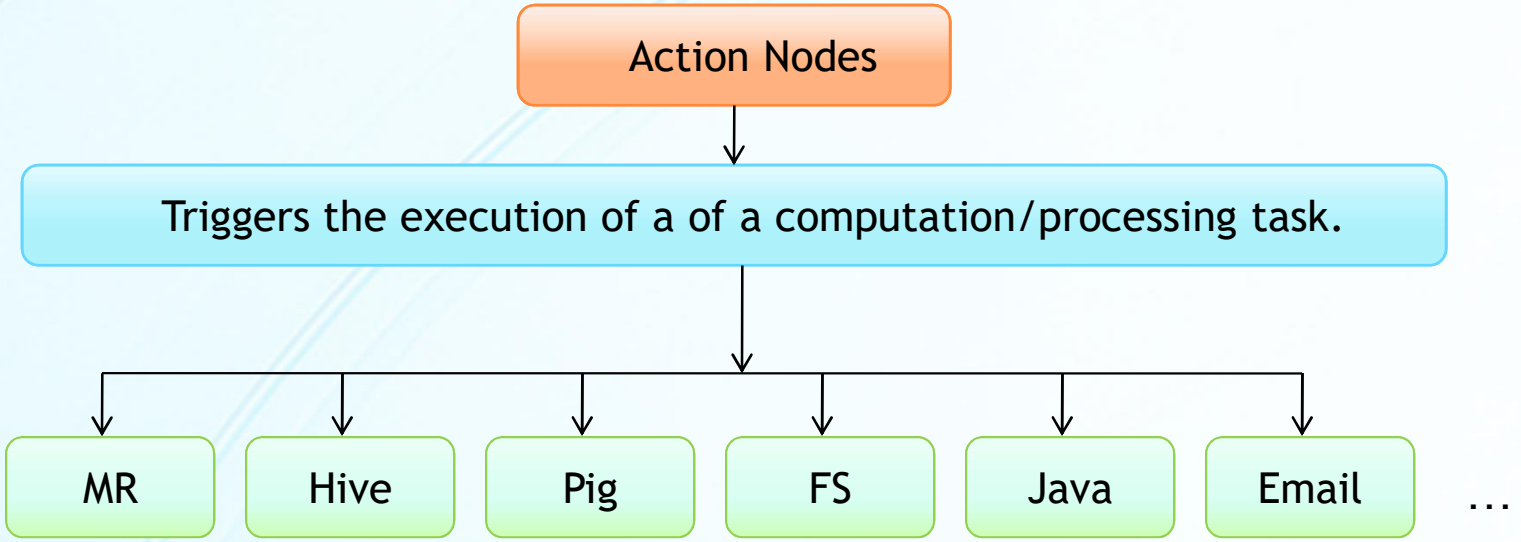
File System Node

Implements HDFS FS commands

```
<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 Workflow

Oozie workflows contain control flow nodes and action nodes.



Control Flow Nodes

Start Node

- Tells the application where to start
- `<start to= "[node-name]" />`

End Node

- Signals the end of Oozie Job
- `<end name= "[node-name]" />`

Kill Node

- The kill node allows a workflow job to kill itself.
-
```
<kill name= "[node-name]" >
   <message> [ message-to-log ]</message>
</error>
```

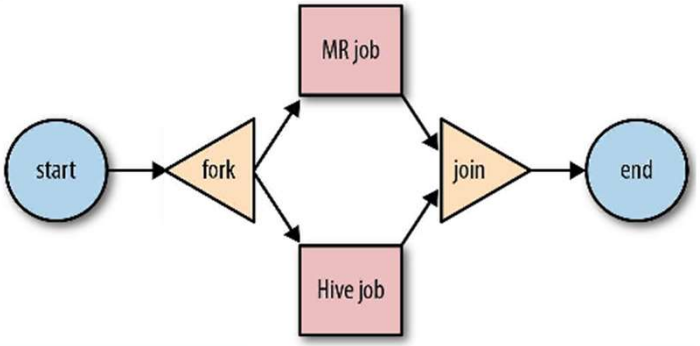
Fork & Join Nodes

Fork Node — Using **forks** we can run multiple jobs in parallel.

Join Node — Join forked 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.

