bisdw
-
A simple ETL tool

Version 0.2.0
2013-10-06

Contents

1 Introduction

Bisdw is a simple ETL tool that is target to extract data from different source. It can use different embedded ways to do the extraction from different kind of databases, files, etc. In addition Bisdw support FTP based transfer of the result of the extraction process. Bisdw use the open source project Scriptellato enable ETL functionality, but can be extended in a simple way.

Bisdw runs as a daemon process. By configuration Bisdw will run different ETL jobs based on scheduling definition that are defined for each job.

2 Configuration

The main configuration file is the bidw.xml. Additional common properties are set in the properties.xml fle.

2.1 bisdw.xml configuration

Bisdw is configured by defining ETLs jobs in the in the bisdw.xml file.

```
<?xml version="1.0" encoding="UTF-8"?>                                  1
<bisdw xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"            2
   xsi:noNamespaceSchemaLocation="../src/main/resources/bisdw.
   xsd">
  <etljob>                                                              3
    <name>read_DB</name>                                                4
    <desc>Database</desc>                                               5
    <schedule>* */30 * * * ?</schedule>                                 6
                                                                        7
    <etlconfig name="read_db">                                          8
      <desc>read db</desc>                                              9
      <class>com.ingby.socbox.bisdw.etlprovider.ETLScriptella</         10
         class>
      <order>10</order>                                                 11
      <property>                                                        12
        <key>configFile</key>                                           13
        <value>read_db.xml</value>                                      14
      </property>                                                       15
  </etlconfig>                                                          16
                                                                        17
    <etlconfig name="send_ftp">                                         18
      <desc>send_ftp</desc>                                             19
      <class>com.ingby.socbox.bisdw.etlprovider.FTPSend</class>         20
      <order>20</order>                                                 21
      <property>                                                        22
        <key>hostname</key>                                             23
        <value>172.25.1.203</value>                                     24
      </property>                                                       25
```

```
      <property>                                          26
        <key>username</key>                               27
        <value>ftpuser</value>                            28
      </property>                                         29
      <property>                                          30
        <key>password</key>                               31
        <value>abcxyz</value>                             32
      </property>                                         33
      <property>                                          34
        <key>localDir</key>                               35
        <value>/tmp/shipment</value>                      36
      </property>                                         37
      <property>                                          38
        <key>remoteDir</key>                              39
        <value>ship</value>                               40
      </property>                                         41
    </etlconfig>                                          42
  </etljob>                                               43
</bisdw>                                                  44
```

Each <etljob> can include one to many <etlconfig>. A <etljob> will execute all etlconfig's part of the job according to the cron expression in the <schedule> tag, but in the sequence order specified in the >order> tag. The etlconfig's will be executed in the order of the lowest order first. This is important if there is a data dependencies between the different etlconfig's. By using the order tag a sequence of execution is quarantined.

Each etlconfig has a class specification that define the java class to be executed by the etljob. A list of properties can be specified that is specific by the specified class.

2.1.1 Scheduling

Each etljob can have multiple schedule tags, but at least one. The scheduling can have two different formats, interval or cron based.

2.1.1.1 Interval scheduling

The simple format describe a interval execution that are repeated forever. The format is just a number and a indicator defining the granularity in seconds (S), minutes (M) or hours (H). 10M specify that the service should be executed every ten minutes.

```
<schedule>10M</schedule>                                  1
```

2.1.1.2 Cron based scheduling

The second format is more advanced and follow the cron specification of Quartz, see `http://www.quartz-scheduler.org`. With this format its possible to define scheduling expression like *"0 15 10 ? * MON-FRI"* which would schedule the service at 10:15am every Monday, Tuesday, Wednesday, Thursday and Friday. For more cron examples please visit `http://www.quartz-scheduler.org/documentation/quartz-2.1.x/tutorials/crontrigger`

```
<schedule>0 15 10 ? * MON-FRI</schedule>
```

2.2 Etlconfig configuration

2.2.1 ETLScriptella configuration

The ETLScriptella takes one property from the etlconfig definition in bisdw.xml. The property is the *configFile* and specify the scriptella script file to read and execute. For more information about Scriptella configuration please visit Scriptella documentation.

