Chapter 9 - About the Islandora Rule Engine

Islandora's Rule Engine is a Python-based module that enables Islandora to listen for and respond to Fedora Repository updates, based on the Content Model associated with an object. The Fedora Microservices Module is available for download at https://github.com/Islandora/islandora_microservices. Once you have downloaded the module, you will need to do the following.

Dependencies

This script relies on Python 2.6 or greater. This script uses the lxml, fcrepo, stompy and yapsy python libraries. These should be downloaded and installed automatically if you run setup.py script:

```
$ python setup.py install
```

Note that the lxml libraries have caused problems with automatic installation on some operating systems.

Ubuntu:

```
apt-get install python-lxml
```

CentOS:

```
yum install python26

yum install python26-lxml
```

If you encounter an error in Cent during installation, you may also need these packages: gcc libxml2 libxml2-devel libxslt libxslt-devel

OSX:

See http://stackoverflow.com/questions/1512530/cant-install-lxml-python-2-6-3-osx-10-6-snow-leopard for Mac OS X 10.6 Snow Leopard installation.

Fedora Setup

1. To use this package with Fedora's built-in ActiveMQ broker you will need to download the Spring distribution from http://www.springsource.org/download. Then, copy the .jar files from the download package's dist directory into $FEDORA_HOME/tomcat/webapps /fedora/WEB-INF/lib to enable Stomp support in Fedora's embedded ActiveMQ broker.

2. Edit your Fedora instance's configuration to add a Stomp broker. Edit $FEDORA_HOME /server/config/fedora.fcfg and edit the messaging section

```
<param name="java.naming.provider.url" value="vm:(broker:(tcp://localhost:61616,
stomp://localhost:61613))"/>
```

Adding the Stomp URL will send messages to both the ActiveMQ port on 61616 and the Stomp port on 61613.

3. Restart Fedora and it should be ready to go.

This package contains a script islandora_listener.py and it also contains an init script islandora_listener. The init script is written for debian based systems but should work with minor modifications on other systems.

4. Run islandora_listener.py with the following command-line parameters:

Usage: islandora_listener.py [options]

Options:

-h, --help show this help message and exit

-C CONFIGFILE, --config-file=CONFIGFILE

Path of the configuration file for this listener

process instance.

-P PLUGINPATH, --plugin-path=PLUGINPATH

Path to a directory with plugin files.

5. Copy the file islandora_microservices.cfg.default to islandora_microservices.cfg, and edit it with your configuration options. The sample config file contains comments that should explain all the options.

```
$ python fedora_listener \-C /path/to/fedora_microservices.cfg
```

This will tell the script to connect to the Stomp port on the server specified in the configuration file and listen for messages on the topics /topic/fedora.apim.update and /topic /islandora. The script parses the recieved messages, and based on the options in the plugin config files calls the required plugins.

The plugins specified in the islandora_microservices.cfg file will be the only ones that run in this instance of the listener. The plugins in turn will only listen to the content models and messages specified in their configuration files. This allows you to have one instance of the script on your ABBYY FineReader server that only does OCR tasks and another instance of the script running on a different server that can do tasks like generating thumbnails, compressed JP2 images, etc. or any other task that doesn't need to be run on a specific server.

Plugins

Extend the functionality of the content model listener by adding a new plugin. The sample_plugin plugin gives a simple template that can be extended. It's important that every plugin has a cfg file or the plugin manager won't recognise it. The sample_plugin file gives an example and there is more documentation about the base class that plugins must extent in the plugin_manger.py file.