HITACHI
Inspire the Next

Deploying Custom Step Plugins for Pentaho MapReduce

This page intentionally left blank.

Contents

This page intentionally left blank.

Overview

Pentaho MapReduce (PMR) allows ETL developers to design and execute transformations that run in Hadoop MapReduce. If one or more custom steps are used in the mapper, reducer, or combiner transformations, additional configuration will be needed to make sure all dependencies are available for use in the Hadoop environment.

Our intended audience is Pentaho Data Integration (PDI) developers or administrators configuring PMR on a Hadoop cluster.

The intention of this document is to speak about topics generally; however, these are the specific versions covered here:

Software	Version(s)
Pentaho	6.x, 7.x, 8.x

The Components Reference in Pentaho Documentation has a complete list of supported software and hardware.

Before You Begin

This document assumes that you have knowledge of Pentaho and that you already have connectivity to a Hadoop cluster. More information about related topics outside of this document can be found in the Pentaho MapReduce documentation.

Pentaho MapReduce Configuration

The big data plugin adds PMR capabilities to PDI. Before PMR can perform PDI transformations in MapReduce, many code libraries must be copied to the Hadoop cluster. When a PMR job is invoked, PDI checks to see if the folder exists on the cluster and copies libraries if the folder does not exist.

You can find details on these topics in the following sections:

Plugin Properties Defined

These properties in the file `$PENTAHO_HOME/data-integration/plugins/pentaho-big-data-plugin/plugin.properties` define what is transferred to HDFS when the folder does not exist:

Table 1: Plugin Properties and Descriptions

Property	Description
`pmr.kettle.installation.id`	The installation ID, allowing multiple versions or configurations of PDI to be used with the same cluster. This defaults to the version of PDI being used when left blank.
`pmr.kettle.dfs.install.dir`	The directory in the Hadoop filesystem where the Pentaho libraries should be rooted. This defaults to `/opt/pentaho/mapreduce`.
`pmr.libraries.archive.file`	A zip file containing core Kettle components and required libraries for executing PDI from a Hadoop map, combine, or reduce context.
`pmr.kettle.additional.plugins`	A comma-separated list of folders located in `$PENTAHO_HOME/data-integration/plugins` to copy to the install directory.

The Copy Process

PDI checks to see if the `${pmr.kettle.dfs.install.dir}/${pmr.kettle.installation.id}` folder exists on the Hadoop cluster, whenever a PMR job is executed. If the folder does not exist, it is created on the destination file system, and the contents of the `${pmr.libraries.archive.file}` zip file are unzipped to that location.

Also, any of the traditional plugin directories listed in `${pmr.kettle.additional.plugins}` are copied to `${pmr.kettle.dfs.install.dir}/${pmr.kettle.installation.id}/plugins/`.

Configuration Changes

If changes are made to the `plugin.properties` file, a user must manually remove the contents of `${pmr.kettle.dfs.install.dir}/${pmr.kettle.installation.id}` on the Hadoop file system for the changes to take effect. With the default options, this can be achieved by running:

```
hadoop fs -rm -R -skipTrash /opt/pentaho/mapreduce/*
```

The next time a PMR job is performed, the folder will not be found, and the **copy** process will place the libraries in the Hadoop file system as configured.

Distributing Custom OSGi Plugins to PMR

An OSGi-based plugin system was added in the Pentaho 6.0 release. Plugins can be deployed and managed with Apache Karaf in their own container, instead of the legacy class loader plugin architecture. There are two methods for making sure any OSGi plugins are available to Kettle running in a MapReduce context. With either method, any changes will need to be re-applied in the event of a patch or minor/major update of Pentaho's product.

Manual Copy

The first method can be done without external tools, but it has the drawback of not being persisted if the copy process is repeated after a configuration change.

Plugins are usually installed to Karaf by placing a bundled JAR file in the `$PENTAHO_HOME/data-integration/system/karaf/deploy` folder. Some plugin installations may require JAR file replacements or xml feature file edits in the `system/karaf/system` folder. The copy process will create the `/opt/pentaho/mapreduce/<version>/` folder in the Hadoop file system, which has a `system/karaf` folder under it. The bundled OSGi JAR dependencies can be added by modifying `system/karaf` in the Hadoop file system with the same process as used for the install on the client.

Performing the following command would add a custom OSGi bundle for use by PMR transformation tasks if you use `pmr.kettle.dfs.install.dir = /opt/pentaho/mapreduce` and `pmr.kettle.installation.id = 7.1`:

```
hadoop fs -put cust-plugin-bundle.jar
/opt/pentaho/mapreduce/7.1/system/karaf/deploy
```

Modify the Archive File

This method requires a zip archive manipulation tool, but will automatically be re-applied if the copy process is repeated. As mentioned earlier, the `pmr.libraries.archive.file` is extracted to the Hadoop file system. The plugin installation will be deployed during that extraction process when a PDI MapReduce job runs, by manipulating the archive file.

1. Make a backup of the archive file, which should be located at `$PENTAHO_HOME/data-integration/plugins/pentaho-big-data-plugin/pentaho-mapreduce-libraries.zip`.
2. Open the original file in your zip file manipulation tool, and apply the changes as required by the installation of the OSGi plugin to the archive. There is a `system/karaf` folder, which contains a deploy folder for bundle installation, or system folder for modifying and replacing existing resources.
3. Execute the process as described in the Configuration Changes section of the document to deploy the changes to the server.
4. Run a Pentaho MapReduce job utilizing the plugin to confirm the installation was successful.

Related Information

Here are some links to information that you may find helpful while using this best practices document:

- Components Reference
- Pentaho MapReduce documentation

Finalization Checklist

This checklist is designed to be added to any implemented project that uses this collection of best practices, to verify that all items have been considered and reviews have been performed. (Compose specific questions about the topics in the document and put them in the table.)

Name of the Project:_____

Date of the Review:_____

Name of the Reviewer:_____

Item	Response	Comments
Did you add the plugin folder name to the `pmr.kettle.additional.plugins` file and invoke a configuration update, for legacy plugins?	YES_____ NO_____	
Did you use the manual copy process method for OSGi plugins?	YES_____ NO_____	
If yes, did you modify all required files for the plugin installation under the `system/karaf` folder?	YES_____ NO_____	
If no, did you make a backup of the `pentaho-mapreduce-libraries.zip` file archive before applying any changes?	YES_____ NO_____	
If yes, did you open the new archive and confirm the changes were made according to the install?	YES_____ NO_____	
If yes, did you force the configuration update and run a job to invoke the plugin deployment?	YES_____ NO_____	