```
<fork name="fork_node">
   <path start="MRJob"/>
   <path start="HiveJob"/>
</fork>
<action name="MRJob">
   . . . .
   <ok to="join_node" />
   . . . .
</action>
<action name="HiveJob">
   . . . .
   <ok to="join_node" />
   . . . .
</action>
<join name="join_node" to="Insert_into_Table"/>
```



Decision Nodes

We can add **decision** tags to check if we want to run an action based on the output of decision.

```
<decision name="external_table_exists">
  <switch>
    <case to="Create_External_Table">
       ${fs:exists('/test/abc') eq 'false'}
    </case>
    <default to="orc_table_exists" />
  </switch>
</decision>
```

Oozie Workflow

Workflow in Oozie is a sequence of actions arranged in a control dependency DAG.

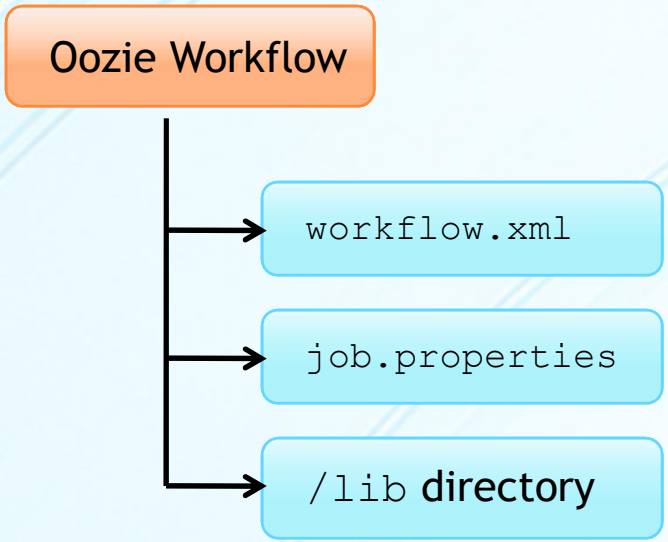
The actions are in controlled dependency as the next action can only run as per the output of current action.

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 can be parameterized. When submitting a workflow job, values for the parameters must be provided using **job.properties** file. If properly parameterized (i.e. using different output directories), several identical workflow jobs can concurrently.

Running Workflow



The file where you define your workflow XML

The file where you define values of parameter values and the location of your workflow.xml file

Put all your jar files here

- The **`job.properties`** should be a local file during submissions, and not in HDFS.
- The **`workflow.xml`** file and any script files need to be in HDFS.

Property Files

- Oozie workflows can be parameterized. The parameters come from a configuration file called as **property file**.
- We can run multiple jobs using same workflow by using multiple **.property** files (one property for each job).
- Suppose we want to change the script name or value of a param, we can specify those in the **property file** and pass it while running the workflow.

Running Workflow

1. Export OOZIE_URL environment variable

```
$export OOZIE_URL=http://localhost:11000/oozie
```

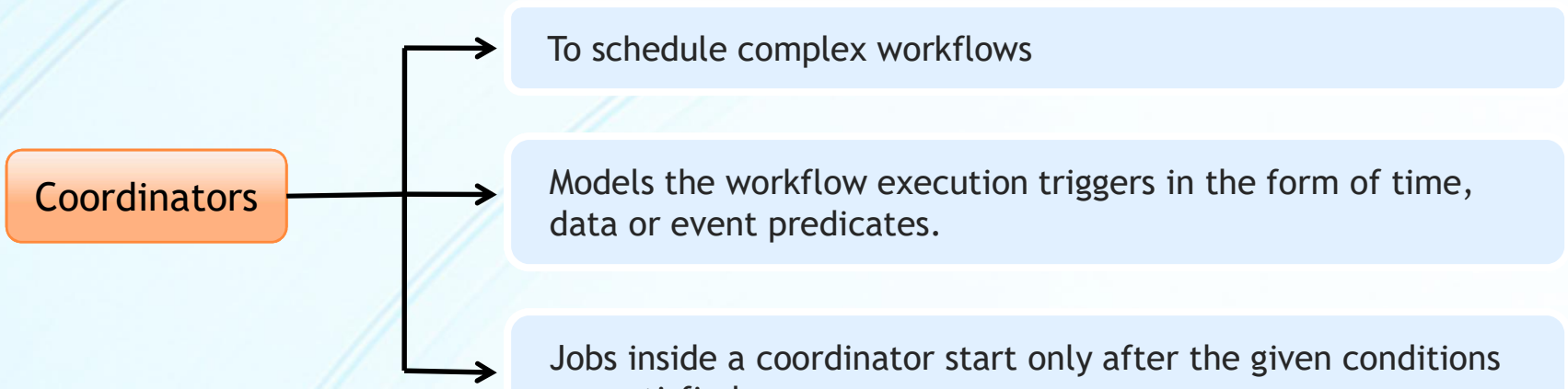
2. Run the job

