北风网 IBEIFENG.COM

大数据Hadoop高薪直通车课程

工作流调度框架Oozie

讲师：轩宇（北风网版权所有)

课程大纲

1. Hadoop 调度框架
2. Oozie 功能架构
3. Oozie 安装部署
4. Oozie 工作流调度
5. Oozie 协作调度

课程大纲

1. Hadoop 调度框架
2. Oozie 功能架构
3. Oozie 安装部署
4. Oozie 工作流调度
5. Oozie 协作调度

Hadoop 调度框架

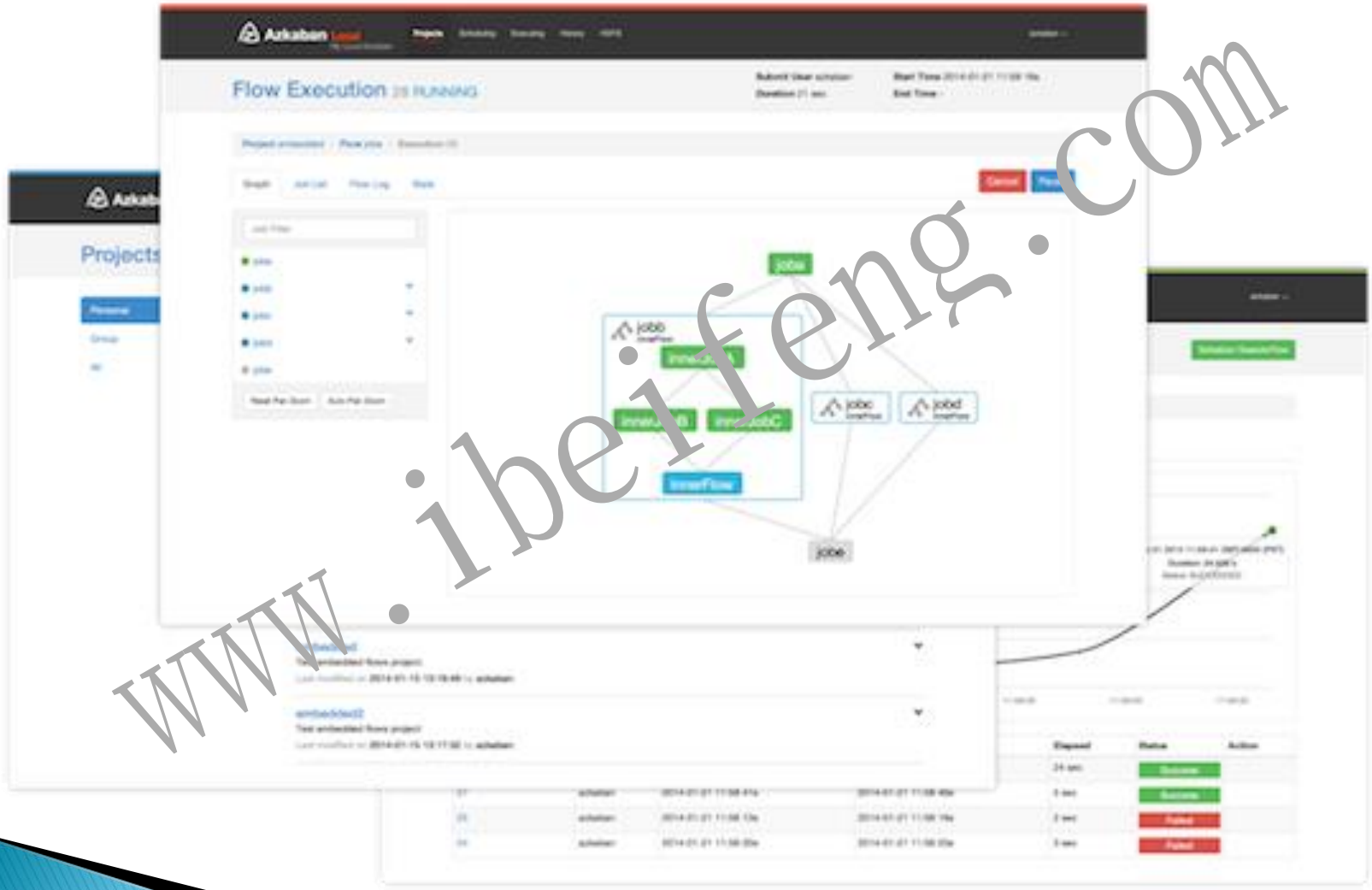
- Linux Crontab
- Azkaban

  https://azkaban.github.io/
- Oozie

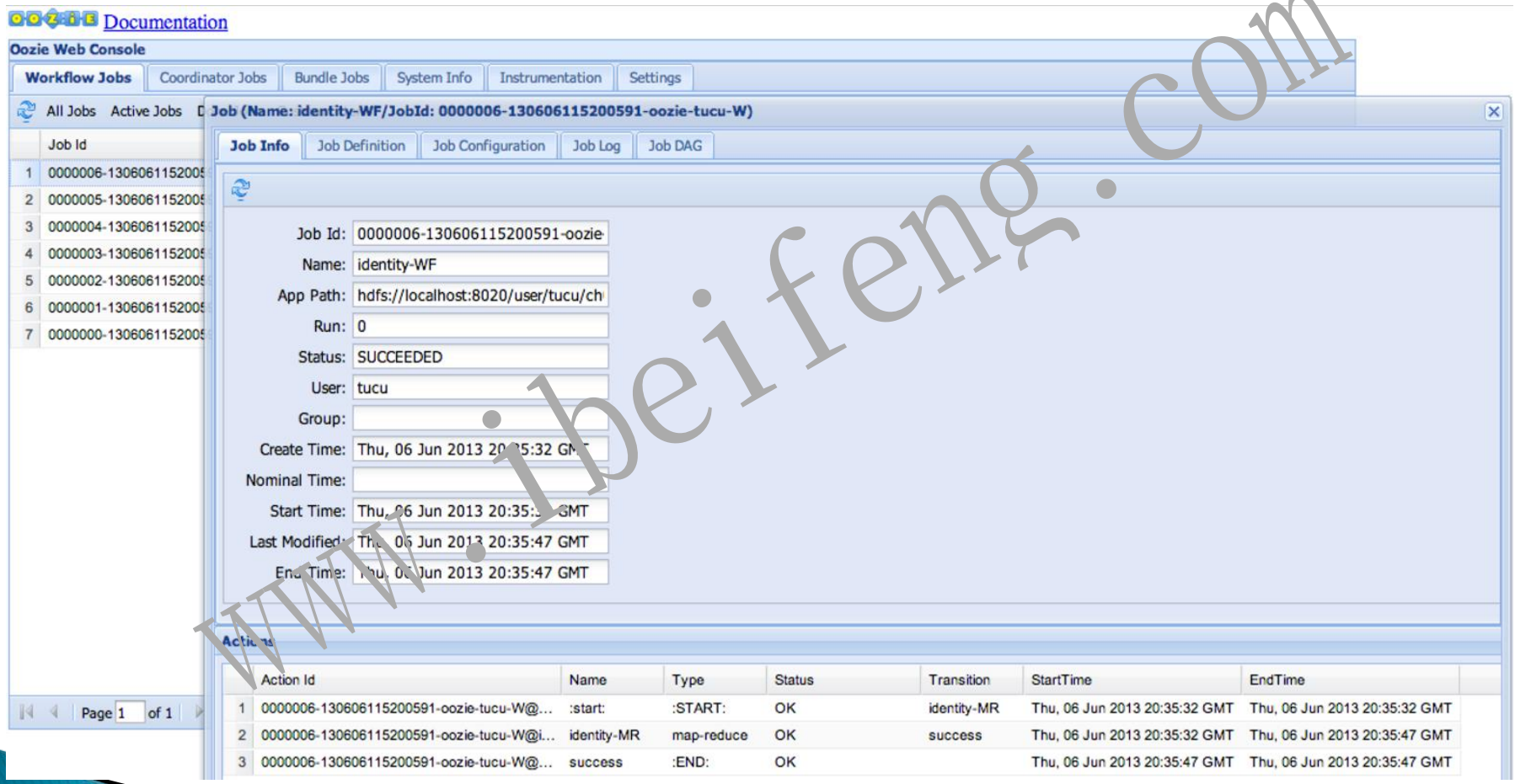
  http://oozie.apache.org/
- Zeus

  https://github.com/michael8335/zeus2

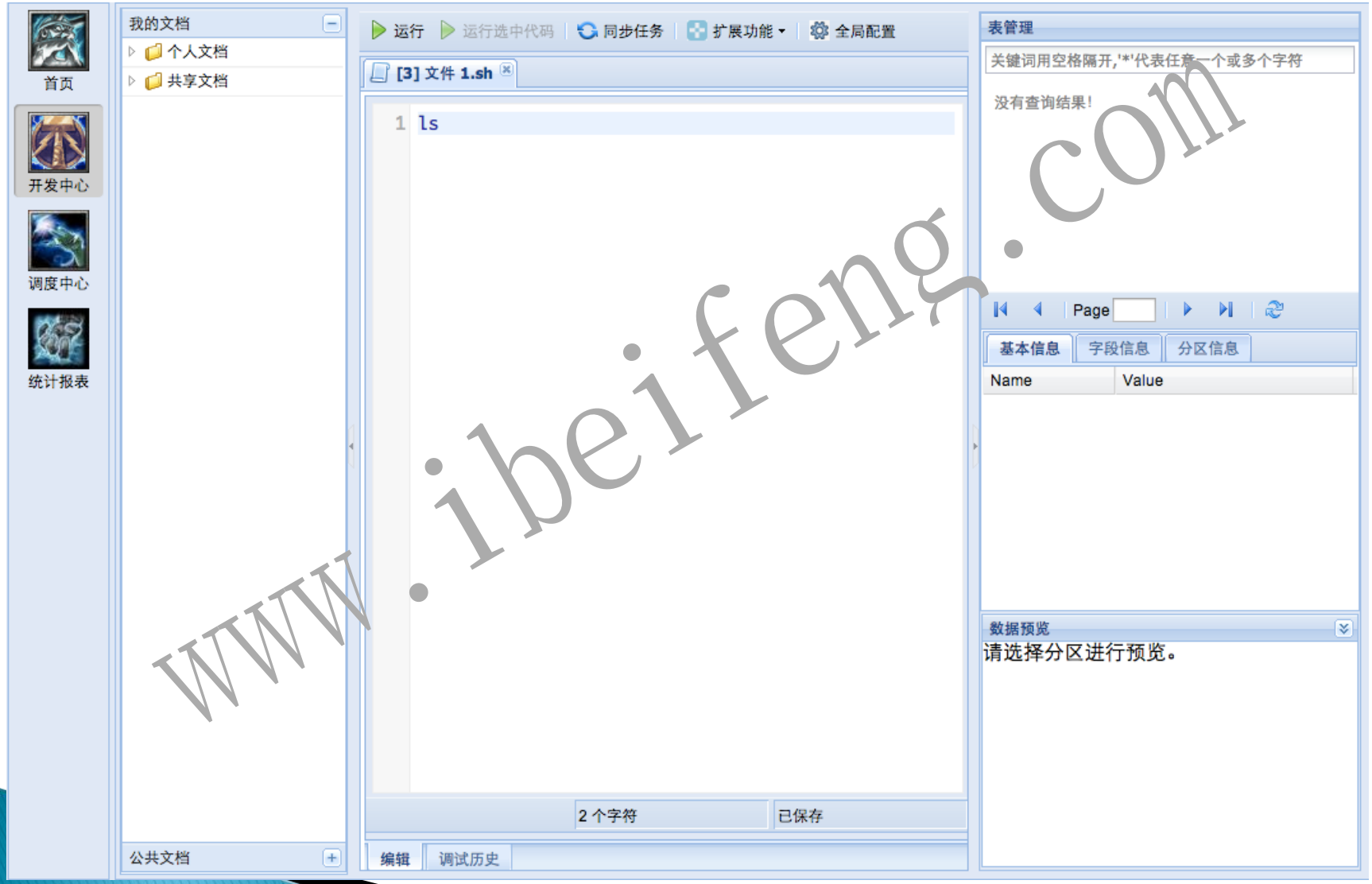
Azkaban, Open-source Workflow Manager



Oozie, Workflow Engine for Apache Hadoop



Zeus2, Hadoop job work flow schedule



课程大纲

1. Hadoop 调度框架
2. Oozie 功能架构
3. Oozie 安装部署
4. Oozie 工作流调度
5. Oozie 协作调度

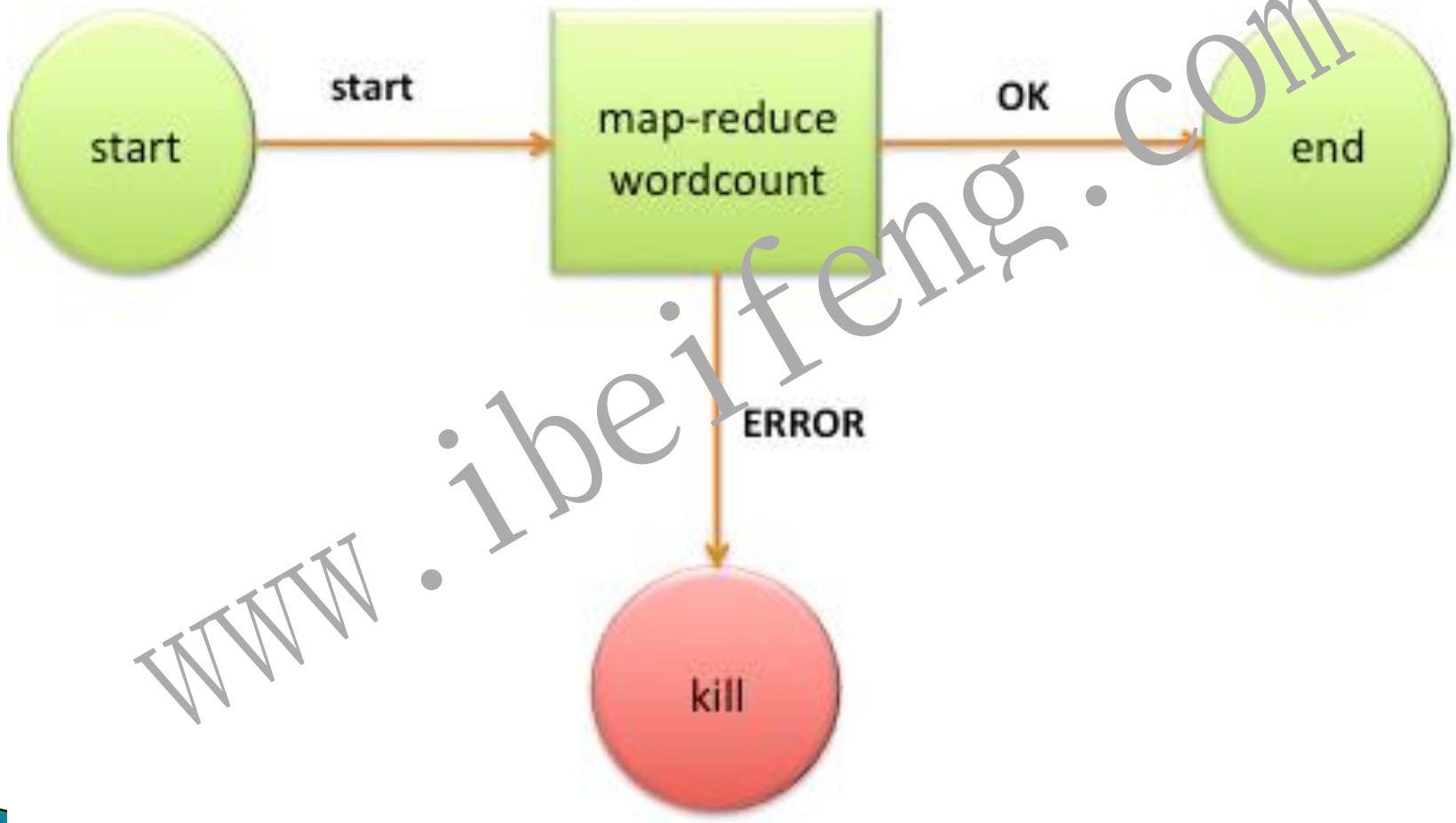
Apache Oozie Workflow Scheduler for Hadoop

- Oozie is **a workflow scheduler system** to manage Apache Hadoop jobs.