Below listing show a simple example of a scriptella configuration that select from a mysql database and create an xml file based on the result and write it to directory */tmp/shipment*. Each file will be written to a unique file name using the *etl.date.today* macro.

```
<!DOCTYPE etl SYSTEM "http://scriptella.javaforge.com/dtd/etl.
   dtd">
<etl>
  <description>Example</description>
  <properties>
    filedepo=/tmp/shipment/
  </properties>
 <connection id="mydb" driver="com.mysql.jdbc.Driver" url="
    jdbc:mysql://localhost/bisdwtest" user="testdb" password="
    testdb" />

  <connection id="outfile" driver="text" url="$filedepo/
     shipment_${etl.date.today('yyyyMMdd-HHmmss')}.xml" >
    null_string=
  </connection>

  <!--Writing header -->
  <script connection-id="outfile">
    &lt;shipments&gt;
  </script>

```

```
  <!-- Run query against db -->                                    18
  <query connection-id="mydb">                                    19
    <!-- Select all shipments with pckp_dt TODAY and inbnd_ind     20
       = 0 from shipmentdb-->
    SELECT ship_id, no_pce, tot_wght FROM shipment;               21
                                                                  22
    <!-- For each row execute a script and write xml structure    23
       to out file -->
    <script connection-id="outfile">                              24
    <![CDATA[                                                     25
      <shipment>                                                  26
      <id>$ship_id</id>                                           27
      <weight>$tot_wght</weight>                                  28
      <pieces>$no_pce</pieces>                                    29
      </shipment>]]>                                              30
    </script>                                                     31
  </query>                                                        32
                                                                  33
  <!--Writing footer -->                                          34
  <script connection-id="outfile">                                35
    &lt;/shipments&gt;                                             36
  </script>                                                       37
</etl>                                                            38
```

2.2.2 FTPSend configuration

The FTPSend class enable transfer over FTP to a remote server. This can typical be used as part of the of a etljob where files are created by a scriptella script that is after creation is transfered to a FTP server.

The FTPSend configuration takes a number of properties to control the connection.

- *hostname* - the name or IP of the ftp server to connect to.
- *port* - the socket port used by the ftp server, default is 21.
- *timeout* - connection timeout, default is 2000 ms.
- *username* - username for the ftp server.
- *password* - password for the username
- *transferMode* - ascii or binary, default is ascii
- *connectionMode* - active or passive, default is passive.

- *remoteDir* - the directory on ftp server side, default is the current directory after login.
- *localDir* - the directory on the local side.
- *moveFileAfterSend* - move the files after transfer to the directory .save in the localDir default is true. If set to false the file is just deleted after successful transfer.

As and example see the listing in the 2.1 on page 4 that read any file located in the */tmp/-shipment* directory.

2.3 properties.xml

The properties.xml include properties used by the core of Bisdw. The properties xml has a simple structure of key/value pair:

```
<properties>                                    1
  <property>                                    2
    <key>akey</key>                             3
    <value>avalue</value>                       4
  </property>                                   5
</properties>                                   6
```

The following properties are currently used by core Bisdw:

- pidfile - the pid file for Bischeck, default is "*/var/tmp/bisdw.pid*".

3 Building Bisdw

To build Bisdw from source is simple. Check out the Bisdw trunk from gforge.ingby.com:

```
$ svn checkout --username anonymous http://gforge.ingby.com/svn
   /bischeck/bisdw/trunk bisdw
```

To build a Bisdw distribution run from the directory where you checked out the Bisdw code:

```
$ ant dist
```

This will create a compressed tar file in the *target* directory, named bisdw-x.y.z.tgz where x.y.z is the version number. Different versions of Bisdw can be checked out from the tags directory located in `http://gforge.ingby.com/svn/bischeck/bisdw/tags`

3.1 Jar customization

To support custom jar files please place them in the directory *customlib*. This would typical be jdbc drivers, etc.

3.2 Developing with Bisdw

Its simple to develop your own ETL config implementation. To develop your own you must follow the interface ETLInf.

4 Installation

The latest binary version of Bisdw is available on `http://gforge.ingby.com/gf/project/bischeck/frs/`.

Download the distribution file and follow the steps below to install. Make sure you have root privileges doing this.

```
# tar xzvf bisdw-x.y.z.tgz
# cd bisdw-x.y.z
# chmod 755 install
# ./install -u #Get usage
# ./install     #Install default
# service bisdw start     #Redhat/Centos
# /etc/init.d/bischeck start #Debian/Ubuntu
```

To get full list of available options to the install script use -u. By default the *install* script will install Bisdw in directory */opt/socbox/addons/bis*dw, referred to as *$BISDW* and with the ownership of the user id *bisdw*. Make sure that the user exist before running install.