```
$oozie job -config <config-file-path> -run
```

- Alternative Command

```
oozie job -oozie http://localhost:11000/oozie
          -config /path/job.properties
          -run
```

Oozie Coordinators



Oozie Coordinators

```
<coordinator-app
     xmlns="uri:oozie:coordinator:0.2"
     name="coord_copydata_from_external_orc"
     frequency="5 * * * *"
     start="2019-01-31T01:00Z"
     end="2025-12-31T00:00Z"
     timezone="America/Los_Angeles">

     <controls>
       <timeout>1</timeout>
       <concurrency>1</concurrency>
       <execution>FIFO</execution>
       <throttle>1</throttle>
     </controls>

     <action>
       <workflow>
       <app-path>pathof_workflow_xml/workflow.xml</app-path>
       </workflow>
     </action>

</coordinator-app>
```

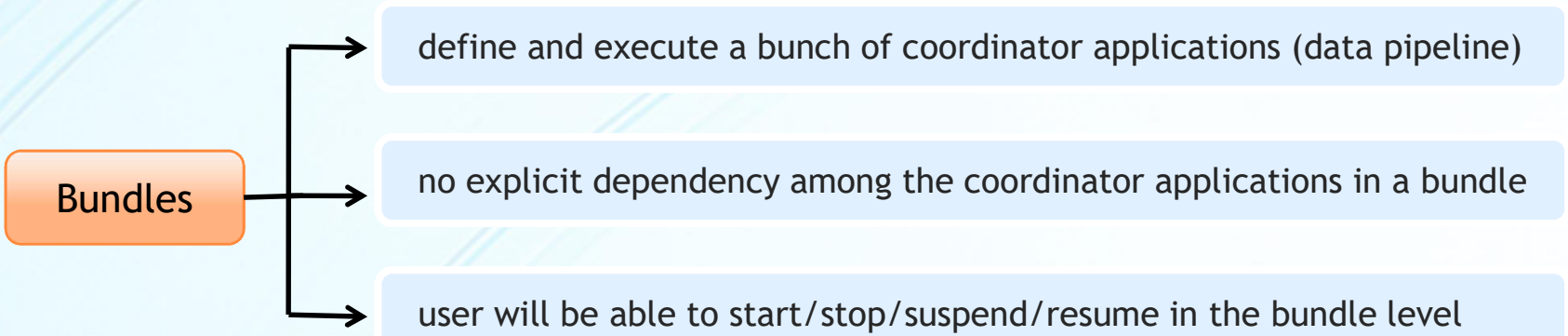
Frequency to materialize actions

Time to start materializing the action

Time to stop actions being materialized

Frequency: 0,10 15 * * 2-6 --> a job that runs every weekday at 3:00 pm and 3:10 pm
1: minutes, 2:hours, 3:days, 4:months, 5:weekdays

Oozie Bundles



Oozie Bundles

```
<bundle-app
   xmlns='uri:oozie:bundle:0.1'
   name='bundle_copydata_from_external_orc'>

   <controls>
     <kick-off-time>${kickOffTime}</kick-off-time>
   </controls>

   <coordinator name='coord_copydata_from_external_orc' >
      <app-path>path_of_coordinator_xml</app-path>
      <configuration>
         <property>
           <name>startTime1</name>
           <value>time to start</value>
         </property>
      </configuration>
   </coordinator>

</bundle-app>
```

THANK YOU