- Oozie Workflow jobs are **Directed Acyclical Graphs (DAGs)** of actions.
- Oozie Coordinator jobs **are recurrent Oozie Workflow jobs triggered** by time (frequency) and data availabilty.
- Oozie is integrated with the rest of the Hadoop stack supporting several types of Hadoop jobs out of the box (such **as Java map-reduce, Streaming map-reduce, Pig, Hive, Sqoop and Distcp**) as well as system specific jobs (**such as Java programs and shell scripts**).
- Oozie is a scalable, reliable and extensible system.

Oozie, Workflow Engine for Apache Hadoop

- 一个基于工作流引擎的开源框架，是由Cloudera公司贡献给Apache的，它能够提供对Hadoop MapReduce和Pig Jobs的任务调度与协调。Oozie需要部署到Java Servlet容器中运行。
- Oozie工作流定义，同JBoss jBPM提供的jPDL一样，也提供了类似的流程定义语言hPDL，通过XML文件格式来实现流程的定义。对于工作流系统，一般都会有很多不同功能的节点，比如分支、并发、汇合等等。
- Oozie定义了**控制流节点（Control Flow Nodes）和动作节点（Action Nodes）**，其中控制流节点定义了流程的开始和结束，以及控制流程的执行路径（Execution Path），如decision、fork、join等；而动作节点包括Hadoop map-reduce、Hadoop文件系统、Pig、SSH、HTTP、eMail和Oozie子流程。

WordCount Workflow Example



WordCount Workflow Example

```
<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='end' />
    </action>
    <kill name='kill'>
        <message>Something went wrong: ${wf:errorCode('wordcount')}</message>
    </kill/>
    <end name='end' />
</workflow-app>
```

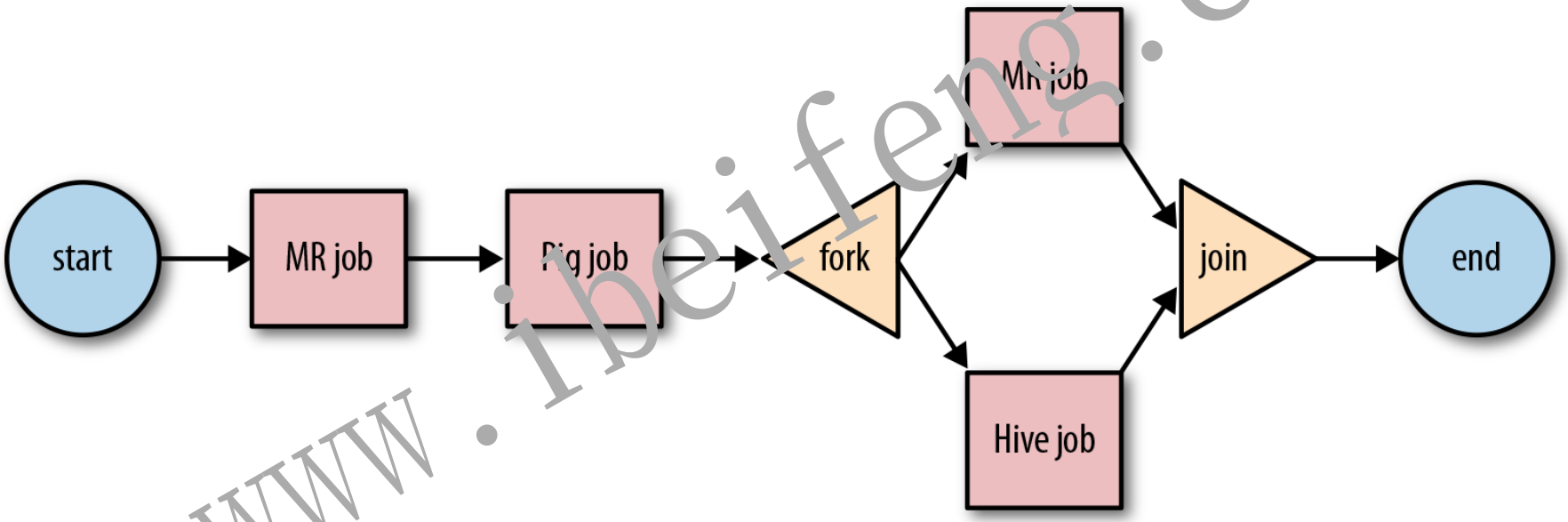
Oozie, Workflow Engine for Apache Hadoop

Oozie v3 is a server based *Bundle Engine* that provides a higher-level oozie abstraction that will batch a set of coordinator applications. The user will be able to start/stop/suspend/resume/rerun a set coordinator jobs in the bundle level resulting a better and easy operational control.