The last commands start the bisdw daemon with the effective user id of the user id set during install, default user *bisdw*. The installation will configure *bisdwd* to start automatically in run level 3, 4 and 5.

The process id of the java process running bisdw in daemon mode is located in a file, default in */var/tmp/bisdw.pid*. This file is used by the *bisdwd* script to stop the java program running Bisdw and make sure that only one instance of Bisdw is started on the server.

4.1 Getting started

In the *$BISDW/etc* directory there are examples of all the configuration files.

4.2 Logging

Bisdw use log4j for log management. The log4j configuration is described in the log4.properties file located in the *resources* directory of the Bisdw installation. By default Bisdw writes log information at level INFO to file */var/tmp/bischeck.log*.

5 Run Bisdw from the command line

The normal way to run Bisdw is as a daemon using the init.d script *bisdwd*, but is also possible to start Bisdw in continues running mode by executing:

```
$ bisdw Execute -d
```

Running in this way have limitations since the execution will not automatically be placed as a background process and the effective user id will be the user starting the process which may not have all permissions according to the installation. Neither will pid files be updated correctly. For production system always use the init.d script.

```
$ sudo /etc/init.d/bisdwd start
```

or

```
# service bisdwd start
```

For testing purpose it can be good to just run Bisdw once and make sure that every thing is executing as expected. This is done by executing:

To show the pid file used for the Bischeck daemon running:

```
$ bischeck ConfigurationManager -p
```

This command is used in the init script bischeckd to retrieve the current pid.

5.1 Validating configuration files

To validate if the xml configuration files are correct the following command will return 0 if correct. Use $? to see return status.

```
$ bischeck ConfigurationManager -v; echo $?
```

6 Releases

6.1 Release 0.2.0 - 2013-10-06

This is the first major version.

7 System requirements

Bisdw should run on any operating system that supports Java 6. The installation script and init scripts are supported on Redhat and Debian equivalent Linux distributions. Running on none Linux operating system has not been tested.

The following jar packages are distributed as part of the Bisdw distribution. All these packages have their own open source licenses.

- scriptella-core.jar, scriptella-drivers.jar, scriptella.jar and scriptella-tools.jar version 1.1 `http://scriptella.javaforge.com/`
  - `http://www.apache.org/licenses/LICENSE-2.0.html`
- commons-net-3.3.jar `http://commons.apache.org/proper/commons-jexl/`
  - `http://www.apache.org/licenses/LICENSE-2.0.html`
- commons-jexl.jar `http://commons.apache.org/proper/commons-jexl/`
  - `http://www.apache.org/licenses/LICENSE-2.0.html`
- commons-logging.jar `http://commons.apache.org/proper/commons-logging/`
  - `http://www.apache.org/licenses/LICENSE-2.0.html`
- metrics-core.jar - `https://github.com/codahale/metrics`
  - `http://www.apache.org/licenses/LICENSE-2.0.html`
- log4j-1.2.16.jar - `http://logging.apache.org/log4j/`
  - `http://www.apache.org/licenses/LICENSE-2.0.html`
- commons-lang-2.5.jar - `http://commons.apache.org/lang/`
  - `http://www.apache.org/licenses/LICENSE-2.0.html`
- commons-cli-1.2.jar - `http://commons.apache.org/cli/`
  - `http://www.apache.org/licenses/LICENSE-2.0.html`
- slf4j-api-1.6.0.jar - `http://www.slf4j.org/`
  - `http://www.opensource.org/licenses/mit-license.php`

- slf4j-log4j12-1.6.0.jar - `http://www.slf4j.org/`
  - `http://www.opensource.org/licenses/mit-license.php`
- quartz-2.2.1.jar - `http://www.quartz-scheduler.org/`
  - `http://www.apache.org/licenses/LICENSE-2.0.html`

All files are distributed as part of Bisdw are located in the lib directory.

8 Bisdw license

Bisdw is licensed under GNU license version 2. For more info please visit `http://www.gnu.org/licenses/gpl-2.0.html`

9 Bug reports and feature requests

Please submit bug reports and feature requests on `gforge.ingby.com`

10 Credits

Thanks to all people who has developed all the great software that Bisdw depends on.