Plugin based Farming

JMS clustering in Geronimo | Clustering and farming | Tomcat Native Clustering

This farming system uses plugins directly and is thus decoupled from the deployment system. It enables the plugin based farming administration for different servers or machines. Farming information is stored in a database using jps. The data model allows a single administration server to manage:

- multiple farms, also termed clusters
- multiple lists of plugins per farm in an M X N relationship
- multiple plugins per plugin list

When a node starts up, it broadcasts a service advertisement including an optional node name and a farm name. When the administration server detects this as new, it instructs the new node to install all the plugin lists associated with that farm. If the plugins are already installed, the system provides no further operation; otherwise the plugins are downloaded and installed from the plugin repository specified in the plugin list.

Farming administrative operations include:

- Add a plugin list to a farm
- Add a plugin to a plugin list
- Add a plugin to a plugin list and the list to a farm
- Remove a plugin list from a farm
- Remove a plugin from a plugin list

These operations are available through the GShell cluster/deploy command and are discussed in details in the #Add and remove plugins and plugin lists section.

Architectural Considerations

Plugin based farming requires two separate sets of functionality:

- A plugin farm controller that tracks the farms node, plugin lists, and plugins and issues instructions to the nodes.
- A geronimo plugin repository accessible to all farm nodes. This can be a geronimo server running the geronimo-as-maven servlet, the local file system maven repository, a file based repository served by httpd, or a maven repository manager such as nexus. If a geronimo server is used, it can be the same server as the plugin farm controller or a separate server.

The farming information is stored in the PluginFarmDatatabase database on the farm controller.

Supporting plugins and assemblies

- `org.apache.geronimo.configs/plugin-farm-member//car` (plugins/clustering/plugin-farm-member) plugin turns a server into a plugin farm node. You can configure the multicast address and port, cluster name, and node name in `<WASCE_HOME>/var/config/config-substitutions.properties`.
- `org.apache.geronimo.assemblies/geronimo-plugin-farm-node//car` (plugins/clustering/geronimo-plugin-farm-node) extends the framework assembly with the plugin-farm-member plugin. This is all you need to start up a farm node.
- `org.apache.geronimo.configs/plugin-farm//car` (plugins/clustering/plugin-farm) is the administration plugin for plugin-based farms. Currently it is only accessible through the Gshell cluster/deploy command.
- `org.apache.geronimo.configs/plugin-farm-datasource//car` (plugins/clustering/plugin-farm-datasource) is a derby datasource used by the plugin farm. It is intended to be replaced in production with a remote datasource. As always, the replacement plugin should use an <artifact-alias> element to redirect dependencies on plugin-farm-datasource to itself.

Make sure that the version of the plugins comply with your server.

Setup steps

The following steps demonstrates the plugin based farming using nodes that all share the same geronimo installation. The admin server acts as the farm controller, application deployment server, and plugin repository. The setup steps for plugin based farming includes:

1. #Set up farm controller
2. #Set up farm node
3. #Deploy a sample plugin to the farm
4. #Add and remove plugins and plugin lists

Set up farm controller

1. Start the controller server and install the following plugins in the server repository. For example, for a version 2.2 server, the plugins are stored in the http ://geronimo/apache/org/plugins/geronimo-2.2 repository:

- `org.apache.geronimo.configs/plugin-farm//car`
- `org.apache.geronimo.configs/plugin-farm-datasource//car`
- `org.apache.geronimo.configs/plugin-farm-member//car`

2. Open the `config-substitutions.properties` file and set the following attributes:

Excerpt from config-substitutions.properties

ClusterName=*farm name*
NodeName=*name*
ServerHostname=*Node_IP*
DefaultPluginRepository=*share install plugin repository*

where

- `ClusterName`: defines the name of the farm that the server works in.
- `NodeName`: is optional and defaults to `host:jndi-port`. For example, if the host IP for this server is 9.186.10.157 and the jndi-port is 1099, the default `Nodename=9.186.10.157:1099`.
- `ServerHostname`:sets the node IP of this server.
- `DefaultPluginRepository`: defines the plugin repository accessible to all farm nodes. For example, you can use the maven repository as the share install plugin repository by defining `DefaultPluginRepository=http\://9.186.10.157\:8080/plugin/maven-repo`.

3. Restart the server.

Set up farm node

Use the following steps to set up every node server in the farm:
1. Start the node server and install the `org.apache.geronimo.configs/plugin-farm-member//car` plugin in the server repository. For example, for a version 2.2 server, the plugin is stored in the http://geronimo/apache/org/plugins/geronimo-2.2 repository.
2. Open the `config-substitutions.properties` file and set the following attributes:

Excerpt from config-substitutions.properties

ClusterName=*farm name*
ServerHostname=*Node_IP*

3. Add the share install plugin repository to the `plugin-repositories.properties` file under `<WASCE_HOME>/var/config/` for each farm node, including access username and password. For example:

Excerpt from plugin-repositories.properties

http\://9.186.10.157\:8080/plugin/maven-repo/=system\=manager

where

- the password `manger` in this example is in plain text. However, it is recommended to encrypt your password with the deploy encrypt command.

4. Restart the server.

Deploy a sample plugin to the farm

Install a plugin to the farm controller with command. For example, use the Gshell command as follows to install a HelloWorld.car plugin:

```
deploy/install-plugin D:/HelloWorld.car -s 9.186.10.157
```

- `-s`: is a command option that indicates the hostname of the server. For this example, the hostname is 9.186.10.157.

Add and remove plugins and plugin lists

Use the cluster/deploy Gshell command to verify the farm that you set up. For example, add and remove plugins and plugin lists with the following commands:
Use this command to add a plugin list p1 to farm cluster1:

```
cluster/deploy add -c cluster1 -l p1 -s 9.186.10.157
```

Use this command to add the `org.apache.geronimo.samples/HelloWorld/1.0/car` plugin to plugin list p1:

```
cluster/deploy add -l p1 -a org.apache.geronimo.samples/HelloWorld/1.0/car -s 9.186.10.157
```

Use this command to add the `org.apache.geronimo.samples/HelloWorld/1.0/car` plugin to plugin list p1 and add p1 to farm cluster1:

```
cluster/deploy add -c cluster1 -l p1 -a org.apache.geronimo.samples/HelloWorld/1.0/car -s 9.186.10.157
```

Use this command to remove a plugin list p1 from farm cluster1:

```
cluster/deploy remove -c cluster1 -l p1
```

Use this command to remove the `org.apache.geronimo.samples/HelloWorld/1.0/car` plugin from plugin list p1:

```
cluster/deploy remove -l p1 -a org.apache.geronimo.samples/HelloWorld/1.0/car -s 9.186.10.157
```

Access the **Plugin** portlet form the **Administration console** to verify the installation result for all farm members.

Future work

The following features could be added in the future:

- Provide **Administration console** support for the administrative operations.
- Enhance the monitoring console to find nodes using multicast discovery.
- Provide better support for setting up and starting multiple servers on one geronimo installation.
- Enhance the car-maven-plugin to assemble servers with multiple instance support in the assembly.