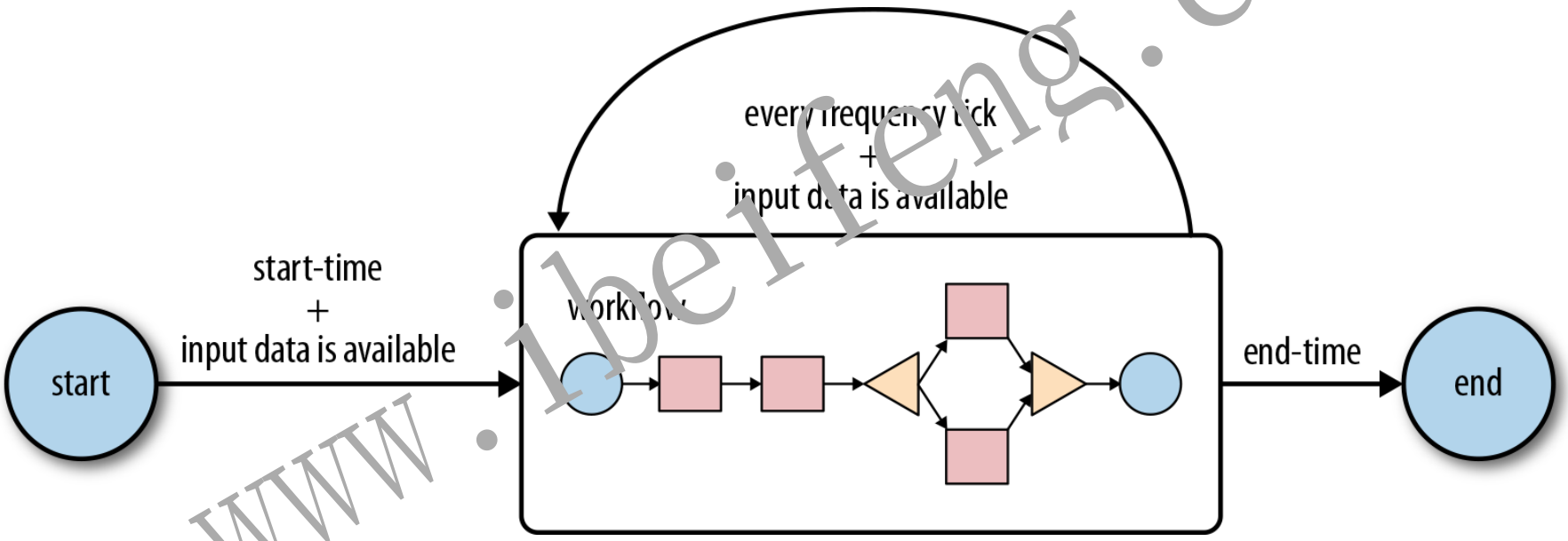
Oozie v2 is a server based *Coordinator Engine* specialized in running workflows based on time and data triggers. It can continuously run workflows based on time (e.g. run it every hour), and data availability (e.g. wait for my input data to exist before running my workflow).

Oozie v1 is a server based *Workflow Engine* specialized in running workflow jobs with actions that execute Hadoop Map/Reduce and Pig jobs.

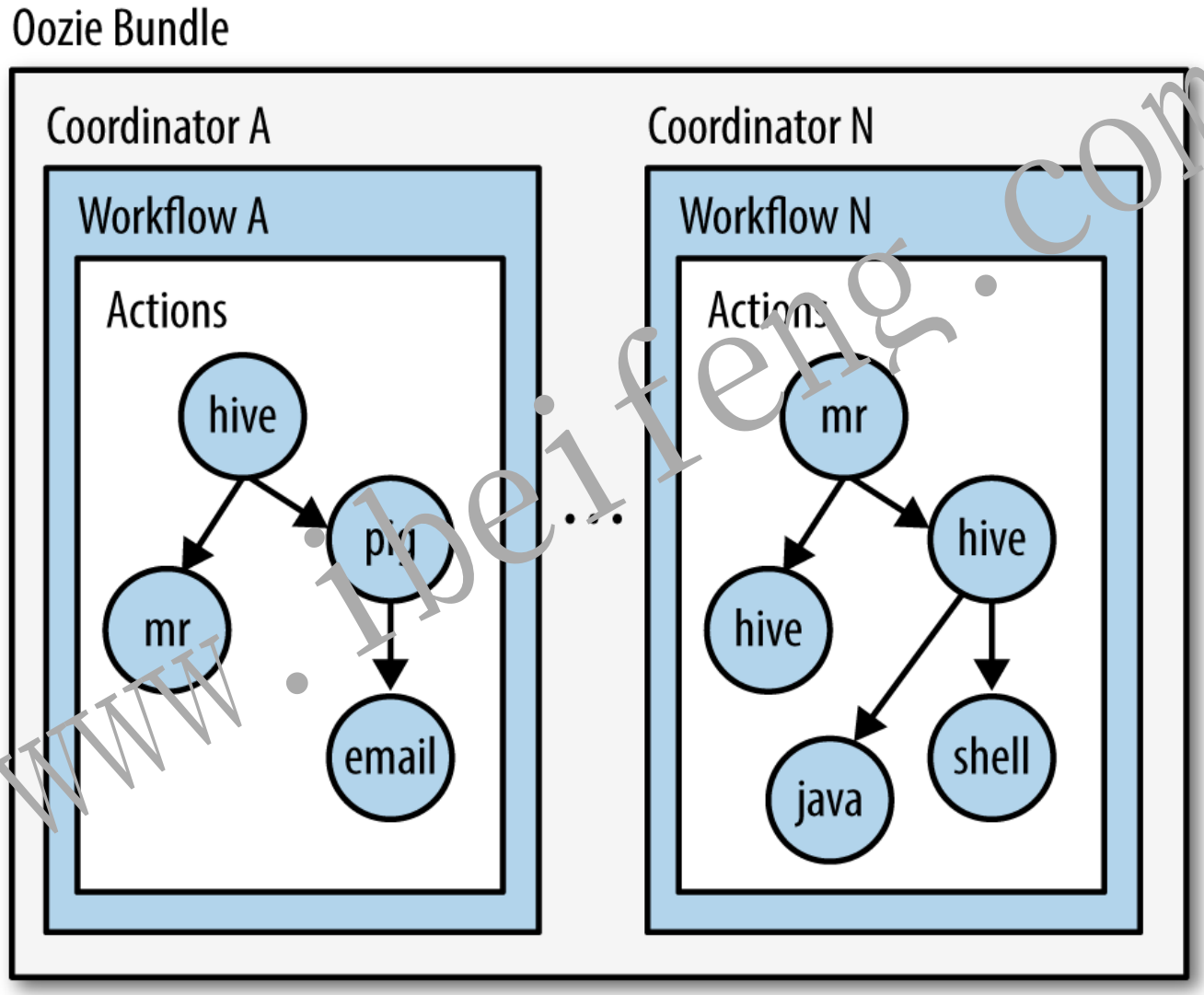
Oozie Workflow



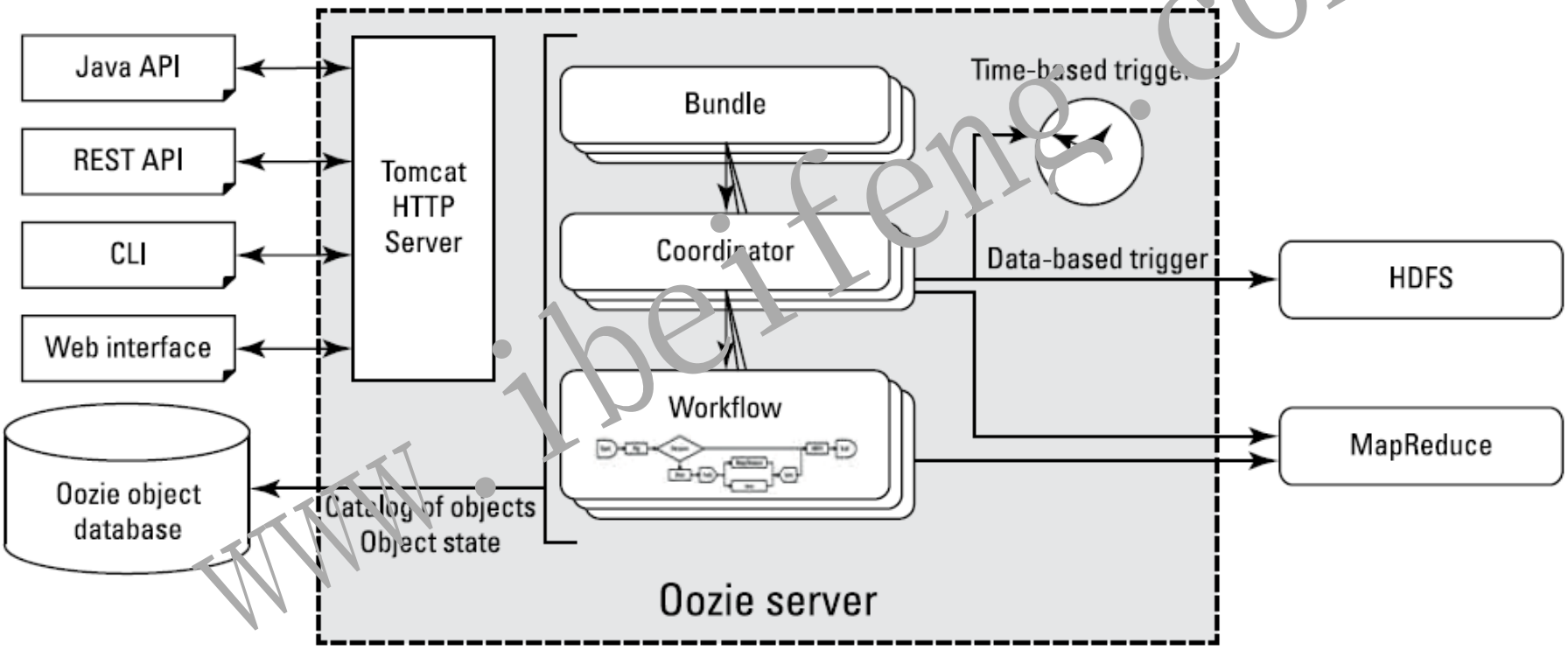
Lifecycle of an Oozie coordinator



Oozie Bundle



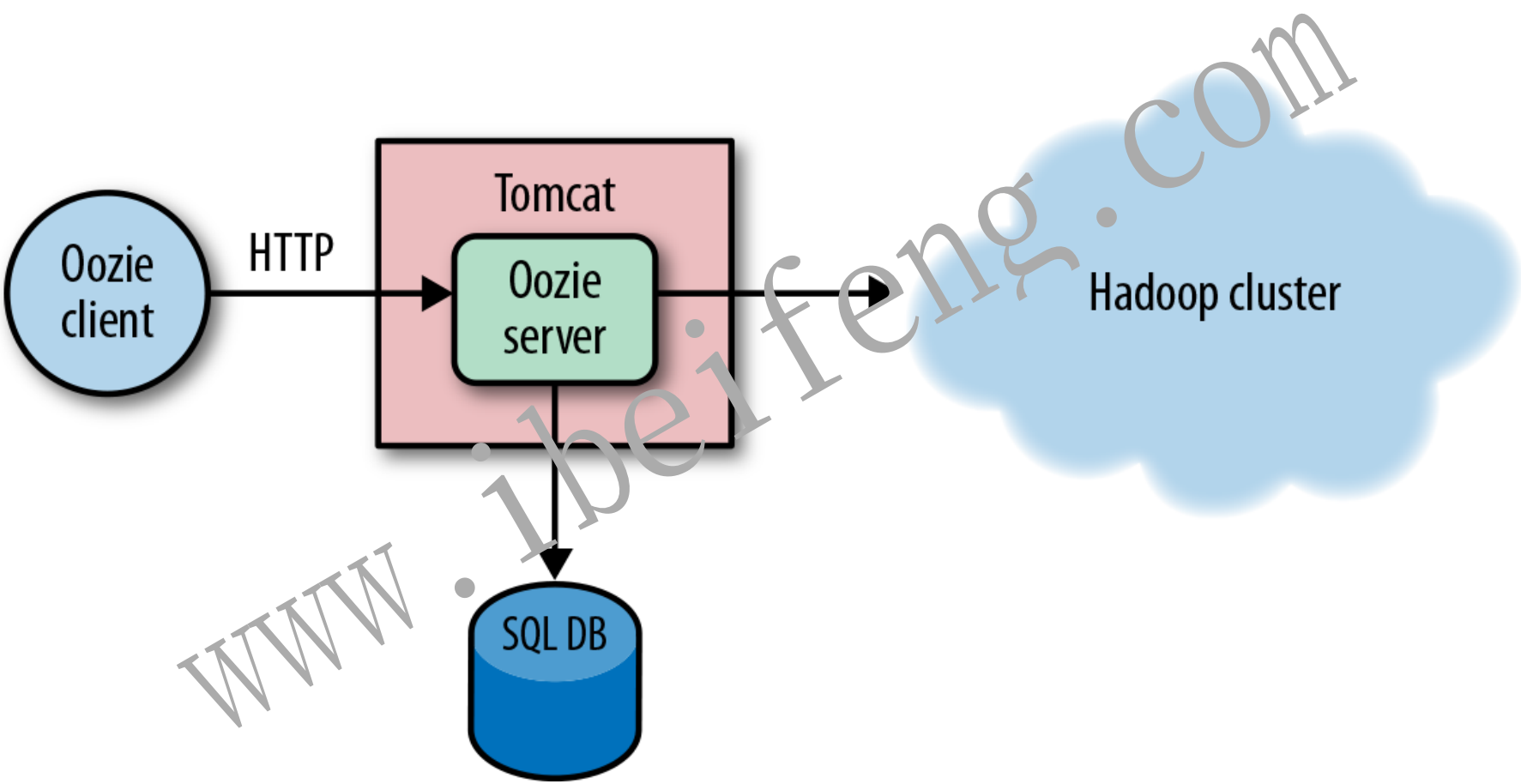
Oozie server components



课程大纲

1. Hadoop 调度框架
2. Oozie 功能架构
3. Oozie 安装部署
4. Oozie 工作流调度
5. Oozie 协作调度

Oozie Server Architecture



Building Oozie

http://archive.cloudera.com/cdh5/cdh/5/oozie-4.0.0-cdh5.3.6/DG_QuickStart.html

System Requirements:

- Unix box (tested on Mac OS X and Linux)
- Java JDK 1.6+
- Maven 3.0.1+
- Hadoop 0.20.2+
- Pig 0.7+

JDK commands (java, javac) must be in the command path.

The Maven command (mvn) must be in the command path.

Building Oozie

Download a source distribution of Oozie from the "Releases" drop down menu on the Oozie site

Expand the source distribution `tar.gz` and change directories into it.

The simplest way to build Oozie is to run the `mkdistro.sh` script:

```
$ bin/mkdistro.sh
```

If you'd like to skip all of the tests, which can take some time:

```
$ bin/mkdistro.sh -DskipTests
```

Running `mkdistro.sh` will create the binary distribution of Oozie.

IMPORTANT: By default it builds against Hadoop 1.1.1. It's possible to build against Hadoop 2.x versions as well, but it is strongly recommend to use a Bigtop distribution if using Hadoop 2.x because the Oozie sharelibs built from the tarball distribution will not work with it.

http://archive.cloudera.com/cdh5/cdh/5/oozie-4.0.0-cdh5.3.6/ENG_Building.html

http://segmentfault.com/a/1190000002738484

Distribution Contents

Oozie distribution consists of a single 'tar.gz' file containing:

- Readme, license, notice & Release log files.
- Oozie server: `oozie-server` directory.
- Scripts: `bin/` directory, client and server scripts.
- Binaries: `lib/` directory, client JAR files.
- Configuration: `conf/` server configuration directory.
- Archives:
  - `oozie-client-*.tar.gz` : Client tools.
  - `oozie.war` : Oozie WAR file.
  - `docs.zip` : Documentation.
  - `oozie-examples-*.tar.gz` : Examples.
  - `oozie-sharelib-*.tar.gz` : Share libraries (with Streaming, Pig JARs).

Server Installation

System Requirements

- Unix (tested in Linux and Mac OS X)
- Java 1.6+
- Hadoop
  - Apache Hadoop (tested with 1.0.0 & 0.23.1)
- ExtJS library (optional, to enable Oozie webconsole)
  - ExtJS 2.2

The Java 1.6+ `bin` directory should be in the command path.

http://archive.cloudera.com/cdh5/cdh/5/oozie-4.0.0-cdh5.3.6/DG_QuickStart.html

http://www.cnblogs.com/blackshirt/p/4447519.html

http://www.cnblogs.com/tovin/p/3885162.html

Server Installation

IMPORTANT: Oozie ignores any set value for `OOZIE_HOME` , Oozie computes its home automatically.

- Build an Oozie binary distribution
- Download a Hadoop binary distribution
- Download ExtJS library (it must be version 2.2)

NOTE: The ExtJS library is not bundled with Oozie because it uses a different license.

NOTE: It is recommended to use a Oozie Unix user for the Oozie server.

Expand the Oozie distribution `tar.gz` .

Expand the Hadoop distribution `tar.gz` (as the Oozie Unix user).

Server Installation

NOTE: Configure the Hadoop cluster with proxyuser for the Oozie process.

The following two properties are required in Hadoop core-site.xml:

```
<!-- OOZIE -->
<property>
  <name>hadoop.proxyuser.[OOZIE_SERVER_USER].hosts</name>
  <value>[OOZIE_SERVER_HOSTNAME]</value>
</property>
<property>
  <name>hadoop.proxyuser.[OOZIE_SERVER_USER].groups</name>
  <value>[USER_GROUPS_THAT_ALLOW_IMPERSONATION]</value>
</property>
```

Replace the capital letter sections with specific values and then restart Hadoop.

Server Installation

- Expand the **Oozie hadooplibs tar.gz** in the same location Oozie distribution tar.gz was expanded. A ***hadooplibs/*** directory will be created containing the Hadoop JARs for the versions of Hadoop that the Oozie distribution supports.
- Create a **libext/** directory in the directory where Oozie was expanded.
- If using a version of Hadoop bundled in Oozie **hadooplibs/** , copy the corresponding Hadoop JARs from**hadooplibs/** to the **libext/** directory.

```
cp oozie-4.0.0-cdh5.3.3/hadooplibs/hadooplib-2.5.0-cdh5.3.3.oozie-4.0.0-cdh5.3.3/* libext/
```

- If using the ExtJS library copy the ZIP file to the **libext/** directory.

Server Installation

Run the `oozie-setup.sh` script to configure Oozie with all the components added to the **libext/** directory.

```
$ bin/oozie-setup.sh prepare-war [-d directory] [-secure]
                     sharelib create -fs <FS_URI> [-locallib <PATH>]
                     sharelib upgrade -fs <FS_URI> [-locallib <PATH>]
                     db create|upgrade|postupgrade -run [-sqlfile <FILE>]
```

The `-secure` option will configure Oozie to use HTTP (SSL); refer to Setting Up Oozie with HTTPS (SSL) for more details.

Server Installation

- A **"sharelib create -fs fs_default_name [-locallib sharelib]"** command is available when running oozie-setup.sh for uploading new sharelib into hdfs where the first argument is the default fs name and the second argument is the Oozie sharelib to install, it can be a tarball or the expanded version of it. If the second argument is omitted, the Oozie sharelib tarball from the Oozie installation directory will be used.

```
bin/oozie-setup.sh sharelib create \
-fs hdfs://hadoop-ehp01.cloudyhadoop.com:8020 \
-locallib oozie-sharelib-4.0.0-cdh5.3.3-yarn.tar.gz
```

Server Installation

- "**prepare-war [-d directory]"** command is for creating war files for oozie with an optional alternative directory other than libext.

bin/oozie-setup.sh prepare-war

- **"db create|upgrade|postupgrade -run [-sqlfile] "**command is for create, upgrade or postupgrade oozie db with an optional sql file

bin/ooziedb.sh create -sqlfile oozie.sql -run DB Connection

Server Installation

Start Oozie as a daemon process run:

```
$ bin/oozied.sh start
```

To start Oozie as a foreground process run:

```
$ bin/oozied.sh run
```

Check the Oozie log file `logs/oozie.log` to ensure Oozie started properly.

Using the Oozie command line tool check the status of Oozie:

```
$ bin/oozie admin -oozie http://localhost:11000/oozie -status
```

Using a browser go to the Oozie web console , Oozie status should be **NORMAL** .

Oozie Examples

http://archive.cloudera.com/cdh5/cdh/5/oozie-4.0.0-cdh5.3.6/DG_Examples.html

Setting Up the Examples

Oozie examples are bundled within the Oozie distribution in the `oozie-examples.tar.gz` file.

Expanding this file will create an `examples/` directory in the local file system.

The `examples/` directory must be copied to the user HOME directory in HDFS:

```
$ hadoop fs -put examples examples
```

NOTE: If an examples directory already exists in HDFS, it must be deleted before copying it again. Otherwise files may not be copied.

Oozie Examples

Add Oozie `bin/` to the environment PATH.

The examples assume the JobTracker is `localhost:8021` and the NameNode is `hdfs://localhost:8020` . If the actual values are different, the job properties files in the examples directory must be edited to the correct values.

The example applications are under the examples/app directory, one directory per example. The directory contains the application XML file (workflow, or worklfow and coordinator), the `job.properties` file to submit the job and any JAR files the example may need.

The inputs for all examples are in the `examples/input-data/` directory.

The examples create output under the `examples/output-data/${EXAMPLE_NAME}` directory.

Note : The `job.properties` file needs to be a local file during submissions, and not a HDFS path.

Oozie Examples

- How to run an example application:

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

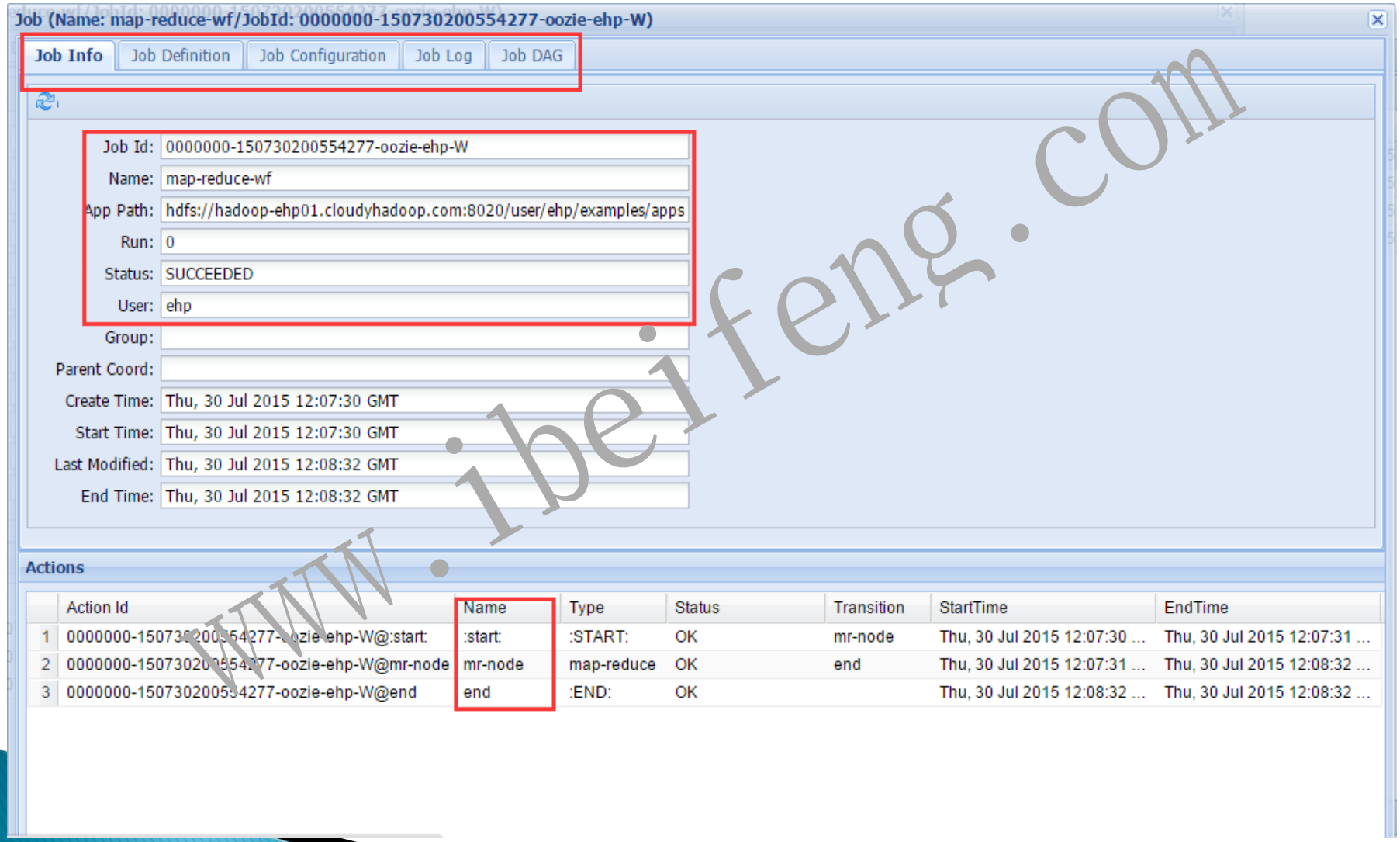
- Check the workflow job status

```
$ bin/oozie job -oozie http://localhost:11000/oozie \
-info 14-20090525161321-oozie-tucu
```

To avoid having to provide the `-oozie` option with the Oozie URL with every `oozie` command, set `OOZIE_URL` env variable to the Oozie URL in the shell environment. For example:

```
$ export OOZIE_URL="http://localhost:11000/oozie"
$
$ oozie job -info 14-20090525161321-oozie-tucu
```

Oozie Examples



Oozie Examples

Oozie Examples

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	VCores Used	VCores Total	VCores Reserved	Active Nodes	Decommissioned Nodes	Lost Nodes	Unhealthy Nodes	Rebooted Nodes
2	0	0	2	0	0 B	8 GB	0 B	0	8	0	1	0	0	0	0

User Metrics for dr.who

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Containers Pending	Containers Reserved	Memory Used	Memory Pending	Memory Reserved	VCores Used	VCores Pending	VCores Reserved
0	0	0	2	0	0	0	0 B	0 B	0 B	0	0	0

Show 20 entries Search:

ID	User	Name	Application Type	Queue	StartTime	FinishTime	State	FinalStatus	Progress	Tracking UI
application_1438257661380_0002	ehp	oozie:action:T=map-reduce:W=map-reduce-wf:A=mr-node:ID=0000000-150730200554277-oozie-ehp-W	APREDUCE	root.ehp	Thu Jul 30 20:07:58 +0800 2015	Thu Jul 30 20:08:29 +0800 2015	FINISHED	SUCCEEDED		History
application_1438257661380_0001	ehp	oozie:launcher:T=map-reduce:W=map-reduce-wf:A=mr-node:ID=0000000-150730200554277-oozie-ehp-W	APREDUCE	root.ehp	Thu Jul 30 20:07:33 +0800 2015	Thu Jul 30 20:07:59 +0800 2015	FINISHED	SUCCEEDED		History

Showing 1 to 2 of 2 entries First Previous 1 Next Last

课程大纲

1. Hadoop 调度框架
2. Oozie 功能架构
3. Oozie 安装部署
4. Oozie 工作流调度
5. Oozie 协作调度

Definitions

Action: An execution/computation task (Map-Reduce job, Pig job, a shell command). It can also be referred as task or 'action node'.

Workflow: A collection of actions 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.

Workflow Definition: A programmatic description of a workflow that can be executed.

Workflow Definition Language: The language used to define a Workflow Definition.

Workflow Job: An executable instance of a workflow definition.

Workflow Engine: A system that executes workflows jobs. It can also be referred as a DAG engine.

Workflow Definition

A workflow definition is a DAG with control flow nodes (start, end, decision, fork, join, kill) or action nodes (map-reduce, pig, etc.), nodes are connected by transitions arrows.

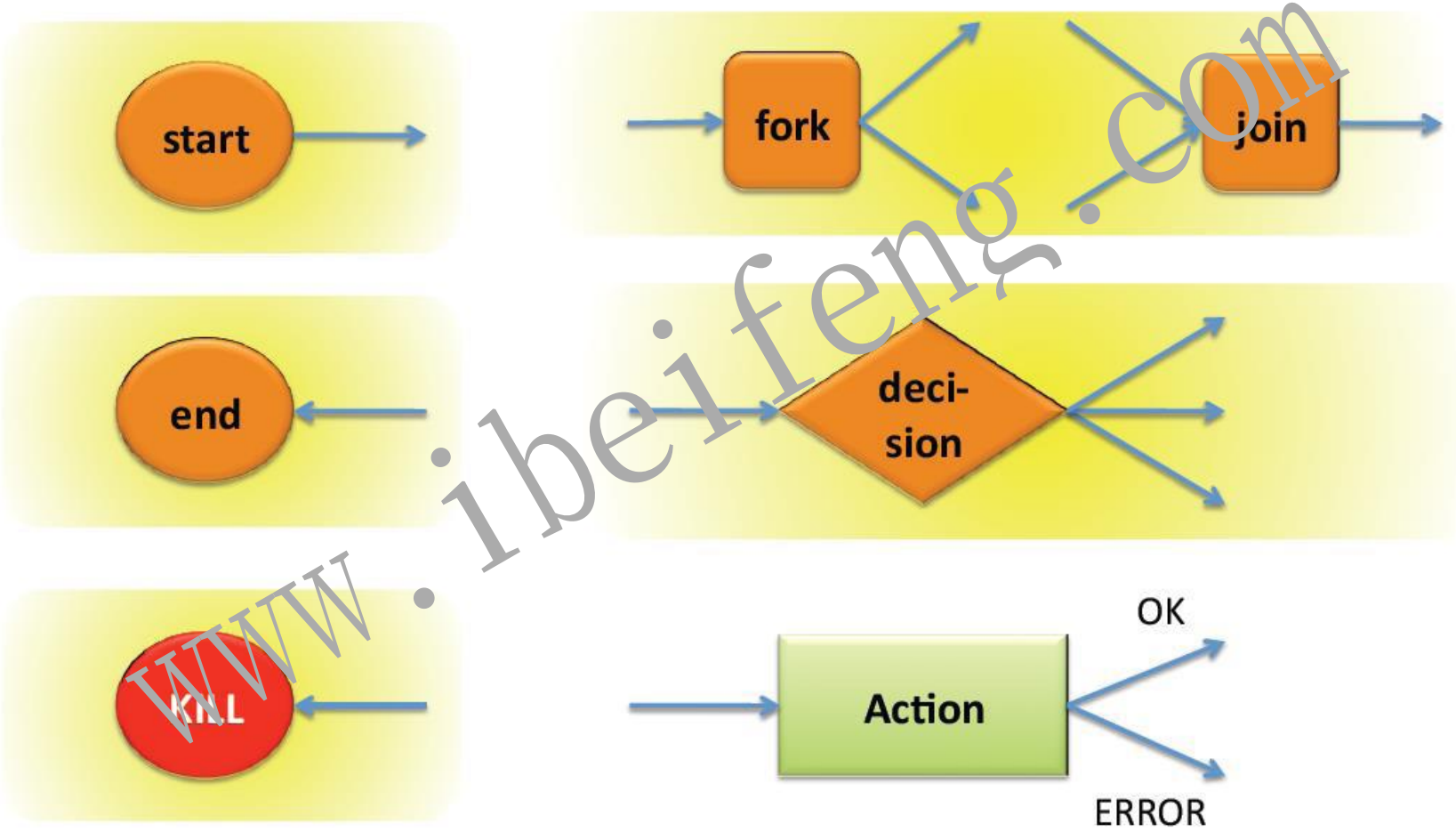
The workflow definition language is XML based and it is called hPDL (Hadoop Process Definition Language).

Workflow nodes are classified in control flow nodes and action nodes:

- **Control flow nodes:** nodes that control the start and end of the workflow and workflow job execution path.
- **Action nodes:** nodes that trigger the execution of a computation/processing task.

Node names and transitions must be conform to the following pattern =[a-zA-Z][\-_a-zA-Z0-0]*=, of up to 20 characters long.

Workflow Nodes



Start Control Node

The `start` node is the entry point for a workflow job, it indicates the first workflow node the workflow job must transition to.

When a workflow is started, it automatically transitions to the node specified in the `start` .

A workflow definition must have one `start` node.

Syntax:

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

The `to` attribute is the name of first workflow node to execute.

Example:

```
<workflow-app name="foo-wf" xmlns="uri:oozie:workflow:0.1">
    ...
    <start to="firstHadoopJob"/>
    ...
</workflow-app>
```

End Control Node

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

When a workflow job reaches the end it finishes successfully (SUCCEEDED).

If one or more actions started by the workflow job are executing when the end node is reached, the actions will be killed. In this scenario the workflow job is still considered as successfully run.

A workflow definition must have one end node.

Syntax:

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

The name attribute is the name of the transition to do to end the workflow job.

Example:

```
<workflow-app name="foo-wf" xmlns="uri:oozie:workflow:0.1">
    ...
    <end name="end"/>
</workflow-app>
```

Kill Control Node

The kill node allows a workflow job to kill itself.

When a workflow job reaches the kill it finishes in error (KILLED).

If one or more actions started by the workflow job are executing when the kill node is reached, the actions will be killed.

A workflow definition may have zero or more kill nodes.

Syntax:

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

The name attribute in the kill node is the name of the Kill action node.

The content of the message element will be logged as the kill reason for the workflow job.

A kill node does not have transition elements because it ends the workflow job, as KILLED .

Workflow Action Nodes

- Action Computation/Processing Is Always Remote
- Actions Are Asynchronous
- Actions Have 2 Transitions, =ok= and =error=
- Action Recovery

Map-Reduce Action

A map-reduce action can be configured to perform file system cleanup and directory creation before starting the map reduce job.

The workflow job will wait until the Hadoop map/reduce job completes before continuing to the next action in the workflow execution path.

The counters of the Hadoop job and job exit status (=FAILED=, KILLED or SUCCEEDED) must be available to the workflow job after the Hadoop jobs ends.

The map-reduce action has to be configured with all the necessary Hadoop JobConf properties to run the Hadoop map/reduce job.

```
<workflow-app name="foo-wf" xmlns="uri:oozie:workflow:0.1">
    ...
    <action name="myfirstHadoopJob">
        <map-reduce>
            <job-tracker>foo:8021</job-tracker>
            <name-node>bar:8020</name-node>
            <prepare>
                <delete path="hdfs://foo:8020/usr/tucu/output-data"/>
            </prepare>
            <job-xml>/myfirstjob.xml</job-xml>
            <configuration>
                <property>
                    <name>mapred.input.dir</name>
                    <value>/usr/tucu/input-data</value>
                </property>
                <property>
                    <name>mapred.output.dir</name>
                    <value>/usr/tucu/input-data</value>
                </property>
            </configuration>
        </map-reduce>
        <ok to="myNextAction"/>
        <error to="errorCleanup"/>
    </action>
    ...
</workflow-app>
```


http://blog.csdn.net/xiao_jun_0820/article/details/40375207

本课程版权归北风网所有

欢迎访问我们的官方网站

www.ibeifeng.com